GigaDevice Semiconductor Inc.

GD32A711,GD32A712
Arm® Cortex®-M7 32-bit MCU

User Manual

Revision 1.2

(Sep. 2026)

Table of Contents

Table of Contents **2**
List of Figures **22**
List of Tables **28**
1. System and memory architecture **32**
1.1. Arm® Cortex®-M7 processor **32**
1.2. System architecture **33**
1.2.1. Interacting with other modules 37
1.3. Memory map **37**
1.3.1. On-chip SRAM memory 42
1.4. Boot configuration **44**
1.4.1. Image vector table(IVT) and Boot configuration word (BCW) 45
1.4.2. Nonsecure boot 46
1.4.3. Host recovery mode 48
1.4.4. Advance LC 49
1.4.5. HSM Identifier 50
1.5. System configuration controller (SYSCFG) **54**
1.6. Bus interconnect matrix (BUSIM) **54**
1.6.1. Overview 54
1.6.2. Arbitration algorithm 56
1.7. Extended memory protection unit (EXTMPU) **57**
1.7.1. Overview 57
1.7.2. Key features: 57
1.7.3. Function overview 57
1.8. Peripheral register protection unit (PRPU) **58**
1.8.1. Overview 58
1.8.2. Characteristics 58
1.8.3. Lock and unlock operation 59
1.9. Interconnect matrix integrity check unit (IMICU) **62**
1.9.1. Overview 62
1.9.2. Characteristics 62
1.9.3. Function overview 62
1.10. System configuration registers definition **66**
1.10.1. Peripheral mode configuration register (SYSCFG_PMCFG) 67
1.10.2. EXTI sources selection register 0 (SYSCFG_EXTISS0) 67
1.10.3. EXTI sources selection register 1 (SYSCFG_EXTISS1) 69
1.10.4. EXTI sources selection register 2 (SYSCFG_EXTISS2) 71
1.10.5. EXTI sources selection register 3 (SYSCFG_EXTISS3) 73
1.10.6. System status register (SYSCFG_STAT) 74
1.10.7. System configuration register (SYSCFG_CFG) 75
1.10.8. Timer break input select register (SYSCFG_TIMERBKINSEL) 76
1.10.9. CPU0 ICACHE error status register(SYSCFG_CPU0ICAC) 77
1.10.10. CPU0 DCACHE error status register (SYSCFG_CPU0DCAC) 78
1.10.11. CPU control register (SYSCFG_CPUCTL) 79
1.10.12. CPU0 control and status register (SYSCFG_CPU0CS) 79
1.10.13. SRAM configuration register (SYSCFG_SRAMCFG) 81
1.10.14. Recovery mode configuration register (SYSCFG_RMCFG) 82
1.10.15. TIMERx configuration register 0 (SYSCFG_TIMERxCFG0) (x = 0, 7, 19, 20, 60, 61, 62, 63) 83
1.10.16. TIMERx configuration register 1 (SYSCFG_TIMERxCFG1) (x = 0, 7, 19, 20, 60, 61, 62, 63) 86
1.10.17. TIMERx configuration register 2 (SYSCFG_TIMERxCFG2) (x = 0, 7, 19, 20, 60, 61, 62, 63)

88
1.10.18. TIMERx configuration register 0 (SYSCFG_TIMERxCFG0) (x = 1, 2) 89
1.10.19. TIMERx configuration register 1 (SYSCFG_TIMERxCFG1) (x = 1, 2) 91
1.10.20. TIMERx configuration register 2 (SYSCFG_TIMERxCFG2) (x = 1, 2) 93
1.10.21. Core processor 0 ID register (SYSCFG_CP0ID) 94
1.10.22. Core processor 0 number register (SYSCFG_CP0NUM) 95
1.10.23. Core processor 0 configuration information register 0 (SYSCFG_CP0CFGINF0) 95
1.10.24. Core processor 0 configuration information register 1 (SYSCFG_CP0CFGINF1) 96
1.10.25. Core processor 0 configuration information register 2 (SYSCFG_CP0CFGINF2) 97
1.10.26. Memory access execution control register(SYSCFG_MEMAEXCTL) 97
1.10.27. Bus matrix 1 interconnect error detection enable register (SYSCFG_BM1EDCEN) 99
1.10.28. Bus matrix 2 interconnect error detection enable register (SYSCFG_BM2EDCEN) 101
1.10.29. Bus check configuration register (SYSCFG_BUSCHKCFG) 103
1.10.30. TMR fault control and status register (SYSCFG_TMRFCS) 106
1.11. BUSIM registers definition 108
1.11.1. BM1 control register x (BUSIM_BM1CTLx) (x = 0,1,3,4,) 108
1.11.2. BM1 slave port priority configuration register x (BUSIM_BM1SPCFGx) (x = 0,1,3,4) 110
1.11.3. BM2 control register x (BUSIM_BM2CTLx) (x = 0...3) 112
1.11.4. BM2 slave port priority configuration register x (BUSIM_BM2SPCFGx) (x = 0...3) 113
1.11.5. TMR fault control and status register (BUSIM_TMRFCS) 116
1.12. EXTMPU registers definition 118
1.12.1. Control and status register 0 (EXTMPU_CS0) 118
1.12.2. Control and status register 1 (EXTMPU_CS1) 119
1.12.3. Error address register x (EXTMPU_EADDRx) (x = 0,2...4,10...12,14) 119
1.12.4. Error information register x (EXTMPU_EINFx) (x = 0,2...4,10...12,14) 119
1.12.5. Error region register x (EXTMPU_EREGx) (x = 0,2...4,10...12,14) 120
1.12.6. Region x control register 0 (EXTMPU_RxCTL0) (x = 0...31) 120
1.12.7. Region x control register 1 (EXTMPU_RxCTL1) (x = 0...31) 121
1.12.8. Region x control register 2 (EXTMPU_RxCTL2) (x = 0...31) 121
1.12.9. Region x control register 3 (EXTMPU_RxCTL3) (x = 0...31) 123
1.13. PRPU registers definition 124
1.13.1. Lock protection register x (PRPU_LKPROTx) (x = 0...104) 124
1.14. IMICU registers definition 125
1.14.1. Error detection control register (IMICU_EDCTL) 125
1.14.2. Error injection control register (IMICU_EICTL) 127
1.14.3. Error status register (IMICU_ERRSTAT) 128
1.15. SRAM registers definition 130
1.15.1. SRAM interrupt enable register (SRAM_INTEN) 130
1.15.2. SRAM error status register (SRAM_ERR) 131
1.15.3. SRAM correctable bit error address 0 for SRAM register (SRAM_CBEADDR0) 132
1.15.4. SRAM correctable bit error address 1 for SRAM register (SRAM_CBEADDR1) 132
1.15.5. SRAM correctable bit error address 2 for SRAM register (SRAM_CBEADDR2) 133
1.15.6. SRAM correctable bit error address 3 for SRAM register (SRAM_CBEADDR3) 133
1.15.7. SRAM uncorrectable bit error address for SRAM register (SRAM_UBEADDR) 134
1.15.8. SRAM buffer valid register (SRAM_BUFVLD) 134
1.15.9. SRAM overflow status register (SRAM_OVF) 135
1.15.10. SRAM write protection register (SRAM_WP) 135
1.15.11. SRAM ECC code error status register (SRAM_ECCCERR) 136
1.16. TCM registers definition 137
1.16.1. TCM interrupt enable register (TCM_INTEN) 137
1.16.2. TCM error status register (TCM_ERR) 138
1.16.3. TCM correctable bit error address 0 for SRAM_m register (TCM_CBEADDR0_m) (m = 0...2) 140
1.16.4. TCM correctable bit error address 1 for SRAM_m register (TCM_CBEADDR1_m) (m = 0...2) 140
1.16.5. TCM correctable bit error address 2 for SRAM_m register (TCM_CBEADDR2_m) (m = 0...2) 141

1.16.6. TCM correctable bit error address 3 for SRAM_m register (TCM_CBEADDR3_m) (m = 0...2) 141
1.16.7. TCM uncorrectable bit error address for SRAM_m register (TCM_UBEADDR_m) (m = 0...2) 142
1.16.8. TCM buffer valid register (TCM_BUFVLD) 142
1.16.9. TCM overflow status register (TCM_OVF) 144
1.16.10. TCM write protection register (TCM_WP) 145
1.16.11. TCM0 ECC code error status register (TCM0_ECCCERR) 146
1.17. Device electronic signature 146
1.17.1. Memory density information 146
1.17.2. Unique device ID (96 bits) 147
2. Flash memory controller (FMC) 149
2.1. Overview 149
2.2. Characteristics 149
2.3. Function overview 150
2.3.1. Flash memory architecture 150
2.3.2. Error Checking and Correcting (ECC) 153
2.3.3. Read operations 155
2.3.4. Quad block architecture with read-while-write (RWW) capability 157
2.3.5. FMC write interface 157
2.3.6. Unlock the FMC_CTLx register 158
2.3.7. Page erase 158
2.3.8. Mass erase 160
2.3.9. Flash programming 162
2.3.10. Flash Fast Programming 165
2.3.11. Check blank command 167
2.3.12. OTP programming 167
2.3.13. Option bytes erase 167
2.3.14. Option bytes programming 168
2.3.15. Option bytes description 169
2.3.16. Page erase / program protection 174
2.3.17. Security protection 177
2.3.18. Error description 178
2.3.19. Life cycle 179
2.3.20. Application debug configuration 181
2.3.21. Cryptography key configuration 182
2.3.22. Security protection key configuration 182
2.3.23. HSM configuration 183
2.4. Register definition 185
2.4.1. Wait state register (FMC_WS) 186
2.4.2. Unlock SPC protection key register (FMC_SPCKEY) 186
2.4.3. Unlock key register x (FMC_KEYx) (x = 0, 1) 187
2.4.4. Status register x (FMC_STATx) (x = 0, 1) 187
2.4.5. Control register x (FMC_CTLx) (x = 0, 1) 189
2.4.6. Address register x (FMC_ADDRx) (x = 0, 1) 191
2.4.7. Write data low x register (FMC_WDATALx) (x = 0, 1) 191
2.4.8. Write data high x register (FMC_WDATAHx) (x = 0, 1) 192
2.4.9. Option byte status register (FMC_OBSTAT) 192
2.4.10. Erase / Program protection register 0 (FMC_WP0) 193
2.4.11. Erase / Program protection register 1 (FMC_WP1) 194
2.4.12. Erase / Program protection register 2 (FMC_WP2) 194
2.4.13. Option byte USER1 register 0 (FMC_OBUSER1_0) 194
2.4.14. Option byte USER1 register 1 (FMC_OBUSER1_1) 195
2.4.15. HSM option byte status register (FMC_HSMOBSTAT) 195
2.4.16. HSM Erase / Program protection register (FMC_HSMWP) 196
2.4.17. Blockx ECC control and status register (FMC_BLKxECCCS) (x = 0,1..3) 197
2.4.18. Blockx uncorrectable ECC address register (FMC_BLKxUCECCADDR) (x = 0,1..3) 199

2.4.19. Blockx correctable ECC address 0 register (FMC_BLKxCECCADDR0) (x = 0,1..3) 199
2.4.20. Blockx correctable ECC address 1 register (FMC_BLKxCECCADDR1) (x = 0,1..3) 200
2.4.21. Blockx correctable ECC address 2 register (FMC_BLKxCECCADDR2) (x = 0,1..3) 200
2.4.22. Blockx correctable ECC address 3 register (FMC_BLKxCECCADDR3) (x = 0,1..3) 201
2.4.23. Bufferx ECC control and status register (FMC_BUFxECCCS) (x = 0,2,3) 202
2.4.24. Bufferx uncorrectable ECC address register (FMC_BUFxUCECCADDR) (x = 0,2,3) 204
2.4.25. Bufferx correctable ECC address 0 register (FMC_BUFxCECCADDR0) (x = 0,2,3) 204
2.4.26. Bufferx correctable ECC address 1 register (FMC_BUFxCECCADDR1) (x = 0,2,3) 205
2.4.27. Bufferx correctable ECC address 2 register (FMC_BUFxCECCADDR2) (x = 0,2,3) 205
2.4.28. Bufferx correctable ECC address 3 register (FMC_BUFxCECCADDR3) (x = 0,2,3) 206
2.4.29. Fast program buffer ECC control and status register (FMC_FPB_ECCCS) 206
2.4.30. Fast program buffer uncorrectable ECC address register (FMC_FPB_UCECCADDR) 208
2.4.31. Fast program buffer correctable ECC address 0 register (FMC_FPB_CECCADDR0) 208
2.4.32. Fast program buffer correctable ECC Address 1 register (FMC_FPB_CECCADDR1) 209
2.4.33. Fast program buffer correctable ECC address 2 register (FMC_FPB_CECCADDR2) 209
2.4.34. Fast program buffer correctable ECC address 3 register (FMC_FPB_CECCADDR3) 210
2.4.35. Product ID register 0 (FMC_PID0) 210
2.4.36. Product ID register 1 (FMC_PID1) 210
3. Power management unit (PMU) 212
3.1. Overview 212
3.2. Characteristics 212
3.3. Function overview 213
3.3.1. Voltage domain description 213
3.3.2. Voltage monitors 214
3.3.3. Temperature monitors 214
3.3.4. ABIST 215
3.3.5. Power work modes 215
3.3.6. Standby mode entrance sequence 216
3.3.7. Chip status at the end of standby mode entrance sequence 217
3.3.8. Chip operations in standby mode 217
3.3.9. Standby mode exit 218
3.3.10. Pad retain 218
3.4. Register definition 220
3.4.1. Low voltage control and status register (PMU_LVCS) 220
3.4.2. Configuration register (PMU_CFG) 223
3.4.3. Analog BIST control register (PMU_ABCTL) 225
3.4.4. Analog BIST status register (PMU_ABSTAT) 225
3.4.5. Version ID register (PMU_VID) 226
3.4.6. Control key register (PMU_CTLKEY) 227
3.4.7. Mode configure register (PMU_MODCFG) 227
3.4.8. Mode update register (PMU_MODUPD) 228
3.4.9. Mode status register (PMU_MODSTAT) 229
3.4.10. Core update register (PMU_CUPD) 229
3.4.11. Control register (PMU_CTL) 229
3.4.12. TMR error enable register (PMU_TMRERREN) 230
3.4.13. TMR error interrupt register (PMU_TMRERRIF) 231
3.4.14. TMR error interrupt clear register (PMU_TMRERRIFC) 232
4. Backup registers (BKP) 234
4.1. Overview 234
4.2. Characteristics 234
4.3. Function overview 234
4.3.1. RTC clock calibration 234
4.3.2. Tamper detection 234
4.4. Register definition 236
4.4.1. Backup data register x (BKP_DATAx) (x= 0...9) 236
4.4.2. RTC signal output control register (BKP_OCTL) 236

4.4.3. Tamper pin control register (BKP_TPCTL) 237
4.4.4. Tamper control and status register (BKP_TPCS) 238
5. Reset and clock unit (RCU) 240
5.1. Reset control unit (RCTL) 240
5.1.1. Reset overview 240
5.1.2. POR reset 241
5.1.3. System reset 242
5.1.4. Subsystem reset 243
5.1.5. Reset pin 246
5.1.6. Standby reset 246
5.1.7. Chip reset and boot 246
5.1.8. Power-on reset watchdog (POR_WDG) 248
5.2. Clock control unit (CCTL) 249
5.2.1. Overview 249
5.2.2. Characteristics 251
5.2.3. Function overview 251
5.3. Reset control unit registers definition 259
5.3.1. System Reset Status register (RCU_SYSRSTSTAT) 260
5.3.2. Bus control register (RCU_BUSCTL) 261
5.3.3. Subsystem/external reset status register (RCU_SUBSERSTAT) 262
5.3.4. Subsystem reset control register (RCU_SUBSRCTL) 264
5.3.5. Subsystem reset pin assert control register (RCU_SUBSRPACTL) 265
5.3.6. Subsystem reset mode upgrade counter register (RCU_SUBSRMUCNT) 265
5.3.7. Subsystem reset mode upgrade threshold register (RCU_SUBSRMUTH) 266
5.3.8. System reset mode upgrade threshold register (RCU_SYSMRUTH) 266
5.3.9. External reset pin assert configuration register (RCU_ERPACFG) 267
5.3.10. Reset during standby mode status register (RCU_RDSMSTAT) 267
5.3.11. Subsystem reset timeout register (RCU_SUBSRTO) 268
5.3.12. Low Power Debug Status Register (RCU_LPDBGSTAT) 268
5.3.13. AHB reset register (RCU_AHBRST) 269
5.3.14. APB1 reset register (RCU_APB1RST) 272
5.3.15. APB2 reset register (RCU_APB2RST) 274
5.3.16. APB2 reset additional register (RCU_APB2RST_ADD) 277
5.3.17. Power-on reset during POR watchdog overflows status register 0 (RCU_PORSTAT0) 278
5.3.18. Power-on reset during POR watchdog overflows status register 1 (RCU_PORSTAT1) 280
5.3.19. Power-on reset during POR watchdog overflows status register 2 (RCU_PORSTAT2) 283
5.3.20. Power-on reset during POR watchdog overflows status register 3 (RCU_PORSTAT3) 286
5.3.21. Triple modular redundancy error status (RCU_TMRERRSTAT) 287
5.3.22. System reset upgrade counter register (RCU_SYSRSTUCNT) 289
5.3.23. Subsystem reset counter register (RCU_SUBSRSTCNT) 290
5.3.24. System reset counter register (RCU_SYSRSTCNT) 290
5.3.25. Subsystem reset all disable register (RCU_SUBSRSTAD) 291
5.4. Clock control unit registers definition 292
5.4.1. Control register (RCU_CTL) 292
5.4.2. Configuration register 0 (RCU_CFG0) 294
5.4.3. Interrupt register (RCU_INT) 297
5.4.4. AHB enable register (RCU_AHBEN) 299
5.4.5. APB2 enable register (RCU_APB2EN) 302
5.4.6. APB1 enable register (RCU_APB1EN) 304
5.4.7. Backup domain control register (RCU_BDCTL) 307
5.4.8. APB2 enable additional register (RCU_APB2EN_ADD) 308
5.4.9. AHB enable additional register (RCU_AHBEN_ADD) 309
5.4.10. Configuration register 1 (RCU_CFG1) 310
5.4.11. Configuration register 2 (RCU_CFG2) 312
5.4.12. Configuration register 3 (RCU_CFG3) 314
5.4.13. PLL loss of lock register (RCU_PLLLOL) 316
5.4.14. Clock detect control register(RCU_CKDCTL) 316

5.4.15. Clock frequency monitor x control register (RCU_CKFMxCTL, x=0..4) 317
5.4.16. Clock frequency monitor x reference count configuration register (RCU_CKFMxRCCFG, x=0..4) 319
5.4.17. CKFMx high threshold configuration register (RCU_CKFMxHTHCFG, x=0..4) 319
5.4.18. CKFMx low threshold configuration register (RCU_CKFMxLTHCFG, x=0..4) 320
5.4.19. CCTL TMR register (RCU_BUSIM_TMR) 321
5.4.20. PLL fraction configuration register (RCU_PLLFRA) 322
5.4.21. PLL clock spread spectrum control register(RCU_PLLSSCTL) 323
6. Interrupt / event controller (EXTI) 325
6.1. Overview 325
6.2. Characteristics 325
6.3. External interrupt and event (EXTI) block diagram 325
6.4. External interrupt and event function overview 325
6.5. Register definition 330
6.5.1. Interrupt enable register 0 (EXTI_INTEN0) 330
6.5.2. Event enable register 0 (EXTI_EVEN0) 331
6.5.3. Rising edge trigger enable register 0 (EXTI_RTEN0) 332
6.5.4. Falling edge trigger enable register 0 (EXTI_FTEN0) 332
6.5.5. Software interrupt event register 0 (EXTI_SWIEV0) 333
6.5.6. Pending register 0 (EXTI_PD0) 333
6.5.7. Interrupt enable register 1 (EXTI_INTEN1) 334
6.5.8. Event enable register 1 (EXTI_EVEN1) 335
6.5.9. Rising edge trigger enable register 1 (EXTI_RTEN1) 335
6.5.10. Falling edge trigger enable register 1 (EXTI_FTEN1) 336
6.5.11. Software interrupt event register 1 (EXTI_SWIEV1) 336
6.5.12. Pending register 1 (EXTI_PD1) 337
6.5.13. Interrupt enable register 2 (EXTI_INTEN2) 337
6.5.14. Event enable register 2 (EXTI_EVEN2) 338
6.5.15. Rising edge trigger enable register 2 (EXTI_RTEN2) 338
6.5.16. Falling edge trigger enable register 2 (EXTI_FTEN2) 338
6.5.17. Software interrupt event register 2 (EXTI_SWIEV2) 339
6.5.18. Pending register 2 (EXTI_PD2) 339
6.5.19. Interrupt enable register 3 (EXTI_INTEN3) 340
6.5.20. Event enable register 3 (EXTI_EVEN3) 340
6.5.21. Rising edge trigger enable register 3 (EXTI_RTEN3) 340
6.5.22. Falling edge trigger enable register 3 (EXTI_FTEN3) 341
6.5.23. Software interrupt event register 3 (EXTI_SWIEV3) 341
6.5.24. Pending register 3 (EXTI_PD3) 342
7. Trigger selection controller (TRIGSEL) 343
7.1. Overview 343
7.2. Characteristics 343
7.3. Function overview 343
7.4. Internal connect 344
7.5. Register definition 352
7.5.1. Trigger selection for EXTOUT register 0 (TRIGSEL_EXTOUT_0) 354
7.5.2. Trigger selection for EXTOUT register 1 (TRIGSEL_EXTOUT_1) 355
7.5.3. Trigger selection for EXTOUT register 2 (TRIGSEL_EXTOUT_2) 355
7.5.4. Trigger selection for EXTOUT register 3 (TRIGSEL_EXTOUT_3) 356
7.5.5. Trigger selection for EXTOUT register 4 (TRIGSEL_EXTOUT_4) 357
7.5.6. Trigger selection for EXTOUT register 5 (TRIGSEL_EXTOUT_5) 357
7.5.7. Trigger selection for EXTOUT register 6 (TRIGSEL_EXTOUT_6) 358
7.5.8. Trigger selection for EXTOUT register 7 (TRIGSEL_EXTOUT_7) 358
7.5.9. Trigger selection for TIMER0_ITI0 register (TRIGSEL_TIMER0ITI0) 359
7.5.10. Trigger selection for TIMER1_ITI0 register (TRIGSEL_TIMER1ITI0) 360
7.5.11. Trigger selection for TIMER2_ITI0 register (TRIGSEL_TIMER2ITI0) 360
7.5.12. Trigger selection for TIMER7_ITI0 register (TRIGSEL_TIMER7ITI0) 361

7.5.13. Trigger selection for TIMER19_ITI0 register (TRIGSEL_TIMER19ITI0) 361
7.5.14. Trigger selection for TIMER20_ITI0register (TRIGSEL_TIMER20ITI0) 362
7.5.15. Trigger selection for TIMER60_ITI0 register (TRIGSEL_TIMER60ITI0) 363
7.5.16. Trigger selection for TIMER61_ITI0 register (TRIGSEL_TIMER61ITI0) 363
7.5.17. Trigger selection for TIMER62_ITI0 register (TRIGSEL_TIMER62ITI0) 364
7.5.18. Trigger selection for TIMER63_ITI0 register (TRIGSEL_TIMER63ITI0) 364
7.5.19. Trigger selection for TIMER0_BRKIN register 0 (TRIGSEL_TIMER0BRKIN_0) 365
7.5.20. Trigger selection for TIMER0_BRKIN register 1 (TRIGSEL_TIMER0BRKIN_1) 366
7.5.21. Trigger selection for TIMER7_BRKIN register 0 (TRIGSEL_TIMER7BRKIN_0) 366
7.5.22. Trigger selection for TIMER7_BRKIN register 1 (TRIGSEL_TIMER7BRKIN_1) 367
7.5.23. Trigger selection for TIMER19_BRKIN register 0 (TRIGSEL_TIMER19BRKIN_0) 368
7.5.24. Trigger selection for TIMER19_BRKIN register 1 (TRIGSEL_TIMER19BRKIN_1) 368
7.5.25. Trigger selection for TIMER20_BRKIN register 0 (TRIGSEL_TIMER20BRKIN_0) 369
7.5.26. Trigger selection for TIMER20_BRKIN register 1 (TRIGSEL_TIMER20BRKIN_1) 370
7.5.27. Trigger selection for TIMER60_BRKIN register 0 (TRIGSEL_TIMER60BRKIN_0) 370
7.5.28. Trigger selection for TIMER60_BRKIN register 1 (TRIGSEL_TIMER60BRKIN_1) 371
7.5.29. Trigger selection for TIMER61_BRKIN register 0 (TRIGSEL_TIMER61BRKIN_0) 372
7.5.30. Trigger selection for TIMER61_BRKIN register 1 (TRIGSEL_TIMER61BRKIN_1) 372
7.5.31. Trigger selection for TIMER62_BRKIN register 0 (TRIGSEL_TIMER62BRKIN_0) 373
7.5.32. Trigger selection for TIMER62_BRKIN register 1 (TRIGSEL_TIMER62BRKIN_1) 374
7.5.33. Trigger selection for TIMER63_BRKIN register 0 (TRIGSEL_TIMER63BRKIN_0) 375
7.5.34. Trigger selection for TIMER63_BRKIN register 1 (TRIGSEL_TIMER63BRKIN_1) 375
7.5.35. Trigger selection for CM7 register (TRIGSEL_CM7) 376
7.5.36. Trigger selection for DAC register (TRIGSEL_DAC) 377
7.5.37. Trigger selection for ADC0_ROUTRG register (TRIGSEL_ADC0ROUTRG) 377
7.5.38. Trigger selection for ADC0_INSTRG register 0 (TRIGSEL_ADC0INSTRG_0) 378
7.5.39. Trigger selection for ADC0_INSTRG register 1 (TRIGSEL_ADC0INSTRG_1) 379
7.5.40. Trigger selection for ADC1_ROUTRG register (TRIGSEL_ADC1ROUTRG) 380
7.5.41. Trigger selection for ADC1_INSTRG register 0 (TRIGSEL_ADC1INSTRG_0) 380
7.5.42. Trigger selection for ADC1_INSTRG register 1 (TRIGSEL_ADC1INSTRG_1) 381
7.5.43. Trigger selection for SPI register 0 (TRIGSEL_SPI_0) 382
7.5.44. Trigger selection for SPI register 1 (TRIGSEL_SPI_1) 382
7.5.45. Trigger selection for SPI register 2 (TRIGSEL_SPI_2) 383
7.5.46. Trigger selection for TIMER_ETI register (TRIGSEL_TIMERETI) 384
7.5.47. Trigger selection for LINFlexD register 0 (TRIGSEL_LINFlexD_0) 385
7.5.48. Trigger selection for LINFlexD register 1 (TRIGSEL_LINFlexD_1) 385
7.5.49. Trigger selection for LINFlexD register 2 (TRIGSEL_LINFlexD_2) 386
8. General-purpose and alternate-function I/Os (GPIO and AFIO) 387
8.1. Overview 387
8.2. Characteristics 387
8.3. Function overview 388
8.3.1. GPIO pin configuration 389
8.3.2. External interrupt / event lines 390
8.3.3. Alternate functions (AF) 390
8.3.4. Additional functions 390
8.3.5. Input configuration 390
8.3.6. Output configuration 391
8.3.7. Analog configuration 392
8.3.8. Alternate function (AF) configuration 392
8.3.9. GPIO locking function 393
8.3.10. GPIO single cycle toggle function 393
8.3.11. Pad input filter 393
8.3.12. External interrupt filter 393
8.3.13. Pad retain 394
8.3.14. Triple-modular redundancy (TMR) 394
8.3.15. Pin States 395
8.4. Register definition 398

8.4.1. Port control register (GPIOx_CTL, x=A…N) 400
8.4.2. Port output mode register (GPIOx_OMODE, x=A...N) 402
8.4.3. Port output speed register (GPIOx_OSPD, x=A...N) 403
8.4.4. Port pull-up / down register (GPIOx_PUD, x=A...N) 407
8.4.5. Port input status register (GPIOx_ISTAT, x=A...N) 409
8.4.6. Port output control register (GPIOx_OCTL, x=A...N) 409
8.4.7. Port bit operate register (GPIOx_BOP, x=A...N) 410
8.4.8. Port configuration lock register (GPIOx_LOCK, x=A...N) 410
8.4.9. Alternate function selected register 0 (GPIOx_AFSEL0, x=A...N) 411
8.4.10. Alternate function selected register 1 (GPIOx_AFSEL1, x=A...N) 412
8.4.11. Bit clear register (GPIOx_BC, x=A..N) 414
8.4.12. Port bit toggle register (GPIOx_TG, x=A...N) 414
8.4.13. Port safe mode control register (GPIOx_SMCCTL, x=A...N) 415
8.4.14. Port filtering enable register (GPIOx_IFEN, x=A...N) 415
8.4.15. Port retain enable register (GPIOx_PREN, x=A...N) 416
8.4.16. Port slew rate enable register (GPIOx_SREN, x=A...N) 416
8.4.17. Interrupt filtering control Register (GPIOx_IFCTL, x=A...N) 417
8.4.18. Interrupt filtering sample counter control register 0(GPIOx_IFSCCTL0, x=A...N) 417
8.4.19. Interrupt filtering sample counter control register 1(GPIOx_IFSCCTL1, x=A...N) 418
8.4.20. TMR alarm status register (GPIOx_TMRASTAT, x=B, E, J, K) 419
9. Cyclic redundancy checks management unit (CRC) 421
9.1. Overview 421
9.2. Characteristics 421
9.3. Function overview 422
9.4. Register definition 423
9.4.1. Data register (CRC_DATA) 423
9.4.2. Free data register (CRC_FDATA) 423
9.4.3. Control register (CRC_CTL) 424
9.4.4. Initialization data register (CRC_IDATA) 425
9.4.5. Polynomial register (CRC_POLY) 425
10. Direct memory access controller (DMA) 426
10.1. Overview 426
10.2. Characteristics 426
10.3. Function overview 427
10.3.1. Block diagram 427
10.3.2. Peripheral handshake 429
10.3.3. Data process 429
10.3.4. Address generation 437
10.3.5. Circular mode 437
10.3.6. Switch-buffer mode 437
10.3.7. Transfer flow controller 438
10.3.8. Transfer operation 439
10.3.9. Transfer finish 439
10.3.10. Channel configuration 441
10.4. Interrupts 442
10.4.1. Flag 443
10.4.2. Exception 443
10.4.3. Error 444
10.4.4. DMA request mapping 445
10.5. Fault interrupt management 446
10.5.1. Channel x transfer timeout monitor 446
10.5.2. Channel x control register enhanced 446
10.6. CRC engine 447
10.6.1. Description 447
10.6.2. Function 447
10.7. Register definition 448

10.7.1. Interrupt flag register 0 (DMA_INTF0) 448
10.7.2. Interrupt flag register 1 (DMA_INTF1) 449
10.7.3. Interrupt flag clear register 0 (DMA_INTC0) 450
10.7.4. Interrupt flag clear register 1 (DMA_INTC1) 451
10.7.5. Channel x control register (DMA_CHxCTL) 452
10.7.6. Channel x counter register (DMA_CHxCNT) 456
10.7.7. Channel x peripheral base address register (DMA_CHxPADDR) 457
10.7.8. Channel x memory 0 base address register (DMA_CHxM0ADDR) 457
10.7.9. Channel x memory 1 base address register (DMA_CHxM1ADDR) 458
10.7.10. Channel x FIFO control register (DMA_CHxFCTL) 459
10.7.11. Channel x peripheral CRC data register (DMA_CHxPCRCDATA) 460
10.7.12. CRC initialization data register (DMA_CRCIDATA) 461
10.7.13. Peripheral port CRC control register (DMA_PCRCCTL) 461
10.7.14. Global control register (DMA_GCTL) 462
10.7.15. Channel x transfer timeout monitor control register (DMA_CHxTOCTL) 463
10.7.16. Timeout interrupt flag register (DMA_TOINTF) 463
10.7.17. Timeout interrupt enable register (DMA_TOINTEN) 464
10.7.18. TMR fault interrupt flag register (DMA_TFINTF) 464
10.7.19. TMR fault interrupt enable register (DMA_TFINTEN) 465
10.7.20. Channel x peripheral port current address register (DMA_CHxPCADDR) 465
10.7.21. Channel x memory port current address register (DMA_CHxMCADDR) 466
11. DMA request multiplexer (DMAMUX) 467
11.1. Overview 467
11.2. Characteristics 467
11.3. Block diagram 468
11.4. Signal description 468
11.5. Function overview 469
11.5.1. DMAMUX request multiplexer 469
11.5.2. DMAMUX request generator 472
11.5.3. Channel configurations 472
11.5.4. Interrupt 473
11.5.5. DMAMUX mapping 473
11.6. Register definition 482
11.6.1. Request multiplexer channel x configuration register (DMAMUX_RM_CHxCFG) 482
11.6.2. Request multiplexer channel interrupt flag register (DMAMUX_RM_INTF) 483
11.6.3. Request multiplexer channel interrupt flag clear register (DMAMUX_RM_INTC) 484
11.6.4. Request generator channel x configuration register (DMAMUX_RG_CHxCFG) 484
11.6.5. Request generator channel interrupt flag register (DMAMUX_RG_INTF) 485
11.6.6. Request generator channel interrupt flag clear register (DMAMUX_RG_INTC) 486
12. Debug (DBG) 487
12.1. Overview 487
12.2. JTAG / SW function overview 487
12.2.1. Pin assignment 487
12.2.2. JTAG daisy chained structure 488
12.2.3. Debug reset 488
12.2.4. JEDEC-106 ID code 488
12.3. Debug hold function overview 488
12.3.1. Debug support for TIMER, I2C, WWDGT, FWDGT and MFCOM 488
12.4. Register definition 489
12.4.1. Control register0 (DBG_CTL0) 489
12.4.2. Control register1 (DBG_CTL1) 489
12.4.3. Control register2 (DBG_CTL2) 491
13. Analog-to-digital converter (ADC) 493
13.1. Overview 493
13.2. Characteristics 493
13.3. Pins and internal signals 494

13.4. Function Description 497
13.4.1. Foreground calibration function 497
13.4.2. Dual clock domain architecture 498
13.4.3. Low power mode 498
13.4.4. Sequence management 499
13.4.5. Operation mode 500
13.4.6. Abort a sequence 501
13.4.7. Programmable presample and sample time 502
13.4.8. Analog multiplexer channel 503
13.4.9. Analog multiplexer decode signals delay 505
13.4.10. Conversion start delay 505
13.4.11. Analog watchdog 506
13.4.12. Data management 506
13.4.13. Trigger source 510
13.4.14. DMA request 510
13.4.15. Programmable resolution - fast conversion mode 511
13.4.16. ADC internal channels 511
13.5. ADC sync mode 512
13.5.1. Free mode 513
13.5.2. Routine parallel mode 513
13.5.3. Inserted parallel mode 514
13.5.4. Follow-up mode 514
13.5.5. Trigger rotation mode 515
13.5.6. Combined routine parallel & inserted parallel mode 516
13.5.7. Combined routine parallel & trigger rotation mode 516
13.6. Self test 517
13.6.1. Self test modes 517
13.6.2. Self test in routine sequence 518
13.6.3. Analog watchdog for self test 521
13.6.4. Watchdog timer for self test 522
13.6.5. Self test steps execution rate control 524
13.6.6. Abort self test 524
13.6.7. Self test steps order checking 525
13.7. DAC-ADC loop test 525
13.8. ADC interrupts 525
13.9. FMU interrupts 526
13.10. ADC registers 527
13.10.1. Control register (ADC_CTL) 529
13.10.2. Status register (ADC_STAT) 532
13.10.3. Error status register (ADC_ESTAT) 534
13.10.4. Watchdog threshold register x (ADC_WDTx) (x = 0…3) 538
13.10.5. Presampling time configure register(ADC_PSMPCFG) 539
13.10.6. Channel configure register n (ADC_CHCFGn) (n = 0...61) 539
13.10.7. Watchdog interrupt flag register 0 (ADC_WDINTF0) 541
13.10.8. Watchdog interrupt flag register 1 (ADC_WDINTF1) 541
13.10.9. External decode delay register (ADC_EDDLY) 542
13.10.10. Conversion start delay register (ADC_CSDLY) 542
13.10.11. Gain configure register (ADC_GAINCFG) 543
13.10.12. Offset configure register x (ADC_OFFCFGx) (x = 0…3) 544
13.10.13. Sync control register (ADC_SYNCCTL) 545
13.10.14. Routine sequence control register (ADC_RCTL) 545
13.10.15. Routine sequence stream register (ADC_RSTREAM) 547
13.10.16. Routine sequence configure register x (ADC_RCFGx) (x = 0…27) 548
13.10.17. Routine sequence channel conversion end interrupt flag register (ADC_RCHINTF) 549
13.10.18. Routine sequence channel data register x (ADC_RCHDATAx) (x = 0…27) 549
13.10.19. Routine sequence data register (ADC_RDATA) 550
13.10.20. Inserted sequence m control register (ADC_ImCTL) (m = 0...5) 551

13.10.21. Inserted sequence m stream register (ADC_ImSTREAM) (m = 0...5) 553
13.10.22. Inserted sequence m configure register x (ADC_ImCFGx) (m = 0...5, x = 0...3) 554
13.10.23. Inserted sequence m channel conversion end interrupt flag register (ADC_ImCHINTF) (m = 0...5) 555
13.10.24. Inserted sequence m channel data register x (ADC_ImCHDATAx) (m = 0...5, x = 0...3)... 555
13.10.25. Inserted sequence m data register (ADC_ImDATA) (m = 0...5) 556
13.10.26. Self test sample time register (ADC_STSAMPT) 558
13.10.27. Self test interrupt enable register (ADC_STINTEN) 559
13.10.28. Self test configure register (ADC_STCFG) 560
13.10.29. Self test status register (ADC_STSTAT) 562
13.10.30. Self test error data register 0 (ADC_STEDATA0) 563
13.10.31. Self test error data register 1 (ADC_STEDATA1) 564
13.10.32. Self test conversion data register (ADC_STDATA) 564
13.10.33. Watchdog for supply self test configure register 0 (ADC_SSTWDCFG0) 565
13.10.34. Watchdog for supply self test configure register 1 (ADC_SSTWDCFG1) 566
13.10.35. Watchdog for supply self test configure register 2 (ADC_SSTWDCFG2) 567
13.10.36. Watchdog for capacitive self test configure register 0 (ADC_CSTWDCFG0) 567
13.10.37. Watchdog for capacitive self test configure register 1 (ADC_CSTWDCFG1) 568
13.10.38. Watchdog for capacitive self test configure register 2 (ADC_CSTWDCFG2) 569
13.10.39. Watchdog for capacitive self test configure register 3 (ADC_CSTWDCFG3) 570
13.10.40. DAC - ADC loop test configure register (ADC_DTALTCFG) 571
14. Digital-to-analog converter (DAC) 573
14.1. Overview 573
14.2. Characteristics 573
14.3. Function overview 575
14.3.1. DAC enable 575
14.3.2. DAC output buffer 575
14.3.3. DAC data configuration 575
14.3.4. DAC trigger 575
14.3.5. DAC conversion 576
14.3.6. DAC noise wave 576
14.3.7. DAC output calculate 577
14.3.8. DMA request 577
14.3.9. Loop test 577
14.4. Registers definition 580
14.4.1. DACx control register 0 (DAC_CTL0) 580
14.4.2. DACx software trigger register (DAC_SWT) 582
14.4.3. DACx_OUT0 12-bit right-aligned data holding register (DAC_OUT0_R12DH) 582
14.4.4. DACx_OUT0 12-bit left-aligned data holding register (DAC_OUT0_L12DH) 582
14.4.5. DACx_OUT0 8-bit right-aligned data holding register (DAC_OUT0_R8DH) 583
14.4.6. DACx_OUT0 data output register (DAC_OUT0_DO) 583
14.4.7. DACx status register 0 (DAC_STAT0) 584
14.4.8. DACx Loop test control register (DAC_LTCTL) 584
14.4.9. DACx Loop test pattern register(DAC_LTPAT) 585
15. Watchdog timer (WDGT) 587
15.1. Free watchdog timer (FWDGT) 587
15.1.1. Overview 587
15.1.2. Characteristics 587
15.1.3. Function overview 587
15.1.4. Register definition 590
15.2. Window watchdog timer (WWDGT) 594
15.2.1. Overview 594
15.2.2. Characteristics 594
15.2.3. Function overview 594
15.2.4. Register definition 597
16. Real-time clock (RTC) 601

16.1. Overview 601
16.2. Characteristics 601
16.3. Function overview 601
16.3.1. RTC reset 602
16.3.2. RTC reading 602
16.3.3. RTC configuration 602
16.3.4. RTC flag assertion 603
16.4. Register definition 605
16.4.1. RTC interrupt enable register (RTC_INTEN) 605
16.4.2. RTC control register (RTC_CTL) 606
16.4.3. RTC prescaler high register (RTC_PSCH) 607
16.4.4. RTC prescaler low register (RTC_PSCL) 607
16.4.5. RTC divider high register (RTC_DIVH) 607
16.4.6. RTC divider low register (RTC_DIVL) 608
16.4.7. RTC counter high register (RTC_CNTH) 608
16.4.8. RTC counter low register (RTC_CNTL) 609
16.4.9. RTC alarm high register (RTC_ALRMH) 609
16.4.10. RTC alarm low register (RTC_ALRML) 609
17. TIMER 611
17.1. Advanced timer (TIMERx, x=0, 7, 19, 20, 60, 61, 62, 63) 613
17.1.1. Overview 613
17.1.2. Characteristics 613
17.1.3. Block diagram 614
17.1.4. Function overview 614
17.1.5. Registers definition (TIMERx, x=0, 7, 19, 20, 60 , 61 , 62 , 63) 664
17.2. General level0 timer (TIMERx, x=1, 2) 740
17.2.1. Overview 740
17.2.2. Characteristics 740
17.2.3. Block diagram 740
17.2.4. Function overview 741
17.2.5. Registers definition (TIMERx, x=1,2) 766
17.3. Basic timer (TIMERx, x=5, 6) 805
17.3.1. Overview 805
17.3.2. Characteristics 805
17.3.3. Block diagram 805
17.3.4. Function overview 805
17.3.5. Registers definition (TIMERx, x=5, 6) 809
18. Local Interconnect Network Flexible Driver (LINFlexD) 821
18.1. Overview 821
18.2. Characteristics 821
18.2.1. LIN mode features 822
18.2.2. UART mode features 822
18.3. Function overview 823
18.3.1. LIN protocol 823
18.3.2. Operating modes 825
18.3.3. Errors 828
18.3.4. Identifier filtering 833
18.3.5. Start detection and break delimiter detection in receiver 835
18.3.6. Baud rate generation 839
18.3.7. Automatic resynchronization 840
18.3.8. Wakeup management 840
18.3.9. Timer 840
18.3.10. UART mode 841
18.3.11. Interrupts 845
18.3.12. DMA operations 846
18.3.13. LINFlexD clock tolerance 850

18.3.14. Low-power mode 851
18.3.15. LINS status 851
18.3.16. Header Trigger 853
18.4. Register definition 854
18.4.1. LIN control register 1(LINCR1) 854
18.4.2. LIN interrupt enable register (LINIER) 857
18.4.3. LIN status register (LINSR) 859
18.4.4. LIN error status register (LINESR) 861
18.4.5. UART mode control register (UARTCR) 862
18.4.6. UART mode status register(UARTSR) 867
18.4.7. LIN timeout control status register (LINTCSR) 870
18.4.8. LIN output compare register (LINOCR) 871
18.4.9. LIN timeout control register (LINTOCR) 871
18.4.10. LIN fractional baud rate register(LINFBRR) 872
18.4.11. LIN integer baud rate register(LINIBRR) 873
18.4.12. LIN checksum field register(LINCFR) 873
18.4.13. LIN control register 2(LINCR2) 874
18.4.14. LIN buffer identifier register (BIDR) 876
18.4.15. LIN buffer data register least significant (BDRL) 876
18.4.16. LIN buffer data register most significant (BDRM) 877
18.4.17. LIN identifier filter enable register (IFER) 877
18.4.18. LIN identifier filter match index (IFMI) 878
18.4.19. LIN identifier filter mode register(IFMR) 878
18.4.20. LIN identifier filter control register (IFCRn) (n = 0…15) 879
18.4.21. LIN global control register (GCR) 880
18.4.22. LIN UART preset timeout register (UARTPTO) 881
18.4.23. LIN UART current timeout register (UARTCTO) 882
18.4.24. LIN DMA Tx enable register (DMATXE) 883
18.4.25. LIN DMA Rx enable register (DMARXE) 883
18.4.26. LIN stop control register (LIN_STCTL) 884
19. Inter-integrated circuit interface (I2C) 886
19.1. Overview 886
19.2. Characteristics 886
19.3. Function overview 886
19.3.1. Clock requirements 887
19.3.2. I2C communication flow 888
19.3.3. Noise filter 890
19.3.4. I2C timings configuration 891
19.3.5. I2C reset 892
19.3.6. Data transfer 893
19.3.7. I2C slave mode 895
19.3.8. I2C master mode 900
19.3.9. SMBus support 905
19.3.10. SMBus mode 908
19.3.11. Use DMA for data transfer 909
19.3.12. I2C error and interrupts 910
19.3.13. I2C debug mode 911
19.4. Register definition 912
19.4.1. Control register 0 (I2C_CTL0) 912
19.4.2. Control register 1 (I2C_CTL1) 914
19.4.3. Slave address register 0 (I2C_SADDR0) 916
19.4.4. Slave address register 1 (I2C_SADDR1) 917
19.4.5. Timing register (I2C_TIMING) 918
19.4.6. Timeout register (I2C_TIMEOUT) 919
19.4.7. Status register (I2C_STAT) 920
19.4.8. Status clear register (I2C_STATC) 923
19.4.9. PEC register (I2C_PEC) 924

19.4.10. Receive data register (I2C_RDATA) 924
19.4.11. Transmit data register (I2C_TDATA) 925
19.4.12. Control register 2 (I2C_CTL2) 925
20. Serial peripheral interface/Inter-IC sound (SPI/I2S) 926
20.1. Overview 926
20.2. Characteristics 926
20.2.1. SPI characteristics 926
20.2.2. I2S characteristics 927
20.3. SPI function overview 927
20.3.1. SPI block diagram 927
20.3.2. SPI clock sources 927
20.3.3. SPI signal description 928
20.3.4. SPI clock timing and data format 929
20.3.5. SPI master receive delay compensation 930
20.3.6. Separate transmission and reception FIFO 931
20.3.7. Idle timeout 932
20.3.8. NSS function 932
20.3.9. NSS switching 934
20.3.10. SPI operation modes 934
20.3.11. DMA function 945
20.3.12. CRC function 945
20.3.13. SPI interrupts 946
20.4. I2S function overview 948
20.4.1. I2S block diagram 948
20.4.2. I2S signal description 948
20.4.3. I2S audio standards 949
20.4.4. I2S clock 956
20.4.5. Operation 957
20.4.6. DMA function 960
20.4.7. I2S interrupts 960
20.5. Register definition 962
20.5.1. Control register 0 (SPI_CTL0) 962
20.5.2. Control register 1 (SPI_CTL1) 964
20.5.3. Status register (SPI_STAT) 967
20.5.4. Data register (SPI_DATA) 969
20.5.5. CRC polynomial register (SPI_CRCPOLY) 969
20.5.6. Transmit CRC register (SPI_TCRC) 970
20.5.7. I2S control register (SPI_I2SCTL) 970
20.5.8. I2S clock prescaler register (SPI_I2SPSC) 972
20.5.9. Master request configuration register (SPI_MREQCFG) 973
20.5.10. Clock configuration register 0 (SPI_CKCFG0) 974
20.5.11. Clock configuration register 1 (SPI_CKCFG1) 975
20.5.12. Continuous control register (SPI_CCTL) 976
20.5.13. Quad-SPI mode control register (SPI_QCTL) 976
21. Quad-SPI interface (QSPI) 978
21.1. Overview 978
21.2. Characteristics 978
21.3. Function overview 978
21.3.1. QSPI block diagram 978
21.3.2. QSPI command format 979
21.3.3. QSPI signal line modes 981
21.3.4. QSPI DDR mode 981
21.3.5. DQS signal 982
21.3.6. CSN and SCK 982
21.4. Operating modes 982
21.4.1. Normal mode 983

21.4.2. Read polling mode 984
21.4.3. Memory map mode 984
21.5. QSPI configuration 985
21.5.1. Flash configuration 985
21.5.2. IP configuration 985
21.6. Send instruction only once 985
21.7. Error and interrupts 986
21.8. Clock phase delay module (CPDM) 986
21.8.1. Overview 986
21.8.2. Characteristics 987
21.8.3. Function overview 987
21.8.4. Operation process 988
21.9. QSPI register definition 990
21.9.1. Control register (QSPI_CTL) 990
21.9.2. Device configuration register (QSPI_DCFG) 993
21.9.3. Status register (QSPI_STAT) 994
21.9.4. Status clear register (QSPI_STATC) 995
21.9.5. Data length register (QSPI_DTLEN) 996
21.9.6. Transfer configuration register (QSPI_TCFG) 997
21.9.7. Address register (QSPI_ADDR) 999
21.9.8. Alternate bytes register (QSPI_ALTE) 999
21.9.9. Data register (QSPI_DATA) 1000
21.9.10. Status mask register (QSPI_STATMK) 1000
21.9.11. Status match register (QSPI_STATMATCH) 1001
21.9.12. Interval register (QSPI_INTERVAL) 1001
21.9.13. Timeout register (QSPI_TMOUT) 1001
21.9.14. FIFO flush register (QSPI_FLUSH) 1002
21.9.15. TMR register (QSPI_TMR) 1002
21.10. CPDM register definition 1003
21.10.1. Control register (CPDM_CTL) 1003
21.10.2. Configuration register (CPDM_CFG) 1004
22. Controller area network (CAN) 1006
22.1. Overview 1006
22.2. Characteristics 1007
22.3. Function overview 1007
22.3.1. Block diagram 1007
22.3.2. Modular CAN (M_CAN) cores 1009
22.3.3. DMU Add-On 1009
22.3.4. Power saving mode 1009
22.3.5. Share memory map 1010
22.3.6. Frame capacity 1010
22.3.7. CAN RAM Arbiter 1011
22.3.8. Interrupts 1012
22.4. Example for a typical configuration flow of CAN 1015
22.5. CAN cluster register definition 1016
22.5.1. CAN system register (CAN_SYSREG) 1018
22.6. Modular controller area network (M_CAN) 1025
22.6.1. Overview 1025
22.6.2. Characteristics 1025
22.6.3. Function overview 1026
22.6.4. Programmer's model 1051
22.6.5. Register definition 1060
22.7. DMA interface unit (DMU) 1106
22.7.1. Overview 1106
22.7.2. Function overview 1107
22.7.3. Register definition 1113

23. Comparator (CMP) 1124
23.1. Overview 1124
23.2. Characteristics 1124
23.3. Function overview 1124
23.3.1. CMP clock 1125
23.3.2. CMP I / O configuration 1125
23.3.3. CMP operating mode 1127
23.3.4. CMP hysteresis 1127
23.3.5. CMP register write protection 1127
23.3.6. CMP output blanking 1128
23.3.7. CMP voltage scaler function 1128
23.3.8. CMP interrupt 1128
23.3.9. CMP circular mode 1128
23.4. Register definition 1131
23.4.1. CMP0 Control / status register (CMP0_CS) 1131
23.4.2. CMP1 Control / status register (CMP1_CS) 1134
23.4.3. Round Robin control register(CMPx_RRCTL) 1137
23.4.4. Round Robin channel control Register (CMPx_RRCHCTL) 1138
23.4.5. Round Robin compare output register(CMPx_RRCOMO) 1139
23.4.6. Round Robin compare result flag register (CMPx_RRCOMF) 1140
23.4.7. Round Robin timer control register (CMPx_RRTCTL) 1141
23.4.8. Round Robin timer trigger value register (CMPx_RRTTRGV) 1142
23.4.9. Round Robin timer counter register (CMPx_RRTCNT) 1142
24. Self-test control module (STCM) 1144
24.1. Overview 1144
24.2. Characteristics 1144
24.3. Function overview 1144
24.3.1. System overview 1144
24.3.2. Block diagram 1145
24.3.3. Self-test process 1146
24.3.4. BIST partition mapping 1147
24.3.5. BIST sequence / BIST phase / MBIST execution 1149
24.3.6. Register access 1149
24.3.7. Clock and reset 1150
24.3.8. AHB interface 1150
24.3.9. Write protection mechanisms 1150
24.3.10. Watchdog mechanism 1151
24.3.11. BIST chain configuration 1152
24.3.12. Recoverable / Unrecoverable error 1152
24.3.13. FSM state machines 1152
24.3.14. Reset events and configurations while self-test 1152
24.3.15. Fault injection 1153
24.4. Register definition 1154
24.4.1. TMR control and status register (STCM_TMRCS) 1154
24.4.2. BIST running control and status register (STCM_BRCS) 1155
24.4.3. Write protection key register (STCM_WPK) 1155
24.4.4. Configuration register 0 (STCM_CFG0) 1156
24.4.5. MBIST watchdog timer register (STCM_MBWDGT) 1157
24.4.6. BIST error state register 0 (STCM_BESTAT0) 1157
24.4.7. BIST error state register 1 (STCM_BESTAT1) 1158
24.4.8. BIST error mapping register (STCM_BEM) 1159
24.4.9. BIST interval register (STCM_BINTVL) 1160
24.4.10. MBIST algorithm selection register (STCM_MBALGSEL) 1161
24.4.11. MBIST phase state register (STCM_MBPSTAT) 1163
24.4.12. MBIST phase end state register (STCM_MBPE_STAT) 1163
24.4.13. MBIST phase error mapping register (STCM_MBPEM) 1164

24.4.14. MBIST phase launch register (STCM_MBLAUNCH) 1164
24.4.15. MBISTx phase configuration register (STCM_MBxPCFG) 1165
25. Memory test controller (MTC) 1167
25.1. Overview 1167
25.2. Characteristics 1167
25.3. Function overview 1167
25.3.1. Block diagram 1167
25.3.2. Error injection enable 1168
25.3.3. Single or multi bit error injection 1168
25.3.4. ECC data injection 1169
25.3.5. ECC address injection 1170
25.3.6. EDC data injection 1171
25.3.7. EDC after ECC injection 1172
25.3.8. Multi-channel parallel test 1173
25.3.9. Memory test flow 1173
25.4. Register definition 1175
25.4.1. Control register 0 (MTC_CTL0) 1175
25.4.2. Channel control register 0 (MTC_CHCTL0) 1176
25.4.3. Channel control register 1 (MTC_CHCTL1) 1176
25.4.4. Control register 1 (MTC_CTL1) 1177
25.4.5. Error bits generate configuration register x (MTC_EBGCFGx) (x = 0…4) 1177
25.4.6. MTC Triple Modular Redundancy enable register (MTC_TMREN) 1178
25.4.7. MTC Triple Modular Redundancy Status register (MTC_TMRSTAT) 1178
25.4.8. MTC Triple Modular Redundancy Flag Clear register (MTC_TMRFC) 1179
26. Interrupt requests management (IRM) 1179
26.1. Overview 1179
26.2. Characteristics 1179
26.3. Block Diagram 1181
26.4. Function overview 1182
26.5. IRM registers 1191
26.5.1. Control register (IRM_CTL) 1191
26.5.2. Interrupt response number register (IRM_IRN) 1191
26.5.3. Interrupt request number select register (IRM_IRQNSELx) (x=0,1,2,3) 1192
26.5.4. Clock threshold register (IRM_CKTHx) (x=0,1,2,3) 1192
26.5.5. Clock counter register (IRM_CKCNTx) (x=0,1,2,3) 1193
26.5.6. Comparison output register (IRM_CMPOx) (x=0,1,2,3) 1193
26.5.7. Routing control register (IRM_RCTLx) (x=0..239) 1194
27. Client configuration (CLTCFG) 1195
27.1. Overview 1195
27.2. Characteristics 1195
27.3. Function overview 1195
27.4. Register definition 1196
27.4.1. System reset status register 1 (CLTCFG_SR_STAT1) 1196
27.4.2. System reset status register 2 (CLTCFG_SR_STAT2) 1196
27.4.3. Subsystem reset status register 1 (CLTCFG_SUBSR_STAT1) 1197
27.4.4. Subsystem reset status register 2 (CLTCFG_SUBSR_STAT2) 1198
27.4.5. Power-on reset control register 0 (CLTCFG_POR_CTL0) 1199
27.4.6. Power-on reset control register 1 (CLTCFG_POR_CTL1) 1199
27.4.7. System reset control register 1 (CLTCFG_SR_CTL1) 1200
27.4.8. Subsystem reset control register 0 (CLTCFG_SUBSR_CTL0) 1200
27.4.9. Subsystem reset control register 1 (CLTCFG_SUBSR_CTL1) 1201
27.4.10. Subsystem reset control register 2 (CLTCFG_SUBSR_CTL2) 1202
27.4.11. Triple modular redundancy control register (CLTCFG_TMR_CTL) 1203
27.4.12. Triple modular redundancy status register (CLTCFG_TMR_STAT) 1203
27.4.13. Triple modular redundancy flag clear register (CLTCFG_TMR_FC) 1204
27.4.14. Life cycle control and status register (CLTCFG_LC_CS) 1205

28. Hardware Security Module (HSM) 1208
28.1. Overview 1208
28.2. Characteristics 1208
28.3. Function overview 1208
28.3.1. HSM system 1208
28.3.2. HSM RCU 1210
28.3.3. HSM Power management 1210
28.3.4. HSM interface(HSMIF) 1211
28.3.5. HSM memory 1211
28.3.6. HSM timers 1211
28.3.7. HSM DMA 1211
28.3.8. HSM Debug 1212
29. Fault management unit (FMU) 1213
29.1. Overview 1213
29.2. Characteristics 1213
29.3. Function description 1214
29.3.1. Block diagram 1214
29.3.2. FMU clock 1214
29.3.3. FMU fault grouping 1214
29.3.4. Fault reactions 1218
29.3.5. FMU behavior monitor unit 1218
29.3.6. Fault reaction programming 1219
29.3.7. Recovery definitions 1220
29.3.8. FMSM description 1221
29.3.9. Principles of fault reactions 1223
29.3.10. Fault recovery 1223
29.3.11. Fatal fault pin signal 1225
29.3.12. FMU initialization preprogramming 1227
29.3.13. FMU initialization programming 1229
29.3.14. Transform FMU to NORMAL work mode 1229
29.3.15. Manage software recoverable faults 1230
29.3.16. Hold status clean 1230
29.3.17. Execute operations 1231
29.4. Register definition 1232
29.4.1. Control register (FMU_CTL) 1233
29.4.2. Control key register (FMU_CTLKEY) 1234
29.4.3. Configuration register (FMU_CFG) 1235
29.4.4. FG configuration register (FMU_FGCFG) 1237
29.4.5. FG subsystem reset configuration register (FMU_FGSUBSRCFG) 1238
29.4.6. FG status register (FMU_FGSTAT) 1239
29.4.7. FG key register (FMU_FGKEY) 1240
29.4.8. FG enable register (FMU_FGEN) 1240
29.4.9. FG CAUTION work mode timeout enable register (FMU_FGCMTEN) 1241
29.4.10. FG CAUTION work mode timeout value register (FMU_FGCMTVAL) 1241
29.4.11. INIT work mode timeout value register (FMU_IMTVAL) 1242
29.4.12. Fault pin control register (FMU_FPCTL) 1243
29.4.13. Status register (FMU_STAT) 1244
29.4.14. NORMAL transform to CAUTION work mode hold status register (FMU_NTCHSTAT) 1245
29.4.15. CAUTION transform to FAULT work mode hold status register (FMU_CTFHSTAT) 1246
29.4.16. NORMAL transform to FAULT work mode hold status register (FMU_NTFHSTAT) 1246
29.4.17. FAULT transform to CAUTION work mode hold status register (FMU_FTCHSTAT) 1247
29.4.18. Fake FG fault number configuration register (FMU_FFGNCFG) 1247
29.4.19. Interrupt request pending register (FMU_INTRPD) 1248
29.4.20. INIT work mode timeout interrupt request enable register (FMU_IMTINTREN) 1249
29.4.21. Temporary configuration lock register (FMU_TCLOCK) 1249
29.4.22. Permanent configuration lock register (FMU_PCLOCK) 1250
29.4.23. Fault output timer register (FMU_FOT) 1250

29.4.24. FG CAUTION work mode interrupt request enable register (FMU_CMINTREN) 1251
29.4.25. FG FAULT work mode NMI interrupt request enable register (FMU_FMNMIINTREN) 1251
29.4.26. FG FFIO pin output enable register (FMU_FPOEN) 1252
29.4.27. FG GFIO pin output enable register (FMU_GPOEN) 1253
29.4.28. Fault holding timer output enable register (FMU_FHTOEN) 1253
29.4.29. FG CAUTION work mode counter register (FMU_CMCNT) 1254
29.4.30. FG INIT work mode counter register (FMU_IMCNT) 1254
29.4.31. Fatal fault output counter register (FMU_FOCNT) 1254
29.4.32. General fault output counter register (FMU_GOCNT) 1255
29.4.33. FG trigger source enable register (FMU_FGTSENx) (x=0...4) 1255
29.4.34. FG trigger source status register (FMU_FGTSSTATx) (x=0...4) 1256
29.4.35. FG software triggered fault enable register (FMU_FGSWTENx) (x=0...2) 1256
29.4.36. TMR enable and status register (FMU_TMRES) 1257
30. Debug system (DBGSYS) 1259
30.1. Overview 1259
30.2. Characteristics 1259
30.3. DAP architecture 1259
30.4. Debug 1260
30.4.1. Test access port(TAP) 1260
30.4.2. Debug access port(DAP) 1261
30.4.3. Application cores debugging from first instruction 1262
30.4.4. Debug authentication 1263
30.4.5. Debug in low-power mode 1263
30.4.6. Debug resets 1264
30.4.7. Debug across device lifecycles 1265
30.5. JTAG controller(JTAGC) 1265
30.5.1. Overview 1265
30.5.2. JTAGC architecture 1266
30.5.3. JTAGC IR codes 1267
30.5.4. JTAGC reset and bypass mode 1267
30.5.5. TAP controller state machine 1267
30.5.6. JTAGC signal description 1268
30.6. JTAG Data Communication(JDC) 1268
30.6.1. Overview 1268
30.6.2. JDC architecture 1269
30.6.3. JDC functional overview 1269
30.6.4. JDC Signal description 1270
30.6.5. JDC instruction 1271
30.6.6. JDC secure challenge/response connection procedure 1271
30.7. Trace 1272
30.7.1. Overview 1272
30.7.2. Trace architecture 1272
30.7.3. TPIU 1273
30.8. Embedded cross trigger (ECT) 1273
30.9. GD_AP register definition 1275
30.9.1. Authentication status register (GD_AP_AUTH_STAT) 1275
30.9.2. Authentication control register (GD_AP_AUTH_CTL) 1276
30.9.3. Status register (GD_AP_STAT) 1277
30.9.4. Control register (GD_AP_CTL) 1278
30.9.5. Key challenge register x (GD_AP_KEYCHLGx) (x = 0...3) 1279
30.9.6. Low power control register 0(GD_AP_LPCTL0) 1280
30.9.7. Low power control register 1(GD_AP_LPCTL1) 1280
30.9.8. Key response register x(GD_AP_KEYRESPx) (x = 0...3) 1281
30.9.9. Debug enable register (GD_AP_DBGEN) 1281
30.9.10. Reset register (GD_AP_RST) 1282
30.9.11. ID code register (GD_AP_ID) 1283

30.10. JDC register definition 1284
30.10.1. Interrupt enable register (JDC_INTEN) 1284
30.10.2. Status register (JDC_STAT) 1284
30.10.3. JOUT APB data register (JDC_JOUT_APB_DATA) 1285
30.10.4. JIN APB data register (JDC_JIN_APB_DATA) 1286
30.10.5. JOUT data register (JDC_JOUT_DATA) 1286
30.10.6. JIN data register (JDC_JIN_DATA) 1286
31. Appendix 1288
31.1. List of abbreviations used in register 1288
31.2. List of terms 1288
31.3. Available peripherals 1289
32. Revision history 1290

List of Figures

Figure 1-1. The structure of the Cortex®-M7 processor 33
Figure 1-2. The system architecture of GD32A711/A712 devices 35
Figure 1-3. The architecture of core wrapper and bus wrapper of GD32A711/A712 devices 36
Figure 1-4. High-level boot sequence flowchart 44
Figure 1-5. Nonsecure boot sequence flowchart 47
Figure 1-6. Host recovery mode sequence flowchart 49
Figure 1-7. Sequence flowchart of lifecycle configuration 50
Figure 1-8. The block diagram of EXTMPU 58
Figure 1-9. The block diagram of IMICU for BM1 master write 63
Figure 1-10. The block diagram of IMICU for BM1 master read 63
Figure 1-11. The block diagram of IMICU for BM2 master write 64
Figure 1-12. The block diagram of IMICU for BM2 master read 64
Figure 2-1. FMC block diagram of GD32A711 / A712 devices 150
Figure 2-2. Buffer0 / Buffer2 / Buffer3 architecture of GD32A711 / A712 devices 157
Figure 2-3. Process of page erase operation 160
Figure 2-4. Process of mass erase operation 162
Figure 2-5. Process of word program operation 164
Figure 2-6. Process of fast programming operation 166
Figure 3-1. GD32A711 /A712 series power supply overview 213
Figure 5-1. Chip reset procedure 240
Figure 5-2. The chip startup sequence 247
Figure 5-3. The chip reset timing 247
Figure 5-4. Clock tree 250
Figure 5-5. HXTAL clock source 252
Figure 5-6. HXTAL clock source in bypass mode 252
Figure 6-1. Block diagram of EXTI 325
Figure 7-1. TRIGSEL main composition example 344
Figure 8-1. Basic structure of a standard I/O port bit 389
Figure 8-2. Input configuration 391
Figure 8-3. Output configuration 391
Figure 8-4. Analog configuration 392
Figure 8-5. Alternate function configuration 393
Figure 9-1. Block diagram of CRC calculation unit 421
Figure 10-1. DMA block diagram 427
Figure 10-2. Data stream for three transfer modes 428
Figure 10-3. Handshake mechanism 429
Figure 10-4. Data packing/unpacking when PWIDTH = '00' 435
Figure 10-5. Data packing/unpacking when PWIDTH = '01' 435
Figure 10-6. Data packing/unpacking when PWIDTH = '10' 436
Figure 10-7. Data packing/unpacking when PWIDTH = '11' 436
Figure 10-8. DMA operation of switch-buffer mode 438
Figure 10-9. System connection of DMA0 and DMA1 445
Figure 10-10. DMA channel x transfer timeout monitor interrupt logic 446
Figure 11-1. DMAMUX block diagram 468
Figure 11-2. Synchronization mode 470
Figure 11-3. Event generation 471
Figure 13-1. ADC module block diagram 497
Figure 13-2. Restart procedure, RRESTART=1, RL=3, IL=1 499
Figure 13-3. Resume procedure, RRESTART=0, RL=3, IL=1 500
Figure 13-4. Abort the routine sequence, RMODE=0 502
Figure 13-5. Abort the routine sequence, RMODE=1 502
Figure 13-6. Presampling and sampling sequence, PSMPEN = 1, DISSMP=0 503
Figure 13-7. Presampling and sampling sequence, PSMPEN = 1, DISSMP=1 503

Figure 13-8. Schematic diagram of analog multiplexing channel EECIN0 504
Figure 13-9. 12-bit Data storage mode 508
Figure 13-10. 10-bit Data storage mode 509
Figure 13-11. 8-bit Data storage mode 509
Figure 13-12. 6-bit Data storage mode 509
Figure 13-13. ADC sync block diagram 513
Figure 13-14. Routine parallel mode on 16 channels 513
Figure 13-15. Inserted parallel mode on 4 channels 514
Figure 13-16. Follow-up mode on 1 channel in continuous mode 515
Figure 13-17. Trigger rotation: inserted sequence 516
Figure 13-18. Combined routine parallel & trigger rotation mode: SYNCM = 4'b0010 517
Figure 13-19. Trigger occurs during inserted sequence conversion: SYNCM = 4'b0010 517
Figure 13-20. Capacitive self test in one-shot mode 519
Figure 13-21. Supply + capacitive self test in continuous mode 521
Figure 13-22. Watchdog timer for supply self test 523
Figure 13-23. Watchdog timer for capacitive self test 523
Figure 13-24. Watchdog timer for supply and capacitive self test 523
Figure 14-1. DAC block diagram 574
Figure 14-2. DAC LFSR algorithm 576
Figure 14-3. DAC triangle noise wave 577
Figure 14-4. DAC-ADC Loop test diagram 578
Figure 15-1. Free watchdog block diagram 587
Figure 15-2. Window watchdog timer block diagram 594
Figure 15-3. Window watchdog timing diagram 595
Figure 16-1. Block diagram of RTC 602
Figure 16-2. RTC second and alarm waveform example (RTC_PSC = 3, RTC_ALRM = 2) 603
Figure 16-3. RTC second and overflow waveform example (RTC_PSC= 3) 604
Figure 17-1. Advanced timer block diagram 614
Figure 17-2. Timing chart of internal clock divided by 1 615
Figure 17-3. Timing chart of PSC value change from 0 to 2 616
Figure 17-4. Timing chart of up counting mode, PSC=0/2 617
Figure 17-5. Timing chart of up counting mode, change TIMERx_CAR on the go 618
Figure 17-6. Timing chart of down counting mode, PSC=0/2 619
Figure 17-7. Timing chart of down counting mode, change TIMERx_CAR on the go 619
Figure 17-8. Timing chart of center-aligned counting mode 621
Figure 17-9. Repetition counter timing chart of center-aligned counting mode 622
Figure 17-10. Repetition counter timing chart of up counting mode 622
Figure 17-11. Repetition counter timing chart of down counting mode 623
Figure 17-12. Channel 0 input capture principle 624
Figure 17-13. Multi mode channel 0 input capture principle 625
Figure 17-14. Channel output compare principle (when MCHxMSEL = 2'00, x=0, 1, 2, 3) 627
Figure 17-15. Channel output compare principle (when MCHxMSEL = 2'01, x=0, 1, 2, 3) 627
Figure 17-16. Channel output compare principle (with complementary output when MCHxMSEL = 2'11, x=0,1,2,3) 627
Figure 17-17. Output-compare under three modes 629
Figure 17-18. EAPWM timechart 630
Figure 17-19. CAPWM timechart 630
Figure 17-20. Channel x output PWM with (CHxVAL < CHxVAL_ADD) 632
Figure 17-21. Channel x output PWM with (CHxVAL = CHxVAL_ADD) 632
Figure 17-22. Channel x output PWM with (CHxVAL > CHxVAL_ADD) 633
Figure 17-23. Channel x output PWM with CHxVAL or CHxVAL_ADD exceeds CARL 633
Figure 17-24. Channel x output PWM duty cycle changing with CHxVAL_ADD 634
Figure 17-25. Four Channels outputs in composite PWM mode 634
Figure 17-26. Channel x output PWM with(CHxVAL < CHxVAL_ADD / CHxVAL_ADD < CHxVAL) 636
Figure 17-27. Channel x output PWM with(CHxVAL = CHxVAL_ADD) 638
Figure 17-28. Channel x output PWM with CHxVAL or CHxVAL_ADD exceeds CARL 638
Figure 17-29. CHx_O output with a pulse in edge-aligned mode (CHxOMPSEL!=2'b00) 639

Figure 17-30. CHx_O output with a pulse in center-aligned mode (CHxOMPSEL!=2'b00) 640
Figure 17-31. Channel output complementary PWM with dead-time insertion 645
Figure 17-32. Dead time shadow function 645
Figure 17-33. Break function diagram 646
Figure 17-34. Output behavior of the channel in response to a break (the break high active) 647
Figure 17-35. Example of counter operation in decoder interface mode(Quadrature decoder mode 2) 648
Figure 17-36. Example of decoder interface mode with CI0FE0 polarity inverted(Quadrature decoder mode 2) 649
Figure 17-37. Quadrature decoder signal disconnection detection block diagram 650
Figure 17-38. Example of counter operation in decoder mode 0 / 1 with CH1P=0 651
Figure 17-39. Example of counter operation in decoder mode 2 / 3 (CH0P / CH1P=0) 652
Figure 17-40. Hall sensor is used for BLDC motor 653
Figure 17-41. Hall sensor timing between two timers 654
Figure 17-42. Restart mode 656
Figure 17-43. Pause mode 656
Figure 17-44. Event mode 657
Figure 17-45. Single pulse mode TIMERx_CHxCV=0x04, TIMERx_CAR=0x99 658
Figure 17-46. TIMER0 master/slave mode example 659
Figure 17-47. Triggering TIMER0 with enable signal of TIMER1 660
Figure 17-48. Triggering TIMER0 and TIMER1 with TIMER1's CI0 input 661
Figure 17-49. Triggering four timers by trigger out of basic timer 662
Figure 17-50. Counting direction in CINITDIR is 0 or 1 662
Figure 17-51. General Level 0 timer block diagram 741
Figure 17-52. Timing chart of internal clock divided by 1 742
Figure 17-53. Timing chart of PSC value change from 0 to 2 743
Figure 17-54. Timing chart of up counting mode, PSC=0/2 744
Figure 17-55. Timing chart of up counting, change TIMERx_CAR on the go 744
Figure 17-56. Timing chart of down counting mode, PSC=0/2 745
Figure 17-57. Timing chart of down counting mode, change TIMERx_CAR on the go 746
Figure 17-58. Timing chart of center-aligned counting mode 747
Figure 17-59. Channels input capture principle 748
Figure 17-60. Channel output compare principle (x=0,1,2,3) 749
Figure 17-61. Output-compare under three modes 750
Figure 17-62. EAPWM timechart 751
Figure 17-63. CAPWM timechart 752
Figure 17-64. Channel x output PWM with (CHxVAL < CHxVAL_ADD) 754
Figure 17-65. Channel x output PWM with (CHxVAL = CHxVAL_ADD) 754
Figure 17-66. Channel x output PWM with (CHxVAL > CHxVAL_ADD) 754
Figure 17-67. Channel x output PWM with CHxVAL or CHxVAL_ADD exceeds CARL 755
Figure 17-68. Channel x output PWM duty cycle changing with CHxVAL_ADD 755
Figure 17-69. Four Channels outputs in composite PWM mode 756
Figure 17-70. Channel x output PWM with(CHxVAL < CHxVAL_ADD / CHxVAL_ADD < CHxVAL) 758
Figure 17-71. Channel x output PWM with(CHxVAL = CHxVAL_ADD) 759
Figure 17-72. Channel x output PWM with CHxVAL or CHxVAL_ADD exceeds CARL 760
Figure 17-73. Restart mode 762
Figure 17-74. Pause mode 763
Figure 17-75. Event mode 763
Figure 17-76. Single pulse mode TIMERx_CHxCV = 0x04, TIMERx_CAR=0x99 764
Figure 17-77. Basic timer block diagram 805
Figure 17-78. Timing chart of internal clock divided by 1 for channel 0 806
Figure 17-79. Timing chart of PSC value change from 0 to 2 for channel 0 806
Figure 17-80. Timing chart of up counting mode, PSC=0/2 for channel 0 807
Figure 17-81. Timing chart of up counting mode, change TIMERx_CAR on the go for channel 0 807
Figure 18-1 Example system using LINFlexD 821
Figure 18-2 LIN frames format 823

Figure 18-3 Structure of LIN frame 823
Figure 18-4 Structure of byte field 824
Figure 18-5 Structure of byte field 824
Figure 18-6 Sync byte field 824
Figure 18-7 Identifier field 824
Figure 18-8 Operating modes 825
Figure 18-9 LINFLEXD in loop back mode 826
Figure 18-10 Incomplete response (for example, missing checksum) 831
Figure 18-11 No response 831
Figure 18-12 IDENTIFIER LIST MODE 834
Figure 18-13 IDENTIFIER MASK MODE 834
Figure 18-14. Start detection and sampling for oversampling rate = 4 (case 1) 837
Figure 18-15. Start detection and sampling for oversampling rate = 4 (case 2A) 837
Figure 18-16. Start detection and sampling for oversampling rate = 4 (case 2B) 838
Figure 18-17. Start detection and sampling for oversampling rate = 4 (case 3) 838
Figure 18-18. Start detection and sampling for oversampling rate = 4 (case 4) 838
Figure 18-19. Start detection and sampling for oversampling rate = 4 (case 5) 839
Figure 18-20. Wakeup sequence 840
Figure 18-21 UART mode 8-bit data frame 841
Figure 18-22 UART mode 9-bit data frame 841
Figure 18-23 UART mode 13-bit data frame 842
Figure 18-24 Structure of 8-byte buffer 842
Figure 18-25 Interrupt diagram 845
Figure 18-26 Faster receiver 850
Figure 18-27 Slower receiver 850
Figure 19-1. I2C module block diagram 887
Figure 19-2. Data validation 888
Figure 19-3. START and STOP signal 888
Figure 19-4. I2C communication flow with 10-bit address (Master Transmit) 889
Figure 19-5. I2C communication flow with 7-bit address (Master Transmit) 889
Figure 19-6. I2C communication flow with 7-bit address (Master Receive) 889
Figure 19-7. I2C communication flow with 10-bit address (Master Receive when HEAD10R=0) 890
Figure 19-8. I2C communication flow with 10-bit address (Master Receive when HEAD10R=1) 890
Figure 19-9. Data hold time 891
Figure 19-10. Data setup time 891
Figure 19-11. Data transmission 893
Figure 19-12. Data reception 894
Figure 19-13. I2C initialization in slave mode 897
Figure 19-14. Programming model for slave transmitting when SS=0 898
Figure 19-15. Programming model for slave transmitting when SS=1 899
Figure 19-16. Programming model for slave receiving 900
Figure 19-17. I2C initialization in master mode 901
Figure 19-18. Programming model for master transmitting (N<=255) 902
Figure 19-19. Programming model for master transmitting (N>255) 903
Figure 19-20. Programming model for master receiving (N<=255) 904
Figure 19-21. Programming model for master receiving (N>255) 905
Figure 19-22. SMBus Master Transmitter and Slave Receiver communication flow 909
Figure 19-23. SMBus Master Receiver and Slave Transmitter communication flow 909
Figure 20-1. Block diagram of SPI 927
Figure 20-2. Block diagram of SPI clock sources 928
Figure 20-3. SPI timing diagram in normal mode 930
Figure 20-4. SPI timing diagram in Quad-SPI mode (CKPL=1, CKPH=1, LF=0, NSSPOL=0) 930
Figure 20-5. SPI master receive delay diagram 931
Figure 20-6. Transmission and reception FIFO 931
Figure 20-7. A typical full-duplex connection 935
Figure 20-8. A typical simplex connection (Master: Receive, Slave: Transmit) 936

Figure 20-9. A typical simplex connection (Master: Transmit only, Slave: Receive) 936
Figure 20-10. A typical bidirectional connection 936
Figure 20-11. Timing diagram of TI master mode with discontinuous transfer 938
Figure 20-12. Timing diagram of TI master mode with continuous transfer 939
Figure 20-13. Timing diagram of TI slave mode 939
Figure 20-14. Timing diagram of NSS pulse with continuous transmit (NSSNSS = 0, NSSSCK = 0, SCKNSS = 0) 940
Figure 20-15. Timing diagram of single-frame mode with continuous transmit (NSSNSS = 0, NSSSCK = 0, SCKNSS = 0) 940
Figure 20-16. Timing diagram of continuous-frame mode with continuous transmit (NSSNSS = 0, NSSSCK = 0, SCKNSS = 0) 941
Figure 20-17. Timing diagram of quad write operation in Quad-SPI mode 942
Figure 20-18. Timing diagram of quad read operation in Quad-SPI mode 943
Figure 20-19. Block diagram of I2S 948
Figure 20-20. I2S Philips standard timing diagram (DTLEN=00, CHLEN=0, CKPL=0) 949
Figure 20-21. I2S Philips standard timing diagram (DTLEN=00, CHLEN=0, CKPL=1) 949
Figure 20-22. I2S Philips standard timing diagram (DTLEN=10, CHLEN=1, CKPL=0) 950
Figure 20-23. I2S Philips standard timing diagram (DTLEN=10, CHLEN=1, CKPL=1) 950
Figure 20-24. I2S Philips standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0) 950
Figure 20-25. I2S Philips standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1) 950
Figure 20-26. I2S Philips standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0) 950
Figure 20-27. I2S Philips standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1) 950
Figure 20-28. MSB justified standard timing diagram (DTLEN=00, CHLEN=0, CKPL=0) 951
Figure 20-29. MSB justified standard timing diagram (DTLEN=00, CHLEN=0, CKPL=1) 951
Figure 20-30. MSB justified standard timing diagram (DTLEN=10, CHLEN=1, CKPL=0) 951
Figure 20-31. MSB justified standard timing diagram (DTLEN=10, CHLEN=1, CKPL=1) 951
Figure 20-32. MSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0) 951
Figure 20-33. MSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1) 951
Figure 20-34. MSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0) 952
Figure 20-35. MSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1) 952
Figure 20-36. LSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0) 952
Figure 20-37. LSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1) 952
Figure 20-38. LSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0) 952
Figure 20-39. LSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1) 953
Figure 20-40. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=0) 953
Figure 20-41. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=1) 953
Figure 20-42. PCM standard short frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=0) 953
Figure 20-43. PCM standard short frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=1) 953
Figure 20-44. PCM standard short frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=0) 954
Figure 20-45. PCM standard short frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=1) 954
Figure 20-46. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=0) 954
Figure 20-47. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=1) 954
Figure 20-48. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=0) 954
Figure 20-49. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=1) 955
Figure 20-50. PCM standard long frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=0) 955
Figure 20-51. PCM standard long frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=1) 955

Figure 20-52. PCM standard long frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=0) 955
Figure 20-53. PCM standard long frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=1) 955
Figure 20-54. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=0) 956
Figure 20-55. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=1) 956
Figure 20-56. Block diagram of I2S clock generator 956
Figure 20-57. I2S initialization sequence 958
Figure 21-1 QSPI diagram 979
Figure 21-2 QSPI command format 980
Figure 21-3 QSPI DDR mode 981
Figure 21-4 DDR read timing using DQS 982
Figure 21-5. Schematic diagram of CPDM 987
Figure 21-6. CPDM delay line length configuration flowchart 988
Figure 21-7. CPDM output clock phase configuration flowchart 989
Figure 22-1. CAN cluster diagram 1008
Figure 22-2. CAN cluster clock structure diagram 1008
Figure 22-3. M_CAN block diagram 1026
Figure 22-4. Transmitter delay measurement 1032
Figure 22-5. Pin control in bus monitoring mode 1034
Figure 22-6. Pin control in loop back modes 1036
Figure 22-7. Standard message ID filter path 1040
Figure 22-8. Extended message ID filter path 1041
Figure 22-9. Rx FIFO status 1042
Figure 22-10. Rx FIFO overflow handling 1043
Figure 22-11. Example of mixed configuration dedicated Tx buffers / Tx FIFO 1048
Figure 22-12. Example of mixed configuration dedicated Tx Buffers / Tx Queue 1049
Figure 22-13. Message RAM configuration 1051
Figure 22-14. Address map 1107
Figure 22-15. DMU integration 1108
Figure 22-16. DMU timing 1112
Figure 23-1. CMP block diagram 1125
Figure 23-2. CMP hysteresis 1127
Figure 23-3. The CMP outputs signal blanking 1128
Figure 23-4. Round Robin diagram 1129
Figure 24-1. STCM system diagram of GD32A711 / A712 devices 1144
Figure 24-2. STCM block diagram of GD32A711 / A712 devices 1145
Figure 24-3. Self-test process 1146
Figure 25-1. MTC block diagram 1168
Figure 26-1.block diagram for GD32A711/A712 devices 1181
Figure 28-1. HSM system architecture 1208
Figure 29-1. FMU module block diagram 1214
Figure 29-2. FMU works mode and transition 1221
Figure 29-3. FG fault hardware recovery 1223
Figure 29-4. FG fault software recovery 1224
Figure 29-5. FG fault CAUTION to FAULT work mode recovery 1224
Figure 30-1. Debug access port architecture of GD32A711 / A712 1260
Figure 30-2. TAP connectivity block diagram 1260
Figure 30-3. Debugging from first instruction sequence 1262
Figure 30-4. JTAG Controller block 1266
Figure 30-5. Data transfer shift 1266
Figure 30-6. TAP controller state machine 1268
Figure 30-7. Block diagram of JDC 1269
Figure 30-8. Block diagram of trace for GD32A711 / A712 devices 1273
Figure 30-9. CTM and CTI interconnection diagram of GD32A711 / A712 1274

List of Tables

Table 1-1. Memory map of GD32A711/A712 devices 37
Table 1-2. SRAM size and width of GD32A711/A712 devices 43
Table 1-3. Descriptions of IVT of GD32A711/A712 devices 45
Table 1-4. Descriptions of BCW of GD32A711/A712 devices 46
Table 1-5. Valid values for lifecycle advancement 46
Table 1-6. host to HSM register (Offset 0x10) 51
Table 1-7. IN_FIELD_RECOVER_MODE-0x1FFFF950 52
Table 1-8. HSM respond host register (Offset 0x20) 52
Table 1-9. HSM status register (Offset 0x40) 53
Table 1-10. The interconnection relationship of the AHB_BM1 of GD32A711/A712 devices 54
Table 1-11. The interconnection relationship of the AHB_BM2 of GD32A711/A712 devices 54
Table 1-12. Master ID of GD32A711/A712 devices 55
Table 1-13. Slaver ID of GD32A711/A712 devices 56
Table 1-14. The memory map of the unlock key and corresponding peripheral registers of GD32A711/A712 devices 59
Table 1-15. Hexadecimal EDC code 65
Table 1-16. SYSCFG register overview 66
Table 1-17. Internal connection of TIMER 83
Table 1-18. BUSIM register overview 108
Table 1-19. EXTMPU register overview 118
Table 1-20. PRPU register overview 124
Table 1-21. IMICU register overview 125
Table 1-22. SRAM register overview 130
Table 1-23. TCM register overview 137
Table 2-1. Base address and size for flash memory of GD32A711 / A712 devices 150
Table 2-2. The relation between WSCNT and AHB clock frequency when LDO is 1.1V 155
Table 2-3. Master Buffer / Prefetch enable bit allocation of GD32A711 / A712 devices 157
Table 2-4. HOST option bytes of GD32A711 / A712 devices 169
Table 2-5. OB_CLTCFG option bytes fields description 172
Table 2-6. HSM Option bytes 174
Table 2-7. OB_BK0WP bit for pages protected 175
Table 2-8. OB_BK1WP bit for pages protected 175
Table 2-9. OB_DFWP bit for pages protected 176
Table 2-10. OB_HSMPFWP bit for protected 176
Table 2-11. OB_HSMDFWP bit for protected 177
Table 2-12. PGSERR conditions 178
Table 2-13. PGERR conditions 179
Table 2-14. WPERR conditions 179
Table 2-15. Life cycle slots status 180
Table 2-16. Resulting Life cycle 180
Table 2-17. Logic address of slots 180
Table 2-18. Logic address of application debug 181
Table 2-19. Logic address of Cryptography key 182
Table 2-20. Logic address of security protection key 182
Table 2-21. Logic address of HSM configuration 183
Table 3-1. General power saving mode summary 215
Table 4-1. BKP register overview 236
Table 5-1. POR reset modules (For GD32A711/A712 devices) 241
Table 5-2. System reset sources 243
Table 5-3. Subsystem reset sources (For GD32A711/A712 devices) 244
Table 5-4. Details of POR_WDG monitoring 248
Table 5-5. CK_OUT0 Clock source select 255
Table 5-6. CK_OUT1 Clock source select 255

Table 5-7. CKFM0 interrupt/reset configuration description 257
Table 6-1. GD32A711/A712 EXTI source 327
Table 6-2. EXTI register overview 330
Table 7-1. Trigger input bit fields selection 344
Table 7-2. TRIGSEL input and output mapping 348
Table 7-3. TRIGSEL register overview 352
Table 8-1. GPIO configuration table 388
Table 8-2. Standby pins 394
Table 8-3. Pin state on power up condition of the MCU 395
Table 8-4. Pin state when any system reset is triggered 395
Table 8-5. Pin behavior when subsystem reset including subsystem entry sequence 396
Table 8-6. Pin behavior when selftest (ERPAETEN bit in RCU_ERPACFG register configured as 1) 396
Table 8-7. Pin behavior when selftest (ERPAETEN bit in RCU_ERPACFG register and ST_FINISH_RPAEN bit in RCU_SUBSRPACTL register are configured as 0) 396
Table 8-8. Pin behavior when FMU fault in run mode when the fault reaction is not reset 397
Table 8-9. Pin behavior when standby (PAD_RET_CFG bit field in CTLCFG_SUBSR_CTL0 register must be configured as 1) 398
Table 8-10. GPIO register overview 399
Table 8-10. Pin type(1) 404
Table 9-1. CRC register overview 423
Table 10-1. Transfer mode 428
Table 10-2. CNT configuration 431
Table 10-3. FIFO counter critical value configuration rules for channel 3~7 432
Table 10-4. FIFO counter critical value configuration rules for channel 0~2 433
Table 10-5. DMA interrupt events 442
Table 10-6. DMA register overview 448
Table 11-1. Interrupt events 473
Table 11-2. Request multiplexer input mapping 474
Table 11-3. Trigger input mapping 480
Table 11-4. Synchronization input mapping 481
Table 11-5. DMAMUX register overview 482
Table 12-1. Pin assignment 487
Table 12-2. DBG register overview 489
Table 13-1. ADC internal signals 494
Table 13-2. ADC pins definition 494
Table 13-3. ADC channel remapping 494
Table 13-4. ADC Analog multiplexing channel signal 504
Table 13-5. Algorithm between raw converted data and offset 508
Table 13-6. Trigger source for routine channels 510
Table 13-7. Trigger source for inserted channels 510
Table 13-8. t_{CONV} timings depending on resolution 511
Table 14-1. DAC I / O description 574
Table 14-2. DAC triggers and outputs summary 574
Table 14-3. Triggers of DAC 575
Table 15-1. Min/max FWDGT timeout period at 40k / 32.768kHz (IRC40K / LXTAL) 589
Table 15-2. Min-max timeout value at 80 MHz (f_{WWDGT_CLK}) 596
Table 16-1. RTC register overview 605
Table 17-1. Timers (Timer) are divided into three sorts 611
Table 17-2. Advanced timer channel description 614
Table 17-3.The composite PWM pulse width 631
Table 17-4.The asymmetric PWM pulse width 636
Table 17-5. Complementary outputs controlled by parameters (MCHxMSEL =2'b11) 642
Table 17-6. Counting direction in different quadrature decoder signals 648
Table 17-7. the counter operation in decoder mode 1 650
Table 17-8. the counter operation in decoder mode 2 / 3 652
Table 17-9. Examples of slave mode 654
Table 17-10. TIMERx register overview 664

Table 17-11.The composite PWM pulse width 753
Table 17-12.The asymmetric PWM pulse width 757
Table 17-13. Examples of slave mode 761
Table 17-14. TIMERx register overview 766
Table 17-15. TIMERx register overview 809
Table 18-1. Start detection and delimiter detection in receiver 835
Table 18-2. BDRL access in UART mode 843
Table 18-3. BDRM access in UART mode 843
Table 18-4. Interrupts 845
Table 18-5. LIN master node transmit dma transfers 847
Table 18-6. LIN master node receive dma transfer 848
Table 18-7. LIN slave node transmit dma transfers 848
Table 18-8. LIN slave node receive dma transfer 849
Table 18-9. LINS encodings 851
Table 18-10. LINFlexD register overview 854
Table 19-1. Definition of I2C-bus terminology (refer to the I2C specification of Philips semiconductors) 887
Table 19-2. Data setup time and data hold time 892
Table 19-3. Communication modes to be shut down 894
Table 19-4. SMBus with PEC configuration 907
Table 19-5. I2C error flags 910
Table 19-6. I2C interrupt events 910
Table 19-7. I2C register overview 912
Table 20-1. SPI signal description 928
Table 20-2. Quad-SPI signal description 929
Table 20-3. NSS function in slave mode 932
Table 20-4. NSS function in master mode 933
Table 20-5. SPI operation modes 934
Table 20-6. SPI Master clock constraints when MREQSEL = 1 943
Table 20-7. SPI interrupt requests 947
Table 20-8. I2S bitrate calculation formulas 956
Table 20-9. Audio sampling frequency calculation formulas 957
Table 20-10. Direction of I2S interface signals for each operation mode 957
Table 20-11. I2S interrupt 961
Table 20-12. SPI register overview 962
Table 21-1. QSPI signal description 978
Table 21-2. QSPI command description 980
Table 21-3. QSPI signal line modes 981
Table 21-4. AHB write access mode and number of bytes add to FIFO 983
Table 21-5. TERR and AHB error conditions 986
Table 21-6. QSPI interrupt events 986
Table 21-7. QSPI register overview 990
Table 22-1. CAN module feature configuration 1006
Table 22-2. CAN cluster memory map 1006
Table 22-3. Shared memory map 1010
Table 22-4. Minimum frame capacity for each CAN / DTM_CAN 1010
Table 22-5. Maximum frame capacity for each CAN / DTM_CAN 1011
Table 22-6. Interrupt events 1012
Table 22-7. CAN and DTM_CAN interrupt mapping 1013
Table 22-8. CAN Cluster register overview 1016
Table 22-9. DTM_CANx (x = 0...7) register distribution (2KB) 1017
Table 22-10. IP user manual versions 1025
Table 22-11. Coding of DLC in CAN FD 1030
Table 22-12. Rx Buffer / FIFO element size 1042
Table 22-13. Example filter configuration for Rx buffers 1044
Table 22-14. Possible Configurations for Frame Transmission 1044
Table 22-15. Tx Buffer / FIFO / Queue element size 1046
Table 22-16. CAN register overview 1060

Table 22-17. IP user manual versions 1106
Table 22-18. Supported M_CAN message buffers 1106
Table 22-19. DMU register overview 1113
Table 22-20. Address map 1113
Table 23-1. CMP inputs and outputs summary 1126
Table 24-1. GD32A711 / A712 MBIST mapping 1147
Table 24-2. STCM address information 1150
Table 24-3. Security key loop 1150
Table 24-4.STCM register overview 1154
Table 25-1. MTC ECC data channel mapping 1169
Table 25-2. MTC ECC address channel mapping 1170
Table 25-3. MTC EDC data channel mapping 1171
Table 25-4. MTC EDC after ECC channel mapping 1173
Table 25-5. MTC register overview 1175
Table 26-1. NVIC exception types in Cortex®-M7 1182
Table 26-2. GD32A711/A712 interrupt vector table 1182
Table 26-3. IRM register overview 1191
Table 28-1. NVIC exception types in Cortex®-M23 1209
Table 28-2. Interrupt vector table for HSM 1209
Table 28-3. HSM low power mode 1211
Table 29-1 FMU fault group definition 1214
Table 29-2. FG reaction 1220
Table 29-3. FMU works mode 1221
Table 29-4. FMU works transition 1222
Table 29-5. Bi-stable mode and FFIO signal 1226
Table 29-6. Initialization registers 1228
Table 29-7. FMU register overview 1232
Table 29-7. Test mode bi-stable encoding 1243
Table 30-1. DAP instruction 1261
Table 30-2. DAP master address mapping of GD32A711 / A712 1262
Table 30-3. Debugging from first instruction step description 1262
Table 30-4. Reset results for different reset types 1265
Table 30-5. Generates different reset methods 1265
Table 30-6. Debug access 1265
Table 30-7. JTAG instruction register 1267
Table 30-8. JTAG signal description 1268
Table 30-9. JDC signal description 1270
Table 30-10. CTI assignments of GD32A711 / A712 1274
Table 30-11. GD_AP register overview 1275
Table 30-12. GD_AP register overview 1284
Table 31-1. List of abbreviations used in register 1288
Table 31-2. List of terms 1288
Table 32-1. Revision history 1290

1. System and memory architecture

The devices of GD32A71x series are 32-bit general-purpose microcontrollers based on the Arm® Cortex®-M7 processor. The Arm® Cortex®-M7 processor includes 64-bit AMBA4 AXI, 32-bit AHB peripheral (AHBP) port, 32-bit AHB slave (AHBS) port for external master to access memories and APB interface for CoreSight debug components. The memory organization uses a Harvard architecture, pre-defined memory map and up to 4 GB of memory space, making the system flexible and extendable.

There is one processor core and one core embedded in HSM module, assist in achieving higher frequency, efficiency and security level.

1.1. Arm® Cortex®-M7 processor

The Arm® Cortex®-M7 processor is a highly efficient high-performance, embedded processor that features low interrupt latency, low-cost debug, and has backwards compatibility with existing Cortex-M profile processors. The processor has an in-order super-scalar pipeline that means many instructions can be dual-issued, including load/load and load/store instruction pairs because of multiple memory interfaces. The Cortex®-M7 is a high-performance processor, which features a 6-stage superscalar pipeline with branch prediction and an optional FPU capable of single-precision and optionally double-precision operations. The instruction and data buses have been enlarged to 64-bit wide over the previous 32-bit buses.

The interfaces that the processor supports include:

- 64-bit AXI4 interface
- 32-bit AHB master interface
- 32-bit AHB slave interface
- 64-bit instruction TCM interface
- 2x32-bit data TCM interfaces

The processor contains the following external interfaces:

- AHBP interface
- AHBS interface
- AHBD interface
- External Private Peripheral Bus
- ATB interfaces
- TCM interface
- Cross Trigger interface
- MBIST interface
- AXIM interface

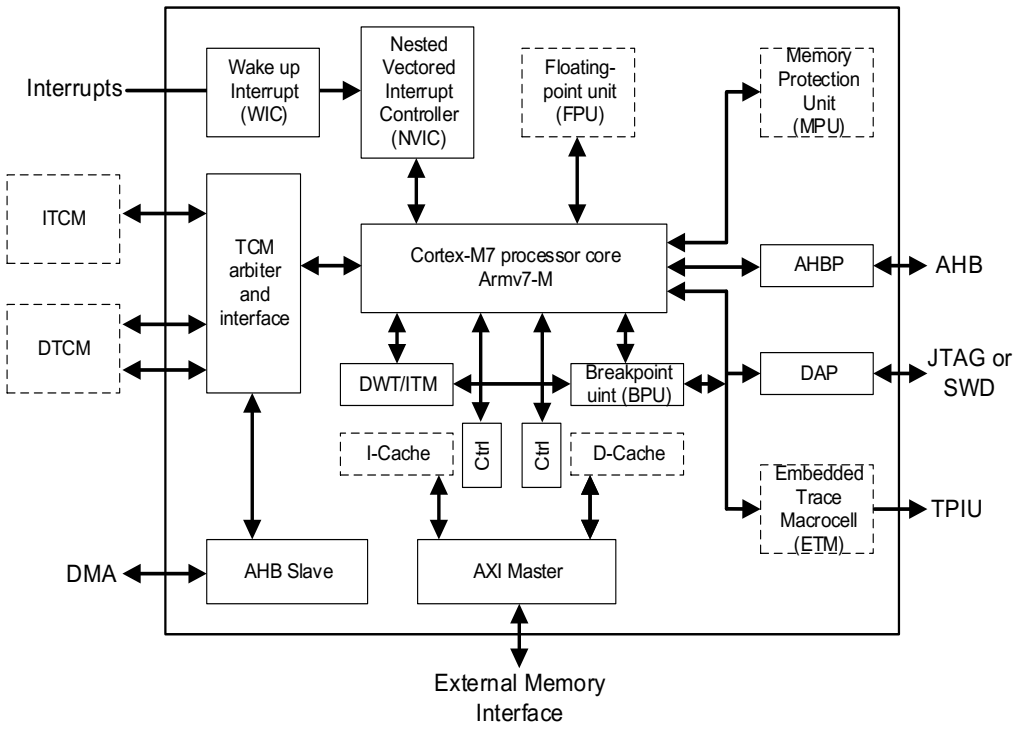
The Cortex®-M7 processor is based on the ARMv7-M architecture and supports a powerful and scalable instruction set including general data processing I/O control tasks, advanced data processing bit field manipulations, DSP and floating point instructions. Some system

peripherals listed below are also provided by Cortex®-M7:

- Nested Vectored Interrupt Controller (NVIC)
- Flash Patch and Breakpoint (FPB)
- Data Watchpoint and Trace (DWT)
- Instrumentation Trace Macrocell (ITM)
- Embedded Trace Macrocell (ETM)
- JTAG or SWD Debug Port
- Trace Port Interface Unit (TPIU)
- Memory Protection Unit (MPU)
- Floating Point Unit (FPU)
- Load Store Unit (LSU)
- Data Processing Unit (DPU)
- Prefetch Unit (PFU)

Figure 1-1. The structure of the Cortex®-M7 processor shows the Cortex®-M7 processor block diagram. For more information, refer to the Arm® Cortex®-M7 Technical Reference Manual.

Figure 1-1. The structure of the Cortex®-M7 processor

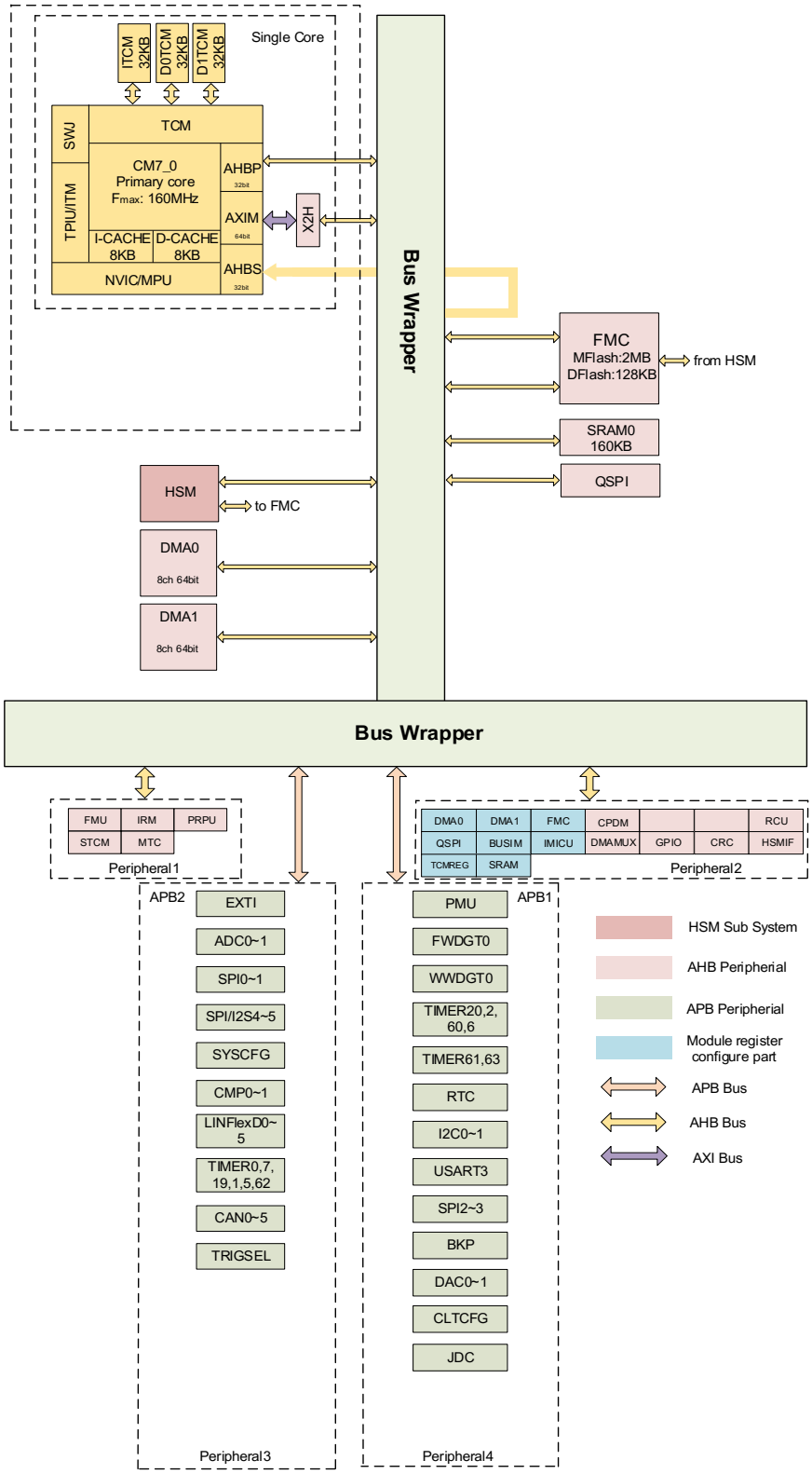


1.2. System architecture

A 64-bit multilayer and a 32-bit multilayer bus are implemented in the GD32A71x devices, which enable parallel access paths between multiple masters and slaves in the system. The bus matrix providing arbitration among the bus masters when they access the same slave

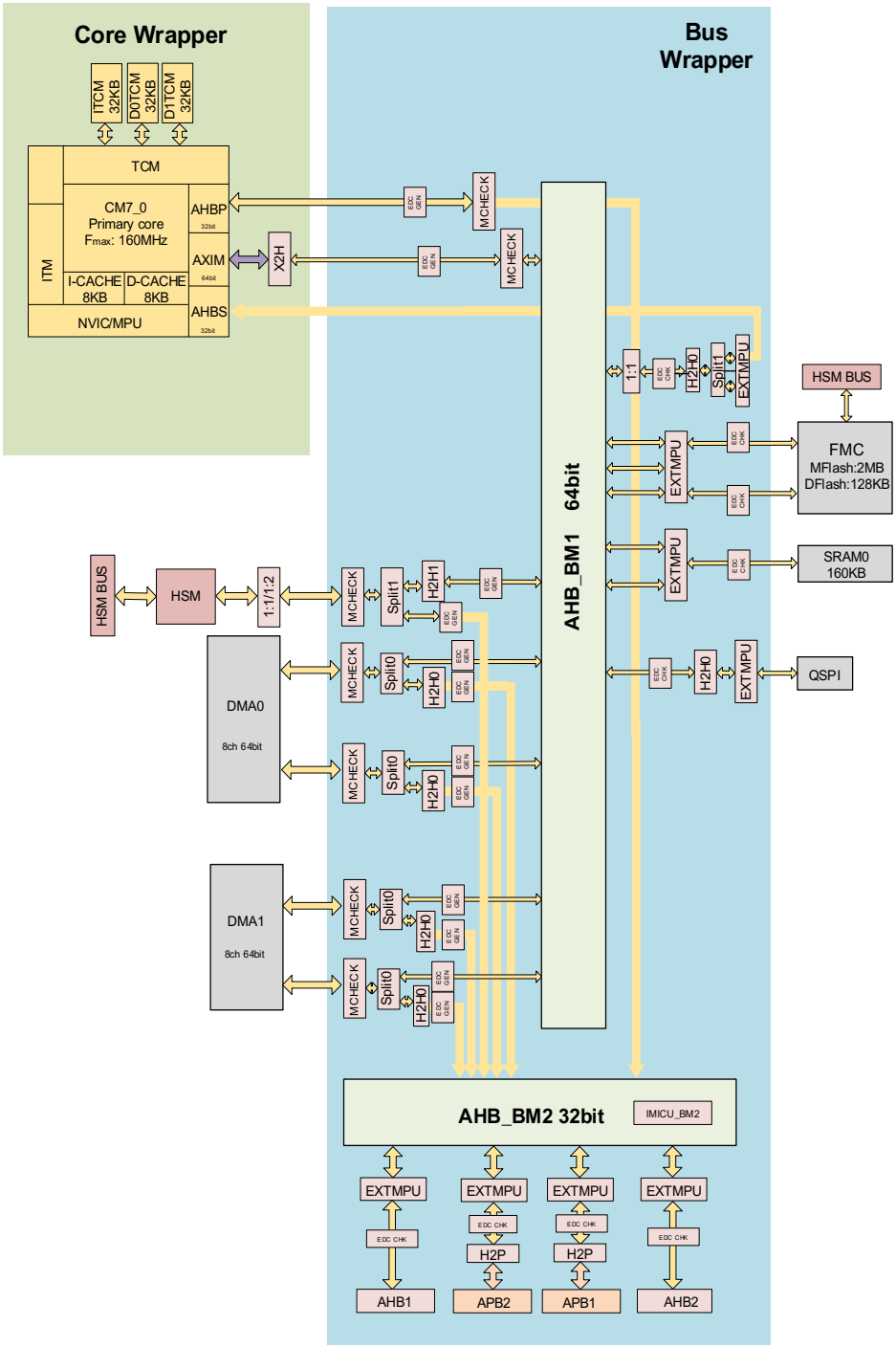
and the arbitration of the bus is Round-Robin or fixed priority. The multilayer bus consists of an AHB interconnect matrix, two AHB bus and two APB buses. The system architecture is shown in ***Figure 1-2. The system architecture of GD32A711/A712 devices***.

Figure 1-2. The system architecture of GD32A711/A712 devices



The detailed architecture of core wrapper and bus wrapper is shown in ***Figure 1-3. The architecture of core wrapper and bus wrapper of GD32A711/A712 devices***.

Figure 1-3. The architecture of core wrapper and bus wrapper of GD32A711/A712 devices



As is shown on the above figures, AHB_BMx(x=1,2) is the AHB matrix. The AHB Split module shall split one 32-bit AHB port into two 32-bit AHB ports based on the address decode. The AHB to APB bridge (H2P) provides the transformation between AHB bus protocol and APB bus protocol. H2H0/H2H1 module converts 64-bit or 32-bit AHB port to 32-bit AHB port or 64-bit AHB port. 1:1/1:2 module is an AHB frequency conversion bridge that facilitates the operation of both master and slave devices at different frequencies. MCHECK is the master

check module, including timeout check and slaver ID permission check. EDC GEN module manages the generation of address and control information for each command phase during crossbar transfers. EDC CHK module manages the generation of EDC information for each data phase during crossbar transfer.

1.2.1. Interacting with other modules

In the bus wrapper, if any components' related functional safety measures encounter errors, they will be forwarded to the FMU for further handling.

SYSCFG enables EDC detection for EDC check of each EDC GEN/EDC CHK. EDC GEN enables detection for rdata, and EDB CHK enables detection for addr and wdata.

MTC serves as the fault injection module for EDC GEN and EDC CHK, allowing injection of errors in wdata, rdata and the transmitted EDC code.

1.3. Memory map

The Arm® Cortex®-M7 processor is structured in Harvard architecture which can use separate buses to fetch instructions and load/store data. ***Memory map of GD32A711/A712 devices*** show the memory map of the GD32A71x series devices, including Code, SRAM, peripheral, and other pre-defined regions.

Table 1-1. Memory map of GD32A711/A712 devices

GD32A711/A712			
Pre-defined Regions	**Bus**	**Address**	**Peripherals**
External RAM		0xA000 0000 - 0xDFFF FFFF	Reserved
		0x9000 0000 - 0x9FFF FFFF	QSPI_MEM
		0x6000 0000 - 0x8FFF FFFF	Reserved
Peripheral	AHB3	0x5006 0C00 - 0x5FFF FFFF	Reserved
		0x5006 0800 - 0x5006 0BFF	HSM_TRNG
		0x5006 0400 - 0x5006 07FF	HSM_HAU
		0x5006 0000 - 0x5006 03FF	HSM_CAU
		0x5002 4C00 - 0x5005 FFFF	Reserved
		0x5002 4000 - 0x5002 4BFF	HSM_SM2
		0x5002 0000 - 0x5002 3FFF	HSM_PKCAU
		0x5001 2400 - 0x5001 FFFF	Reserved
		0x5001 2000 - 0x5001 23FF	HSM_HSMIF
		0x5001 1400 - 0x5001 1FFF	Reserved
		0x5001 1000 - 0x5001 13FF	HSM_SM4
		0x5001 0C00 - 0x5001 0FFF	HSM_SM3
		0x5001 0800 - 0x5001 0BFF	Reserved
		0x5001 0400 - 0x5001 07FF	HSM_RCU

GD32A711/A712			
Pre-defined Regions	**Bus**	**Address**	**Peripherals**
		0x5001 0000 - 0x5001 03FF	HSM_DMA
	APB3	0x5000 FC00 - 0x5000 FFFF	HSM_JDC
		0x5000 0C00 - 0x5000 FBFF	Reserved
		0x5000 0800 - 0x5000 0BFF	HSM_TIMER1
		0x5000 0400 - 0x5000 07FF	HSM_TIMER0
		0x5000 0000 - 0x5000 03FF	HSM_WWDGT
	AHB2	0x4900 A400 - 0x4FFF FFFF	Reserved
		0x4900 A000 - 0x4900 A7FF	EXTMPU
		0x4900 9C00 - 0x4900 9FFF	Reserved
		0x4900 9800 - 0x4900 9BFF	IMICU_BM2
		0x4900 9400 - 0x4900 97FF	IMICU_BM1
		0x4900 9000 - 0x4900 93FF	BUSIM
		0x4900 8C00 - 0x4900 8FFF	HSMIF
		0x4900 8800 - 0x4900 8BFF	Reserved
		0x4900 8400 - 0x4900 87FF	Reserved
		0x4900 4400 - 0x4900 83FF	Reserved
		0x4900 4000 - 0x4900 43FF	TCM_REG
		0x4900 3C00 - 0x4900 3FFF	RCU_RCTL
		0x4900 3400 - 0x4900 3BFF	Reserved
		0x4900 3000 - 0x4900 33FF	CRC
		0x4900 2400 - 0x4900 2FFF	Reserved
		0x4900 2000 - 0x4900 23FF	FMC
		0x4900 1C00 - 0x4900 1FFF	Reserved
		0x4900 1800 - 0x4900 1BFF	CPDM
		0x4900 1400 - 0x4900 17FF	QSPI_REG
		0x4900 1000 - 0x4900 13FF	RCU_CCTL
		0x4900 0C00 - 0x4900 0FFF	SRAM_REG
		0x4900 0800 - 0x4900 0BFF	DMAMUX
		0x4900 0400 - 0x4900 07FF	DMA1
		0x4900 0000 - 0x4900 03FF	DMA0
		0x4800 3800 - 0x48FF FFFF	Reserved
		0x4800 3400 - 0x4800 37FF	GPION
		0x4800 3000 - 0x4800 33FF	GPIOM
		0x4800 2C00 - 0x4800 2FFF	GPIOL
		0x4800 2800 - 0x4800 2BFF	GPIOK
		0x4800 2400 - 0x4800 27FF	GPIOJ
		0x4800 2000 - 0x4800 23FF	GPIOI
		0x4800 1C00 - 0x4800 1FFF	GPIOH

GD32A711/A712			
Pre-defined Regions	**Bus**	**Address**	**Peripherals**
		0x4800 1800 - 0x4800 1BFF	GPIOG
		0x4800 1400 - 0x4800 17FF	GPIOF
		0x4800 1000 - 0x4800 13FF	GPIOE
		0x4800 0C00 - 0x4800 0FFF	GPIOD
		0x4800 0800 - 0x4800 0BFF	GPIOC
		0x4800 0400 - 0x4800 07FF	GPIOB
		0x4800 0000 - 0x4800 03FF	GPIOA
	AHB1	0x4026 0000 - 0x47FF FFFF	Reserved
		0x4025 0400 - 0x4025 FFFF	GD_AP
		0x4025 0300 - 0x4025 03FF	Reserved
		0x4025 0200 - 0x4025 02FF	AHB-AP(CM23)
		0x4025 0100 - 0x4025 01FF	AHB-AP(CM7)
		0x4025 0000 - 0x4025 00FF	APB-AP
		0x4003 8400 - 0x4024 FFFF	Reserved
		0x4003 8000 - 0x4003 83FF	PRPU
		0x4002 C000 - 0x4003 7FFF	Reserved
		0x4002 8000 - 0x4002 BFFF	CAN2 cluster (16KB)
		0x4002 5C00 - 0x4002 7FFF	Reserved
		0x4002 5000 - 0x4002 5BFF	STCM
		0x4002 4800 - 0x4002 4FFF	Reserved
		0x4002 4400 - 0x4002 47FF	MTC
		0x4002 4000 - 0x4002 43FF	Reserved
		0x4002 3C00 - 0x4002 3FFF	Reserved
		0x4002 3800 - 0x4002 3BFF	IRM
		0x4002 3400 - 0x4002 37FF	FMU
		0x4002 0000 - 0x4002 33FF	Reserved
	APB2	0x4001 C000 - 0x4001 FFFF	CAN1 cluster (16KB)
		0x4001 8000 - 0x4001 BFFF	CAN0 cluster (16KB)
		0x4001 7C00 - 0x4001 7FFF	CMP0
		0x4001 7800 - 0x4001 7BFF	SPI1
		0x4001 7400 - 0x4001 77FF	Reserved
		0x4001 7000 - 0x4001 73FF	Reserved
		0x4001 6C00 - 0x4001 6FFF	CMP1
		0x4001 6800 - 0x4001 6BFF	Reserved
		0x4001 6400 - 0x4001 67FF	TIMER62
		0x4001 6000 - 0x4001 63FF	TIMER5
		0x4001 5C00 - 0x4001 5FFF	TIMER1
		0x4001 5400 - 0x4001 5BFF	Reserved

GD32A711/A712			
Pre-defined Regions	**Bus**	**Address**	**Peripherals**
		0x4001 5000 - 0x4001 53FF	TIMER19
		0x4001 4C00 - 0x4001 4FFF	Reserved
		0x4001 4800 - 0x4001 4BFF	SPI5/I2S5
		0x4001 4400 - 0x4001 47FF	SPI4/I2S4
		0x4001 4000 - 0x4001 43FF	Reserved
		0x4001 3C00 - 0x4001 3FFF	Reserved
		0x4001 3800 - 0x4001 3BFF	Reserved
		0x4001 3400 - 0x4001 37FF	TIMER7
		0x4001 3000 - 0x4001 33FF	SPI0
		0x4001 2C00 - 0x4001 2FFF	TIMER0
		0x4001 2400 - 0x4001 2BFF	Reserved
		0x4001 2000 - 0x4001 23FF	TRIGSEL
		0x4001 1C00 - 0x4001 1FFF	LINFlexD5
		0x4001 1800 - 0x4001 1BFF	LINFlexD4
		0x4001 1400 - 0x4001 17FF	LINFlexD3
		0x4001 1000 - 0x4001 13FF	LINFlexD2
		0x4001 0C00 - 0x4001 0FFF	LINFlexD1
		0x4001 0800 - 0x4001 0BFF	LINFlexD0
		0x4001 0400 - 0x4001 07FF	EXTI
		0x4001 0000 - 0x4001 03FF	SYSCFG
		0x4000 F800 - 0x4000 FFFF	ADC1
		0x4000 F000 - 0x4000 F7FF	ADC0
	APB1	0x4000 EC00 - 0x4000 EFFF	HOST_JDC(JTAG)
		0x4000 E800 - 0x4000 EBFF	SPI3
		0x4000 E400 - 0x4000 E7FF	SPI2
		0x4000 E000 - 0x4000 E3FF	Reserved
		0x4000 DC00 - 0x4000 DFFF	Reserved
		0x4000 D800 - 0x4000 DBFF	Reserved
		0x4000 D400 - 0x4000 D7FF	TIMER63
		0x4000 D000 - 0x4000 D3FF	TIMER61
		0x4000 CC00 - 0x4000 CFFF	DAC1
		0x4000 C000 - 0x4000 CBFF	Reserved
		0x4000 A000 - 0x4000 BFFF	Reserved
		0x4000 9C00 - 0x4000 9FFF	Reserved
		0x4000 9800 - 0x4000 9BFF	Reserved
		0x4000 9400 - 0x4000 97FF	Reserved
		0x4000 9000 - 0x4000 93FF	Reserved
		0x4000 8C00 - 0x4000 8FFF	Reserved

GD32A711/A712			
Pre-defined Regions	**Bus**	**Address**	**Peripherals**
		0x4000 8800 - 0x4000 8BFF	Reserved
		0x4000 7800 - 0x4000 87FF	Reserved
		0x4000 7400 - 0x4000 77FF	DAC0
		0x4000 7000 - 0x4000 73FF	PMU
		0x4000 6C00 - 0x4000 6FFF	BKP
		0x4000 6000 - 0x4000 6BFF	Reserved
		0x4000 5C00 - 0x4000 5FFF	Reserved
		0x4000 5800 - 0x4000 5BFF	I2C1
		0x4000 5400 - 0x4000 57FF	I2C0
		0x4000 4C00 - 0x4000 53FF	Reserved
		0x4000 4800 - 0x4000 4BFF	Reserved
		0x4000 4400 - 0x4000 47FF	Reserved
		0x4000 4000 - 0x4000 43FF	Reserved
		0x4000 3C00 - 0x4000 3FFF	Reserved
		0x4000 3800 - 0x4000 3BFF	Reserved
		0x4000 3400 - 0x4000 37FF	Reserved
		0x4000 3000 - 0x4000 33FF	FWDGT0
		0x4000 2C00 - 0x4000 2FFF	WWDGT0
		0x4000 2800 - 0x4000 2BFF	RTC
		0x4000 2400 - 0x4000 27FF	CLTCFG
		0x4000 1800 - 0x4000 23FF	Reserved
		0x4000 1400 - 0x4000 17FF	TIMER6
		0x4000 1000 - 0x4000 13FF	TIMER60
		0x4000 0800 - 0x4000 0FFF	Reserved
		0x4000 0400 - 0x4000 07FF	TIMER2
		0x4000 0000 - 0x4000 03FF	TIMER20
SRAM		0x2501 0000 - 0x3FFF FFFF	Reserved
		0x2500 0000 - 0x2500 FFFF	HSM SRAM(64KB)
		0x2405 0000 - 0x24FF FFFF	Reserved
		0x2402 8000 - 0x2404 FFFF	Reserved
		0x2400 0000 - 0x2402 7FFF	SRAM0(160KB)
		0x2142 0000 - 0x23FF FFFF	Reserved
		0x2140 0000 - 0x2141 FFFF	Reserved
		0x2101 0000 - 0x213F FFFF	Reserved
		0x2100 0000 - 0x2100 FFFF	CM7 DTCM Backdoor (64KB)
		0x2001 0000 - 0x20FF FFFF	Reserved
		0x2000 0000 - 0x2000 FFFF	CM7_DTCM (64KB)
Code		0x1FFF F9F8 - 0x1FFF FFFF	Reserved

GD32A711/A712			
Pre-defined Regions	**Bus**	**Address**	**Peripherals**
		0x1FFF F900 - 0x1FFF F9F7	HSM OTP(248B)
		0x1FFF F890 - 0x1FFF F8FF	Reserved
		0x1FFF F880 - 0x1FFF F88F	HSM Option Bytes(16B)
		0x1FFF F800 - 0x1FFF F87F	Reserved
		0x1FFF 9800 - 0x1FFF F7FF	System memory(24KB)
		0x1FFF 8840 - 0x1FFF 97FF	Reserved
		0x1FFF 8800 - 0x1FFF 883F	Option Bytes (64B)
		0x1FFF 7800 - 0x1FFF 87FF	Reserved
		0x1FFF 7000 - 0x1FFF 77FF	OTP(2KB)
		0x1141 0000 - 0x1FFF 6FFF	Reserved
		0x1140 0000 - 0x1140 FFFF	Reserved
		0x1100 8000 - 0x113F FFFF	Reserved
		0x1100 0000 - 0x1100 7FFF	CM7 ITCM Backdoor (32KB)
		0x0900 1000 - 0x10FF FFFF	Reserved
		0x0900 0000 - 0x0900 0FFF	HSM ROM (4KB)
		0x0889 8000 - 0x08FF FFFF	Reserved
		0x0888 0000 - 0x0889 7FFF	HSM Dflash (96KB)
		0x0882 0000 - 0x0887 FFFF	Reserved
		0x0880 0000 - 0x0881 FFFF	Dflash (128KB)
		0x084C 0000 - 0x087F FFFF	Reserved
		0x0844 0000 - 0x084B FFFF	HSM code(512KB)
		0x0840 8000 - 0x0843 FFFF	Reserved
		0x0840 0000 - 0x0840 7FFF	sBAF
		0x0820 0000 - 0x083F FFFF	Reserved
		0x0810 0000 - 0x081F FFFF	Main Flash bank1 (1MB) 256KB shared with HSM bank1
		0x0800 0000 - 0x080F FFFF	Main Flash bank0 (1MB) 256KB shared with HSM bank0
		0x0000 8000 - 0x07FF FFFF	Reserved
		0x0000 0000 - 0x0000 7FFF	CM7_ITCM (32KB)

1.3.1. On-chip SRAM memory

- Support 64-bit AHB interface
- Support configurable read access timing
- Support data ECC
- Support address EDC
- Support hardware EDC after ECC
- Support write protection

- Support two buffers (CBE and UBE) to record single-bit correctable error address and uncorrectable error address

The devices of GD32A711/A712 series contain up to 256KB of on-chip SRAM (96KB TCM,160KB SRAM0). All of AHB SRAM support byte, half-word (16 bits), word (32 bits) and double word(64bit) accesses

Table 1-2. SRAM size and width of GD32A711/A712 devices

Single core	SIZE	Width bits	ECC bits	Address	Backdoor Address
SRAM0	32KB (retention in standby mode)	64	9	0x2400 0000 - 0x2400 7FFF	-
	128KB			0x2400 8000 - 0x2402 7FFF	-
CM7_ITCM	32KB	64	9	0x0000 0000 - 0x0000 7FFF	0x1100 0000 - 0x1100 7FFF
CM7_DTCM	32KB (D0TCM)	32	8	0x2000 0000 - 0x2000 FFFF	0x2100 0000 - 0x2100 FFFF
	32KB (D1TCM)	32	8		

ITCM/DTCM SRAM access frequency

If TCM_WAITSTATE in SYSCFG_SRAMCFG register is set to 0, the corresponding TCM SRAM can be accessed below a maximum frequency of f_{TWW} which can be referred to the datasheet. TCM_WAITSTATE is set to 0 after system reset.

SRAM0 access frequency

If SRAM_WAITSTATE in SYSCFG_SRAMCFG register is set to 0, the corresponding SRAM0 can be accessed below a maximum frequency of f_{TWW} which can be referred to the datasheet. SRAM_WAITSTATE is set to 0 after system reset.

TCM/SRAM Initialization

Before TCM/SRAM initialization is complete, the Cortex-M7 core may issue speculative accesses to uninitialized TCM/SRAM regions, resulting in ECC error indications. This may occur even if software does not explicitly read these regions. For details on speculative accesses, refer to the Arm Cortex-M7 Processor Technical Reference Manual, r1p2, Section 5.2, "Speculative accesses", which states in part:

"Speculative instruction fetches can be initiated to any Normal, executable memory address. This can occur regardless of whether the fetched instruction gets executed or, in rare cases, whether the memory address contains any valid program instruction."

Software shall perform the following steps in order:

1. During early power-on startup, initialize all TCM/SRAM regions using the write access widths and address ranges specified in Table 1-2. Software shall not execute code from or read data from these regions until initialization is complete.

2. After the initialization writes have completed, clear the ECC-related flags in accordance with the clearing rules specified for the TCM/SRAM error status registers, buffer validity registers, and overflow status registers.

3. If another ECC error is detected during operation after the previously recorded error status has been cleared, software shall treat it as a genuine ECC fault and respond in accordance with the fault-handling procedure. Software shall not attribute the error to speculative reads or handle it solely by clearing the flags as described in Step 2.

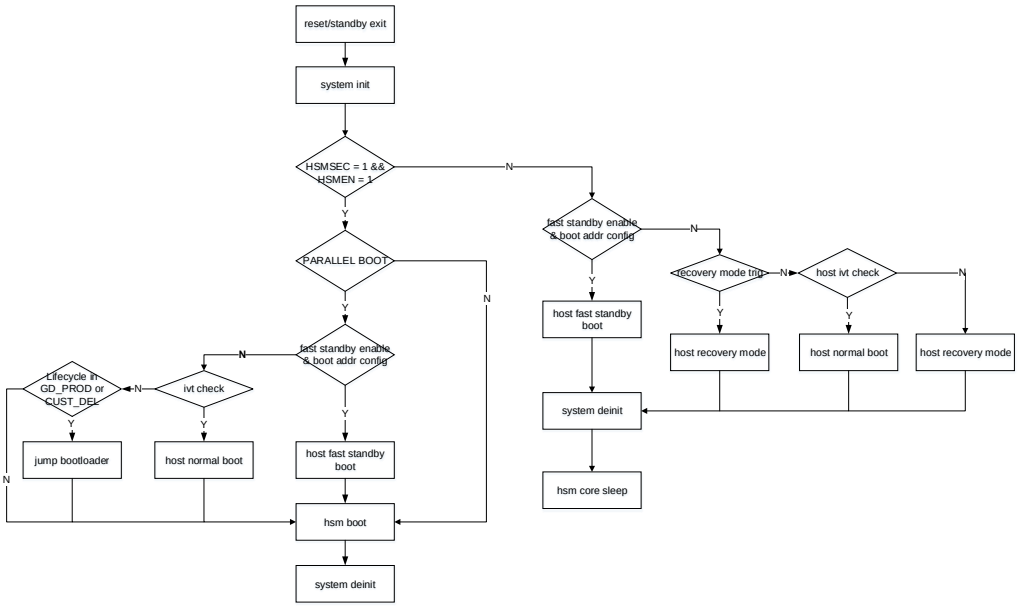
1.4. Boot configuration

- Support 3 kinds of reset, including POR, system reset and subsystem reset
- Support startup function of host CPU
- Support multiple core modes: dual core, single core lock-step core, non-secure Cortex®-M23 core
- Support non-HSM mode and HSM mode
- Support boot from various power modes
- Support various lifecycle
- Support 3 unexpected modes, including normal startup, recovery mode, and HSM recovery mode
- Support debug from first instruction function

The embedded bootloader is stored in system memory and only supports downloading and uploading to SRAM, without supporting operations on FLASH and DFLASH. In GD32A7xx devices, the bootloader can interact with external devices via LINFlexD and CAN. For details, refer to the product datasheet.

The boot sequence flowchart is shown in *Figure 1-4. High-level boot sequence flowchart*.

Figure 1-4. High-level boot sequence flowchart



Note:

1. The fast standby mode can set the standby exit start address to achieve a quick startup. For more details, please refer to **AN_Auto-00921-GD32A7xx-Application Note-Fast Standby Boot**.
2. The sBAF boot code configures GPIO PD0, PD1, and PA10 as push-pull outputs. The ports are left unconnected, and the pins output a low level. After the reset pin PB7 is released and pulled high, PD0 generates a high-level pulse approximately 762 μs later. The pulse lasts approximately 1.5 μs, indicating that sBAF is executing normally. Please consider the potential impact when using these pins.

1.4.1. Image vector table(IVT) and Boot configuration word (BCW)

The descriptions of IVT and BCW are ***Descriptions of IVT of GD32A711/A712 devices*** and ***Descriptions of BCW of GD32A711/A712 devices***. Users need to write the correct IVT (Image Vector Table) at the 3 macro base addresses (Bank0, Bank1, Dflash0) for the CPU to start up normally from BOOT_ADDR_X. If multiple IVTs are present, the CPU boots using only the first valid IVT found. The sBAF checks them in the following order: Bank0, Bank1, and DFlash0. For example, if the IVT in Bank0 is invalid while the IVTs in Bank1 and DFlash0 are both valid, the system uses the IVT in Bank1. For detailed boot information, refer to Section 1.4.2, Non-Secure Boot.

Table 1-3. Descriptions of IVT of GD32A711/A712 devices

Offset	Name	Byte size	Description
0x00	Tag	4	Magic number 0x5AA55AA5
0x04	Boot configuration word (BCW)	4	Boot configuration bit field.
0x08	Reserved	4	
0x0C	BOOT_ADDR_0	4	Core #0 (CM7_0) boot address. It must follow the VTOR register alignment restrictions.
0x10	Reserved	4	
0x14	Reserved	4	
0x18	Reserved	4	
0x1C	Reserved	4	
0x20	Reserved	4	
0x24	Lifecycle Configuration word	4	Address of LC configuration.
0x28	Reserved	4	
0x2C	Reserved	4	
0x30	Reserved	180	
0xE4	Reserved	12	
0xF0	Reserved	16	

Table 1-4. Descriptions of BCW of GD32A711/A712 devices

Bit	Name	Description
31	Reserved	
.....		
6	Reserved	
5	Reserved	
4	Reserved	
3	Reserved	
2	Reserved	
1	Reserved	
0	BOOT_TARGET_0 (CM7)	Indicates whether the Cortex-M7 application core clock is gated after boot 0b - Gated 1b - Ungated

Valid values for advancement in next lifecycles are mentioned in ***Table 1-5. Valid values for lifecycle advancement***. For all other values at given address lifecycle advancement is not allowed.

1. Only when IN_FIELD_RECOVER_MODE(0x1FFFF950U) bit 0-15 is full 0xFF and host debug is disabled, host can enter the recovery mode. If the value of IN_FIELD_RECOVER_MODE(0x1FFFF950U) bit 0-15 is not full 0xFF, host can not enter the recovery mode.

2. Only when IN_FIELD_RECOVER_MODE(0x1FFFF950U) bit 32-48 is not full 0xFF, the fast_standby mode can not be entered. If the value of IN_FIELD_RECOVER_MODE(0x1FFFF950U) bit 32-48 is full 0xFF, the fast_standby mode can be entered.

Note: The IN_FIELD_RECOVER_MODE area can only be written by the HSM core, and must comply with the Flash programming rules. It can only be written in 64-bit increments, cannot be erased, and cannot be reprogrammed.

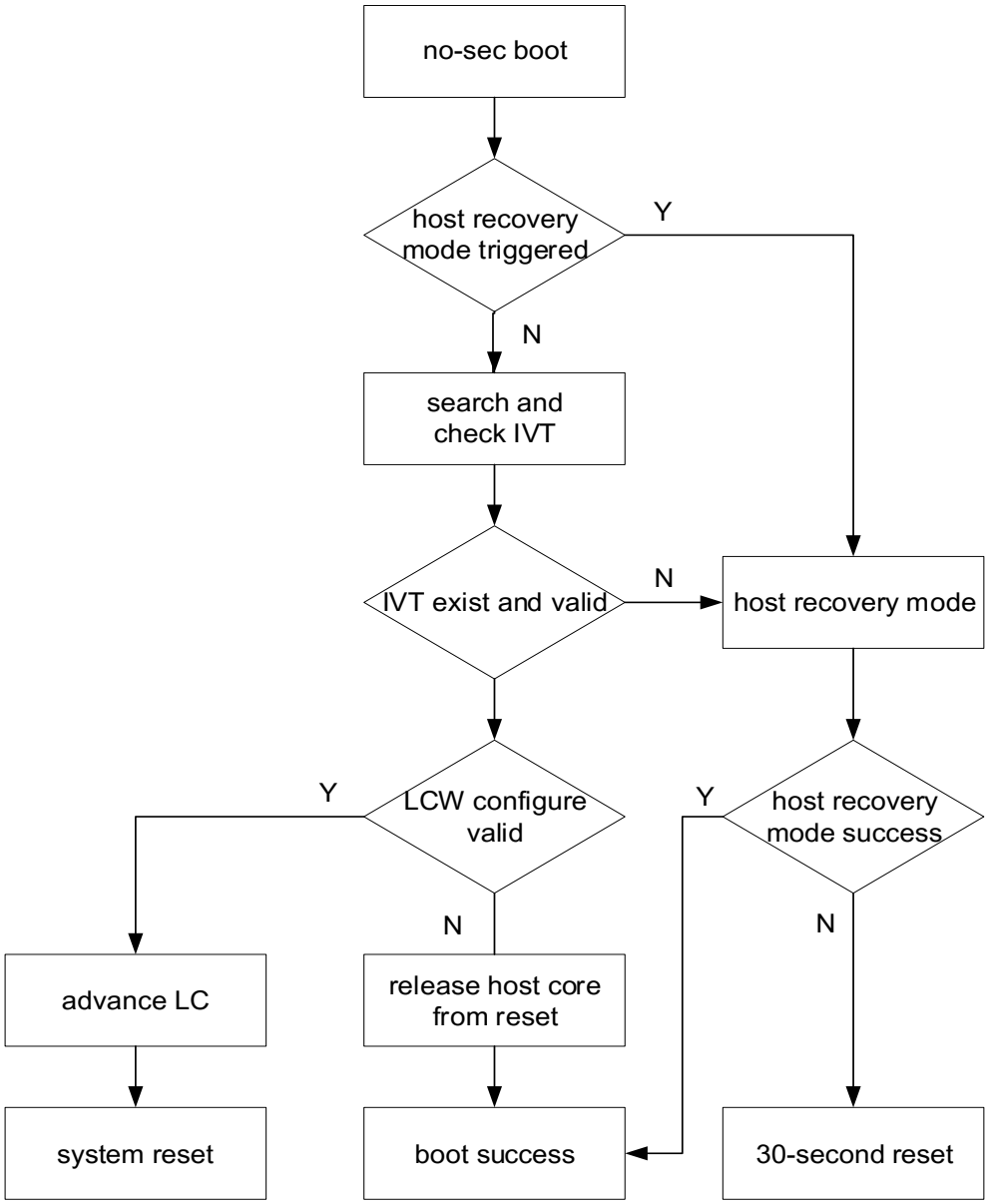
Table 1-5. Valid values for lifecycle advancement

Target LC	LCW advance value
OEM_PROD	0xDADA_DADA
IN_FIELD	0xBABA_BABA
FA	0x5A5A_5A5A

1.4.2. Nonsecure boot

The nonsecure boot sequence flowchart is shown in ***Figure 1-5. Nonsecure boot sequence flowchart***.

Figure 1-5. Nonsecure boot sequence flowchart



In nonsecure boot flow, sBAF boot the application core from image vector table(IVT), which contains the boot information of host system. IVT contains start address of each core. IVT must be programmed at least at one of the locations described in Memory map.

1. If none of the 3 macro base addresses have the correct IVT, the system will enter the host recovery mode, with CM7_0 starting from the bootloader and CM7_1 not being released;
2. If there are multiple IVTs, the CPU will only start according to the first correct IVT. The sBAF detection sequence is Bank0, Bank1, Dflash0. For example, if the IVT in Bank0 is incorrect, but the IVTs in Bank1 and Dflash0 are correct, the system will use the IVT from Bank1;

3. The system only supports normal startup from the Flash area, with SRAM addresses being considered invalid IVTs; CM7_X only supports the Host Flash area;
4. The startup status can be found in the HSM status register as described in section *1.4.5*;
5. If host or HSM Low/High-Level Security Protection(SPC) is configured, connecting a debugger will trigger read protection. If the read protection is not unlocked, the next system reset will cause sBAF to run abnormally, requiring a power reset.

After reset, sBAF searches for the first valid IVT starting from the lowest address. If there are multiple valid IVT at IVT locations at the same time, the IVT with lowest address is used.

1.4.3. Host recovery mode

Possible scenarios for entering recovery mode include:

1. IVT can not be found in the Flash;
2. The user program experiences 8 subsystem resets or 8 system resets (the corresponding reset escalation in OB CLTCFG must be enabled), and the relevant bit in SYSCFG_RMCFG is set.

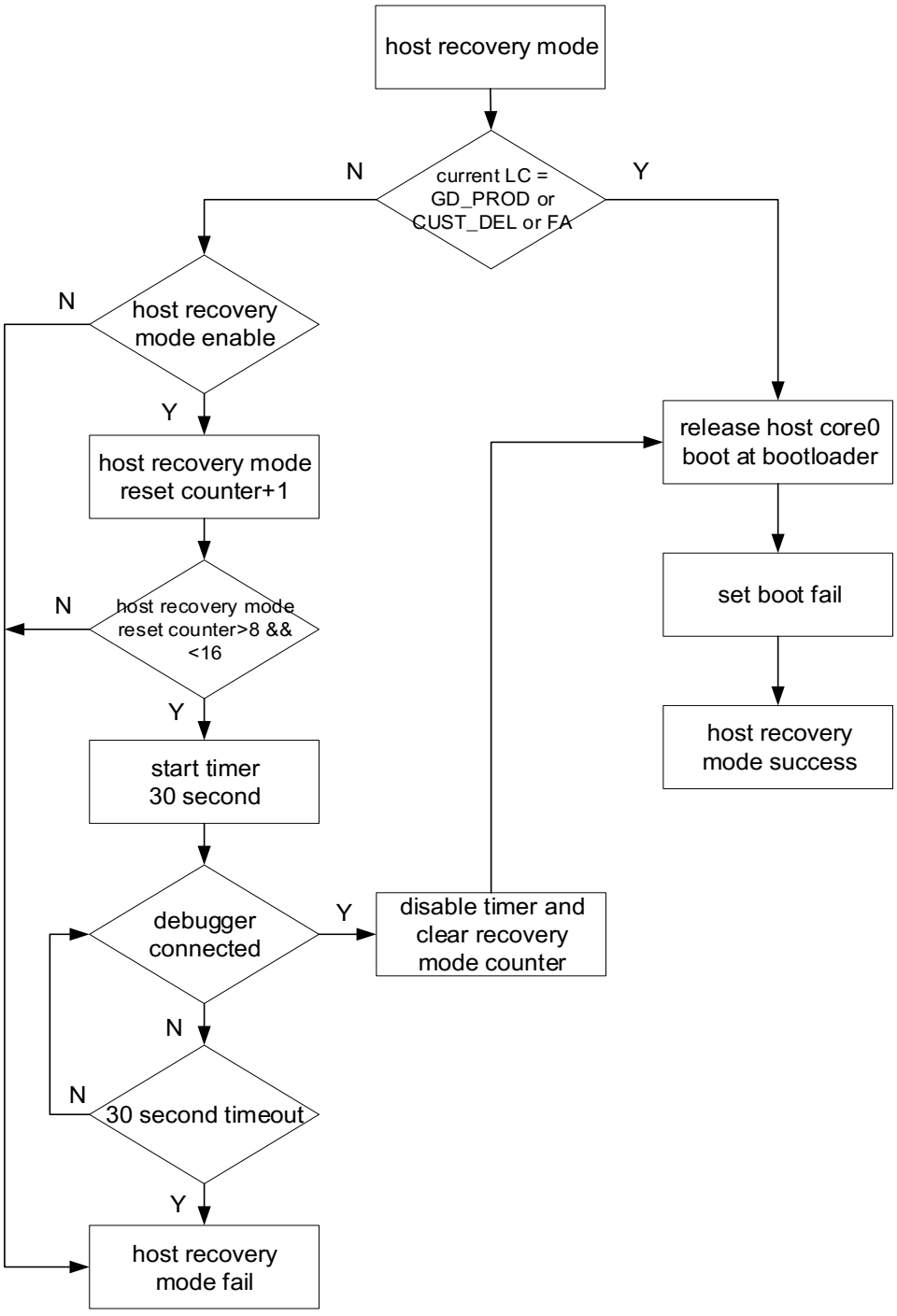
During the OEM_PROD, IN_FIELD phase of LC, the actual address of LC IN_FIELD_RECOVER_MODE address bit0-15 is not full FF or host debug disable. static authentication is not allowed.

Note:

1. Once it is confirmed that the user program is running stably, it is recommended that the user clears the subsystem reset count;
2. If the Host Security protection(SPC) is configured, entering the host recovery mode will enable software read protection. The user cannot reset the chip during the authentication process; otherwise, a Power-On Reset (POR) will be required to re-authenticate.
3. After entering recovery mode more than 16 times, a POR reset is required; otherwise, it will no longer enter recovery mode.
4. If the user disables recovery mode by configuring SYSCFG_RMCFG, the reset counter must be cleared by software before entering standby mode, and SYSCFG_RMCFG must be reconfigured after waking up from standby mode.

To avoid device "bricking", recovery mode allows host to debug and reprogram the corrupted IVT and abnormal application in all lifecycle. Below are the reasons causing recovery mode. The sequence flowchart of host recovery mode is shown in *Figure 1-6. Host recovery mode sequence flowchart*.

Figure 1-6. Host recovery mode sequence flowchart



1.4.4. Advance LC

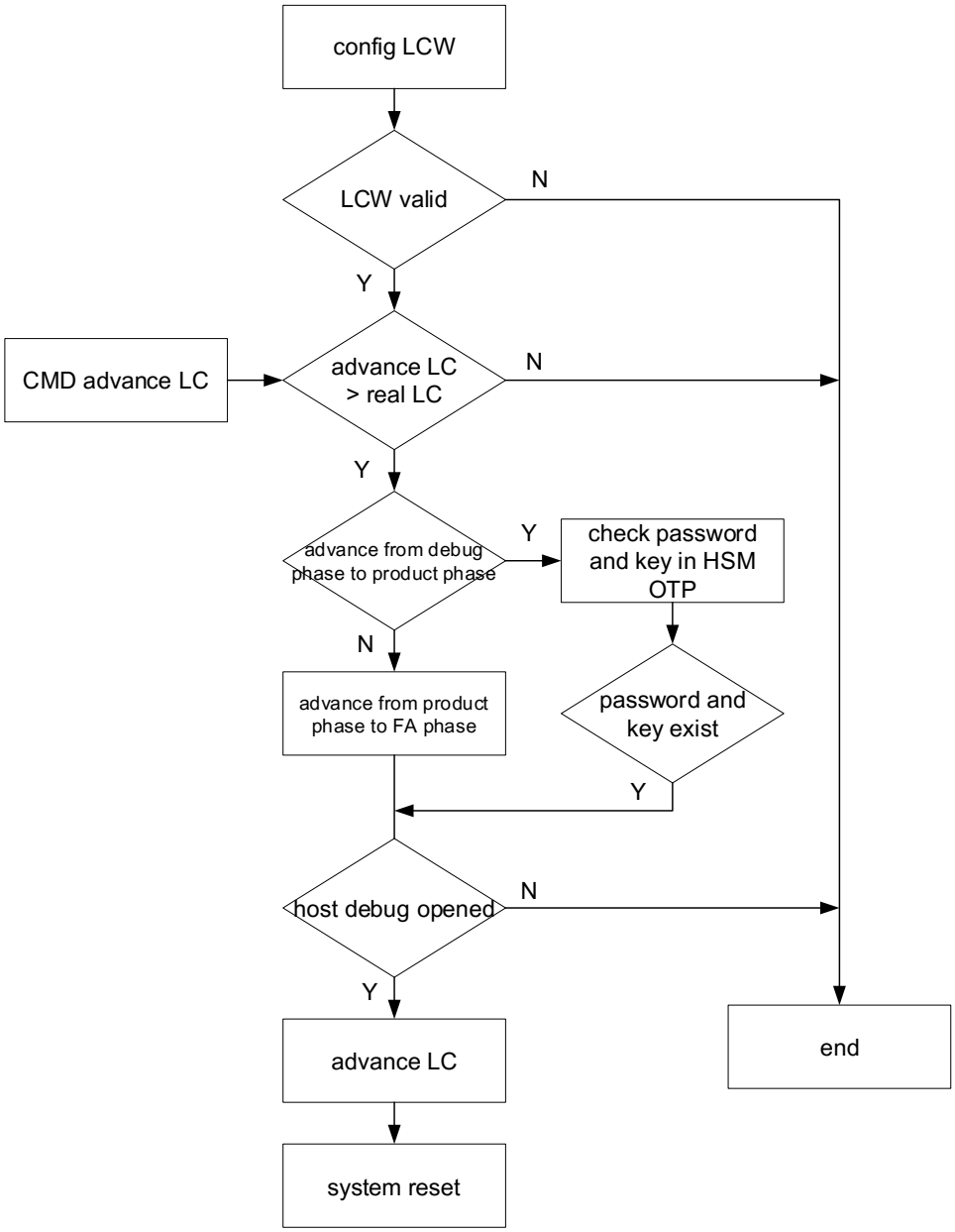
Lifecycle configuration word field allows User to Advance LC. To advance lifecycle User must program a valid 32 bits wide value at an address which must be given in IVT at address offset 24h. This address must be 4 bytes aligned and should not lie in Secure BAF/HSM FW protected areas. The sequence flowchart of lifecycle configuration is shown in *Figure 1-7.*

Sequence flowchart of lifecycle configuration.

Users can advance the Lifecycle (LC) through the following two methods:

1. By configuring the correct Lifecycle Configuration Word (LCW), which supports both Flash and SRAM addresses, and then performing a subsystem reset to advance the LC;
2. By sending a command to the HSM through the HSM Interface (HSMIF).

Figure 1-7. Sequence flowchart of lifecycle configuration



1.4.5. HSM Identifier

The host can obtain the HSM status through the HSMIF, with the register base address at

0x49008C00.

The host can send commands to the HSM through the host to HSM register to perform a swap or advance the Lifecycle (LC). After sending the command, the execution status of the command can be obtained through the HSM respond host register.

Note: In the no-sec boot state, users must use CMD to perform swap or advance LC; modifying it through library functions on their own may lead to the host being unable to access HSM resources.

Table 1-6. host to HSM register (Offset 0x10)

Bit	Read/Write	Description
31		
.....		
28		boot parallel disable
27		hsm image check enable
26		fast standby enable(1)
25		hsm recovery mode disable
24		host recovery mode disable
.....		
6		
5		
4		
3		
0-2 sBAF CMD	rw	NO CM: no command ----------0x00 CMD_BK_SWAP_R: bank swap bit set and reset ----0x01 CMD_BK_SWAP_NR: bank swap bit set no reset ----0x02 CMD_LC_AD_OEM: advance to OEM_PROD -------0x03 CMD_LC_AD_IF: advance to IN_FIELD ----------0x04 CMD_LC_AD_FA: advance to FA ----------0x05 CMD_OTP_CONFIG----------0x06

When the HSM core is disabled, users can modify the IN_FIELD_RECOVER_MODE (0x1FFFF950) area in the HSM OTP using the CMD_OTP_CONFIG command. When sending the CMD_OTP_CONFIG command, at least one bit in bit[24:28] must be set to 1, otherwise it will result in an error CMD_CONFIG_DATA_ERROR. For instance, sending 0x04000006 will configure FAST_STANDBY_CONFIG to a non-FF state.

Note: The IN_FIELD_RECOVER_MODE (0x1FFF F950~0x1FFF F957) area is initially set to all FF. The CMD_OTP_CONFIG command can only be successfully used once. After the command is successfully executed, the IN_FIELD_RECOVER_MODE will be programmed once, and cannot be erased or reprogrammed subsequently.

(1) In SBAF version 1.0.3 and subsequent versions, this bit is used to disable fast standby. In versions prior to 1.0.3, it was used to enable it.

Table 1-7. IN_FIELD_RECOVER_MODE-0x1FFFF950

	IN_FIELD_RECOVER_MODE-0x1FFFF950		
Offset		**Byte size**	**Description**
0x00	HOST_RECOVERY_MODE_CONFIG	2	0xFFFF-HOST recovery mode allowed Others - HOST recovery mode prohibited
0x02	HSM_RECOVERY_MODE_CONFIG	2	0xFFFF- HSM recovery mode allowed Others - HSM recovery mode prohibited
0x04	FAST_STANDBY_CONFIG	2	0xFFFF- fast stanby allowed Others - fast standby prohibited
0x06	HSM_IMG_AUTH_CHECK_CONFIG	1	0xFF - No secure boot Others - Secure boot enabled (sec boot)
0x07	BOOT_PARALLEL_CONFIG	1	0xFF- parallel boot allowed Others - parallel boot prohibited

Table 1-8. HSM respond host register (Offset 0x20)

Bit	Read/Write	Description
31		
.....		
28	rc_w1	boot parallel disable 0:boot parallel enable 1:boot parallel disable
27	rc_w1	hsm image check enable 0: secure boot disable 1:secure boot enable
26	rc_w1	fast standby disable 0:fast standby enable 1:fast standby disable
25	rc_w1	hsm recovery mode disable 0:hsm recovery mode enable 1:hsm recovery mode disable
24	rc_w1	host recovery mode disable 0:host recovery mode enable 1:host recovery mode disable
...		
6		
5		
4		
3		
2		
1		
0 CMD state	r	0: no cmd or cmd on process 1: CMD finished

Table 1-9. HSM status register (Offset 0x40)

Bit	Read/Write	Description
28-31	host is readable in no-sec boot	Major version
24-27	host is readable in no-sec boot	Minor version
20-23	host is readable in no-sec boot	Patch version
.....		
8-10 CMD error flag	r	0x0: CMD_NO_ERROR 0x1: CMD_CODE_ERROR 0x2: CMD_LC_AD_INVALID 0x3: CMD_KEY_INVALID 0x4: CMD_LC_AD_NO_AUTH 0x5: CMD_CONFIG_DATA_ERROR 0x7: CMD_GENERAL_ERROR
4-7 Boot error flag	r	0x0: BOOT_NO_ERROR 0x6: BOOT_NO_INVALID_IVT 0x7: BOOT_MULTI_RESET 0x8: BOOT_FW_CHECK_ERROR 0xF: BOOT_GENERAL_ERROR
3		
2 Parallel host boot state, valid only in sec-boot	r	Indicate if host boot ok when parallel mode is selected in hsm option byte. It is the hsm firmware's responsibility to handle this error condition 0: host boot ok 1: host boot failed
0-1 Boot state	r	Indicate if system boot successfully. When hsm is disabled, these bits indicate if host boot successfully, while hsm is enabled, these bits indicate if hsm boot successfully. 00: boot code running 01: boot complete successfully 10: recovery mode entered 11: fast standby boot

1.5. System configuration controller (SYSCFG)

The system configuration controller main functions are the following:

- Remapping of some I/O ports
- I2C Fm+ configuration
- Selection of the Ethernet PHY interface.
- Management of the external interrupt line connection to the GPIOs
- Management timer break input selection
- Exception of floating point unit monitor and interrupt control
- Support core processor configuration information software accessible
- Interrupt router core processor control
- Memory access execution control

1.6. Bus interconnect matrix (BUSIM)

1.6.1. Overview

The bus matrix connects other AHB components through its slave and master ports. The bus matrix provides access from a master to a slave, enabling concurrent access and efficient operation even when several high-speed peripherals work simultaneously. A 64/32-bit multilayer bus is implemented in the devices, which enables parallel access paths between multiple masters and slaves in the system. The multilayer bus consists of an AHB interconnect matrix, two AHB bus. The interconnection relationship of the AHB interconnect matrix is shown in *Table 1-10. The interconnection relationship of the AHB_BM1 of GD32A711/A712* and *Table 1-11. The interconnection relationship of the AHB_BM2 of GD32A711/A712*. In the following tables, "1" indicates the corresponding master is able to access the corresponding slave through the AHB interconnect matrix, while the blank means the corresponding master cannot access the corresponding slave through the AHB interconnect matrix.

Table 1-10. The interconnection relationship of the AHB_BM1 of GD32A711/A712 devices

	CM7_0	DMA0_MEM	DMA1_MEM	HSM	DMA0_PERIPH	DMA1_PERIPH
FMC-0	1					
FMC-2		1	1		1	1
SRAM0	1	1	1	1	1	1
TCMSRAM	1	1	1	1	1	1
QSPI	1	1	1	1	1	1

Table 1-11. The interconnection relationship of the AHB_BM2 of GD32A711/A712

devices

	HSM	DMA0_MEM	DMA1_MEM	AHBP(CM7_0)	DMA0_PERIPH	DMA1_PERIPH
AHB1	1	1	1	1	1	1
AHB2	1	1	1	1	1	1
APB1	1	1	1	1	1	1
APB2	1	1	1	1	1	1

As is shown on the above tables, there are several masters connected with the AHB interconnect matrix, including CM7_0, DMA0, DMA1, HSM.

CM7_0 is the main bus of the Cortex®-M7 core.

DMA bus connects the AHB master interface of the DMA to the bus matrix that manages the access of CPU and DMA to SRAM, Flash memory and AHB/APB peripherals. DMAx_MEM is the master for DMA accessing the memory space, and DMAx_PERIPH is the master for DMA accessing the peripheral space.

There are also several slaves connected with the AHB interconnect matrix, including FMC, SRAM, AHB1, AHB2.

FMC is the bus interface of the flash memory controller. FMC_0 is initiated by CM7_0, and FMC_2 is initiated by other master.

SRAM is on-chip static random access memories.

AHB1 is the AHB bus connected with all of the AHB1 slaves and AHB-to-APB bridges. AHB2 is the AHB bus connected with AHB2 slaves. While AHB-to-APB bridges are the two APB buses connected with all of the APB slaves. The two APB buses connect with all the APB peripherals. APB1 is limited to 80Mhz, APB2 is limited to 80Mhz for GD32A711/A712 devices.

The master id and slave id are shown in ***Table 1-12. Master ID of GD32A711/A712 devices*** and ***Table 1-13. Slaver ID of GD32A711/A712 devices***. Each master's port will check the slave ID. If it conflicts with the bus matrix access attribute, an error will be reported to the FMU.

Table 1-12. Master ID of GD32A711/A712 devices

Module	Master ID
AXIM (CM7_0)	4'b0000
DMA0_MEM	4'b0010
DMA1_MEM	4'b0011
HSM_CM23	4'b0100
DMA0_PEIRPH	4'b1010
DMA1_PEIRPH	4'b1011
HSM_DMA	4b'1100
AHBP (CM7_0)	4'b1110

Table 1-13. Slaver ID of GD32A711/A712 devices

Module	Slaver ID
FMC-0	4'b0000
FMC-2	4'b0010
SRAM	4'b0011
TCMSRAM	4'b0101
QSPI	4'b0110
AHB1	4'b1000
AHB2	4'b1001
APB1	4'b1010
APB2	4'b1011

1.6.2. Arbitration algorithm

The bus matrix also manages the access arbitration between masters. The arbitration uses a Round-Robin algorithm and fixed priority algorithm. The arbitration algorithm is selected in the ARBMOD of BUSIM_BM1CTLx and BUSIM_BM2CTLx.

Round-Robin algorithm

In this mode, the master will be assigned a priority based on its port number, then compare the priority with the port number of the master that performed the last transfer on the slave bus. The master with highest priority will be granted to access of the slave bus. When the bus is locked or a fixed length burst transmission is in progress, the master will own the slave bus after the next transmission is completed.

Default access does not affect the round-robin mode unless the master with default access performs a transfer. After one arbitration cycle, the next master in line switches.

Fixed priority algorithm

In this mode, the priority of each master is defined in slave port priority configuration register (BUSIM_BM1SPCFGx, x = 0...4 and BUSIM_BM2SPCFGx, x = 0...3). The master with highest priority will gain control over the slave port if different masters request access to the same slave port. If there is a request from master to slave port, the slave port checks the priority of the new requesting master and the arbitration check is performed at every clock edge. The new master will have control of the slave port if its priority is higher than the master currently controlling the slave port, unless the slave port is in the default access state.

Note: A unique priority is assigned for each master port. If different masters are configured with the same priority within the slave port priority configuration register (BUSIM_BM1SPCFGx, x = 0...4 and BUSIM_BM2SPCFGx, x = 0...3), a bus error will be responded by bus matrix and the incorrect configuration will not be successfully configured in the register.

1.7. Extended memory protection unit (EXTMPU)

1.7.1. Overview

Extended memory protection unit (EXTMPU) provides an access protection mechanism for system bus memory references and independent control for each region, including memory space definition and access rights.

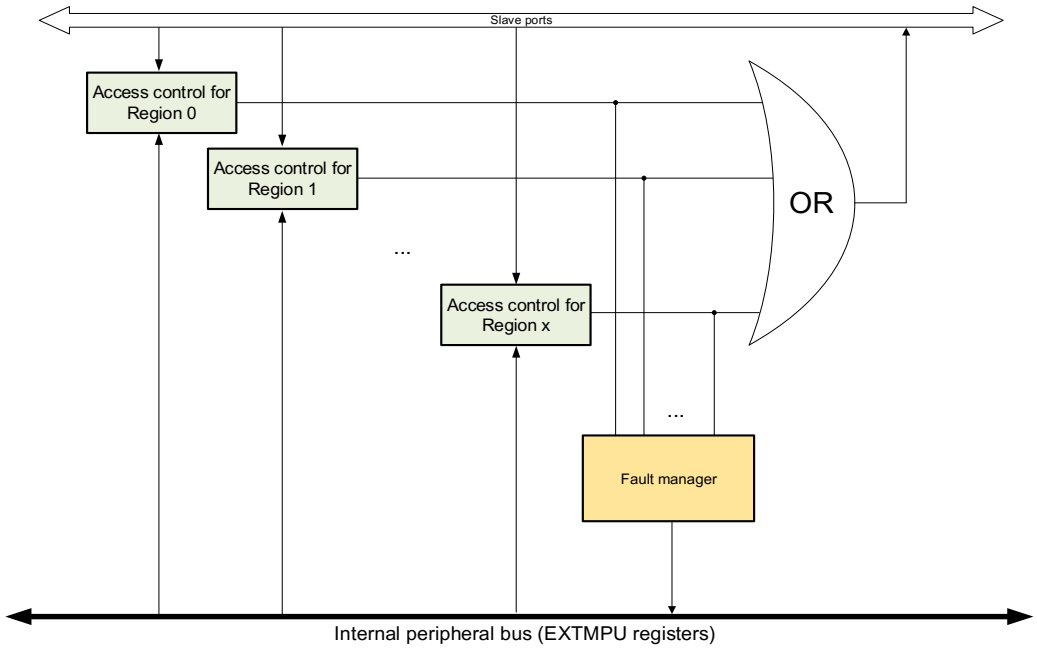
1.7.2. Key features:

- Supports up to 32 program-visible 128-bit region descriptors, accessible as four 32-bit words each.
  - Each region descriptor defines an arbitrarily sized space, aligned anywhere in
- Memory Region sizes can vary from a minimum of 128 bytes to a maximum of (4 GB minus 1 byte).
- Read/write access control permissions defined in region descriptor
  - privilege access control permissions defined in region descriptor
  - Hardware-assisted maintenance of the descriptor valid bit minimizes coherency issues
  - Priority given to granting permission over denying access for overlapping region descriptors
- Error registers (per bus master ID) capture the first faulting address, attributes, and other information.
- Global EXTMPU enable/disable control bit
- Region configuration register read only control bit
- Region privilege access limit bit.(read and write distinctions restrictions)
- Region valid bit for each region.

1.7.3. Function overview

EXTMPU processes the address signals of slave ports and internal peripheral bus. The block diagram of extended memory protection unit (EXTMPU) is shown in ***Figure 1-8. The block diagram of EXTMPU***.

Figure 1-8. The block diagram of EXTMPU



The access control module receives and processes the address phase signals of slave port and control signals from the EXTMPU registers. If there is a hit failure or access permission violation, the fault manager module will output error information to the EXTMPU registers.

The master id and slave id are shown in ***Table 1-12. Master ID of GD32A711/A712 devices*** and ***Table 1-13. Slaver ID of GD32A711/A712 devices.***

Memory access without match will be denied, and if multiple regions are hit, access will not be allowed.

1.8. Peripheral register protection unit (PRPU)

1.8.1. Overview

Peripheral register protection unit (PRPU) provides the register unlock protecting function of data transferring in slave ports (including AHB and APB slave). Every peripheral of slave ports owns a dedicated unlock key to protect configurable registers. The unlock key is accessed by AHB bus. Only when corresponding peripheral unlock key is opened, data would be transferred correctly.

1.8.2. Characteristics

- Writing protect key is accessed by AHB bus.
- Every module in AHB1/AHB2/APB1/APB2 slave has dedicated unlock key.
- Lock protection register is accessed by word only.
- When a write-access tries to configure registers without writing an unlock key, an error indication will be generated and refused access.

- The bus fault will be generated when an illegal write-access.
- Reset value is 1 and the default state is unlocked state.
- Each peripheral register has an independent unlock key.

1.8.3. Lock and unlock operation

There is a dedicated lock protection register for each module, to protect all configurable registers against unintended write-access.

Allowable write-access follows steps are shown below:

1. Firstly write a matched key into lock protection register, then the lock protection of registers is unlocked.
2. Then perform the intended write-access to any configurable registers, it will be successful.
3. Finally write a different key into lock protection register, then the lock protection of registers locked again.
4. Re-perform a write-access to any configurable registers, it will be refused and asserts a bus error indication.

The relationship between unlock key and corresponding peripheral registers is shown in the table ***Table 1-14. The memory map of the unlock key and corresponding peripheral registers of GD32A711/A712 devices***.

Table 1-14. The memory map of the unlock key and corresponding peripheral registers of GD32A711/A712 devices

Register offset	Peripheral name	No.	Unlock key
0x0	FMU	1	0xA1B2C4
0x4	IRM	2	0xA1B2C5
0x8	-	3	-
0xC	STCM	4	0xA1B2C7
0x10	MTC	5	0xA1B2C8
0x14	DMA0	6	0xA1B2C9
0x18	DMA1	7	0xA1B2CA
0x1C	SRAM_REG	8	0xA1B2CB
0x20	FMC	9	0xA1B2CC
0x24	CRC	10	0xA1B2CD
0x28	GPIOA	11	0xA1B2CE
0x2C	GPIOB	12	0xA1B2CF
0x30	GPIOC	13	0xA1B2D0
0x34	GPIOD	14	0xA1B2D1
0x38	GPIOE	15	0xA1B2D2
0x3C	GPIOF	16	0xA1B2D3
0x40	GPIOG	17	0xA1B2D4
0x44	GPIOH	18	0xA1B2D5
0x48	GPIOI	19	0xA1B2D6
0x4C	GPIOJ	20	0xA1B2D7

Register offset	Peripheral name	No.	Unlock key
0x50	GPIOK	21	0xA1B2D8
0x54	GPIOL	22	0xA1B2D9
0x58	GPIOM	23	0xA1B2DA
0x5C	GPION	24	0xA1B2DB
0x60	DMAMUX	25	0xA1B2DC
0x64	RCU_CCTL	26	0xA1B2DD
0x68	QSPI_REG	27	0xA1B2DE
0x6C	-	28	-
0x70	TCM_REG	29	0xA1B2E0
0x74	-	30	-
0x78	-	31	-
0x7C	-	32	-
0x80	HSMIF	33	0xA1B2E4
0x84	RCU_RCTL	34	0xA1B2E5
0x88	TIMER2	35	0xA1B2E6
0x8C	WWDGT0	36	0xA1B2E7
0x90	DAC0	37	0xA1B2E8
0x94	DAC1	38	0xA1B2E9
0x98	-	39	-
0x9C	TIMER20	40	0xA1B2EB
0xA0	TIMER60	41	0xA1B2EC
0xA4	TIMER6	42	0xA1B2ED
0xA8	RTC	43	0xA1B2EE
0xAC	-	44	-
0xB0	-	45	-
0xB4	-	46	-
0xB8	-	47	-
0xBC	I2C0	48	0xA1B2F3
0xC0	I2C1	49	0xA1B2F4
0xC4	TIMER61	50	0xA1B2F5
0xC8	TIMER63	51	0xA1B2F6
0xCC	-	52	-
0xD0	-	53	-
0xD4	-	54	-
0xD8	-	55	-
0xDC	-	56	-
0xE0	-	57	-
0xE4	-	58	-
0xE8	-	59	-
0xEC	SPI2	60	0xA1B2FF
0xF0	SPI3	61	0xA1B300
0xF4	FWDGT0	62	0xA1B301

Register offset	Peripheral name	No.	Unlock key
0xF8	CLTCFG	63	0xA1B302
0xFC	PMU	64	0xA1B303
0x100	ADC0	65	0xA1B304
0x104	ADC1	66	0xA1B305
0x108	-	67	-
0x10C	TIMER1	68	0xA1B307
0x110	SYSCFG	69	0xA1B308
0x114	LINFlexD0	70	0xA1B309
0x118	LINFlexD1	71	0xA1B30A
0x11C	LINFlexD2	72	0xA1B30B
0x120	LINFlexD3	73	0xA1B30C
0x124	LINFlexD4	74	0xA1B30D
0x128	LINFlexD5	75	0xA1B30E
0x12C	TRIGSEL	76	0xA1B30F
0x130	TIMER0	77	0xA1B310
0x134	SPI0	78	0xA1B311
0x138	TIMER7	79	0xA1B312
0x13C	-	80	-
0x140	-	81	-
0x144	-	82	-
0x148	SPI4/I2S4	83	0xA1B316
0x14C	SPI5/I2S5	84	0xA1B317
0x150	TIMER19	85	0xA1B318
0x154	TIMER5	86	0xA1B319
0x158	TIMER62	87	0xA1B31A
0x15C	-	88	-
0x160	SPI1	89	0xA1B31C
0x164	CAN0 cluster	90	0xA1B31D
0x168	CAN1 cluster	91	0xA1B31E
0x16C	CAN2 cluster	92	0xA1B31F
0x170	-	93	-
0x174	-	94	-
0x178	-	95	-
0x17C	-	96	-
0x180	-	97	-
0x184	EXTI	98	0xA1B325
0x188	CMP1	99	0xA1B326
0x18C	CMP0	100	0xA1B327
0x190	-	101	-
0x194	BKP	102	0xA1B329
0x198	HOST_JDC	103	0xA1B32A
0x19C	DBGSYS	104	0xA1B32B

Register offset	Peripheral name	No.	Unlock key
0x1A0	CPDM	105	0xA1B32C

1.9. Interconnect matrix integrity check unit (IMICU)

1.9.1. Overview

Interconnect matrix integrity check unit (IMICU) mainly provides the calculation and check of EDC values for the attribute information. There are 2 instances in this series which are AHB_BM1 and AHB_BM2.

1.9.2. Characteristics

- Verification of the attribute information and feedback information.
- EDC (Error Detection Code) detects single- / double-bit errors.
- Error injection function for testing.
- Programmable integrity check function for master and slave port

1.9.3. Function overview

IMICU is mainly used for the integrity check of the transfers of interconnect matrix. The integrity check of the signal from master to slave is done by using a 7-bit Error Detection Code (EDC) and EDC detects any single- / double-bit error. The block diagram of IMICU is shown in ***Figure 1-9. The block diagram of IMICU for BM1 master write***, ***Figure 1-10. The block diagram of IMICU for BM1 master read***, ***Figure 1-11. The block diagram of IMICU for BM2 master write*** and ***Figure 1-12. The block diagram of IMICU for BM2 master read***.

Figure 1-9. The block diagram of IMICU for BM1 master write

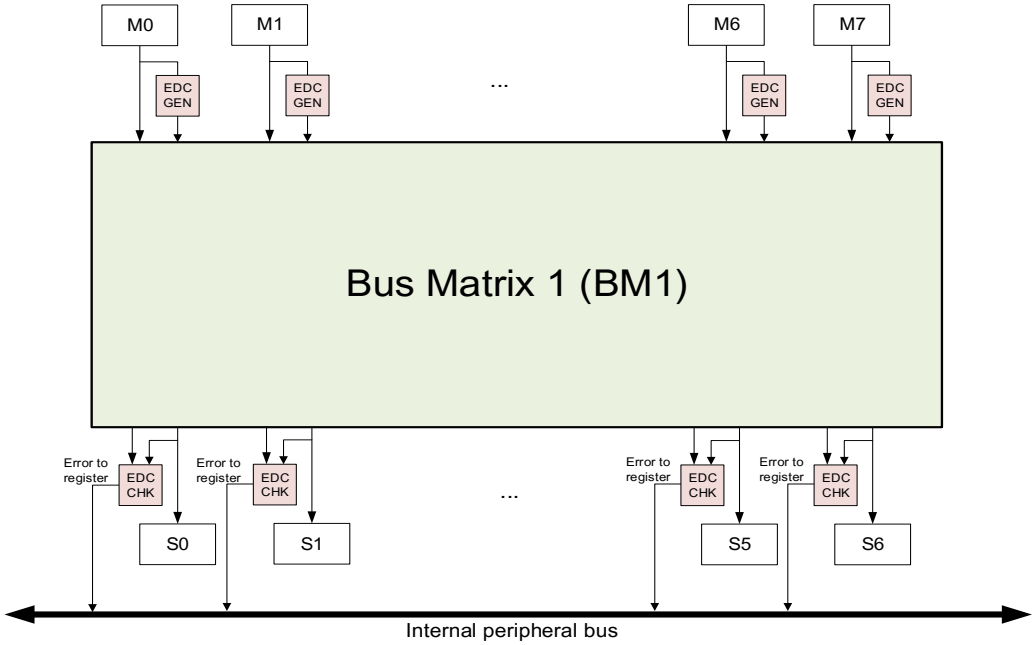


Figure 1-10. The block diagram of IMICU for BM1 master read

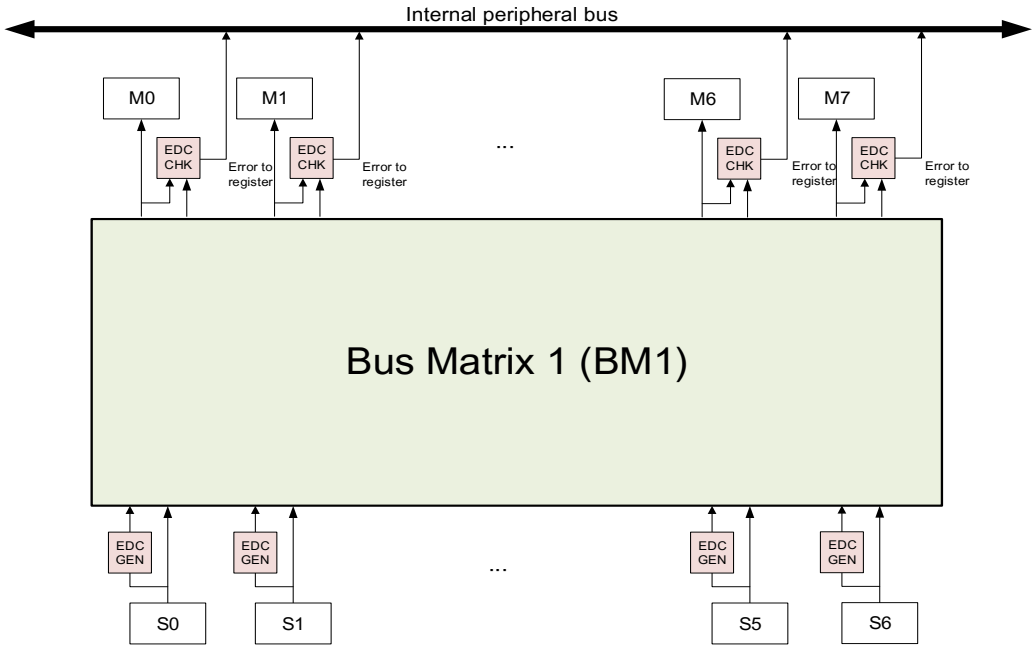


Figure 1-11. The block diagram of IMICU for BM2 master write

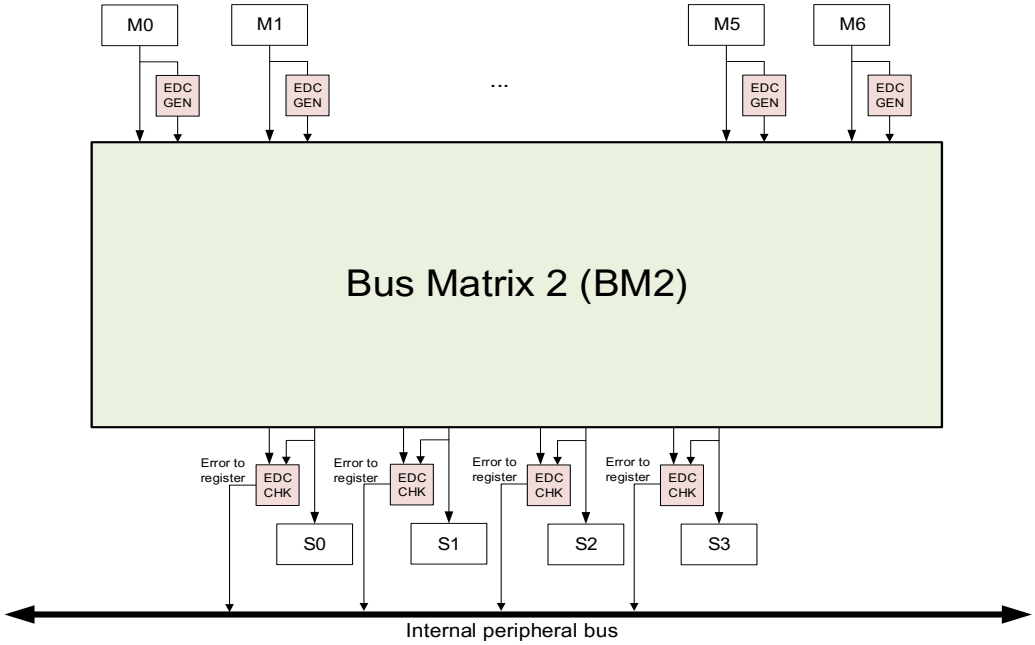
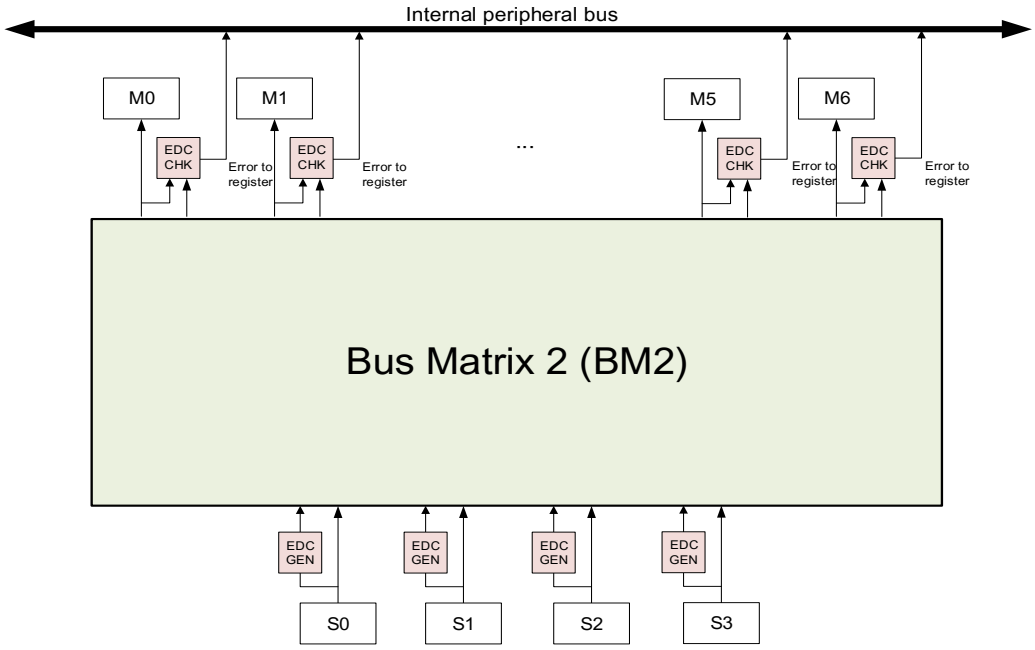


Figure 1-12. The block diagram of IMICU for BM2 master read



The IMICU generates a 7-bit EDC checkbit for every transfer. During the data phase of the system bus pipelined protocol, the IMICU compares the three signals of hready, hresp0, hresp1 from entering the crossbar on the slave side to leaving the crossbar on the master side. Data phase is often more than one clock cycle due to wait cycles. All three signals are compared clock by clock during the whole data cycle. The value shown in ***Table 1-15. Hexadecimal EDC code*** helps decode single-bit error reported in EDCC_DET[6:0] in IMICU_ERRSTAT register.

Table 1-15. Hexadecimal EDC code

signal	DET	signal	DET	signal	DET	signal	DET
0	0x23	1	0x25	2	0x26	3	0x29
4	0x2A	5	0x2C	6	0x31	7	0x34
8	0x0E	9	0x0B	10	0x13	11	0x15
12	0x16	13	0x19	14	0x1A	15	0x1C
16	0x4F	17	0x4A	18	0x52	19	0x54
20	0x57	21	0x58	22	0x5B	23	0x5D
24	0x62	25	0x64	26	0x67	27	0x68
28	0x6B	29	0x6D	30	0x70	31	0x75
EDC[0]	0x1	EDC[1]	0x2	EDC[2]	0x4	EDC[3]	0x8
EDC[4]	0x10	EDC[5]	0x20	EDC[6]	0x40		

1.10. System configuration registers definition

SYSCFG base address: 0x4001 0000

Table 1-16. SYSCFG register overview

Register name	Symbol	Address offset	Reset value
Peripheral mode configuration register	SYSCFG_PMCFG	0x04	0x0000 0000
EXTI sources selection register 0	SYSCFG_EXTISS0	0x08	0x0000 0000
EXTI sources selection register 1	SYSCFG_EXTISS1	0x0C	0x0000 0000
EXTI sources selection register 2	SYSCFG_EXTISS2	0x10	0x0000 0000
EXTI sources selection register 3	SYSCFG_EXTISS3	0x14	0x0000 0000
System status register	SYSCFG_STAT	0x20	0x0000 0000
System configuration register	SYSCFG_CFG	0x24	0x0028 0000
Timer break input select register	SYSCFG_TIMERBKINSEL	0x2C	0x0000 0000
CPU0 ICACHE error status register	SYSCFG_CPU0ICAC	0x30	0x0000 0000
CPU0 DCACHE error status register	SYSCFG_CPU0DCAC	0x34	0x0000 0000
CPU control register	SYSCFG_CPUCTL	0x50	0x0000 0000
CPU platform status register	SYSCFG_CPU0CS	0x58	0x7C20 XXX0
SRAM configuration register	SYSCFG_SRAMCFG	0x68	0x0000 0000
Reset Mode configuration register	SYSCFG_RMCFG	0x6C	0x0000 0000
TIMERx configuration register 0	SYSCFG_TIMERxCFG0	0x16C for TIMER63	0x0000 0000
TIMERx configuration register 1	SYSCFG_TIMERxCFG1	0x170 for TIMER63	0x0000 0000
TIMERx configuration register 2	SYSCFG_TIMERxCFG2	0x174 for TIMER63	0x0000 0000
TIMERx configuration register 0	SYSCFG_TIMERxCFG0	0x118 for TIMER2	0x0000 0000
TIMERx configuration register 1	SYSCFG_TIMERxCFG1	0x11C for TIMER2	0x0000 0000
TIMERx configuration register 2	SYSCFG_TIMERxCFG2	0x120 for TIMER2	0x0000 0000
Core processor ID register	SYSCFG_CP0ID	0x240	0x434D 3730
Core porcessor number register	SYSCFG_CP0NUM	0x244	0x0000 0000
Core processor configuration information register 0	SYSCFG_CP0CFGINF0	0x24C	0x0502 0504
Core processor configuration information register 1	SYSCFG_CP0CFGINF1	0x254	0xXXXX 0000
Core processor configuration information register 2	SYSCFG_CP0CFGINF2	0x258	0x0000 000B
Memory access execution control register	SYSCFG_MEMAEXCTL	0x350	0x4000 0000
Bus matrix 1 interconnect error detection enable register	SYSCFG_BM1EDCEN	0x360	0x0000 0000
Bus matrix 2 interconnect error detection enable register	SYSCFG_BM2EDCEN	0x364	0x0000 0000

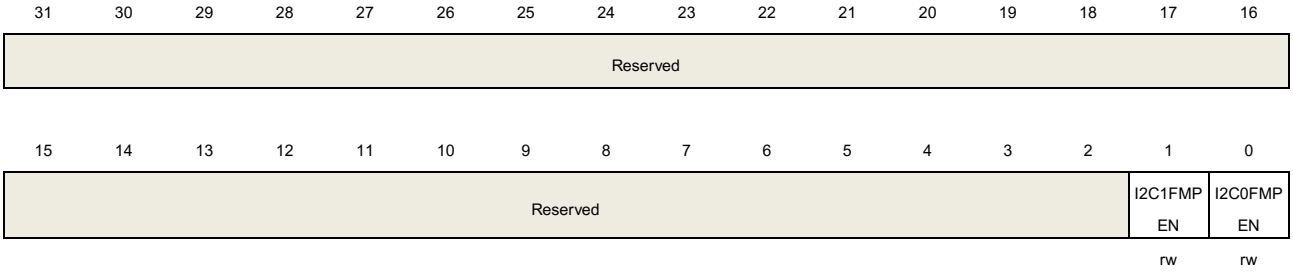
Register name	Symbol	Address offset	Reset value
Bus check configuration register	SYSCFG_BUSCHKCFG	0x368	0x0000 0000
TMR fault control and status register	SYSCFG_TMRFCS	0x370	0x0000 0007

1.10.1. Peripheral mode configuration register (SYSCFG_PMCFG)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



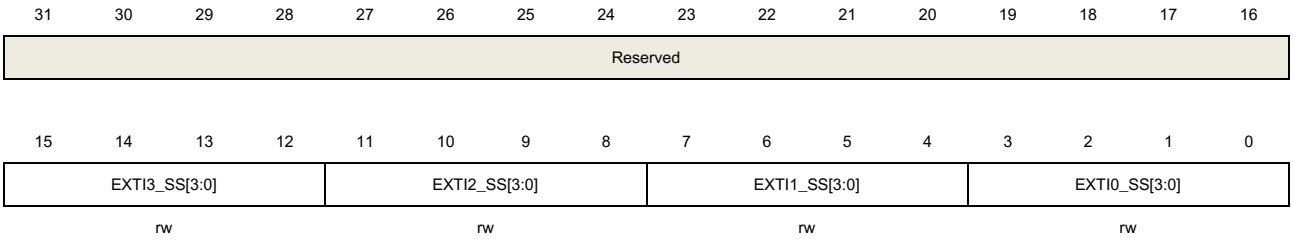
Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	I2C1FMPEN	I2C1 Fm+ This bit enables Fm+ on I2C1. 0: Fm+ disabled 1: Fm+ enabled
0	I2C0FMPEN	I2C0 Fm+ This bit enables Fm+ on I2C0. 0: Fm+ disabled 1: Fm+ enabled

1.10.2. EXTI sources selection register 0 (SYSCFG_EXTISS0)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	EXTI3_SS[3:0]	EXTI 3 sources selection 0000: PA3 pin 0001: PB3 pin 0010: PC3 pin 0011: PD3 pin 0100: PE3 pin 0101: PF3 pin 0110: PG3 pin 0111: PH3 pin 1000: PI3 pin 1001: PJ3 pin 1010: PK3 pin 1011: PL3 pin 1100: PM3 pin 1101: PN3 pin
11:8	EXTI2_SS[3:0]	EXTI 2 sources selection 0000: PA2 pin 0001: PB2 pin 0010: PC2 pin 0011: PD2 pin 0100: PE2 pin 0101: PF2 pin 0110: PG2 pin 0111: PH2 pin 1000: PI2 pin 1001: PJ2 pin 1010: PK2 pin 1011: PL2 pin 1100: PM2 pin 1101: PN2 pin
7:4	EXTI1_SS[3:0]	EXTI 1 sources selection 0000: PA1 pin 0001: PB1 pin 0010: PC1 pin 0011: PD1 pin 0100: PE1 pin 0101: PF1 pin 0110: PG1 pin 0111: PN1 pin 1000: PI1 pin 1001: PJ1 pin

		1010: PK1 pin 1011: PL1 pin 1100: PM1 pin 1101: PN1 pin
3:0	EXTI0_SS[3:0]	EXTI 0 sources selection 0000: PA0 pin 0001: PB0 pin 0010: PC0 pin 0011: PD0 pin 0100: PE0 pin 0101: PF0 pin 0110: PG0 pin 0111: PN0 pin 1000: PI0 pin 1001: PJ0 pin 1010: PK0 pin 1011: PL0 pin 1100: PM0 pin 1101: PN0 pin

1.10.3. EXTI sources selection register 1 (SYSCFG_EXTISS1)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTI7_SS[3:0]				EXTI6_SS[3:0]				EXTI5_SS[3:0]				EXTI4_SS[3:0]			
rw				rw				rw				rw			

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	EXTI7_SS[3:0]	EXTI 7 sources selection 0000: PA7 pin 0001: PB7 pin 0010: PC7 pin 0011: PD7 pin 0100: PE7 pin 0101: PF7 pin

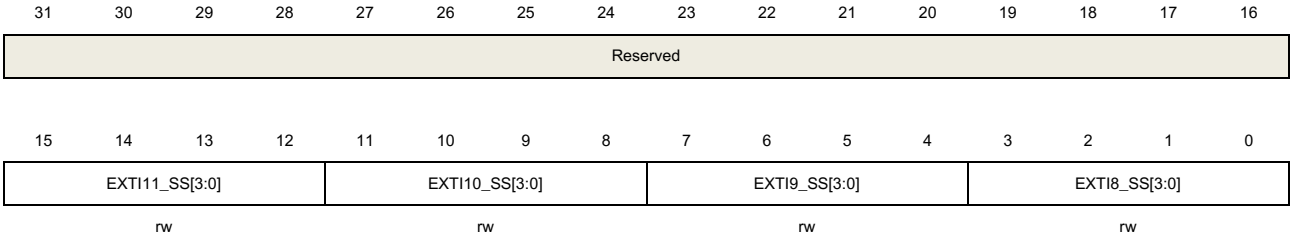
		0110: PG7 pin 0111: PH7 pin 1000: PI7 pin 1001: PJ7 pin 1010: PK7 pin 1011: PL7 pin 1100: PM7 pin 1101: PN7 pin
11:8	EXTI6_SS[3:0]	EXTI 6 sources selection 0000: PA6 pin 0001: PB6 pin 0010: PC6 pin 0011: PD6 pin 0100: PE6 pin 0101: PF6 pin 0110: PG6 pin 0111: PH6 pin 1000: PI6 pin 1001: PJ6 pin 1010: PK6 pin 1011: PL6 pin 1100: PM6 pin 1101: PN6 pin
7:4	EXTI5_SS[3:0]	EXTI 5 sources selection 0000: PA5 pin 0001: PB5 pin 0010: PC5 pin 0011: PD5 pin 0100: PE5 pin 0101: PF5 pin 0110: PG5 pin 0111: PH5 pin 1000: PI5 pin 1001: PJ5 pin 1010: PK5 pin 1011: PL5 pin 1100: PM5 pin 1101: PN5 pin
3:0	EXTI4_SS[3:0]	EXTI 4 sources selection 0000: PA4 pin 0001: PB4 pin 0010: PC4 pin 0011: PD4 pin

0100: PE4 pin
0101: PF4 pin
0110: PG4 pin
0111: PH4 pin
1000: PI4 pin
1001: PJ4 pin
1010: PK4 pin
1011: PL4 pin
1100: PM4 pin
1101: PN4 pin

1.10.4. EXTI sources selection register 2 (SYSCFG_EXTISS2)

Address offset: 0x10
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	EXTI11_SS[3:0]	EXTI 11 sources selection 0000: PA11 pin 0001: PB11 pin 0010: PC11 pin 0011: PD11 pin 0100: PE11 pin 0101: PF11 pin 0110: PG11 pin 0111: PH11 pin 1000: PI11 pin 1001: PJ11 pin 1010: PK11 pin 1011: PL11 pin 1100: PM11 pin 1101: PN11 pin
11:8	EXTI10_SS[3:0]	EXTI 10 sources selection

0000: PA10 pin
0001: PB10 pin
0010: PC10 pin
0011: PD10 pin
0100: PE10 pin
0101: PF10 pin
0110: PG10 pin
0111: PH10 pin
1000: PI10 pin
1001: PJ10 pin
1010: PK10 pin
1011: PL10 pin
1100: PM10 pin
1101: PN10 pin

7:4	EXTI9_SS[3:0]	EXTI 9 sources selection 0000: PA9 pin 0001: PB9 pin 0010: PC9 pin 0011: PD9 pin 0100: PE9 pin 0101: PF9 pin 0110: PG9 pin 0111: PH9 pin 1000: PI9 pin 1001: PJ9 pin 1010: PK9 pin 1011: PL9 pin 1100: PM9 pin 1101: PN9 pin
3:0	EXTI8_SS[3:0]	EXTI 8 sources selection 0000: PA8 pin 0001: PB8 pin 0010: PC8 pin 0011: PD8 pin 0100: PE8 pin 0101: PF8 pin 0110: PG8 pin 0111: PH8 pin 1000: PI8 pin 1001: PJ8 pin 1010: PK8 pin 1011: PL8 pin 1100: PM8 pin

1101: PN8 pin

1.10.5. EXTI sources selection register 3 (SYSCFG_EXTISS3)

Address offset: 0x14

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTI15_SS[3:0]				EXTI14_SS[3:0]				EXTI13_SS[3:0]				EXTI12_SS[3:0]			
rw				rw				rw				rw			

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:12	EXTI15_SS[3:0]	EXTI 15 sources selection 0000: PA15 pin 0001: PB15 pin 0010: PC15 pin 0011: PD15 pin 0100: PE15 pin 0101: PF15 pin 0110: PG15 pin 0111: PH15 pin 1000: PI15 pin 1001: PJ15 pin 1010: PK15 pin 1011: PL15 pin 1100: PM15 pin
11:8	EXTI14_SS[3:0]	EXTI 14 sources selection 0000: PA14 pin 0001: PB14 pin 0010: PC14 pin 0011: PD14 pin 0100: PE14 pin 0101: PF14 pin 0110: PG14 pin 0111: PH14 pin 1000: PI14 pin 1001: PJ14 pin

		1010: PK14 pin 1011: PL14 pin 1100: PM14 pin
7:4	EXTI13_SS[3:0]	EXTI 13 sources selection 0000: PA13 pin 0001: PB13 pin 0010: PC13 pin 0011: PD13 pin 0100: PE13 pin 0101: PF13 pin 0110: PG13 pin 0111: PH13 pin 1000: PI13 pin 1001: PJ13 pin 1010: PK13 pin 1011: PL13 pin 1100: PM13 pin
3:0	EXTI12_SS[3:0]	EXTI 12 sources selection 0000: PA12 pin 0001: PB12 pin 0010: PC12 pin 0011: PD12 pin 0100: PE12 pin 0101: PF12 pin 0110: PG12 pin 0111: PH12 pin 1000: PI12 pin 1001: PJ12 pin 1010: PK12 pin 1011: PL12 pin 1100: PM12 pin

1.10.6. System status register (SYSCFG_STAT)

Address offset: 0x20

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	FMUNMIIF	NMIPINIF	CKMNMIIF	Reserved
	r	r	rc_w1	

Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5	FMUNMIIF	NMI interrupt flag from FMU. 0: No FMU monitor error 1: FMU NMI interrupt is detected.
4	NMIPINIF	NMI interrupt flag from EXTI pin 0: No input 1: NMI interrupt input from EXTI
3	CKMNMIIF	HXTAL clock monitor NMI interrupt flag. The software can clear it by writing 1. It can be cleared by writing 1 to CKMIC bit in the RCU_INT register. 0: No HXTAL clock monitor error 1: HXTAL clock monitor is detected.
2:0	Reserved	Must be kept at reset value.

1.10.7. System configuration register (SYSCFG_CFG)

Address offset: 0x24

Reset value: 0x0028 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							CM23_TXEVEN	Reserved		FMUNMIIE0	NMIPINIE0	CKMNMIIE0	Reserved		CM7_0_TXEVEN
							rw			rw	rw	rw			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits	Fields	Descriptions
31:25	Reserved	Must be kept at reset value.
24	CM23_TXEVEN	This bit can be used to enable the EVENTOUT from CM23. 0: Disable 1: Enable
23:22	Reserved	Must be kept at reset value.
21	FMUNMIIE0	This bit can be used to enable the interrupt flag from FMU connection to CPU0 NMI

		interrupt. 0: Disable 1: Enable
20	NMIPINIE0	This bit can be used to enable the interrupt flag from EXTI connection to CPU0 NMI interrupt. 0: Disable 1: Enable
19	CKMNMIIE0	This bit can be used to enable the HXTAL clock monitor NMI interrupt flag connection to CPU0 NMI interrupt. 0: Disable 1: Enable
18:17	Reserved	Must be kept at reset value.
16	CM7_0_TXEVEN	This bit can be used to enable the EVENTOUT from CM7_0. 0: Disable 1: Enable
15:0	Reserved	Must be kept at reset value.

1.10.8. Timer break input select register (SYSCFG_TIMERBKINSEL)

Address offset: 0x2C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIMER0_BRKIN0_SEL	TIMER0_BRKIN1_SEL	TIMER0_BRKIN2_SEL	TIMER0_BRKIN3_SEL	TIMER7_BRKIN0_SEL	TIMER7_BRKIN1_SEL	TIMER7_BRKIN2_SEL	TIMER7_BRKIN3_SEL	TIMER19_BRKIN0_SEL	TIMER19_BRKIN1_SEL	TIMER19_BRKIN2_SEL	TIMER19_BRKIN3_SEL	TIMER20_BRKIN0_SEL	TIMER20_BRKIN1_SEL	TIMER20_BRKIN2_SEL	TIMER20_BRKIN3_SEL
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMER60_BRKIN0_SEL	TIMER60_BRKIN1_SEL	TIMER60_BRKIN2_SEL	TIMER60_BRKIN3_SEL	TIMER61_BRKIN0_SEL	TIMER61_BRKIN1_SEL	TIMER61_BRKIN2_SEL	TIMER61_BRKIN3_SEL	TIMER62_BRKIN0_SEL	TIMER62_BRKIN1_SEL	TIMER62_BRKIN2_SEL	TIMER62_BRKIN3_SEL	TIMER63_BRKIN0_SEL	TIMER63_BRKIN1_SEL	TIMER63_BRKIN2_SEL	TIMER63_BRKIN3_SEL
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31/30/29/28	TIMER0_BRKINx_SEL	Timer0 break input x select (x=0...3) 0: from GPIO pin 1: from TRIGSEL
27/26/25/24	TIMER7_BRKINx_SEL	Timer7 break input x select (x=0...3) 0: from GPIO pin 1: from TRIGSEL

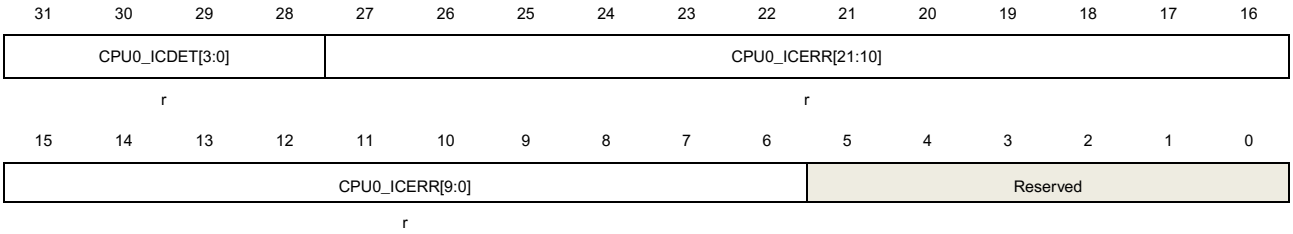
23/22/21/20	TIMER19_BRKINx_SEL	Timer19 break input x select (x=0...3) 0: from GPIO pin 1: from TRIGSEL
19/18/17/16	TIMER20_BRKINx_SEL	Timer20 break input x select (x=0...3) 0: from GPIO pin 1: from TRIGSEL
15/14/13/12	TIMER60_BRKINx_SEL	Timer60 break input x select (x=0...3) 0: from GPIO pin 1: from TRIGSEL
11/10/9/8	TIMER61_BRKINx_SEL	Timer61 break input x select (x=0...3) 0: from GPIO pin 1: from TRIGSEL
7/6/5/4	TIMER62_BRKINx_SEL	Timer62 break input x select (x=0...3) 0: from GPIO pin 1: from TRIGSEL
3/2/1/0	TIMER63_BRKINx_SEL	Timer63 break input x select (x=0...3) 0: from GPIO pin 1: from TRIGSEL

1.10.9. CPU0 ICACHE error status register(SYSCFG_CPU0ICAC)

Address offset: 0x30

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:28	CPU0_ICDET[3:0]	The icache error detection information These bits are provided by the CPU0 to indicate the icache error detection information. Bit3: Fatak error on data RAM read Bit2: Correctable error on data RAM read Bit1: Fatal error on tag RAM read Bit0: Correctable error on tag RAM read
27:6	CPU0_ICERR[21:0]	The icache error bank information

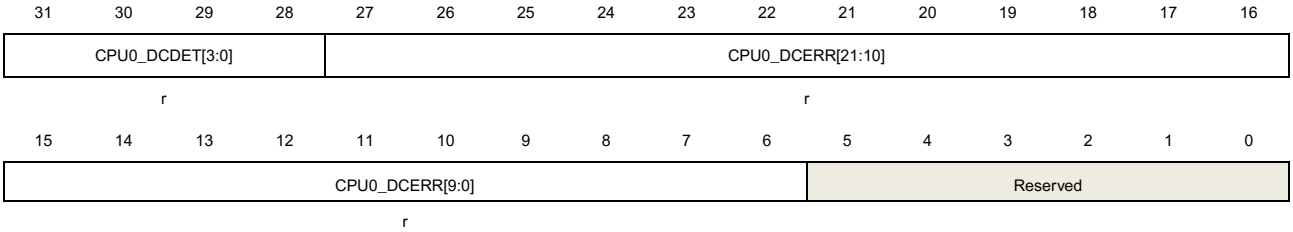
		These bits are provided by the CPU0 to indicate the icache error bank information. Bit21: Indicates whether xEBR1 is locked. 1 represents locked. Bit20: Indicates whether xEBR1 is valid. 1 represents valid. Bit19: Indicates whether xEBR0 is locked. 1 represents locked. No hardware allocation. Bit18: Indicates whether xEBR0 is valid. 1 represents valid. Bit17~Bit2: Data to be allocated. Corresponds to bits[17:2] of data to be written to Error Bank Register. Only valid when xCERR[1:0] is not 0b00. Bit1~Bit0: Allocation status. 0b00 represents No allocation. 0b10 represents Allocation into xEBR0. 0b11 represents Allocation into xEBR1. 0b01 represents Attempted allocation but both entries are locked. Will not allocate. **Note:** refer to the ARM official manual, DDI0489F_cortex_m7_trm.
5:0	Reserved	Must be kept at reset value.

1.10.10. CPU0 DCACHE error status register (SYSCFG_CPU0DCAC)

Address offset: 0x34

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:28	CPU0_DCDET[3:0]	The dcache error detection information These bits are provided by the CPU0 to indicate the dcache error detection information. Bit3: Fatak error on data RAM read Bit2: Correctable error on data RAM read Bit1: Fatal error on tag RAM read Bit0: Correctable error on tag RAM read
27:6	CPU0_DCERR[21:0]	The dcache error bank information These bits are provided by the CPU0 to indicate the dcache error bank information. Bit21: Indicates whether xEBR1 is locked. 1 represents locked. Bit20: Indicates whether xEBR1 is valid. 1 represents valid. Bit19: Indicates whether xEBR0 is locked. 1 represents locked. No hardware allocation. Bit18: Indicates whether xEBR0 is valid. 1 represents valid.

		Bit17~Bit2: Data to be allocated. Corresponds to bits[17:2] of data to be written to Error Bank Register.Only valid when xCERR[1:0] is not 0b00. Bit1~Bit0: Allocation status. 0b00 represents No allocation. 0b10 represents Allocation into xEBR0. 0b11 represents Allocation into xEBR1. 0b01 represents Attempted allocation but both entries are locked. Will not allocate. **Note:** refer to the ARM official manual, DDI0489F_cortex_m7_trm.
5:0	Reserved	Must be kept at reset value.

1.10.11. CPU control register (SYSCFG_CPUCTL)

Address offset: 0x50

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				CPU0_AHBSPRI	Reserved							CPU0_HOLD	Reserved		
				rw								rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27	CPU0_AHBSPRI	AHB slave port priority of the CM7_0 0: Round-robin arbitration 1: AHB slave port has higher access priority than core access
26:20	Reserved	Must be kept at reset value.
19	CPU0_HOLD	CPU0 hold released This field is used to start the host processor (CM7_0) operation. 0: CPU0 held 1: CPU0 released
18:0	Reserved	Must be kept at reset value.

1.10.12. CPU0 control and status register (SYSCFG_CPU0CS)

Address offset: 0x58

Reset value: 0x7C20 XXX0

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIXIE	FIDIE	FOVFIE	FUFIE	FDZIE	FIOPIE	Reserved				WAIE	Reserved				
rw	rw	rw	rw	rw	rw					rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIXIF	FIDIF	FOVFIF	FUFIF	FDZIF	FIOPIF	Reserved			WASOVF	WAS	Reserved				
r	r	r	r	r	r				r	rc_w1					

Bits	Fields	Descriptions
31	FIXIE	FPU inexact interrupt enable 0: Disable 1: Enable
30	FIDIE	FPU input denormal interrupt enable 0: Disable 1: Enable
29	FOVFIE	FPU overflow interrupt enable 0: Disable 1: Enable
28	FUFIE	FPU underflow interrupt enable 0: Disable 1: Enable
27	FDZIE	FPU divide-by-zero interrupt enable 0: Disable 1: Enable
26	FIOPIE	FPU invalid operation interrupt enable 0: Disable 1: Enable
25:22	Reserved	Must be kept at reset value.
21	WAIE	TCM write abort interrupt enable 0: Disable 1: Enable
20:16	Reserved	Must be kept at reset value.
15	FIXIF	FPU inexact interrupt flag status This bit is a copy of the core's IXC bit in FPSCR register. 0: No interrupt 1: Interrupt occurred
14	FIDIF	FPU input denormal interrupt flag status. This bit is a copy of the core's IDC bit in FPSCR register. 0: No interrupt

		1: Interrupt occurred
13	FOVFIF	FPU overflow interrupt flag status This bit is a copy of the core's OFC bit in FPSCR register. 0: No interrupt 1: Interrupt occurred
12	FUFIF	FPU underflow interrupt flag status This bit is a copy of the core's UFC bit in FPSCR register. 0: No interrupt 1: Interrupt occurred
11	FDZIF	FPU divide-by-zero interrupt flag status This bit is a copy of the core's DZC bit in FPSCR register. 0: No interrupt 1: Interrupt occurred
10	FIOPIF	FPU invalid operation interrupt flag status This bit is a copy of the core's IOC bit in FPSCR register. 0: No interrupt 1: Interrupt occurred
9:7	Reserved	Must be kept at reset value.
6	WASOVF	Write abort on slave overrun flag It is reported only if WAIE is 1. 0: No write abort overrun 1: Write abort on slave overrun occurred
5	WAS	Write abort on AHB slave 0: No write abort occurred 1: Write abort occurred
4:0	Reserved	Must be kept at reset value.

1.10.13. SRAM configuration register (SYSCFG_SRAMCFG)

Address offset: 0x68

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	SRAM_WAITSTATE	TCM_WAITSTATE
	rw	rw

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	SRAM_WAITSTATE	SRAM wait state configuration This bit is set and cleared by software. This bit is used to insert wait-state in SRAM0. **Note:** The chip can only operate at the highest frequency after configuring this bit to 1 and reading back that the bit has already been configured to 1. 0: No wait-state 1: Insert wait-state
0	TCM_WAITSTATE	TCM wait state configuration This bit is set and cleared by software. This bit is used to insert wait-state in ITCM/ D0TCM/ D1TCM. 0: No wait-state 1: Insert wait-state

1.10.14. Recovery mode configuration register (SYSCFG_RMCFG)

Address offset: 0x6C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														RM_SYSRST_DIS	RM_SUBSRST_DIS
														rw	rw

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	RM_SYSRST_DIS	Disable recovery mode on system reset. This bit is set and cleared by software, and is reset by power on reset. Application can set this bit to disable recovery mode on system reset. 0: Allows recovery mode sequence on system reset 1: Disables recovery mode on system reset
0	RM_SUBSRST_DIS	Disable recovery mode on subsystem reset. This bit is set and cleared by software, and is reset by power on reset.

Application can set this bit to disable recovery mode on subsystem reset.

0: Allows recovery mode sequence on subsystem reset

1: Disables recovery mode on subsystem reset

1.10.15. TIMERx configuration register 0 (SYSCFG_TIMERxCFG0) (x = 0, 7, 19, 20, 60, 61, 62, 63)

Address offset: 0x100 for TIMER0
Address offset: 0x124 for TIMER7
Address offset: 0x130 for TIMER19
Address offset: 0x13C for TIMER20
Address offset: 0x148 for TIMER60
Address offset: 0x154 for TIMER61
Address offset: 0x160 for TIMER62
Address offset: 0x16C for TIMER63
Reset value: 0x0000 0000

TSCFG0[4:0], TSCFG1[4:0]..TSCFG11[4:0] are mutually exclusive and cannot be configured at the same time.

The internal connection of TIMER please refer *Table 1-17. Internal connection of TIMER*.

Table 1-17. Internal connection of TIMER

TIMERx / ITIx	TIMER0	TIMER7	TIMER19	TIMER20	TIMER60	TIMER61	TIMER62	TIMER63	TIMER1	TIMER2
ITI0	TRGSEL	TRGSEL	TRGSEL	TRGSEL	TRGSEL	TRGSEL	TRGSEL	TRGSEL	TRGSEL	TRGSEL
ITI1	TIMER7	TIMER0	TIMER0	TIMER0	TIMER0	TIMER0	TIMER0	TIMER0	TIMER0	TIMER0
ITI2	TIMER19	TIMER19	TIMER7	TIMER7	TIMER7	TIMER7	TIMER7	TIMER7	TIMER7	TIMER7
ITI3	TIMER20	TIMER20	TIMER20	TIMER19	TIMER19	TIMER19	TIMER19	TIMER19	TIMER19	TIMER19
ITI4	TIMER60	TIMER60	TIMER60	TIMER60	TIMER20	TIMER20	TIMER20	TIMER20	TIMER20	TIMER20
ITI5	TIMER61	TIMER61	TIMER61	TIMER61	TIMER61	TIMER60	TIMER60	TIMER60	TIMER60	TIMER60
ITI6	TIMER62	TIMER62	TIMER62	TIMER62	TIMER62	TIMER62	TIMER61	TIMER61	TIMER61	TIMER61
ITI7	TIMER63	TIMER63	TIMER63	TIMER63	TIMER63	TIMER63	TIMER63	TIMER62	TIMER62	TIMER62
ITI8	TIMER1	TIMER1	TIMER1	TIMER1	TIMER1	TIMER1	TIMER1	TIMER1	TIMER63	TIMER63
ITI9	TIMER2	TIMER2	TIMER2	TIMER2	TIMER2	TIMER2	TIMER2	TIMER2	TIMER2	TIMER1

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	TSCFG5[4:0]					TSCFG4[4:0]					TSCFG3[4:0]				
	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	TSCFG2[4:0]					TSCFG1[4:0]					TSCFG0[4:0]				
	rw					rw					rw				

Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30:26	TSCFG5[4:0]	Event mode configuration A rising edge of the trigger input enables the counter. 00000: Event mode disable 00001: Internal trigger input 0 (ITI0) 00010: Internal trigger input 1 (ITI1) 00011: Internal trigger input 2 (ITI2) 00100: Internal trigger input 3 (ITI3) 00101: Internal trigger input 4 (ITI4) 00110: Internal trigger input 5 (ITI5) 00111: Internal trigger input 6 (ITI6) 01000: Internal trigger input 7 (ITI7) 01001: Internal trigger input 8 (ITI8) 01010: Internal trigger input 9 (ITI9) 10101: CI0 edge flag (CI0F_ED) 10110: The filtered output of channel 0 input (CI0FE0) 10111: The filtered output of channel 1 input (CI1FE1) 11000: The filtered output of external trigger input (ETIFP) 11001：The filtered output of channel 2 input (CI2FE2) 11010：The filtered output of channel 3 input (CI3FE3) 11011：The filtered output of multi mode channel 0 input (MCI0FEM0) 11100：The filtered output of multi mode channel 1 input (MCI1FEM1) 11101：The filtered output of multi mode channel 2 input (MCI2FEM2) 11110：The filtered output of multi mode channel 3 input (MCI3FEM3) Others：Reserved
25:21	TSCFG4[4:0]	Pause mode configuration The trigger input enables the counter clock when it is high and disables the counter when it is low when these bits are not 0. 00000: Pause mode disable 00001: Internal trigger input 0 (ITI0) 00010: Internal trigger input 1 (ITI1) 00011: Internal trigger input 2 (ITI2) 00100: Internal trigger input 3 (ITI3) 00101: Internal trigger input 4 (ITI4)

00110: Internal trigger input 5 (ITI5)
00111: Internal trigger input 6 (ITI6)
01000: Internal trigger input 7 (ITI7)
01001: Internal trigger input 8 (ITI8)
01010: Internal trigger input 9 (ITI9)
10101: CI0 edge flag (CI0F_ED)
10110: The filtered output of channel 0 input (CI0FE0)
10111: The filtered output of channel 1 input (CI1FE1)
11000: The filtered output of external trigger input (ETIFP)
11001: The filtered output of channel 2 input (CI2FE2)
11010: The filtered output of channel 3 input (CI3FE3)
11011: The filtered output of multi mode channel 0 input (MCI0FEM0)
11100: The filtered output of multi mode channel 1 input (MCI1FEM1)
11101: The filtered output of multi mode channel 2 input (MCI2FEM2)
11110: The filtered output of multi mode channel 3 input (MCI3FEM3)
Others: Reserved

20:16	TSCFG3[4:0]	Restart mode configuration The counter is reinitialized and the shadow registers are updated on the rising edge of the selected trigger input when these bits are not 0. 00000: Restart mode disable 00001: Internal trigger input 0 (ITI0) 00010: Internal trigger input 1 (ITI1) 00011: Internal trigger input 2 (ITI2) 00100: Internal trigger input 3 (ITI3) 00101: Internal trigger input 4 (ITI4) 00110: Internal trigger input 5 (ITI5) 00111: Internal trigger input 6 (ITI6) 01000: Internal trigger input 7 (ITI7) 01001: Internal trigger input 8 (ITI8) 01010: Internal trigger input 9 (ITI9) 10101: CI0 edge flag (CI0F_ED) 10110: The filtered output of channel 0 input (CI0FE0) 10111: The filtered output of channel 1 input (CI1FE1) 11000: The filtered output of external trigger input (ETIFP) 11001: The filtered output of channel 2 input (CI2FE2) 11010: The filtered output of channel 3 input (CI3FE3) 11011: The filtered output of multi mode channel 0 input (MCI0FEM0) 11100: The filtered output of multi mode channel 1 input (MCI1FEM1) 11101: The filtered output of multi mode channel 2 input (MCI2FEM2) 11110: The filtered output of multi mode channel 3 input (MCI3FEM3) Others: Reserved
15	Reserved	Must be kept at reset value.
14:10	TSCFG2[4:0]	Quadrature decoder mode 2 configuration

		00000: Quadrature decoder mode 2 disable Others: The counter counts on both CI0FE0 and CI1FE1 edges, while the direction depends on each other
9:5	TSCFG1[4:0]	Quadrature decoder mode 1 configuration 00000: Quadrature decoder mode 1 disable Others: The counter counts on CI1FE1 edge, while the direction depends on CI0FE0 level
4:0	TSCFG0[4:0]	Quadrature decoder mode 0 configuration 00000: Quadrature decoder mode 0 disable Others: The counter counts on CI0FE0 edge, while the direction depends on CI1FE1 level.

1.10.16. TIMERx configuration register 1 (SYSCFG_TIMERxCFG1) (x = 0, 7, 19, 20, 60, 61, 62, 63)

Address offset: 0x104 for TIMER0
Address offset: 0x128 for TIMER7
Address offset: 0x134 for TIMER19
Address offset: 0x140 for TIMER20
Address offset: 0x14C for TIMER60
Address offset: 0x158 for TIMER61
Address offset: 0x164 for TIMER62
Address offset: 0x170 for TIMER63
Reset value: 0x0000 0000

TSCFG0[4:0], TSCFG1[4:0]..TSCFG11[4:0] are mutually exclusive and cannot be configured at the same time.

The internal connection of TIMER please refer *Table 1-17. Internal connection of TIMER*.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	TSCFG11[4:0]					TSCFG10[4:0]					TSCFG9[4:0]				
													rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	TSCFG8[4:0]					TSCFG7[4:0]					TSCFG6[4:0]				
	rw					rw					rw				

Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30:26	TSCFG11[4:0]	Decoder mode 3 configuration 00000: Decoder mode 3 disable Others: the counter will count on rising or falling edge of CI0FE0 and CI1FE1

		signals. When CHxP=0, the counter will counter on the high level or the falling edge or the CIxFEx signal; When CHxP=1, the counter will counter on the low level or the rising edge or the CIxFEx signal.
25:21	TSCFG10[4:0]	Decoder mode 2 configuration 00000: Decoder mode 2 disable Others: The counter will count on both rising and falling edges of CI0FE0 and CI1FE1 signals. And the counting direction determined by the CH0P and CH1P bits.
20:16	TSCFG9[4:0]	Decoder mode 1 configuration 00000: Decoder mode 1 disable Others: The CI0FE0 is used as the count direction signal and the CI1FE1 signal is used as the count pulse. And the counter counts on the edge of CI1FE1, which is set by the CH1P.
15	Reserved	Must be kept at reset value.
14:10	TSCFG8[4:0]	Decoder mode 0 configuration 00000: Decoder mode 0 disable Others: The CI0FE0 is used as the count direction signal and the CI1FE1 signal is used as the count pulse. The counter will count on both rising and falling edges of the CI1FE1 signal.
9:5	TSCFG7[4:0]	Restart + event mode configuration The counter is reinitialized and started, the shadow registers are updated on the rising edge of the selected trigger input when these bits are not 0. 00000: Restart + event mode disable 00001: Internal trigger input 0 (ITI0) 00010: Internal trigger input 1 (ITI1) 00011: Internal trigger input 2 (ITI2) 00100: Internal trigger input 3 (ITI3) 00101: Internal trigger input 4 (ITI4) 00110: Internal trigger input 5 (ITI5) 00111: Internal trigger input 6 (ITI6) 01000: Internal trigger input 7 (ITI7) 01001: Internal trigger input 8 (ITI8) 01010: Internal trigger input 9 (ITI9) 10101: CI0 edge flag (CI0F_ED) 10110: The filtered output of channel 0 input (CI0FE0) 10111: The filtered output of channel 1 input (CI1FE1) 11000: The filtered output of external trigger input (ETIFP) 11001: The filtered output of channel 2 input (CI2FE2) 11010: The filtered output of channel 3 input (CI3FE3) 11011: The filtered output of multi mode channel 0 input (MCI0FEM0) 11100: The filtered output of multi mode channel 1 input (MCI1FEM1) 11101: The filtered output of multi mode channel 2 input (MCI2FEM2) 11110: The filtered output of multi mode channel 3 input (MCI3FEM3)

Others：Reserved

4:0	TSCFG6[4:0]	External clock mode 0 configuration The counter counts on the rising edges of the selected trigger when these bits are not 0. 00000: External clock mode 0 mode disable 00001: Internal trigger input 0 (ITI0) 00010: Internal trigger input 1 (ITI1) 00011: Internal trigger input 2 (ITI2) 00100: Internal trigger input 3 (ITI3) 00101: Internal trigger input 4 (ITI4) 00110: Internal trigger input 5 (ITI5) 00111: Internal trigger input 6 (ITI6) 01000: Internal trigger input 7 (ITI7) 01001: Internal trigger input 8 (ITI8) 01010: Internal trigger input 9 (ITI9) 10101: CI0 edge flag (CI0F_ED) 10110: The filtered output of channel 0 input (CI0FE0) 10111: The filtered output of channel 1 input (CI1FE1) 11000: The filtered output of external trigger input (ETIFP) 11001：The filtered output of channel 2 input (CI2FE2) 11010：The filtered output of channel 3 input (CI3FE3) 11011：The filtered output of multi mode channel 0 input (MCI0FEM0) 11100：The filtered output of multi mode channel 1 input (MCI1FEM1) 11101：The filtered output of multi mode channel 2 input (MCI2FEM2) 11110：The filtered output of multi mode channel 3 input (MCI3FEM3) Others：Reserved

1.10.17. TIMERx configuration register 2 (SYSCFG_TIMERxCFG2) (x = 0, 7, 19, 20, 60, 61, 62, 63)

Address offset: 0x108 for TIMER0
Address offset: 0x12C for TIMER7
Address offset: 0x138 for TIMER19
Address offset: 0x144 for TIMER20
Address offset: 0x150 for TIMER60
Address offset: 0x15C for TIMER61
Address offset: 0x168 for TIMER62
Address offset: 0x174 for TIMER63
Reset value: 0x0000 0000

TSCFG0[4:0], TSCFG1[4:0]..TSCFG9[4:0] are mutually exclusive and cannot be configured at the same time.

The internal connection of TIMER please refer ***Table 1-17. Internal connection of TIMER***.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											TSCFG15[4:0]				
													rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits	Fields	Descriptions
31:21	Reserved	Must be kept at reset value.
20:16	TSCFG15[4:0]	Internal trigger input source configuration 00000: Reserved 00001: Internal trigger input 0 (ITI0) 00010: Internal trigger input 1 (ITI1) 00011: Internal trigger input 2 (ITI2) 00100: Internal trigger input 3 (ITI3) 00101: Internal trigger input 4 (ITI4) 00110: Internal trigger input 5 (ITI5) 00111: Internal trigger input 6 (ITI6) 01000: Internal trigger input 7 (ITI7) 01001: Internal trigger input 8 (ITI8) 01010: Internal trigger input 9 (ITI9) 10101: CI0 edge flag (CI0F_ED) Others：Reserved
15:0	Reserved	Must be kept at reset value.

1.10.18. TIMERx configuration register 0 (SYSCFG_TIMERxCFG0) (x = 1, 2)

Address offset: 0x10C for TIMER1
Address offset: 0x118 for TIMER2
Reset value: 0x0000 0000

TSCFG0[4:0], TSCFG1[4:0]..TSCFG15[4:0] are mutually exclusive and cannot be configured at the same time.

The internal connection of TIMER please refer ***Table 1-17. Internal connection of TIMER***.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	TSCFG5[4:0]					TSCFG4[4:0]					TSCFG3[4:0]				
	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	TSCFG2[4:0]					TSCFG1[4:0]					TSCFG0[4:0]				

rw rw rw

Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30:26	TSCFG5[4:0]	Event mode configuration A rising edge of the trigger input enables the counter. 00000: Event mode disable 00001: Internal trigger input 0 (ITI0) 00010: Internal trigger input 1 (ITI1) 00011: Internal trigger input 2 (ITI2) 00100: Internal trigger input 3 (ITI3) 00101: Internal trigger input 4 (ITI4) 00110: Internal trigger input 5 (ITI5) 00111: Internal trigger input 6 (ITI6) 01000: Internal trigger input 7 (ITI7) 01001: Internal trigger input 8 (ITI8) 01010: Internal trigger input 9 (ITI9) 10101: CI0 edge flag (CI0F_ED) 10110: The filtered output of channel 0 input (CI0FE0) 10111: The filtered output of channel 1 input (CI1FE1) 11000: The filtered output of external trigger input (ETIFP) Others：Reserved
25:21	TSCFG4[4:0]	Pause mode configuration The trigger input enables the counter clock when it is high and disables the counter when it is low when these bits are not 0. 00000: Pause mode disable 00001: Internal trigger input 0 (ITI0) 00010: Internal trigger input 1 (ITI1) 00011: Internal trigger input 2 (ITI2) 00100: Internal trigger input 3 (ITI3) 00101: Internal trigger input 4 (ITI4) 00110: Internal trigger input 5 (ITI5) 00111: Internal trigger input 6 (ITI6) 01000: Internal trigger input 7 (ITI7) 01001: Internal trigger input 8 (ITI8) 01010: Internal trigger input 9 (ITI9) 10101: CI0 edge flag (CI0F_ED) 10110: The filtered output of channel 0 input (CI0FE0) 10111: The filtered output of channel 1 input (CI1FE1) 11000: The filtered output of external trigger input (ETIFP) Others：Reserved
20:16	TSCFG3[4:0]	Restart mode configuration The counter is reinitialized and the shadow registers are updated on the rising edge

		of the selected trigger input when these bits are not 0. 00000: Restart mode disable 00001: Internal trigger input 0 (ITI0) 00010: Internal trigger input 1 (ITI1) 00011: Internal trigger input 2 (ITI2) 00100: Internal trigger input 3 (ITI3) 00101: Internal trigger input 4 (ITI4) 00110: Internal trigger input 5 (ITI5) 00111: Internal trigger input 6 (ITI6) 01000: Internal trigger input 7 (ITI7) 01001: Internal trigger input 8 (ITI8) 01010: Internal trigger input 9 (ITI9) 10101: CI0 edge flag (CI0F_ED) 10110: The filtered output of channel 0 input (CI0FE0) 10111: The filtered output of channel 1 input (CI1FE1) 11000: The filtered output of external trigger input (ETIFP) Others：Reserved
15	Reserved	Must be kept at reset value.
14:10	TSCFG2[4:0]	Quadrature decoder mode 2 configuration 00000: Quadrature decoder mode 2 disable Others: The counter counts on both CI0FE0 and CI1FE1 edges, while the direction depends on each other
9:5	TSCFG1[4:0]	Quadrature decoder mode 1 configuration 00000: Quadrature decoder mode 1 disable Others: The counter counts on CI1FE1 edge, while the direction depends on CI0FE0 level
4:0	TSCFG0[4:0]	Quadrature decoder mode 0 configuration 00000: Quadrature decoder mode 0 disable Others: The counter counts on CI0FE0 edge, while the direction depends on CI1FE1 level.

1.10.19. TIMERx configuration register 1 (SYSCFG_TIMERxCFG1) (x = 1, 2)

Address offset: 0x110 for TIMER1
Address offset: 0x11C for TIMER2
Reset value: 0x0000 0000

TSCFG0[4:0], TSCFG1[4:0]..TSCFG11[4:0] are mutually exclusive and cannot be configured at the same time.

The internal connection of TIMER please refer ***Table 1-17. Internal connection of TIMER***.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved	TSCFG11[4:0]	TSCFG10[4:0]	TSCFG9[4:0]
			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	TSCFG8[4:0]	TSCFG7[4:0]	TSCFG6[4:0]
	rw	rw	rw

Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30:26	TSCFG11[4:0]	Decoder mode 3 configuration 00000: Decoder mode 3 disable Others: the counter will count on rising or falling edge of CI0FE0 and CI1FE1 signals. When CHxP=0, the counter will counter on the high level or the falling edge or the CIxFEx signal; When CHxP=1, the counter will counter on the low level or the rising edge or the CIxFEx signal.
25:21	TSCFG10[4:0]	Decoder mode 2 configuration 00000: Decoder mode 2 disable Others: The counter will count on both rising and falling edges of CI0FE0 and CI1FE1 signals. And the counting direction determined by the CH0P and CH1P bits.
20:16	TSCFG9[4:0]	Decoder mode 1 configuration 00000: Decoder mode 1 disable Others: The CI0FE0 is used as the count direction signal and the CI1FE1 signal is used as the count pulse. And the counter counts on the edge of CI1FE1, which is set by the CH1P.
15	Reserved	Must be kept at reset value.
14:10	TSCFG8[4:0]	Decoder mode 0 configuration 00000: Decoder mode 0 disable Others: The CI0FE0 is used as the count direction signal and the CI1FE1 signal is used as the count pulse. The counter will count on both rising and falling edges of the CI1FE1 signal.
9:5	TSCFG7[4:0]	Restart + event mode configuration The counter is reinitialized and started, the shadow registers are updated on the rising edge of the selected trigger input when these bits are not 0. 00000: Restart + event mode disable 00001: Internal trigger input 0 (ITI0) 00010: Internal trigger input 1 (ITI1) 00011: Internal trigger input 2 (ITI2) 00100: Internal trigger input 3 (ITI3) 00101: Internal trigger input 4 (ITI4) 00110: Internal trigger input 5 (ITI5) 00111: Internal trigger input 6 (ITI6) 01000: Internal trigger input 7 (ITI7)

01001: Internal trigger input 8 (ITI8)
01010: Internal trigger input 9 (ITI9)
10101: CI0 edge flag (CI0F_ED)
10110: The filtered output of channel 0 input (CI0FE0)
10111: The filtered output of channel 1 input (CI1FE1)
11000: The filtered output of external trigger input (ETIFP)
Others：Reserved Others：Reserved

4:0	TSCFG6[4:0]	External clock mode 0 configuration The counter counts on the rising edges of the selected trigger when these bits are not 0. 00000: Restart mode disable 00001: Internal trigger input 0 (ITI0) 00010: Internal trigger input 1 (ITI1) 00011: Internal trigger input 2 (ITI2) 00100: Internal trigger input 3 (ITI3) 00101: Internal trigger input 4 (ITI4) 00110: Internal trigger input 5 (ITI5) 00111: Internal trigger input 6 (ITI6) 01000: Internal trigger input 7 (ITI7) 01001: Internal trigger input 8 (ITI8) 01010: Internal trigger input 9 (ITI9) 10101: CI0 edge flag (CI0F_ED) 10110: The filtered output of channel 0 input (CI0FE0) 10111: The filtered output of channel 1 input (CI1FE1) 11000: The filtered output of external trigger input (ETIFP) Others：Reserved

1.10.20. TIMERx configuration register 2 (SYSCFG_TIMERxCFG2) (x = 1, 2)

Address offset: 0x114 for TIMER1
Address offset: 0x120 for TIMER2
Reset value: 0x0000 0000

TSCFG0[4:0], TSCFG1[4:0]..TSCFG9[4:0] are mutually exclusive and cannot be configured at the same time.

The internal connection of TIMER please refer ***Table 1-17. Internal connection of TIMER***.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											TSCFG15[4:0]				
													rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits	Fields	Descriptions
31:21	Reserved	Must be kept at reset value.
20:16	TSCFG15[4:0]	Internal trigger input source configuration 00000: Reserved 00001: Internal trigger input 0 (ITI0) 00010: Internal trigger input 1 (ITI1) 00011: Internal trigger input 2 (ITI2) 00100: Internal trigger input 3 (ITI3) 00101: Internal trigger input 4 (ITI4) 00110: Internal trigger input 5 (ITI5) 00111: Internal trigger input 6 (ITI6) 01000: Internal trigger input 7 (ITI7) 01001: Internal trigger input 8 (ITI8) 01010: Internal trigger input 9 (ITI9) 10101: CI0 edge flag (CI0F_ED) Others：Reserved
15:0	Reserved	Must be kept at reset value.

Offset address range	Purpose
220h - 238h	Defines the core x information. Only the CM7 cores on the chip can access this region; reads by non-core bus masters are treated as read as zero (RAZ) accesses. Write attempts are not permitted and terminate with a system bus error.
240h - 258h	Defines the information for core 0 (CP0). Any bus master can access this region. Write attempts are not permitted. Write attempts are not permitted and terminate with a system bus error.
260h - 278h	Defines the information for core 1 (CP1). Any bus master can access this region. Write attempts are not permitted and terminate with a system bus error.

1.10.21. Core processor 0 ID register (SYSCFG_CP0ID)

Address offset: 0x240

Reset value: 0x434D 3730

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CPID[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CPID[15:0]															
r															

Bits	Fields	Descriptions
31:0	CPID[31:0]	Core processor ID Defines the processor ID for Cortex-M7_0. For Cortex®-M7 core 0, the process ID = 43_4D_37_30h.

1.10.22. Core processor 0 number register (SYSCFG_CP0NUM)

Address offset: 0x244

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														CPNUM[1:0]	
														r	

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1:0	CPNUM[1:0]	CPU core number Provides the logical processor number for CP0. For Cortex®-M7 core 0, processor number = 0.

1.10.23. Core processor 0 configuration information register 0 (SYSCFG_CP0CFGINF0)

Address offset: 0x24C

Reset value: 0x0502 0504

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IC_SZ[7:0]								IC_WAY[7:0]							
r								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DC_SZ[7:0]								DC_WAY[7:0]							
r								r							

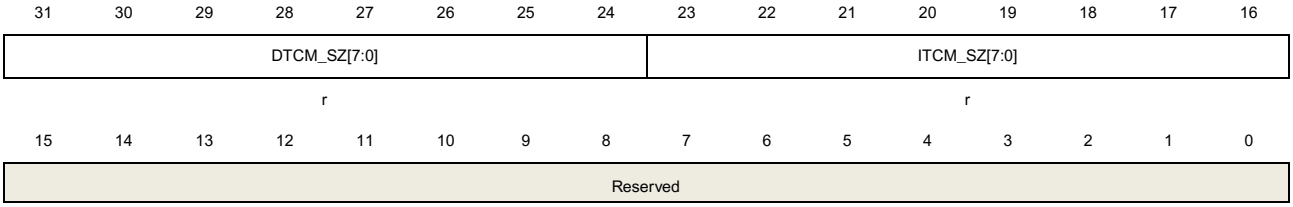
Bits	Fields	Descriptions
31:24	IC_SZ[7:0]	L1 instruction cache size **Note:** IC_SZ = 0 means the memory is not present. IC_SZ = 5h means 8KB
23:16	IC_WAY[7:0]	L1 instruction cache ways IC_WAY = 2h means 2-way set-associative
15:8	DC_SZ[7:0]	L1 data cache size **Note:** DC_SZ = 0 means the memory is not present. DC_SZ = 5h means 8KB
7:0	DC_WAY[7:0]	L1 data cache ways DC_WAY = 4h means 4-way set-associative

1.10.24. Core processor 0 configuration information register 1 (SYSCFG_CP0CFGINF1)

Address offset: 0x254

Reset value: 0xXXXX 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:24	DTCM_SZ[7:0]	DTCM memory size The definition of memory capacity (byte) is defined by the formula $2^{(8+DTCM_SZ)}$. **Note:** DTCM_SZ = 0 means the memory is not present.
23:16	ITCM_SZ[7:0]	ITCM memory size The definition of memory capacity (byte) is defined by the formula $2^{(8+ITCM_SZ)}$. **Note:** ITCM_SZ = 0 means the memory is not present.
15:0	Reserved	Must be kept at reset value.

1.10.25. Core processor 0 configuration information register 2 (SYSCFG_CP0CFGINF2)

Address offset: 0x258

Reset value: 0x0000 000B

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											CRYP	CMPU	MMU	SIMD	FPU
											r	r	r	r	r

Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
4	CRYP	Cryptography support 0: Not supported 1: Supported
3	CMPU	Core memory protection unit support 0: Not included 1: Included
2	MMU	Memory management unit support 0: Not supported 1: Supported
1	SIMD	SIMD/NEON instruction support 0: Not included 1: Included
0	FPU	Floating point unit support 0: Not provided 1: Provided

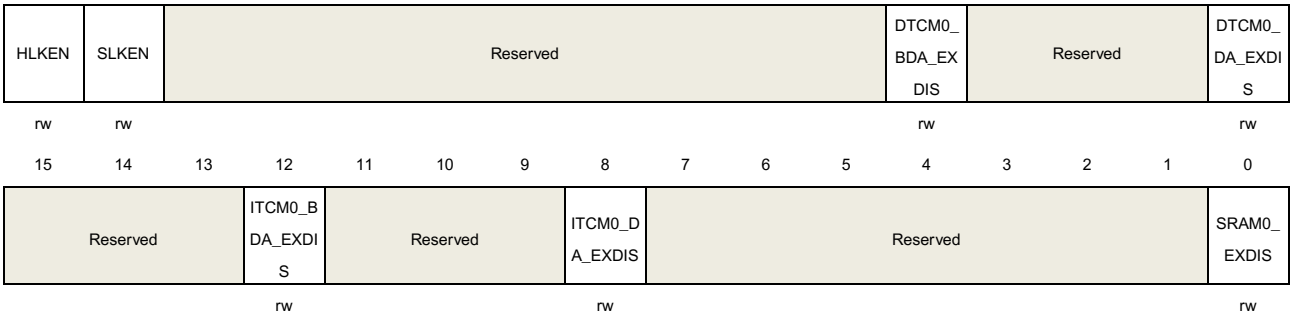
1.10.26. Memory access execution control register(SYSCFG_MEMAEXCTL)

Address offset: 0x350

Reset value: 0x4000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16



Bits	Fields	Descriptions
31	HLKEN	Hard lock enable Set this bit to limit the ability the write to this register until the next hardware reset. If this bit is set to 1, writing 0 to it by the software is invalid. 0: Disabled 1: Enabled
30	SLKEN	Soft lock enable This bit indicates soft lock the register. If it is needed to modify those configuration bits in the low field by the software, it needs to clear SLKEN to 0 first. 0: Disabled 1: Enabled
29:21	Reserved	Must be kept at reset value
20	DTCM0_BDA_EXDIS	DTCM core 0 backdoor access execution control Indicates whether control is allowed for DTCM backdoor access. 0: Allowed 1: Disabled
19:17	Reserved	Must be kept at reset value
16	DTCM_DA_EXDIS	DTCM core 0 direct access execution control Indicates whether execution control is allowed for DTCM direct access. 0: Allowed 1: Disabled
15:13	Reserved	Must be kept at reset value
12	ITCM_BDA_EXDIS	ITCM core 0 backdoor access execution control Indicates whether control is allowed for ITCM backdoor access. 0: Allowed 1: Disabled
11:9	Reserved	Must be kept at reset value
8	ITCM_DA_EXDIS	ITCM0 direct access execution control Indicates whether execution control is allowed for ITCM direct access. 0: Allowed

		1: Disabled
7:1	Reserved	Must be kept at reset value
0	SRAM0_EXDIS	SRAM0 controller access execution control Indicates whether execution control is allowed for the SRAM controller. 0: Allowed 1: Disabled

1.10.27. Bus matrix 1 interconnect error detection enable register (SYSCFG_BM1EDCEN)

Address offset: 0x360

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						ADDC_TCM_EN	WDC_TCM_EN	ADDC_FMC_2_EN	WDC_FMC_2	Reserved		ADDC_FMC_0_EN	WDC_FMC_0	ADDC_QSPI_EN	WDC_QSPI_EN
						rw	rw	rw	rw			rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ADDC_SRAM0_EN	WDC_SRAM0_EN	Reserved					RDC_CM7_0_EN	Reserved	RDC_HSM_BM1_EN	RDC_DMA1_P_EN	RDC_DMA1_M_EN	RDC_DMA0_P_EN	RDC_DMA0_M_EN
		rw	rw						rw		rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value
25	ADDC_TCM_EN	Address check TCM SRAM enable Enables or disables the address check for TCM SRAM path. 0: Disabled 1: Enabled
24	WDC_TCM_EN	Write data check TCM SRAM enable Enables or disables the write data check for TCM SRAM path. 0: Disabled 1: Enabled
23	ADDC_FMC_2_EN	Enable address check FMC_2 Enables or disables the address check for FMC_2 path. 0: Disabled 1: Enabled
22	WDC_FMC_2_EN	Enable write data check FMC_2 Enables or disables the write data check for FMC_2 path.

		0: Disabled 1: Enabled
21:20	Reserved	Must be kept at reset value
19	ADDC_FMC_0_EN	Enable address check FMC_0 Enables or disables the address check for FMC_0 path. 0: Disabled 1: Enabled
18	WDC_FMC_0_EN	Enable write data check FMC_0 Enables or disables the write data check for FMC_0 path. 0: Disabled 1: Enabled
17	ADDC_QSPI_EN	Enable address check Quad-SPI Enables or disables the address check for Quad-SPI path. 0: Disabled 1: Enabled
16	WDC_QSPI_EN	Enable write data check Quad-SPI Enables or disables the write data check for Quad-SPI path. 0: Disabled 1: Enabled
15:14	Reserved	Must be kept at reset value
13	ADDC_SRAM0_EN	Enable address check SRAM0 Enables or disables the address check for SRAM path. 0: Disabled 1: Enabled
12	WDC_SRAM0_EN	Enable write data check SRAM0 Enables or disables the write data check for SRAM path. 0: Disabled 1: Enabled
11:7	Reserved	Must be kept at reset value
6	RDC_CM7_0_EN	Enable read data check Cortex-M7_0_AXIM Enables or disables the read data check for Cortex-M7_0_AXIM path. 0: Disabled 1: Enabled
5	Reserved	Must be kept at reset value
4	RDC_HSM_BM1_EN	Enable read data check HSM BM1 Enables or disables the read data check for HSM path. 0: Disabled 1: Enabled

3	RDC_DMA1_P_EN	Enable read data check DMA1 peripheral accesses Enables or disables the read data check for DMA1 peripheral accesses path. 0: Disabled 1: Enabled
2	RDC_DMA1_M_EN	Enable read data check DMA1 memory accesses Enables or disables the read data check for DMA1 memory accesses path. 0: Disabled 1: Enabled
1	RDC_DMA0_P_EN	Enable read data check DMA0 peripheral accesses Enables or disables the read data check for DMA0 peripheral accesses path. 0: Disabled 1: Enabled
0	RDC_DMA0_M_EN	Enable read data check DMA0 memory accesses Enables or disables the read data check for DMA0 memory accesses path. 0: Disabled 1: Enabled

1.10.28. Bus matrix 2 interconnect error detection enable register (SYSCFG_BM2EDCEN)

Address offset: 0x364
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				ADDC_AHB2_EN	WDC_AHB2_EN	ADDC_AHB1_EN	WDC_AHB1_EN	ADDC_APB2_EN	WDC_APB2_EN	ADDC_APB1_EN	WDC_APB1_EN	Reserved			
				rw	rw	rw	rw	rw	rw	rw	rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RDC_AHBP0_EN	Reserved	RDC_HSM_BM2_EN	Reserved	RDC_DMA1_P_EN	RDC_DMA1_M_EN	RDC_DMA0_P_EN	RDC_DMA0_M_EN
								rw		rw		rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value
27	ADDC_AHB2_EN	Enable address check AHB2 Enables or disables the AHB2 address check. 0: Disabled 1: Enabled
26	WDC_AHB2_EN	Enable write data check AHB2

		Enables or disables the AHB2 write data check. 0: Disabled 1: Enabled
25	ADDC_AHB1_EN	Enable address check AHB1 Enables or disables the AHB1 address check. 0: Disabled 1: Enabled
24	WDC_AHB1_EN	Enable write data check AHB1 Enables or disables the AHB1 write data check. 0: Disabled 1: Enabled
23	ADDC_APB2_EN	Enable address check APB2 Enables or disables the APB2 address check. 0: Disabled 1: Enabled
22	WDC_APB2_EN	Enable write data check APB2 Enables or disables the APB2 write data check. 0: Disabled 1: Enabled
21	ADDC_APB1_EN	Enable address check APB1 Enables or disables the APB1 address check. 0: Disabled 1: Enabled
20	WDC_APB1_EN	Enable write data check APB1 Enables or disables the APB1 write data check. 0: Disabled 1: Enabled
19:8	Reserved	Must be kept at reset value
7	RDC_AHBP_EN	Enable read data check Cortex-M7_0_AHBP Enables or disables the read data check for Cortex-M7_0_AHBP path. 0: Disabled 1: Enabled
6	Reserved	Must be kept at reset value
5	RDC_HSM_BM2_EN	Enable read data check HSM BM2 Enables or disables the read data check for HSM path. 0: Disabled 1: Enabled
4	Reserved	Must be kept at reset value

3	RDC_DMA1_P_EN	Enable read data check DMA1 peripheral accesses Enables or disables the read data check for DMA1 peripheral accesses path. 0: Disabled 1: Enabled
2	RDC_DMA1_M_EN	Enable read data check DMA1 memory accesses Enables or disables the read data check for DMA1 memory accesses path. 0: Disabled 1: Enabled
1	RDC_DMA0_P_EN	Enable read data check DMA0 peripheral accesses Enables or disables the read data check for DMA0 peripheral accesses path. 0: Disabled 1: Enabled
0	RDC_DMA0_M_EN	Enable read data check DMA0 memory accesses Enables or disables the read data check for DMA0 memory accesses path. 0: Disabled 1: Enabled

1.10.29. Bus check configuration register (SYSCFG_BUSCHKCFG)

Address offset: 0x368

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	AHB2_CHKEN	AHB1_CHKEN	APB2_CHKEN	APB1_CHKEN	Reserved	TCM_WGT_CHKEN	Reserved	TCM_H2H0_WGT_CHKEN	QSPI_H2H0_WGT_CHKEN	DMA1P_H2H0_WGT_CHKEN	DMA1M_H2H0_WGT_CHKEN	DMA0P_H2H0_WGT_CHKEN	DMA0M_H2H0_WGT_CHKEN	HSM_H2H1_WGT_CHKEN	TCM_SPLT_WGT_CHKEN
	rw	rw	rw	rw		rw		rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSM_SPLT_WGT_CHKEN	DMA1P_SPLT_WGT_CHKEN	DMA1M_SPLT_WGT_CHKEN	DMA0P_SPLT_WGT_CHKEN	DMA0M_SPLT_WGT_CHKEN	Reserved	AXIM_CM7_0_CHKEN	Reserved	AHBP_CM7_0_CHKEN	DMA1P_CHKEN	DMA1M_CHKEN	DMA0P_CHKEN	DMA0M_CHKEN	HSM_CHKEN	Reserved	
rw	rw	rw	rw	rw		rw		rw	rw	rw	rw	rw	rw		

Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value
30	AHB2_CHKEN	Enable or disable AHB2 redundant check. 0: Disabled 1: Enabled

29	AHB1_CHKEN	Enable or disable AHB1 redundant check. 0: Disabled 1: Enabled
28	APB2_CHKEN	Enable or disable APB2 redundant check. 0: Disabled 1: Enabled
27	APB1_CHKEN	Enable or disable APB1 redundant check. 0: Disabled 1: Enabled
26	Reserved	Must be kept at reset value
25	TCM_WGT_CHKEN	Enable or disable TCM 1:1 widget check. 0: Disabled 1: Enabled
24	Reserved	Must be kept at reset value
23	TCM_H2H0_WGT_CHKEN	Enable or disable H2H0 widget of TCM redundant check. 0: Disabled 1: Enabled
22	QSPI_H2H0_WGT_CHKEN	Enable or disable H2H0 widget of QSPI redundant check. 0: Disabled 1: Enabled
21	DMA1P_H2H0_WGT_CHKEN	Enable or disable H2H0 widget of DMA1 peripheral redundant check. 0: Disabled 1: Enabled
20	DMA1M_H2H0_WGT_CHKEN	Enable or disable H2H0 widget of DMA1 memory redundant check. 0: Disabled 1: Enabled
19	DMA0P_H2H0_WGT_CHKEN	Enable or disable H2H0 widget of DMA0 peripheral redundant check. 0: Disabled 1: Enabled
18	DMA0M_H2H0_WGT_CHKEN	Enable or disable H2H0 widget of DMA0 memory redundant check. 0: Disabled 1: Enabled
17	HSM_H2H1_WGT_CHKEN	Enable or disable H2H1 widget of HSM redundant check. 0: Disabled 1: Enabled
16	Reserved	Must be kept at reset value
15	HSM_SPLT_WGT_CHKEN	Enable or disable Split1 widget of HSM redundant check. 0: Disabled

		1: Enabled
14	DMA1P_SPLT_WGT_CHKEN	Enable or disable Split0 widget of DMA1 peripheral redundant check. 0: Disabled 1: Enabled
13	DMA1M_SPLT_WGT_CHKEN	Enable or disable Split0 widget of DMA1 memory redundant check. 0: Disabled 1: Enabled
12	DMA0P_SPLT_WGT_CHKEN	Enable or disable Split0 widget of DMA0 peripheral redundant check. 0: Disabled 1: Enabled
11	DMA0M_SPLT_WGT_CHKEN	Enable or disable Split0 widget of DMA0 memory redundant check. 0: Disabled 1: Enabled
10	Reserved	Must be kept at reset value
9	AXIM_CM7_0_CHKEN	Enable or disable CM7_0 AXIM port timeout and slaver ID check. 0: Disabled 1: Enabled
8	Reserved	Must be kept at reset value
7	AHBP_CM7_0_CHKEN	Enable or disable CM7_0 AHBP port timeout and slaver ID check. 0: Disabled 1: Enabled
6	DMA1P_CHKEN	Enable or disable DMA1 peripheral port timeout and slaver ID check. 0: Disabled 1: Enabled
5	DMA1M_CHKEN	Enable or disable DMA1 memory port timeout and slaver ID check. 0: Disabled 1: Enabled
4	DMA0P_CHKEN	Enable or disable DMA0 peripheral port timeout and slaver ID check. 0: Disabled 1: Enabled
3	DMA0M_CHKEN	Enable or disable DMA0 memory port timeout and slaver ID check. 0: Disabled 1: Enabled
2	HSM_CHKEN	Enable or disable HSM port timeout and slaver ID check. 0: Disabled 1: Enabled
1:0	Reserved	Must be kept at reset value

1.10.30. TMR fault control and status register (SYSCFG_TMRFCS)

Address offset: 0x370

Reset value: 0x0000 0007

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													TMRERR_MEMEXECRIF	TMRERR_CPENIF	TMRERR_FMUNMIENIF
													rc_w1	rc_w1	rc_w1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TMRERR_MEMEXECRIE	TMRERR_CPENIE	TMRERR_FMUNMIENIE
													rw	rw	rw

Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18	TMRERR_MEMEXECRIF	TMR fault interrupt flag of the memory access execution control register. Hardware set and software can clear these fields by writing 1. 0: No TMR error occurred on the memory access execution control register. 1: TMR error occurred on the memory access execution control register.
17	TMRERR_CPENIF	TMR fault interrupt flag of the CPU (CPU0) released enable. Hardware set and software can clear these fields by writing 1. 0: No TMR error occurred on the CPU (CPU0) released enable. 1: TMR error occurred on the CPU (CPU0) released enable.
16	TMRERR_FMUNMIENIF	TMR fault interrupt flag of the FMU NMI interrupt flag enable. Hardware set and software can clear these fields by writing 1. 0: No TMR error occurred on the FMU NMI interrupt flag enable 1: TMR error occurred on the FMU NMI interrupt flag enable.
15:3	Reserved	Must be kept at reset value.
2	TMRERR_MEMEXECRIE	TMR fault interrupt enable of the memory access execution control register. This bit is set and cleared by software. 0: TMR fault interrupt of the memory access execution control register disabled 1: TMR fault interrupt of the memory access execution control register enabled
1	TMRERR_CPENIE	TMR fault interrupt enable of the CPU (CPU0) released. This bit is set and cleared by software. 0: TMR fault interrupt of the CPU (CPU0) released disabled 1: TMR fault interrupt of the CPU (CPU0) released enabled
0	TMRERR_FMUNMIENIE	TMR fault interrupt enable of the FMU NMI interrupt flag enable.

NIE	This bit is set and cleared by software. 0: TMR fault interrupt of the FMU NMI interrupt flag enable disabled 1: TMR fault interrupt of the FMU NMI interrupt flag enable enabled

1.11. BUSIM registers definition

BUSIM base address: 0x4900 9000

Access: privilege read/write

Table 1-18. BUSIM register overview

Register name	Symbol	Address offset	Reset value
BM1 control register x	BUSIM_BM1CTLx	0x00 + 0x10 * x	0x0000 0008
BM1 slave port priority configuration register x	BUSIM_BM1SPCFGx	0x04 + 0x10 * x	0xf654 3200(x=0)/0x7654 3210(x=1-4)
BM2 control register x	BUSIM_BM2CTLx	0x80 + 0x10 * x	0x0000 0008
BM2 slave port priority configuration register x	BUSIM_BM2SPCFGx	0x84 + 0x10 * x	0x0654 3210
TMR fault control and status register	BUSIM_TMRFCS	0x200	0x0000 0003

1.11.1. BM1 control register x (BUSIM_BM1CTLx) (x = 0,1,3,4,)

Address offset: 0x00 + 0x10 * x

Reset value: 0x0000 0008

The corresponding relationships of the IDs are as follows:

BUSIM_CTL0 - FMC2

BUSIM_CTL1 - SRAM

BUSIM_CTL3 - TCMSRAM

BUSIM_CTL4 - QSPI

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31	ROEN	Register read-only enable. Only a system reset clears this bit if it is being set to 1. 0: BUSIM_BM1SPCFGx and BUSIM_BM1CTLx registers can be written 1: BUSIM_BM1SPCFGx and BUSIM_BM1CTLx registers are read-only
30:16	Reserved	Must be kept at reset value.
15	HPM7EN	High priority mode for master port 7 enable bit. (BUSIM_BM1CTLx, only x=1-4 exist) If this bit is enabled, the priority of the master port is granted to the highest.

		0: Disabled 1: Enabled
14	HPM6EN	High priority mode for master port 6 enable bit. (BUSIM_BM1CTLx, only x=1-4 exist) If this bit is enabled, the priority of the master port is granted to the highest. 0: Disabled 1: Enabled
13	Reserved	Must be kept at reset value.
12	HPM4EN	High priority mode for master port 4 enable bit. If this bit is enabled, the priority of the master port is granted to the highest. 0: Disabled 1: Enabled
11	HPM3EN	High priority mode for master port 3 enable bit. If this bit is enabled, the priority of the master port is granted to the highest. 0: Disabled 1: Enabled
10	HPM2EN	High priority mode for master port 2 enable bit. If this bit is enabled, the priority of the master port is granted to the highest. 0: Disabled 1: Enabled
9	Reserved	Must be kept at reset value.
8	HPM0EN	High priority mode for master port 0 enable bit. If this bit is enabled, the priority of the master port is granted to the highest. 0: Disabled 1: Enabled
7:5	Reserved	Must be kept at reset value.
4	ARBMOD	Arbitration mode select 0: Round-robin priority mode selected 1: Fixed priority mode selected
3	DMPSEL	Default master port strategy select 0: Default access the slave port by the master port which is defined by DMP[2:0] bits when no master initiates request 1: Default access the slave port by the master port which is the last master accessed the slave port when no master initiates request
2:0	DMP[2:0]	Default master port 000: Default access the slave port by the master port M0 001: Default access the slave port by the master port M1 010: Default access the slave port by the master port M2 011: Default access the slave port by the master port M3 100: Default access the slave port by the master port M4

101: Default access the slave port by the master port M5
110: Default access the slave port by the master port M6
111: Default access the slave port by the master port M7

1.11.2. BM1 slave port priority configuration register x (BUSIM_BM1SPCFGx) (x = 0,1,3,4)

Address offset: 0x04 + 0x10 * x
Reset value: 0x7604 3200

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	M7PRIO[2:0]			Reserved	M6PRIO[2:0]			Reserved					M4PRIO[2:0]		
	rw				rw								rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	M3PRIO[2:0]			Reserved	M2PRIO[2:0]			Reserved					M0PRIO[2:0]		
	rw				rw								rw		

Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30:28	M7PRIO[2:0]	Priority of master port 7 (BUSIM_BM1SPCFGx, only x=1-4 exist) L1 to L8 represent different priority levels, with L1 being the highest priority and L8 being the lowest priority. 000: Master has L1 priority on associated slave port 001: Master has L2 priority on associated slave port 010: Master has L3 priority on associated slave port 011: Master has L4 priority on associated slave port 100: Master has L5 priority on associated slave port 101: Master has L6 priority on associated slave port 110: Master has L7 priority on associated slave port 111: Master has L8 priority on associated slave port
27	Reserved	Must be kept at reset value.
26:24	M6PRIO[2:0]	Priority of master port 6 (BUSIM_BM1SPCFGx，only x=1-4 exist) L1 to L8 represent different priority levels, with L1 being the highest priority and L8 being the lowest priority. 000: Master has L1 priority on associated slave port 001: Master has L2 priority on associated slave port 010: Master has L3 priority on associated slave port 011: Master has L4 priority on associated slave port 100: Master has L5 priority on associated slave port 101: Master has L6 priority on associated slave port 110: Master has L7 priority on associated slave port

		111: Master has L8 priority on associated slave port
23:19	Reserved	Must be kept at reset value.
18:16	M4PRIO[2:0]	Priority of master port 4 L1 to L8 represent different priority levels, with L1 being the highest priority and L8 being the lowest priority. 000: Master has L1 priority on associated slave port 001: Master has L2 priority on associated slave port 010: Master has L3 priority on associated slave port 011: Master has L4 priority on associated slave port 100: Master has L5 priority on associated slave port 101: Master has L6 priority on associated slave port 110: Master has L7 priority on associated slave port 111: Master has L8 priority on associated slave port
15	Reserved	Must be kept at reset value.
14:12	M3PRIO[2:0]	Priority of master port 3 L1 to L8 represent different priority levels, with L1 being the highest priority and L8 being the lowest priority. 000: Master has L1 priority on associated slave port 001: Master has L2 priority on associated slave port 010: Master has L3 priority on associated slave port 011: Master has L4 priority on associated slave port 100: Master has L5 priority on associated slave port 101: Master has L6 priority on associated slave port 110: Master has L7 priority on associated slave port 111: Master has L8 priority on associated slave port
11	Reserved	Must be kept at reset value.
10:8	M2PRIO[2:0]	Priority of master port 2 L1 to L8 represent different priority levels, with L1 being the highest priority and L8 being the lowest priority. 000: Master has L1 priority on associated slave port 001: Master has L2 priority on associated slave port 010: Master has L3 priority on associated slave port 011: Master has L4 priority on associated slave port 100: Master has L5 priority on associated slave port 101: Master has L6 priority on associated slave port 110: Master has L7 priority on associated slave port 111: Master has L8 priority on associated slave port
7:3	Reserved	Must be kept at reset value.
2:0	M0PRIO[2:0]	Priority of master port 0 L1 to L8 represent different priority levels, with L1 being the highest priority and L8

being the lowest priority.
000: Master has L1 priority on associated slave port
001: Master has L2 priority on associated slave port
010: Master has L3 priority on associated slave port
011: Master has L4 priority on associated slave port
100: Master has L5 priority on associated slave port
101: Master has L6 priority on associated slave port
110: Master has L7 priority on associated slave port
111: Master has L8 priority on associated slave port

1.11.3. BM2 control register x (BUSIM_BM2CTLx) (x = 0...3)

Address offset: 0x80 + 0x10 * x
Reset value: 0x0000 0008

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ROEN	Reserved														
r_w1															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	HPM6EN	HPM5EN	HPM4EN	HPM3EN	HPM2EN	HPM1EN	HPM0EN	Reserved			ARBMOD	DMPSEL	DMP[2:0]		
	rw	rw	rw	rw	rw	rw	rw				rw	rw	rw		

Bits	Fields	Descriptions
31	ROEN	Register read-only enable. Only a system reset clears this bit if it is being set to 1. 0: BUSIM_BM2SPCFGx and BUSIM_BM2CTLx registers can be written 1: BUSIM_BM2SPCFGx and BUSIM_BM2CTLx registers are read-only
30:15	Reserved	Must be kept at reset value.
14	HPM6EN	High priority mode for master port 6 enable bit. If this bit is enabled, the priority of the master port is granted to the highest. 0: Disabled 1: Enabled
13	HPM5EN	High priority mode for master port 5 enable bit. If this bit is enabled, the priority of the master port is granted to the highest. 0: Disabled 1: Enabled
12	HPM4EN	High priority mode for master port 4 enable bit. If this bit is enabled, the priority of the master port is granted to the highest. 0: Disabled 1: Enabled
11	HPM3EN	High priority mode for master port 3 enable bit.

		If this bit is enabled, the priority of the master port is granted to the highest. 0: Disabled 1: Enabled
10	HPM2EN	High priority mode for master port 2 enable bit. If this bit is enabled, the priority of the master port is granted to the highest. 0: Disabled 1: Enabled
9	HPM1EN	High priority mode for master port 1 enable bit. If this bit is enabled, the priority of the master port is granted to the highest. 0: Disabled 1: Enabled
8	HPM0EN	High priority mode for master port 0 enable bit. If this bit is enabled, the priority of the master port is granted to the highest. 0: Disabled 1: Enabled
7:5	Reserved	Must be kept at reset value.
4	ARBMOD	Arbitration mode select 0: Round-robin priority mode selected 1: Fixed priority mode selected
3	DMPSEL	Default master port strategy select 0: Default access the slave port by the master port which is defined by DMP[2:0] bits when no master initiates request 1: Default access the slave port by the master port which is the last master accessed the slave port when no master initiates request
2:0	DMP[2:0]	Default master port 000: Default access the slave port by the master port M0 001: Default access the slave port by the master port M1 010: Default access the slave port by the master port M2 011: Default access the slave port by the master port M3 100: Default access the slave port by the master port M4 101: Default access the slave port by the master port M5 110: Default access the slave port by the master port M6 111: Default access the slave port by the master port M7

1.11.4. BM2 slave port priority configuration register x (BUSIM_BM2SPCFGx) (x = 0...3)

Address offset: 0x84 + 0x10 * x

Reset value: 0x0650 3210

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					M6PRIO[2:0]			Reserved	M5PRIO[2:0]			Reserved	M4PRIO[2:0]		
					rw				rw				rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	M3PRIO[2:0]			Reserved	M2PRIO[2:0]			Reserved	M1PRIO[2:0]			Reserved	M0PRIO[2:0]		
	rw				rw				rw				rw		

Bits	Fields	Descriptions
31:27	Reserved	Must be kept at reset value.
26:24	M6PRIO[2:0]	Priority of master port 6 L1 to L8 represent different priority levels, with L1 being the highest priority and L8 being the lowest priority. 000: Master has L1 priority on associated slave port 001: Master has L2 priority on associated slave port 010: Master has L3 priority on associated slave port 011: Master has L4 priority on associated slave port 100: Master has L5 priority on associated slave port 101: Master has L6 priority on associated slave port 110: Master has L7 priority on associated slave port 111: Master has L8 priority on associated slave port
23	Reserved	Must be kept at reset value.
22:20	M5PRIO[2:0]	Priority of master port 5 L1 to L8 represent different priority levels, with L1 being the highest priority and L8 being the lowest priority. 000: Master has L1 priority on associated slave port 001: Master has L2 priority on associated slave port 010: Master has L3 priority on associated slave port 011: Master has L4 priority on associated slave port 100: Master has L5 priority on associated slave port 101: Master has L6 priority on associated slave port 110: Master has L7 priority on associated slave port 111: Master has L8 priority on associated slave port
19	Reserved	Must be kept at reset value.
18:16	M4PRIO[2:0]	Priority of master port 4 L1 to L8 represent different priority levels, with L1 being the highest priority and L8 being the lowest priority. 000: Master has L1 priority on associated slave port 001: Master has L2 priority on associated slave port 010: Master has L3 priority on associated slave port 011: Master has L4 priority on associated slave port 100: Master has L5 priority on associated slave port

		101: Master has L6 priority on associated slave port 110: Master has L7 priority on associated slave port 111: Master has L8 priority on associated slave port
15	Reserved	Must be kept at reset value.
14:12	M3PRIO[2:0]	Priority of master port 3 L1 to L8 represent different priority levels, with L1 being the highest priority and L8 being the lowest priority. 000: Master has L1 priority on associated slave port 001: Master has L2 priority on associated slave port 010: Master has L3 priority on associated slave port 011: Master has L4 priority on associated slave port 100: Master has L5 priority on associated slave port 101: Master has L6 priority on associated slave port 110: Master has L7 priority on associated slave port 111: Master has L8 priority on associated slave port
11	Reserved	Must be kept at reset value.
10:8	M2PRIO[2:0]	Priority of master port 2 L1 to L8 represent different priority levels, with L1 being the highest priority and L8 being the lowest priority. 000: Master has L1 priority on associated slave port 001: Master has L2 priority on associated slave port 010: Master has L3 priority on associated slave port 011: Master has L4 priority on associated slave port 100: Master has L5 priority on associated slave port 101: Master has L6 priority on associated slave port 110: Master has L7 priority on associated slave port 111: Master has L8 priority on associated slave port
7	Reserved	Must be kept at reset value.
6:4	M1PRIO[2:0]	Priority of master port 1 L1 to L8 represent different priority levels, with L1 being the highest priority and L8 being the lowest priority. 000: Master has L1 priority on associated slave port 001: Master has L2 priority on associated slave port 010: Master has L3 priority on associated slave port 011: Master has L4 priority on associated slave port 100: Master has L5 priority on associated slave port 101: Master has L6 priority on associated slave port 110: Master has L7 priority on associated slave port 111: Master has L8 priority on associated slave port
3	Reserved	Must be kept at reset value.

2:0	M0PRIO[2:0]	Priority of master port 0 L1 to L8 represent different priority levels, with L1 being the highest priority and L8 being the lowest priority. 000: Master has L1 priority on associated slave port 001: Master has L2 priority on associated slave port 010: Master has L3 priority on associated slave port 011: Master has L4 priority on associated slave port 100: Master has L5 priority on associated slave port 101: Master has L6 priority on associated slave port 110: Master has L7 priority on associated slave port 111: Master has L8 priority on associated slave port

1.11.5. TMR fault control and status register (BUSIM_TMRFCS)

Address offset: 0x200

Reset value: 0x0000 0003

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												TMRERR_STAT1	TMRERR_STAT0	TMRERR_IE1	TMRERR_IE0
												rc_w1	rc_w1	rw	rw

Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3	TMRERR_STAT1	IMICU enable register TMR error status Hardware set and software can clear these fields by writing 1. 0: No TMR error occurred 1: TMR error occurred
2	TMRERR_STAT0	EXTMPU enable register TMR error status Hardware set and software can clear these fields by writing 1. 0: No TMR error occurred 1: TMR error occurred
1	TMRERR_IE1	TMR error 1 (IMICU) output to FMU enable This bit is set and cleared by software. 0: TMR fault disabled 1: TMR fault enabled
0	TMRERR_IE0	TMR error 0 (EXTMPU) output to FMU enable This bit is set and cleared by software. 0: TMR fault disabled

1: TMR fault enabled

1.12. EXTMPU registers definition

EXTMPU base address: 0x4900 A000

Access: privilege read/write

Table 1-19. EXTMPU register overview

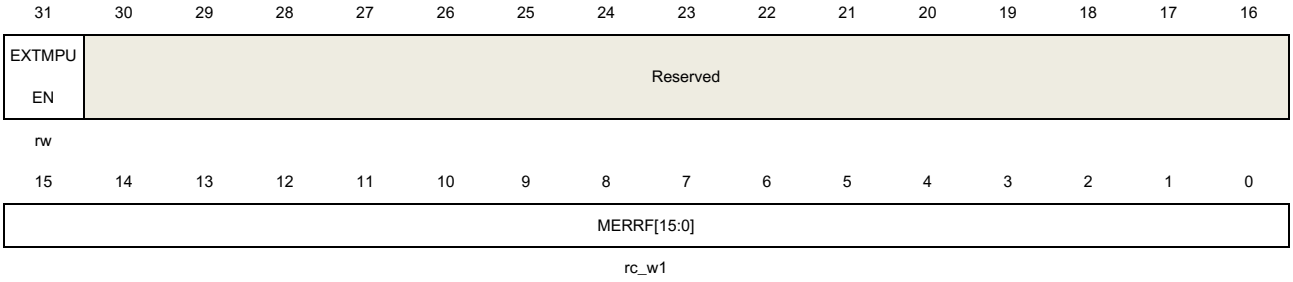
Register Name	Symbol	Address Offset	Reset Value
Control and status register 0	EXTMPU_CS0	0x00	0x0000 0000
Control and status register 1	EXTMPU_CS1	0x04	0x0000 0000
Error address register x (x=0, 2...4, 10...12,14)	EXTMPU_EADDRx	0x100 + x*0x10	0x0000 0000
Error information register x (x=0, 2...4, 10...12,14)	EXTMPU_EINFx	0x104 + x*0x10	0x0000 0000
Error region register x (x=0... 0, 2...4, 10...12,14)	EXTMPU_EREGx	0x108 + x*0x10	0x0000 0000
Region x control register 0(x=0...31)	EXTMPU_RxCTL0	0x200 + x*0x10	0x0000 0000
Region x control register1(x=0...31)	EXTMPU_RxCTL1	0x204 + x*0x10	0x0000 007F
Region x control register2(x=0...31)	EXTMPU_RxCTL2	0x208 + x*0x10	0x0000 0000
Region x control register3(x=0...31)	EXTMPU_RxCTL3	0x20C + x*0x10	0x0000 0000

1.12.1. Control and status register 0 (EXTMPU_CS0)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31	EXTMPUEN	Extended MPU enable 0: EXTMPU disable 1: EXTMPU enable
30:16	Reserved	Must be kept at reset value.
15:0	MERRF[15:0]	Error flag of master x It is cleared when writing 1 to it and corresponding error bit in EXTMPU_CS1 and EXTMPU_EINF will be cleared. 0: No error occurred 1: An error occurred for master x

1.12.2. Control and status register 1 (EXTMPU_CS1)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOVF[15:0]															
r															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	MOVF[15:0]	Overrun flag of master x 0: No error occurred 1: An overrun error occurred for master x

1.12.3. Error address register x (EXTMPU_EADDRx) (x = 0,2...4,10...12,14)

Address offset: 0x100 + x * 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ERRADDR[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERRADDR[15:0]															
r															

Bits	Fields	Descriptions
31:0	ERRADDR[31:0]	Error address bits These bits indicate the address from master which generates the access error.

1.12.4. Error information register x (EXTMPU_EINFx) (x = 0,2...4,10...12,14)

Address offset: 0x104 + x * 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved				EATYPE	ERRD[1:0]		Reserved								
				r	r										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27	EATYPE	Error access type 0: Read 1: Write
26:25	ERRD[1:0]	Error detailed information 00: Instruction access in user mode 01: Data access in user mode 10: Instruction access in supervisor mode 11: Data access in supervisor mode **Note**: Meaningless for ENET/DMA, it is the default value 00.
24:0	Reserved	Must be kept at reset value.

1.12.5. Error region register x (EXTMPU_EREGx) (x = 0,2...4,10...12,14)

Address offset: 0x108 + x * 0x10
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EACINF[31:16]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EACINF[15:0]															
r															

Bits	Fields	Descriptions
31:0	EACINF[31:0]	Error access control information bits These bits indicate the detailed access error region.

1.12.6. Region x control register 0 (EXTMPU_RxCTL0) (x = 0...31)

Address offset: 0x200 + x * 0x10
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SADDR[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR[15:0]															
rw															

Bits	Fields	Descriptions
31:0	SADDR[31:0]	Start address of region Defines the start address of region. Specifies the most significant bits of the 0-modulo-128 byte start address of the memory region. The minimum region size is 128 bytes. SADDR[6:0] is full with 0.

1.12.7. Region x control register 1 (EXTMPU_RxCTL1) (x = 0...31)

Address offset: 0x204 + x * 0x10
Reset value: 0x0000 007F

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EADDR[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EADDR[15:0]															
rw															

Bits	Fields	Descriptions
31:0	EADDR[31:0]	End address of region Defines the end address of region. Specifies the most significant bits of the 127-module-128 byte start address of the memory region. The minimum region size is 128 bytes. EADDR[6:0] is full with 1.

1.12.8. Region x control register 2 (EXTMPU_RxCTL2) (x = 0...31)

Address offset: 0x208 + x * 0x10
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
M15AP[1:0]		M14AP[1:0]		M13AP[1:0]		M12AP[1:0]		M11AP[1:0]		M10AP[1:0]		M9AP[1:0]		M8AP[1:0]	
rw		rw		rw		rw		rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

M7AP[1:0]	M6AP[1:0]	M5AP[1:0]	M4AP[1:0]	M3AP[1:0]	M2AP[1:0]	M1AP[1:0]	M0AP[1:0]
rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:30	M15AP[1:0]	Master 15 access permission Defines the read or write permission of master 15. Refer to M0AP[1:0] description
29:28	M14AP[1:0]	Master 14 access permission Defines the read or write permission of master 14. Refer to M0AP[1:0] description
27:26	M13AP[1:0]	Master 13 access permission Defines the read or write permission of master 13. Refer to M0AP[1:0] description
25:24	M12AP[1:0]	Master 12 access permission Defines the read or write permission of master 12. Refer to M0AP[1:0] description
23:22	M11AP[1:0]	Master 11 access permission Defines the read or write permission of master 11. Refer to M0AP[1:0] description
21:20	M10AP[1:0]	Master 10 access permission Defines the read or write permission of master 10. Refer to M0AP[1:0] description
19:18	M9AP[1:0]	Master 9 access permission Defines the read or write permission of master 9. Refer to M0AP[1:0] description
17:16	M8AP[1:0]	Master 8 access permission Defines the read or write permission of master 8. Refer to M0AP[1:0] description
15:14	M7AP[1:0]	Master 7 access permission Defines the read or write permission of master 7. Refer to M0AP[1:0] description
13:12	M6AP[1:0]	Master 6 access permission Defines the read or write permission of master 6. Refer to M0AP[1:0] description
11:10	M5AP[1:0]	Master 5 access permission Defines the read or write permission of master 5. Refer to M0AP[1:0] description
9:8	M4AP[1:0]	Master 4 access permission Defines the read or write permission of master 4.

		Refer to M0AP[1:0] description
7:6	M3AP[1:0]	Master 3 access permission Defines the read or write permission of master 3. Refer to M0AP[1:0] description
5:4	M2AP[1:0]	Master 2 access permission Defines the read or write permission of master 2. Refer to M0AP[1:0] description
3:2	M1AP[1:0]	Master 1 access permission Defines the read or write permission of master 1. Refer to M0AP[1:0] description
1:0	M0AP[1:0]	Master 0 access permission Defines the read or write permission of master 0. Bit0 controls the read access permission and bit1 controls the write access permission. If one of these two bits is set, the specific access permission is allowed. Otherwise, if it is 0, the specific access permission is not allowed.

1.12.9. Region x control register 3 (EXTMPU_RxCTL3) (x = 0...31)

Address offset: 0x20C + x * 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														RROEN	RVD
														rw	rw

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	RROEN	Region read only control bit Setting this bit to lock control register 0~3 of the specific region. When this bit is set, it can only be cleared by system reset. 0: Region can be read or written 1: Region is read only and write to the region is ignored
0	RVD	Region valid bit 0: Region is invalid 1: Region is valid

1.13. PRPU registers definition

PRPU base address: 0x4003 8000

Access: privilege read/write

Table 1-20. PRPU register overview

Register Name	Symbol	Address Offset	Reset Value
Lock protection register x(x = 0...104)	PRPU_LKPROTx	0x00+ x * 0x04	0x0000 0001

1.13.1. Lock protection register x (PRPU_LKPROTx) (x = 0...104)

Address offset: 0x00 + x * 0x04

Reset value: 0x0000 0001

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LKPROT_KEY[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LKPROT_KEY[15:0]															
rw															

Bits	Fields	Descriptions
31:0	LKPROT_KEY[31:0]	Lock peripheral register protection key bits These bits can be read and written by software. Read this register will return the result of locking the corresponding peripheral registers. Refer to ***Table 1-14. The memory map of the unlock key and corresponding peripheral registers of GD32A711/A712 devices*** for details. 1 means the peripheral registers can be configured and 0 means the peripheral registers are locked.

1.14. IMICU registers definition

IMICU_BM1 base address: 0x4900 9400

IMICU_BM2 base address: 0x4900 9800

Access: privilege read/write

Table 1-21. IMICU register overview

Register name	Symbol	Address offset	Reset value
Error detection control register	IMICU_EDCTL	0x00	0xFFFF 0000
Error injection control register	IMICU_EICTL	0x04	0x0000 0000
Error status register	IMICU_ERRSTAT	0x08	0x0000 0000

1.14.1. Error detection control register (IMICU_EDCTL)

Address offset: 0x00

Reset value: 0xFFFF 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S0EDCEN	S1EDCEN	S2EDCEN	S3EDCEN	S4EDCEN	S5EDCEN	S6EDCEN	S7EDCEN	M0EDCEN	M1EDCEN	M2EDCEN	M3EDCEN	Reserved	M5EDCEN	M6EDCEN	M7EDCEN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits	Fields	Descriptions
31	S0EDCEN	Slave port 0 (CM7_0 of BM1, BM1toBM2 of BM2) error detection enable bit This bit is set and cleared by software. 0: Feedback integrity check function disabled 1: Feedback integrity check function enabled
30	S1EDCEN	Slave port 1 (CM7_1 of BM1, DMA0_MEM of BM2) error detection enable bit This bit is set and cleared by software. 0: Feedback integrity check function disabled 1: Feedback integrity check function enabled
29	S2EDCEN	Slave port 2 (DMA0_MEM of BM1, DMA1MEM of BM2) error detection enable bit This bit is set and cleared by software. 0: Feedback integrity check function disabled 1: Feedback integrity check function enabled
28	S3EDCEN	Slave port 3 (DMA1_MEM of BM1, AHBP(CM7_0) of BM2) error detection enable bit This bit is set and cleared by software.

		0: Feedback integrity check function disabled 1: Feedback integrity check function enabled
27	S4EDCEN	Slave port 4 (HSM of BM1, AHBP(CM7_1) of BM2) error detection enable bit This bit is set and cleared by software. 0: Feedback integrity check function disabled 1: Feedback integrity check function enabled
26	S5EDCEN	Slave port 5 (ENET of BM1, DMA0_PERI of BM2) error detection enable bit This bit is set and cleared by software. 0: Feedback integrity check function disabled 1: Feedback integrity check function enabled
25	S6EDCEN	Slave port 6 (DMA0_PERI of BM1, DMA1_PERI of BM2) error detection enable bit This bit is set and cleared by software. 0: Feedback integrity check function disabled 1: Feedback integrity check function enabled
24	S7EDCEN	Slave port 7 (DMA1_PERI of BM1, reserved of BM2) error detection enable bit This bit is set and cleared by software. 0: Feedback integrity check function disabled 1: Feedback integrity check function enabled
23	M0EDCEN	Master port 0 (FMC-0 of BM1, AHB1 of BM2) error detection enable bit This bit is set and cleared by software. 0: Attribute integrity check function disabled 1: Attribute integrity check function enabled
22	M1EDCEN	Master port 1 (FMC-1 of BM1, AHB2 of BM2) error detection enable bit This bit is set and cleared by software. 0: Attribute integrity check function disabled 1: Attribute integrity check function enabled
21	M2EDCEN	Master port 2 (FMC-2 of BM1, APB1 of BM2) error detection enable bit This bit is set and cleared by software. 0: Attribute integrity check function disabled 1: Attribute integrity check function enabled
20	M3EDCEN	Master port 3 (SRAM0 of BM1, APB2 of BM2) error detection enable bit This bit is set and cleared by software. 0: Attribute integrity check function disabled 1: Attribute integrity check function enabled
19	Reserved	Must be kept at reset value.
18	M5EDCEN	Master port 5 (TCMSRAM of BM1, reserved of BM2) error detection enable bit This bit is set and cleared by software. 0: Attribute integrity check function disabled 1: Attribute integrity check function enabled

17	M6EDCEN	Master port 6 (QSPI of BM1, reserved of BM2) error detection enable bit This bit is set and cleared by software. 0: Attribute integrity check function disabled 1: Attribute integrity check function enabled
16	M7EDCEN	Master port 7 (reserved of BM1, reserved of BM2 for GD32A711/A712 devices) error detection enable bit This bit is set and cleared by software. 0: Attribute integrity check function disabled 1: Attribute integrity check function enabled
15:0	Reserved	Must be kept at reset value.

1.14.2. Error injection control register (IMICU_EICTL)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EIEN_DATAP	SPID_DATAP[3:0]				MPID_DATAP[3:0]				EDCC_DATAPL[6:0]						
rw	rw				rw				rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EIEN	SPID[3:0]				MPID[3:0]				EDCC[6:0]						
rw	rw				rw				rw						

Bits	Fields	Descriptions
31	EIEN_DATAP	Error injection function enable bit with data phase 0: Error injection function disabled 1: Error injection function enable
30:27	SPID_DATAP[3:0]	Target slave port ID According to master ID. **Note:** SPID_DATAP bits cannot be configured to 4b1100. Values 4b0000, 4b0001, 4b0010, 4b0011, 4b0100, 4b0101, 4b0111, 4b1010, 4b1011, 4b1110 are supported, refer to ***Table 1-12. Master ID of GD32A711/A712 devices***. Specify the target slave ID for error injection.
26:23	MPID_DATAP[3:0]	Target master port ID According to slave ID Specify the target master ID for error injection
22:16	EDCC_DATAP[6:0]	Calculated EDC code value with data phase
15	EIEN	Error injection function enable bit

		0: Error injection function disabled 1: Error injection function enable
14:11	SPID[3:0]	Target slave port ID According to master ID. Specify the target slave ID for error injection.
10:7	MPID[3:0]	Target master port ID According to slave ID. Specify the target master ID for error injection
6:0	EDCC[6:0]	Calculated EDC code value

1.14.3. Error status register (IMICU_ERRSTAT)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ERR_DATADET	SPID_DATADET[3:0]				MPID_DATADET[3:0]				EDCC_DATADET[6:0]						
r	r				r				r						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR_DET	SPID_DET[3:0]				MPID_DET[3:0]				EDCC_DET[6:0]						
r	r				r				r						

Bits	Fields	Descriptions
31	ERR_DATADET	Error detected flag 0: No error detected 1: Error detected
30:27	SPID_DATADET[3:0]	Target slave port with integrity check error detected According to master ID.
26:23	MPID_DATADET[3:0]	Target master port with integrity check error detected According to slave ID.
22:16	EDCC_DATADET[6:0]	EDC code calculated with data phase error detected
15	ERR_DET	Error detected flag 0: No error detected 1: Error detected
14:11	SPID_DET[3:0]	Target slave port with integrity check error detected

		According to master ID.
10:7	MPID_DET[3:0]	Target master port with integrity check error detected According to slave ID.
6:0	EDCC_DET[6:0]	EDC code calculated with integrity check error detected

1.15. SRAM registers definition

SRAM_REG base address: 0x4900 0C00

Table 1-22. SRAM register overview

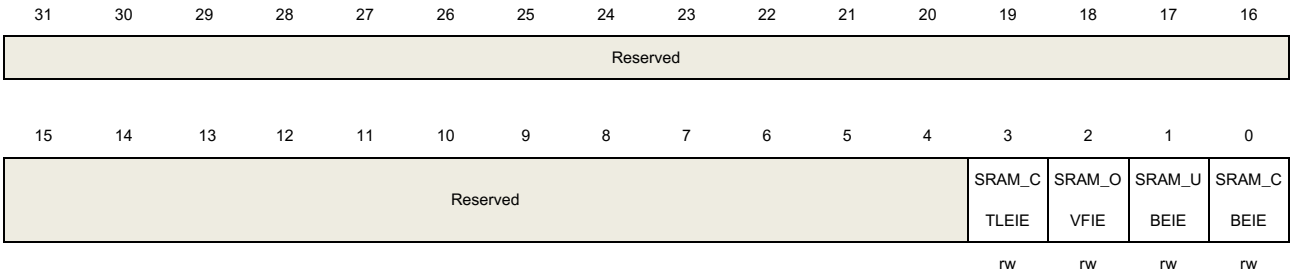
Register name	Symbol	Address offset	Reset value
SRAM interrupt enable register	SRAM_INTEN	0x00	0x0000 0000
SRAM error status register	SRAM_ERR	0x04	0x0000 0000
SRAM correctable bit error address 0 for SRAM_m register	SRAM_CBEADDR0_m	0x08	0x2400 0000
SRAM correctable bit error address 1 for SRAM_m register	SRAM_CBEADDR1_m	0x10	0x2400 0000
SRAM correctable bit error address 2 for SRAM_m register	SRAM_CBEADDR2_m	0x18	0x2400 0000
SRAM correctable bit error address 3 for SRAM_m register	SRAM_CBEADDR3_m	0x20	0x2400 0000
SRAM uncorrectable bit error address for SRAM_m register	SRAM_UBEADDR_m	0x28	0x2400 0000
SRAM buffer valid register	SRAM_BUFVLD	0x30	0x0000 0000
SRAM overflow status register	SRAM_OVF	0x34	0x0000 0000
SRAM write protection register	SRAM_WP	0x38	0x0000 0000
SRAM ECC code error status register	SRAM_ECCCERR	0x3C	0x0000 0000

1.15.1. SRAM interrupt enable register (SRAM_INTEN)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3	SRAM_CTLEIE	SRAM0 controller redundancy check error interrupt enable 0: SRAM0 controller redundancy check error interrupt disable 1: SRAM0 controller redundancy check error interrupt enable

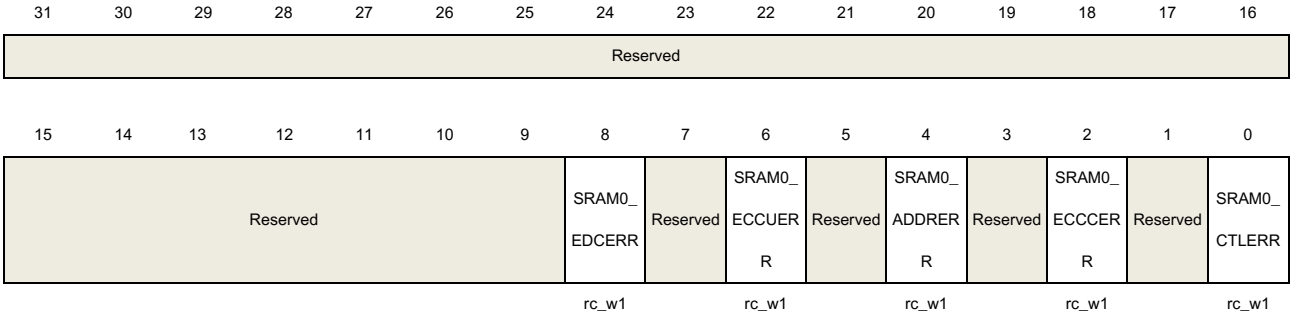
2	SRAM_OVFIE	SRAM0 address buffer overflow interrupt enable 0: SRAM0 address buffer overflow interrupt disable 1: SRAM0 address buffer overflow interrupt enable
1	SRAM_UBEIE	SRAM0 uncorrectable bit error interrupt enable 0: SRAM0 uncorrectable bit error interrupt disable 1: SRAM0 uncorrectable bit error interrupt enable
0	SRAM_CBEIE	SRAM0 correctable bit error interrupt enable 0: SRAM0 correctable bit error interrupt disable 1: SRAM0 correctable bit error interrupt enable

1.15.2. SRAM error status register (SRAM_ERR)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8	SRAM0_EDCERR	SRAM0 EDC after ECC checker error flag The software can clear it by writing 1. 0: No SRAM EDC after ECC checker error 1: SRAM EDC after ECC checker error generated
7	Reserved	Must be kept at reset value.
6	SRAM0_ECCUERR	SRAM0 ECC double bit uncorrectable error flag The software can clear it by writing 1. 0: No SRAM ECC double bit uncorrectable error 1: SRAM ECC double bit uncorrectable error generated
5	Reserved	Must be kept at reset value.
4	SRAM0_ADDRERR	SRAM0 ECC single bit address error flag The software can clear it by writing 1. 0: No SRAM ECC single bit address error

		1: SRAM ECC single bit address error generated
3	Reserved	Must be kept at reset value.
2	SRAM0_ECCCERR	SRAM0 ECC single bit correctable error flag The software can clear it by writing 1. 0: No SRAM ECC single bit correctable error 1: SRAM ECC single bit correctable error generated
1	Reserved	Must be kept at reset value.
0	SRAM0_CTLERR	SRAM0 controller redundancy check error flag The software can clear it by writing 1. 0: No SRAM controller redundancy check error 1: SRAM controller redundancy check error generated

1.15.3. SRAM correctable bit error address 0 for SRAM register (SRAM_CBEADDR0)

Address offset: 0x08

Reset value: 0x2400 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRAM_CBEADDR0[31:16]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRAM_CBEADDR0[15:0]															
r															

Bits	Fields	Descriptions
31:0	SRAM_CBEADDR0[31:0]	Record the first correctable bit error base address (aligned to 64bit). Any bits in 64bit may be wrong.

1.15.4. SRAM correctable bit error address 1 for SRAM register (SRAM_CBEADDR1)

Address offset: 0x10

Reset value: 0x2400 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRAM_CBEADDR1[31:16]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

SRAM_CBEADDR1[15:0]
r

Bits	Fields	Descriptions
31:0	SRAM_CBEADDR1[31:0]	Record the second correctable bit error base address (aligned to 64bit). Any bits in 64bit may be wrong.

1.15.5. SRAM correctable bit error address 2 for SRAM register (SRAM_CBEADDR2)

Address offset: 0x18

Reset value: 0x2400 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRAM_CBEADDR2[31:16]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRAM_CBEADDR2[15:0]															
r															

Bits	Fields	Descriptions
31:0	SRAM_CBEADDR2[31:0]	Record the third correctable bit error base address (aligned to 64bit). Any bits in 64bit may be wrong.

1.15.6. SRAM correctable bit error address 3 for SRAM register (SRAM_CBEADDR3)

Address offset: 0x20

Reset value: 0x2400 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRAM_CBEADDR3[31:16]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRAM_CBEADDR3[15:0]															
r															

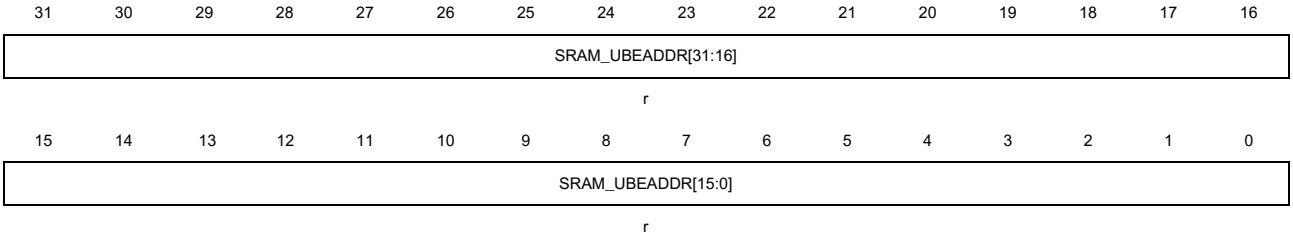
Bits	Fields	Descriptions
31:0	SRAM_CBEADDR3[31:0]	Record the fourth correctable bit error base address (aligned to 64bit). Any bits in 64bit may be wrong.

1.15.7. SRAM uncorrectable bit error address for SRAM register (SRAM_UBEADDR)

Address offset: 0x28

Reset value: 0x2400 0000

This register has to be accessed by word (32-bit).



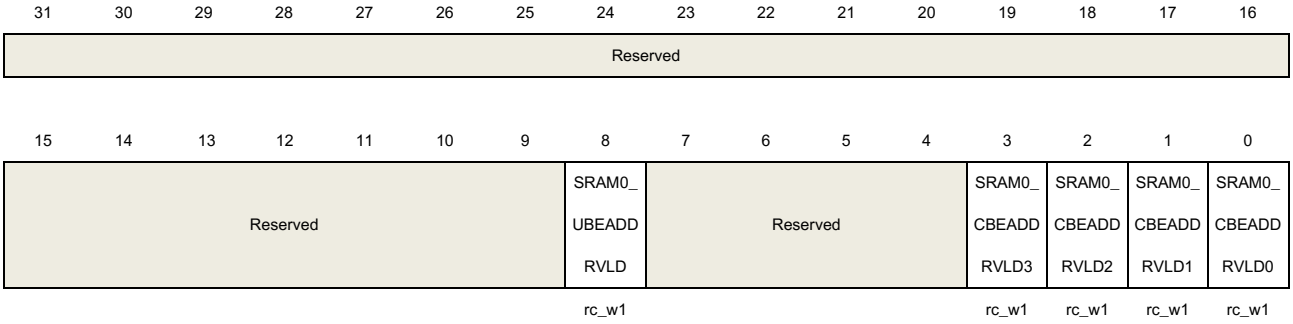
Bits	Fields	Descriptions
31:0	SRAM_UBEADDR[31:0]	Record the un-correctable bit error base address (aligned to 64bit). Any bits in 64bit may be wrong.

1.15.8. SRAM buffer valid register (SRAM_BUFVLD)

Address offset: 0x30

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



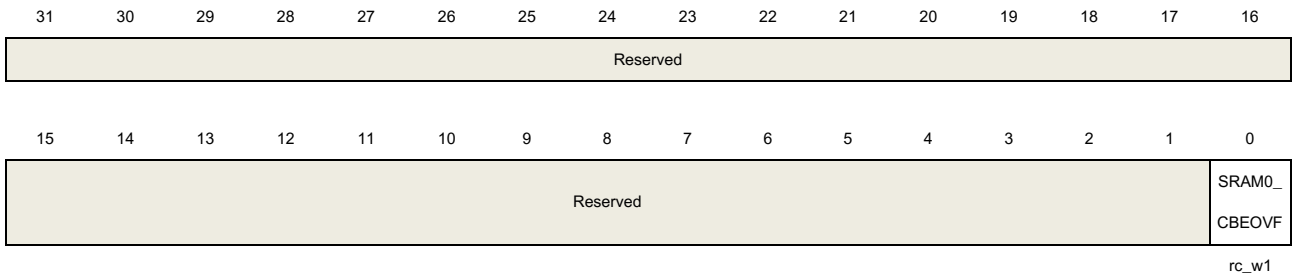
Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8	SRAM0_UBEADDRVLD	SRAM0 un-correctable bit error address valid flag The software can clear it by writing 1. 0: No SRAM uncorrectable bit error address in buffer 1: There is SRAM un-correctable bit error address in buffer
7:4	Reserved	Must be kept at reset value.
3	SRAM0_CBEADDRVLD3	SRAM0 correctable bit error address3 valid flag The software can clear it by writing 1. 0: No SRAM correctable bit error address3 in buffer

		1: There is SRAM correctable bit error address3 in buffer
2	SRAM0_CBEADDRV LD2	SRAM0 correctable bit error address2 valid flag The software can clear it by writing 1. 0: No SRAM correctable bit error address2 in buffer 1: There is SRAM correctable bit error address2 in buffer
1	SRAM0_CBEADDRV LD1	SRAM0 correctable bit error address1 valid flag The software can clear it by writing 1. 0: No SRAM correctable bit error address1 in buffer 1: There is SRAM correctable bit error address1 in buffer
0	SRAM0_CBEADDRV LD0	SRAM0 correctable bit error address0 valid flag The software can clear it by writing 1. 0: No SRAM correctable bit error address0 in buffer 1: There is SRAM correctable bit error address0 in buffer

1.15.9. SRAM overflow status register (SRAM_OVF)

Address offset: 0x34
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	SRAM0_CBEOVF	SRAM0 correctable bit error address buffer overflow The software can clear it by writing 1. 0: No SRAM correctable bit error address buffer overflow 1: SRAM correctable bit error address buffer overflow

1.15.10. SRAM write protection register (SRAM_WP)

Address offset: 0x38
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SRAM_P 9WP	SRAM_P 8WP	SRAM_P 7WP	SRAM_P 6WP	SRAM_P 5WP	SRAM_P 4WP	SRAM_P 3WP	SRAM_P 2WP	SRAM_P 1WP	SRAM_P 0WP
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:0	SRAM_PxWP (x=0 to 9)	SRAM page x write protection (a page granularity is 16Kbyte) 0: Write protection of SRAM page x is disabled. 1: Write protection of SRAM page x is enabled.

1.15.11. SRAM ECC code error status register (SRAM_ECCCERR)

Address offset: 0x3C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							SRAM0_ECCCDET[8:0]								
							r								

Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8:0	SRAM0_ECCCDET[8:0]	SRAM0 ecc code calculated with check error detected to identify the position on a single-bit inversion

1.16. TCM registers definition

TCM_REG base address: 0x4900 4000

Table 1-23. TCM register overview

Register name	Symbol	Address offset	Reset value
TCM interrupt enable register	TCM_INTEN	0x00	0x0000 0000
TCM error status register	TCM_ERR	0x04	0x0000 0000
TCM correctable bit error address 0 for SRAM_m register	TCM_CBEADDR0_m	0x08 + 0x04*m	0x0000 0000 (m=0)
TCM correctable bit error address 1 for SRAM_m register	TCM_CBEADDR1_m	0x20+ 0x04*m	0x0000 0000 (m=0)
TCM correctable bit error address 2 for SRAM_m register	TCM_CBEADDR2_m	0x38+ 0x04*m	0x0000 0000 (m=0)
TCM correctable bit error address 3 for SRAM_m register	TCM_CBEADDR3_m	0x50+ 0x04*m	0x0000 0000 (m=0)
TCM uncorrectable bit error address for SRAM_m register	TCM_UBEADDR_m	0x68+ 0x04*m	0x0000 0000 (m=0)
TCM buffer valid register	TCM_BUFVLD	0x80	0x0000 0000
TCM overflow status register	TCM_OVF	0x84	0x0000 0000
TCM write protection register	TCM_WP	0x88	0x0000 0000
TCM0 ECC code error status register	TCM0_ECCCERR	0x8C	0x0000 0000

1.16.1. TCM interrupt enable register (TCM_INTEN)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TCM0_CTLEIE	Reserved	TCM0_OVFIE	Reserved	TCM0_UBEIE	Reserved	TCM0_CBEIE
									rw		rw		rw		rw

Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	TCM0_CTLEIE	TCM of CM7_0 controller redundancy check error interrupt enable 0: TCM of CM7_0 controller redundancy check error interrupt disable 1: TCM of CM7_0 controller redundancy check error interrupt enable

5	Reserved	Must be kept at reset value.
4	TCM0_OVFIE	TCM of CM7_0 address buffer overflow interrupt enable 0: TCM of CM7_0 address buffer overflow interrupt disable 1: TCM of CM7_0 address buffer overflow interrupt enable
3	Reserved	Must be kept at reset value.
2	TCM0_UBEIE	TCM of CM7_0 uncorrectable bit error interrupt enable 0: TCM of CM7_0 uncorrectable bit error interrupt disable 1: TCM of CM7_0 uncorrectable bit error interrupt enable
1	Reserved	Must be kept at reset value.
0	TCM0_CBEIE	TCM of CM7_0 correctable bit error interrupt enable 0: TCM of CM7_0 correctable bit error interrupt disable 1: TCM of CM7_0 correctable bit error interrupt enable

1.16.2. TCM error status register (TCM_ERR)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					D1TCM0_EDCERR	D0TCM0_EDCERR	ITCM0_EDCERR	Reserved					D1TCM0_ECCUERR	D0TCM0_ECCUERR	ITCM0_ECCUERR
					rc_w1	rc_w1	rc_w1						rc_w1	rc_w1	rc_w1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					D1TCM0_ADDRERR	D0TCM0_ADDRERR	ITCM0_ADDRERR	Reserved			D1TCM0_ECCCERR	D0TCM0_ECCCERR	ITCM0_ECCCERR	Reserved	TCM0_CTLERR
					rc_w1	rc_w1	rc_w1				rc_w1	rc_w1	rc_w1		rc_w1

Bits	Fields	Descriptions
31:27	Reserved	Must be kept at reset value.
26	D1TCM0_EDCERR	D1TCM of CM7_0 EDC after ECC checker error flag The software can clear it by writing 1. 0: No D1TCM of CM7_0 EDC after ECC checker error 1: D1TCM of CM7_0 EDC after ECC checker error generated
25	D0TCM0_EDCERR	D0TCM of CM7_0 EDC after ECC checker error flag The software can clear it by writing 1. 0: No D0TCM of CM7_0 EDC after ECC checker error

		1: D0TCM of CM7_0 EDC after ECC checker error generated
24	ITCM0_EDCERR	ITCM of CM7_0 EDC after ECC checker error flag The software can clear it by writing 1. 0: No ITCM of CM7_0 EDC after ECC checker error 1: ITCM of CM7_0 EDC after ECC checker error generated
23:19	Reserved	Must be kept at reset value.
18	D1TCM0_ECCUERR	D1TCM of CM7_0 ECC double bit uncorrectable error flag The software can clear it by writing 1. 0: No D1TCM of CM7_0 ECC double bit uncorrectable error 1: D1TCM of CM7_0 ECC double bit uncorrectable error generated
17	D0TCM0_ECCUERR	D0TCM of CM7_0 ECC double bit uncorrectable error flag The software can clear it by writing 1. 0: No D0TCM of CM7_0 ECC double bit uncorrectable error 1: D0TCM of CM7_0 ECC double bit uncorrectable error generated
16	ITCM0_ECCUERR	ITCM of CM7_0 ECC double bit uncorrectable error flag The software can clear it by writing 1. 0: No ITCM of CM7_0 ECC double bit uncorrectable error 1: ITCM of CM7_0 ECC double bit uncorrectable error generated
15:11	Reserved	Must be kept at reset value.
10	D1TCM0_ADDRERR	D1TCM of CM7_0 single bit address error flag The software can clear it by writing 1. 0: No D1TCM of CM7_0 single bit address error 1: D1TCM of CM7_0 ECC single bit address error generated
9	D0TCM0_ADDRERR	D0TCM of CM7_0 single bit address error flag The software can clear it by writing 1. 0: No D0TCM of CM7_0 single bit address error 1: D0TCM of CM7_0 ECC single bit address error generated
8	ITCM0_ADDRERR	ITCM of CM7_0 single bit address error flag The software can clear it by writing 1. 0: No ITCM of CM7_0 single bit address error 1: ITCM of CM7_0 ECC single bit address error generated
7:5	Reserved	Must be kept at reset value.
4	D1TCM0_ECCCERR	D1TCM of CM7_0 single bit correctable error flag The software can clear it by writing 1. 0: No D1TCM of CM7_0 single bit correctable error 1: D1TCM of CM7_0 ECC single bit correctable error generated
3	D0TCM0_ECCCERR	D0TCM of CM7_0 single bit correctable error flag The software can clear it by writing 1.

		0: No D0TCM of CM7_0 single bit correctable error 1: D0TCM of CM7_0 ECC single bit correctable error generated
2	ITCM0_ECCCERR	ITCM of CM7_0 single bit correctable error flag The software can clear it by writing 1. 0: No ITCM of CM7_0 single bit correctable error 1: ITCM of CM7_0 ECC single bit correctable error generated
1	Reserved	Must be kept at reset value.
0	TCM0_CTLERR	TCM of CM7_0 controller redundancy check error flag The software can clear it by writing 1. 0: No TCM of CM7_0 controller redundancy check error 1: TCM of CM7_0 controller redundancy check error generated

1.16.3. TCM correctable bit error address 0 for SRAM_m register (TCM_CBEADDR0_m) (m = 0...2)

Address offset: 0x08 + 0x04*m

Reset value:

0x0000_0000 for CM7_0_ITCM (m=0)

0x0000_0000 for CM7_0_D0TCM (m=1)

0x0000_0004 for CM7_0_D1TCM (m=2)

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TCM_CBEADDR0[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCM_CBEADDR0[15:0]															
r															

Bits	Fields	Descriptions
31:0	TCM_CBEADDR0[31:0]	Record the first correctable bit error offset address (ITCM is aligned to 64bit and DTCM is aligned to 32bit). Any bits in 64bit/32bit may be wrong.

1.16.4. TCM correctable bit error address 1 for SRAM_m register (TCM_CBEADDR1_m) (m = 0...2)

Address offset: 0x20 + 0x04*m

Reset value:

0x0000_0000 for CM7_0_ITCM (m=0)

0x0000_0000 for CM7_0_D0TCM (m=1)

0x0000_0004 for CM7_0_D1TCM (m=2)

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TCM_CBEADDR1[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCM_CBEADDR1[15:0]															
r															

Bits	Fields	Descriptions
31:0	TCM_CBEADDR1[31:0]	Record the second correctable bit error offset address (ITCM is aligned to 64bit and DTCM is aligned to 32bit). Any bits in 64bit/32bit may be wrong.

1.16.5. TCM correctable bit error address 2 for SRAM_m register (TCM_CBEADDR2_m) (m = 0...2)

Address offset: 0x38 + 0x04*m

Reset value:

0x0000_0000 for CM7_0_ITCM (m=0)

0x0000_0000 for CM7_0_D0TCM (m=1)

0x0000_0004 for CM7_0_D1TCM (m=2)

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TCM_CBEADDR2[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCM_CBEADDR2[15:0]															
r															

Bits	Fields	Descriptions
31:0	TCM_CBEADDR2[31:0]	Record the third correctable bit error offset address (ITCM is aligned to 64bit and DTCM is aligned to 32bit). Any bits in 64bit/32bit may be wrong.

1.16.6. TCM correctable bit error address 3 for SRAM_m register (TCM_CBEADDR3_m) (m = 0...2)

Address offset: 0x50 + 0x04*m

Reset value:

0x0000_0000 for CM7_0_ITCM (m=0)

0x0000_0000 for CM7_0_D0TCM (m=1)

0x0000_0004 for CM7_0_D1TCM (m=2)

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TCM_CBEADDR3[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCM_CBEADDR3[15:0]															
r															

Bits	Fields	Descriptions
31:0	TCM_CBEADDR3[31:0]	Record the third correctable bit error offset address (ITCM is aligned to 64bit and DTCM is aligned to 32bit). Any bits in 64bit/32bit may be wrong.

1.16.7. TCM uncorrectable bit error address for SRAM_m register (TCM_UBEADDR_m) (m = 0...2)

Address offset: 0x68 + 0x04*m

Reset value:

0x0000_0000 for CM7_0_ITCM (m=0)

0x0000_0000 for CM7_0_D0TCM (m=1)

0x0000_0004 for CM7_0_D1TCM (m=2)

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TCM_UBEADDR[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCM_UBEADDR[15:0]															
r															

Bits	Fields	Descriptions
31:0	TCM_UBEADDR[31:0]	Record the un-correctable bit error offset address (ITCM is aligned to 64bit and DTCM is aligned to 32bit). Any bits in 64bit/32bit may be wrong.

1.16.8. TCM buffer valid register (TCM_BUFVLD)

Address offset: 0x80

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved	D1TCM0_UBEADDRVLD	D0TCM0_UBEADDRVLD	ITCM0_UBEADDRVLD	Reserved
	rc_w1	rc_w1	rc_w1	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				D1TCM0_CBEADDRVLD3	D1TCM0_CBEADDRVLD2	D1TCM0_CBEADDRVLD1	D1TCM0_CBEADDRVLD0	D0TCM0_CBEADDRVLD3	D0TCM0_CBEADDRVLD2	D0TCM0_CBEADDRVLD1	D0TCM0_CBEADDRVLD0	ITCM0_CBEADDRVLD3	ITCM0_CBEADDRVLD2	ITCM0_CBEADDRVLD1	ITCM0_CBEADDRVLD0
				rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions
31:29	Reserved	Must be kept at reset value.
28	D1TCM0_UBEADDRVLD	D1TCM of CM7_0 un-correctable bit error address valid flag The software can clear it by writing 1. 0: No D1TCM of CM7_0 un-correctable bit error address in buffer 1: There is D1TCM of CM7_0 un-correctable bit error address in buffer
27	D0TCM0_UBEADDRVLD	D0TCM of CM7_0 un-correctable bit error address valid flag The software can clear it by writing 1. 0: No D0TCM of CM7_0 un-correctable bit error address in buffer 1: There is D0TCM of CM7_0 un-correctable bit error address in buffer
26	ITCM0_UBEADDRVLD	ITCM of CM7_0 un-correctable bit error address valid flag The software can clear it by writing 1. 0: No ITCM of CM7_0 un-correctable bit error address in buffer 1: There is ITCM of CM7_0 uncorrectable bit error address in buffer
25:12	Reserved	Must be kept at reset value.
11	D1TCM0_CBEADDRVLD3	D1TCM of CM7_0 correctable bit error address3 valid flag The software can clear it by writing 1. 0: No D1TCM of CM7_0 correctable bit error address3 in buffer 1: There is D1TCM of CM7_0 correctable bit error address3 in buffer
10	D1TCM0_CBEADDRVLD2	D1TCM of CM7_0 correctable bit error address2 valid flag The software can clear it by writing 1. 0: No D1TCM of CM7_0 correctable bit error address2 in buffer 1: There is D1TCM of CM7_0 correctable bit error address2 in buffer
9	D1CM_CBEADDRVLD1	D1TCM of CM7_0 correctable bit error address1 valid flag The software can clear it by writing 1. 0: No D1TCM of CM7_0 correctable bit error address1 in buffer 1: There is D1TCM of CM7_0 correctable bit error address1 in buffer
8	D1TCM0_CBEADDRVLD0	D1TCM of CM7_0 correctable bit error address0 valid flag The software can clear it by writing 1. 0: No D1TCM of CM7_0 correctable bit error address0 in buffer

		1: There is D1TCM of CM7_0 correctable bit error address0 in buffer
7	D0TCM0_CBEADDR VLD3	D0TCM of CM7_0 correctable bit error address3 valid flag The software can clear it by writing 1. 0: No D0TCM of CM7_0 correctable bit error address3 in buffer 1: There is D0TCM of CM7_0 correctable bit error address3 in buffer
6	D0TCM0_CBEADDR VLD2	D0TCM of CM7_0 correctable bit error address2 valid flag The software can clear it by writing 1. 0: No D0TCM of CM7_0 correctable bit error address2 in buffer 1: There is D0TCM of CM7_0 correctable bit error address2 in buffer
5	D0CM_CBEADDRVL D1	D0TCM of CM7_0 correctable bit error address1 valid flag The software can clear it by writing 1. 0: No D0TCM of CM7_0 correctable bit error address1 in buffer 1: There is D0TCM of CM7_0 correctable bit error address1 in buffer
4	D0TCM0_CBEADDR VLD0	D0TCM of CM7_0 correctable bit error address0 valid flag The software can clear it by writing 1. 0: No D0TCM of CM7_0 correctable bit error address0 in buffer 1: There is D0TCM of CM7_0 correctable bit error address0 in buffer
3	ITCM0_CBEADDRVL D3	ITCM of CM7_0 correctable bit error address3 valid flag The software can clear it by writing 1. 0: No ITCM of CM7_0 correctable bit error address3 in buffer 1: There is ITCM of CM7_0 correctable bit error address3 in buffer
2	ITCM0_CBEADDRVL D2	ITCM of CM7_0 correctable bit error address2 valid flag The software can clear it by writing 1. 0: No ITCM of CM7_0 correctable bit error address2 in buffer 1: There is ITCM of CM7_0 correctable bit error address2 in buffer
1	ITCM0_CBEADDRVL D1	ITCM of CM7_0 correctable bit error address1 valid flag The software can clear it by writing 1. 0: No ITCM of CM7_0 correctable bit error address1 in buffer 1: There is ITCM of CM7_0 correctable bit error address1 in buffer
0	ITCM0_CBEADDRVL D0	ITCM of CM7_0 correctable bit error address0 valid flag The software can clear it by writing 1. 0: No ITCM of CM7_0 correctable bit error address0 in buffer 1: There is ITCM of CM7_0 correctable bit error address0 in buffer

1.16.9. TCM overflow status register (TCM_OVF)

Address offset: 0x84

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													D1TCM0_CBEOVF	D0TCM0_CBEOVF	ITCM0_CBEOVF
													rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2	D1TCM0_CBEOVF	D1TCM of CM7_0 correctable bit error address buffer overflow The software can clear it by writing 1. 0: No D1TCM of CM7_0 correctable bit error address buffer overflow 1: D1TCM of CM7_0 correctable bit error address buffer overflow
1	D0TCM0_CBEOVF	D0TCM of CM7_0 correctable bit error address buffer overflow The software can clear it by writing 1. 0: No D0TCM of CM7_0 correctable bit error address buffer overflow 1: D0TCM of CM7_0 correctable bit error address buffer overflow
0	ITCM0_CBEOVF	ITCM of CM7_0 correctable bit error address buffer overflow The software can clear it by writing 1. 0: No ITCM of CM7_0 correctable bit error address buffer overflow 1: ITCM of CM7_0 correctable bit error address buffer overflow

1.16.10. TCM write protection register (TCM_WP)

Address offset: 0x88

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								DTCM0_P7WP	DTCM0_P6WP	DTCM0_P5WP	DTCM0_P4WP	DTCM0_P3WP	DTCM0_P2WP	DTCM0_P1WP	DTCM0_P0WP
								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												ITCM0_P3WP	ITCM0_P2WP	ITCM0_P1WP	ITCM0_P0WP
												rw	rw	rw	rw

Bits	Fields	Descriptions

31:24	Reserved	Must be kept at reset value.
23:16	DTCM0_PxWP (x=0 to 7)	DTCM of CM7_0 page x write protection (a page granularity is 8 Kbyte) 0: Write protection of DTCM of CM7_0 x is disabled. 1: Write protection of DTCM of CM7_0 x is enabled.
15:8	Reserved	Must be kept at reset value.
3:0	ITCM0_PxWP (x=0 to 3)	ITCM of CM7_0 page x write protection (a page granularity is 8 Kbyte) 0: Write protection of ITCM of CM7_0 x is disabled. 1: Write protection of ITCM of CM7_0 x is enabled.

1.16.11. TCM0 ECC code error status register (TCM0_ECCCERR)

Address offset: 0x8C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:24	D1TCM0_ECCCDET[7:0]	When a correctable error is detected, the calculated D1TCM of CM7_0 ECC code is used to determine the specific error location.
23:16	D0TCM0_ECCCDET[7:0]	When a correctable error is detected, the calculated D0TCM of CM7_0 ECC code is used to determine the specific error location.
15:9	Reserved	Must be kept at reset value.
8:0	ITCM0_ECCCDET[8:0]	When a correctable error is detected, the calculated ITCM of CM7_0 ECC code is used to determine the specific error location.

1.17. Device electronic signature

The device electronic signature contains memory size information and the 96-bit unique device ID. It is stored in the information block of the Flash memory. The 96-bit unique device ID is unique for any device. It can be used as serial numbers, or part of security keys, etc.

1.17.1. Memory density information

Base address: 0x1FFF F7E0

The value is factory programmed and can never be altered by user.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRAM_DENSITY[15:0]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLASH_DENSITY[15:0]															
r															

Bits	Fields	Descriptions
31:16	SRAM_DENSITY [15:0]	SRAM density The value indicates the on-chip SRAM density of the device in Kbytes. Example: 0x0080 indicates 128 Kbytes.
15:0	FLASH_DENSITY [15:0]	Flash memory density The value indicates the Flash memory density of the device in Kbytes. Example: 0x0200 indicates 512 Kbytes.

1.17.2. Unique device ID (96 bits)

Base address: 0x1FFF F7E8

The value is factory programmed and can never be altered by user.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UNIQUE_ID[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNIQUE_ID[15:0]															
r															

Bits	Fields	Descriptions
31:0	UNIQUE_ID[31:0]	Unique device ID

Base address: 0x1FFF F7EC

The value is factory programmed and can never be altered by user.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UNIQUE_ID[63:48]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNIQUE_ID[47:32]															
r															

Bits	Fields	Descriptions

31:0	UNIQUE_ID[63:32]	Unique device ID

Base address: 0x1FFF F7F0

The value is factory programmed and can never be altered by user.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UNIQUE_ID[95:80]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNIQUE_ID[79:64]															
r															

Bits	Fields	Descriptions
31:0	UNIQUE_ID[95:64]	Unique device ID

2. Flash memory controller (FMC)

2.1. Overview

The flash memory controller, FMC, provides all the necessary functions for the on-chip flash memory. The flash memory controller provides flash configuration and control functions and manages the interface between the flash memory array and the AHB bus (two 64-bit AHB buses and two 32-bit AHB buses are implemented on FMC). It also provides page erase, mass erase, and program operations for flash memory.

2.2. Characteristics

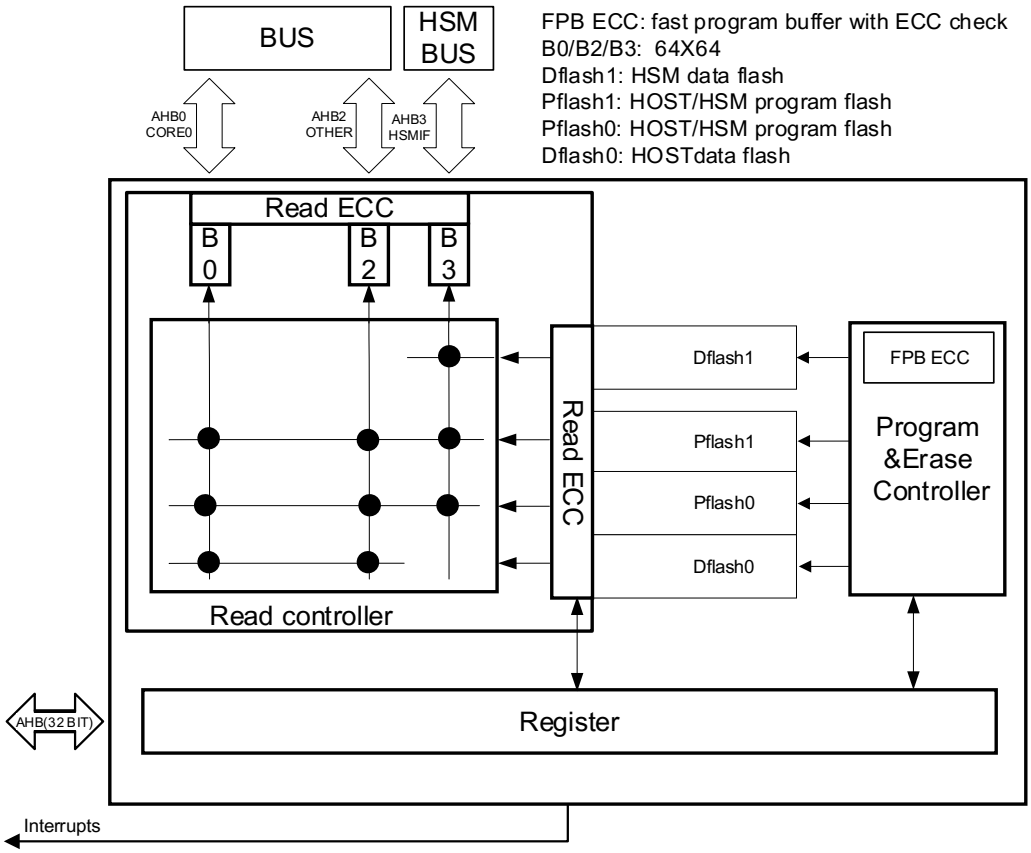
- Up to 2048 KB of main flash memory for instruction and data. Up to 2KB OTP. Up to 128KB Data flash0 which can be emulated as EEPROM by software. Up to 96KB Data flash1 for HSM.
  - Bank0: 1024KB, shared with HSM and HSM flash size is configurable
  - Bank1: 1024KB, shared with HSM and HSM flash size is configurable
  - Data flash0 block: 128KB shared for data flash
  - Data flash1 block: 96KB for HSM data flash
  - OTP: 2KB
- Quad flash block architecture for read-while-write (RWW) capability.
- ECC with single bit error corrected and double bit errors detected.
- Triple voted flops for safety-critical flash functions (internal trimming, redundancy, mode control, etc.).
- 0~5 waiting time within Bank0 / Bank1 / Data flash0 when CPU executes instructions and read data.
- Flexible buffers to speed read operations for three AHB port.
- The flash page size is 2KB.
- Double word programming, page erase and mass erase operation.
- 2KB OTP (one-time program) block used for user data storage.
- 64B option bytes block for user application requirements.
- 16B HSM option bytes.
- Option bytes are uploaded to the option byte control registers when the system is reset.
- Flash security protection to prevent illegal code / data access.
- Page erase / program protection to prevent unexpected operation.
- Fast program support.
- Support TMR (Triple Modular Redundancy) for important control register.
- Support controller redundancy.
- Support 64 / 128 / 1024 bit ECC (data + address).

2.3. Function overview

2.3.1. Flash memory architecture

The flash memory controller has three ports, each with dedicated read buffers. These dedicated read buffers allow each master to access Flash without its buffered data being displaced by read operations from other masters. As well, by having these separate ports, it is possible to have separate connections for each CPU's instruction bus, and one port for all data accesses as shown in *Figure 2-1. FMC block diagram of GD32A711 / A712 devices*.

Figure 2-1. FMC block diagram of GD32A711 / A712 devices



The flash memory consists of up to 2048 KB HOST main flash, which is organized into 1024 pages with 2KB capacity, a 24 KB information block for the boot loader, up to 128KB Data flash0 block for HOST, up to 96KB Data flash1 block for HSM. Each page of HOST main flash memory can be erased individually. *Table 2-1. Base address and size for flash memory of GD32A711 / A712 devices* shows the base address and size.

Table 2-1. Base address and size for flash memory of GD32A711 / A712 devices

Block			Name	Address	Size (bytes)	ECC error address area
Main Flash Block	Pflash0(1)	Bank0 (Max	Bank0	HOSTSIZE = 0b11: 0x0800 0000 ~	768KB ~ 1024KB	User area: 0 ~ 0x1FFFF

Block			Name	Address	Size (bytes)	ECC error address area
Shared with HSM		1024KB) swap		(0x080F FFFF - HSM SIZE)		
				HOSTSIZE = 0b01: 0x0800 0000 - 0x0807 FFFF	512KB	User area: 0 ~ 0xFFFF
		HSM bank0 (0 ~ 256KB) swap	HSM bank0	0x0844 0000 ~ (0x0844 0000 +HSM SIZE)	0KB ~ 256KB	System area: 18'hxxxxx ~ 0x1FFFF
	Pflash1(1)	Bank1 (Max 1024KB) swap	Bank1	HOSTSIZE = 0b11: (0x0810 0000 - HSM SIZE) ~ (0x081F FFFF - HSM SIZE*2)	768KB ~ 1024KB	User area: 0 ~ 0x1FFFF
				HOSTSIZE = 0b01: 0x0808 0000 - 0x080F FFFF	512KB	User area: 0 ~ 0xFFFF
		HSM bank1 (0 ~ 256KB) swap	HSM bank1	0x0848 0000 ~ (0x0848 0000 +HSM SIZE)	0KB ~ 256KB	System area: 18'hxxxxx ~ 0x1FFFF
Dflash0 Block			Data flash(2)	0x0880 0000 ~ 0x0881 FFFF	128KB	User area: 0 ~ 0x3FFF
Dflash1 Block			HSM Data flash	0x0888 0000 ~ 0x0889 7FFF	96KB	User area: 0 ~ 0x2FFF
			sBAF	0x0840 0000 ~ 0x0840 7FFF	32KB	User area: 0x3000 ~ 0x3FFF
Information Block			Bootloader(3)	0x1FFF 9800 ~ 0x1FFF C7FF @Pflash0	12KB	System area: 0 ~ 0x5FF
				0x1FFF C800 ~ 0x1FFF F7FF @Pflash1	12KB	System area: 0 ~ 0x5FF
Option byte Block			HOST Option bytes	0x1FFF 8800 ~ 0x1FFF 883F @Dflash0	64B	System area: 0 ~ 0x7
			HSM Option bytes	0x1FFF F880 ~ 0x1FFF F88F @Dflash1	16B	System area: 0 ~ 0x1
One-time program Block			OTP bytes(4)	0x1FFF 7000 ~ 0x1FFF 77FF @Dflash0	2KB	System area: 0x100 ~ 0x1FF

Block	Name	Address	Size (bytes)	ECC error address area
	HSM OTP bytes	0x1FFFF900 ~ 0x1FFF F957 (88 bytes for life cycle) @Dflash1	88B	System area: 0x200 ~ 0x20A
		0x1FFF F958 ~ 0x1FFF F95F (8 bytes for HOST dbg disable) @Dflash1 0x1FFF F960 ~ 0x1FFF F967 (8 bytes for HSM dbg disable) @Dflash1	16B	System area: 0x10B ~ 0x10C
		0x1FFF F968 ~ 0x1FFF F977 (16 bytes for HOST dbg password) @Dflash1 0x1FFF F978 ~ 0x1FFF F987 (16 bytes for HSM dbg password) @Dflash1	32B	System area: 0x10D ~ 0x110
		0x1FFF F988 ~ 0x1FFF F9C7 (64 bytes for cryp key) @Dflash1	64B	System area: 0x111 ~ 0x118
		0x1FFF F9C8 ~ 0x1FFF F9D7 (16 bytes for spc key) @Dflash1	16B	System area: 119 ~ 11A
		0x1FFF F9D8 ~ 0x1FFF F9DF (8 bytes for HSM size) @Dflash1	8B	System area: 0x11B
		0x1FFF F9E0 ~ 0x1FFF F9E7 (8 bytes for HSM security) @Dflash1	8B	System area: 0x11C
		0x1FFF F9E8 ~ 0x1FFF F9F7 (16 bytes for factory mode key) @Dflash1	16B	System area: 0x11D ~ 0x11E

Note:

1. When HOSTSIZE = 0b11, Bank0 flash and HSM bank0 share total 1024KB flash block at Pflash0, Bank1 flash and HSM bank1 share total 1024KB flash block at Pflash1. HSM bank0 / HSM bank1 size can be obtained by calculating HSMSIZE[4:0] bit fields, HOST bank0 / HOST bank1 size can be obtained by calculating HOSTSIZE[1:0] bit fields, which refer to ***HSM Erase / Program protection register (FMC_HSMWP)***. Bank0 and Bank1 are aliases of HOST bank0 and HOST bank1 in this section.

2. Data flash in Dflash0 Block is used for HOST.

3. The Information Block stores the boot loader. This block cannot be programmed or erased by user.

4. 2 Kbyte (256 double word) OTP (one-time programmable) data area is for user, the OTP area is only operated by Data flash register. The OTP data cannot be erased and can be written only once. If any bit has been set 0, the entire double word cannot be written anymore, even with the value 0x0000 0000 0000 0000.

5. HSM OTP modification is valid after subsystem reset.

2.3.2. Error Checking and Correcting (ECC)

The ECC mechanism supports:

- One error detection and correction.
- Two errors detection.
- Support 128bit ECC (data + address) for Buffer0 / Buffer1 / Buffer2 / Buffer3 / Fast program buffer.
- Support 1024bit ECC (data + address) for Pflash0 / Pflash1.
- Support 64bit ECC for Dflash0 / Dflash1.
- Support hardware EDC after ECC feature to check the ECC logic for Buffer0 / Buffer1 / Buffer2 / Buffer3 / Fast program buffer / Pflash0 / Pflash1.
- Support fault injection for ECC (data + address) and EDC after ECC.
- Pflash0, Pflash1, Dflash0 and Dflash1 have individual ECC check function.
- Buffer0, Buffer1, Buffer2, Buffer3 and fast program buffer have individual ECC check function.
- 4 correctable error status and error address registers per macro.
- 1 uncorrectable error status and error address register per macro.

When one error is detected during read Buffer or flash memory:

- When occurred in any area include Pflash0, Plash1, Dflash0 and Dflash1, the ECCCORy (y = 0, 1, 2, 3) bit in FMC_BLKxECCCS register will be set. If the ECCCORIE bit in the FMC_BLKxECCCS register is set, the correctable ECC error signal is reported to the FMU. The FMC_BLKxCECCADDRy register indicates the offset address in each area. User can transform offset address to logic address by memory map. When more 4 correctable error occurred, ECCCOROV bit in FMC_BLKxECCCS register will be set.

- When occurred in any Buffer include Buffer0, Buffer1, Buffer2 and Buffer3, the ECCCORy (y = 0, 1, 2, 3) bit in FMC_BUFxECCCS register will be set. If the ECCCORIE bit in the FMC_BUFxECCCS register is set, the correctable ECC error signal is reported to the FMU. The FMC_BUFxCECCADDRy register indicates the logical address in each area. When more 4 correctable error occurred, ECCCOROV bit in FMC_BUFxECCCS register will be set.
- When occurred in fast program buffer area, the ECCCORy (y = 0, 1, 2, 3) bit in FMC_FPB_ECCCS register will be set. If the ECCCORIE bit in the FMC_FPB_ECCCS register is set, the correctable ECC error signal is reported to the FMU. The FMC_FPB_CECCADDRx register indicates the offset address in this area. User can transform offset address to logic address by adding FMC_ADDRx. When more 4 correctable error occurred, ECCCOROV bit in FMC_FPB_ECCCS register will be set.

When two errors are detected or EDC after ECC error are detected:

- When occurred in any area include Pflash0, Plash1, Dflash0 and Dflash1, the ECCDET bit in FMC_BLKxECCCS register will be set. And the NMI interrupt will be generated by FMU. The ECCDETIE bit enables reporting of uncorrectable ECC errors to the FMU, while the ECCCOROVIE bit enables reporting of correctable ECC error overflow to the FMU. The FMC_BLKxUCECCADDR register indicates the offset address in each area. And read the error address which returns all F data. User can transform offset address to logic address by memory map.
- When occurred in any area include Buffer0, Buffer1, Buffer2 and Buffer3, the ECCDET bit in FMC_BUFxECCCS register will be set. And the NMI interrupt will be generated by FMU. The ECCDETIE bit enables reporting of uncorrectable ECC errors to the FMU, while the ECCCOROVIE bit enables reporting of correctable ECC error overflow to the FMU. The FMC_BUFxUCECCADDR register indicates the logical address in each area.
- When occurred in fast program buffer area, the ECCDET bit in FMC_FPB_ECCCS register will be set. And the NMI interrupt can be generated by FMU. The ECCDETIE bit enables reporting of uncorrectable ECC errors to the FMU, while the ECCCOROVIE bit enables reporting of correctable ECC error overflow to the FMU. The FMC_FPB_UCECCADDR register indicates the offset address in this area. User can transform offset address to logic address by adding FMC_ADDRx.

Note:

1. For a virgin data 0xFF FFFF FFFF FFFF FFFF, ECC is not supported.

2. The correctable and uncorrectable error address indicate the first error occurred in single bit error and double bit errors. If ECCCORy or ECCDET set, the value will not be changed even if a new error occurred.

3. When an ECC error is reported, a new read at the error address may not generate an ECC error if the data is still present in the buffer, even if ECCCOR and ECCDET are cleared.

4. Each time the fast program buffer is used, the same address will be read twice. If a fixed ECC error occurs, twice errors will be generated.

5. In flash programming (not flash fast programming), there is a read operation to check whether the programming address data is 0xFFFF FFFF FFFF FFFF, and the read process will execute ECC check.

6. Check blank operation will execute ECC check.

7. For the GD32A711 / A712 devices, Buffer1 does not exist.

8. After the prefetch function is enabled, when the CPU executes a new read request and the buffer misses, if the ECC error occurs in the prefetch address (the address not read by the CPU), the ECC error will be directly reported.

When developing FLS (Flash Driver) drivers, if cache and prefetch function is used, the FLS driver read judgment method need to follow these steps:

1. Firstly, clear the corresponding ECC error flag after two ECC errors occur.
2. Then, disable cache and prefetch functions and read the same address again. If the ECC error flag is set again and the ECC error address is consistent with the read address, it is considered a real ECC error.
3. Finally, clear the corresponding ECC error flag again, turn on the cache and prefetch functions and the read failure status is returned to FEE (Flash EEPROM Emulation).

2.3.3. Read operations

The flash can be addressed directly as a common memory space. Any instruction fetches and the data access from the flash are through the AHB64 BUS from the master (Core0 / HSM / DMA0 / DMA1 / ENET).

HSM master can read all flash resource.

HOST master (Core0 / DMA0 / DMA1 / ENET) cannot read Dflash1 resource anytime.

HOST master can read HSM bank0 / HSM bank1 when HSM security is not set, HOST master cannot read HSM bank0 / HSM bank1 when HSM security is set.

Wait state added

The WSCNT bits in the FMC_WS register needs to be configured correctly depend on the AHB clock frequency when reading the flash memory. The relation between WSCNT and AHB clock frequency is show as the ***Table 2-2. The relation between WSCNT and AHB clock frequency when LDO is 1.1V***.

Table 2-2. The relation between WSCNT and AHB clock frequency when LDO is 1.1V

AHB clock frequency	WSCNT configured
<= 23MHz	0 (0 wait state added)
<= 47MHz	1 (1 wait state added)
<= 71MHz	2 (2 wait state added)
<= 94MHz	3 (3 wait state added)
<= 120MHz	4 (4 wait state added)

<= 160MHz	5 (5 wait state added)

Note:

1. If it is necessary to increase the AHB clock frequency, firstly, refer to ***Table 2-2. The relation between WSCNT and AHB clock frequency when LDO is 1.1V***, configure the WSCNT bits according to the target AHB clock frequency. Then, increase the AHB clock frequency to the target frequency. It is forbidden to increase the AHB clock frequency before configuring the WSCNT.

2. If it is necessary to decrease the AHB clock frequency, firstly, decrease the target AHB clock frequency. Then refer to ***Table 2-2. The relation between WSCNT and AHB clock frequency when LDO is 1.1V***, configure the WSCNT bits according the target AHB clock frequency. It is forbidden to configure the WSCNT bits before decrease the AHB clock frequency.

Considering that the wait state is added, the read efficiency is very low (such as add 4 wait state when 120MHz). In order to speed up the read access, there are some functions performed as below.

Buffer and prefetch function

The flash memory controller has three ports, each with dedicated read buffers. Thus allowing the line buffers to be more efficiently utilized because the various masters have dedicated buffers which cannot be compromised by read operations performed by the other masters. The buffer mechanism serves to deliver flash memory read data with zero-wait state response on lines that reside in cache. AHB requests that miss the prefetch cache generate the needed flash memory array access and are forwarded to the AHB upon completion.

Two 64-bit AHB ports and one 32-bit AHB port allow simultaneous access to their respective dedicated read buffers.

Read buffering and line prefetching function are configurable. Each AHB port can get a single-cycle buffer hit read response via a mini-cache and a prefetch controller.

When AHB read request occurs, a lookup of mini-cache and evaluation of access permission are performed during the AHB addressing phase. If the requested data is present in a read buffer, it is returned on the system bus with zero wait states. Otherwise, the controller accesses the Flash memory.

The buffers are shared for code and data fetches, and can be controlled independently by FMC_BUFxECCCS registers.

A read access may trigger prefetch of the next sequential line in the cycle following the request. The prefetched data is loaded into the least-recently-used read buffer. The access address is incremented by 32 bytes and a subsequent flash access is executed.

Buffers architecture shows in ***Figure 2-2. Buffer0 / Buffer2 / Buffer3 architecture of GD32A711 / A712 devices***.

The read buffer and prefetch configuration for each master is shown in *Table 2-3. Read Buffer / Prefetch enable bit allocation of GD32A711 / A712 devices*.

Figure 2-2. Buffer0 / Buffer2 / Buffer3 architecture of GD32A711 / A712 devices

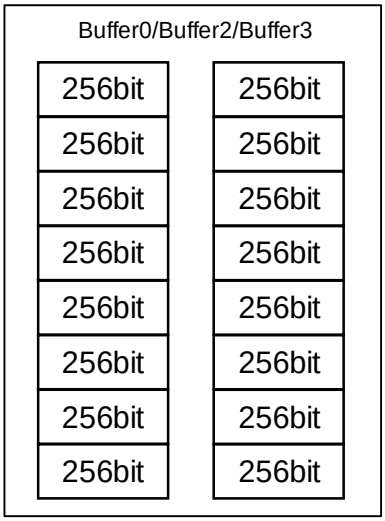


Table 2-3. Read Buffer / Prefetch enable bit allocation of GD32A711 / A712 devices

Master	Register	Buffer enable bit	Prefetch enable bit
Cortex-M7_0	FMC_BUF0ECCCS	CBUFEN, DBUFEN	CPFEN, DPFEN
DMA + ENET + HSM (other)	FMC_BUF2ECCCS	CBUFEN, DBUFEN	CPFEN, DPFEN
HSM (Dflash1 + HSM bank0/1)	FMC_BUF3ECCCS	CBUFEN, DBUFEN	CPFEN, DPFEN

2.3.4. Quad block architecture with read-while-write (RWW) capability

The Flash memory features a quad block architecture based on Pflash0 (Block0), Pflash1 (Block1), Dflash0 (Block2) and Dflash1 (Block3). This architecture supports the RWW (read-while-write) capability. It means that while a read or program operation is performed in a block, the other block can be accessed for another operation (read or program) without the need to wait for the end of operation on the first block.

2.3.5. FMC write interface

There are two write control interface, interface 0 and interface 1.

Interface 0 include FMC_KEY0, FMC_STAT0, FMC_CTL0, FMC_ADDR0, FMC_WDATAL0, FMC_WDATAH0 register, which can program and erase Pflash0, Pflash1 and Dflash0.

Interface 1 include FMC_KEY1, FMC_STAT1, FMC_CTL1, FMC_ADDR1, FMC_WDATAL1, FMC_WDATAH1 register, which can only program and erase Dflash1.

Note:

1. For interface 0, When HSM SEC is enabled, the HSM zone can only be written or erased by the HSM (configurable for non-secure applications), if the condition is not satisfied, PGSERR error will occur. For interface 1, When HSM SEC is enabled, the HSM zone can only be written or erased by the HSM (configurable for non-secure applications), if the condition is not satisfied, FMC_KEY1 will not unlock.

2. Before performing erase / program operations using FMC_CTLx (x = 0,1), ensure that the PMU clock is enabled and the VALVRNF flag in the PMU_LVCS register is cleared to zero. Otherwise, the Flash will enter a prohibited state for erase / program operations.

2.3.6. Unlock the FMC_CTLx register

After reset, the FMC_CTLx (x = 0,1) register is not accessible in write mode, except for the OBRLD bit, which is used for reloading the option byte, and the LK bit in FMC_CTLx register is 1. An unlocking sequence consists of two write operations to the FMC_KEYx register can open the access to the FMC_CTLx register. When the FMC is in lock state, the two write operations are writing 0x45670123 + x and 0xCDEF89AB + x to the FMC_KEYx register by same AHB master. If the second write operations to FMC_KEYx is not 0xCDEF89AB + x or AHB master is not same, the write operations must restart from writing 0x45670123 + x.

After the two write operations, the LK bit in FMC_CTLx register is cleared by hardware. When the writing operation (program, erase or check blank) finish or invalid program or erase operation finish, LK bit in FMC_CTLx register is set by hardware. The software can lock the FMC_CTLx again by setting the LK bit in FMC_CTLx register, with same master.

If there are any wrong operations on the FMC_KEYx register, the LK bit in FMC_CTLx register will be set, and the FMC_CTLx register will be locked, then it will generate a bus error.

Note:

1. The second write to FMC_KEYx need to check whether the master id is same with the first write to FMC_KEYx. If the master id is not same which will not unlock the FMC and no bus error occurs.

2. The FMC_CTLx can only be locked by the same master which last unlocked it.

The master id is shown in ***Table 1-12. Master ID of*** .

2.3.7. Page erase

The FMC provides a page erase function which is used for initializing the contents of a flash memory (Bank0 / Bank1/ HSM bank0 / HSM bank1 / HOST data flash / HSM data flash) page to a high state. Each page can be erased independently without affecting the contents of other pages. The following steps show the access sequence of the register for a page erase operation.

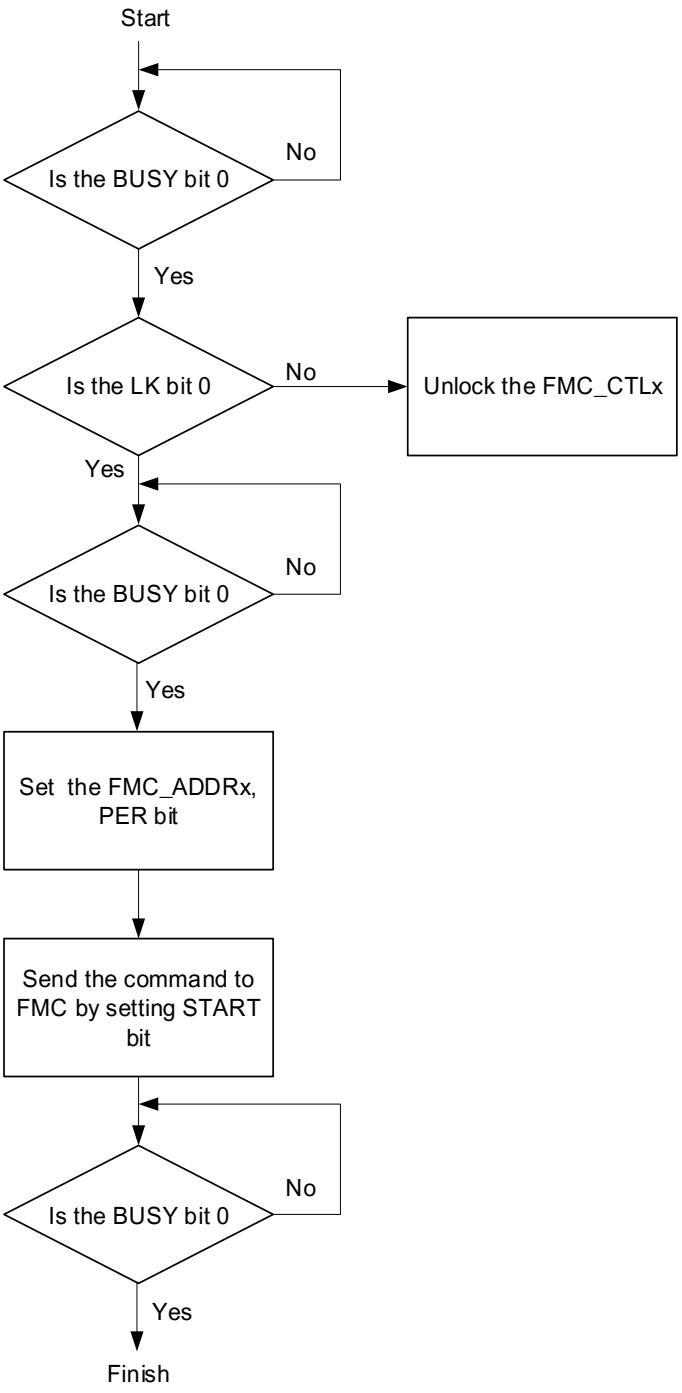
After Unlock the FMC_CTLx register, the FMC_ADDRx and FMC_CTLx must can only be written by same master.

- Check the BUSY bit in FMC_STATx register to confirm that no flash memory operation is in progress (BUSY equal to 0). Otherwise, wait until the operation has been finished.
- Unlock the FMC_CTLx register.
- Double check the BUSY bit in FMC_STATx register to confirm that no flash memory operation is in progress (BUSY equal to 0). Otherwise, wait until the operation has been finished.
- Write the page address into the FMC_ADDRx register.
- Write the page erase command into PER bit in FMC_CTLx register.
- Send the page erase command to the FMC by setting the START bit in FMC_CTLx register.
- Wait until all the operations have been completed by checking the value of the BUSY bit in FMC_STATx register.
- Read and verify the page if required using a read access.

When the operation is executed successfully, the ENDF in FMC_STATx register will be set. An interrupt will be triggered by FMC if the ENDIE bit in the FMC_CTLx register is set. Note that a correct target page address must be confirmed. Otherwise, the software may run out of control if the target erase page is being used for fetching codes or accessing data. The FMC will not provide any notification when it occurs. Additionally, the page erase operation will be ignored on protected pages. Flash operation error interrupt will be triggered by the FMC if the ERRIE bit in the FMC_CTLx register is set. The software can check the WPERR bit in the FMC_STATx register to detect this condition in the interrupt handler. The ***Figure 2-3. Process of page erase operation*** shows the page erase operation flow.

Note: Before performing erase operations using FMC_CTLx (x = 0,1), ensure that the PMU clock is enabled and the VALVRNF flag in the PMU_LVCS register is cleared to zero. Otherwise, the Flash will enter a prohibited state for erase operations.

Figure 2-3. Process of page erase operation



2.3.8. Mass erase

The FMC provides a complete erase function which is used for initializing the Flash Block (Bank0 / Bank1 / HOST data flash) contents. The following steps show the mass erase register access sequence.

After Unlock the FMC_CTL0 register, the FMC_ADDR0 and FMC_CTL0 must can only be

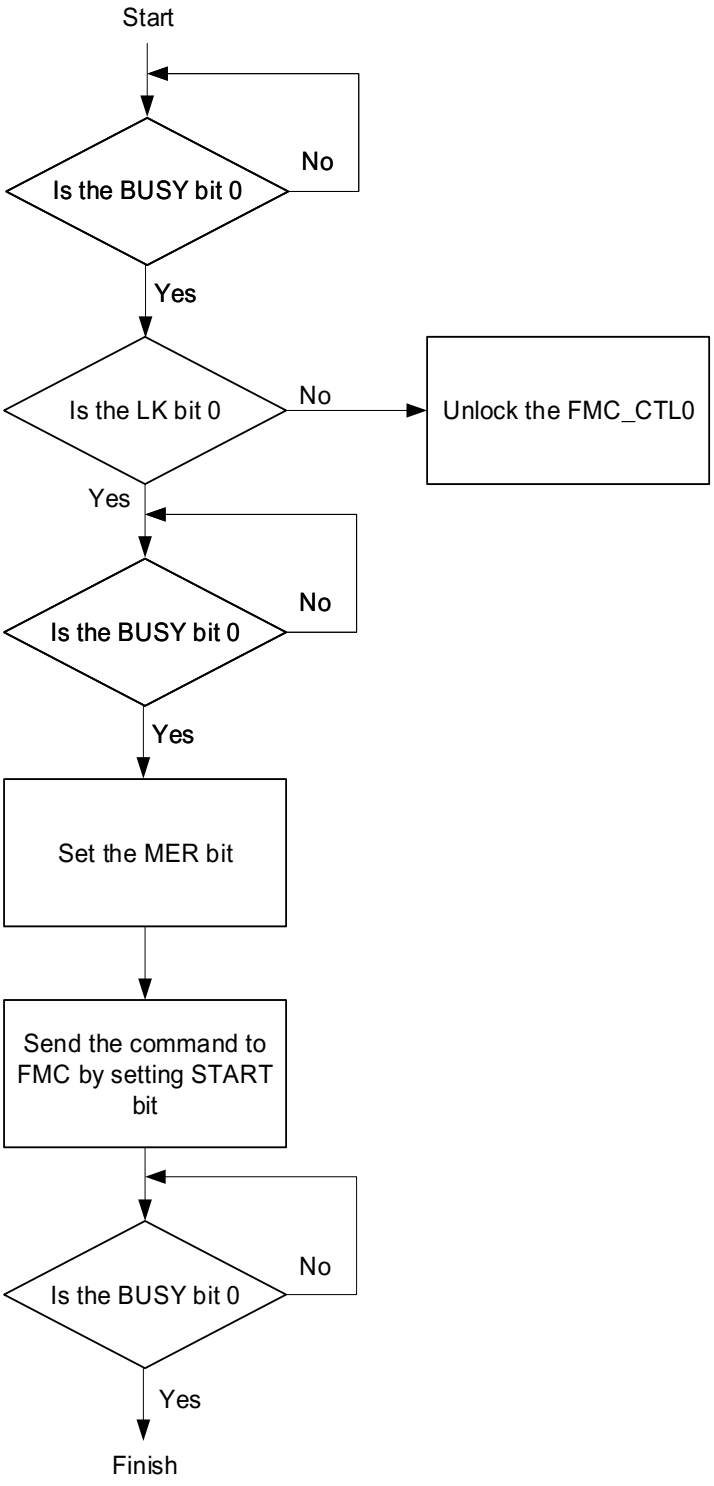
written by same master.

- Check the BUSY bit in FMC_STAT0 register to confirm that no flash memory operation is in progress (BUSY equal to 0). Otherwise, wait until the operation has been finished.
- Unlock the FMC_CTL0 register.
- Double check the BUSY bit in FMC_STAT0 register to confirm that no flash memory operation is in progress (BUSY equal to 0). Otherwise, wait until the operation has been finished.
- Write the any block (Bank0 / Bank1/ HOST data flash) address into the FMC_ADDR0 register.
- Write the mass erase command into MER bit in FMC_CTL0 register.
- Send the mass erase command to the FMC by setting the START bit in FMC_CTL0 register.
- Wait until all the operations have been completed by checking the value of the BUSY bit in FMC_STAT0 register.
- Unlocks and clears MER bits.
- Read and verify the flash memory if required using a read access.

When the operation is executed successfully, an interrupt will be triggered by FMC if the ENDIE bit in the FMC_CTL0 register is set, and the ENDF in FMC_STAT0 register is set. Since all flash data will be reset to a value of 0xFFFF FFFF, the mass erase operation can be implemented by using a program that runs in SRAM or by using the debugging tool to access the FMC registers directly. Additionally, the mass erase operation will be ignored if any page is erase / program protected. In this condition, a flash operation error interrupt will be triggered by the FMC if the ERRIE bit in the FMC_CTL0 register is set. The software can check the WPERR bit in the FMC_STAT0 register to detect this condition in the interrupt handler. The ***Figure 2-4. Process of mass erase operation*** indicates the mass erase operation flow.

Note: Before performing mass erase operations using FMC_CTL0, ensure that the PMU clock is enabled and the VALVRNF flag in the PMU_LVCS register is cleared to zero. Otherwise, the Flash will enter a prohibited state for mass erase operations.

Figure 2-4. Process of mass erase operation



Note: The mass erase of Bank0 / Bank1 is split to page erase by hardware.

2.3.9. Flash programming

The FMC provides programming function by WDATALx / WDATAHx which is used to modify

the flash memory (Bank0 / Bank1/ HSM bank0 / HSM bank1 / HOST data flash / HSM data flash) contents. While actually, the data program to flash memory is 64-bits. The following steps show the register access sequence of the programming operation.

After unlock the FMC_CTLx register, the FMC_ADDRx, FMC_CTLx, WDATALx and WDATAHx must can only be written by same master.

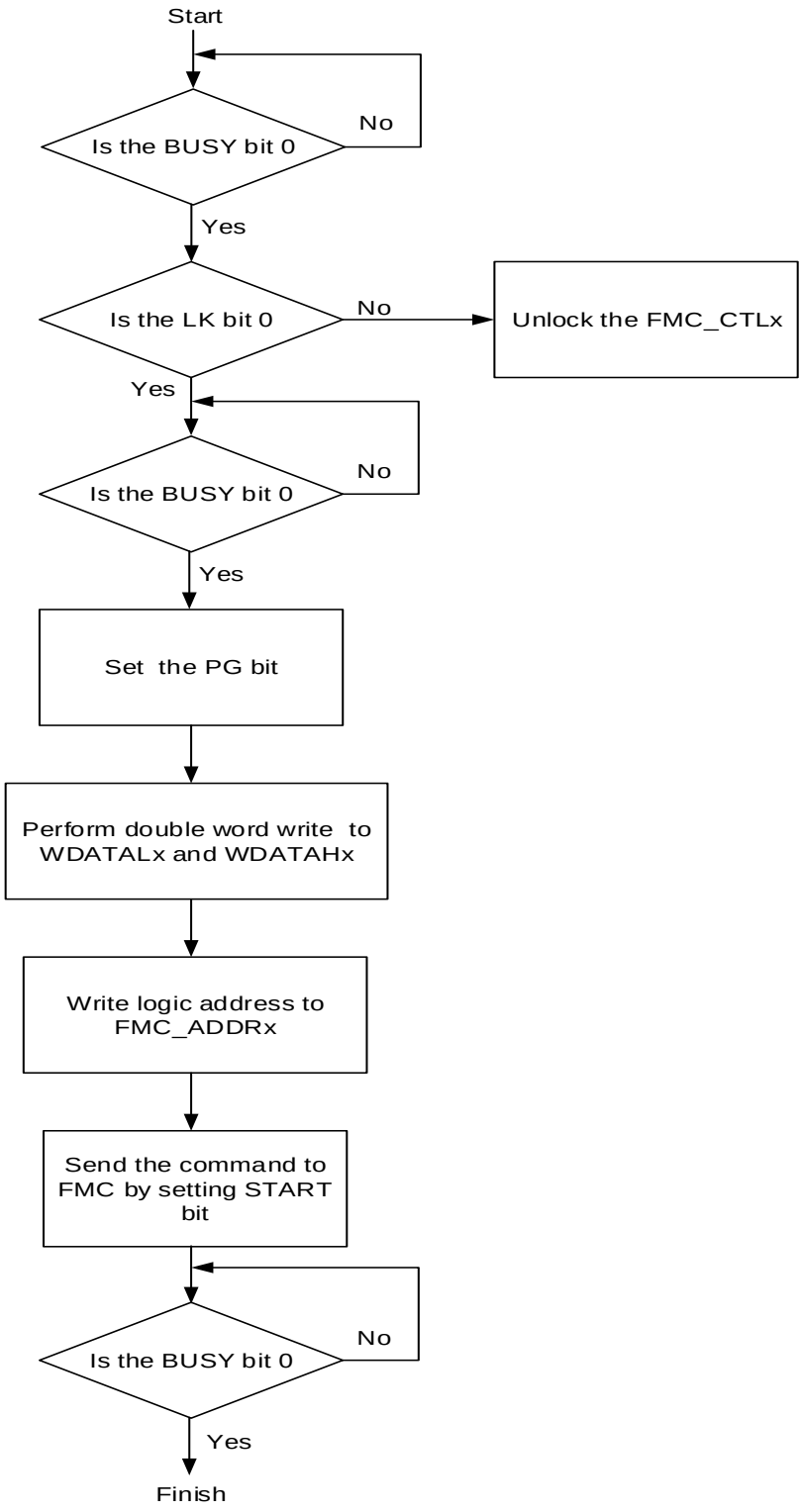
- Check the BUSY bit in FMC_STATx register to confirm that no flash memory operation is in progress (BUSY equal to 0). Otherwise, wait until the operation has been finished.
- Unlock the FMC_CTLx register.
- Double check the BUSY bit in FMC_STATx register to confirm that no flash memory operation is in progress (BUSY equal to 0). Otherwise, wait until the operation has been finished.
- Write the word program command into the PG bit in FMC_CTLx register.
- Write the data to WDATALx and WDATAHx register.
- Write the logic address into the FMC_ADDRx register, must double-word alignment.
- Send the double-words programming command to the FMC by setting the START bit in FMC_CTLx register.
- Wait until all the operations have been completed by checking the value of the BUSY bit in FMC_STATx register.
- Read and verify the flash memory if required using a read access.

When the operation is executed successfully, an interrupt will be triggered by FMC if the ENDIE bit in the FMC_CTLx register is set, and the ENDF in FMC_STATx register is set. Note that before the double word programming operation you should check the address that it has been erased. If the address has not been erased, PGERR bit will set when programming the address even if programming 0x0. Additionally, the program operation will be ignored on protected pages. Flash operation error interrupt will be triggered by the FMC if the ERRIE bit in the FMC_CTLx register is set. The software can check the PGERR bit in the FMC_STATx register to detect this condition in the interrupt handler.

In these conditions, a flash operation error interrupt will be triggered by the FMC if the ERRIE bit in the FMC_CTLx register is set. The software can check the PGERR bit or WPERR bit in the FMC_STATx register to detect which condition occurred in the interrupt handler. The ***Figure 2-5. Process of word program operation*** displays the word programming operation flow.

Note: Before performing program operations using FMC_CTLx (x = 0,1), ensure that the PMU clock is enabled and the VALVRNF flag in the PMU_LVCS register is cleared to zero. Otherwise, the Flash will enter a prohibited state for program operations.

Figure 2-5. Process of word program operation



Note: Reading the flash should be avoided when a program / erase operation is ongoing in the same block.

When programming double word, the ECC byte calculated from the 64 bits will be added following the 64 bits, then the total 72 / 76 bits will be programmed at once, even if the double

word is 0xFFFF FFFF FFFF FFFF.

If the program / erase operation is interrupted by a power down, reset, etc., the contents in flash will not be guaranteed and leave in an indeterminate state. So appropriate measures should be taken to avoid data loss by interrupt of program / erase.

2.3.10. Flash Fast Programming

The FMC provides a fast programming function by fast program RAM which is used to modify the flash memory (Bank0 / Bank1/ HSM bank0 / HSM bank1 / HOST data flash / HSM data flash) contents with write interface0 and interface 1. A row (64 double-word) could be programmed to flash memory in this mode to reduce the page programming time by eliminating the need for verifying the flash locations before they are programmed and to avoid rising and falling time of high voltage for each double-words.

After unlock the FMC_CTLx register, the FMC_ADDRx, FMC_CTLx, WDATALx and WDATAHx must can only be written by same master.

The following steps show the register access sequence of the programming operation.

- Check the row (64 double-word) in flash to confirm all data in flash is all 0xFF. The check blank command can be used to check the page the row in.
- Unlock the FMC_CTLx register.
- Check the BUSY bit in FMC_STATx register to confirm that no flash memory operation is in progress (BUSY equals to 0). Otherwise, wait until the operation has finished.
- Set the FSTPG bit in FMC_CTLx register.
- Write the one row data (64 double-word) to be programed to WDATALx and WDATAHx. The 64 double-word should be written sequentially and continuously.
- Write the logic address into the FMC_ADDRx register, must 64 double-word alignment.
- Send the fast programming command to the FMC by setting the START bit in FMC_CTLx register.
- Wait until all the operations have been finished by checking the value of the BUSY bit in FMC_STATx register.
- Read and verify the Flash memory if required using a AHB access. It is recommended to flush the cache if cache is enabled.

When the operation is executed successfully, the ENDF in FMC_STATx register is set, and an interrupt will be generated if the ENDIE bit in the FMC_CTLx register is set.

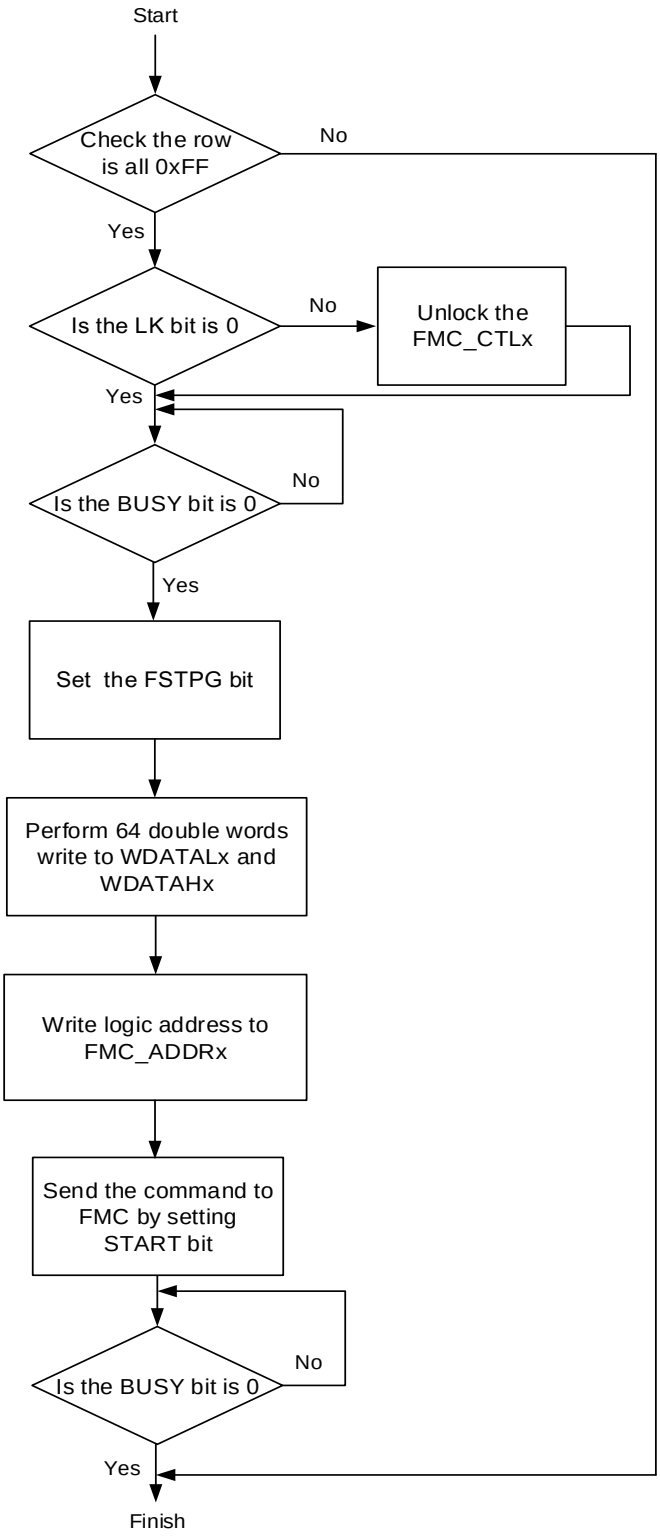
The program operation will be ignored on erase / program protected pages and WPERR bit in FMC_STATx will be set.

In these conditions, a flash operation error interrupt will be generated if the ERRIE bit in the FMC_CTLx register is set. The software can check the PGSERR / WPERR / PGERR bit in the FMC_STATx register to detect which condition occurred in the interrupt handler. ***Figure 2-6. Process of fast programming operation*** shows the fast programming operation flow.

Note: Before performing fast rogramming operations using FMC_CTLx (x = 0,1), ensure that

the PMU clock is enabled and the VALVRNF flag in the PMU_LVCS register is cleared to zero. Otherwise, the Flash will enter a prohibited state for fast programming operations.

Figure 2-6. Process of fast programming operation



Note:

1. The 64 double-word must be written successively.

2. The 64 double-word must be aligned.

3. Because the fast program does not check 0xFF in flash macro by hardware, the software must check 0xFF first and don't program one row twice or more between 2 erases. If program one row twice or more between 2 erases, unpredictable result may occur.

4. Between setting FSTPG and START, read operation is allowed, but the read address should not be the address of the row where to be programmed, otherwise the data read out may be old data.

5. Because the interface 0 and interface 1 fast program use the same SRAM, software must care the SRAM conflict.

2.3.11. Check blank command

The check blank command is used to check if the flash area which is specified by FMC_ADDRx and CBCMDLEN bits in FMC_CTLx register are all 0xFF or not. Configure the check blank command by setting the CBCMD bit in FMC_CTLx register and send the check blank command to the FMC by setting the START bit in FMC_CTLx register, the BUSY bit will be set, and the hardware will check if the flash area is all 0xFF or not. If the flash area is all 0xFF, the ENDF will be set, and the BUSY bit will be cleared, or else the CBCMDERR and ENDF will be set. The check blank command only support for Bank0 / Bank1/ HSM bank0 / HSM bank1 / HOST data flash / HSM data flash.

Note: The flash area to be checked must be in one page and should not exceed 2KB boundary.

2.3.12. OTP programming

The OTP programming operation flow is as same as the flash programming. The OTP block can only be programed once and cannot be erased.

Note: It must ensure the OTP programming sequence completely without any unexpected interrupt, such as system reset or power down. If unexpected interrupt occurs, there is very little probability of corrupt the data stored in flash memory.

Note: Before performing program operations using FMC_CTLx (x = 0,1), ensure that the PMU clock is enabled and the VALVRNF flag in the PMU_LVCS register is cleared to zero. Otherwise, the OTP will enter a prohibited state for program operations.

2.3.13. Option bytes erase

The FMC provides an erase function which is used to initialize the option bytes block in flash. The following steps show the erase sequence.

After unlock the FMC_CTLx register, the FMC_ADDRx and FMC_CTLx must can only be written by same master.

- Unlock the FMC_CTLx register.
- Check the BUSY bit in the FMC_STATx register to confirm that no flash memory operation is in progress (BUSY equal to 0). Otherwise, wait until the operation has finished.
- Write the page address into the FMC_ADDRx register. (HOST option bytes / HSM option bytes logic address)
- Write the page erase command into PER bit in FMC_CTLx register.
- Send the page erase command to the FMC by setting the START bit in FMC_CTLx register.
- Wait until all the operations have been finished by checking the value of the BUSY bit in the FMC_STATx register.
- Read and verify the flash memory using a AHB read access if required.

When the operation is executed successful, the ENDF bit in the FMC_STATx register is set, and an interrupt will be generated if the ENDIE bit in the FMC_CTLx register is set.

Note: Before performing erase operations using FMC_CTLx (x = 0,1), ensure that the PMU clock is enabled and the VALVRNF flag in the PMU_LVCS register is cleared to zero. Otherwise, the option bytes will enter a prohibited state for erase operations.

2.3.14. Option bytes programming

The FMC provides a 64-bit double word programming function which is used for modifying the HOST option bytes / HSM option bytes contents. The following steps show the programming operation sequence.

After unlock the FMC_CTLx register, the FMC_ADDRx, FMC_CTLx, WDATALx and WDATAHx must can only be written by same master.

- Check the BUSY bit in FMC_STATx register to confirm that no flash memory operation is in progress (BUSY equal to 0). Otherwise, wait until the operation has been finished.
- Unlock the FMC_CTLx register.
- Double check the BUSY bit in FMC_STATx register to confirm that no flash memory operation is in progress (BUSY equal to 0). Otherwise, wait until the operation has been finished.
- Write the word program command into the PG bit in FMC_CTLx register.
- Write the data to WDATALx and WDATAHx.
- Write the option bytes logic address into the FMC_ADDRx register, must double-word alignment.
- Send the option bytes programming command to the FMC by setting the START bit in FMC_CTLx register.
- Wait until all the operations have been completed by checking the value of the BUSY bit in FMC_STATx register.
- Read and verify the flash memory if required using a AHB access.

When the operation is executed successfully, an interrupt will be generated if the ENDIE bit in the FMC_CTLx register is set, and the ENDF in FMC_STATx register is set. Note that to

check the address that it has been erased before the word programming operation. If the address has not been erased, PGERR bit in the FMC_STATx register will be set when programming the address even if programming 0x0. The end of this operation is indicated by the ENDF bit in the FMC_STATx register.

Note: Before performing program operations using FMC_CTLx (x = 0,1), ensure that the PMU clock is enabled and the VALVRNF flag in the PMU_LVCS register is cleared to zero. Otherwise, the option bytes will enter a prohibited state for program operations.

Note:

1. The modified option bytes effect only after a system reset or setting the OBRLD bit in FMC_CTL0 register.

2. The option bytes complement values are set by hardware.

2.3.15. Option bytes description

HOST option bytes description

The HOST option bytes block of flash memory reloaded to FMC_OBSTAT, FMC_WPx and FMC_USER1_x registers after each system reset or OBRLD bit set in FMC_CTL0 register, and the HOST option bytes work. The option complement bytes are the opposite of HOST option bytes. When HOST option bytes reload, if the option complement bytes and HOST option bytes does not match, the OBERR bit in FMC_OBSTAT register is set, and the option byte is set to 0xFF. The *Table 2-4. HOST option bytes of* are the detail of option bytes.

Table 2-4. HOST option bytes of GD32A711 / A712 devices

Address	Name	Facotry default	Description
0x1fff 8800	OB_SPC	0xA5	HOST option bytes security protection value 0xA5 : no security protection any value except 0xA5 or 0xCC : protection level low 0xCC : protection level high
0x1fff 8801	OB_SPC_N	0x5A	OB_SPC complement value
0x1fff 8802	OB_USER	0xFF	[7]: reserved [6]: nFWDG_STDBY 0: free watchdog0 stops counting in standby mode 1: free watchdog0 continues counting in standby mode [5:3]: reserved [2]: nRST_STDBY 0: generate a reset instead of entering standby mode 1: no reset when entering standby mode [1]: reserved

Address	Name	Facotry default	Description
			[0]: nFWDG0_HW 0: hardware free watchdog 0 1: software free watchdog 0
0x1fff 8803	OB_USER_N	0x00	OB_USER complement value
0x1fff 8804	OB_DATA[7:0]	0xFF	user defined data bit 7 to 0
0x1fff 8805	OB_DATA_N[7:0]	0x00	OB_DATA complement value bit 7 to 0
0x1fff 8806	OB_DATA[15:8]	0xFF	user defined data bit 15 to 8
0x1fff 8807	OB_DATA_N[15:8]	0x00	OB_DATA complement value bit 15 to 8
0x1fff 8808	OB_BK0WP[7:0]	0xFF	Page erase / program protection of bank0 bit 7 to 0 0: protection active 1: unprotected
0x1fff 8809	OB_BK0WP_N[7:0]	0x00	OB_BK0WP complement value bit 7 to 0
0x1fff 880a	OB_BK0WP[15:8]	0xFF	Page erase / program protection of bank0 bit 15 to 8 0: protection active 1: unprotected
0x1fff 880b	OB_BK0WP_N[15:8]	0x00	OB_BK0WP complement value bit 15 to 8
0x1fff 880c	OB_BK0WP[23:16]	0xFF	Page erase / program protection of bank0 bit 23 to 16 0: protection active 1: unprotected
0x1fff 880d	OB_BK0WP_N[23:16]	0x00	OB_BK0WP complement value bit 23 to 16
0x1fff 880e	OB_BK0WP[31:24]	0xFF	Page erase / program protection of bank0 bit 31 to 24 0: protection active 1: unprotected
0x1fff 880f	OB_BK0WP_N[31:24]	0x00	OB_BK0WP complement value bit 31 to 24
0x1fff 8810	OB_BK1WP[7:0]	0xFF	Page erase / program protection of bank1 bit 7 to 0 0: protection active 1: unprotected
0x1fff 8811	OB_BK1WP_N[7:0]	0x00	OB_BK1WP complement value bit 7 to 0
0x1fff 8812	OB_BK1WP[15:8]	0xFF	Page erase / program protection of bank1 bit 15 to 8 0: protection active 1: unprotected
0x1fff 8813	OB_BK1WP_N[15	0x00	OB_BK1WP complement value bit 15 to 8

Address	Name	Facotry default	Description
	:8]		
0x1fff 8814	OB_BK1WP[23:16]	0xFF	Page erase / program protection of bank1 bit 23 to 16 0: protection active 1: unprotected
0x1fff 8815	OB_BK1WP_N[23:16]	0x00	OB_BK1WP complement value bit 23 to 16
0x1fff 8816	OB_BK1WP[31:24]	0xFF	Page erase / program protection of bank1 bit 31 to 24 0: protection active 1: unprotected
0x1fff 8817	OB_BK1WP_N[31:24]	0x00	OB_BK1WP complement value bit 31 to 24
0x1fff 8818	OB_USER1[7:0]	0xFF	OB_USER1[7:0] value bit 7 to 0
0x1fff 8819	OB_USER1_N[7:0]	0x00	OB_USER1 complement value bit 7 to 0
0x1fff 881a	OB_USER1[15:8]	0xFF	OB_USER1 value bit 15 to 8
0x1fff 881b	OB_USER1_N[15:8]	0x00	OB_USER1 complement value bit 15 to 8
0x1fff 881c	OB_USER1[23:16]	0xFF	OB_USER1 value bit 23 to 16
0x1fff 881d	OB_USER1_N[23:16]	0x00	OB_USER1 complement value bit 23 to 16
0x1fff 881e	OB_USER1[31:24]	0xFF	OB_USER1 value bit 31 to 24
0x1fff 881f	OB_USER1_N[31:24]	0x00	OB_USER1 complement value bit 31 to 24
0x1fff 8820	OB_USER1[39:32]	0xFF	OB_USER1 value bit 39 to 32
0x1fff 8821	OB_USER1_N[39:32]	0x00	OB_USER1 complement value bit 39 to 32
0x1fff 8822	OB_USER1[47:40]	0xFF	OB_USER1 value bit 47 to 40
0x1fff 8823	OB_USER1_N[47:40]	0x00	OB_USER1 complement value bit 47 to 40
0x1fff 8824	OB_USER1[55:48]	0xFF	OB_USER1 value bit 55 to 48
0x1fff 8825	OB_USER1_N[55:48]	0x00	OB_USER1 complement value bit 55 to 48
0x1fff 8826	OB_USER1[63:56]	0xFF	OB_USER1 value bit 63 to 56
0x1fff 8827	OB_USER1_N[63:	0x00	OB_USER1 complement value bit 63 to 56

Address	Name	Facotry default	Description
	56]		
0x1fff 8828	OB_DFWP[7:0]	0xFF	Page erase / program protection of Data flash bit 7 to 0 0: protection active 1: unprotected
0x1fff 8829	OB_DFWP_N[7:0]	0x00	OB_DFWP complement value bit 7 to 0
0x1fff 882A	OB_CLTCFG[7:0]	0xFD	Bit field description, refer to ***Table 2-5. OB_CLTCFG option bytes fields description***
0x1fff 882B	OB_CLTCFG_N[7:0]	0x02	OB_CLTCFG complement value bit 7 to 0
0x1fff 882C	OB_CLTCFG[15:8]	0x00	Bit field description, refer to ***Table 2-5. OB_CLTCFG option bytes fields description***
0x1fff 882D	OB_CLTCFG_N[15:8]	0xFF	OB_CLTCFG complement value bit 15 to 8
0x1fff 882E	OB_CLTCFG[23:16]	0x00	Bit field description, refer to ***Table 2-5. OB_CLTCFG option bytes fields description***
0x1fff 882F	OB_CLTCFG_N[23:16]	0xFF	OB_CLTCFG complement value bit 23 to 16
0x1fff 8830	OB_CLTCFG[31:24]	0x00	Bit field description, refer to ***Table 2-5. OB_CLTCFG option bytes fields description***
0x1fff 8831	OB_CLTCFG_N[31:24]	0xFF	OB_CLTCFG complement value bit 31 to 24
0x1fff 8832	OB_CLTCFG[39:32]	0x00	Bit field description, refer to ***Table 2-5. OB_CLTCFG option bytes fields description***
0x1fff 8833	OB_CLTCFG_N[39:32]	0xFF	OB_CLTCFG complement value bit 39 to 32
0x1fff 8834	OB_CLTCFG[47:40]	0x00	Bit field description, refer to ***Table 2-5. OB_CLTCFG option bytes fields description***
0x1fff 8835	OB_CLTCFG_N[47:40]	0xFF	OB_CLTCFG complement value bit 47 to 40

Table 2-5. OB_CLTCFG option bytes fields description

Fields	Name	Description
OB_CLTCFG[47:39]	Reserved	-
OB_CLTCFG[38]	DBG_SYS_RST_UEN	Debug system reset upgrade enable 0：DBG_SYS_RST system reset upgrade disable 1：DBG_SYS_RST system reset upgrade enable
OB_CLTCFG[37]	SW_SYS_RST_UEN	Software system reset upgrade enable 0：SW_SYS_RST system reset upgrade disable 1：SW_SYS_RST system reset upgrade enable
OB_CLTCFG[36:27]	Reserved	-
OB_CLTCFG[26]	HSM_SYS_RST_UEN	HSM system reset detection upgrade enable

Fields	Name	Description
		0：HSM_SYS_RST system reset upgrade disable 1：HSM_SYS_RST system reset upgrade enable
OB_CLTCFG[25:18]	Reserved	-
OB_CLTCFG[17]	LOPRST_UEN	Loss of PLL error upgrade enable 0：LOPRST system reset upgrade disable 1：LOPRST system reset upgrade enable
OB_CLTCFG[16]	LOHRST_UEN	Loss of HXTAL error upgrade enable 0：LOHRST system reset upgrade disable 1：LOHRST system reset upgrade enable
OB_CLTCFG[15]	Reserved	-
OB_CLTCFG[14]	RCTL_SUBSRU_UEN	Subsystem reset upgrade enable 0：RCTL_SUBSRU upgrade disable 1：RCTL_SUBSRU system reset upgrade enable
OB_CLTCFG[13]	Reserved	-
OB_CLTCFG[12]	STCM_REC_FAIL_UEN	STCM recover failed upgrade enable 0：STCM_REC_FAIL system reset upgrade disable 1：STCM_REC_FAIL system reset upgrade enable
OB_CLTCFG[11]	FMU_REA_FAIL_UEN	FMU react failed upgrade enable 0：FMU_REA_FAIL system reset upgrade disable 1：FMU_REA_FAIL system reset upgrade enable
OB_CLTCFG[10:8]	Reserved	-
OB_CLTCFG[7]	PWDGEN	POR watchdog enable 0: POR watchdog disabled. 1: POR watchdog enabled.
OB_CLTCFG[6:2]	Reserved	-
OB_CLTCFG[1]	FFIOMOD	configuration of the fatal fault pins mode 0: FFIO[0] and FFIO[1] acts as general GPIO pins. 1: FFIO[0] and FFIO[1] acts as dedicated fatal fault reaction pins.
OB_CLTCFG[0]	RSTPIN_MOD	Reset pin mode 0: Reset pin acts as general GPIO pin multiplexed with non-reset function. 1: Reset pin acts as dedicated.

HSM option bytes description

The HSM option bytes block of flash memory reloaded to FMC_HSMOBSTAT register after each system reset or OBRLD bit set in FMC_CTL0 register, and the HSM option bytes work. The option complement bytes are the opposite of HSM option bytes. When HSM option bytes reload, if the option complement bytes and HSM option bytes does not match, the OBERR bit in FMC_HSMOBSTAT register is set, and the option byte is set to 0xFF. The ***Table 2-6. HSM Option bytes*** is the detail of HSM option bytes. Note that HSM bank0 and HSM bank1

use the same HSM option bytes.

Table 2-6. HSM Option bytes

Address	Name	Facotry default	Description
0x1fff f880	OB_HSMUSER[7:0]	0xFF	[5:0]: HSM USER [6]: HSM_nWWDG_HW 0: hardware HSM window watch dog 1: software HSM window watch dog [7]: BSWAP
0x1fff f881	OB_HSMUSER_N[7:0]	0x00	OB_HSMUSER complement value bit 7 to 0
0x1fff f882	OB_HSMSPC[7:0]	0xA5	HSM security protect code
0x1fff f883	OB_HSMSPC_N[7:0]	0x5A	OB_HSMSPC complement value bit 7 to 0
0x1fff f884	OB_HSMPFWP[7:0]	0xFF	Page erase / program protection of HSM Pflash bit 7 to 0 0: protection active 1: unprotected
0x1fff f885	OB_HSMPFWP_N[7:0]	0x00	OB_HSMPWP complement value bit 7 to 0
0x1fff f886	OB_HSMDFWP[7:0]	0xFF	Page erase / program protection of HSM Dflash bit 7 to 0 0: protection active 1: unprotected
0x1fff f887	OB_HSMDFWP_N[7:0]	0x00	OB_HSMDWP complement value bit 7 to 0

Note: OB_HSMUSER[4] and OB_HSMUSER[5] can be used for HSM SBAF PLL enable configuration and parallel / serial boot configuration respectively, refer to "GD32A711_A712_HSM User Manual" for details.

2.3.16. Page erase / program protection

The FMC provides page erase / program protection functions to prevent inadvertent operations on the flash memory. The page erase or program will not be accepted by the FMC on protected pages. If the page erase or program command is sent to the FMC on a protected page, the WPERR bit in the FMC_STATx register will then be set by the FMC. If the WPERR bit is set and the ERRIE bit is set, then the flash operation error interrupt will be generated to get the attention of the CPU.

Page erase / program protection of Bank0

The page erase / program protection of bank 0 can be individually enabled by configuring the OB_BK0WP[31:0] bit field to 0 in the HOST option bytes. If a page erase operation is executed on the Option bytes region, all the flash memory page erase / program protection functions will be disabled. When setting or resetting OB_BK0WP[31:0] in the HOST option bytes, the software need to set OBRLD in FMC_CTL0 register or a system reset to reload the OB_BK0WP[31:0] bits. The *Table 2-7. OB_BK0WP bit for pages protected* shows which

pages are protected by setting OB_BK0WP[31:0].

Table 2-7. OB_BK0WP bit for pages protected

OB_BK0WP bit	pages protected
OB_BK0WP[0]	BANK0_SIZE / 32
OB_BK0WP[1]	BANK0_SIZE / 32
OB_BK0WP[2]	BANK0_SIZE / 32
. . .	. . .
OB_BK0WP[30]	BANK0_SIZE / 32
OB_BK0WP[31]	BANK0_SIZE / 32

Note: BANK0_SIZE is the memory size of Bank0.

Page erase / program protection of Bank1

The page erase / program protection of bank 1 can be individually enabled by configuring the OB_BK1WP[31:0] bit field to 0 in the HOST option bytes. If a page erase operation is executed on the Option bytes region, all the flash memory page erase / program protection functions will be disabled. When setting or resetting OB_BK1WP[31:0] in the HOST option bytes, the software need to set OBRLD in FMC_CTL0 register or a system reset to reload the OB_BK1WP[31:0] bits. The *Table 2-8. OB_BK1WP bit for pages protected* shows which pages are protected by setting OB_BK1WP[31:0].

Table 2-8. OB_BK1WP bit for pages protected

OB_BK1WP bit	pages protected
OB_BK1WP[0]	BANK1_SIZE / 32
OB_BK1WP[1]	BANK1_SIZE / 32
. . .	. . .
OB_BK1WP[30]	BANK1_SIZE / 32
OB_BK1WP[31]	BANK1_SIZE / 32

Note:

1. BANK1_SIZE is the memory size of Bank1.

2. OTP write protection is controlled by OB_BK1WP[31].

Page erase / program protection of HOST data flash

The page erase / program protection of HOST data flash can be individually enabled by configuring the OB_DFWP[7:0] bit field to 0 in the HOST option bytes. Each bit of OB_DFWP[7:0] represents one eighth of HOST data flash. If a page erase operation is executed on the Option bytes region, all the flash memory page erase / program protection functions will be disabled. When setting or resetting OB_DFWP[7:0] in the HOST option bytes,

the software need to set OBRLD in FMC_CTL0 register or a system reset to reload the OB_DFWP[7:0] bits. The *Table 2-9. OB_DFWP bit for pages protected* shows which pages are protected by setting OB_DFWP[7:0].

Table 2-9. OB_DFWP bit for pages protected

OB_DFWP bit	pages protected
OB_DFWP[0]	DFLASH_SIZE / 8
OB_DFWP[1]	DFLASH_SIZE / 8
.	.
.	.
.	.
OB_DFWP[6]	DFLASH_SIZE / 8
OB_DFWP[7]	DFLASH_SIZE / 8

Note: DFLASH_SIZE is the memory size of HOST data flash.

Page erase / program protection of HSM Pflash

The page erase / program protection of HSM Pflash (HSM bank0 / HSM bank1) can be individually enabled by configuring the OB_HSMPFWP[7:0] bit field to 0 in the HSM option bytes. If a page erase operation is executed on the Option bytes region, all the flash memory page erase / program protection functions will be disabled. When setting or resetting OB_HSMPFWP[7:0] in the HSM option bytes, the software need to set OBRLD in FMC_CTL0 register or a system reset to reload the OB_HSMPFWP[7:0] bits. The *Table 2-10. OB_HSMPFWP bit for protected* shows which pages are protected by setting OB_HSMPFWP[7:0].

Table 2-10. OB_HSMPFWP bit for protected

OB_HSMPFWP bit	pages protected
OB_HSMPFWP[0]	HSMPFLASH_SIZE / 8
OB_HSMPFWP[1]	HSMPFLASH_SIZE / 8
.	.
.	.
.	.
OB_HSMPFWP[6]	HSMPFLASH_SIZE / 8
OB_HSMPFWP[7]	HSMPFLASH_SIZE / 8

Note: HSMPFLASH_SIZE is the memory size of HSM Pflash. HSMPFLASH_SIZE is 256KB when HSM bank size is more than 128KB; HSMPFLASH_SIZE is 128KB when HSM bank size is less than or equal 128KB. The OB_HSMPFWP protect HSM bank0 and HSM bank1 simultaneously.

Page erase / program protection of HSM data flash

The page erase / program protection of HSM data flash can be individually enabled by configuring the OB_HSMDFWP[7:0] bit field to 0 in the HSM option bytes. If a page erase operation is executed on the Option bytes region, all the flash memory page erase / program

protection functions will be disabled. When setting or resetting OB_HSMDFWP[7:0] in the HSM option bytes, the software need to set OBRLD in FMC_CTL0 register or a system reset to reload the OB_HSMDFWP[7:0] bits. The *Table 2-11. OB_HSMDFWP bit for protected* shows which pages are protected by setting OB_HSMDFWP[7:0].

Table 2-11. OB_HSMDFWP bit for protected

OB_HSMDFWP bit	pages protected
OB_HSMDFWP[0]	16KB
OB_HSMDFWP[1]	16KB
.	.
.	.
.	.
OB_HSMDFWP[5]	16KB
OB_HSMDFWP[6]	reserved
OB_HSMDFWP[7]	reserved

2.3.17. Security protection

The FMC provides a security protection function to prevent illegal code / data access on the flash memory. This function is useful for protecting the software / firmware from illegal users. HOST system and HSM system have security protection function individually.

There are 3 levels for protection at HOST system:

No protection: when setting OB_SPC byte and its complement value to 0x5AA5, no protection performed after a system reset. The HOST main flash, HOST data flash and HOST option bytes block are accessible by all operations.

Low level protection: when setting OB_SPC byte and its complement value to any value except 0x5AA5 or 0x33CC, low level protection performed. The HOST main flash, OTP, HOST data flash can only be accessed by user code. In debug mode or HSMSPCSET is set, all operations to HOST main flash, OTP, HOST data flash is forbidden. If a read operation is executed to HOST main flash in debug mode or HSMSPCSET is set, a bus error will be generated. If a program / erase / check blank operation is executed to HOST main flash, OTP, HOST data flash in debug mode or HSMSPCSET is set, the WPERR bit in FMC_STATx register will be set. At low level protection, option byte block is accessible by all operations. If program back to no protection level by setting OB_SPC byte and its complement value to 0x5AA5, a mass erase for HOST main flash will be performed.

High level protection: when set OB_SPC byte and its complement value to 0x33CC, high level protection performed. The HOST main flash block is accessible by all operations from user code. The option byte cannot be erased, and the OB_SPC byte and its complement value cannot be reprogrammed. So, if high level protection is programmed, it cannot move back to protection level low or no protection level. The HOST main flash, OTP, HOST data flash can only be accessed by user code. Regarding read protection, the protection strategy for high level protection is the same as that for low level protection.

There are 3 levels for protection at HSM system:

No protection: when setting OB_HSMSPC byte and its complement value to 0x5AA5, no protection performed after a system reset. The HSM Pflash, HSM Dfalsh and HSM option bytes block are accessible by all operations.

Low level protection: when setting OB_HSMSPC byte and its complement value to any value except 0x5AA5 or 0x33CC, low level protection performed. The HSM Pflash, HSM Dflash and HSM OTP can only be accessed by user code. In debug mode or HSMSPCSET is set, all operations to the HSM Pflash, HSM Dflash is forbidden. If a read operation is executed to the HSM Pflash, HSM Dflash and HSM OTP in debug mode or HSMSPCSET is set, a bus error will be generated. If a program / erase / check blank operation is executed to the HSM Pflash, HSM Dfalsh in debug mode or HSMSPCSET is set, the WPERR bit in FMC_STATx register will be set. At low level protection, option bytes block is accessible by all operations. If program back to no protection level by setting OB_HSMSPC byte and its complement value to 0x5AA5 a mass erase for HSM main flash will be performed.

High level protection: when set OB_HSMSPC byte and its complement value to 0x33CC, high level protection performed. The HSM main flash block is accessible by all operations from user code. The option byte cannot be erased, and the OB_HSMSPC byte and its complement value cannot be reprogrammed. So, if high level protection is programmed, it cannot move back to protection level low or no protection level. The HSM Pflash, HSM Dflash and HSM OTP can only be accessed by user code. Regarding read protection, the protection strategy for high level protection is the same as that for low level protection.

2.3.18. Error description

PGSERR bit in FMC_CTLx register will be set if one of the conditions occurs in *Table 2-12. PGSERR conditions*.

Table 2-12. PGSERR conditions

Mode	Condition	Operation
program	CBCMD / FSTPG / MER / PER are not cleared FMC_ADDRx is not writeable address	Set START
fast program	CBCMD / PG / MER / PER are not cleared FMC_ADDRx is not writeable address	Set START
mass erase	CBCMD / FSTPG / PG / PER are not cleared FMC_ADDRx is not writeable address	Set START
page erase	CBCMD / FSTPG / PG / MER are not cleared FMC_ADDRx is not valid address	Set START
check blank	FMC_ADDRx and CBCMDLEN configure error: 1. exceed 2KB boundary	Set START

Mode	Condition	Operation
	2. not valid address	
check blank	FSTPG / MER / PER / PG are not cleared	Set CBCMD
HOST option bytes program / erase	HOST High level protection	Set START
HSM option bytes program / erase	HSM High level protection	Set START

PGERR bit in FMC_CTLx register will be set if one of the conditions occurs in ***Table 2-13. PGERR conditions***.

Table 2-13. PGERR conditions

Mode	Condition	Operation
program	If the program address is not erased	Set START

WPERR bit in FMC_CTLx register will be set if one of the conditions occurs in ***Table 2-14. WPERR conditions***.

Table 2-14. WPERR conditions

Mode	Condition	Operation
program	1.The program address is write protected by option byte 2. read protection low and debug mode	Set START
erase	1.The erase address is write protected by option byte 2. read protection low and debug mode	Set START
check blank	read protection low and debug mode	Set START

2.3.19. Life cycle

The device's Life Cycle (LC) is determined by reading the Life Cycle slots in the Dflash1 flash memory area. The read operation is done during the reset phase with normal timings and it is protected by both operating monitors and ECC check. In addition, a set of integrity checks performed on the LC read data, which guarantees the integrity of the final LC value.

At the end of the reset phase, the LC value is one of the following values:

- GD production (GD_PROD)
- Customer delivery (CUST_DEL)
- OEM production (OEM_PROD)
- In field (IN_FIELD)
- Failure analysis (FA)

The LC value is written into 5 slots (128 bits each slot) at fixed locations in Dflash1 Macro flash memory block which is organized into two fields:

- The valid field
- The invalid field

Each LC slot can have one of the possible 4 status depending on these fields value combination shown in *Table 2-15. Life cycle slots status* and the Life Cycle result is determined as shown in *Table 2-16. Resulting Life cycle*.

Table 2-15. Life cycle slots status

LC slots		LC slot value
Valid field (64 bits)	Invalid field (64 bits)	
Erased	Erased	Erased
Marked	Erased	Active
Marked	Marked	Inactive
Other values		Illegal

Note:

1. "Marked" in this case means that the value has been programmed with the bit pattern 0x55AA 50AF 55AA 50AF.

2. "Erased" means that the bit pattern 0xFFFF FFFF FFFF FFFF.

Table 2-16. Resulting Life cycle

LC Slot 0 GD_PROD	LC Slot 1 CUST_DEL	LC Slot 2 OEM_PROD	LC Slot 3 IN_FIELD	LC Slot 4 FA	Resulting LC
Active	Erased	Erased	Erased	Erased	GD Production
Inactive	**Active**	Erased	Erased	Erased	Customer delivery
Inactive	Inactive	**Active**	Erased	Erased	OEM production
Inactive	Inactive	Inactive	**Active**	Erased	In field
Inactive	Inactive	Inactive	Inactive	**Active**	FA
Other values				Erased	In field
Other values					System reset

Note: "Erased": the slot value is all '1', and it is an area that has not been programmed; programming all F does not work either.

Each slot logic address is shown in *Table 2-17. Logic address of slots*.

Table 2-17. Logic address of slots

Slot num	Logic address	R/W
Slot 0	0x1FFF F900 ~ 0x1FFF F90F	HSM read / HSM OTP write
Slot 1	0x1FFF F910 ~ 0x1FFF F91F	HSM read / HSM OTP write
Slot 2	0x1FFF F920 ~ 0x1FFF F92F	HSM read / HSM OTP write
Slot 3	0x1FFF F930 ~ 0x1FFF F93F	HSM read / HSM OTP write
Slot 4	0x1FFF F940 ~ 0x1FFF F94F	HSM read / HSM OTP write
IN_FIELD_RECOVER_MODE	0x1FFF F950 ~ 0x1FFF F957	HSM read / HSM OTP write

Note:

1. The initial value of 'GD Production' is written in the FT/CP stage

2. HOST / HSM debug are open and backdoor functions required for chip development and

testing are all enabled in LC == 'GD Production' phase.

3. Before chip test and chip delivery, LC should be converted to 'CUST_DEL' at FT/CP stage

4. User should pay attention to the configuration of password of APP DBG, otherwise the default value is processed in LC == 'CUST_DEL' phase.

5. All configurable items that are not available to customers are closed and locked. HOST / HSM debug are trust in LC == 'OEM_PROD' phase.

6. All configurable items that are not available to customers are closed and locked. HOST / HSM debug are trust in LC == ' IN_FIELD' phase.

7. HOST / HSM debug are open in LC = ' FA' phase.

8. Flash protection function (HOST / HSM is in Low level protection / High level protection) is enabled, the flash cannot be read after the debug interface connects in LC = ' FA' phase.

9. Set the flash test mode disable function, which can disable access to chip flash.

10. HSM configuration OTP can be erased and programmed in LC == 'GD_PROD' or 'CUST_DEL'phase. HOST / HSM dbg disable OTP can be written in any phase of LC, but only is valid in the 'OEM_PROD' and 'IN_FIELD' phases.

2.3.20. Application debug configuration

Application debug (APP DBG) logic address is shown in *Table 2-18. Logic address of application debug*.

Table 2-18. Logic address of application debug

Slot num	Logic address	R/W
HOST dbg disable	0x1FFF F958 ~ 0x1FFF F95F	HSM read / HSM OTP write HOST dbg disable: 64'hxxxx xxxx 5A0F A5F0 HOST dbg enable: other values In the LC == 'OEM_PROD' or 'IN_FIELD' or 'FA' stage, the OTP can be written by the HSM when the OTP is erased.
HSM dbg disable	0x1FFF F960 ~ 0x1FFF F967	HSM read / HSM OTP write HSM dbg disable: 64'hxxxx xxxx 5A0F A5F0 HSM dbg enable: other values In the LC == 'OEM_PROD' or 'IN_FIELD' or 'FA' stage, the OTP can be written by the HSM when the OTP is erased.
password	0x1FFF F968 ~ 0x1FFF F977 (HOST password) 0x1FFF F978 ~ 0x1FFF F987 (HSM password)	HSM read / HSM OTP write Password is 128-bits which can only be configured / read when LC == 'GD_PROD' or 'CUST_DEL' stage. In the LC == 'OEM_PROD' or 'IN_FIELD' or 'FA' stage, the OTP cannot be read and written by the HSM and HOST.

Note: To use the above passwords, the password values must not be configured as all zeros or all Fs; otherwise, the keys cannot function properly.

2.3.21. Cryptography key configuration

Cryptography (CRYP) key logic address is shown in *Table 2-19. Logic address of Cryptography key*.

Table 2-19. Logic address of Cryptography key

CRYP key	Logic address	R/W
CRYP KEY1[255:0]	0x1FFF F988 ~ 0x1FFF F9A7	HSM read / HSM OTP write Only the LC == 'GD_PROD' or 'CUST_DEL' stage can be configured and read. In the LC == 'OEM_PROD' or 'IN_FIELD' or 'FA' stage, the OTP cannot be read and written by the HSM and HOST.
CRYP KEY2[255:0]	0x1FFF F9A8 ~ 0x1FFF F9C7	HSM read / HSM OTP write Only the LC == 'GD_PROD' or 'CUST_DEL' stage can be configured and read. In the LC == 'OEM_PROD' or 'IN_FIELD' or 'FA' stage, the OTP cannot be read and written by HSM and HOST.

Note: To use the above keys, their values must not be configured as all zeros or all Fs; otherwise, the keys cannot function properly.

2.3.22. Security protection key configuration

Security protection (SPC) key logic address is shown in *Table 2-20. Logic address of security protection key*.

Table 2-20. Logic address of security protection key

Slot num	Logic addresss	R/W
Password of unlock SPC	0x1FFF F9C8 ~ 0x1FFF F9D7	Password is 128-bits which can only be configured / read when LC = 'GD_PROD' or 'CUST_DEL' stage. In the LC == 'OEM_PROD' or 'IN_FIELD' or 'FA' stage, the OTP cannot be read and written by HSM and HOST.

Note: User can write password of unlock SPC to FMC_SPCKEY register continuously to unlock security protection (including Low level protection / High level protection due to LC = ' FA' phase).

2.3.23. HSM configuration

HSM configuration logic address is shown in *Table 2-21. Logic address of HSM configuration*.

Table 2-21. Logic address of HSM configuration

Slot num	Logic addresss	R/W
HSM SIZE	0x1FFF F9D8 ~ 0x1FFF F9DF	[4:0]: HSMSIZE Only the LC == 'GD_PROD' or 'CUST_DEL' stage can be configured and read. In the LC == 'OEM_PROD' or 'IN_FIELD' or 'FA' stage, the OTP cannot be read and written by HSM and HOST.
HSM SEC	0x1FFF F9E0 ~ 0x1FFF F9E7	[7:0] = AA or [15:8] = AA: HSM SEC enable [23:16] = AA: Factory test mode disable. Factory cannot execute failures analysis. Only the LC == 'GD_PROD' or 'CUST_DEL' stage can be configured and read. In the LC == 'OEM_PROD' or 'IN_FIELD' or 'FA' stage, the OTP cannot be read and written by HSM and HOST. When HSM security set: 1. FMC write interface 0 can be written by HSM system 2. HSM bank0 / HSM bank1 can be read/write by HSM system
Factory mode key	0x1FFF F9E8 ~ 0x1FFF F9F7	128-bits factory mode key. User can write key to this address. GD can use this key to implement chip fault analysis even if the factory test mode is disabled. Once GD successfully executes this key, it will remain active until the next power reset. Only the LC == 'GD_PROD' or 'CUST_DEL' stage can be configured and read. In the LC == 'OEM_PROD' or 'IN_FIELD' or 'FA' stage, the OTP cannot be read and written by HSM and HOST.

When HSM security protection is low level / high level protection or HSM SEC is enabled, HSM resource can only be access by HSM, as follow:

1. Control interface 1 can only be unlock by HSM CPU, other HOST or HSM DMA unlock will generate bus error.
2. HSM bank0 / HSM bank1 can only be written by HSM through control interface 0, other master will generate program sequence error.
3. HSM bank0 / HSM bank1 can only be read by HSM, other HOST read will generate bus error.
4. Dflash1 resource can only be read by HSM , other HOST master read will generate bus error.

When HSM SEC is disabled, HOST master can write Dflash1 resources but can not read Dflash1 resources.

2.4. Register definition

FMC base address: 0x4900 2000

Table 2-22. FMC register overview

Register Name	Symbol	Address Offset	Reset Value
Wait state register	FMC_WS	0x00	0x0000 4003
Unlock SPC protection key register	FMC_SPCKEY	0x4	0x0000 0000
Unlock key register x	FMC_KEYx	0x14	0x0000 0000
Status register x	FMC_STATx	0x18	0x0000 0000
Control register x	FMC_CTLx	0x1C	0x0000 0080
Address register x	FMC_ADDRx	0x20	0x0000 0000
Write data low x register	FMC_WDATALx	0x24	0x0000 0000
Write data high x register	FMC_WDATAHx	0x28	0x0000 0000
Option byte status register	FMC_OBSTAT	0x30	0xXXXX XXXX
Erase / Program protection register 0	FMC_WP0	0x34	0xXXXX XXXX
Erase / Program protection register 1	FMC_WP1	0x38	0xXXXX XXXX
Erase / Program protection register 2	FMC_WP2	0x3C	0xXXXX XXXX
Option byte USER1 register 0	FMC_OBUSER1_0	0x40	0xXXXX XXXX
Option byte USER1 register 1	FMC_OBUSER1_1	0x44	0xXXXX XXXX
HSM option byte status register	FMC_HSMOBSTAT	0x48	0xXXXX XXXX
HSM Erase / Program protection register	FMC_HSMWP	0x4C	0xXXXX XXXX
Blockx ECC control and status register	FMC_BLKxECCCS	0x60	0x0000 0000
Blockx uncorrectable ECC address register	FMC_BLKxUCECCADDR	0x64	0x0000 0000
Blockx correctable ECC address 0 register	FMC_BLKxCECCADDR0	0x68	0x0000 0000
Blockx correctable ECC address 1 register	FMC_BLKxCECCADDR1	0x6C	0x0000 0000
Blockx correctable ECC address 2 register	FMC_BLKxCECCADDR2	0x70	0x0000 0000
Blockx correctable ECC address 3 register	FMC_BLKxCECCADDR3	0x74	0x0000 0000
Bufferx ECC control and status register	FMC_BUFxECCCS	0x80	0x0000 0000
Bufferx uncorrectable ECC address register	FMC_BUFxUCECCADDR	0x84	0x0000 0000
Bufferx correctable ECC address 0 register	FMC_BUFxCECCADDR0	0x88	0x0000 0000
Bufferx correctable ECC address 1 register	FMC_BUFxCECCADDR1	0x8C	0x0000 0000
Bufferx correctable ECC address 2 register	FMC_BUFxCECCADDR2	0x90	0x0000 0000
Bufferx correctable ECC address 3 register	FMC_BUFxCECCADDR3	0x94	0x0000 0000
Fast program buffer ECC control and status register	FMC_FPB_ECCCS	0xA0	0x0000 0000
Fast program buffer uncorrectable ECC address register	FMC_FPB_UCECCADDR	0xA4	0x0000 0000
Fast program buffer correctable ECC address 0 register	FMC_FPB_CECCADDR0	0xA8	0x0000 0000

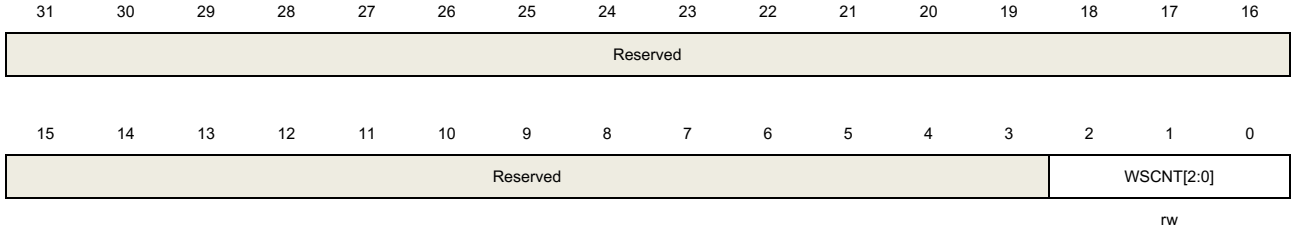
Register Name	Symbol	Address Offset	Reset Value
Fast program buffer correctable ECC Address 1 register	FMC_FPB_CECCADDR1	0xAC	0x0000 0000
Fast program buffer correctable ECC address 2 register	FMC_FPB_CECCADDR2	0xB0	0x0000 0000
Fast program buffer correctable ECC address 3 register	FMC_FPB_CECCADDR3	0xB4	0x0000 0000
Product ID register 0	FMC_PID0	0xD0	0xXXXX XXXX
Product ID register 1	FMC_PID1	0xD4	0xXXXX XXXX

2.4.1. Wait state register (FMC_WS)

Address offset: 0x00

Reset value: 0x0000 0003

This register has to be accessed by word (32-bit).



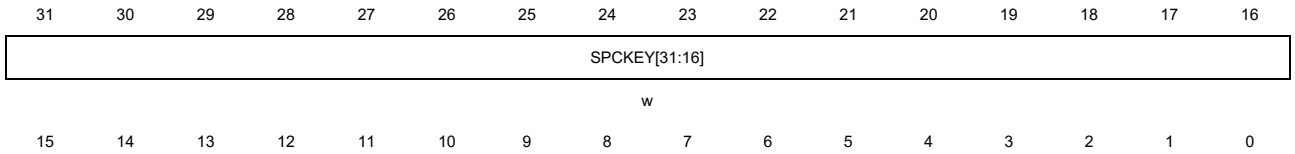
Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2:0	WSCNT[2:0]	Wait state counter These bits are configured by software. 000: 0 wait state added 001: 1 wait state added 010: 2 wait state added 011: 3 wait state added 100: 4 wait state added 101: 5 wait state added 110: Reserved 111: Reserved

2.4.2. Unlock SPC protection key register (FMC_SPCKEY)

Address offset: 0x4

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



SPCKEY[15:0]
w

Bits	Fields	Descriptions
31:0	SPCKEY[31:0]	SPC protection unlock key Need to write ***Password of unlock SPC*** to SPCKEY[31:0] continuously to unlock security protection, otherwise cannot unlock and lock SPCKEY function which only be operation after power reset.

2.4.3. Unlock key register x (FMC_KEYx) (x = 0, 1)

Address offset: 0x14 + 0x100*x
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY[31:16]															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY[15:0]															
w															

Bits	Fields	Descriptions
31:0	KEY[31:0]	FMC_CTLx unlock key These bits can only be written by software. Write KEY[31:0] with keys to unlock FMC_CTLx register.

2.4.4. Status register x (FMC_STATx) (x = 0, 1)

Address offset: 0x18 + 0x100*x
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSTERR	TMRERR	COMPERR	LOADECCERR	Reserved					CBCMDERR	ENDF	WPERR	Reserved	PGERR	PGSERR	BUSY
rc_w1	r	rc_w1	rc_w1						rc_w1	rc_w1	rc_w1		rc_w1	rc_w1	r

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	RSTERR	Reset error during erase / program flag

		If the voltage is below 2.8V or a system reset occurs during flash programming or erasing, an error will be generated and this bit will be set. When the error is occurred, the data in the current address unreliable, and it is necessary to erase and program again. If the voltage is lower than POR, the value of this bit will be reset after the POR reset, but retained after a system reset. **Note:** Programming / erasing is not recommended when the voltage is below 2.8V.
14	TMRERR	TMR error flag bit When TMR error occurs, this bit is set by hardware. If ERRIE bit is enabled, the FMC interrupt will occur.
13	COMPERR	Complement error in writing data When Complement error occurs or the redundancy design of the read controller is faulty , this bit is set by hardware. If ERRIE bit is enabled, the FMC interrupt will occur. The software can clear it by writing 1. **Note:** This bit is not available for FMC_STAT1.
12	LOADECCERR	ECC error when load Find uncorrected ECC error when load HOST OPT / HSM OPT / HSM OTP. The software can clear it by writing 1. **Note:** This bit is not available for FMC_STAT1.
11:7	Reserved	Must be kept at reset value.
6	CBCMDERR	The checked area by the check blank command is all 0xFF or not. 0: The checked area is all 0xFF. 1: The checked area is not all 0xFF. **Note:** The address that write all F and then read, this bit will be set.
5	ENDF	End of operation flag bit When the operation executed successfully, this bit is set by hardware. The software can clear it by writing 1.
4	WPERR	Erase / Program protection error flag bit When erase / program on protected pages, this bit is set by hardware. The software can clear it by writing 1.
3	Reserved	Must be kept at reset value.
2	PGERR	Program error flag bit When program to the flash while it is not 0xFFFF, this bit is set by hardware. The software can clear it by writing 1.
1	PGSERR	Program sequence error flag bit. The software can clear it by writing 1.
0	BUSY	The flash busy flag. When the operation is in progress, this bit is set to 1. When the operation is end or

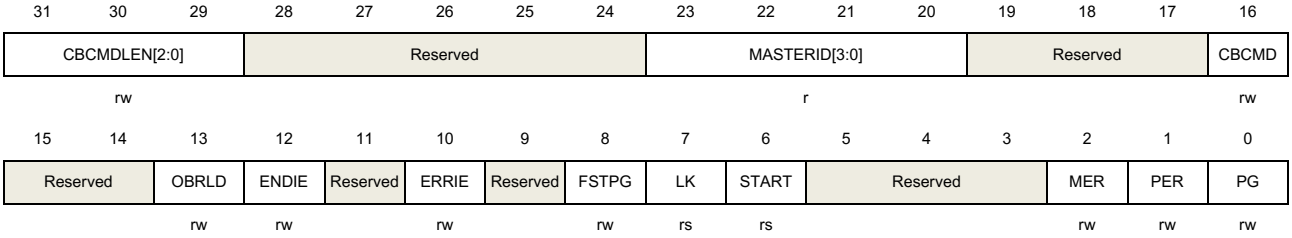
an error is generated, this bit is cleared to 0.

2.4.5. Control register x (FMC_CTLx) (x = 0, 1)

Address offset: 0x1C + 0x100*x

Reset value: 0x0000 0080

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:29	CBCMDLEN[2:0]	CBCMD read length 2^(CBCMDLEN + 1). The read length by check blank command. The read length is 2^ (CBCMDLEN + 1) double words. 000: check blank size for 16 bytes 001: check blank size for 32 bytes 010: check blank size for 64 bytes 011: check blank size for 128 bytes 100: check blank size for 256 bytes 101: check blank size for 512 bytes 110: check blank size for 1024 bytes 111: check blank size for 2048 bytes
28:24	Reserved	Must be kept at reset value.
23:20	MASTERID[3:0]	The Master ID of unlock FMC_CTLx register 0000: AXIM_CM7_0 0001: reserved 0010: DMA0_MEM 0011: DMA1_MEM 0100: HSM_CM23 0101: reserved 0110: reserved 0111: reserved 1000: reserved 1001: reserved 1010: DMA0_PERIPH 1011: DMA1_PERIPH 1100: HSM DMA 1110: AHBP_CM7_0

		Others: reserved
19:17	Reserved	Must be kept at reset value.
16	CBCMD	The command to check the selected area is blank or not.
15:14	Reserved	Must be kept at reset value.
13	OBRLD	Option byte reload bit This bit is set by software. 0: No effect 1: Force option byte reload, and generate a subsystem reset **Note:** This bit is not available for FMC_CTL1.
12	ENDIE	End of operation interrupt enable bit This bit is set or cleared by software 0: No interrupt generated by hardware. 1: End of operation interrupt enable
11	Reserved	Must be kept at reset value.
10	ERRIE	Error interrupt enable bit This bit is set or clear by software 0: No interrupt generated by hardware 1: Error interrupt enable **Note:** This bit supports TMR.
9	Reserved	Must be kept at reset value.
8	FSTPG	Flash fast program command bit This bit is set or clear by software 0: No effect 1: Flash fast program command **Note:** only one FSTPG is valid meanwhile. To use FSTPG0, FSTPG1 must be set to 0
7	LK	FMC_CTLx lock bit This bit is cleared by hardware when right sequence written to the FMC_KEYx register. This bit can be set by software. **Note:** Lock operation is active only when MASTERID match which prevent mistake lock.
6	START	Send command to FMC bit This bit is set by software to send command to FMC. This bit is cleared by hardware when the BUSY bit is cleared. **Note:** This bit supports TMR.
5:3	Reserved	Must be kept at reset value.
2	MER	Flash mass erase command bit This bit is set or cleared by software

		0: No effect 1: Flash mass erase command **Note:** This bit is not available for FMC_CTL1. **Note:** This bit supports TMR.
1	PER	Flash page erase command bit This bit is set or clear by software 0: No effect 1: Flash page erase command **Note:** This bit supports TMR.
0	PG	Flash program command bit This bit is set or clear by software 0: No effect 1: Flash program command **Note:** This bit supports TMR.

Note: This register should be reset after the corresponding flash operation completed.

2.4.6. Address register x (FMC_ADDRx) (x = 0, 1)

Address offset: 0x20 + 0x100*x

Reset value: 0x0000 0000

Note: FMC_ADDR0 support complement register check.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR[15:0]															
rw															

Bits	Fields	Descriptions
31:0	ADDR[31:0]	Flash erase / program command address bits These bits are configured by software. ADDR bits are the address of flash to be erased / programmed.

2.4.7. Write data low x register (FMC_WDATALx) (x = 0, 1)

Address offset: 0x24 + 0x100*x

Reset value: 0x0000 0000

Note: FMC_WDATAL0 support complement register check.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WDATA[31:16]															

rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDATA[15:0]															
rw															

Bits	Fields	Descriptions
31:0	WDATA[31:0]	Flash program command write data bits These bits are configured by software. WDATA bits are the date of flash to be programmed. WDATAL0 register has 64 level depth to support fast program

2.4.8. Write data high x register (FMC_WDATAHx) (x = 0, 1)

Address offset: 0x28 + 0x100*x

Reset value: 0x0000 0000

Note: FMC_WDATAH0 support complement register check.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WDATA[63:48]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDATA[47:32]															
rw															

Bits	Fields	Descriptions
31:0	WDATA[63:32]	Flash program command write data bits These bits are configured by software. WDATA bits are the date of flash to be programmed. WDATAH0 register has 64 level depth to support fast program

2.4.9. Option byte status register (FMC_OBSTAT)

Address offset: 0x30

Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA[15:0]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nFWDG_STDBY	Reserved			nRST_STDBY	Reserved	nFWDG_HW	Reserved				SPCS	PLEVEL[1:0]		OBERR
	r	r			r		r					r	r		r

Bits	Fields	Descriptions
31:16	DATA[15:0]	Store OB_DATA[15:0] of option byte block after system reset
15	Reserved	Must be kept at reset value
14	nFWDG_STDBY	Free watchdog count mode configuration bit in standby mode 0: Free watchdog stops counting in standby mode 1: Free watchdog continues counting in standby mode
13:11	Reserved	Must be kept at reset value
10	nRST_STDBY	Reset mode configuration bit when entering standby mode 0: Generate a reset instead of entering standby mode 1: No reset when entering standby mode
9	Reserved	Must be kept at reset value
8	nFWDG_HW	Free watchdog mode configuration bit 0: Hardware free watchdog 1: Software free watchdog
7:4	Reserved	Must be kept at reset value
3	SPCS	HOST security protection state 0: HOST security protection state is invalid 1: HOST security protection state is valid **Note:** This bit indicates security protection whether triggered.
2:1	PLEVEL[1:0]	Security protection level 00: No protection level 01: Protect level low 10: Reserved 11: Protect level high
0	OBERR	Option byte read error bit. This bit is set by hardware when the HOST option byte and its complement byte do not match, and the option byte set 0xFF.

2.4.10. Erase / Program protection register 0 (FMC_WP0)

Address offset: 0x34

Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BK0WP[31:16]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BK0WP[15:0]															

r

Bits	Fields	Descriptions
31:0	BK0WP[31:0]	Store OB_BK0WP[31:0] of HOST option bytes block after system reset.

2.4.11. Erase / Program protection register 1 (FMC_WP1)

Address offset: 0x38

Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).



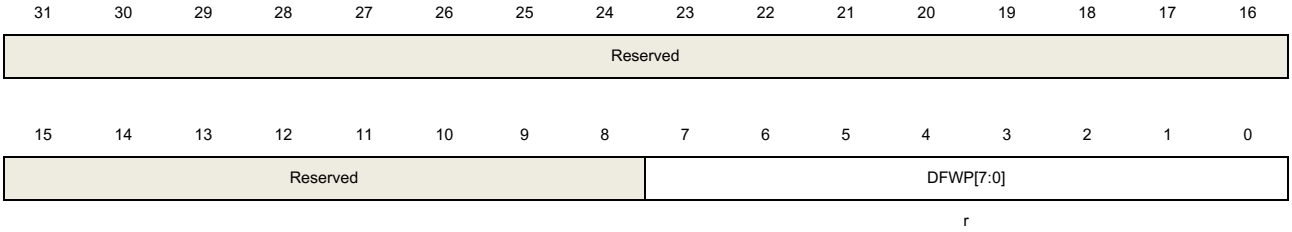
Bits	Fields	Descriptions
31:0	BK1WP[31:0]	Store OB_BK1WP[31:0] of HOST option bytes block after system reset.

2.4.12. Erase / Program protection register 2 (FMC_WP2)

Address offset: 0x3C

Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	DFWP[7:0]	Store OB_DFWP[7:0] of HOST option bytes block after system reset.

2.4.13. Option byte USER1 register 0 (FMC_OBUSER1_0)

Address offset: 0x40

Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
USER1[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USER1[15:0]															
r															

Bits	Fields	Descriptions
31:0	USER1[31:0]	Store OB_USER1[31:0] of HOST option bytes block after system reset.

2.4.14. Option byte USER1 register 1 (FMC_OBUSER1_1)

Address offset: 0x44

Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
USER1[63:48]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USER1[47:32]															
r															

Bits	Fields	Descriptions
31:0	USER1[63:32]	Store OB_USER1[63:32] of HOST option bytes block after system reset.

2.4.15. HSM option byte status register (FMC_HSMOBSTAT)

Address offset: 0x48

Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OB_HSMSPC[7:0]								BSWAP	HSM_nW WDG_HW	OB_HSMUSER[5:0]					
r								r	r	r					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													HSMSPCS	HSMSPCSET	OBERR
													r	rw	r

Bits	Fields	Descriptions
31:24	OB_HSMSPC[7:0]	HSM option byte security protection code 0xA5: No security protection

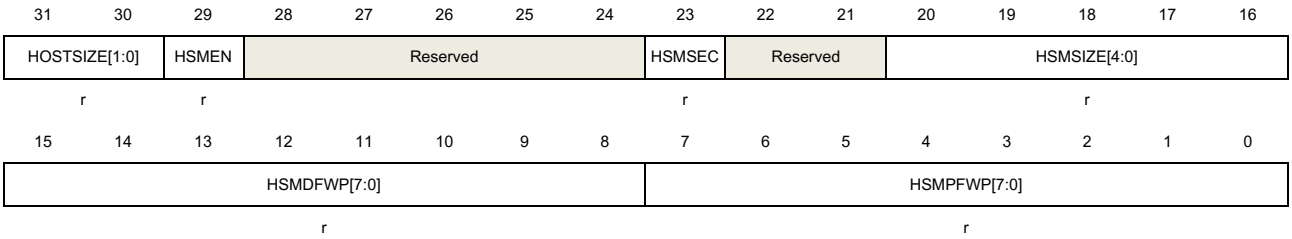
		0xCC: Security protection level high any value except 0xA5 or 0xCC: security protection level low.
23	BSWAP	Bank swap 0: Bank0 and HSM bank 0 at Pflash1, Bank1 and HSM bank 1 at Pflash0 1: Bank0 and HSM bank 0 at Pflash0, Bank1 and HSM bank 1 at Pflash1 **Note:** This bit is updated after system reset.
22	HSM_nWWDG_HW	HSM window watch dog mode configuration bit 0: Hardware HSM window watch dog 1: Software HSM window watch dog
21:16	OB_HSMUSER[5:0]	Store OB_HSMUSER[5:0] of HSM option byte block after system reset
15:3	Reserved	Must be kept at reset value.
2	HSMSPCS	HSM security protection state 0: HSM security protection state is invalid 1: HSM security protection state is valid **Note:** This bit indicates HSM security protection whether triggered.
1	HSMSPCSET	HSM security protection set This bit can be set to trigger security protection when HOST / HSM in Low level protection / High level protection. writing 0 has no effect and can be cleared by power reset.
0	OBERR	HSM option byte read error bit. This bit is set by hardware when the HSM option byte and its complement byte do not match, and the option byte set 0xFF.

2.4.16. HSM Erase / Program protection register (FMC_HSMWP)

Address offset: 0x4C

Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:30	HOSTSIZE[1:0]	HOST Bank0 / Bank1 size 00: reserved 01: 512KB 10: reserved

		11: 1024KB
29	HSMEN	HSM enable 0: No HSM function 1: Has HSM function
28:24	Reserved	Must be kept at reset value.
23	HSMSEC	HSM security 0: The HSM is in insecure state. HOST can read HSM bank0 / bank1. 1: The HSM is in secure state. HOST cannot read HSM bank0 / bank1. Please refer to ***HSM configuration*** for the description.
22:21	Reserved	Must be kept at reset value.
20:16	HSMSIZE[4:0]	HSM bank 0 / bank 1 size 0x00-0x0F: HSM bank 0 / bank 1 size is HSMSIZE*16K 0x10-0x1E: HSM bank 0 / bank 1 size is 256KB 0x1F: HSM bank 0 / bank 1 size is 0 KB
15:8	HSMDFWP[7:0]	Store OB_HSMDFWP [7:0] of HSM option bytes block after system reset.
7:0	HSMPFWP[7:0]	Store OB_HSMPFWP [7:0] of HSM option bytes block after system reset.

2.4.17. Blockx ECC control and status register (FMC_BLKxECCCS) (x = 0,1..3)

Address offset: 0x60 + 0x100*x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECCDET	ECCEDC	ECCCOROV	ECCAERR	Reserved	ECCCOROVIE	ECCDETIE	ECCCORIE	Reserved			ECCORC	ECCCOR3	ECCCOR2	ECCCOR1	ECCCOR0
rc_w1	rc_w1	rc_w1	rc_w1		rw	rw	rw				rc_w1	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits	Fields	Descriptions
31	ECCDET	Two bit errors detect flag This bit set when two bit errors is detected. And cleared by writing 1 to ECCDET bit in FMC_ECCCS register. 0: Two bit ECC errors are not detected 1: Two bit ECC errors are detected
30	ECCEDC	EDC after ECC checker error flag The software can clear it by writing 1. 0: No EDC after ECC checker error

		1: EDC after ECC checker error generated and two bit ECC error is not detected **Note:** This bit is invalid when two bit ECC error occurs. This bit is only for block 0 / block 1.
29	ECCCOROV	More than 4 one bit error detected This bit is cleared by writing 1. 0: No More than 4 one bit error detected 1: More than 4 one bit error detected
28	ECCAERR	One bit ECC error in address stage This bit is cleared by writing 1. 0: No ECC one bit error in address stage 1: ECC one bit error in address stage **Note:** This bit is only for block 0 / block 1.
27	Reserved	Must be kept at reset value.
26	ECCCOROVIE	More than 4 one bit error detect interrupt enable 0: More than 4 one bit errors detect interrupt disable. 1: More than 4 one bit detect interrupt enable.
25	ECCDETIE	Two bit errors detect / EDC error / address ECC error interrupt enable 0: Uncorrectable bit errors detect interrupt disable 1: Uncorrectable bit errors detect interrupt enable **Note:** This bit supports TMR.
24	ECCCORIE	One bit error correct interrupt enable 0: One bit error correct interrupt disable 1: One bit error correct interrupt enable **Note:** This bit supports TMR.
23:21	Reserved	Must be kept at reset value.
20	ECCORC	Write 1 to clear ECCORx (x = 0, 1, 2, 3)
19	ECCCOR3	One bit error detected and corrected flag 3 This bit is cleared by writing 1 to ECCORC. 0: No ECC error is detected and corrected 1: An ECC error is detected and corrected
18	ECCCOR2	One bit error detected and corrected flag 2 This bit is cleared by writing 1 to ECCORC. 0: No ECC error is detected and corrected 1: An ECC error is detected and corrected
17	ECCCOR1	One bit error detected and corrected flag 1 This bit is cleared by writing 1 to ECCORC. 0: No ECC error is detected and corrected 1: An ECC error is detected and corrected

16	ECCCOR0	One bit error detected and corrected flag 0 This bit is cleared by writing 1 to ECCORC. 0: No ECC error is detected and corrected 1: An ECC error is detected and corrected
15:0	Reserved	Must be kept at reset value.

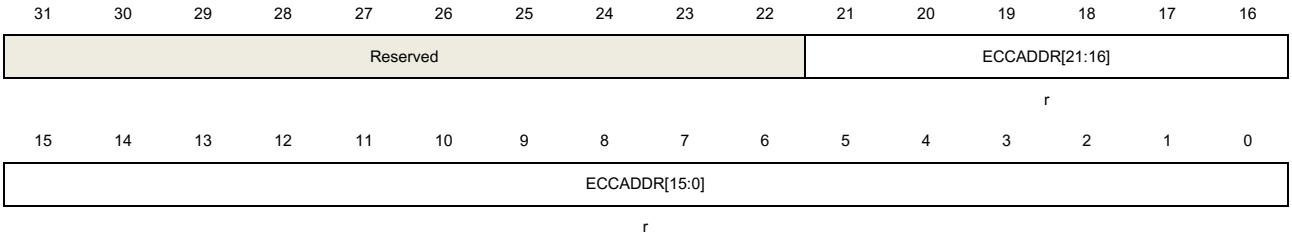
2.4.18. Blockx uncorrectable ECC address register (FMC_BLKxUCECCADDR) (x = 0,1..3)

Address offset: 0x64 + 0x100*x

Reset value: 0x0000 0000

Note: Uncorrectable ECC error include two bit ECC error and EDC error.

This register has to be accessed by word (32-bit).



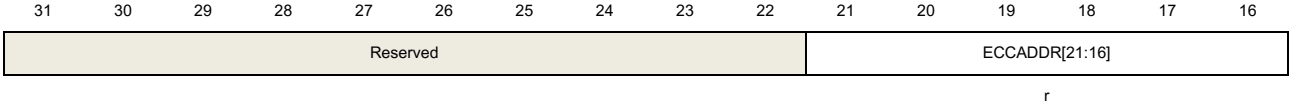
Bits	Fields	Descriptions
31:22	Reserved	Must be kept at reset value.
21:0	ECCADDR[21:0]	The offset address of double word where an uncorrectable ECC error is detected. The address is in ECC error address area. The offset of logical address can be obtained by ECCADDR multiplying 8. **Note:** ECCADDR[21] = 0 : User area ECCADDR[21] = 1 : System area Uncorrectable ECC error include two bit ECC error and EDC error. The next ECCADDR can only be updated when ECCDET and ECCEDC flag is cleared.

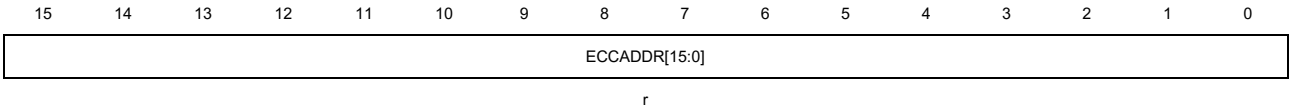
2.4.19. Blockx correctable ECC address 0 register (FMC_BLKxCECCADDR0) (x = 0,1..3)

Address offset: 0x68 + 0x100*x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



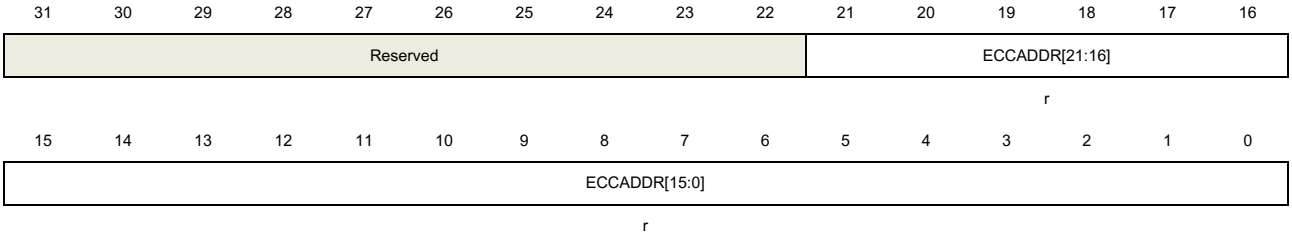


Bits	Fields	Descriptions
31:22	Reserved	Must be kept at reset value.
21:0	ECCADDR[21:0]	The offset address of double word where an correctable ECC error is detected. The address is in ECC error address area. The offset of logical address can be obtained by ECCADDR multiplying 8. **Note:** ECCADDR[21] = 0 : User area ECCADDR[21] = 1 : System area

2.4.20. Blockx correctable ECC address 1 register (FMC_BLKxCECCADDR1) (x = 0,1..3)

Address offset: 0x6C + 0x100*x
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:22	Reserved	Must be kept at reset value.
21:0	ECCADDR[21:0]	The offset address of double word where an correctable ECC error is detected. The address is in ECC error address area. The offset of logical address can be obtained by ECCADDR multiplying 8. **Note:** ECCADDR[21] = 0 : User area ECCADDR[21] = 1 : System area

2.4.21. Blockx correctable ECC address 2 register (FMC_BLKxCECCADDR2) (x = 0,1..3)

Address offset: 0x70 + 0x100*x
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										ECCADDR[21:16]					
										r					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECCADDR[15:0]															
r															

Bits	Fields	Descriptions
31:22	Reserved	Must be kept at reset value.
21:0	ECCADDR[21:0]	The offset address of double word where an correctable ECC error is detected. The address is in ECC error address area. The offset of logical address can be obtained by ECCADDR multiplying 8. **Note:** ECCADDR[21] = 0 : User area ECCADDR[21] = 1 : System area

2.4.22. Blockx correctable ECC address 3 register (FMC_BLKxCECCADDR3) (x = 0,1..3)

Address offset: 0x74 + 0x100*x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										ECCADDR[21:16]					
										r					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECCADDR[15:0]															
r															

Bits	Fields	Descriptions
31:22	Reserved	Must be kept at reset value.
21:0	ECCADDR[21:0]	The offset address of double word where an correctable ECC error is detected. The address is in ECC error address area. The offset of logical address can be obtained by ECCADDR multiplying 8. **Note:** ECCADDR[21] = 0 : User area ECCADDR[21] = 1 : System area

2.4.23. Bufferx ECC control and status register (FMC_BUFxECCCS) (x = 0,2,3)

x = 0, 2, 3

Address offset: 0x80 + 0x100*x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECCDET	ECCEDC	ECCCOROV	ECCAERR	Reserved	ECCCOROVIE	ECCDETIE	ECCCORIE	Reserved			ECCORC	ECCCOR3	ECCCOR2	ECCCOR1	ECCCOR0
rc_w1	rc_w1	rc_w1	rc_w1		rw	rw	rw				rc_w1	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											BUFEN	DPFEN	CPFEN	DBUFEN	CBUFEN
											rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31	ECCDET	Two bit errors detect flag This bit set when two bit errors is detected. And cleared by writing 1 to ECCDET bit in FMC_ECCCS register. 0: Two ECC errors are not detected. 1: Two ECC errors are detected.
30	ECCEDC	EDC after ECC checker error flag The software can clear it by writing 1. 0: No EDC after ECC checker error 1: EDC after ECC checker error generated and two bit ECC error is not detected **Note:** This bit is invalid when two bit ECC error occurs.
29	ECCCOROV	More than 4 one bit error detected This bit is cleared by writing 1. 0: No More than 4 one bit error detected. 1: More than 4 one bit error detected.
28	ECCAERR	One bit ECC error in address stage This bit is cleared by writing 1. 0: No ECC one bit error in address stage 1: ECC one bit error in address stage
27	Reserved	Must be kept at reset value.
26	ECCCOROVIE	More than 4 one bit error detect interrupt enable This bit is set, an interrupt will be generated. 0: More than 4 one bit errors detect interrupt disable. 1: More than 4 one bit detect interrupt enable.
25	ECCDETIE	Two bit errors detect / EDC error / address ECC error interrupt enable This bit is set, an interrupt will be generated. 0: Uncorrectable bit errors detect interrupt disable.

		1: Uncorrectable bit errors detect interrupt enable. **Note:** This bit supports TMR.
24	ECCCORIE	One bit error correct interrupt enable. 0: One bit error correct interrupt disable. 1: One bit error correct interrupt enable.
23:21	Reserved	Must be kept at reset value.
20	ECCORC	Write 1 to clear ECCORx (x = 0,1,2,3)
19	ECCCOR3	One bit error detected and corrected flag 3. This bit is cleared by writing 1 to ECCORC. 0: No ECC error is detected and corrected. 1: An ECC error is detected and corrected.
18	ECCCOR2	One bit error detected and corrected flag 2. This bit is cleared by writing 1 to ECCORC. 0: No ECC error is detected and corrected. 1: An ECC error is detected and corrected.
17	ECCCOR1	One bit error detected and corrected flag 1. This bit is cleared by writing 1 to ECCORC. 0: No ECC error is detected and corrected. 1: An ECC error is detected and corrected.
16	ECCCOR0	One bit error detected and corrected flag 0. This bit is cleared by writing 1 to ECCORC. 0: No ECC error is detected and corrected. 1: An ECC error is detected and corrected.
15:5	Reserved	Must be kept at reset value.
4	BUFEN	Buffers enable 0: Disable 1: Enable
3	DPFEN	Data prefetch enable Enables or disables data prefetching initiated by a read access. If this bit set 1, DBFEN must set 1. 0: Disable 1: Enable
2	CPFEN	Code prefetch enable Enables or disables code prefetching initiated by a read access. If this bit set 1, CBFEN must set 1. 0: Disable 1: Enable
1	DBUFEN	Read data buffers enable

		Enables or disables line read data buffer hits. It is also used to invalidate the buffers. 0: Disable 1: Enable
0	CBUFEN	Read code buffers enable Enables or disables line read code buffer hits. It is also used to invalidate the buffers. 0: Disable 1: Enable

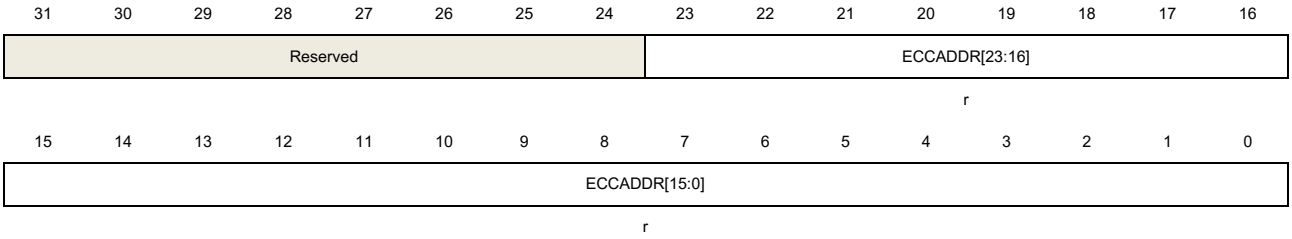
2.4.24. Bufferx uncorrectable ECC address register (FMC_BUFxUCECCADDR) (x = 0,2,3)

x = 0, 2, 3

Address offset: 0x84 + 0x100*x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:0	ECCADDR[23:0]	The offset address of double word where an uncorrectable ECC error is detected. The address is in ECC error address area (0x08xx xxxx).

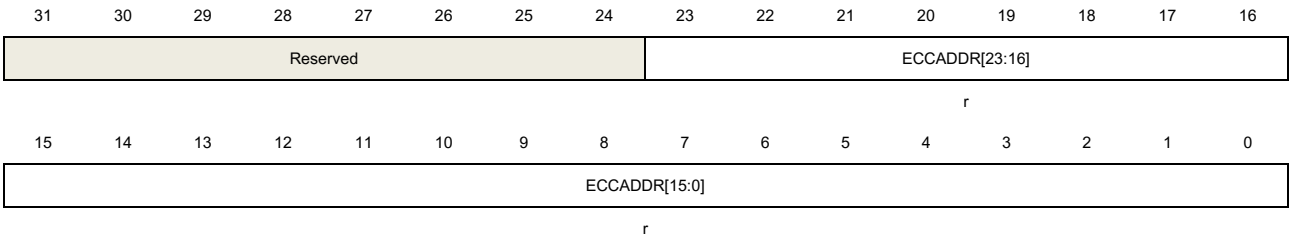
2.4.25. Bufferx correctable ECC address 0 register (FMC_BUFxCECCADDR0) (x = 0,2,3)

x = 0, 2, 3

Address offset: 0x88+0x100*x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

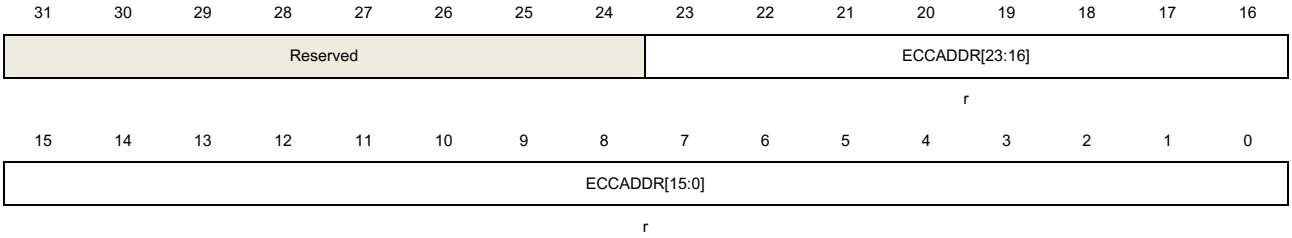


Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:0	ECCADDR[23:0]	The offset address of double word where an correctable ECC error is detected. The address is in ECC error address area (0x08xx xxxx).

2.4.26. Bufferx correctable ECC address 1 register (FMC_BUFxCECCADDR1) (x = 0,2,3)

x = 0, 2, 3 (for GD32A711 / A712 devices)
Address offset: 0x8C+0x100*x
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

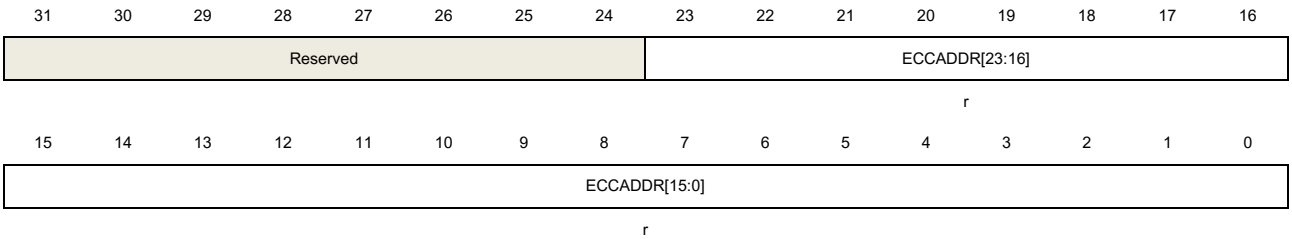


Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:0	ECCADDR[23:0]	The offset address of double word where an correctable ECC error is detected. The address is in ECC error address area (0x08xx xxxx).

2.4.27. Bufferx correctable ECC address 2 register (FMC_BUFxCECCADDR2) (x = 0,2,3)

x = 0, 2, 3 (for GD32A711 / A712 devices)
Address offset: 0x90+0x100*x
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions

31:24	Reserved	Must be kept at reset value.
23:0	ECCADDR[23:0]	The offset address of double word where an correctable ECC error is detected. The address is in ECC error address area (0x08xx xxxx).

2.4.28. Bufferx correctable ECC address 3 register (FMC_BUFxCECCADDR3) (x = 0,2,3)

x = 0, 2, 3 (for GD32A711 / A712 devices)

Address offset: 0x94+0x100*x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								ECCADDR[23:16]							
								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECCADDR[15:0]															
r															

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:0	ECCADDR[23:0]	The offset address of double word where an correctable ECC error is detected. The address is in ECC error address area (0x08xx xxxx).

2.4.29. Fast program buffer ECC control and status register (FMC_FPB_ECCCS)

Address offset: 0xA0

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECCDET	ECCEDC	ECCCOROV	ECCAER	Reserved	ECCCOROVIE	ECCDETIE	ECCCORIE	Reserved			ECCORC	ECCCOR3	ECCCOR2	ECCCOR1	ECCCOR0
rc_w1	rc_w1	rc_w1	rc_w1		rw	rw	rw				rc_w1	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits	Fields	Descriptions
31	ECCDET	Two bit errors detect flag This bit set when two bit errors is detected. And cleared by writing 1 to ECCDET bit in FMC_ECCCS register 0: Two bit ECC errors are not detected.

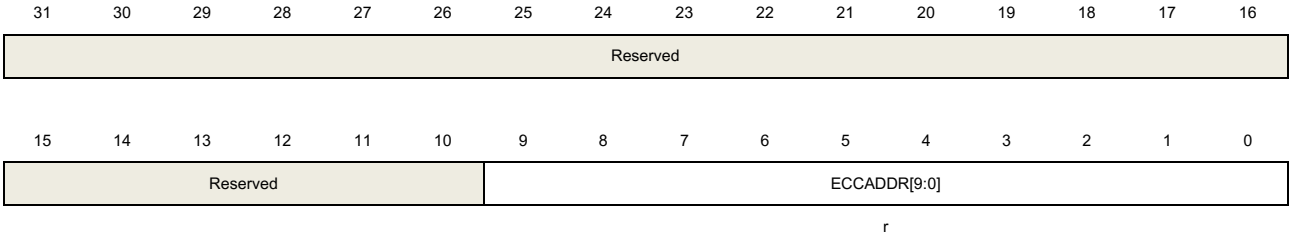
		1: Two bit ECC errors are detected.
30	ECCEDC	EDC after ECC checker error flag The software can clear it by writing 1. 0: No EDC after ECC checker error 1: EDC after ECC checker error generated
29	ECCCOROV	More than 4 one bit error detected This bit is cleared by writing 1. 0: No More than 4 one bit error detected. 1: More than 4 one bit error detected.
28	ECCAERR	one bit ECC error in address stage This bit is cleared by writing 1. 0: No ECC one bit error in address stage 1: ECC one bit error in address stage
27	Reserved	Must be kept at reset value.
26	ECCCOROVIE	More than 4 one bit error detect interrupt enable This bit is set, an interrupt will be generated. 0: More than 4 one bit errors detect interrupt disable. 1: More than 4 one bit detect interrupt enable.
25	ECCDETIE	Two bit errors detect / EDC error / address ECC error interrupt enable This bit is set, an interrupt will be generated. 0: Uncorrectable bit errors detect interrupt disable. 1: Uncorrectable bit errors detect interrupt enable. **Note:** This bit supports TMR.
24	ECCCORIE	One bit error correct interrupt enable. 0: One bit error correct interrupt disable. 1: One bit error correct interrupt enable.
23:21	Reserved	Must be kept at reset value.
20	ECCORC	Write 1 to clear ECCORx (x = 0,1,2,3)
19	ECCCOR3	One bit error detected and corrected flag 3 This bit is cleared by writing 1 to ECCORC. 0: No ECC error is detected and corrected. 1: An ECC error is detected and corrected.
18	ECCCOR2	One bit error detected and corrected flag 2 This bit is cleared by writing 1 to ECCORC. 0: No ECC error is detected and corrected. 1: An ECC error is detected and corrected.
17	ECCCOR1	One bit error detected and corrected flag 1 This bit is cleared by writing 1 to ECCORC.

		0: No ECC error is detected and corrected. 1: An ECC error is detected and corrected.
16	ECCCOR0	One bit error detected and corrected flag 0 This bit is cleared by writing 1 to ECCORC. 0: No ECC error is detected and corrected. 1: An ECC error is detected and corrected.
15:0	Reserved	Must be kept at reset value.

2.4.30. Fast program buffer uncorrectable ECC address register (FMC_FPB_UCECCADDR)

Address offset: 0xA4
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

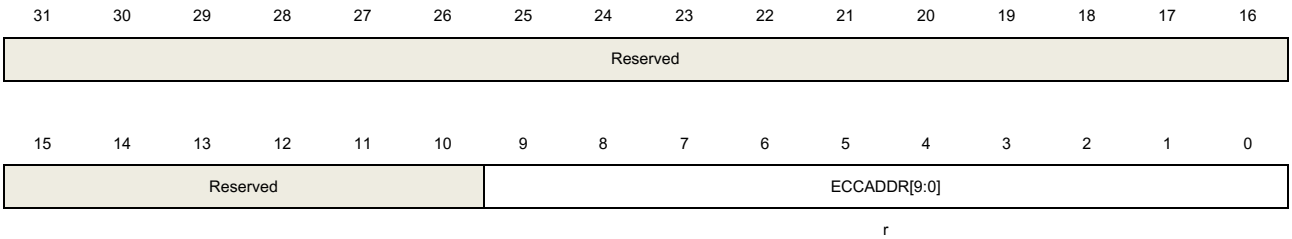


Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:0	ECCADDR[9:0]	The offset address of double word where an uncorrectable ECC error is detected. The address is in ECC error address area (fast program buffer).

2.4.31. Fast program buffer correctable ECC address 0 register (FMC_FPB_CECCADDR0)

Address offset: 0xA8
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



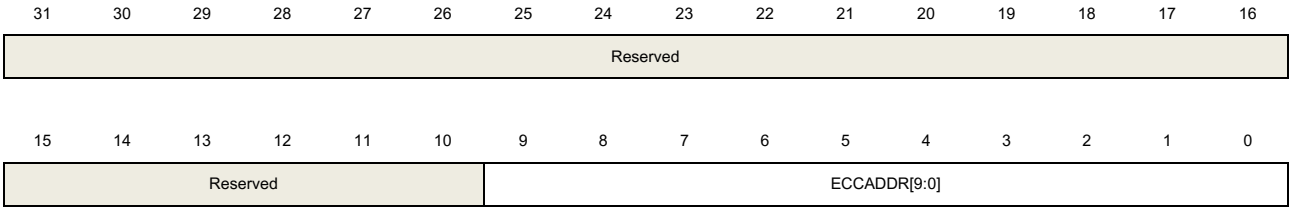
Bits	Fields	Descriptions

31:10	Reserved	Must be kept at reset value.
9:0	ECCADDR[9:0]	The offset address of double word where an correctable ECC error is detected The address is in ECC error address area (fast program buffer).

2.4.32. Fast program buffer correctable ECC Address 1 register (FMC_FPB_CECCADDR1)

Address offset: 0xAC
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

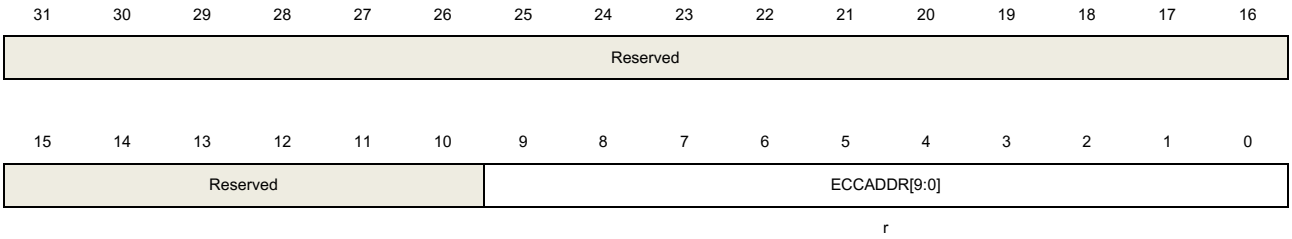


Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:0	ECCADDR[9:0]	The offset address of double word where an correctable ECC error is detected. The address is in ECC error address area (fast program buffer).

2.4.33. Fast program buffer correctable ECC address 2 register (FMC_FPB_CECCADDR2)

Address offset: 0xB0
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

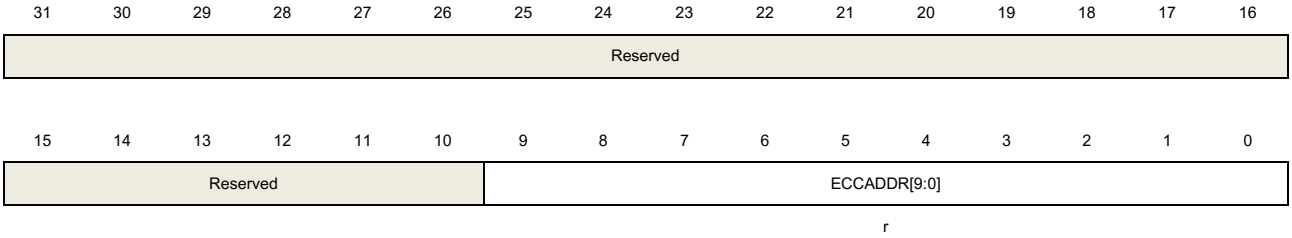


Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:0	ECCADDR[9:0]	The offset address of double word where an correctable ECC error is detected. The address is in ECC error address area (fast program buffer).

2.4.34. Fast program buffer correctable ECC address 3 register (FMC_FPB_CECCADDR3)

Address offset: 0xB4
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

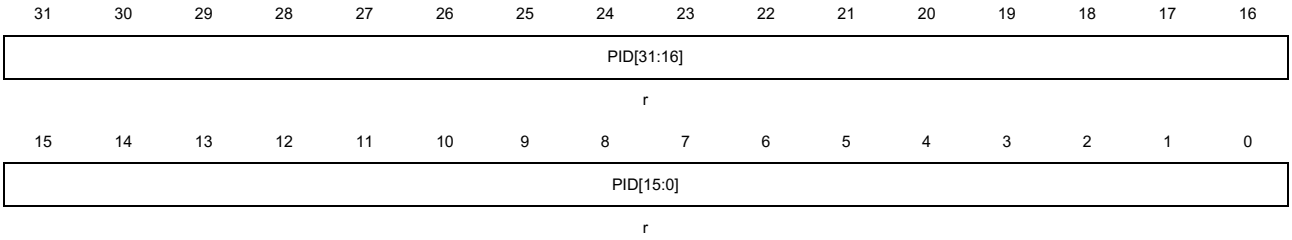


Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:0	ECCADDR[9:0]	The offset address of double word where an correctable ECC error is detected. The address is in ECC error address area (fast program buffer).

2.4.35. Product ID register 0 (FMC_PID0)

Address offset: 0xD0
Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:0	PID[31:0]	Product reserved ID code register 0 These bits are read only by software. These bits are unchanged constant after power on. These bits are one time program when the chip produced.

2.4.36. Product ID register 1 (FMC_PID1)

Address offset: 0xD4
Reset value: 0xXXXX XXXX

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PID[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PID[15:0]															
r															

Bits	Fields	Descriptions
31:0	PID[31:0]	Product reserved ID code register 1 These bits are read only by software. These bits are unchanged constant after power on. These bits are one time program when the chip produced.

3. Power management unit (PMU)

3.1. Overview

The power consumption is regarded as one of the most important issues for the devices. According to the Power Management Unit (PMU), it provides two types of work modes. These modes reduce the power consumption and allow the application to achieve the best tradeoff among the conflicting demands of CPU operating time, speed and power consumption. Pad retaining feature is supported that retains I/O configuration after exiting Standby mode. For safety consideration, analog BIST test for LDO and TSEN (Temperature sensor) is supported in PMU. The junction temperature can be monitored by comparing it with two temperature threshold levels.

For GD32A711/A712 series, there are four supply power domains, including V_{DDA}, V_{25}, V_{11}_Run and V_{11}_Standby domain. The power of the V_{DDA} domain is supplied directly by V_{DDA}. An embedded BOOT LDO (LDO_BOOT) is used to supply the V11 voltage for MCU to boot and work in low run mode. An embedded standby LDO (LCM, low power consumption mode) in V_{DDA} power domain is used to supply the V_{11}_Standby domain.

3.2. Characteristics

- LVD/HVD interrupts, voltage detector can issue an interrupt or event when the power is lower than or higher than a specific threshold.
- Analog BIST test for LDO and TEMP is supported.
- Temperature monitor is supported.
- Pad retaining feature is supported that retains I/O configuration after exiting standby mode.
- Application core boot address for CPU

- Two power work modes: low run and standby mode.
- BOOT internal voltage regulator (LDO_BOOT) supplies 1.1V for MCU boot and run.
- Standby internal voltage regulator (LDO_STB, LCM) supplies around 1.1V voltage for V_{11}_Standby domain when MCU is in standby mode.
- 2.5V internal voltage regulator (LDO_V25) supplies around 2.5V voltage for flash and other analog modules.

3.3. Function overview

3.3.1. Voltage domain description

Figure 3-1. GD32A711 /A712 series power supply overview

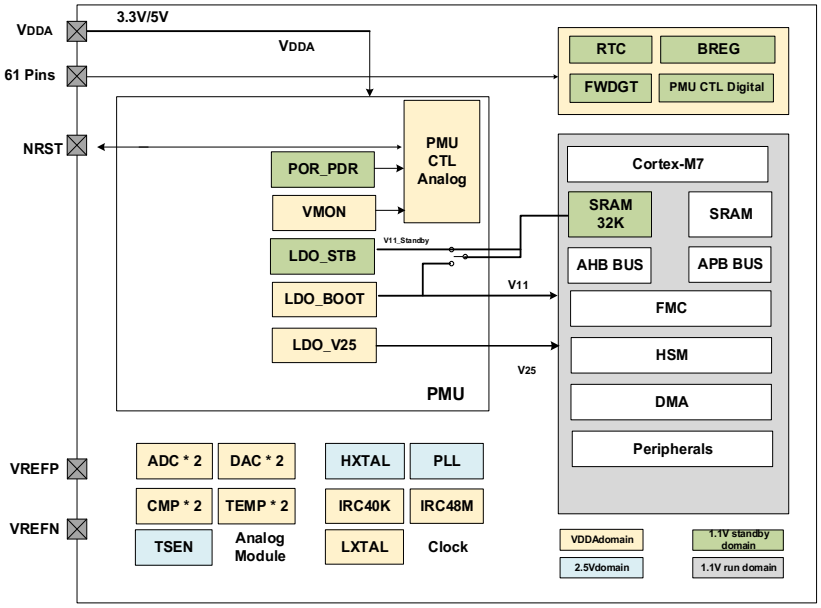


Figure 3-1. GD32A711 /A712 series power supply overview provides details on the internal configuration of the PMU， which is based on GD32A711 /A712 series.

Note: 1.The peripherals in the diagram refer to those not in the standby domain.

2.The connection methods of VREFP/VREFN vary depending on the package type. For detailed information, refer to the *VDDA supply power domain*.

VDDA supply power domain

V_{DDA} domain includes LDO_V25, LDO_STB, LDO_BOOT, BJT_CTL, POR_PDR (Power On/Down Reset), and analog monitors etc. V_{DDA} domain includes ADC/DAC (AD/DA Converter), IRC48M (Internal 48MHz RC Oscillator), IRC40K (Internal 40KHz RC oscillator), LVD (Low Voltage Detector), HVD (High Voltage Detector). The Backup domain is powered by the V_{DDA}, supplies power for RTC unit.

The POR_PDR circuit is implemented to detect V_{DDA} and generate the power reset signal which resets the whole chip except the backup domain when the supply voltage is lower than the specified threshold.

Most of analog circuits are powered by V_{DDA}. To improve the ADC and DAC conversion accuracy, the independent power supply V_{DDA} is implemented to achieve better performance of analog circuits. To ensure high accuracy of ADC and DAC, the independent external reference voltage for ADC/DAC can be connected to the VREFP/VREFN pins. According to

different package types, the VREFP pin can be connected to an external reference voltage, and the VREFN pin can be connected to the VSS pin. The VREFN pin is only available in packages with no less than 100 pins; otherwise, the VREFN pin is unavailable and internally connected to VSS.

V_{25} supply power domain

LDO_V25 shall generate a 2.5V voltage for flash to use. V_{25} power domain also includes HXTAL, PLL, TSEN(temperature sensor) and TEMP (temperature monitor).

V_{11} supply power domain

For GD32A711/A712 series, two LDOs can provides 1.1V voltage for digital logic domain. LDO_BOOT can be used to boot MCU and supply run domain when MCU is in low run mode. LDO_STB can be used to supply standby domain when MCU is in standby mode. There are two power domains in digital logics of MCU. Run domain shall work when MCU is in low run mode and be powered off when MCU in standby mode to save power. V_{11}_RUN supply power for run domain. Standby power domain is always on when MCU is powered up. V_{11}_Standby supplies power for standby domain.

HSM power domain

The HSM default power status is powered on after system reset.

3.3.2. Voltage monitors

For GD32A711/A712 series, low voltage monitors and high voltage monitors for V_{11}, V_{DDA}, V_{25} are implemented in PMU. When low voltage reset event is detected on V_{11}, V_{DDA}, V_{25}, a POR reset shall be generated.

After the initial power-up, the MCU will set the PORF and LVR flags of the PMU LVCS register to 1, and user could clear the corresponding flag bits according to their own needs.

The LVD/HVD status bit reflects the error condition of the analog voltage, and its corresponding flag bit is set only when the status bit changes from 0 to 1. Therefore, it is recommended to check the corresponding status bit before clearing the LVD/HVD flag bit to avoid situations where the flag bit cannot be set again under continuous LVD/HVD conditions, which would prevent FMU response from being triggered again. After the flag bit is cleared, the fault condition can only be determined by reading the status bit.

Note: LVD monitoring applies only to V_{DDA}5V power supply systems. For V_{DDA}3.3V power supply systems, the corresponding FMU monitoring does not need to be enabled.

3.3.3. Temperature monitors

Two high temperature monitors for LDO are implemented in PMU.

3.3.4. ABIST

Analog BIST circuits for test voltage monitors and temperature monitors are implemented in PMU to test whether analog circuits work properly.

Analog BIST is mainly targeted at voltage comparators for voltage detection across various voltage domains, as well as voltage comparators for temperature detection. It is implemented by injecting pattern voltage to each voltage monitor. The injection sequence is controlled by the analog BIST clock, and if the comparator output does not toggle as expected, a fault indication signal will be triggered.

(Note: It is recommended to clear the PMU_ABSTAT status bits and the corresponding flags in PMU_LVCS after ABIST execution is completed.)

3.3.5. Power work modes

After a system reset or a power reset, the MCU operates at low run mode with only LDO_BOOT is enabled if V_{15} is not ready.

Low run mode

After system reset/ power reset or wakeup from standby mode. The MCU enter low run mode with only LDO_BOOT enabled. LDO_BOOT is used to boot the MCU. In the low run mode, PLL could be on and all the peripherals could be on.

Standby mode

In standby mode, the whole 1.1V run domain is powered off, the LDO_BOOT are shut down, and HXTAL, LXTAL, IRC48M, IRC40K can be selectively enabled.

Table 3-1. General power saving mode summary

Series	State/Mode	LDO status	Description
GD32A711/A712	LOW_RUN	LDO_BOOT on LDO_STB off LDO_V25 on	After system reset / power reset or wakeup from standby mode, MCU enters low run mode for safe boot.
	STANDBY	LDO_BOOT off LDO_STB on LDO_V25 on optionally	In standby mode, the 1.1V RUN domain is powered off, the LDO is shut down, and HXTAL, LXTAL, IRC48M, IRC40K can be selectively enabled.

3.3.6. Standby mode entrance sequence

The standby mode entrance sequence contains three operation stages: software standby mode entrance sequence, hardware standby mode entrance sequence, standby mode entrance.

Software standby mode entrance sequence

Before starting the software standby mode sequence, PLL is not available in standby mode, the system clock source should be configured as IRC48M. Meanwhile, all clock sources could be disabled. Through writing 1 to V25LVRDEN bit of PMU_CFG register, user could use HXTAL.

The software standby mode entrance sequence includes four steps:
SW0: Module shutdown procedure
SW1: Application core shutdown procedure
SW2: Final step regulator control
SW3: Main core shutdown procedure

SW0: Module shutdown procedure

Considering that there is module clock disabling step in standby mode entrance sequence, it is not necessary to disable the module for standby mode.

Note: User must clear the IO pad settings, so as to avoid the unwanted pad retaining settings. For any wake source, before entering standby mode, merely wake up event of source must be enabled, and the corresponding interrupt must be disabled.

The IO and module configuration for standby mode is shown as follow:
1. Clear all pending interrupt and disable all communications.
2. Configure the RCU clock field about standby mode, configure the chip use IRC48M clock and update WSCNT.
3. Confirm whether there is necessary HXTAL to configure after exit standby mode.
4. If yes, configure HXTAL clock after standby mode and write 1 to V25LVRDEN bit of PMU_CFG register.
5. If no, verify V25LVRDEN bit of PMU_CFG register equals 0 and disable 2.5V during standby mode.
6. After step 4 or step 5, write 0 to the bit PAD_RET_CFG of CLTCFG_SUBSR_CTL0 register to enable pad retaining function.
7. Write 1 to PRENx field of GPIOx_PREN register of IO which user want active in standby mode.
8. Configure modules register which is used in standby mode.
9. Application core shutdown.

SW1: Application core shutdown procedure

The main core (enabling standby entry process) must be set in SW1, thus, SW3 includes the WFI execution.

SW2: Final step regulator

The final step regulator disable sequence is shown as follow:
1. Write 0 to FSREN bit and FSRCTLEN bit of PMU_CFG register meanwhile.
2. Software operates GPIOs, that is final step regulator is disabled.
3. Continue standby mode disable procedure.

The 1.1V capacitor is charging once exiting standby mode. During the 1.1V is charging or after the capacitor has been charged, user can charge the 1.5 V capacitor. If user charge the 1.5 V capacitor subsequently after the 1.1 V capacitor, user will spend more time on completing charging process. Keeping FSRCTLEN bit of PMU_CFG register equals 1 could save time at exiting Standby mode, meanwhile, avoid any additional current.

SW3: Main core shutdown procedure

Before disabling interrupts which is used to avoid missing any events, user must configure WKUP in SW0, user must disable interrupts and configure standby mode entrance (refer to SW0: Module shutdown process for more information). In SW3, user must perform only the main-core WFI execution.

Hardware standby mode entrance sequence

After software standby mode entrance sequence finish, launch the hardware standby mode entrance sequence. After hardware standby mode entrance sequence finish, launch standby mode entrance sequence, which includes following stages:
1. Once receiving standby mode entrance request, verify and launch the standby mode entrance.
2. Enable standby regulator and turn off SW0.
3. During the low-power procedure, disable boot regulator.
4. Disable V_{25} regulator and NCM(Normal Power Consumption Mode) LVR monitors.

3.3.7. Chip status at the end of standby mode entrance sequence

The chip complete standby mode entrance sequence is as follows, after standby mode entranced finished.
1. According to SW0, configure standby domain peripherals.
2. According to SW0, enable IOs pad retaining.
3. Except standby SRAM32K, power down the whole memory, during the standby mode, the V25 regulator could be remained on.
4. Standby domain is functional, while run domain is kept in reset.
5. Turn off the boot regulators.
6. Wait for wake-up event, so as to wakeups from standby mode.

3.3.8. Chip operations in standby mode

The chip supports functionalities in standby mode as follow:

- During standby mode, hold standby domain SRAM32K content.

- Up to 61 digital inputs could be used to wake up.
- analog inputs(CMP) in trigger mode functionality could be used to wake up
- RTC and FWDGT could be used to wake up
- In standby mode, selectively enable HXTAL, LXTAL, IRC48M, IRC40K; if HXTAL is enabled, the HXTALEN bit in the RCU_CTL register must be set to 1 and the V25LVRDEN bit in the PMU_CFG register must be set to 1 before entering standby mode

3.3.9. Standby mode exit

Subsystem reset event and system reset event could exit from standby mode, moreover, wake-up events are in a similar way.

After exit from Standby mode, the following events occur as following:
1. Wakeup event occurs.
2. If IRC48M is disabled in standby mode, IRC48M starts powering up.
3. Transition process to NCM mode.
4. The chip experience subsystem reset exit sequence.
5. The chip enters run mode.

Faster Standby mode exit

Based on higher capacitor recharging current consumption, the chip supports an optional configuration to faster exit standby mode, refer to PMU_CFG register FREC field.

Final-step regulator enable sequence

When the exit from standby mode, user must re-enable the final-step regulator firstly, and then increase the clock frequencies. The final step regulator enable sequence is shown as follow:
1. If uses BJT to generate 1.5V, turn on BJT collector voltage and write 1 to FSRCTLEN bit of PMU_CFG register. Check whether the FREC bit of PMU_CFG register is equal to 0, if 0, wait 50 microseconds, else wait 15 microseconds.
2. If does not use BJT to generate 1.5V, launch request to turn on 1.5V.
3. After step 1 or step 2, check whether the V15LVDSTAT bit of PMU_LVSC register is equal to 0, if 0, write 1 to V15LVDF bit of PMU_LVSC register , else continue to check the V15LVDSTAT value.
4. Write 1 to the FSREN bit of PMU_CFG register.
5. Wait t_{settle_fs} time before proceeding and then continue standby mode exit process.

3.3.10. Pad retain

In case of standby mode exit by reset, the pad retaining of the reset pin is removed when the chip is reset on standby mode. The signal corresponds to wake-up via reset event case.

Pad retaining on run domain pins

User should set the bit PAD_RET_CFG of CLTCFG_SUBSR_CTL0 register to be 0 before

configuring IO pad retaining.

If the chip exits standby mode, user should set the bit PAD_RET_CFG of CLTCFG_SUBSR_CTL0 register to be 1, so as to disable the pad retaining function.

For pad retaining on run domain pins, when the chip is in standby mode, even if PRENx field of GPIOx_PREN register is 0 or the bit PAD_RET_CFG of CLTCFG_SUBSR_CTL0 register is 1, pad should retain its state. if pad retaining is enabled on this pad before entering standby mode, until the bit PAD_RET_CFG of CLTCFG_SUBSR_CTL0 register is 1, pad should retain its value after exiting standby mode.

Pad retaining on standby domain pins

Pad availability in standby mode could enable or disable pad retaining, and standby mode pad retaining on IO is configured by writing to PRENx field of GPIOx_PREN register. It is necessary to make sure that the bit PAD_RET_CFG of CLTCFG_SUBSR_CTL0 register is 0, if the value is not 0, write 0 to the field before standby mode pad retaining entrance. PRENx field of GPIOx_PREN register must be 0 if the corresponding pad is not essential during standby mode.

For pad retaining on standby domain active pins, when the chip is in standby mode, if PRENx field of GPIOx_PREN register is 0 or the bit PAD_RET_CFG of CLTCFG_SUBSR_CTL0 register is 1, pad should not retain its state. if pad retaining is enabled on this pad before entering standby mode, until the bit PAD_RET_CFG of CLTCFG_SUBSR_CTL0 register is 1, pad should retain its value after exiting standby mode.

Pad retaining configuration procedure

The sequence which is used to enable IO pad retaining is shown as below:
1. Control the pad state before standby mode entrance by configuring GPIOx_PREN register.
2. IO pad retaining state during standby mode is needed to configure PRENx field of GPIOx_PREN register.
3. Write 0 to bit PAD_RET_CFG of CLTCFG_SUBSR_CTL0 register.
4. If user want to disable pad retaining, it is necessary to write 1 to CLTCFG bit PAD_RET_CFG of CLTCFG_SUBSR_CTL0 register field on standby mode.

3.4. Register definition

PMU base address: 0x4000 7000

Table 3-1. PMU register overview

Register Name	Symbol	Address Offset	Reset Value
Low voltage control and status register	PMU_LVCS	0x00	0xXXXX XXXX
Configuration register	PMU_CFG	0x04	0x9921 0000
Analog BIST control register	PMU_ABCTL	0x08	0x0000 0000
Analog BIST status register	PMU_ABSTAT	0x0C	0x0000 0000
Version ID register	PMU_VID	0x20	0xF000 0001
Control key register	PMU_CTLKEY	0x24	0x0000 5AF0
Mode configure register	PMU_MODCFG	0x28	0x0000 0000
Mode update register	PMU_MODUPD	0x2C	0x0000 0000
Mode status register	PMU_MODSTAT	0x30	0x0000 0000
Core update register	PMU_CUPD	0x40	0x0000 0000
Control register	PMU_CTL	0x48	0x0000 0000
TMR error enable register	PMU_TMRERREN	0x54	0x0000 0000
TMR error interrupt register	PMU_TMRERRIF	0x58	0x0000 0000
TMR error interrupt clear register	PMU_TMRERRIFC	0x5C	0x0000 0000

3.4.1. Low voltage control and status register (PMU_LVCS)

Address offset: 0x00

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PORF	Reserved	IRC40KSTB	IRC48MSTB	Reserved				V11LVRNLF	V11LVRNF	V25LVRNLF	V25LVRNF	Reserved		VALVRNLF	VALVRNF
r	r	r	r	r				rc_w1	rc_w1	rc_w1	rc_w1	r		rc_w1	rc_w1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPM1O	TPM0O	Reserved	V5ALVDSTAT	V11HVDSTAT	V25HVDSTAT	Reserved	VAHVDSTAT	TMP1F	TMP0F	Reserved	V5ALVDF	V11HVDF	V25HVDF	Reserved	VAHVDF
r	r	r	r	r	r	r	r	rc_w1	rc_w1	r	rc_w1	rc_w1	rc_w1	r	rc_w1

Bits	Fields	Descriptions
31	PORF	POR flag Indicates that a power-on reset event has occurred. Writing 1 to this field clears it, and other reset sources have no effect. 0: No power-on reset event has occurred. 1: Power-on reset event has occurred.
30	Reserved	Must be kept at reset value.

29	IRC40KSTB	IRC40K stabilization status 0: IRC40K has not been stabilization. 1: IRC40K has been stabilization.
28	IRC48MSTB	IRC48M stabilization status 0: IRC48M has not been stabilization. 1: IRC48M has been stabilization.
27:24	Reserved	Must be kept at reset value.
23	V11LVRNLF	1.1V domain low voltage reset flag (NCM or LCM) Indicates that a LVR event has occurred on the 1.1V power domain (NCM or LCM). Writing 1 to the field clears it, and other reset sources have no effect. 0: No low-voltage reset event has occurred. 1: Low-voltage reset event has occurred.
22	V11LVRNF	1.1V domain low voltage reset flag in NCM Indicates that a LVR event has occurred on the 1.1V power domain in the NCM. Writing 1 to this field clears it, and other reset sources have no effect. 0: No low-voltage reset event has occurred. 1: Low-voltage reset event has occurred.
21	V25LVRNLF	2.5V domain low voltage reset flag (NCM or LCM) Indicates that a LVR event has occurred on the 2.5V power domain (NCM or LCM). Writing 1 to this field clears it, and other reset sources have no effect. 0: No low-voltage reset event has occurred. 1: Low-voltage reset event has occurred.
20	V25LVRNF	2.5V domain low voltage reset flag in NCM Indicates that a LVR event has occurred on the 2.5V power domain in NCM. Writing 1 to this field clears it, and other reset sources have no effect. 0: No low-voltage reset event has occurred. 1: Low-voltage reset event has occurred.
19:18	Reserved	Must be kept at reset value.
17	VALVRNLF	V_{DDA} domain low voltage reset flag (NCM or LCM) Indicates that a LVR event has occurred on the V_{DDA} power domain (NCM or LCM). Writing 1 to this field clears it, and other reset sources have no effect. 0: No low-voltage reset event has occurred. 1: Low-voltage reset event has occurred.
16	VALVRNF	V_{DDA} domain low voltage reset in NCM Indicates that a LVR event has occurred on the V_{DDA} power domain in NCM. Writing 1 to this field clears it, and other reset sources have no effect. 0: No low-voltage reset event has occurred. 1: Low-voltage reset event has occurred. **Note:** All operations except reading the flash cannot be performed successfully

when this bit is set.

15	TPM1O	Temperature monitor 1 output. 0: temperature below the TEMP1 high level. 1: temperature equal or above the TEMP1 high level.
14	TPM0O	Temperature monitor 0 output. 0: temperature below the TEMP0 high level. 1: temperature equal or above the TEMP0 high level.
13	Reserved	Must be kept at reset value.
12	V5ALVDSTAT	V_{DDA} domain 5V LVD status in NCM The feature is only available in NCM and disabled in LCM. After a reset or wakeup from LCM, the software should clear the V5ALVDF flag and check the status bit V5ALVDSTAT to determine voltage level on V_{DDA} supply. 0: Voltage on V_{DDA} is above low-voltage detect threshold. 1: Voltage on V_{DDA} is below low-voltage detect threshold.
11	V11HVDSTAT	1.1V domain HVD status on in NCM This feature is only available in NCM and disabled in LCM. 0: Voltage on V_{11} is below high-voltage detect threshold or LCM. 1: Voltage on V_{11} is above high-voltage detect threshold and NCM.
10	V25HVDSTAT	2.5V domain HVD status in NCM The feature is only available in NCM and disabled in LCM. 0: Voltage on V_{25} is below high-voltage detect threshold or LCM. 1: Voltage on V_{25} is above high-voltage detect threshold and NCM.
9	Reserved	Must be kept at reset value.
8	VAHVDSTAT	V_{DDA} domain HVD status on in NCM The feature is only available in NCM and disabled in LCM. 0: Voltage on V_{DDA} is below high-voltage detect threshold or LCM. 1: Voltage on V_{DDA} is above high-voltage detect threshold and NCM.
7	TMP1F	Temperature monitor 1 interrupt flag for FMU. 0: Interrupt flag has not been set. 1: Interrupt flag has been set.
6	TMP0F	Temperature monitor 0 interrupt flag for FMU. 0: Interrupt flag has not been set. 1: Interrupt flag has been set.
5	Reserved	Must be kept at reset value.
4	V5ALVDF	V_{DDA} domain 5V LVD flag on in NCM set 1 to this field when V5ALVDSTAT status field is changed from 0 to 1. To clear V5ALVDF, write 1 to it. If enabled, V5ALVDF causes an interrupt request. 0: V5ALVDSTAT has not changed.

		1: V5ALVDSTAT has changed.
3	V11HVDF	1.1V domain HVD flag on in NCM set 1 to this field when the V11HVDSTAT status field is changed from 0 to 1. To clear V11HVDF, write 1 to it. If enabled, V11HVDF causes an interrupt request. 0: V11HVDSTAT has not changed. 1: V11HVDSTAT has changed.
2	V25HVDF	2.5V domain HVD flag in NCM set 1 to the V25HVDF field when V25HVDSTAT status field is changed from 0 to 1. To clear V25HVDF, write 1 to it. If enabled, V25HVDF causes an interrupt request. 0: V25HVDSTAT has not changed. 1: V25HVDSTAT has changed.
1	Reserved	Must be kept at reset value.
0	VAHVDF	V_{DDA} domain HVD flag in NCM set 1 to the VAHVDF field when VAHVDSTAT status field is changed from 0 to 1. To clear VAHVDF, write 1 to it. If enabled, VAHVDF causes an interrupt request. 0: VAHVDSTAT has not changed. 1: VAHVDSTAT has changed.

3.4.2. Configuration register (PMU_CFG)

Address offset: 0x04
Reset value: 0x9921 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								VALVRCTL	Reserved		STBIRC40KCTL	STBIRC48MCTL	Reserved		
r								rw	r		rw	rw	r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TEMP1SEL	TEMP1EN	TEMP0SEL	TEMP0EN	LVDIE	HVDIE	TMPIE	Reserved			V25LVRDEN	Reserved		
r		rw	rw	rw	rw	rw	rw	rw	r			rw	r		

Bits	Fields	Descriptions
31:24	Reserved	Must kept as reserved value.
23	VALVRCTL	Control V_{DDA} LVR 0: Once V_{DDA} is lower than LVR threshold, VALVRNLF and VALVRNF flag set, while, no reset generate. 1: Once V_{DDA} is lower than LVR threshold, generate POR reset.
22:21	Reserved	Must kept as reserved value.

20	STBIRC40KCTL	Control whether IRC40K is on in standby mode 0: IRC40K is off when enter standby mode. 1: IRC40K is on when enter standby mode.
19	STBIRC48MCTL	Control whether IRC48M is on in standby mode 0: IRC48M is off when enter standby mode. 1: IRC48M is on when enter standby mode.
18:14	Reserved	Must kept as reserved value.
13	TEMP1SEL	High temperature sensor 1 monitoring level selection 0: temperature threshold around 145°C 1: temperature threshold around 155°C
12	TEMP1EN	TEMP1 enable 0: TEMP1 disable 1: TEMP1 enable
11	TEMP0SEL	High temperature sensor 0 monitoring level selection 0: temperature threshold around 145°C 1: temperature threshold around 155°C
10	TEMP0EN	TEMP0 enable 0: TEMP0 disable 1: TEMP0 enable
9	LVDIE	Low voltage detects fmu enable Enables hardware fmu requests if any of V5ALVDF and V15LVDF flags is set. LVD fmu must be disabled before going into LCM. 0: LVD hardware fmu is disabled. 1: Request an LVD hardware fmu when V5ALVDF =1 or V15LVDF =1.
8	HVDIE	High voltage detect fmu enable Enables hardware fmu requests if any of the following flags is set: VAHVDF, VBHVDF, V25HVDF, V15HVDF, V11HVDF. 0: HVD hardware fmu is disabled. 1: Request an HVD hardware fmu when VAHVDF=1, VBHVDF =1, V25HVDF =1, V15HVDF =1 or V11HVDF =1.
7	TMPIE	Temperature detect interrupt enable for FMU 0: Temperature detect interrupt is disabled. 1: Temperature detect interrupt is enabled.
6:4	Reserved	Must kept as reserved value.
3	V25LVRDEN	2.5V domain LVR detection enable bit during LCM Controls whether the V_{25} regulator and low-voltage reset detection are active or inactive in LCM 0: V_{25} regulator and V25LVRNLF are disabled in LCM.

1: V_{25} regulator and V25LVRNLF are enabled in LCM.

2:0	Reserved	Must kept as reserved value.

3.4.3. Analog BIST control register (PMU_ABCTL)

Address offset: 0x08

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ABEN
															rw

Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	ABEN	Analog BIST enable signal 0: Write 0 has no effect. 1: Write 1 shall trigger an analog BIST test.

3.4.4. Analog BIST status register (PMU_ABSTAT)

Address offset: 0x0C

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			LVDVAAB	Reserved	HVDVAAB	HVDV25AB	HVDV11AB	Reserved	TEMP1AB	TEMP0AB	VAAB	Reserved	V25AB	V11STBAB	V11RUNAB
			rc_w1		rc_w1	rc_w1	rc_w1		rc_w1	rc_w1	rc_w1		rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12	LVDVAAB	Status of LVD V_{DDA} ABIST 0：Failed LVD V_{DDA} ABIST execution 1：Successful LVD V_{DDA} ABISTexecution
11	Reserved	Must be kept at reset value.

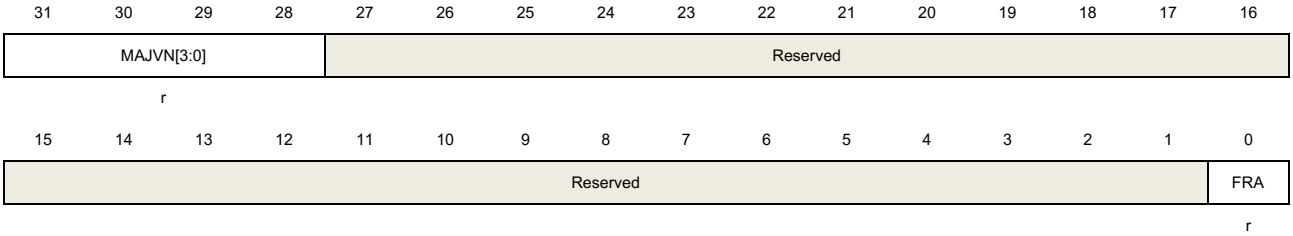
10	HVDVAAB	Status of HVD V_{DDA} ABIST 0：Failed HVD V_{DDA} ABIST execution 1：Successful HVD V_{DDA} ABISTexecution
9	HVDV25AB	Status of HVD V_{25} ABIST 0：Failed HVD V_{25} ABIST execution 1：Successful HVD V_{25} ABISTexecution
8	HVDV11AB	Status of HVD V_{11} ABIST 0：Failed HVD V_{11} ABIST execution 1：Successful HVD V_{11} ABISTexecution
7	Reserved	Must be kept at reset value.
6	TEMP1AB	Status of temperature 1 ABIST 0：Failed temperature 1 ABIST execution 1：Successful temperature 1 ABISTexecution
5	TEMP0AB	Status of temperature 0 ABIST 0：Failed temperature 0 ABIST execution 1：Successful temperature 0 ABISTexecution
4	VAAB	Status of V_{DDA} ABIST test 0：Failed V_{DDA} ABIST execution 1：Successful V_{DDA} ABISTexecution
3	Reserved	Must be kept at reset value.
2	V25AB	Status of 2.5V BIST 0：Failed 2.5V ABIST execution 1：Successful 2.5V ABISTexecution
1	V11STBAB	Status of 1.1V Standby ABIST 0：Failed 1.1V Standby ABIST execution 1：Successful 1.1V Standby ABISTexecution
0	V11RUNAB	Status of 1.1V RUN ABIST 0：Failed 1.1V RUN ABIST execution 1：Successful 1.1V RUN ABISTexecution

3.4.5. Version ID register (PMU_VID)

Address offset: 0x20

Reset value: 0xF000 0001

This register can be accessed by half-word (16-bit) or word (32-bit).

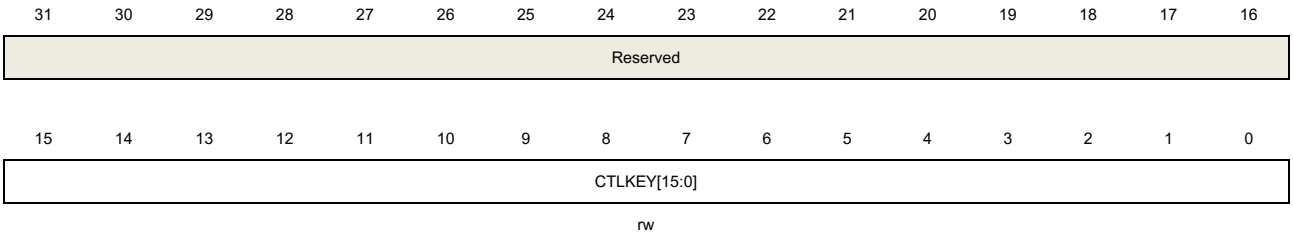


Bits	Fields	Descriptions
31:28	MAJVN[3:0]	Major version number Returns the version number for the specification
27:1	Reserved	Must kept as reserved value
0	FRA	Final step Regulator availability Indicates whether the final step regulator (1.5 V to 1.1 V) is available. 0: Not available 1: Available

3.4.6. Control key register (PMU_CTLKEY)

Address offset: 0x24
Reset value: 0x0000 5AF0

This register can be accessed by half-word (16-bit) or word (32-bit).

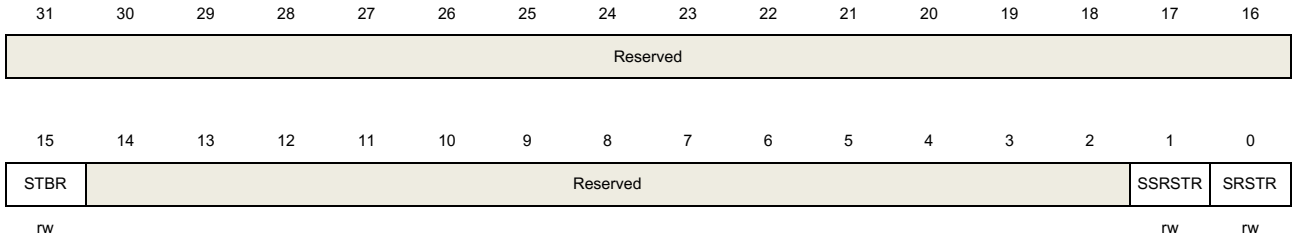


Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CTLKEY[15:0]	Control key Key for starting the hardware processes. Firstly write key and secondly write inverted key. The hexadecimal value of key is 0x5AF0 whereas for inverted key is 0xA50F. Writes with a value other than key or inverted key are ignored. Reads return bit inverted value corresponding to last write.

3.4.7. Mode configure register (PMU_MODCFG)

Address offset: 0x28
Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).



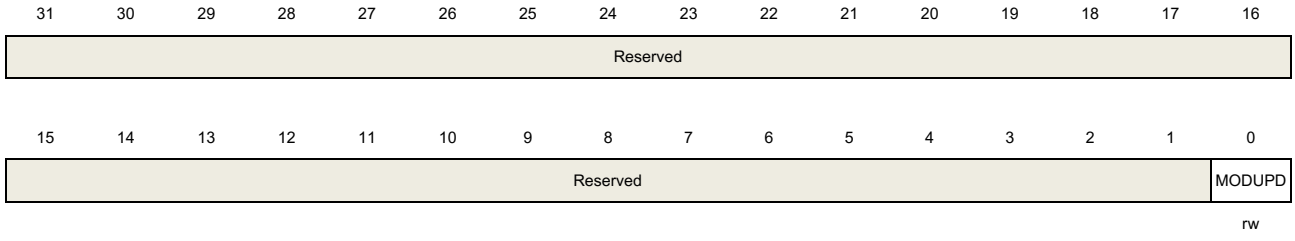
Bits	Fields	Descriptions
31:16	Reserved	Must kept as reserved value
15	STBR	Standby request, needs to be used in conjunction with writing the correct control code key value. 0: No effect 1: Set standby request
14:2	Reserved	Must kept as reserved value
1	SSRSTR	Subsystem reset request, needs to be used in conjunction with writing the correct control code key value. 0: No effect 1: Set subsystem reset request
0	SRSTR	System reset request, needs to be used in conjunction with writing the correct control code key value. 0: No effect 1: Set system reset request

3.4.8. Mode update register (PMU_MODUPD)

Address offset: 0x2C

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).



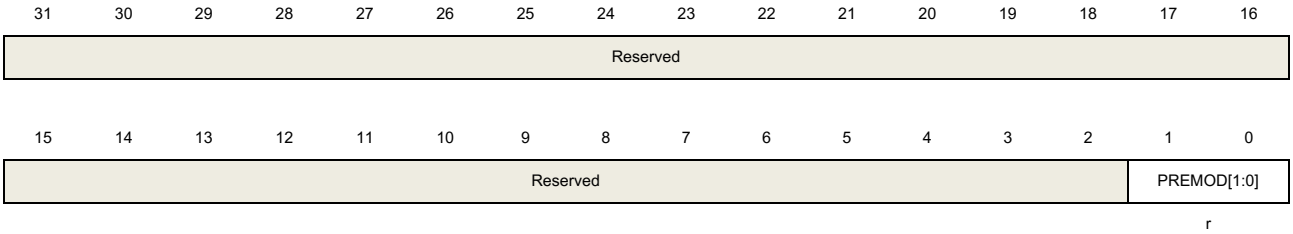
Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	MODUPD	Mode update 0: No effect 1: Writing a logic-1 to this bit, followed by a valid key combination initiates a mode change as per the PMU_MODCFG register.

3.4.9. Mode status register (PMU_MODSTAT)

Address offset: 0x30

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).



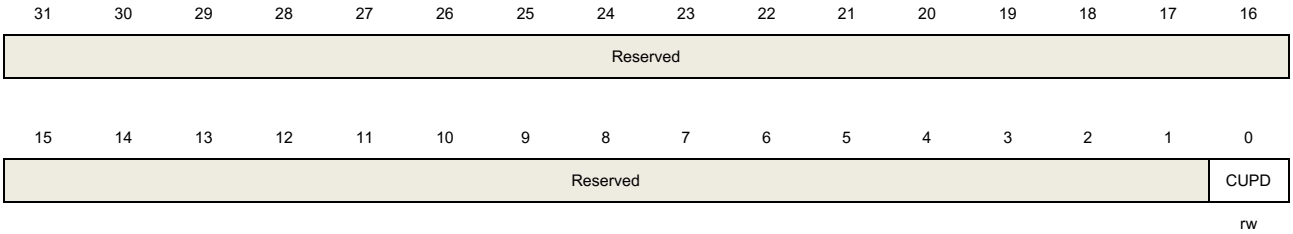
Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1:0	PREMOD[1:0]	Previous mode This bit shows the status of the previous mode. 00: The previous mode was reset (any reset). 01: The previous mode was standby

3.4.10. Core update register (PMU_CUPD)

Address offset: 0x40

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).



Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	CUPD	Core update 0: No effect 1: update core

3.4.11. Control register (PMU_CTL)

Address offset: 0x48

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														BKPWEN	Reserved
														rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits	Fields	Descriptions
31:18	Reserved	Must be kept at reset value.
17	BKPWEN	Write operation permission enable for backup domain 0: disable backup domain write operation 1: enable backup domain write operation
16:0	Reserved	Must be kept at reset value.

3.4.12. TMR error enable register (PMU_TMRERREN)

Address offset: 0x54

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															CUPDEN
															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODUPDEN	TKEYEN	ABEN	ABENEN	Reserved	FSRATOEN	TEMPENEN	LVDIE	HVDIE	FSRCTLEN	FSREN	PORFEN	LVREN	TEMPEN	LVDEN	HVDEN
rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value.
16	CUPDEN	0: disable core update TMR error interrupt 1: enable core update TMR error interrupt
15	MODUPDEN	0: disable mode update TMR error interrupt 1: enable mode update TMR error interrupt
14	TKEYEN	0: disable test key TMR error interrupt 1: enable test key TMR error interrupt
13	ABEN	0: disable ABIST TMR error interrupt 1: enable ABIST TMR error interrupt
12	ABENEN	0: disable ABIST enable interrupt 1: enable ABIST enable interrupt

11	Reserved	Must be kept at reset value.
10	FSRATOEN	0: disable FSRATO interrupt 1: enable FSRATO interrupt
9	TEMPENEN	0: disable temperature enable interrupt 1: enable temperature enable interrupt
8	LVDIE	0: disable LVD interrupt 1: enable LVD interrupt
7	HVDIE	0: disable HVD interrupt 1: enable HVD interrupt
6	FSRCTLEN	0: disable FSRCTL interrupt 1: enable FSRCTL interrupt
5	FSREN	0: disable FSR interrupt 1: enable FSR interrupt
4	PORFEN	0: disable PORF TMR error interrupt 1: enable PORF TMR error interrupt
3	LVREN	0: disable LVR TMR error interrupt 1: enable LVR TMR error interrupt
2	TEMPEN	0: disable temperature interrupt 1: enable temperature interrupt
1	LVDEN	0: disable LVD TMR error interrupt 1: enable LVD TMR error interrupt
0	HVDEN	0: disable HVD TMR error interrupt 1: enable HVD TMR error interrupt

3.4.13. TMR error interrupt register (PMU_TMRERRIF)

Address offset: 0x58

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															CUPDIF
															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODUPDIF	TKEYIF	ABIF	ABENIF	Reserved	FSRATOIF	TEMPENIF	LVDIEIF	HVDIEIF	FSRCTLIF	FSRIF	PORFIF	LVRIF	TEMPIF	LVDIF	HVDIF
rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions

31:17	Reserved	Must be kept at reset value.
16	CUPDIF	Core update TMR error interrupt flag
15	MODUPDIF	Mode update TMR error interrupt flag
14	TKEYIF	Test key TMR error interrupt flag
13	ABIF	ABIST TMR error interrupt flag
12	ABENIF	ABIST enable interrupt flag
11	Reserved	Must be kept at reset value.
10	FSRATOIF	FSRATO interrupt flag
9	TEMPENIF	Temperature enable interrupt flag
8	LVDIEIF	LVD interrupt enable flag
7	HVDIEIF	HVD interrupt enable flag
6	FSRCTLIF	FSRCTL interrupt flag
5	FSRIF	FSR interrupt flag
4	PORFIF	PORF TMR error interrupt flag
3	LVRIF	LVR TMR error interrupt flag
2	TEMPIF	Temperature interrupt flag
1	LVDIF	LVD TMR error interrupt flag
0	HVDIF	HVD TMR error interrupt flag

3.4.14. TMR error interrupt clear register (PMU_TMRERRIFC)

Address offset: 0x5C

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															CUPDIFC
															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODUPDIFC	TKEYIFC	ABIFC	ABENIFC	Reserved	FSRATOIFC	TEMPENIFC	LVDIEIFC	HVDIEIFC	FSRCTLIFC	FSRIFC	PORFIFC	LVRIFC	TEMPIFC	LVDIFC	HVDIFC
rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value.

16	CUPDIFC	Core update TMR error interrupt flag clear
15	MODUPDIFC	Mode update TMR error interrupt flag clear
14	TKEYIFC	Test key TMR error interrupt flag clear
13	ABIFC	ABIST TMR error interrupt flag clear
12	ABENIFC	ABIST enable interrupt flag clear
11	Reserved	Must be kept at reset value.
10	FSRATOIFC	FSRATO interrupt flag clear
9	TEMPENIFC	Temperature enable interrupt flag clear
8	LVDIEIFC	LVD interrupt enable flag clear
7	HVDIEIFC	HVD interrupt enable flag clear
6	FSRCTLIFC	FSRCTL interrupt flag clear
5	FSRIFC	FSR interrupt flag clear
4	PORFIFC	PORF TMR error interrupt flag clear
3	LVRIFC	LVR TMR error interrupt flag clear
2	TEMPIFC	Temperature interrupt flag clear
1	LVDIFC	LVD TMR error interrupt flag clear
0	HVDIFC	HVD TMR error interrupt flag clear

4. Backup registers (BKP)

4.1. Overview

The Backup registers are located in the Backup domain that remains powered-on by V_{DDA} power, there are ten 16-bit (20 bytes) registers for data protection of user application data, and the wake-up action from standby mode or system reset do not affect these registers.

In addition, the BKP registers can be used to implement the tamper detection and RTC calibration function.

After reset, any writing access to the registers in Backup domain is disabled, that is, the Backup registers and RTC cannot be written to access. In order to enable access to the Backup registers and RTC, the power and Backup interface clocks should be enabled firstly by setting the PMUEN and BKPEN bits in the RCU_APB1EN register, and writing access to the registers in Backup domain should be enabled by setting the BKPWEN bit in the PMU_CTL register.

4.2. Characteristics

- 20 bytes Backup registers which can keep data under power saving mode. If tamper event is detected, Backup registers will be reset.
- The active level of tamper source (PC13) can be configured.
- RTC clock calibration register provides RTC alarm and second output selection, and sets the calibration value.
- Tamper control and status register (BKP_TPCS) can control tamper detection with interrupt or event capability.

4.3. Function overview

4.3.1. RTC clock calibration

In order to improve the RTC clock accuracy, the MCU provides the RTC output for calibration function. The RTC clock, or a clock with the frequency is $f_{RTCCLK}/64$, can be output on the PC13. It is enabled by setting the COEN bit in the BKP_OCTL register.

The calibration value is set by RCCV[6:0] in the BKP_OCTL register, and the calibration function can slow down or speed up the RTC clock by steps of 1000000/2^20 ppm.

4.3.2. Tamper detection

In order to protect the important user data, the MCU provides the tamper detection function, and it can be independently enabled on RTC_TAMPER pin by setting corresponding TPEN

bit in the BKP_TPCTL register. To prevent the tamper event from losing, the edge detection is logically ANDed with the TPEN bit, used for tamper detection signal. So the tamper detection configuration should be set before enable RTC_TAMPER pin. When the tamper event is detected, the corresponding TEF bit in the BKP_TPCS register will be set. Tamper event can generate an interrupt if tamper interrupt is enabled. Any tamper event will reset all Backup data registers.

Note: When TPAL=0 / 1, if the RTC_TAMPER pin is already high / low before it is enabled (by setting TPEN bit), an extra tamper event is detected, while there was no rising / falling edge on the RTC_TAMPER pin after TPEN bit was set.

4.4. Register definition

BKP base address: 0x4000 6C00

Table 4-1. BKP register overview

Register name	Symbol	Address offset	Reset value
Backup data register x	BKP_DATAx	0x04	0x0000 0000
RTC signal output control register	BKP_OCTL	0x2C	0x0000 0000
Tamper pin control register	BKP_TPCTL	0x30	0x0000 0000
Tamper control and status register	BKP_TPCS	0x34	0x0000 0000

4.4.1. Backup data register x (BKP_DATAx) (x= 0...9)

Address offset: 0x04 to 0x28

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA [15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	DATA[15:0]	Backup data These bits are used for general purpose data storage. The contents of the BKP_DATAx register will remain even if wake-up action from standby mode or system reset.

4.4.2. RTC signal output control register (BKP_OCTL)

Address offset: 0x2C

Reset value: 0x0000 0000

This register can be accessed by half-word(16-bit) or word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CALDIR	CCOSEL	Reserved				ROSEL	ASOEN	COEN	RCCV[6:0]						
rw	rw					rw	rw	rw	rw						

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	CALDIR	RTC clock calibration direction 0: Slowed down 1: Speed up This bit is reset only by a Backup domain reset.
14	CCOSEL	RTC clock output selection 0: RTC clock div 64 1: RTC clock This bit is reset only by a POR.
13:10	Reserved	Must be kept at reset value.
9	ROSEL	RTC output selection 0: RTC alarm pulse is selected as the RTC output 1: RTC second pulse is selected as the RTC output This bit is reset only by a Backup domain reset.
8	ASOEN	RTC alarm or second signal output enable 0: Disable RTC alarm or second output 1: Enable RTC alarm or second output When enable, the RTC_TAMPER pin will output the RTC output. This bit is reset only by a Backup domain reset.
7	COEN	RTC clock calibration output enable 0: Disable RTC clock calibration output 1: Enable RTC clock calibration output When enable, the RTC_TAMPER pin will output the RTC clock or RTC clock divided by 64. ASOEN has the priority over COEN. When ASOEN is set, the RTC_TAMPER pin will output the RTC alarm or second signal whether COEN is set or not. This bit is reset only by a POR.
6:0	RCCV[6:0]	RTC clock calibration value The value indicates how many clock pulses are ignored or added every 2^20 RTC clock pulses. This bit is reset only by a Backup domain reset.

4.4.3. Tamper pin control register (BKP_TPCTL)

Address offset: 0x30

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														TPAL	TPEN
														rw	rw

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	TPAL	RTC_TAMPER pin active level 0: The RTC_TAMPER pin is active high 1: The RTC_TAMPER pin is active low
0	TPEN	TAMPER detection enable 0: The RTC_TAMPER pin is free for GPIO functions 1: The RTC_TAMPER pin is dedicated for the Backup reset function. The active level on the RTC_TAMPER pin resets all data of the BKP_DATAx register.

4.4.4. Tamper control and status register (BKP_TPCS)

Address offset: 0x34

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						TIF	TEF	Reserved					TPIE	TIR	TER
						r	r						rw	w	w

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9	TIF	Tamper interrupt flag 0: No tamper interrupt occurred 1: A tamper interrupt occurred This bit is reset by writing 1 to the TIR bit or the TPIE bit being 0.
8	TEF	Tamper event flag 0: No tamper event occurred 1: A tamper event occurred This bit is reset by writing 1 to the TER bit.
7:3	Reserved	Must be kept at reset value
2	TPIE	Tamper interrupt enable 0: Disable the tamper interrupt 1: Enable the tamper interrupt

		This bit is reset only by a system reset and wake-up from Standby mode. **Note:** Non-reset wakeup cannot reset this bit.
1	TIR	Tamper interrupt reset 0: No effect 1: Reset the TIF bit This bit is always read as 0.
0	TER	Tamper event reset 0: No effect 1: Reset the TEF bit This bit is always read as 0.

5. Reset and clock unit (RCU)

5.1. Reset control unit (RCTL)

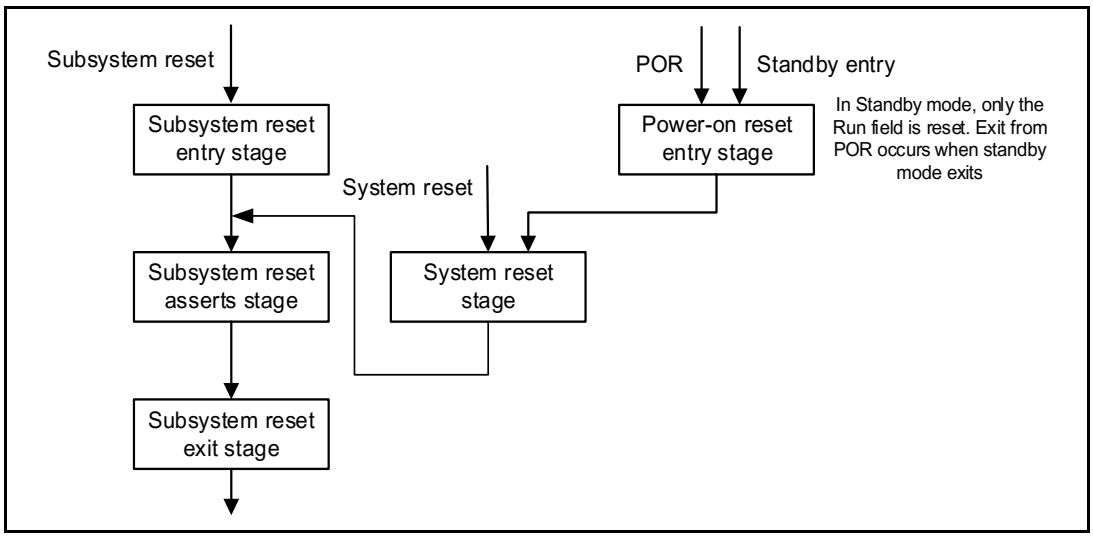
5.1.1. Reset overview

GD32A711 / A712 devices reset control includes the control of three kinds of reset: power-on reset, system reset and subsystem reset. The power-on reset resets the chip according to the complete chip reset sequence. A system reset resets most parts of the chip include the SRAM. A subsystem reset resets every communication peripheral and core, and the communication protocols need to be reinitialized after reset. The function of some modules (more details refer to ***Subsystem reset*** section) and the contents of the SRAM are not destroyed after a subsystem reset.

(Note: SRAM, TCM, and CPU cache in the GD32A711/A712 are not initialized by hardware. After a power-on reset or system reset, software must initialize them as required by the application to prevent ECC error reports.)

The chip reset follows a certain reset sequence in which the POR has the highest priority and the subsystem reset has the lowest priority. The chip reset logic is shown as follows.

Figure 5-1. Chip reset procedure



No matter in which stage POR occurs, the chip begins to enter power-on entry reset stage. When the system reset occurs during power-on stage, the chip reset procedure does not change, otherwise the chip begins to enter system reset stage. When subsystem reset occurs out of reset, the chip begins to enter subsystem entry reset stage, otherwise the chip reset procedure does not change.

Chip reset actions

When RCTL captures the chip reset events, the following actions are performed:

1. The corresponding reset event flag bits in RCU_SYSRSTSTAT and RCU_SUBSERSTAT are set.
2. Restore all pins to the default state (More information is in the GPIO section).
3. NRST pin is asserted.
4. Begins to enter the chip reset stage according to the chip reset procedure.

5.1.2. POR reset

Power-on reset sources

Power-on reset is generated by the PMU or POR_WDG, and asserts the NRST pin. The flag PORF corresponding to the power-on reset is recorded in RCU_SYSRSTSTAT. The followings are the types of POR reset:

- POR in the low-voltage domain.
- LVR on 1.1 V supply in Standby mode.
- LVR on 1.1 V supply in Run mode.
- LVR on 2.5 V supply in Standby mode.
- LVR on 2.5 V supply in Run mode.
- LVR on VDDA supply in Standby mode.
- LVR on VDDA supply in Run mode.
- LVR on VDDB supply in Standby mode.
- LVR on VDDB supply in Run mode.
- POR_WDG timeout (refer to the POR_WDG chapter for more information).

Power-on reset stage

There are two stages of power-on reset:

1. **POWER-UP** stage: the chip enters the POWER-UP stage after the POR reset occurs. During this process, the chip waits for the system to be powered up and exits the POWER-UP stage to enter the IRC48M_START stage after all POR reset sources are cleared.
2. **IRC48M_START** stage: the chip starts IRC48M_START after POWER-UP stage completes. If IRC48M is enabled, it is available after stabilization. Stability duration depends on the clock startup time (refer to the datasheet for more information). After IRC48M stabilization step completes, switch to the system reset stage.

When POR reset occurs, all modules which are shown in *Table 5-1. POR reset modules (For GD32A711/A712 devices)* reset.

Table 5-1. POR reset modules (For GD32A711/A712 devices)

Module	Power domain	Part of LBIST
RCTL	Standby	No
TRIGSEL	Run	No
BKP	Standby	No
CCTL	Run	No

Module	Power domain	Part of LBIST
HXTAL	Standby	No
PLL	Run	No
CLTCFG	Standby	No
HSM	Run	No
DTM_CAN	Run	No
I2C	Run	No
SPI/I2S	Run	No
QSPI	Run	No
ADC	Run	Yes
DAC	Run	Yes
TIMER	Run	Yes
CRC	Run	Yes
FMU	Run	Yes
STCM	Run	No
WWDGT	Run	Yes
DMA	Run	Yes
DMAMUX	Run	Yes
PMU	Standby	No
FMC	Run	Yes
IRC40K	Standby	No
SYSCFG	Run	No
RTC	Run	No
FWDGT0	Standby	No
FMC	Run	No
SENT	Run	No
LINFlexD	Run	No
IRM	Run	Yes
MTC	Run	Yes
GPIO	Run	No
JTAG	Run	No
Cortex-M7_0	Run	No

5.1.3. System reset

System reset sources

Table 5-2. System reset sources

Reset source	Flag in RCU_SYSRSTSTAT	System reset cause
FMU	FMU_REA_FAIL	FMU react failed
DBG	DBG_SYS_RSTF	Debug system reset
STCM	STCM_REC_FAIL	STCM recover failed
RCTL	RCTL_SUBSRUF	Subsystem reset upgrade
PLL	LOPRSTF	Loss of PLL error
CKFM0	LOHRSTF	Loss of HXTAL error
CKFM1	SYS_CLK_FAIL	Core clock fail
CKFM2	CORE_CLK_FAIL	System clock fail
HSM	HSM_SYS_RSTF	HSM system reset detection
PMU	SW_SYS_RSTF	Software system reset
NRST pin	EXRF	NRST pin asserted

Note: All system resets can be escalated, but only the LOPRSTF reset can be demoted to an interrupt. More details refer to ***Table 2-5. OB_CLTCFG option bytes fields description***.

System reset stage

SYSRST0 stage: the entire chip is reset except the running POR logic sequence, until all system reset events are cleared. The system reset assertion waits for at least ten IRC48M cycles and deasserts after the system reset stage is complete.

When the system reset occurs, part of the GPIO, IRC40K, FMC, PMU, CLTCFG, IRC48M, CCTL, and RCTL modules are reset, the JTAG module is not reset. Other modules showing in ***Table 5-1. POR reset modules (For GD32A711/A712 devices)*** are completely reset.

System reset upgrade

The system reset upgrade function is enabled by writing a non-zero value to the RCU_SYSMRUTH register. After this function is enabled, each system reset event increases the counter value. When the counter reaches the preset value of RCU_SYSMRUTH register, the device enters and maintains the SYSRST0 stage until the next power-on reset.

CLTCFG configures the reset upgrade function of the system reset source by option bytes. Power-on reset and external reset cannot be upgraded. More details refer to ***Table 2-5. OB_CLTCFG option bytes fields description***.

5.1.4. Subsystem reset

Subsystem reset sources

Table 5-3. Subsystem reset sources (For GD32A711/A712 devices)

Reset source	Flag in RCU_SUBSERSTAT	NRST assertion	Downgrade to IRQ	Upgrade	Subsystem reset cause
FMU	FMU_RSTF	Always	Yes	Yes	FMU reset reaction
DBG	DBG_RSTF	Always	Yes	Yes	Debug subsystem reset
STCM	ST_FINISH	Configurable	No	No	Self-test finished
FWDGT0	FWDGT0_RSTF	Always	Yes	Yes	FWDGT0 reset request
WWDGT0	WWDGT0_RSTF	Always	Yes	Yes	WWDGT0 reset request
CPU	SWRSTF	Always	NO	Yes	The SYSRESETREQ bit in Cortex®-M7 application interrupt and reset control register is set (SW_RSTn)
HSM	HSM_WDGT_RSTF	Always	No	Yes	HSM WDGT timeout
HSM	HSM_APP_RSTF	Always	No	Yes	HSM app reset
PMU	SW_SUBS_RSTF	Always	No	Yes	Software subsystem reset
FMC	OBL_RSTF	Always	Yes	Yes	Option byte loader reset
PMU	OBSTDBY_RSTF	Always	No	Yes	Standby reset

Subsystem reset stage

The subsystem reset stages are as follows:

1. **SUBSRST0** stage: The chip begins to enter this stage after any subsystem reset event occurs. In order to avoid false faults or reset, FMU fault monitoring are masked in this stage.
2. **SUBSRST1** stage: RCTL triggers a halt signal to SRAM and TCMSRAM, lock the SRAM and prohibit other access, and can halt bus master and slave access by configuring the bus control register (RCU_BUSCTL).
3. **SUBSRST2** stage: System clock switches to IRC48M.
4. **SUBSRST3** stage: All CCTL hardware clock multiplexers restore frequency divider to the default state except the PLL multiplication and division factors.
5. **SUBSRST4** stage: Turn off PLL synchronization.

6. **SUBSRST5** stage: Switch off HXTAL synchronization. SUBSRST4 and SUBSRST5 stages ensure that PLL is cleanly disabled to ensure that the PLL clock does not malfunction due to a reset.

7. **SUBSRST6** stage: The clocks of the modules working in the LBIST process and system reset are enabled to ensure that the reset synchronization requirements in these process are met when relevant conditions occur. For the self-test logical sequence, the system reset exits after this stage completes and the safe state is removed. All logical parts in the self-test (POR, system and subsystem reset) are reset after the self-test completes, while the rest of the chip undergoes a subsystem reset sequence due to the self-test completion.

8. **SUBSRST7** stage: The RCTL enables subsystem resets and triggers the counter (up to 64 cycles) running on IRC48M to enable the clock of the module with synchronous reset requests. Flash memory exits reset after the stage completes. The flash memory is reset during subsystem reset, and it completes and exits the reset at the beginning of the subsystem reset exit stage.

9. **SUBSRST8** stage: Wait for FLASH to be initialized and ready.

10. **SUBSRST9** stage: After entering this stage, RCTL checks whether the signal is asserted externally and stops driving NRST. If low-power debug is enable, RCTL waits a debug signal in this stage. Completion of this stage indicates that the RCTL has completed the reset sequence, releasing the subsystem reset.

In the above subsystem reset stages, SUBSRST0~SUBSRST6 belong to the subsystem reset entry stage, and SUBSRST7~SUBSRST9 belong to the subsystem reset exit stage.

When the chip is subsystem reset, the Cortex-M7_0, GPIO, IRC40K, FMC, PMU, CLTCFG, IRC48M, CCTL, RCTL modules are partially reset, and the STCM, FMU, JTAG module is not reset. The SRAM content is retained, but its control logic will be reset. Other modules showing in ***Table 5-1. POR reset modules (For GD32A711/A712 devices)*** are completely reset.

Subsystem reset upgrade and downgrade

The subsystem reset upgrade is enabled by writing a non-zero value to the register RCU_SUBSRMUTH. After this function is enabled, each subsystem reset increases the count value. If the count value reaches the threshold before resetting, a system reset is generated. When the counter reaches the preset value of register RCU_SUBSRMUTH, the device enters the system reset stage.

In addition to reset upgrades, the chip can be configured to downgrade subsystem reset to an interrupt (more details refer to Subsystem reset control register (RCU_SUBSRCTL)).

Subsystem reset entry timer

The purpose of the subsystem reset entry timer is to refer to any situation in which the subsystem reset entry sequence is not completed within the preset or default (chip factory settings) time.

The timer starts from the subsystem reset trigger and terminates if the subsystem reset entry sequence completes successfully or reaches the timeout value.

The timeout value is configured by SUBSRTO[30:0], if the bit field is zero, the default timeout (fixed for the chip) is used.

The default timeout value of the subsystem reset entry timer is 2048 clocks of RCTL clock (HXTAL). If a timeout occurs, the POR_WDG event is generated.

5.1.5. Reset pin

NRST pin is a bidirectional reset pin indicating the reset status. Its default function is a dedicated reset signal, which can be used as an external system reset source or as an indicator of subsystem reset and system reset.

RCTL manages external resets from NRST. Falling edge detection on NRST resets the sequence from the beginning of the system reset entry sequence. The status flag associated with the external reset falling edge event (the EXRF bit in RCU_SUBSERSTAT register) is set when an external reset is asserted without a power-on reset.

RCTL asserts an external reset if the reset event (POR, system reset, subsystem reset) triggers the reset sequence. When the device is in the reset sequence, RCTL asserts the NRST pin and holds the assertion that the other chip pins remain in a safe state based on the reset type until the end of the reset sequence. During the main reset field self-test of the chip, RCTL asserts the NRST by writing 1 to the ERPAETEN bit in RCU_ERPACFG register by software without starting the reset sequence.

5.1.6. Standby reset

In addition to the reset sequence in the chip standby entry sequence, the logic sequence in the run domain is also reset, where all resets are asserted (POR, system, and subsystem) to the run domain logic. The standby entry sequence of the chip has no effect on the logic of the standby power domain. Wake-up from standby mode removes the reset to the run domain in the standby exit sequence.

5.1.7. Chip reset and boot

Chip reset includes different types of reset sources and reset stages corresponding to the reset type. Different types of reset events have different entry points, but they all follow certain reset processes to ensure the deterministic state after the completion of chip reset. ***Figure 5-2. The chip startup sequence*** shows the entire process of the chip startup sequence. ***Figure 5-3. The chip reset timing*** shows the chip reset timing.

Figure 5-2. The chip startup sequence

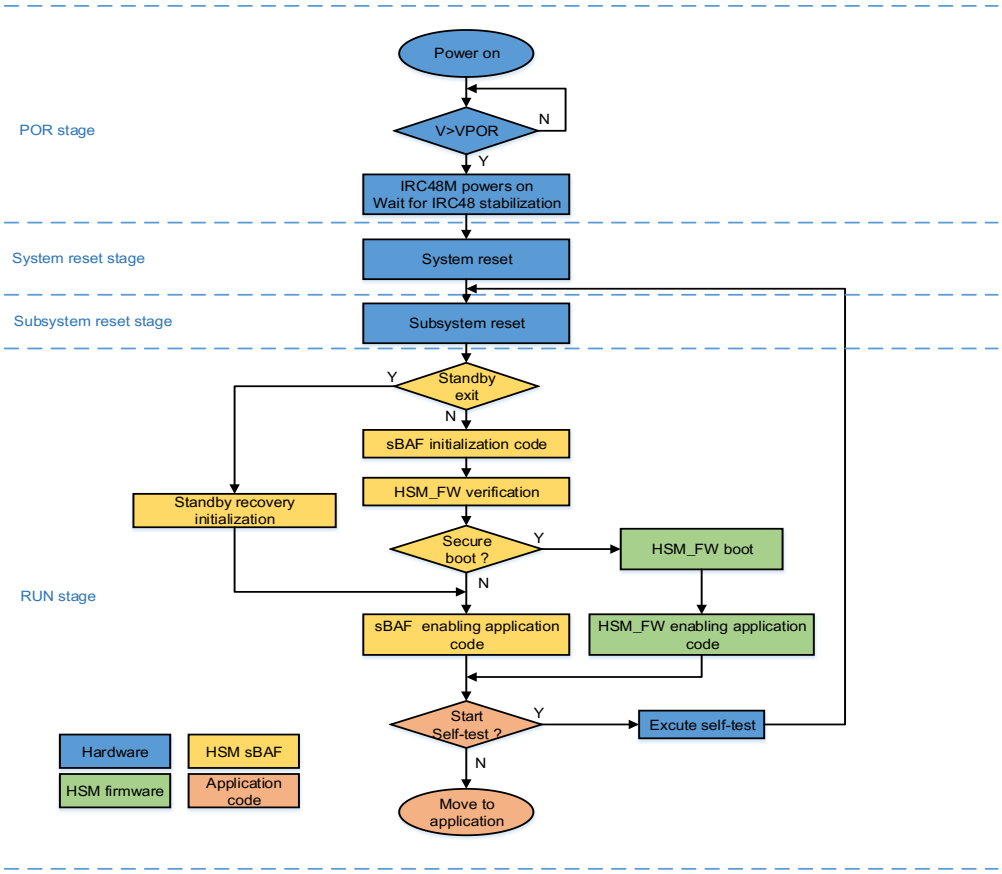
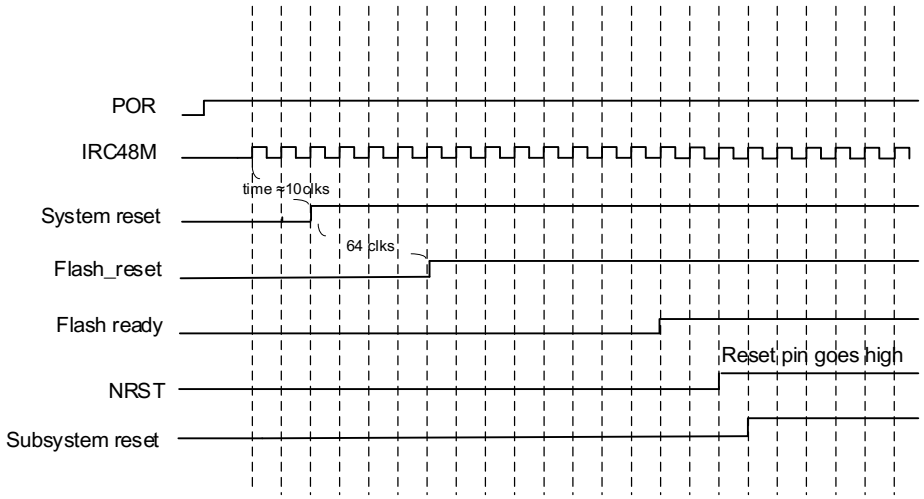


Figure 5-3. The chip reset timing



5.1.8. Power-on reset watchdog (POR_WDG)

POR_WDG is used to detect the fault of the chip stuck or suspended in the reset and standby sequences. A POR reset occurs when the watchdog timer exceeds a given time threshold. The threshold for the POR watchdog counter is configurable. The clock source of POR_WDG is IRC40K.

POR_WDG can be configured with 4 different thresholds, which can obtain the status of the input signal in the corresponding register, monitor reset, standby entry and standby exit.

POR_WDG has different configurations and application scenarios for the monitoring of reset and standby entry/exit. *Table 5-4. Details of POR_WDG monitoring* describes the details of POR_WDG monitoring.

PORWDG only reset the digital module, does not reset the analog module, LVR reset the entire chip including digital module and analog module.

Table 5-4. Details of POR_WDG monitoring

POR_WDG monitoring	Inactive monitoring scenarios	Monitoring scenarios
Reset monitoring	■ Configure the bit7 of the option byte OB_CLTCFG[7:0] . ■ System reset upgrade when the device remains in the reset state. ■ Maintain the reset by holding the external reset pin asserted. ■ During self-testing, write 1 to the ERPAETEN bit in RCU_ERPACFG register.	■ Assert reset pin. ■ Subsystem reset is asserted in standby domain.
Standby entry/exit monitoring	■ Configure the PWD_MONDIS bit in CLTCFG_POR_CTL0 register.	■ Monitoring from standby entry sequence start until end ■ The chip wakeup event occurs until the RUN domain reset is complete

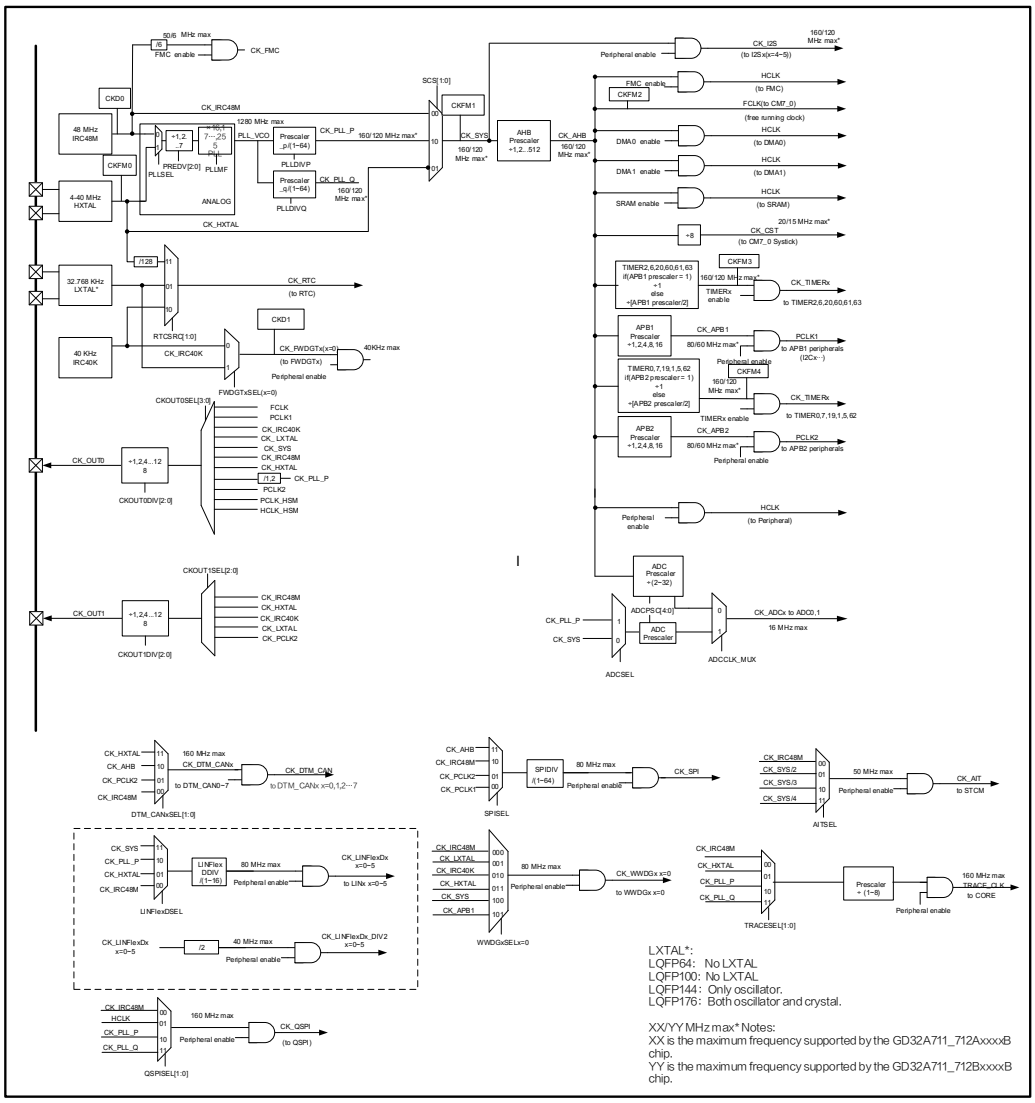
5.2. Clock control unit (CCTL)

5.2.1. Overview

The clock control unit provides a range of frequencies and clock functions. These include an Internal 48 MHz RC oscillator (IRC48M), a High Speed Crystal Oscillator (HXTAL), an Internal 40KHz RC oscillator (IRC40K), a Low Speed Crystal Oscillator (LXTAL), a Phase Lock Loop (PLL), a HXTAL clock monitor, clock prescalers, clock multiplexers, clock output, clock detect, and clock gating circuitry.

The clocks of the AHB, APB, Cortex-M7_0 is derived from the system clock (CK_SYS) which can source from the IRC48M, HXTAL or PLL. The maximum operating frequency of the system clock (CK_SYS) can be up to 160 MHz. The Free Watchdog Timer (FWDGT) has independent clock source (IRC40K), and Real Time Clock (RTC) use the IRC40K, LXTAL or HXTAL/128 as its clock source.

Figure 5-4. Clock tree



The frequency of AHB, APB2 and the APB1 domains can be configured by each prescaler. The maximum frequency of the AHB, APB2 and APB1 domains is 160 / 80 / 80 MHz. The Cortex System Timer (SysTick) external clock is clocked with the AHB clock (HCLK) divided by 8. The systick can work either with this clock or with the AHB clock (HCLK), configurable in the systick control and status register.

The ADCx(x=0~1) are clocked by the clock of CK_AHB or CK_PLL_P or CK_SYS divided by 2~32 selected by ADCPSC[4:0] bits in configuration register 2 (RCU_CFG2).

The DTM_CANx(x=0~7) is clocked by IRC48M clock or CK_APB2 or CK_AHB or HXTAL clock, which selected by DTM_CANxSEL(x=0~7) bits in configuration register 2 (RCU_CFG2) and configuration register 3 (RCU_CFG3).

The RTC is clocked by LXTAL clock or IRC40K clock or HXTAL clock divided by 128 which select by RTCSRC[1:0] bits in backup domain control register (RCU_BDCTL).

The FWDGT0 is clocked by CK_IRC40K or CK_LXTAL, which selected by FWDGT0SEL bits

in configuration register 1 (RCU_CFG1).

If the APB prescaler is 1, the TIMER clock frequencies are set to AHB frequency divide by 1. Otherwise, they are set to the AHB frequency divide by half of APB prescaler.

The TRACE clock source is configured with the TRACESEL[1:0] bit of the RCU_CFG3 register to select CK_HXTAL, CK_PLL_P, CK_PLL_Q, or CK_IRC48M, and the frequency division coefficients from 1 to 8 are configured with the TRACEDIV[2:0] bit of the RCU_CFG3 register.

The WWDGT0 clock source is configured with the WWDGT0SEL[2:0] bit of the RCU_CFG3 register to select CK_SYS, CK_IRC40K, CK_HXTAL, CK_APB1, CK_LXTAL, or CK_IRC48M.

The LINFlexD clock source is configured with the LINFLEXDSEL[1:0] bit of the RCU_CFG3 register to select CK_SYS, CK_PLL_P, CK_HXTAL, CK_IRC48M. The frequency division coefficients of LINFlexD clock source from 1 to 15 are configured with the LINFLEXDDIV[3:0] bit of the RCU_CFG3 register.

The AIT clock source is configured with the AITSEL[1:0] bit of the RCU_CFG1 register to select CK_IRC48M, CK_SYS/2, CK_SYS/3, or CK_SYS/4.

5.2.2. Characteristics

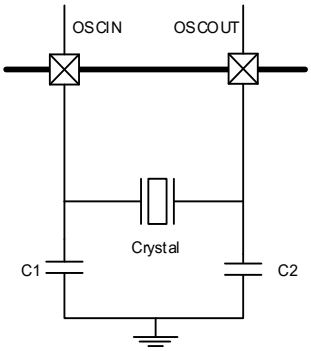
- 4 to 40 MHz High speed crystal oscillator (HXTAL).
- Internal 48 MHz RC oscillator (IRC48M).
- 32.768 KHz Low speed crystal oscillator (LXTAL).
- Internal 40 KHz RC oscillator (IRC40K).
- PLL clock source can be HXTAL or IRC48M.
- HXTAL clock monitor.
- HXTAL clock frequency monitor.

5.2.3. Function overview

High Speed Crystal Oscillator (HXTAL)

The high speed crystal oscillator (HXTAL), which has a frequency from 4 to 40 MHz, produces a highly accurate clock source for use as the system clock. A crystal with a specific frequency must be connected and located close to the two HXTAL pins. The external resistor and capacitor components connected to the crystal are necessary for proper oscillation.

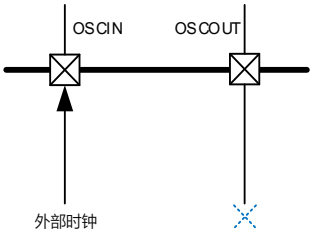
Figure 5-5. HXTAL clock source



The HXTAL crystal oscillator can first be selected using the HXTALSCAL bit in the control register, RCU_CTL, and then be switched on or off using the HXTALEN bit in the control register, RCU_CTL. The HXTALSTB flag in the control register, RCU_CTL indicates if the high-speed external crystal oscillator is stable. When the HXTAL is powered up, it will not be released for use until this HXTALSTB bit is set by the hardware. This specific delay period is known as the oscillator "Start-up time". As the HXTAL becomes stable, an interrupt will be generated if the related interrupt enables bit HXTALSTBIE in the interrupt register RCU_INT is set. At this point the HXTAL clock can be used directly as the system clock source or the PLL input clock.

Select external clock bypass mode by setting the HXTALBPS and HXTALEN bits in the control register, RCU_CTL, as shown in ***Figure 5-6. HXTAL clock source in bypass mode***. The CK_HXTAL is equal to the external clock which drives the OSCIN pin.

Figure 5-6. HXTAL clock source in bypass mode



Select the HXTAL frequency scale by using the HXTALSCAL[1:0] bit in the control register, RCU_CTL.

Internal 48 MHz RC Oscillator (IRC48M)

The internal 48 MHz RC oscillator, IRC48M, has a fixed frequency of 48 MHz and is the default clock source selection for the CPU when the device is powered up. The IRC48M oscillator provides a lower cost type clock source as no external components are required. The IRC48M RC oscillator can be switched on or off using the STBIRC48MCTL bit in the PMU_CFG register only in standby mode.The IRC48MSTB flag in the low voltage control and status register, PMU_LVCS is used to indicate if the internal 48 MHz RC oscillator is stable. The start-up time of the IRC48M oscillator is shorter than the HXTAL crystal oscillator. The

IRC48M clock can also be used as the PLL input clock.

The frequency accuracy of the IRC48M can be calibrated by the manufacturer, but its operating frequency is still less accurate than HXTAL. The application requirements, environment and cost will determine which oscillator type is selected.

Phase Locked Loop (PLL)

The internal phase locked loop, PLL, can provide 640~1280 MHz clock output which is 16 ~255 multiples of a fundamental reference frequency of 4 ~ 48 MHz. When configuring the PLL, it is strictly prohibited to exceed the upper and lower limits of the PLL_VCO clock, and the system clock must not exceed the maximum frequency range.

Note: The clock range after PLL pre-division (PREDV[2:0]) must be 8 ~ 40MHz.

For each PLL, When PLLFRAEN bit is '1' , PLLFRAN value is not '0' in RCU_PLLFRA register,and SSCGBYP bit is '1' , the PLLx is in fraction mode, for example:

$$CK_PLLVCO = CK_PLLVCOSRC * \left(PLLN + \frac{PLLFRAN + 0.5}{2^{14}}\right) \quad (5\text{-}1)$$

When PLLFRAEN bit is '0', the PLL is in integer mode, for example:

$$CK_PLLVCO = CK_PLLVCOSRC * PLLN \quad (5\text{-}2)$$

The PLL can be switched on or off by using the PLLEN bit in the control register, RCU_CTL. The PLLSTB flag in the control register, RCU_CTL will indicate if the PLL clock is stable. An interrupt can be generated if the related interrupt enable bit, PLLSTBIE, in the interrupt register, RCU_INT, is set as the PLL becomes stable. In addition, PLLLOLCTL[1:0] can be used to configure the detection accuracy of PLL lock or loss of lock. Before modifying any PLL configuration or the input clock, you must set PLLEN to 0 to disable the PLL. After disabling the PLL, you must maintain this state for at least 5 microseconds before re-enabling the PLL.

When PLLFRAEN bit is '1' and SSCGBYP bit is '0' , the PLLx is in spread spectrum modulation mode.

Note: When HXTAL is used as the source for PLL, if the PLL is not used as the system clock, the PLL will not be automatically disabled in case of an HXTAL failure. Moreover, by outputting the PLL through CK_OUT pin, it is still possible to detect frequencies in the KHz range.

Low Speed Crystal Oscillator (LXTAL)

The low speed crystal or ceramic resonator oscillator, which has a frequency of 32,768 Hz, produces a low power but highly accurate clock source for the RTC circuit. The LXTAL oscillator can be switched on or off using the LXTALEN bit in the backup domain control register (RCU_BDCTL). The LXTALSTB flag in the backup domain control register (RCU_BDCTL) will indicate if the LXTAL clock is stable. An interrupt can be generated if the

related interrupt enable bit, LXTALSTBIE, in the interrupt register RCU_INT is set when the LXTAL becomes stable.

Select external clock bypass mode by setting the LXTALBPS and LXTALEN bits in the backup domain control register (RCU_BDCTL). The CK_LXTAL is equal to the external clock which drives the OSC32IN pin.

Supported LXTAL types for different chip packages:

LQFP64: Cannot connect a crystal oscillator due to the absence of LXTAL pins.

LQFP100: Cannot connect a crystal oscillator due to the absence of LXTAL pins.

LQFP144: Only supports oscillator.

LQFP176: Supports both crystal and oscillator.

Internal 40 KHz RC Oscillator (IRC40K)

The internal 40 KHz RC oscillator has a frequency of about 40 KHz and is a low power clock source for RTC or FWDGT. The IRC40K offers a low cost clock source as no external components are required. The IRC40K RC oscillator can be switched off by using the STBIRC40KCTL bit in the PMU_CFG register only in standby mode.The IRC40KSTB flag in PMU_LVCS register will indicate if the IRC40K clock is stable.

System Clock (CK_SYS) Selection

After the system reset, the default CK_SYS source will be IRC48M and can be switched to HXTAL, PLL or IRC48M by changing the system clock switch bits, SCS[1:0], in the configuration register 0, RCU_CFG0. When the SCS value is changed, the CK_SYS will continue to operate using the original clock source until the target clock source is stable. When a clock source is used directly by the PLL, it is not possible to stop it.

PLL Clock Monitor (PLLM)

The PLL clock monitor function is enabled automatically when PLLEN is set in RCU_CTL.This function should be enabled after the PLL start-up delay and disabled when the PLL is stopped or slow down. Once the PLL failure is detected, Loss-of-PLL reset will generate. System reset will caused by Loss-of-PLL.

The PLL will be automatically disabled. If the PLL is selected as the clock source of CK_SYS, the PLL failure will force the CK_SYS source to IRC48M and the PLL will be disabled automatically.

HXTAL Clock Monitor (CKM)

The HXTAL clock monitor function is enabled by the HXTAL clock monitor enable bit, CKMEN, in the control register, RCU_CTL. This function should be enabled after the HXTAL start-up delay and disabled when the HXTAL is stopped. Once the HXTAL failure is detected, HXTAL clock stuck interrupt will generate or an NMI Interrupt may generate decided by the CKNMIIE0

bit.

The HXTAL clock stuck flag, CKMIF, in the interrupt register, RCU_INT, will be set and the HXTAL failure event will be generated. The HXTAL will be automatically disabled. This failure interrupt is sent to the Cortex®-M23, Cortex®-M7_0. If the HXTAL is selected as the clock source of CK_SYS or PLL, the HXTAL failure will force the CK_SYS source to IRC48M and the PLL will not be disabled automatically.

Clock Output Capability(CK_OUT)

The clock output capability is ranging from 32 KHz to 100 MHz. There are several clock signals can be selected via the CK_OUT0 Clock Source Selection bits, CKOUT0SEL[3:0], in the configuration register 1 (RCU_CFG1). The corresponding GPIO pin should be configured in the properly Alternate Function I/O (AFIO) mode to output the selected clock signal.

Table 5-5. CK_OUT0 Clock source select

Clock Source Selection bits	Clock Source
0000	FCLK
0001	PCLK1
0010	Internal 40K RC oscillator clock selected
0011	External low speed oscillator clock selected
0100	System clock selected
0101	Internal 48MHz RC oscillator clock selected
0110	External high speed oscillator clock selected
0111	CK_PLL_P
1000	reserved
1001	PCLK2
1010	PCLK from HSM
1011	HCLK from HSM

The CK_OUT0 frequency can be reduced by a configurable binary divider, controlled by the CKOUT0DIV[2:0] bits, in the configuration register 1 (RCU_CFG1).CK_OUT0 is usually used in run mode.

There are several clock signals can be selected via the CK_OUT1 Clock Source Selection bits, CKOUT1SEL[3:0], in the Configuration register 0 (RCU_CFG0). The corresponding GPIO pin should be configured in the properly Alternate Function I/O (AFIO) mode to output the selected clock signal.

Table 5-6. CK_OUT1 Clock source select

Clock Source Selection bits	Clock Source
000	Internal 48MHz RC oscillator clock selected
001	External high speed oscillator clock selected
010	Internal 40K RC oscillator clock selected
011	External low speed oscillator clock selected
100	System clock selected

Clock Source Selection bits	Clock Source
101	reserved
110	reserved
111	reserved

The CK_OUT1 frequency can be reduced by a configurable binary divider, controlled by the CKOUT1DIV[2:0] bits, in the configuration register 0 (RCU_CFG0).

CK_OUT1 is often used in standby mode.

Clock Detect (CKD)

For GD32A711/A712, the CKDx module uses HXTAL as a reference clock to monitor other clocks, and the monitoring clock of CKDx(x=0,1) can reference the clock tree in ***Figure 5-4. Clock tree***.

The CKDx module needs software to ensure that HXTALEN and the monitored clock are enabled. If the clock is not turned on, the monitoring function cannot work normally. The CKDx module only monitors the presence or absence of clock, and has little requirement on the sensitivity of clock monitoring.

The CKDx module supports independent interrupt enablement, and each clock failure can independently trigger an interrupt

After the clock detection function is enabled, the clock detection window is continuous. In each detection window, the clock fail signal is continuously set. If the detected clock runs normally during the current window, the clock fail flag is cleared; if the detected clock does not run in a certain detection window, the clock fail flag is set further.

If the IRC48MFR bit is set in RCU_CFG1, the failure of IRC48M will set FFIO0 and FFIO1 pins to High-Z state (refer to AFIO for more information).

Clock frequency monitor (CKFM)

When the chip is running, if the clock frequency is too high or too low, the timing, protocol, and performance will be faulty. CKFMx uses IRC48M as a reference clock to monitor whether other clocks are within the preset frequency range of the user to ensure the continuous and stable operation of the monitored clock.The monitoring clock of CKFMx (x=0..4) can refer to the clock tree in ***Figure 5-4. Clock tree***.

Clock frequency check

After CKFM starts clock monitoring, the monitored clock counter and the reference clock counter are counted at the same time (the clock check window is configured by REF_CNT[15:0] bits in RCU_CKFMxRCCFG register), and at the end of each clock check window, the count value of the monitored clock is compared with the programmed threshold.

When the count value of the monitored clock is higher than HFMAX, an FHMAX event is

generated and the corresponding clock failed system reset or interrupt is generated.

When the count value of the monitored clock is lower than LFMAX, an FLMIN event is generated and the corresponding clock failed system reset or interrupt is generated.

Note:

For CKFM0, when 'FHMAXSRE' bit or 'FLMINSRE' bit is set, and 'FHMAXIE' bit or 'FLMINIE' bit is not set, the FLMIN or FHMAX event will cause Loss of HXTAL system reset.When 'FHMAXSRE' bit or 'FLMINSRE' bit is not set, and 'FHMAXIE' bit or 'FLMINIE' bit is set, the FLMIN or FHMAX event will cause RCU clock fail interrupt. When the 'FHMAXSRE' bit or 'FLMINSRE' bit is set, and the 'FHMAXIE' bit or 'FLMINIE' bit is also set, an FLMIN or FHMAX event will trigger an RCU clock failure interrupt. Additionally, 'FHMAXSRE' bit and 'FLMINSRE' bit only exists in CKFM0.

Table 5-7. CKFM0 interrupt/reset configuration description

Reset Enable	Interrupt Enable	Event Type
FHMAXSRE/FLMINSRE	FHMAXIE/FLMINIE	
1	0	System Reset
0	1	Interrupt
1	1	Interrupt
0	0	No Event

For CKFM1, when'FHMAXIE' bit or 'FLMINIE' bit is set, the FLMIN or FHMAX event will cause System clock fail system reset in CKFM1.

For CKFM2, when'FHMAXIE' bit or 'FLMINIE' bit is set, the FLMIN or FHMAX event will cause Core clock fail system reset in CKFM2.

For CKFM3, when'FHMAXIE' bit or 'FLMINIE' bit is set, the FLMIN or FHMAX event will cause RCU clock fail interrupt.

For CKFM4, when'FHMAXIE' bit or 'FLMINIE' bit is set, the FLMIN or FHMAX event will cause RCU clock fail interrupt.

Clock frequency check timeout

If the following two conditions occur during the clock monitoring process, CKFM will wait for the monitored clock to run normally by extending the check window period. If the monitored clock has not run normally after an extended period, FLMIN events will be generated. The CKFM waiting time varies under different conditions:

1. After CKFM monitoring is enabled, the monitored clock does not run.

FLMIN wait time = $\left(2 \times \text{RCU_CKFMxRCCFG[REF_CNT[15:0]]+12}\right)$ reference clock cycles+15 bus clock cycles

2. After CKFM monitoring is enabled, the monitored clock stops running during the check window period.

FLMIN wait time = $\left(2 \times \text{RCU_CKFMxRCCFG[REF_CNT[15:0]]+2}\right)$ reference clock cycles+3 bus clock cycles

5.3. Reset control unit registers definition

RCTL base address: 0x4900 3C00

Table 5-8. RCTL register overview

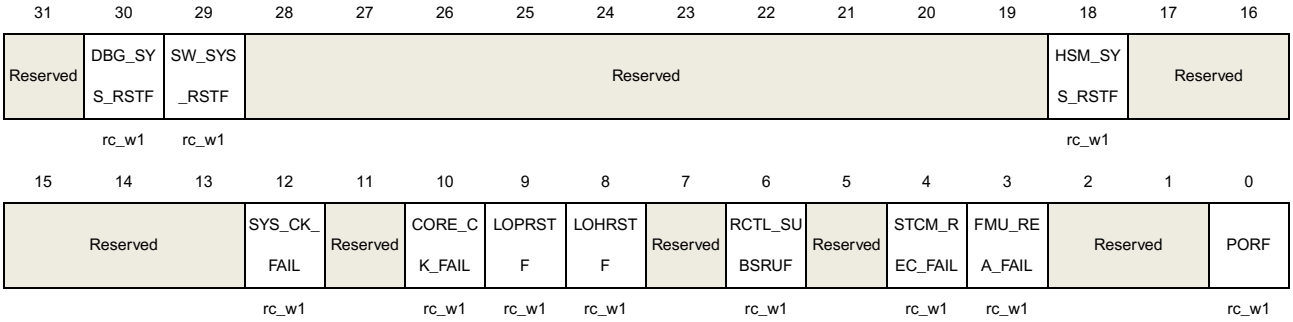
Register name	Symbol	Address offset	Reset value
System Reset Status register	RCU_SYSRSTSTAT	0x00	0x0000 0001
Bus control register	RCU_BUSCTL	0x04	0x0000 0000
Subsystem/external reset status register	RCU_SUBSERSTAT	0x08	0x0000 0000
Subsystem reset control register	RCU_SUBSRCTL	0x0C	0x0000 0000
Subsystem reset pin assert control register	RCU_SUBSRPACTL	0x10	0x0000 0000
Subsystem reset mode upgrade counter register	RCU_SUBSRMUCNT	0x14	0x0000 0000
Subsystem reset mode upgrade threshold register	RCU_SUBSRMUTH	0x18	0x0000 000F
System reset mode upgrade threshold register	RCU_SYSMRUTH	0x1C	0x0000 0000
External reset pin assert configuration register	RCU_ERPACFG	0x20	0x0000 0000
Reset during standby mode status register	RCU_RDSMSTAT	0x24	0x0000 0000
Subsystem reset timeout register	RCU_SUBSRTO	0x28	0x0000 0001
Low Power Debug Status Register	RCU_LPDBGSTAT	0x2C	0x0000 0000
AHB reset register	RCU_AHBRST	0x30	0x0000 0000
APB1 reset register	RCU_APB1RST	0x34	0x0000 0000
APB2 reset register	RCU_APB2RST	0x38	0x0000 0000
APB2 reset additional register	RCU_APB2RST_ADD	0x3C	0x0000 0000
Power-on reset during POR watchdog overflows status register 0	RCU_PORSTAT0	0x40	0x0000 0000
Power-on reset during POR watchdog overflows status register 1	RCU_PORSTAT1	0x44	0x0000 0000
Power-on reset during POR watchdog overflows status register 2	RCU_PORSTAT2	0x48	0x0000 0000
Power-on reset during POR watchdog overflows status register 3	RCU_PORSTAT3	0x4C	0x0000 0000
Triple modular redundancy error status	RCU_TMRERRSTAT	0x50	0x0055 1555
System reset upgrade counter register	RCU_SYSRSTUCNT	0x54	0x0000 0000
Subsystem reset counter register	RCU_SUBSRSTCNT	0x58	0x0000 0000
System reset counter register	RCU_SYSRSTCNT	0x5C	0x0000 0001
Subsystem reset all disable register	RCU_SUBSRSTAD	0x60	0x0000 0000

5.3.1. System Reset Status register (RCU_SYSRSTSTAT)

Address offset: 0x00

Reset value: 0x0000 0001

All bits are reset only on POR.



Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30	DBG_SYS_RSTF	'System' reset flag DBG_SYS_RSTF 0: 'System' reset event DBG_SYS_RST has not been triggered since the last clear or power-on reset assertion. 1: 'System' reset event DBG_SYS_RST has been triggered.
29	SW_SYS_RSTF	'System' reset flag SW_SYS_RSTF 0: 'System' reset event SW_SYS_RST has not been triggered since the last clear or power-on reset assertion. 1: 'System' reset event SW_SYS_RST has been triggered.
28:19	Reserved	Must be kept at reset value.
18	HSM_SYS_RSTF	'System' reset flag HSM_SYS_RSTF 0: 'System' reset event HSM_SYS_RST has not been triggered since the last clear or power-on reset assertion. 1: 'System' reset event HSM_SYS_RST has been triggered.
17:13	Reserved	Must be kept at reset value.
12	SYS_CK_FAIL	'System' reset flag SYS_CK_FAIL 0: 'System' reset event SYS_CK_FAIL has not been triggered since the last clear or power-on reset assertion. 1: 'System' reset event SYS_CK_FAIL has been triggered.
11	Reserved	Must be kept at reset value.
10	CORE_CK_FAIL	'System' reset flag CORE_CK_FAIL 0: 'System' reset event CORE_CK_FAIL has not been triggered since the last clear or power-on reset assertion. 1: 'System' reset event CORE_CK_FAIL has been triggered.

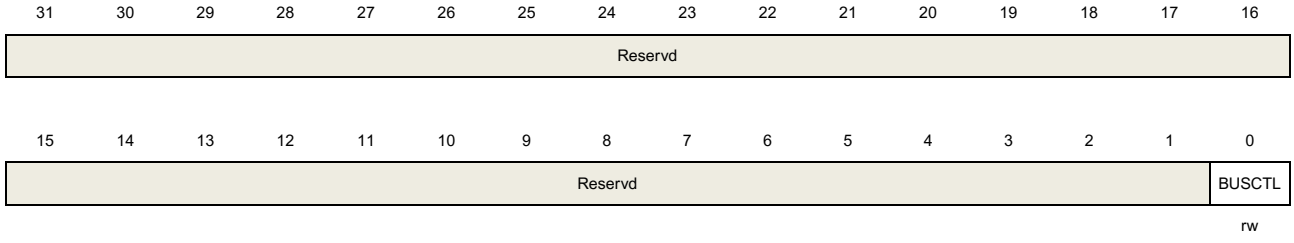
9	LOPRSTF	Lost of PLL error reset flag Set by hardware when a Lost of PLL error reset generated. 0: No lost of PLL error reset generated 1: Lost of PLL error reset generated
8	LOHRSTF	Lost of HXTAL error reset flag Set by hardware when a Lost of HXTAL Error reset generated. 0: No lost of HXTAL error reset generated 1: Lost of HXTAL error reset generated
7	Reserved	Must be kept at reset value.
6	RCTL_SUBSRUF	'System' reset flag RCTL_SUBSRUF 0: 'System' reset event RCTL_SUBSRUF has not been triggered since the last clear or power-on reset assertion. 1: 'System' reset event RCTL_SUBSRUF has been triggered.
5	Reserved	Must be kept at reset value.
4	STCM_REC_FAIL	'System' reset flag STCM_REC_FAIL 0: 'System' reset event STCM_REC_FAIL has not been triggered since the last clear or power-on reset assertion. 1: 'System' reset event STCM_REC_FAIL has been triggered.
3	FMU_REA_FAIL	'System' reset flag FMU_REA_FAIL 0: 'System' reset event FMU_REA_FAIL has not been triggered since the last clear or power-on reset assertion. 1: 'System' reset event FMU_REA_FAIL has been triggered.
2:1	Reserved	Must be kept at reset value.
0	PORF	Power-on reset flag 0: No POR event has occurred since the last clear. 1: A POR event has occurred

5.3.2. Bus control register (RCU_BUSCTL)

Address offset: 0x04

Reset value: 0x0000 0000

All bits are reset only on POR.



Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	BUSCTL	This bit is 1, if a subsystem reset occurs in the SUBSRST1 stage, bus access is forbidden until the reset is released.(rw) 0: Bus access is allowed. 1: Halt the bus access.

5.3.3. Subsystem/external reset status register (RCU_SUBSERSTAT)

Address offset: 0x08
Reset value: 0x0000 0000

All bits are reset only on POR.

Note: If any bit fields other than the PORF bit are set in the RCU_SYSRSTSTAT register, all bits in this register must be ignored. In this case, it indicates that a system reset occurred while a subsystem reset source was active. The subsystem reset source is in an unreliable state.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OBL_RSTF	DBG_RSTF	SW_SUBS_RSTF	Reserved								HSM_APP_RSTF	Reserved			HSM_WDGT_RSTF
rc_w1	rc_w1	rc_w1									rc_w1				rc_w1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					WWDGT0_RSTF	SWRSTF	Reserved		FWDGT0_RSTF	Reserved	ST_FINISH	FMU_RSTF	OBSTDBY_RSTF	Reserved	EXRF
					rc_w1	rc_w1			rc_w1		rc_w1	rc_w1	rc_w1		rc_w1

Bits	Fields	Descriptions
31	OBL_RSTF	Option byte loader reset flag Set by hardware when an option byte loader generated. 0: No option byte loader reset generated 1: Option byte loader reset generated
30	DBG_RSTF	Debug reset or interrupt flag DBG_RSTF 0: Subsystem reset or interrupt event DBG_RST has not been triggered since the last clear or power-on reset assertion 1: Subsystem reset or interrupt event DBG_RST has been triggered.
29	SW_SUBS_RSTF	Subsystem reset flag SW_SUBS_RSTF 0: Subsystem reset event SW_SUBS_RST has not been triggered since the last clear or power-on reset assertion 1: Subsystem reset event SW_SUBS_RST has been triggered.
28:21	Reserved	Must be kept at reset value.

20	HSM_APP_RSTF	Subsystem reset flag HSM_APP_RSTF 0: Subsystem reset event HSM_APP_RST has not been triggered since the last clear or power-on reset assertion 1: Subsystem reset event HSM_APP_RST has been triggered.
19:17	Reserved	Must be kept at reset value.
16	HSM_WDGT_RSTF	Subsystem reset flag HSM_WDGT_RSTF 0: Subsystem reset event HSM_WDGT_RST has not been triggered since the last clear or power-on reset assertion 1: Subsystem reset event HSM_WDGT_RST has been triggered.
15:11	Reserved	Must be kept at reset value.
10	WWDGT0_RSTF	Subsystem reset or interrupt flag WWDGT0_RSTF 0: Subsystem reset or interrupt event WWDGT0_RST has not been triggered since the last clear or power-on reset assertion 1: Subsystem reset or interrupt event WWDGT0_RST has been triggered.
9	SWRSTF	Software reset flag Set by hardware when a software reset generated. 0: No software reset generated 1: Software reset generated
8:7	Reserved	Must be kept at reset value.
6	FWDGT0_RSTF	Subsystem reset flag FWDGT0_RSTF 0: Subsystem reset event FWDGT0_RST has not been triggered since the last clear or power-on reset assertion 1: Subsystem reset event FWDGT0_RST has been triggered.
5	Reserved	Must be kept at reset value.
4	ST_FINISH	Subsystem reset flag ST_FINISH 0: Subsystem reset event ST_FINISH has not been triggered since the last clear or power-on reset assertion 1: Subsystem reset event ST_FINISH has been triggered.
3	FMU_RSTF	Subsystem reset or interrupt flag FMU_RSTF 0: Subsystem reset or interrupt event FMU_RST has not been triggered since the last clear or power-on reset assertion 1: Subsystem reset or interrupt event FMU_RST has been triggered.
2	OBSTDBY_RSTF	Subsystem reset flag OBSTDBY_RSTF 0: Subsystem reset event OBSTDBY_RST has not been triggered since the last clear or power-on reset assertion 1: Subsystem reset event OBSTDBY_RST has been triggered.
1	Reserved	Must be kept at reset value.
0	EXRF	External reset flag EXRF.

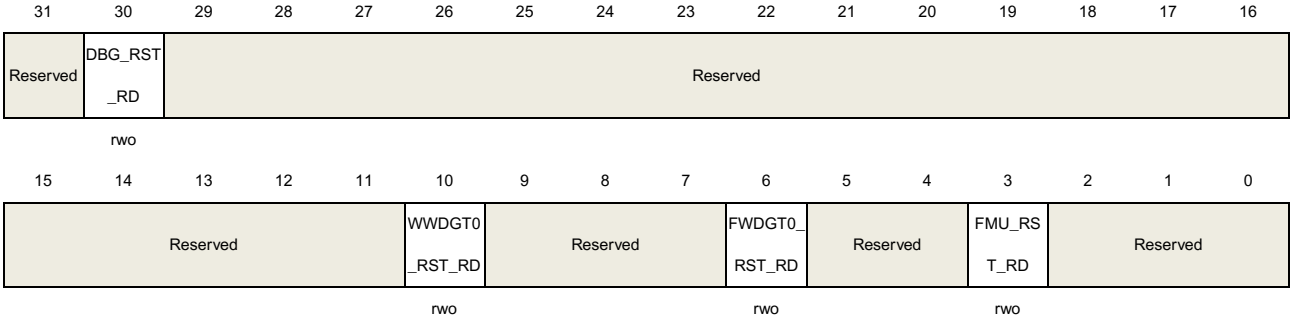
External reset is a source of system reset
0: External reset has not been triggered since the last clear or power-on reset assertion
1: External reset event has been triggered.

5.3.4. Subsystem reset control register (RCU_SUBSRCTL)

Address offset: 0x0C
Reset value: 0x0000 0000

Each byte can be written to only once after a system or POR reset and this register is reset only on POR reset and any system reset.

This register has to be accessed by word(32-bit).



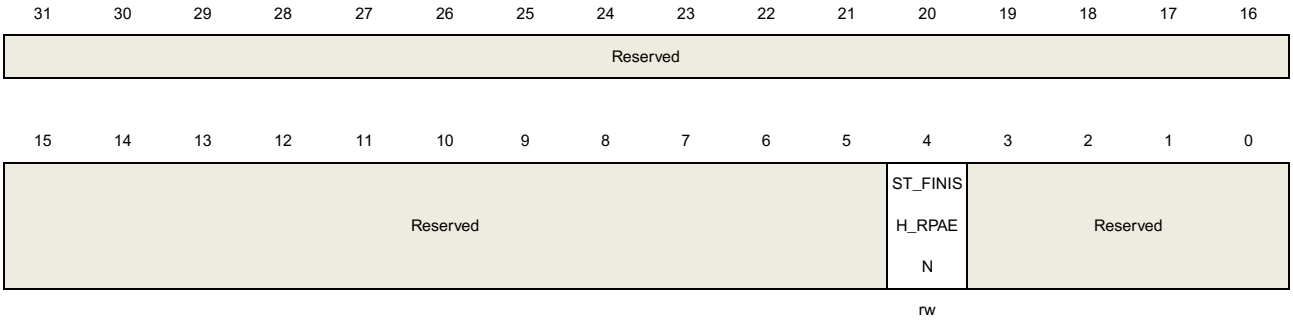
Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30	DBG_RST_RD	DBG_RST disable control 0: Subsystem reset event DBG_RST triggers a reset sequence. 1: Subsystem reset event DBG_RST generates an interrupt request.
29:11	Reserved	Must be kept at reset value.
10	WWDGT0_RST_RD	WWDGT0_RST disable control 0: Subsystem reset event WWDGT0_RST triggers a reset sequence. 1: Subsystem reset event WWDGT0_RST generates an interrupt request.
9:7	Reserved	Must be kept at reset value.
6	FWDGT0_RST_RD	FWDGT0_RST_RD disable control 0: Subsystem reset event FWDGT0_RST triggers a reset sequence. 1: Subsystem reset event FWDGT0_RST generates an interrupt request.
5:4	Reserved	Must be kept at reset value.
3	FMU_RST_RD	FMU_RST Disable Control 0b - Subsystem reset event FMU_RST triggers a reset sequence. 1b - Subsystem reset event FMU_RST generates an interrupt request

2:0	Reserved	Must be kept at reset value.

5.3.5. Subsystem reset pin assert control register (RCU_SUBSRPACTL)

Address offset: 0x10
Reset value: 0x0000 0000

All bits are reset only on POR or any system reset. Except for bit 4, the other bits are read-only.

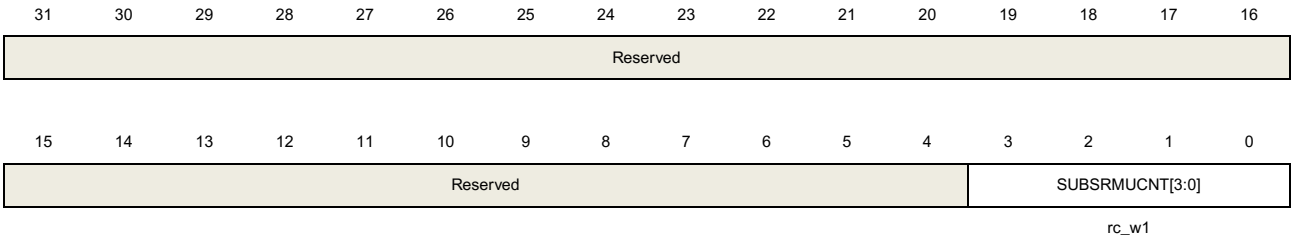


Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
4	ST_FINISH_RPAEN	Reset pin assert enable for subsystem reset ST_FINISH 0: External reset is asserted on a subsystem reset ST_FINISH event if the reset is enabled 1: External reset is not asserted on a subsystem reset ST_FINISH event
3:0	Reserved	Must be kept at reset value.

5.3.6. Subsystem reset mode upgrade counter register (RCU_SUBSRMUCNT)

Address offset: 0x14
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3:0	SUBSRMUCNT[3:0]	Subsystem reset mode upgrade counter

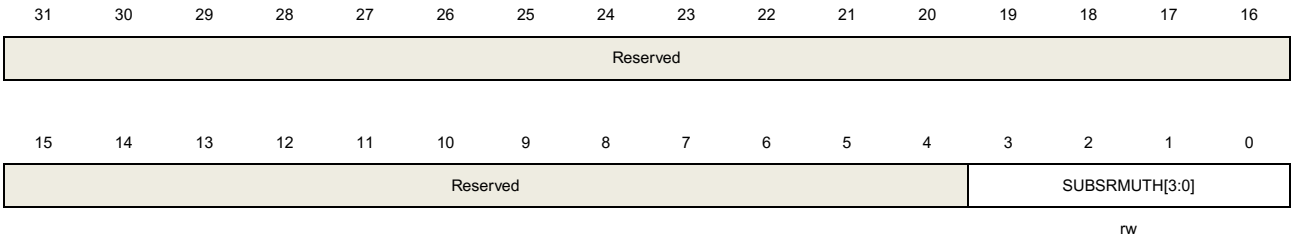
Provide the value of subsystem reset upgrade counter. When subsystem reset upgrade function is disabled or resetting the RCU_SUBSRMUTH threshold value, the SUBSRMUCNT[3:0] are cleared

5.3.7. Subsystem reset mode upgrade threshold register (RCU_SUBSRMUTH)

Address offset: 0x18

Reset value: 0x0000 000F

This register has to be accessed by word(32-bit).



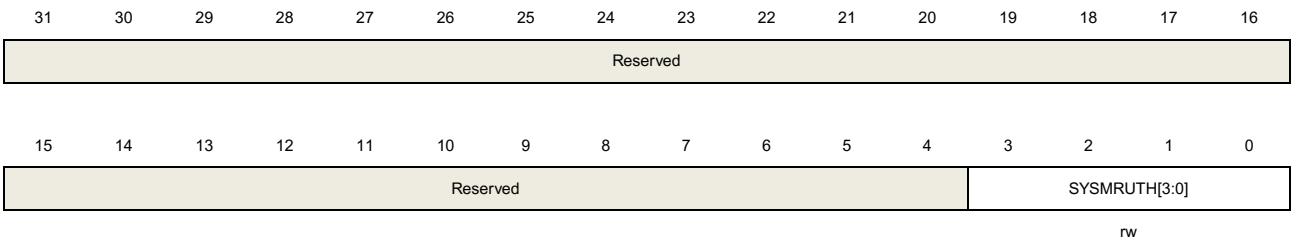
Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3:0	SUBSRMUTH[3:0]	Subsystem reset mode upgrade threshold If the value of SUBSRMUTH[3:0] is 0, the subsystem reset upgrade is disabled. Any other value is the number of subsystem resets that causes a system reset if the SUBSRMUTH register isn't written to beforehand. Configuring this register will reset the SUBSRMUCNT.

5.3.8. System reset mode upgrade threshold register (RCU_SYSMRUTH)

Address offset: 0x1C

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



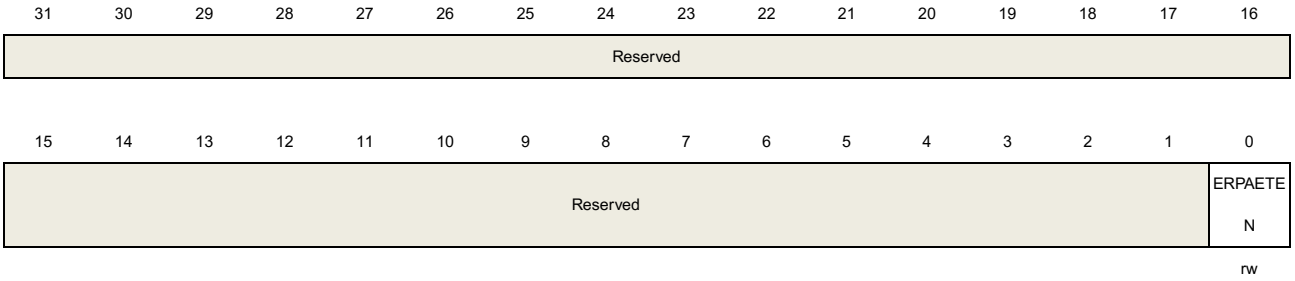
Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3:0	SYSMRUTH[3:0]	System reset mode upgrade threshold If the value of SYSMRUTH[3:0] is 0, the system reset upgrade function is

disabled. Any other value is the number of system resets which keeps the chip in the reset state until the next POR triggers a new reset sequence if the SYSMRUTH register isn't written to beforehand. Configuring this register will reset the RCU_SYSRSTUCNT.

5.3.9. External reset pin assert configuration register (RCU_ERPACFG)

Address offset: 0x20
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	ERPAETEN	External reset pin assert enable 0: No change 1: External reset pin is asserted **Note:** ERPAETEN being set to 1 will result in most chip pins being in a safe state. Software must use the ERPAETEN bit for this purpose only as part of the main reset domain self-test entry procedure. Using it at any other time can lead to unpredictable system behavior.

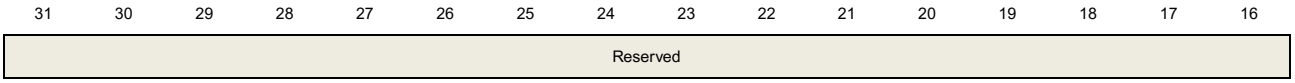
5.3.10. Reset during standby mode status register (RCU_RDSMSTAT)

Address offset: 0x24
Reset value: 0x0000 0000

Note: On exiting a reset sequence after standby mode exit, the software must perform a read operation on PMU_MODSTAT[PREMOD[1:0]] and RCU_RDSMSTAT register.

If any bit of the RCU_RDSMSTAT register is set, the software must ignore the status reported by PMU_MODSTAT[PREMOD[1:0]] register otherwise the status of PMU_MODSTAT [PREMOD[1:0]] register reports the device status.

If PMU_MODSTAT indicates that the last mode is RESET, perform reset and exit in the software. Otherwise, perform standby exit.



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														SUBSRESF	SYSRESF
														rc_w1	rc_w1

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	SUBSRESF	Subsystem reset event occurred flag in standby mode 0: No subsystem reset event occurred in standby mode. 1: Subsystem reset event occurred in standby mode
0	SYSRESF	System reset event occurred flag in standby mode 0: No System reset event occurred in standby mode. 1: System reset event occurred during standby mode

5.3.11. Subsystem reset timeout register (RCU_SUBSRTO)

Address offset: 0x28

Reset value: 0x0000 0001

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SUBSRTO[30:15]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SUBSRTO[14:0]															SUBSRTOEN
rw															rw

Bits	Fields	Descriptions
31:1	SUBSRTO[30:0]	Subsystem reset timer value These bits allows to override the timeout duration of the subsystem reset entry timer. The counter value is determined according to the internal RC oscillator clock cycles (typical 48 MHz).
0	SUBSRTOEN	Subsystem reset timer enable 0: Subsystem reset timer is disabled. 1: Subsystem reset timer is enabled

5.3.12. Low Power Debug Status Register (RCU_LPDBGSTAT)

Address offset: 0x2C

Reset value: 0x0000 0000

This register is used to enable the data collection function before entering low-power debug.

This register is reset during a subsystem reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															LPDBGF
															r

Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	LPDBGF	Low-Power debug flag 0: No low-power debug flag 1: Low-power debug flag

5.3.13. AHB reset register (RCU_AHBRST)

Address offset: 0x30
Reset value: 0x0000 0000

Before using the register to reset the module, the corresponding module should be deactivated. If the corresponding module cannot be deactivated, the corresponding module register should be restored to the power-on reset value and then the register should be used to reset the module.

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MTCRST	STCMRST	Reserved	IRMRST	Reserved		HSMIFRST	Reserved				CRCRST	Reserved	QSPIRST	Reserved	DMAMUXRST
rw	rw		rw			rw					rw		rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA1RST	DMA0RST	PNRST	PMRST	PLRST	PKRST	PJRST	PIRST	PHRST	PGRST	PFRST	PERST	PDRST	PCRST	PBRST	PARST
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31	MTCRST	MTC reset enable This bit is set and reset by software. 0: Disabled MTC reset 1: Enabled MTC reset
30	STCMRST	STCM reset enable

		This bit is set and reset by software. 0: Disabled STCM reset 1: Enabled STCM reset
29	Reserved	Must be kept at reset value.
28	IRMRST	IRM reset enable This bit is set and reset by software. 0: Disabled IRM reset 1: Enabled IRM reset
27:26	Reserved	Must be kept at reset value.
25	HSMIFRST	HSMIF reset enable This bit is set and reset by software. 0: Disabled HSMIF reset 1: Enabled HSMIF reset
24:21	Reserved	Must be kept at reset value.
20	CRCRST	CRC reset enable This bit is set and reset by software. 0: Disabled CRC reset 1: Enabled CRC reset
19	Reserved	Must be kept at reset value.
18	QSPIRST	QSPI reset enable This bit is set and reset by software. 0: Disabled QSPI reset 1: Enabled QSPI reset
17	Reserved	Must be kept at reset value.
16	DMAMUXRST	DMAMUX reset enable This bit is set and reset by software. 0: Disabled DMAMUX reset 1: Enabled DMAMUX reset
15	DMA1RST	DMA1 reset enable This bit is set and reset by software. 0: Disabled DMA1 reset 1: Enabled DMA1 reset
14	DMA0RST	DMA0 reset enable This bit is set and reset by software. 0: Disabled DMA0 reset 1: Enabled DMA0 reset
13	PNRST	GPIO port N reset enable This bit is set and reset by software.

		0: Disabled GPIO port N reset 1: Enabled GPIO port N reset
12	PMRST	GPIO port M reset enable This bit is set and reset by software. 0: Disabled GPIO port M reset 1: Enabled GPIO port M reset
11	PLRST	GPIO port L reset enable This bit is set and reset by software. 0: Disabled GPIO port L reset 1: Enabled GPIO port L reset
10	PKRST	GPIO port K reset enable This bit is set and reset by software. 0: Disabled GPIO port K reset 1: Enabled GPIO port K reset
9	PJRST	GPIO port J reset enable This bit is set and reset by software. 0: Disabled GPIO port J reset 1: Enabled GPIO port J reset
8	PIRST	GPIO port I reset enable This bit is set and reset by software. 0: Disabled GPIO port I reset 1: Enabled GPIO port I reset
7	PHRST	GPIO port H reset enable This bit is set and reset by software. 0: Disabled GPIO port H reset 1: Enabled GPIO port H reset
6	PGRST	GPIO port G reset enable This bit is set and reset by software. 0: Disabled GPIO port G reset 1: Enabled GPIO port G reset
5	PFRST	GPIO port F reset enable This bit is set and reset by software. 0: Disabled GPIO port F reset 1: Enabled GPIO port F reset
4	PERST	GPIO port E reset enable This bit is set and reset by software. 0: Disabled GPIO port E reset 1: Enabled GPIO port E reset
3	PDRST	GPIO port D reset enable

This bit is set and reset by software.
0: Disabled GPIO port D reset
1: Enabled GPIO port D reset

2	PCRST	GPIO port C reset enable This bit is set and reset by software. 0: Disabled GPIO port C reset 1: Enabled GPIO port C reset
1	PBRST	GPIO port B reset enable This bit is set and reset by software. 0: Disabled GPIO port B reset 1: Enabled GPIO port B reset
0	PARST	GPIO port A reset enable This bit is set and reset by software. 0: Disabled GPIO port A reset 1: Enabled GPIO port A reset

5.3.14. APB1 reset register (RCU_APB1RST)

Address offset: 0x34
Reset value: 0x0000 0000

Before using the register to reset the module, the corresponding module should be deactivated. If the corresponding module cannot be deactivated, the corresponding module register should be restored to the power-on reset value and then the register should be used to reset the module.

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		SPI3RST	SPI2RST	Reserved		DAC1RST	Reserved							TIMER63RST	TIMER61RST
		rw	rw			rw								rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	DAC0RST	Reserved	BKPRST	I2C1RST	I2C0RST	Reserved				FWDGT0RST	WWDGT0RST	TIMER6RST	TIMER60RST	TIMER2RST	TIMER20RST
	rw		rw	rw	rw					rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value
29	SPI3RST	SPI3 reset enable This bit is set and reset by software. 0: Disabled SPI3 reset 1: Enabled SPI3 reset

28	SPI2RST	SPI2 reset enable This bit is set and reset by software. 0: Disabled SPI2 reset 1: Enabled SPI2 reset
27:26	Reserved	Must be kept at reset value
25	DAC1RST	DAC1 reset enable This bit is set and reset by software. 0: Disabled DAC1 reset 1: Enabled DAC1 reset
24:18	Reserved	Must be kept at reset value
17	TIMER63RST	TIMER63 timer reset enable This bit is set and reset by software. 0: Disabled TIMER63 timer reset 1: Enabled TIMER63 timer reset
16	TIMER61RST	TIMER61 timer reset enable This bit is set and reset by software. 0: Disabled TIMER61 timer reset 1: Enabled TIMER61 timer reset
15	Reserved	Must be kept at reset value
14	DAC0RST	DAC0 reset enable This bit is set and reset by software. 0: Disabled DAC0 reset 1: Enabled DAC0 reset
13	Reserved	Must be kept at reset value
12	BKPRST	BKP reset enable This bit is set and reset by software. 0: Disabled BKP reset 1: Enabled BKP reset
11	I2C1RST	I2C1 reset enable This bit is set and reset by software. 0: Disabled I2C1 reset 1: Enabled I2C1 reset
10	I2C0RST	I2C0 reset enable This bit is set and reset by software. 0: Disabled I2C0 reset 1: Enabled I2C0 reset
9:6	Reserved	Must be kept at reset value
5	FWDGT0RST	Free watchdog timer reset enable

This bit is set and reset by software.
0: Disabled free watchdog timer reset
1: Enabled free watchdog timer reset

4	WWDGT0RST	Window watchdog timer reset enable This bit is set and reset by software. 0: Disabled window watchdog timer reset 1: Enabled window watchdog timer reset
3	TIMER6RST	TIMER6 timer reset enable This bit is set and reset by software. 0: Disabled TIMER6 timer reset 1: Enabled TIMER6 timer reset
2	TIMER60RST	TIMER60 reset enable This bit is set and reset by software. 0: Disabled TIMER60 timer reset 1: Enabled TIMER60 timer reset
1	TIMER2RST	TIMER2 reset enable This bit is set and reset by software. 0: Disabled TIMER2 timer reset 1: Enabled TIMER2 timer reset
0	TIMER20RST	TIMER20 reset enable This bit is set and reset by software. 0: Disabled TIMER20 timer reset 1: Enabled TIMER20 timer reset

5.3.15. APB2 reset register (RCU_APB2RST)

Address offset: 0x38
Reset value: 0x0000 0000

These register is located in RAM.

Before using the register to reset the module, the corresponding module should be deactivated. If the corresponding module cannot be deactivated, the corresponding module register should be restored to the power-on reset value and then the register should be used to reset the module.

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					CMP0RST	SPI1RST	Reserved	CMP1RST	TIMER62RST	TIMER5RST	TIMER1RST	TIMER19RST	Reserved	SPI5RST	SPI4RST
					rw	rw		rw	rw	rw	rw	rw		rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			TIMER7R	SPI0RST	TIMER0R	ADC1RS	ADC0RS	Reserved	LINFLEX	LINFLEX	LINFLEX	LINFLEX	LINFLEX	LINFLEX	SYSCFG

	ST		ST	T	T		D5RST	D4RST	D3RST	D2RST	D1RST	D0RST	RST
	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:27	Reserved	Must be kept at reset value
26	CMP0RST	Comparator0 reset enable This bit is set and reset by software. 0: Disabled comparator0 reset 1: Enabled comparator0 reset
25	SPI1RST	SPI1 reset enable This bit is set and reset by software. 0: Disabled SPI1 reset 1: Enabled SPI1 reset
24	Reserved	Must be kept at reset value
23	CMP1RST	Comparator1 reset enable This bit is set and reset by software. 0: Disabled comparator reset 1: Enabled comparator reset
22	TIMER62RST	TIMER62 timer reset enable This bit is set and reset by software. 0: Disabled TIMER62 timer reset 1: Enabled TIMER62 timer reset
21	TIMER5RST	TIMER5 timer reset enable This bit is set and reset by software. 0: Disabled TIMER5 timer reset 1: Enabled TIMER5 timer reset
20	TIMER1RST	TIMER1 timer reset enable This bit is set and reset by software. 0: Disabled TIMER1 timer reset 1: Enabled TIMER1 timer reset
19	TIMER19RST	TIMER19 timer reset enable This bit is set and reset by software. 0: Disabled TIMER19 timer reset 1: Enabled TIMER19 timer reset
18	Reserved	Must be kept at reset value
17	SPI5RST	SPI5 reset enable This bit is set and reset by software. 0: Disabled SPI5 reset 1: Enabled SPI5 reset

16	SPI4RST	SPI4 reset enable This bit is set and reset by software. 0: Disabled SPI4 reset 1: Enabled SPI4 reset
15:13	Reserved	Must be kept at reset value
12	TIMER7RST	TIMER7 timer reset enable This bit is set and reset by software. 0: Disabled TIMER7 timer reset 1: Enabled TIMER7 timer reset
11	SPI0RST	SPI0 reset enable This bit is set and reset by software. 0: Disabled SPI0 reset 1: Enabled SPI0 reset
10	TIMER0RST	TIMER0 timer reset enable This bit is set and reset by software. 0: Disabled TIMER0 timer reset 1: Enabled TIMER0 timer reset
9	ADC1RST	ADC1 interface reset enable This bit is set and reset by software. 0: Disabled ADC1 interface reset 1: Enabled ADC1 interface reset
8	ADC0RST	ADC interface reset enable This bit is set and reset by software. 0: Disabled ADC0 interface reset 1: Enabled ADC0 interface reset
7	Reserved	Must be kept at reset value
6	LINFLEXD5RST	LINFLEXD5 reset enable This bit is set and reset by software. 0: Disabled LINFLEXD5 reset 1: Enabled LINFLEXD5 reset
5	LINFLEXD4RST	LINFLEXD4 reset enable This bit is set and reset by software. 0: Disabled LINFLEXD4 reset 1: Enabled LINFLEXD4 reset
4	LINFLEXD3RST	LINFLEXD3 reset enable This bit is set and reset by software. 0: Disabled LINFLEXD3 reset 1: Enabled LINFLEXD3 reset
3	LINFLEXD2RST	LINFLEXD2 reset enable

		This bit is set and reset by software. 0: Disabled LINFLEXD2 reset 1: Enabled LINFLEXD2 reset
2	LINFLEXD1RST	LINFLEXD1 reset enable This bit is set and reset by software. 0: Disabled LINFLEXD1 reset 1: Enabled LINFLEXD1 reset
1	LINFLEXD0RST	LINFLEXD0 reset enable This bit is set and reset by software. 0: Disabled LINFLEXD0 reset 1: Enabled LINFLEXD0 reset
0	SYSCFGRST	System configuration reset enable This bit is set and reset by software. 0: Disabled system configuration reset 1: Enabled system configuration reset

5.3.16. APB2 reset additional register (RCU_APB2RST_ADD)

Address offset: 0x3C
Reset value: 0x0000 0000

Before using the register to reset the module, the corresponding module should be deactivated. If the corresponding module cannot be deactivated, the corresponding module register should be restored to the power-on reset value and then the register should be used to reset the module.

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					CAN2RST	CAN1RST	CAN0RST	Reserved							
					rw	rw	rw								

Bits	Fields	Descriptions
31:11	Reserved	Must be kept at reset value
10	CAN2RST	CAN2 cluster reset enable This bit is set and reset by software. 0: Disabled CAN2 cluster reset 1: Enabled CAN2 cluster reset
9	CAN1RST	CAN1 cluser reset enable This bit is set and reset by software.

		0: Disabled CAN1 cluser reset 1: Enabled CAN1 cluser reset
8	CAN0RST	CAN0 cluser reset enable This bit is set and reset by software. 0: Disabled CAN0 cluser reset 1: Enabled CAN0 cluser reset
7:0	Reserved	Must be kept at reset value

5.3.17. Power-on reset during POR watchdog overflows status register 0 (RCU_PORSTAT0)

Address offset: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PORWDG_OVF31	PORWDG_OVF30	PORWDG_OVF29	Reserved								PORWDG_OVF20	Reserved			
rc_w1	rc_w1	rc_w1									rc_w1				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				PORWDG_OVF11	PORWDG_OVF10	Reserved			PORWDG_OVF6	PORWDG_OVF5	PORWDG_OVF4	PORWDG_OVF3	PORWDG_OVF2	PORWDG_OVF1	PORWDG_OVF0
				rc_w1	rc_w1				rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions
31	PORWDG_OVF31	PORWDG_OVF31 captures RCTL reset event (if occurred) while the device is in standby mode. **Note:** This bit is applicable only for standby sequence monitoring. 0: RCTL reset event not detected or no reset event arrived in standby mode when PORWDG overflows. 1:RCTL reset event flag is detected in standby mode when PORWDG overflows.
30	PORWDG_OVF30	PORWDG_OVF30 captures the status of standby exit acknowledgement by PMU when POR_WDG overflows. **Note:** This bit is applicable only for standby sequence monitoring. 0: PMU has not acknowledged standby exit when PORWDG overflows. 1: PMU has acknowledged standby exit when PORWDG overflows.
29	PORWDG_OVF29	PORWDG_OVF29 captures the status of standby entry request initiated by PMU when PORWDG overflows **Note:** This bit is applicable only for standby sequence monitoring. 0: PMU standby request is active when PORWDG overflows. 1: PMU standby request is not active when PORWDG overflows
28:21	Reserved	Must be kept at reset value.

20	PORWDG_OVF20	PORWDG_STAT20 captures the status of subsystem reset sequence process SYSRST0 when PORWDG overflows. 0: SYSRST0 is inactive when PORWDG overflows. 1: SYSRST0 is active when PORWDG overflows
19:12	Reserved	Must be kept at reset value.
11	PORWDG_OVF11	PORWDG_OVF11 captures the status of subsystem reset sequence process SUBSRST8 when PORWDG overflows. 0: SUBSRST8 is inactive when PORWDG overflows. 1: SUBSRST8 is active when PORWDG overflows.
10	PORWDG_OVF10	PORWDG_OVF10 captures the status of subsystem reset sequence process SUBSRST7 when PORWDG overflows. 0: SUBSRST7 is inactive when PORWDG overflows. 1: SUBSRST7 is active when PORWDG overflows.
9:7	Reserved	Must be kept at reset value.
6	PORWDG_OVF6	PORWDG_OVF6 captures the status of subsystem reset sequence process SUBSRST6 when PORWDG overflows. 0: SUBSRST6 is inactive when PORWDG overflows. 1: SUBSRST6 is active when PORWDG overflows.
5	PORWDG_OVF5	PORWDG_OVF5 captures the status of subsystem reset sequence process SUBSRST5 when PORWDG overflows. 0: SUBSRST5 is inactive when PORWDG overflows. 1: SUBSRST5 is active when PORWDG overflows.
4	PORWDG_OVF4	PORWDG_OVF4 captures the status of subsystem reset sequence process SUBSRST4 when PORWDG overflows. 0: SUBSRST4 is inactive when PORWDG overflows. 1: SUBSRST4 is active when PORWDG overflows.
3	PORWDG_OVF3	PORWDG_OVF3 captures the status of subsystem reset sequence process SUBSRST3 when PORWDG overflows. 0: SUBSRST3 is inactive when PORWDG overflows. 1: SUBSRST3 is active when PORWDG overflows.
2	PORWDG_OVF2	PORWDG_OVF2 captures the status of subsystem reset sequence process SUBSRST2 when PORWDG overflows. 0: SUBSRST2 is inactive when PORWDG overflows. 1: SUBSRST2 is active when PORWDG overflows.
1	PORWDG_OVF1	PORWDG_OVF1 captures the status of subsystem reset sequence process SUBSRST1 when PORWDG overflows. 0: SUBSRST1 is inactive when PORWDG overflows. 1: SUBSRST1 is active when PORWDG overflows.
0	PORWDG_OVF0	PORWDG_OVF0 captures the status of subsystem reset sequence process

SUBSRST0 when PORWDG overflows.

0: SUBSRST0 is inactive when PORWDG overflows.

1: SUBSRST0 is active when PORWDG overflows.

5.3.18. Power-on reset during POR watchdog overflows status register 1 (RCU_PORSTAT1)

Address offset: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PORWDG_OVF63	PORWDG_OVF62	PORWDG_OVF61	PORWDG_OVF60	PORWDG_OVF59	PORWDG_OVF58	PORWDG_OVF57	PORWDG_OVF56	PORWDG_OVF55	PORWDG_OVF54	PORWDG_OVF53	PORWDG_OVF52	PORWDG_OVF51	PORWDG_OVF50	PORWDG_OVF49	PORWDG_OVF48
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PORWDG_OVF47	PORWDG_OVF46	PORWDG_OVF45	PORWDG_OVF44	PORWDG_OVF43	PORWDG_OVF42	PORWDG_OVF41	PORWDG_OVF40	PORWDG_OVF39	PORWDG_OVF38	PORWDG_OVF37	PORWDG_OVF36	PORWDG_OVF35	PORWDG_OVF34	PORWDG_OVF33	PORWDG_OVF32
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions
31	PORWDG_OVF63	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[31] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[31] is 1 when PORWDG overflows.
30	PORWDG_OVF62	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[30] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[30] is 1 when PORWDG overflows.
29	PORWDG_OVF61	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[29] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[29] is 1 when PORWDG overflows.
28	PORWDG_OVF60	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[28] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[28] is 1 when PORWDG overflows.
27	PORWDG_OVF59	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[27] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[27] is 1 when PORWDG overflows.
26	PORWDG_OVF58	This bit captures RCTL subsystem/external event status register when PORWDG

		overflows. 0: RCU_SUBSERSTAT[26] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[26] is 1 when PORWDG overflows.
25	PORWDG_OVF57	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[25] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[25] is 1 when PORWDG overflows.
24	PORWDG_OVF56	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[24] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[24] is 1 when PORWDG overflows.
23	PORWDG_OVF55	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[23] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[23] is 1 when PORWDG overflows.
22	PORWDG_OVF54	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[22] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[22] is 1 when PORWDG overflows.
21	PORWDG_OVF53	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[21] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[21] is 1 when PORWDG overflows.
20	PORWDG_OVF52	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[20] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[20] is 1 when PORWDG overflows.
19	PORWDG_OVF51	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[19] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[19] is 1 when PORWDG overflows.
18	PORWDG_OVF50	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[18] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[18] is 1 when PORWDG overflows.
17	PORWDG_OVF49	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[17] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[17] is 1 when PORWDG overflows.

16	PORWDG_OVF48	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[16] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[16] is 1 when PORWDG overflows.
15	PORWDG_OVF47	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[15] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[15] is 1 when PORWDG overflows.
14	PORWDG_OVF46	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[14] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[14] is 1 when PORWDG overflows.
13	PORWDG_OVF45	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[13] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[13] is 1 when PORWDG overflows.
12	PORWDG_OVF44	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[12] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[12] is 1 when PORWDG overflows.
11	PORWDG_OVF43	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[11] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[11] is 1 when PORWDG overflows.
10	PORWDG_OVF42	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[10] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[10] is 1 when PORWDG overflows.
9	PORWDG_OVF41	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[9] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[9] is 1 when PORWDG overflows.
8	PORWDG_OVF40	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[8] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[8] is 1 when PORWDG overflows.
7	PORWDG_OVF39	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[7] is 0 when PORWDG overflows.

1: RCU_SUBSERSTAT[7] is 1 when PORWDG overflows.

6	PORWDG_OVF38	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[6] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[6] is 1 when PORWDG overflows.
5	PORWDG_OVF37	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[5] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[5] is 1 when PORWDG overflows.
4	PORWDG_OVF36	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[4] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[4] is 1 when PORWDG overflows.
3	PORWDG_OVF35	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[3] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[3] is 1 when PORWDG overflows.
2	PORWDG_OVF34	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[2] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[2] is 1 when PORWDG overflows.
1	PORWDG_OVF33	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[1] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[1] is 1 when PORWDG overflows.
0	PORWDG_OVF32	This bit captures RCTL subsystem/external event status register when PORWDG overflows. 0: RCU_SUBSERSTAT[0] is 0 when PORWDG overflows. 1: RCU_SUBSERSTAT[0] is 1 when PORWDG overflows.

5.3.19. Power-on reset during POR watchdog overflows status register 2 (RCU_PORSTAT2)

Address offset: 0x48

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PORWDG_OVF95	PORWDG_OVF94	PORWDG_OVF93	PORWDG_OVF92	PORWDG_OVF91	PORWDG_OVF90	PORWDG_OVF89	PORWDG_OVF88	PORWDG_OVF87	PORWDG_OVF86	PORWDG_OVF85	PORWDG_OVF84	PORWDG_OVF83	PORWDG_OVF82	PORWDG_OVF81	PORWDG_OVF80
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

PORWDG_OVF79	PORWDG_OVF78	PORWDG_OVF77	PORWDG_OVF76	PORWDG_OVF75	PORWDG_OVF74	PORWDG_OVF73	PORWDG_OVF72	PORWDG_OVF71	PORWDG_OVF70	PORWDG_OVF69	PORWDG_OVF68	PORWDG_OVF67	PORWDG_OVF66	PORWDG_OVF65	PORWDG_OVF64
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions
31	PORWDG_OVF95	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[31] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[31] is 1 when PORWDG overflows.
30	PORWDG_OVF94	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[30] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[30] is 1 when PORWDG overflows.
29	PORWDG_OVF93	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[29] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[29] is 1 when PORWDG overflows.
28	PORWDG_OVF92	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[28] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[28] is 1 when PORWDG overflows.
27	PORWDG_OVF91	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[27] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[27] is 1 when PORWDG overflows.
26	PORWDG_OVF90	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[26] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[26] is 1 when PORWDG overflows.
25	PORWDG_OVF89	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[25] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[25] is 1 when PORWDG overflows.
24	PORWDG_OVF88	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[24] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[24] is 1 when PORWDG overflows.
23	PORWDG_OVF87	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[23] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[23] is 1 when PORWDG overflows.
22	PORWDG_OVF86	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[22] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[22] is 1 when PORWDG overflows.
21	PORWDG_OVF85	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[21] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[21] is 1 when PORWDG overflows.

20	PORWDG_OVF84	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[20] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[20] is 1 when PORWDG overflows.
19	PORWDG_OVF83	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[19] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[19] is 1 when PORWDG overflows.
18	PORWDG_OVF82	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[18] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[18] is 1 when PORWDG overflows.
17	PORWDG_OVF81	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[17] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[17] is 1 when PORWDG overflows.
16	PORWDG_OVF80	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[16] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[16] is 1 when PORWDG overflows.
15	PORWDG_OVF79	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[15] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[15] is 1 when PORWDG overflows.
14	PORWDG_OVF78	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[14] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[14] is 1 when PORWDG overflows.
13	PORWDG_OVF77	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[13] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[13] is 1 when PORWDG overflows.
12	PORWDG_OVF76	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[12] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[12] is 1 when PORWDG overflows.
11	PORWDG_OVF75	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[11] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[11] is 1 when PORWDG overflows.
10	PORWDG_OVF74	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[10] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[10] is 1 when PORWDG overflows.
9	PORWDG_OVF73	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[9] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[9] is 1 when PORWDG overflows.
8	PORWDG_OVF72	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[8] is 0 when PORWDG overflows.

		1: RCU_SYSRSTSTAT[8] is 1 when PORWDG overflows.
7	PORWDG_OVF71	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[7] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[7] is 1 when PORWDG overflows.
6	PORWDG_OVF70	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[6] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[6] is 1 when PORWDG overflows.
5	PORWDG_OVF69	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[5] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[5] is 1 when PORWDG overflows.
4	PORWDG_OVF68	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[4] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[4] is 1 when PORWDG overflows.
3	PORWDG_OVF67	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[3] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[3] is 1 when PORWDG overflows.
2	PORWDG_OVF66	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[2] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[2] is 1 when PORWDG overflows.
1	PORWDG_OVF65	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[1] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[1] is 1 when PORWDG overflows.
0	PORWDG_OVF64	This bit captures RCTL system event status register when PORWDG overflows. 0: RCU_SYSRSTSTAT[0] is 0 when PORWDG overflows. 1: RCU_SYSRSTSTAT[0] is 1 when PORWDG overflows.

5.3.20. Power-on reset during POR watchdog overflows status register 3 (RCU_PORSTAT3)

Address offset: 0x4C
Reset value: 0x0000 0000

All bits of this register should be configured in Inactive mode only, because they are blocked by hardware in other modes.

All bits are reset during subsystem reset

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	PORWDG_OVF96
	rc_w1

Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	PORWDG_OVF96	This bit captures PORWDG reset event if PORWDG initiates a POR sequence. 0: PORWDG is inactive. 1: PORWDG has detected a stuck scenario and raised device POR event. Refer PORWDG_OVF[95:0] for device status when PORWDG overflows

5.3.21. Triple modular redundancy error status (RCU_TMRERRSTAT)

Address offset: 0x50

Reset value: 0x0055 1555

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								TMR_PORWDG4	TMR_PORWDG4EN	TMR_PORWDG3	TMR_PORWDG3EN	TMR_PORWDG2	TMR_PORWDG2EN	TMR_PORWDG1	TMR_PORWDG1EN
								rc_w1	rw	rc_w1	rw	rc_w1	rw	rc_w1	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TMR_APBRST	TMR_APBRSTEN	TMR_AHBRST	TMR_AHBRSTEN	TMR_RCU_ERPACFG	TMR_RCU_ERPACFGEN	TMR_RCU_SYSMRUTH	TMR_RCU_SYSMRUTHEN	TMR_RCU_SUBSRPACTL	TMR_RCU_SUBSRPACTLEN	TMR_RCU_SUBSRCTL	TMR_RCU_SUBSRCTLEN	TMR_RCU_SYSRSTSTAT	TMR_RCU_SYSRSTATEN
		rc_w1	rw	rc_w1	rw	rc_w1	rw	rc_w1	rw	rc_w1	rw	rc_w1	rw	rc_w1	rw

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23	TMR_PORWDG4	TMR_PORWDG4 TMR error status 0: Normal 1: Error
22	TMR_PORWDG4EN	TMR_PORWDG4 TMR error output enable 0: Disabled 1: Enabled
21	TMR_PORWDG3	TMR_PORWDG3 TMR error status 0: Normal 1: Error
20	TMR_PORWDG3EN	TMR_PORWDG3 TMR error output enable

		0: Disabled 1: Enabled
19	TMR_PORWDG2	TMR_PORWDG32 TMR error status 0: Normal 1: Error
18	TMR_PORWDG2EN	TMR_PORWDG2 TMR error output enable 0: Disabled 1: Enabled
17	TMR_PORWDG1	TMR_PORWDG1 TMR error status 0: Normal 1: Error
16	TMR_PORWDG1EN	TMR_PORWDG1 TMR error output enable 0: Disabled 1: Enabled
15:14	Reserved	Must be kept at reset value.
13	TMR_APBRST	APBRST1/2 TMR error status The TMR error occurred in any of the CLTCFG RST bit, the PMURST bit in RCU_APB1RST register and the CFGRST bit in RCU_APB1RST register. 0: Normal 1: Error
12	TMR_APBRSTEN	APBRST1/2 TMR error output enable The CLTCFGRST bit, the PMURST bit in RCU_APB1RST register and the CFGRST bit in RCU_APB1RST register TMR error output enable 0: Disabled 1: Enabled
11	TMR_AHBRST	AHBRST TMR error status The TMR error occurred in any of the FMURST bit, the HWSEMRST bit, the MCMUB bit, the MCMUARST bit, the FMCRST bit, the DMAMUXRST bit, the DMA1RST bit and the DMA0RST bit in RCU_AHBRST register 0: Normal 1: Error
10	TMR_AHBRSTEN	AHBRST TMR error output enable The FMURST bit, the HWSEMRST bit, the MCMUB bit, the MCMUARST bit, the FMCRST bit, the DMAMUXRST bit, the DMA1RST bit and the DMA0RST bit in RCU_AHBRST register TMR error output enable. 0: Disabled 1: Enabled
9	TMR_RCU_ERPACFG	RCU_ERPACFG TMR error status 0: Normal

		1: Error
8	TMR_RCU_ERPACFGEN	RCU_ERPACFG TMR error output enable 0: Disabled 1: Enabled
7	TMR_RCU_SYSMRUTH	RCU_SYSMRUTH TMR error status 0: Normal 1: Error
6	TMR_RCU_SYSMRUTHEN	RCU_SYSMRUTH TMR error output enable 0: Disabled 1: Enabled
5	TMR_RCU_SUBSRPACTL	RCU_SUBSRPACTL TMR error status 0: Normal 1: Error
4	TMR_RCU_SUBSRPACTLEN	RCU_SUBSRPACTL TMR error output enable 0: Disabled 1: Enabled
3	TMR_RCU_SUBSRCTL	RCU_SUBSRCTL TMR error status 0: Normal 1: Error
2	TMR_RCU_SUBSRCTLEN	RCU_SUBSRCTL TMR error output enable 0: Disabled 1: Enabled
1	TMR_RCU_SYSRSTSTAT	RCU_SYSRSTSTAT TMR error status 0: Normal 1: Error
0	TMR_RCU_SYSRSTATEN	RCU_SYSRSTSTAT TMR error output enable 0: Disabled 1: Enabled

5.3.22. System reset upgrade counter register (RCU_SYSRSTUCNT)

Address offset: 0x54

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												SYSRTOEN[3:0]			

r

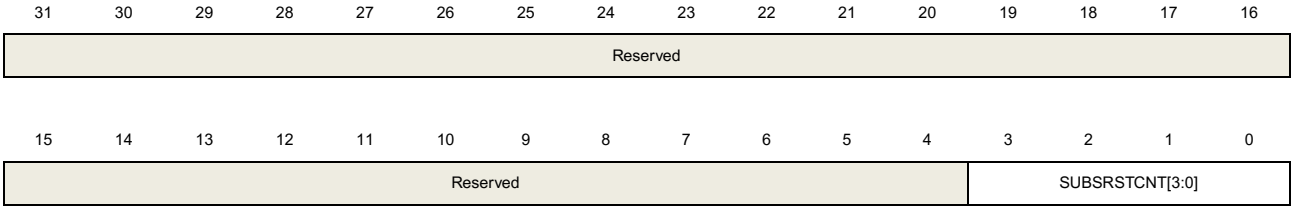
Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3:0	SYSRTOEN[3:0]	System reset upgrade counter The value of system reset upgrade counter is provided by this bit field. When system reset upgrade function is disabled, the SYSRTOEN[3:0] are cleared

5.3.23. Subsystem reset counter register (RCU_SUBSRSTCNT)

Address offset: 0x58

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



rc_w1

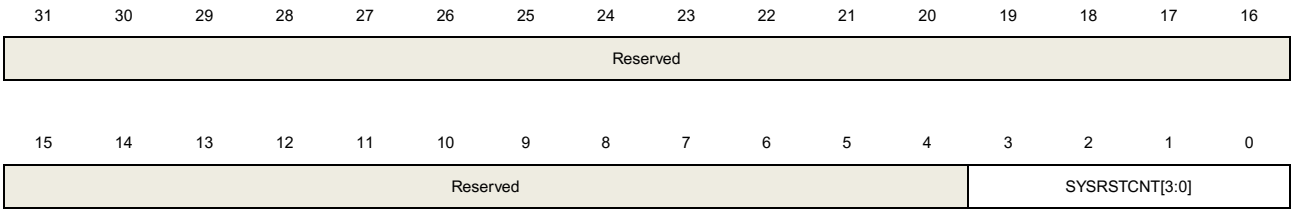
Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3:0	SUBSRSTCNT[3:0]	Subsystem reset counter

5.3.24. System reset counter register (RCU_SYSRSTCNT)

Address offset: 0x5C

Reset value: 0x0000 0001

This register has to be accessed by word(32-bit).



rw

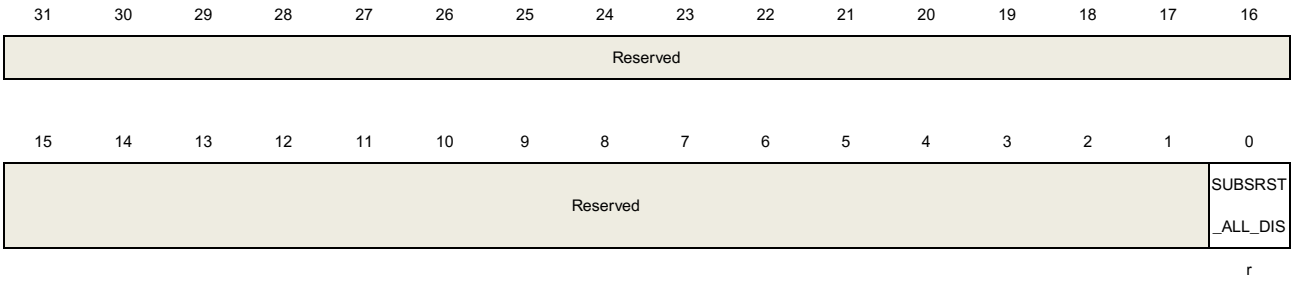
Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3:0	SYSRSTCNT[3:0]	System reset counter

5.3.25. Subsystem reset all disable register (RCU_SUBSRSTAD)

Address offset: 0x60

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	SUBSRST_ALL_DIS	Disable all subsystem reset 0: No effect 1: Disable all subsystem reset request Writing 0xA211A211 to this register will set bit 0 (SUBSRST_ALL_DIS), while writing 0x112B112B will clear bit 0 (SUBSRST_ALL_DIS). **Note**: If the register is configured as SUBSRST_ALL_DIS, it can only be awakened through system reset or POR reset.

5.4. Clock control unit registers definition

CCTL base address: 0x4900 1000

Table5-9. CCTL register overview

Register name	Symbol	Address offset	Reset value
Control register	RCU_CTL	0x00	0x0000 XX80
Configuration register 0	RCU_CFG0	0x04	0x0020 2400
Interrupt register	RCU_INT	0x08	0x0000 0000
AHB enable register	RCU_AHBEN	0x14	0x080A 0000
APB2 enable register	RCU_APB2EN	0x18	0x0203 0800
APB1 enable register	RCU_APB1EN	0x1C	0x7000 0000
Backup domain control register	RCU_BDCTL	0x20	0x0000 0010
APB2 enable additional register	RCU_APB2EN_ADD	0x24	0x0000 0000
AHB enable additional register	RCU_AHBEN_ADD	0x28	0x0000 0003
Configuration register 1	RCU_CFG1	0x2C	0x0000 0000
Configuration register 2	RCU_CFG2	0x30	0x0000 0000
Configuration register 3	RCU_CFG3	0x34	0x0000 0000
PLL loss of lock register	RCU_PLLLOL	0x38	0x0000 0000
Clock detect control register	RCU_CKDCTL	0x3C	0x0000 0000
Clock frequency monitor x control register	RCU_CKFMxCTL, x=0..4	0x40+0x10*x	0x0000 0000
Clock frequency monitor x reference count configuration register	RCU_CKFMxRCCFG, x=0..4	0x44+0x10*x	0x0000 0000
CKFMx high threshold configuration register	RCU_CKFMxHTHCFG, x=0..4	0x48+0x10*x	0x0000 0000
CKFMx low threshold configuration register	RCU_CKFMxLTHCFG, x=0..4	0x4C+0x10*x	0x0000 0000
CCTL TMR register	RCU_BUSIM_TMR	0x90	0x0000 000f
PLL fraction configuration register	RCU_PLLFRA	0x98	0x0000 0000
PLL clock spread spectrum control register	RCU_PLLSSCTL	0x9c	0x0000 0000

5.4.1. Control register (RCU_CTL)

Address offset: 0x00

Reset value: 0x0000 XX80. X indicates undefined.

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			HXTALSTBC	Reserved		PLLSTB	PLLEN	HXTALSCAL[1:0]		Reserved	LXTALSTBC	CKMEN	HXTALBPS	HXTALSTB	HXTALEN
			w			r	rw	rw			w	rw	rw	r	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

IRC48MCALIB[7:0]	IRC48MADJ[4:0]	Reserved
r	rw	

Bits	Fields	Descriptions
31:29	Reserved	Must be kept at reset value.
28	HXTALSTBC	Clear HXTALSTB 0: HXTALSTB is not cleared 1: HXTALSTB is cleared
27:26	Reserved	Must be kept at reset value.
25	PLLSTB	PLL clock stabilization flag Set by hardware to indicate if the PLL output clock is stable and ready for use. 0: PLL is not stable 1: PLL is stable
24	PLLEN	PLL enable Set and reset by software. 0: PLL is switched off 1: PLL is switched on
23:22	HXTALSCAL[1:0]	HXTAL frequency scale select The HXTALSCAL bit can be written only if the HXTALEN is 0. When HXTAL is 8, 16, or 32MHz, the recommended values are 00, 01, or 10. 00: HXTAL scale is 4 ~ 8MHz 01: HXTAL scale is 8 ~ 16MHz 10: HXTAL scale is 16 ~ 32MHz 11: HXTAL scale is 32 ~ 40MHz
21	Reserved	Must be kept at reset value.
20	LXTALSTBC	Clear LXTALSTB 0: LXTALSTB is not cleared 1: LXTALSTB is cleared
19	CKMEN	HXTAL clock monitor enable 0: Disable the external 4 ~ 40 MHz crystal oscillator (HXTAL) clock monitor 1: Enable the external 4~ 40 MHz crystal oscillator (HXTAL) clock monitor When the hardware detects that the HXTAL clock is stuck at a low or high state, the internal hardware will switch the system clock to be the internal high speed IRC48M RC clock. The way to recover the original system clock is by either an external reset, power on reset or clearing CKMIF by software. **Note:** When the HXTAL clock monitor is enabled, the hardware will automatically enable the IRC48M internal RC oscillator.
18	HXTALBPS	External crystal oscillator (HXTAL) clock bypass mode enable The HXTALBPS bit can be written only if the HXTALEN is 0. 0: Disable the HXTAL bypass mode

		1: Enable the HXTAL bypass mode in which the HXTAL output clock is equal to the input clock.
17	HXTALSTB	External crystal oscillator (HXTAL) clock stabilization flag Set by hardware to indicate if the HXTAL oscillator is stable and ready for use. 0: HXTAL oscillator is not stable 1: HXTAL oscillator is stable
16	HXTALEN	External high speed oscillator enable Set and reset by software. This bit cannot be reset if the HXTAL clock is used as the system clock or the PLL input clock. 0: External 4~ 40 MHz crystal oscillator disabled 1: External 4 ~ 40 MHz crystal oscillator enabled
15:8	IRC48MCALIB[7:0]	Internal 48M RC oscillator calibration value register These bits are load automatically at power on.
7:3	IRC48MADJ[4:0]	Internal 48M RC oscillator clock trim adjust value These bits are set by software. The trimming value is these bits (IRC48MADJ) added to the IRC48MCALIB[7:0] bits. The trimming value should trim the IRC48M to 48 MHz ± 1%.
2:0	Reserved	Must be kept at reset value.

5.4.2. Configuration register 0 (RCU_CFG0)

Address offset: 0x04

Reset value: 0x0020 2400

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PLLDV	CKOUT1DIV[2:0]			PLLMF[7]	CKOUT1SEL[2:0]			PLLMF[6:0]							PLLSEL
rw	rw			rw	rw			rw							rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		APB2PSC[2:0]			APB1PSC[2:0]			AHBPSC[3:0]				SCSS[1:0]		SCS[1:0]	
		rw			rw			rw				r		rw	

Bits	Fields	Descriptions
31	PLLDV	The CK_PLL_P divide by 1 or 2 for CK_OUT 0: CK_PLL_P divide by 2 for CK_OUT 1: CK_PLL_P divide by 1 for CK_OUT

30:28	CKOUT1DIV[2:0]	The CK_OUT1 divider which the CK_OUT1 frequency can be reduced, see bits 26:24 of RCU_CFG0 for CK_OUT1 000: The CK_OUT1 is divided by 1 001: The CK_OUT1 is divided by 2 010: The CK_OUT1 is divided by 4 011: The CK_OUT1 is divided by 8 100: The CK_OUT1 is divided by 16 101: The CK_OUT1 is divided by 32 110: The CK_OUT1 is divided by 64 111: The CK_OUT1 is divided by 128
27	PLLMF[7]	Bit 7 of PLLMF register see bits 23:17 of RCU_CFG0.
26:24	CKOUT1SEL[2:0]	CK_OUT1 clock source selection Set and reset by software. 000: Internal 48MHz RC oscillator clock selected 001: External high speed oscillator clock selected 010: Internal 40K RC oscillator clock selected 011: External low speed oscillator clock selected 100: CK_APB2 101: reserved 110: reserved 111: reserved
23:17	PLLMF[6:0]	PLL multiply factors These bits and bit 27 of RCU_CFG0 are written by software to define the PLL multiplication factor. **Note:** The minimum support for integer frequency doubling is 16, and the minimum support for decimal frequency division and spread mode is only 21 00000000: reserved 00000001: reserved …… 00001111: reserved 00010000: (PLL source clock x 16) …… 00010100: (PLL source clock x 20) 00010101: (PLL source clock x 21) 00010110: (PLL source clock x 22) 00010111: (PLL source clock x 23) …… 11111111: (PLL source clock x 255) **Note:** The PLL output frequency must not exceed 160 MHz.
16	PLLSEL	PLL clock source selection Set and reset by software to control the PLL clock source.

		0: IRC48M clock selected as source clock of PLL 1: HXTAL selected as source clock of PLL
15:14	Reserved	Must be kept at reset value
13:11	APB2PSC[2:0]	APB2 prescaler selection Set and reset by software to control the APB2 clock division ratio. 0xx: CK_AHB selected 100: (CK_AHB / 2) selected 101: (CK_AHB / 4) selected 110: (CK_AHB / 8) selected 111: (CK_AHB / 16) selected
10:8	APB1PSC[2:0]	APB1 prescaler selection Set and reset by software to control the APB1 clock division ratio. 0xx: CK_AHB selected 100: (CK_AHB / 2) selected 101: (CK_AHB / 4) selected 110: (CK_AHB / 8) selected 111: (CK_AHB / 16) selected
7:4	AHBPSC[3:0]	AHB prescaler selection Set and reset by software to control the AHB clock division ratio 0xxx: CK_SYS selected 1000: (CK_SYS / 2) selected 1001: (CK_SYS / 4) selected 1010: (CK_SYS / 8) selected 1011: (CK_SYS / 16) selected 1100: (CK_SYS / 64) selected 1101: (CK_SYS / 128) selected 1110: (CK_SYS / 256) selected 1111: (CK_SYS / 512) selected
3:2	SCSS[1:0]	System clock switch status Set and reset by hardware to indicate the clock source of system clock. 00: select CK_IRC48M as the CK_SYS source 01: select CK_HXTAL as the CK_SYS source 10: select CK_PLL_P as the CK_SYS source 11: reserved
1:0	SCS[1:0]	System clock switch Set by software to select the CK_SYS source. Because the change of CK_SYS has inherent latency, software should read SCSS to confirm whether the switching is complete or not. The switch will be forced to IRC48M when leaving Standby mode or by HXTAL clock monitor when the HXTAL failure is detected and the HXTAL is selected as the clock source of CK_SYS or PLL. 00: select CK_IRC48M as the CK_SYS source

01: select CK_HXTAL as the CK_SYS source

10: select CK_PLL_P as the CK_SYS source

11: reserved

5.4.3. Interrupt register (RCU_INT)

Address offset: 0x08

Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								CKMIC	Reserved		PLL STBIC	HXTAL STBIC	Reserved	LXTAL STBIC	Reserved
								w			w	w		w	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			PLL STBIE	HXTAL STBIE	Reserved	LXTAL STBIE	Reserved	CKMIF	Reserved		PLL STBIF	HXTAL STBIF	Reserved	LXTAL STBIF	Reserved
			rw	rw		rw		r			r	r		r	

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value
23	CKMIC	HXTAL clock stuck interrupt clear Write 1 by software to reset the CKMIF flag. 0: Not reset CKMIF flag 1: Reset CKMIF flag
22:21	Reserved	Must be kept at reset value
20	PLLSTBIC	PLL stabilization interrupt clear Write 1 by software to reset the PLLSTBIF flag. 0: Not reset PLLSTBIF flag 1: Reset PLLSTBIF flag
19	HXTALSTBIC	HXTAL stabilization interrupt clear Write 1 by software to reset the HXTALSTBIF flag. 0: Not reset HXTALSTBIF flag 1: Reset HXTALSTBIF flag
18	Reserved	Must be kept at reset value
17	LXTALSTBIC	LXTAL stabilization interrupt clear Write 1 by software to reset the LXTALSTBIF flag. 0: Not reset LXTALSTBIF flag 1: Reset LXTALSTBIF flag
16:13	Reserved	Must be kept at reset value

12	PLLSTBIE	PLL stabilization interrupt enable Set and reset by software to enable/disable the PLL stabilization interrupt. 0: Disable the PLL stabilization interrupt 1: Enable the PLL stabilization interrupt
11	HXTALSTBIE	HXTAL stabilization interrupt enable Set and reset by software to enable/disable the HXTAL stabilization interrupt 0: Disable the HXTAL stabilization interrupt 1: Enable the HXTAL stabilization interrupt
10	Reserved	Must be kept at reset value
9	LXTALSTBIE	LXTAL stabilization interrupt enable LXTAL stabilization interrupt enable/disable control 0: Disable the LXTAL stabilization interrupt 1: Enable the LXTAL stabilization interrupt
8	Reserved	Must be kept at reset value
7	CKMIF	HXTAL clock stuck interrupt flag Set by hardware when the HXTAL clock is stuck. Reset by software when setting the CKMIC bit. Reset by software when setting the CKNMIIF bit in SYSCFG_STAT register. 0: Clock operating normally 1: HXTAL clock stuck
6:5	Reserved	Must be kept at reset value
4	PLLSTBIF	PLL stabilization interrupt flag Set by hardware when the PLL is stable and the PLLSTBIE bit is set. Reset by software when setting the PLLSTBIC bit. 0: No PLL stabilization interrupt generated 1: PLL stabilization interrupt generated
3	HXTALSTBIF	HXTAL stabilization interrupt flag Set by hardware when the external 4 ~ 40 MHz crystal oscillator clock is stable and the HXTALSTBIE bit is set. Reset by software when setting the HXTALSTBIC bit. 0: No HXTAL stabilization interrupt generated 1: HXTAL stabilization interrupt generated
2	Reserved	Must be kept at reset value
1	LXTALSTBIF	LXTAL stabilization interrupt flag Set by hardware when the external 32,768 Hz crystal oscillator clock is stable and the LXTALSTBIE bit is set. Reset by software when setting the LXTALSTBIC bit. 0: No LXTAL stabilization interrupt generated 1: LXTAL stabilization interrupt generated

0	Reserved	Must be kept at reset value

5.4.4. AHB enable register (RCU_AHBEN)

Address offset: 0x14

Reset value: 0x080A 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MTCEN	STCMEN	Reserved	IRMEN	FMUEN	Reserved	HSMIFEN	Reserved				CRCEN	Reserved	QSPIEN	Reserved	DMAMUXEN
rw	rw		rw	rw		rw					rw		rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA1EN	DMA0EN	PNEN	PMEN	PLEN	PKEN	PJEN	PIEN	PHEN	PGEN	PFEN	PEEN	PDEN	PCEN	PBEN	PAEN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31	MTCEN	MTC clock enable This bit is set and reset by software. 0: Disabled MTC clock 1: Enabled MTC clock
30	STCMEN	STCM clock enable This bit is set and reset by software. 0: Disabled STCM clock 1: Enabled STCM clock
29	Reserved	Must be kept at reset value
28	IRMEN	IRM clock enable This bit is set and reset by software. 0: Disabled IRM clock 1: Enabled IRM clock
27	FMUEN	FMU clock enable This bit is set and reset by software. 0: Disabled FMU clock 1: Enabled FMU clock
26	Reserved	Must be kept at reset value
25	HSMIFEN	HSMIF clock enable This bit is set and reset by software. 0: Disabled HSMIF clock 1: Enabled HSMIF clock
24:21	Reserved	Must be kept at reset value

20	CRCEN	CRC clock enable This bit is set and reset by software. 0: Disabled CRC clock 1: Enabled CRC clock
19	Reserved	Must be kept at reset value
18	QSPIEN	QSPI clock enable This bit is set and reset by software. 0: Disabled QSPI clock 1: Enabled QSPI clock
17	Reserved	Must be kept at reset value
16	DMAMUXEN	DMAMUX clock enable This bit is set and reset by software. 0: Disabled DMAMUX clock 1: Enabled DMAMUX clock
15	DMA1EN	DMA1 clock enable This bit is set and reset by software. 0: Disabled DMA1 clock 1: Enabled DMA1 clock
14	DMA0EN	DMA0 clock enable This bit is set and reset by software. 0: Disabled DMA0 clock 1: Enabled DMA0 clock
13	PNEN	GPIO port N clock enable This bit is set and reset by software. 0: Disabled GPIO port N clock 1: Enabled GPIO port N clock
12	PMEN	GPIO port M clock enable This bit is set and reset by software. 0: Disabled GPIO port M clock 1: Enabled GPIO port M clock
11	PLEN	GPIO port L clock enable This bit is set and reset by software. 0: Disabled GPIO port L clock 1: Enabled GPIO port L clock
10	PKEN	GPIO port K clock enable This bit is set and reset by software. 0: Disabled GPIO port K clock 1: Enabled GPIO port K clock
9	PJEN	GPIO port J clock enable

		This bit is set and reset by software. 0: Disabled GPIO port J clock 1: Enabled GPIO port J clock
8	PIEN	GPIO port I clock enable This bit is set and reset by software. 0: Disabled GPIO port I clock 1: Enabled GPIO port I clock
7	PHEN	GPIO port H clock enable This bit is set and reset by software. 0: Disabled GPIO port H clock 1: Enabled GPIO port H clock
6	PGEN	GPIO port G clock enable This bit is set and reset by software. 0: Disabled GPIO port G clock 1: Enabled GPIO port G clock
5	PFEN	GPIO port F clock enable This bit is set and reset by software. 0: Disabled GPIO port F clock 1: Enabled GPIO port F clock
4	PEEN	GPIO port E clock enable This bit is set and reset by software. 0: Disabled GPIO port E clock 1: Enabled GPIO port E clock
3	PDEN	GPIO port D clock enable This bit is set and reset by software. 0: Disabled GPIO port D clock 1: Enabled GPIO port D clock
2	PCEN	GPIO port C clock enable This bit is set and reset by software. 0: Disabled GPIO port C clock 1: Enabled GPIO port C clock
1	PBEN	GPIO port B clock enable This bit is set and reset by software. 0: Disabled GPIO port B clock 1: Enabled GPIO port B clock
0	PAEN	GPIO port A clock enable This bit is set and reset by software. 0: Disabled GPIO port A clock 1: Enabled GPIO port A clock

5.4.5. APB2 enable register (RCU_APB2EN)

Address offset: 0x18

Reset value: 0x0203 0800

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					CMP0EN	SPI1EN	Reserved	CMP1EN	TIMER62EN	TIMER5EN	TIMER1EN	TIMER19EN	Reserved	SPI5EN	SPI4EN
					rw	rw		rw	rw	rw	rw	rw		rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reservd			TIMER7EN	SPI0EN	TIMER0EN	ADC1EN	ADC0EN	TRIGSELEN	LINFLEXD5EN	LINFLEXD4EN	LINFLEXD3EN	LINFLEXD2EN	LINFLEXD1EN	LINFLEXD0EN	SYSCFGEN
			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:27	Reserved	Must be kept at reset value
26	CMP0EN	Comparator0 clock enable This bit is set and reset by software. 0: Disabled comparator0 clock 1: Enabled comparator0 clock
25	SPI1EN	SPI1 clock enable This bit is set and reset by software. 0: Disabled SPI1 clock 1: Enabled SPI1 clock
24	Reserved	Must be kept at reset value
23	CMP1EN	Comparator1 clock enable This bit is set and reset by software. 0: Disabled comparator clock 1: Enabled comparator clock
22	TIMER62EN	TIMER62 timer clock enable This bit is set and reset by software. 0: Disabled TIMER62 timer clock 1: Enabled TIMER62 timer clock
21	TIMER5EN	TIMER5 timer clock enable This bit is set and reset by software. 0: Disabled TIMER5 timer clock 1: Enabled TIMER5 timer clock
20	TIMER1EN	TIMER1 timer clock enable

		This bit is set and reset by software. 0: Disabled TIMER1 timer clock 1: Enabled TIMER1 timer clock
19	TIMER19EN	TIMER19 timer clock enable This bit is set and reset by software. 0: Disabled TIMER19 timer clock 1: Enabled TIMER19 timer clock
18	Reserved	Must be kept at reset value
17	SPI5EN	SPI5 clock enable This bit is set and reset by software. 0: Disabled SPI5 clock 1: Enabled SPI5 clock
16	SPI4EN	SPI4 clock enable This bit is set and reset by software. 0: Disabled SPI4 clock 1: Enabled SPI4 clock
15:13	Reserved	Must be kept at reset value
12	TIMER7EN	TIMER7 timer clock enable This bit is set and reset by software. 0: Disabled TIMER7 timer clock 1: Enabled TIMER7 timer clock
11	SPI0EN	SPI0 clock enable This bit is set and reset by software. 0: Disabled SPI0 clock 1: Enabled SPI0 clock
10	TIMER0EN	TIMER0 timer clock enable This bit is set and reset by software. 0: Disabled TIMER0 timer clock 1: Enabled TIMER0 timer clock
9	ADC1EN	ADC1 interface clock enable This bit is set and reset by software. 0: Disabled ADC1 interface clock 1: Enabled ADC1 interface clock
8	ADC0EN	ADC interface clock enable This bit is set and reset by software. 0: Disabled ADC0 interface clock 1: Enabled ADC0 interface clock
7	TRIGSELEN	TRIGSEL clock enable

		This bit is set and reset by software. 0: Disabled TRIGSEL clock 1: Enabled TRIGSEL clock
6	LINFLEXD5EN	LINFLEXD5 clock enable This bit is set and reset by software. 0: Disabled LINFLEXD5 clock 1: Enabled LINFLEXD5 clock
5	LINFLEXD4EN	LINFLEXD4 clock enable This bit is set and reset by software. 0: Disabled LINFLEXD4 clock 1: Enabled LINFLEXD4 clock
4	LINFLEXD3EN	LINFLEXD3 clock enable This bit is set and reset by software. 0: Disabled LINFLEXD3 clock 1: Enabled LINFLEXD3 clock
3	LINFLEXD2EN	LINFLEXD2 clock enable This bit is set and reset by software. 0: Disabled LINFLEXD2 clock 1: Enabled LINFLEXD2 clock
2	LINFLEXD1EN	LINFLEXD1 clock enable This bit is set and reset by software. 0: Disabled LINFLEXD1 clock 1: Enabled LINFLEXD1 clock
1	LINFLEXD0EN	LINFLEXD0 clock enable This bit is set and reset by software. 0: Disabled LINFLEXD0 clock 1: Enabled LINFLEXD0 clock
0	SYSCFGEN	System configuration clock enable This bit is set and reset by software. 0: Disabled system configuration clock 1: Enabled system configuration clock

5.4.6. APB1 enable register (RCU_APB1EN)

Address offset:0x1C

Reset value: 0x7000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	CLTCFGEN	SPI3EN	SPI2EN	Reserved		DAC1EN	Reserved							TIMER63EN	TIMER61EN

	rw	rw	rw			rw								rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	DAC0EN	PMUEN	BKPEN	I2C1EN	I2C0EN	Reserved				FWDGT0EN	WWDGT0EN	TIMER6EN	TIMER60EN	TIMER2EN	TIMER20EN
	rw	rw	rw	rw	rw					rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value
30	CLTCFGEN	CLTCFG clock enable This bit is set and reset by software. 0: Disabled CLTCFG clock 1: Enabled CLTCFG clock
29	SPI3EN	SPI3 clock enable This bit is set and reset by software. 0: Disabled SPI3 clock 1: Enabled SPI3 clock
28	SPI2EN	SPI2 clock enable This bit is set and reset by software. 0: Disabled SPI2 clock 1: Enabled SPI2 clock
27:26	Reserved	Must be kept at reset value
25	DAC1EN	DAC1clock enable This bit is set and reset by software. 0: Disabled DAC1 clock 1: Enabled DAC1 clock
24:18	Reserved	Must be kept at reset value
17	TIMER63EN	TIMER63 timer clock enable This bit is set and reset by software. 0: Disabled TIMER63 timer clock 1: Enabled TIMER63 timer clock
16	TIMER61EN	TIMER61 timer clock enable This bit is set and reset by software. 0: Disabled TIMER61 timer clock 1: Enabled TIMER61 timer clock
15	Reserved	Must be kept at reset value
14	DAC0EN	DAC0 clock enable This bit is set and reset by software. 0: Disabled DAC0 clock 1: Enabled DAC0 clock

13	PMUEN	Power interface clock enable This bit is set and reset by software. 0: Disabled Power interface clock 1: Enabled Power interface clock
12	BKPEN	Back-up interface clock enable This bit is set and reset by software. 0: Disabled Back-up interface clock 1: Enabled Back-up interface clock
11	I2C1EN	I2C1 clock enable This bit is set and reset by software. 0: Disabled I2C1 clock 1: Enabled I2C1 clock
10	I2C0EN	I2C0 clock enable This bit is set and reset by software. 0: Disabled I2C0 clock 1: Enabled I2C0 clock
9:6	Reserved	Must be kept at reset value
5	FWDGT0EN	Free watchdog timer clock enable This bit is set and reset by software. 0: Disabled Free watchdog timer clock 1: Enabled Free watchdog timer clock
4	WWDGT0EN	Window watchdog timer clock enable This bit is set and reset by software. 0: Disabled window watchdog timer clock 1: Enabled window watchdog timer clock
3	TIMER6EN	TIMER6 timer clock enable This bit is set and reset by software. 0: Disabled TIMER6 timer clock 1: Enabled TIMER6 timer clock
2	TIMER60EN	TIMER60 timer clock enable This bit is set and reset by software. 0: Disabled TIMER60 timer clock 1: Enabled TIMER60 timer clock
1	TIMER2EN	TIMER2 timer clock enable This bit is set and reset by software. 0: Disabled TIMER2 timer clock 1: Enabled TIMER2 timer clock
0	TIMER20EN	TIMER20 timer clock enable This bit is set and reset by software.

0: Disabled TIMER20 timer clock

1: Enabled TIMER20 timer clock

5.4.7. Backup domain control register (RCU_BDCTL)

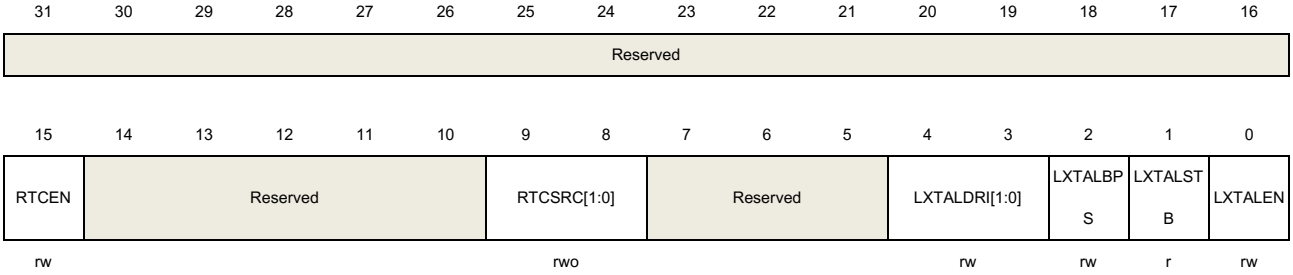
Address offset: 0x20

Reset value: 0x0000 0010.

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

Note: The LXTALEN, LXTALBPS, RTCSRC and RTCEN bits of the Backup domain control register (RCU_BDCTL) are only reset after a Backup domain Reset. These bits can be modified only when the BKPWEN bit in the power control register (PMU_CTL) has to be set.

Note: It will lock if BKP clock is off.



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15	RTCEN	RTC clock enable This bit is set and reset by software. 0: Disabled RTC clock 1: Enabled RTC clock
14:10	Reserved	Must be kept at reset value
9:8	RTCSRC[1:0]	RTC clock entry selection Set and reset by software to control the RTC clock source. 00: No clock selected 01: CK_LXTAL selected as RTC source clock 10: CK_IRC40K selected as RTC source clock 11: (CK_HXTAL / 128) selected as RTC source clock
7:5	Reserved	Must be kept at reset value
4:3	LXTALDRI[1:0]	LXTAL drive capability Set and reset by software. Backup domain reset resets this value. 00: lower driving capability 01: medium low driving capability 10: medium high driving capability 11: higher driving capability (reset value)

		Note: The LXTALDRI is not in bypass mode.
2	LXTALBPS	LXTAL bypass mode enable Set and reset by software. 0: Disable the LXTAL Bypass mode 1: Enable the LXTAL Bypass mode
1	LXTALSTB	External low-speed oscillator stabilization Set by hardware to indicate if the LXTAL output clock is stable and ready for use. 0: LXTAL is not stable 1: LXTAL is stable
0	LXTALEN	LXTAL enable Set and reset by software. 0: Disable LXTAL 1: Enable LXTAL

5.4.8. APB2 enable additional register (RCU_APB2EN_ADD)

Address offset: 0x24
Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

For specific DTM_CAN clock configuration considerations, please refer to the description in Section *22.3.1*.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DTM_CAN7EN	DTM_CAN6EN	DTM_CAN5EN	DTM_CAN4EN	DTM_CAN3EN	DTM_CAN2EN	DTM_CAN1EN	DTM_CAN0EN
								rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value
7	DTM_CAN7EN	DTM_CAN7 clock enable This bit is set and reset by software. 0: Disabled DTM_CAN7 clock 1: Enabled DTM_CAN7 clock
6	DTM_CAN6EN	DTM_CAN6 clock enable This bit is set and reset by software. 0: Disabled DTM_CAN6 clock 1: Enabled DTM_CAN6 clock
5	DTM_CAN5EN	DTM_CAN5 clock enable

This bit is set and reset by software.
0: Disabled DTM_CAN5 clock
1: Enabled DTM_CAN5 clock

4	DTM_CAN4EN	DTM_CAN4 clock enable This bit is set and reset by software. 0: Disabled DTM_CAN4 clock 1: Enabled DTM_CAN4 clock
3	DTM_CAN3EN	DTM_CAN3 clock enable This bit is set and reset by software. 0: Disabled DTM_CAN3 clock 1: Enabled DTM_CAN3 clock
2	DTM_CAN2EN	DTM_CAN2 clock enable This bit is set and reset by software. 0: Disabled DTM_CAN2 clock 1: Enabled DTM_CAN2 clock
1	DTM_CAN1EN	DTM_CAN1 clock enable This bit is set and reset by software. 0: Disabled DTM_CAN1 clock 1: Enabled DTM_CAN1 clock
0	DTM_CAN0EN	DTM_CAN0 clock enable This bit is set and reset by software. 0: Disabled DTM_CAN0 clock 1: Enabled DTM_CAN0 clock

5.4.9. AHB enable additional register (RCU_AHBEN_ADD)

Address offset: 0x28
Reset value: 0x0000 0003

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CM7_0_EN
															rw

Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value
0	CM7_0_EN	CPU CM7_0 clock enable

This bit is set and reset by software.

0: Disabled CM7_0 clock

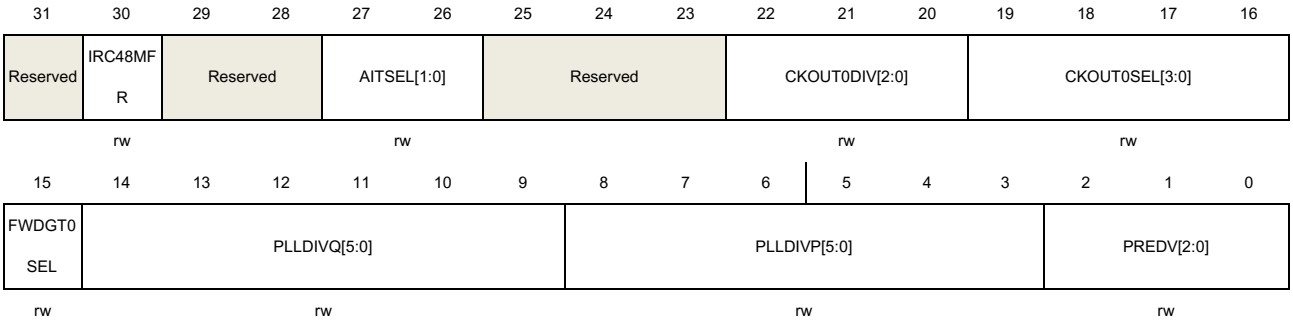
1: Enabled CM7_0 clock

5.4.10. Configuration register 1 (RCU_CFG1)

Address offset: 0x2C

Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)



Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value
30	IRC48MFR	IRC48M fail response 0: When IRC48M serves as the system clock, the POR_WDG of RCTL is triggered when IRC48M clock monitoring fails. If the error flag is not cleared before timeout, POR reset is triggered. 1: The failure of IRC48M will set FFIO0 and FFIO1 pins to High-Z state (refer to AFIO for more information)
29:28	Reserved	Must be kept at reset value
27:26	AITSEL[1:0]	AIT clock of STCM 00:CK_IRC48M 01:CK_PLL_Q/2 10: CK_PLL_Q/3 11: CK_PLL_Q/4
25:23	Reserved	Must be kept at reset value
22:20	CKOUT0DIV[2:0]	The CK_OUT0 divider which the CK_OUT0 frequency can be reduced, 000: The CK_OUT0 is divided by 1 001: The CK_OUT0 is divided by 2 010: The CK_OUT0 is divided by 4 011: The CK_OUT0 is divided by 8 100: The CK_OUT0 is divided by 16 101: The CK_OUT0 is divided by 32 110: The CK_OUT0 is divided by 64

		111: The CK_OUT0 is divided by 128
19:16	CKOUT0SEL[3:0]	CK_OUT0 clock source selection Set and reset by software. 0000: core clock 0001: PCLK1 0010: Internal 40K RC oscillator clock selected 0011: External low speed oscillator clock selected 0100: System clock selected 0101: Internal 48MHz RC oscillator clock selected 0110: External high speed oscillator clock selected 0111: (CK_PLL_P / 2) or CK_PLL_P selected depend on PLLDV 1000: reserved 1001: PCLK2 1010: PCLK from HSM 1011: CK_AHB from HSM
15	FWDGT0SEL	FWDGT0SEL clock select 0: IRC40K 1: LXTAL
14:9	PLLDIVQ[5:0]	PLL VCO clock divider CK_PLL_Q=PLL_VCO/(plldivq+1) CK_PLL_Q clock divider 000000: The CK_PLL_Q clock is divided by 1 000001: The CK_PLL_Q clock is divided by 2 000010: The CK_PLL_Q clock is divided by 3 …… 111101: The CK_PLL_Q clock is divided by 62 111110: The CK_PLL_Q clock is divided by 63 111111: The CK_PLL_Q clock is divided by 64
8:3	PLLDIVP[5:0]	PLL VCO clock divider CK_PLL_P=PLL_VCO/(plldivp+1) CK_PLL_P clock divider 000000: The CK_PLL_P clock is divided by 1 000001: The CK_PLL_P clock is divided by 2 000010: The CK_PLL_P clock is divided by 3 …… 111101: The CK_PLL_P clock is divided by 62 111110: The CK_PLL_P clock is divided by 63 111111: The CK_PLL_P clock is divided by 64
2:0	PREDV[2:0]	CK_HXTAL divider previous PLL This bit is set and reset by software. These bits can be written when PLL

is disable

The CK_HXTAL is divided by PREDV

000: input to PLL not divided

001: input to PLL not divided

010: input to PLL divided by 2

011: input to PLL divided by 3

100: input to PLL divided by 4

101: input to PLL divided by 5

110: input to PLL divided by 6

111: input to PLL divided by 7

5.4.11. Configuration register 2 (RCU_CFG2)

Address offset: 0x30

Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADCPSC[4:0]					Reserved	ADCSEL	ADCCLK_MUX	DTM_CAN6SEL[1:0]		DTM_CAN5SEL[1:0]		DTM_CAN4SEL[1:0]		DTM_CAN3SEL[1:0]	
rw						rw	rw	rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DTM_CAN2SEL[1:0]		DTM_CAN1SEL[1:0]		DTM_CAN0SEL[1:0]		Reserved									
rw		rw		rw											

Bits	Fields	Descriptions
31:27	ADCPSC[4:0]	ADC clock prescaler selection, set and cleared by software. CK_ADC = CK_SOURCE / (ADCPSC + 1) CK_SOURCE is one of CK_PLL_P, CK_AHB and CK_SYS. 00000: (CK_SOURCE / 1) selected 00001: (CK_SOURCE / 2) selected 00010: (CK_SOURCE / 3) selected 00011: (CK_SOURCE / 4) selected …… 11101: (CK_SOURCE / 30) selected 11110: (CK_SOURCE / 31) selected 11111: (CK_SOURCE / 32) selected
26	Reserved	Must be kept at reset value
25	ADCSEL	ADC clock source selection 0: CK_SYS 1: CK_PLL_P
24	ADCCLK_MUX	ADCCLK mux signal

		0:CK_AHB 1:CK_PLL_P/CK_SYS
23:22	DTM_CAN6SEL[1:0]	CK_DTM_CAN6 clock source selection This bit is set and reset by software. 00: CK_DTM_CAN6 select CK_IRC48M 01: CK_DTM_CAN6 select CK_PCLK2 10: CK_DTM_CAN6 select CK_AHB 11: CK_DTM_CAN6 select CK_HXTAL
21:20	DTM_CAN5SEL[1:0]	CK_DTM_CAN5 clock source selection This bit is set and reset by software. 00: CK_DTM_CAN5 select CK_IRC48M 01: CK_DTM_CAN5 select CK_PCLK2 10: CK_DTM_CAN5 select CK_AHB 11: CK_DTM_CAN5 select CK_HXTAL
19:18	DTM_CAN4SEL[1:0]	CK_DTM_CAN4 clock source selection This bit is set and reset by software. 00: CK_DTM_CAN4 select CK_IRC48M 01: CK_DTM_CAN4 select CK_PCLK2 10: CK_DTM_CAN4 select CK_AHB 11: CK_DTM_CAN4 select CK_HXTAL
17:16	DTM_CAN3SEL[1:0]	CK_DTM_CAN3 clock source selection This bit is set and reset by software. 00: CK_DTM_CAN3 select CK_IRC48M 01: CK_DTM_CAN3 select CK_PCLK2 10: CK_DTM_CAN3 select CK_AHB 11: CK_DTM_CAN3 select CK_HXTAL
15:14	DTM_CAN2SEL[1:0]	CK_DTM_CAN2 clock source selection This bit is set and reset by software. 00: CK_DTM_CAN2 select CK_IRC48M 01: CK_DTM_CAN2 select CK_PCLK2 10: CK_DTM_CAN2 select CK_AHB 11: CK_DTM_CAN2 select CK_HXTAL
13:12	DTM_CAN1SEL[1:0]	CK_DTM_CAN1 clock source selection This bit is set and reset by software. 00: CK_DTM_CAN1 select CK_IRC48M 01: CK_DTM_CAN1 select CK_PCLK2 10: CK_DTM_CAN1 select CK_AHB 11: CK_DTM_CAN1 select CK_HXTAL
11:10	DTM_CAN0SEL[1:0]	CK_DTM_CAN0 clock source selection This bit is set and reset by software. 00: CK_DTM_CAN0 select CK_IRC48M

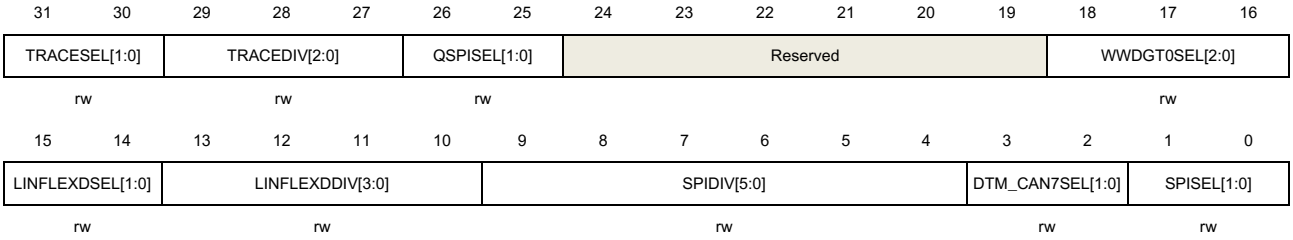
		01: CK_DTM_CAN0 select CK_PCLK2 10: CK_DTM_CAN0 select CK_AHB 11: CK_DTM_CAN0 select CK_HXTAL
9:0	Reserved	Must be kept at reset value

5.4.12. Configuration register 3 (RCU_CFG3)

Address offset: 0x34

Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)



Bits	Fields	Descriptions
31:30	TRACESEL[1:0]	TRACE clock select clock source 00: IRC48M 01: HXTAL 10: CK_PLL_P 11: CK_PLL_Q
29:27	TRACEDIV[2:0]	TRACE clock divider / (TRACEDIV +1) 000: CK_TRACE is not divided 001: CK_TRACE is divided by 2 010: CK_TRACE is divided by 3 011: CK_TRACE is divided by 4 100: CK_TRACE is divided by 5 101: CK_TRACE is divided by 6 110: CK_TRACE is divided by 7 111: CK_TRACE is divided by 8
26:25	QSPISEL[1:0]	QSPI CLK select clock source 00: IRC48M 01: HCLK 10: CK_PLL_P 11: CK_PLL_Q
24:19	Reserved	Must be kept at reset value
18:16	WWDGT0SEL[2:0]	WWDGT0 clock select 000: IRC48M

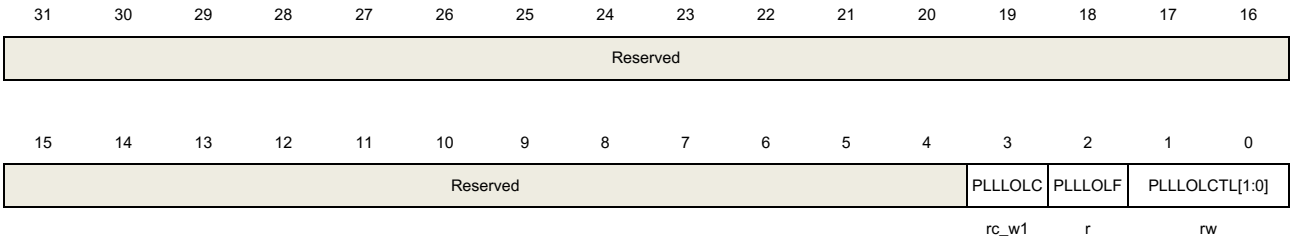
		001: CK_LXTAL 010: IRC40K 011: HXTAL 100: CK_SYS 101: CK_APB1 **Note:** The maximum frequency of WWDGT0 input clock does not exceed 80MHz
15:14	LINFLEXDSEL[1:0]	LINFLEXD clock select 00: IRC48M 01: HXTAL 10: CK_PLL_P 11: CK_SYS
13:10	LINFLEXDDIV[3:0]	LINFlexD clock divider /(LINFLEXDDIV +1) 0000: CK_LINFLEXD is not divided 0001: CK_LINFLEXD is divided by 2 0010: CK_LINFLEXD is divided by 3 0011: CK_LINFLEXD is divided by 4 0100: CK_LINFLEXD is divided by 5 0101: CK_LINFLEXD is divided by 6 0110: CK_LINFLEXD is divided by 7 0111: CK_LINFLEXD is divided by 8 1000: CK_LINFLEXD is divided by 9 1001: CK_LINFLEXD is divided by 10 1010: CK_LINFLEXD is divided by 11 1011: CK_LINFLEXD is divided by 12 1100: CK_LINFLEXD is divided by 13 1101: CK_LINFLEXD is divided by 14 1110: CK_LINFLEXD is divided by 15
9:4	SPIDIV[5:0]	SPI clock divider /(SPIDIV +1) 000000: CK_SPI is not divided 000001: CK_SPI is divided by 2 … 111110: CK_SPI is divided by 63 111111: CK_SPI is divided by 64
3:2	DTM_CAN7EL[1:0]	CK_DTM_CAN7clock source selection This bit is set and reset by software. 00: CK_DTM_CAN7select CK_IRC48M 01: CK_DTM_CAN7select CK_APB2 10: CK_DTM_CAN7select CK_AHB 11: CK_DTM_CAN7select CK_HXTAL
1:0	SPISEL[1:0]	SPI CLK select clock source

00: CK_APB1
01: CK_APB2
10: CK_IRC48M
11: CK_AHB

5.4.13. PLL loss of lock register (RCU_PLLLOL)

Address offset: 0x38
Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit)



Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value
3	PLLLOLFC	Clear PLLLOLF flag 0: PLLLOLF is not cleared. 1: PLLLOLF is cleared.
2	PLLLOLF	Loss of lock detector flag. 0: No loss of lock detected. 1: Loss of lock detected. **Note**: When clearing the PLL loss-of-lock interrupt flag, the LOPRSTF flag bit in the RCU_SYSRSTSTAT register and the PLLLOLF flag in the RCU_PLLLOL register must be cleared.
1:0	PLLLOLCTL[1:0]	Adjust the loss of lock accuracy. 00: Unlock range = expected value ± 5 01: Unlock range = expected value ± 9 10: Unlock range = expected value ± 17 11: Unlock range = expected value ± 33 **Note**: After PLL is started, every 100 reference clock cycles is taken as a time window. In this time window, the feedback clock is counted. When the counting result is 100±X cycles, PLL is judged to be locked and PLLSTB is set to 1. If the count result is any other value, PLL is judged to be unlocked or out of lock, and PLLSTB remains 0.

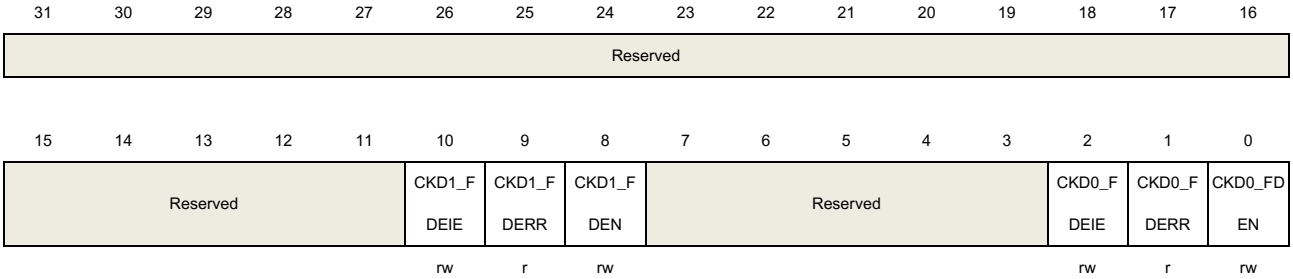
5.4.14. Clock detect control register(RCU_CKDCTL)

Address offset: 0x3C

Reset value: 0x0000 0000

The CKDx module uses HXTAL as the reference clock to monitor other clocks. The monitoring clock of CKDx(x=0.1) can refer to the clock tree.

The register can be accessed in bytes (8 bits), half words (16 bits), or words (32 bits).



Bits	Fields	Descriptions
31:11	Reserved	Must be kept at reset value
10	CKD1_FDEIE	Frequency detect error interrupt enable 0: Disable FDE synchronous interrupt at the module boundary. 1: Enable FDE synchronous interrupt at the module boundary.
9	CKD1_FDERR	Frequency detect error 0: Frequency detect normal 1: Frequency detect error occurs
8	CKD1_FDEN	Frequency detect enable 0: Stops detect monitor 1: Starts detect monitor
7:3	Reserved	Must be kept at reset value
2	CKD0_FDEIE	Frequency detect error interrupt enable 0: Disable FDE synchronous interrupt at the module boundary. 1: Enable FDE synchronous interrupt at the module boundary.
1	CKD0_FDERR	Frequency detect error 0: Frequency detect normal 1: Frequency detect error occurs
0	CKD0_FDEN	Frequency detect enable 0: Stops detect monitor 1: Starts detect monitor

5.4.15. Clock frequency monitor x control register (RCU_CKFMxCTL, x=0..4)

Address offset: 0x40+0x10*x
Reset value: 0x0000 0000

The CKFMx module uses IRC48M as the reference clock to monitor other clocks, and the

monitoring clock of CKFMx(x=0..4) can refer to the clock tree.

The register can be accessed in bytes (8 bits), half words (16 bits), or words (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									FHMAXSRE	FLMINSRE	FHMAXIE	FLMINIE	FHMAX	FLMIN	FME
									rw	rw	rw	rw	rc_w1	rc_w1	rw

Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value
6	FHMAXSRE	Frequency higher than FHMAX system reset enable (only for CKFM0) 0: Disable frequency higher than FHMAX reset. 1: Enable frequency higher than FHMAX reset.
5	FLMINSRE	Frequency lower than FLMIN system reset enable (only for CKFM0) 0: Disable frequency lower than FLMIN reset. 1: Enable frequency lower than FLMIN reset.
4	FHMAXIE	Frequency higher than FHMAX interrupt enable (for CKFM0 \ CKFM3 \ CKFM4) 0: Disable frequency higher than FHMAX interrupt. 1: Enable frequency higher than FHMAX interrupt. Frequency higher than FHMAX system reset enable (for CKFM1 \ CKFM2) 0: Disable frequency higher than FHMAX reset 1: Enable frequency higher than FHMAX reset
3	FLMINIE	Frequency lower than FLMIN interrupt enable (for CKFM0\ CKFM3 \ CKFM4) 0: Disable frequency lower than FLMIN interrupt. 1: Enable frequency lower than FLMIN interrupt. Frequency lower than FLMIN system reset enable (for CKFM1 \ CKFM2) 0: Disable frequency lower than FLMIN reset 1: Enable frequency lower than FLMIN reset
2	FHMAX	Frequency higher than high frequency reference threshold event status Hardware writes 1 to FHMAX when the monitored clock frequency becomes higher than the high threshold value (refer to HFMAX[23:0]). FHMAX clears when software writes 1. 0: No FHMAX event 1: FHMAX event occurred
1	FLMIN	Frequency lower than low frequency reference threshold event status Hardware writes 1 to FLMIN when the monitored clock frequency becomes lower than the low threshold value in (refer to LFMAX[23:0]). FLMIN clears when software writes 1.

		0: No FLMIN event 1: FLMIN event occurred
0	FME	Frequency monitor Enable 0: Stops frequency monitor 1: Starts frequency monitor

5.4.16. Clock frequency monitor x reference count configuration register (RCU_CKFMxRCCFG, x=0..4)

Address offset: 0x44+0x10*x
Reset value:0x0000 0000

The CKFMx module uses IRC48M as the reference clock to monitor other clocks, and the monitoring clock of CKFMx (x=0..4) can refer to the clock tree.

The register can be accessed in bytes (8 bits), half words (16 bits), or words (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REF_CNT[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	REF_CNT[15:0]	Reference clock count Total number of counts of reference_clock for which frequency check runs. This field defines the duration of one frequency check window. Parameter calculation: $REF_CNT_{minimum}$=Ceiling value of $MAX\left[3\times\left(\frac{f_{ref_ck}}{2f_{bus_ck}}\right), 8+5\times\left(\frac{f_{ref_ck}}{2f_{mon_ck}}\right)\right]$ where f_{ref_ck} is the reference clock frequency, f_{bus_ck} is the bus clock frequency, f_{mon_ck} is the monitored clock frequency.

5.4.17. CKFMx high threshold configuration register (RCU_CKFMxHTHCFG, x=0..4)

Address offset: 0x48+0x10*x
Reset value: 0x0000 0000

The CKFMx module uses IRC48M as the reference clock to monitor other clocks, and the monitoring clock of CKFMx (x=0..4) can refer to the clock tree.

The register can be accessed in bytes (8 bits), half words (16 bits), or words (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								HFMAX[23:16]							
								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HFMAX[15:0]															
rw															

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value
23:0	HFMAX[23:0]	High frequency reference threshold HFMAX determines the high reference value for the monitored clock frequency. HFMAX calculation: $$HFMAX=\text{Ceiling value of}\left[\frac{f_{mon_ck}\times(1+\Delta_{mon_ck})}{f_{ref_ck}\times(1-\Delta_{ref_ck})}\times REF_CNT\right]+\Delta_{CKM}$$ where f_{mon_ck} is the theoretical monitored clock frequency, f_{ref_ck} is the theoretical reference clock frequency, Δ_{mon_ck} is the variation of the monitored clock frequency, Δ_{ref_ck} is the variation of the reference clock, Δ_{CKM} is the maximum estimation deviation of the monitored clock frequency by the CKFM, Δ_{CKM} =3, REF_CNT is refer to the REF_CNT[15:0] bits in RCU_CKFMxRCCFG register.

5.4.18. CKFMx low threshold configuration register (RCU_CKFMxLTHCFG, x=0..4)

Address offset: 0x4C+0x10*x
Reset value: 0x0000 0000

The CKFMx module uses IRC48M as the reference clock to monitor other clocks, and the monitoring clock of CKFMx (x=0..4) can refer to the clock tree.

The register can be accessed in bytes (8 bits), half words (16 bits), or words (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								LFMIN[23:16]							
								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LFMIN[15:0]															
rw															

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value
23:0	LFMIN[23:0]	Low Frequency Reference Threshold LFMIN determines the low reference value for the monitored clock frequency. LFMIN calculation:

$$LFMIN=Ceiling\ value\ of\left[\frac{f_{mon_ck}\times(1-\Delta_{mon_ck})}{f_{ref_ck}\times(1+\Delta_{ref_ck})}\times REF_CNT\right]-\Delta_{CKM}$$

where f_{mon_ck} is the theoretical monitored clock frequency, f_{ref_ck} is the theoretical reference clock frequency, Δ_{mon_ck} is the variation of the monitored clock frequency, Δ_{ref_ck} is the variation of the reference clock, Δ_{CKM} is the maximum estimation deviation of the monitored clock frequency by the CKFM, Δ_{CKM} =3, REF_CNT is refer to the REF_CNT[15:0] bits in RCU_CKFMxRCCFG register.

5.4.19. CCTL TMR register (RCU_BUSIM_TMR)

Address offset: 0x90

Reset value: 0x0000 000F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TMR_ST3	TMR_ST2	TMR_ST1	TMR_ST0	TMR_EN3	TMR_EN2	TMR_EN1	TMR_EN0
								rc_w1	rc_w1	rc_w1	rc_w1	rw	rw	rw	rw

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value
7	TMR_ST3	The TMR error occurs in any bit of TRIM register 0: Normal 1: Error
6	TMR_ST2	CLK monitor enable register TMR error status The TMR error occurs in the bit CKMEN in RCU_CTL register 0: Normal 1: Error
5	TMR_ST1	Clock enable TMR error status The TMR error occurs in any bit of the FMUEN bit in RCU_AHBEN register, the PLLSEL bit in RCU_CFG0 register and the STBIRC48MCTL bit in PMU_CFG register. 0: Normal 1: Error
4	TMR_ST0	System clock switching TMR error status The TMR error occurs in the SCS[1:0] bits of RCU_CFG0 register. 0: Normal 1: Error
3	TMR_EN3	TMR error 3 output to fmu enable

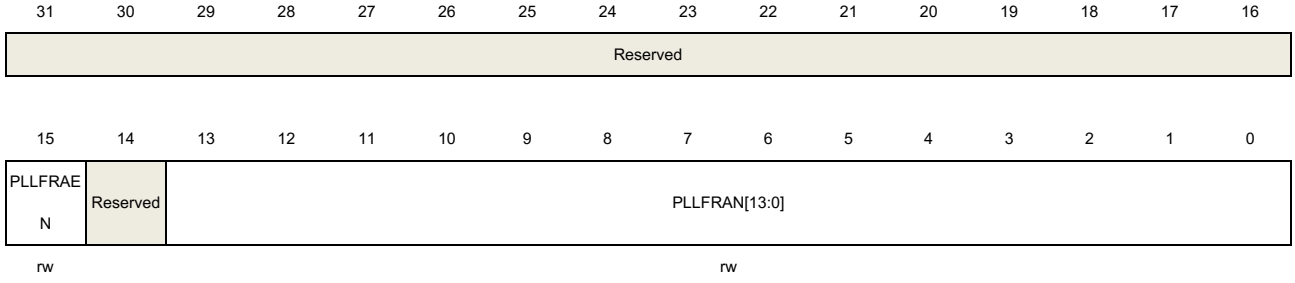
		any bit of TRIM register TMR error output enable. 0: Disable 1: Enable
2	TMR_EN2	TMR error 2 output to fmu enable any bit of the CKMEN in RCU_CTL register TMR error output enable. 0: Disable 1: Enable
1	TMR_EN1	TMR error 1 output to fmu enable any bit of the FMUEN bit in RCU_AHBEN register, the PLLSEL bit in RCU_CFG0 register and the STBIRC48MCTL bit in PMU_CFG register TMR error output enable. 0: Disable 1: Enable
0	TMR_EN0	TMR error 0 output to fmu enable The SCS[1:0] bits of RCU_CFG0 register TMR error output enable. 0: Disable 1: Enable

5.4.20. PLL fraction configuration register (RCU_PLLFRA)

Address offset: 0x98

Reset value: 0x0000 0000

The register can be accessed in bytes (8 bits), half words (16 bits), or words (32 bits).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15	PLLFRAEN	PLL fractional latch enable. 0：Enable integer frequency multiplication mode, disable spread spectrum and fractional frequency multiplication. 1：Enable fractional frequency multiplication
14	Reserved	Must be kept at reset value
13:0	PLLFRACN[13:0]	Fractional part of the multiplication factor for PLL VCO These bits are set and reset by software. These bits can control the fractional part

of the multiplication for PLL VCO. This bit field can be modified dynamically to fine-tune the PLL VCO.

If you want to change the PLLFRACN[13:0] value in real time with PLL enabled, the application must do the following:

- Set bit PLLFRAEN to '0',
- Write the new small value to PLLFRACN[13:0],
- Set bit PLLFRAEN to '1'.

5.4.21. PLL clock spread spectrum control register(RCU_PLLSSCTL)

Address offset: 0x9C

Reset value: 0x0000 0000

Spread spectrum modulation is only applicable to master PLL clocks.

The RCU_PLLSSCTL register can be written only if PLL is disabled.

This register is used to configure the generation of PLL spread spectrum clock according to the following formula:

$$MODCNT = round(f_{PLLIN}/2/f_{mod})$$

$$MODSTEP = round(mdamp*PLLMF*2^{14}/(MODCNT*100))$$

f_{PLLIN} represents PLL input clock frequency after PREDV, f_{mod} represents spread-spectrum modulation frequency, mdamp represents spread-spectrum modulation amplitude (expressed in percentage), and PLLMF represents PLL clock frequency doubling factor ($PLLMF=PLLN+\frac{PLLFRAN+0.5}{2^{14}}$).

The register can be accessed in bytes (8 bits), half words (16 bits), or words (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SSCGBYP	SS_TYPE	Reserved							MODSTEP[9:3]						
rw	rw								rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODSTEP[2:0]			Reserved		MODCNT[10:0]										
rw					rw										

Bits	Fields	Descriptions
31	SSCGBYP	PLL spread spectrum modulation is enabled 0: Disable spread spectrum modulation bypass mode 1: Enable spread spectrum modulation bypass mode
30	SS_TYPE	PLL spread spectrum modulation type selection 0: Select center spread spectrum 1: Select down spread spectrum

29:23	Reserved	Must be kept at reset value
22:13	MODSTEP[9:0]	These bits are configured with PLL spread spectrum modulation curve amplitude The following conditions must be met: MODSTEP*MODCNT≤2^{14}
12:11	Reserved	Must be kept at reset value
10:0	MODCNT[10:0]	These bits are configured with PLL spread spectrum modulation curve frequency. The following conditions must be met: MODSTEP*MODCNT≤2^{14}

6. Interrupt / event controller (EXTI)

6.1. Overview

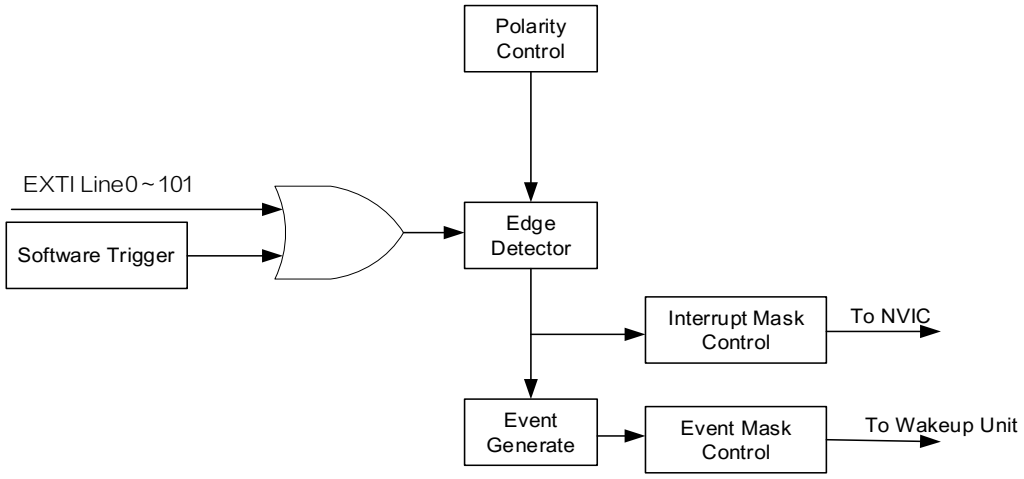
EXTI (interrupt / event controller) contains multiple independent edge detectors and generates interrupt requests or events to the processer. The EXTI has three trigger types: rising edge, falling edge and both edges. Each edge detector in the EXTI can be configured and masked independently.

6.2. Characteristics

- Wake up system from power saving mode
- GD32A711/A712 features up to 83 independent edge detectors
- Three trigger types: rising, falling and both edges
- Software interrupt or event trigger
- Trigger sources configurable

6.3. External interrupt and event (EXTI) block diagram

Figure 6-1. Block diagram of EXTI



6.4. External interrupt and event function overview

The EXTI trigger source includes 77 external lines from GPIO pins and 6 lines from internal modules(including LVD,HVD, RTC Alarm, CMPx output, FWDGT Timeout) which refers to ***Table 6-1. GD32A711/A712 EXTI source***. The RTC Alarm, CMPx output and FWDGT Timeout can wakeup standby mode. All GPIO pins can be selected as an EXTI trigger source by configuring SYSCFG_EXTISSx registers in SYSCFG module (please refer to ***System***

configuration registers definition for detail). The time interval between two consecutive input triggers must be greater than the EXTI bus clock frequency * 2.

The EXTI adds pad input signal from pad for standby wakeup. As shown in below, there are 61 IO(EXTI42~EXTI101 and EXTI18) inputs. OR operations can be performed independently through edge detector on these 61 IOs to wake up the processor from standby mode.

Note: If the IO pin configuration is falling edge wake-up for standby mode, the pad retain function of the IO pin needs to be enabled.

EXTI can provide not only interrupts but also event signals to the processor. The Cortex®-M7 processor fully implements the Wait For Interrupt (WFI), Wait For Event (WFE) and the Send Event (SEV) instructions. The Wake-up Interrupt Controller (WIC) enables the processor and NVIC to be put into a very low-power mode, and leaving the WIC to identify and prioritize interrupts and events. EXTI can be used to wake up processor and the whole system when some expected event occurs, such as a special GPIO pin toggling or RTC alarm.

EXTI will provide the CPU with an NMI signal sourced from PJ7 when NMIPINIEx bit of SYSCFG_CFG register is enable.

Hardware trigger

Hardware trigger may be used to detect the voltage change of external or internal signals. The software should follow these steps to use this function:

1. Configure EXTI sources in SYSCFG module based on application requirement.
2. Configure EXTI_RTEN and EXTI_FTEN to enable the rising or falling detection on related pins. (Software may set both RTENx and FTENx for a pin at the same time to detect both rising and falling changes on this pin).
3. Enable interrupts or events by setting related EXTI_INTEN or EXTI_EVEN bits.
4. EXTI starts to detect changes on the configured pins. The related interrupt or event will be triggered when desired change is detected on these pins. If the interrupt is triggered, the related PDx is set; if the event is triggered, the related PDx is not set. The software should response to the interrupts or events and clear these PDx bits.

Software trigger

Software may also trigger EXTI interrupts or events following these steps:

1. Enable interrupts or events by setting related EXTI_INTEN or EXTI_EVEN bits.
2. Configure EXTI_RTEN and EXTI_FTEN to enable the rising or falling detection on related pins. (Software may set both RTENx and FTENx for a pin at the same time to detect both rising and falling changes on this pin).
3. Set SWIEVx bits in EXTI_SWIEV register, If SWIEVx bits is set from 0 to 1 and EXTI_RTEN is enable or If SWIEVx bits is set from 1 to 0 and EXTI_FTEN is enable, the related interrupt or event will be triggered immediately. If the interrupt is triggered, the related PDx is set; if the event is triggered, the related PDx is not set. Software should response to these interrupts, and clear related PDx bits.

Table 6-1. GD32A711/A712 EXTI source

EXTI Line Number	Source
0	PA0 / PB0 / PC0 / PD0 / PE0 / PF0 / PG0 / PI0 / PJ0 / PK0 / PL0 / PM0 / PN0
1	PA1 / PB1 / PC1 / PD1 / PE1 / PF1 / PG1 / PI1 / PJ1 / PK1 / PL1 / PM1 / PN1
2	PA2 / PB2 / PC2 / PD2 / PE2 / PF2 / PG2 / PH2 / PI2 / PJ2 / PK2 / PL2 / PM2 / PN2
3	PA3 / PB3 / PC3 / PD3 / PE3 / PF3 / PG3 / PH3 / PI3 / PJ3 / PK3 / PL3 / PM3 / PN3
4	PA4 / PB4 / PC4 / PD4 / PE4 / PF4 / PG4 / PH4 / PI4 / PJ4 / PK4 / PL4 / PM4 / PN4
5	PA5 / PB5 / PC5 / PD5 / PE5 / PF5 / PG5 / PH5 / PI5 / PJ5 / PK5 / PL5 / PM5 / PN5
6	PA6 / PB6 / PC6 / PD6 / PE6 / PF6 / PG6 / PH6 / PI6 / PJ6 / PK6 / PL6 / PM6 / PN6
7	PA7 / PB7 / PC7 / PD7 / PE7 / PF7 / PG7 / PH7 / PI7 / PJ7 / PK7 / PL7 / PM7 / PN7
8	PA8 / PB8 / PC8 / PD8 / PE8 / PF8 / PG8 / PH8 / PI8 / PJ8 / PK8 / PL8 / PM8 / PN8
9	PA9 / PB9 / PC9 / PD9 / PE9 / PF9 / PG9 / PH9 / PI9 / PJ9 / PK9 / PL9 / PM9 / PN9
10	PA10 / PB10 / PC10 / PD10 / PE10 / PF10 / PG10 / PH10 / PI10 / PJ10 / PK10 / PL10 / PM10 / PN10
11	PA11 / PB11 / PC11 / PD11 / PE11 / PF11 / PG11 / PH11 / PI11 / PJ11 / PK11 / PL11 / PM11 / PN11
12	PA12 / PB12 / PC12 / PD12 / PE12 / PF12 / PG12 / PH12 / PI12 / PJ12 / PK12 / PL12 / PM12
13	PA13 / PB13 / PC13 / PD13 / PE13 / PF13 / PG13 / PH13 / PI13 / PJ13 / PK13 / PL13 / PM13
14	PA14 / PB14 / PC14 / PD14 / PE14 / PF14 / PG14 / PH14 / PI14 / PJ14 / PK14 / PL14 / PM14
15	PA15 / PB15 / PC15 / PD15 / PE15 / PF15 / PG15 / PH15 / PI15 / PJ15 / PK15 / PL15 / PM15
16	LVD
17	RTC Alarm
18	PA2
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	CMP0 output
27	CMP1 output
28	Reserved
29	Reserved
30	Reserved
31	Reserved
32	Reserved
33	Reserved
34	Reserved
35	Reserved
36	Reserved
37	Reserved

EXTI Line Number	Source
38	Reserved
39	Reserved
40	FWDGT0 Timeout
41	HVD
42	PA8
43	PA10
44	PA11
45	PA12
46	PB5
47	PB12
48	PB15
49	PC1
50	PC8
51	PC9
52	PL4
53	PC15
54	PD8
55	PD9
56	PD10
57	PD13
58	PD14
59	PE4
60	PE5
61	PE8
62	PE10
63	PE11
64	PE12
65	PE13
66	PE14
67	PF8
68	PF10
69	PF12
70	PF13
71	PG0
72	PG2
73	PG4
74	PG5
75	PG7
76	PG8
77	PH7
78	PH9

EXTI Line Number	Source
79	PH12
80	PH13
81	PH14
82	PI10
83	PI11
84	PI14
85	PI15
86	PJ3
87	PJ5
88	PJ6
89	PJ7
90	PJ8
91	PJ9
92	PJ11
93	PK1
94	PL2
95	PL3
96	PM0
97	PM1
98	PM3
99	PN1
100	PN3
101	PN4

6.5. Register definition

EXTI base address: 0x4001 0400

Table 6-2. EXTI register overview

Register name	Symbol	Address offset	Reset value
Interrupt enable register 0	EXTI_INTEN0	0x00	0x0000 0000
Event enable register 0	EXTI_EVEN0	0x04	0x0000 0000
Rising edge trigger enable register 0	EXTI_RTEN0	0x08	0x0000 0000
Falling edge trigger enable register 0	EXTI_FTEN0	0x0C	0x0000 0000
Software interrupt event register 0	EXTI_SWIEV0	0x10	0x0000 0000
Pending register 0	EXTI_PD0	0x14	0xXXXX XXXX X indicates undefined
Interrupt enable register 1	EXTI_INTEN1	0x18	0x0000 0000
Event enable register 1	EXTI_EVEN1	0x1C	0x0000 0000
Rising edge trigger enable register 1	EXTI_RTEN1	0x20	0x0000 0000
Falling edge trigger enable register 1	EXTI_FTEN1	0x24	0x0000 0000
Software interrupt event register 1	EXTI_SWIEV1	0x28	0x0000 0000
Pending register 1	EXTI_PD1	0x2C	0x0000 00XX X indicates undefined
Interrupt enable register 2	EXTI_INTEN2	0x30	0x0000 0000
Event enable register 2	EXTI_EVEN2	0x34	0x0000 0000
Rising edge trigger enable register 2	EXTI_RTEN2	0x38	0x0000 0000
Falling edge trigger enable register 2	EXTI_FTEN2	0x3C	0x0000 0000
Software interrupt event register 2	EXTI_SWIEV2	0x40	0x0000 0000
Pending register 2	EXTI_PD2	0x44	0x0000 00XX X indicates undefined
Interrupt enable register 3	EXTI_INTEN3	0x48	0x0000 0000
Event enable register 3	EXTI_EVEN3	0x4C	0x0000 0000
Rising edge trigger enable register 3	EXTI_RTEN3	0x50	0x0000 0000
Falling edge trigger enable register 3	EXTI_FTEN3	0x54	0x0000 0000
Software interrupt event register 3	EXTI_SWIEV3	0x58	0x0000 0000
Pending register 3	EXTI_PD3	0x5C	0xXXXX XXXX X indicates undefined

6.5.1. Interrupt enable register 0 (EXTI_INTEN0)

For GD32A711/A712/ devices

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				INTEN27	INTEN26	Reserved							INTEN18	INTEN17	INTEN16
				rw	rw								rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTEN15	INTEN14	INTEN13	INTEN12	INTEN11	INTEN10	INTEN9	INTEN8	INTEN7	INTEN6	INTEN5	INTEN4	INTEN3	INTEN2	INTEN1	INTEN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:26	INTENx	Interrupt enable bit x (x = 26…27) 0: Interrupt from linex is disabled. 1: Interrupt from linex is enabled.
25:19	Reserved	Must be kept at reset value.
18:0	INTENx	Interrupt enable bit x (x = 0…18) 0: Interrupt from linex is disabled. 1: Interrupt from linex is enabled.

6.5.2. Event enable register 0 (EXTI_EVEN0)

For GD32A711/A712/ devices

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				EVEN27	EVEN26	Reserved							EVEN18	EVEN17	EVEN16
				rw	rw								rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVEN15	EVEN14	EVEN13	EVEN12	EVEN11	EVEN10	EVEN9	EVEN8	EVEN7	EVEN6	EVEN5	EVEN4	EVEN3	EVEN2	EVEN1	EVEN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:26	EVENx	Event enable bit x (x = 26…27) 0: Event from linex is disabled 1: Event from linex is enabled
25:19	Reserved	Must be kept at reset value.
18:0	EVENx	Event enable bit x (x = 0…18) 0: Event from linex is disabled

1: Event from linex is enabled

6.5.3. Rising edge trigger enable register 0 (EXTI_RTEN0)

For GD32A711/A712/ devices

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				RTEN27	RTEN26	Reserved							RTEN18	RTEN17	RTEN16
				rw	rw								rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTEN15	RTEN14	RTEN13	RTEN12	RTEN11	RTEN10	RTEN9	RTEN8	RTEN7	RTEN6	RTEN5	RTEN4	RTEN3	RTEN2	RTEN1	RTEN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:26	RTENx	Rising edge trigger enable bit x (x =26…27) 0: Rising edge of linex is invalid 1: Rising edge of linex is valid as an interrupt / event request
25:19	Reserved	Must be kept at reset value.
18:0	RTENx	Rising edge trigger enable bit x (x = 0…18) 0: Rising edge of linex is invalid 1: Rising edge of linex is valid as an interrupt / event request

6.5.4. Falling edge trigger enable register 0 (EXTI_FTEN0)

For GD32A711/A712/ devices

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				FTEN27	FTEN26	Reserved							FTEN18	FTEN17	FTEN16
				rw	rw								rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FTEN15	FTEN14	FTEN13	FTEN12	FTEN11	FTEN10	FTEN9	FTEN8	FTEN7	FTEN6	FTEN5	FTEN4	FTEN3	FTEN2	FTEN1	FTEN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions

31:28	Reserved	Must be kept at reset value.
27:26	FTENx	Falling edge trigger enable bit x (x = 26…27) 0: Falling edge of linex is invalid 1: Falling edge of linex is valid as an interrupt / event request
25:19	Reserved	Must be kept at reset value.
18:0	FTENx	Falling edge trigger enable bit x (x = 0…18) 0: Falling edge of linex is invalid 1: Falling edge of linex is valid as an interrupt / event request

6.5.5. Software interrupt event register 0 (EXTI_SWIEV0)

For GD32A711/A712/ devices

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				SWIEV27	SWIEV26	Reserved							SWIEV18	SWIEV17	SWIEV16
				rw	rw								rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWIEV15	SWIEV14	SWIEV13	SWIEV12	SWIEV11	SWIEV10	SWIEV9	SWIEV8	SWIEV7	SWIEV6	SWIEV5	SWIEV4	SWIEV3	SWIEV2	SWIEV1	SWIEV0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:26	SWIEVx	Interrupt / event software trigger bit x (x = 26…27) 0: Deactivate the EXTIx software interrupt / event request 1: Activate the EXTIx software interrupt / event request
25:19	Reserved	Must be kept at reset value.
18:0	SWIEVx	Interrupt / event software trigger bit x (x = 0…18) 0: Deactivate the EXTIx software interrupt / event request 1: Activate the EXTIx software interrupt / event request

6.5.6. Pending register 0 (EXTI_PD0)

For GD32A711/A712/ devices

Address offset: 0x14

Reset value: 0xXXXX XXXX where X is undefined.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				PD27	PD26	Reserved							PD18	PD17	PD16
				rc_w1	rc_w1								rc_w1	rc_w1	rc_w1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD15	PD14	PD13	PD12	PD11	PD10	PD9	PD8	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:26	PDx	Interrupt pending status bit x (x = 26…27) 0: EXTI linex is not triggered 1: EXTI linex is triggered. This bit is cleared to 0 by writing 1 to it.
25:19	Reserved	Must be kept at reset value.
18:0	PDx	Interrupt pending status bit x (x = 0…18) 0: EXTI linex is not triggered 1: EXTI linex is triggered. This bit is cleared to 0 by writing 1 to it.

6.5.7. Interrupt enable register 1 (EXTI_INTEN1)

For GD32A711/A712/ devices

Address offset: 0x18

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						INTEN41	INTEN40	Reserved							
						rw	rw								

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:8	INTENx	Interrupt enable bit x (x = 40…41) 0: Interrupt from linex is disabled. 1: Interrupt from linex is enabled.
7:0	Reserved	Must be kept at reset value.

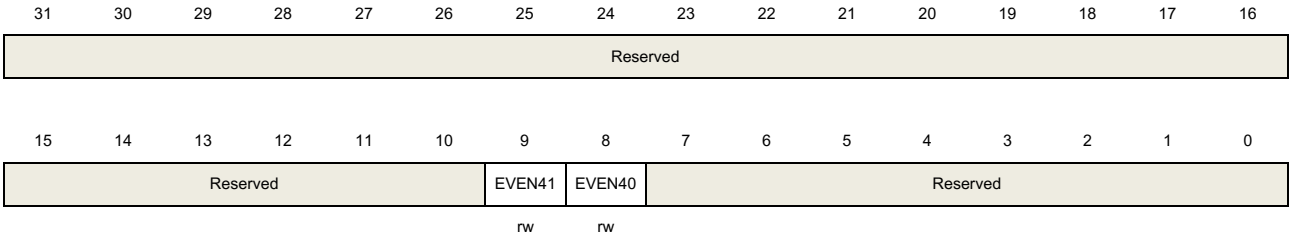
6.5.8. Event enable register 1 (EXTI_EVEN1)

For GD32A711/A712/ devices

Address offset: 0x1C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:8	EVENx	Event enable bit x (x = 40…41) 0: Event from linex is disabled. 1: Event from linex is enabled.
7:0	Reserved	Must be kept at reset value.

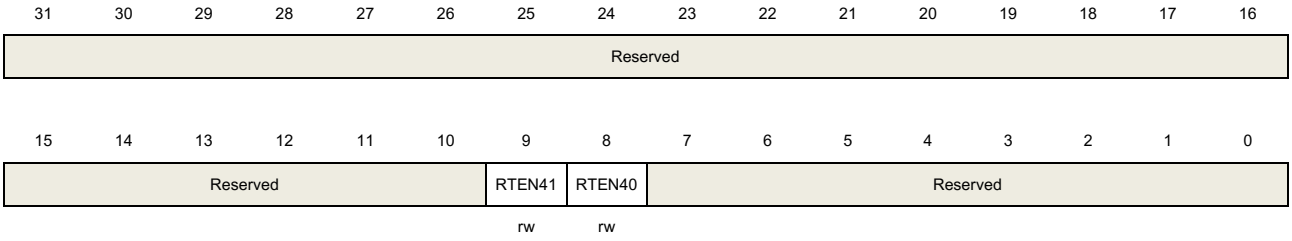
6.5.9. Rising edge trigger enable register 1 (EXTI_RTEN1)

For GD32A711/A712/ devices

Address offset: 0x20

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:8	RTENx	Rising edge trigger enable bit x (x = 40…41) 0: Rising edge of linex is invalid 1: Rising edge of linex is valid as an interrupt / event request

7:0	Reserved	Must be kept at reset value.

6.5.10. Falling edge trigger enable register 1 (EXTI_FTEN1)

For GD32A711/A712/ devices

Address offset: 0x24

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						FTEN41	FTEN40	Reserved							
						rw	rw								

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:8	FTENx	Falling edge trigger enable bit x (x = 40…41) 0: Falling edge of linex is invalid 1: Falling edge of linex is valid as an interrupt / event request
7:0	Reserved	Must be kept at reset value.

6.5.11. Software interrupt event register 1 (EXTI_SWIEV1)

For GD32A711/A712/ devices

Address offset: 0x28

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SWIEV41	SWIEV40	Reserved							
						rw	rw								

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:8	SWIEVx	Interrupt / event software trigger bit x (x = 40…41) 0: Deactivate the EXTIx software interrupt / event request

1: Activate the EXTIx software interrupt / event request

7:0	Reserved	Must be kept at reset value.

6.5.12. Pending register 1 (EXTI_PD1)

For GD32A711/A712/ devices

Address offset: 0x2C

Reset value: 0x0000 00XX where X is undefined.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						PD41	PD40	Reserved							
						rc_w1	rc_w1								

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:8	PDx	Interrupt pending status bit x (x = 40…41) 0: EXTI linex is not triggered 1: EXTI linex is triggered. This bit is cleared to 0 by writing 1 to it.
7:0	Reserved	Must be kept at reset value.

6.5.13. Interrupt enable register 2 (EXTI_INTEN2)

Address offset: 0x30

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INTEN73	INTEN72	INTEN71	INTEN70	INTEN69	INTEN68	INTEN67	INTEN66	INTEN65	INTEN64	INTEN63	INTEN62	INTEN61	INTEN60	INTEN59	INTEN58
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTEN57	INTEN56	INTEN55	INTEN54	INTEN53	INTEN52	INTEN51	INTEN50	INTEN49	INTEN48	INTEN47	INTEN46	INTEN45	INTEN44	INTEN43	INTEN42
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:0	INTENx	Interrupt enable bit x (x = 42…73) 0: Interrupt from Linex is disabled. 1: Interrupt from Linex is enabled.

6.5.14. Event enable register 2 (EXTI_EVEN2)

Address offset: 0x34

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EVEN73	EVEN72	EVEN71	EVEN70	EVEN69	EVEN68	EVEN67	EVEN66	EVEN65	EVEN64	EVEN63	EVEN62	EVEN61	EVEN60	EVEN59	EVEN58
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVEN57	EVEN56	EVEN55	EVEN54	EVEN53	EVEN52	EVEN51	EVEN50	EVEN49	EVEN48	EVEN47	EVEN46	EVEN45	EVEN44	EVEN43	EVEN42
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:0	EVENx	Event enable bit x (x = 42…73) 0: Event from Linex is disabled. 1: Event from Linex is enabled.

6.5.15. Rising edge trigger enable register 2 (EXTI_RTEN2)

Address offset: 0x38

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RTEN73	RTEN72	RTEN71	RTEN70	RTEN69	RTEN68	RTEN67	RTEN66	RTEN65	RTEN64	RTEN63	RTEN62	RTEN61	RTEN60	RTEN59	RTEN58
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTEN57	RTEN56	RTEN55	RTEN54	RTEN53	RTEN52	RTEN51	RTEN50	RTEN49	RTEN48	RTEN47	RTEN46	RTEN45	RTEN44	RTEN43	RTEN42
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:0	RTENx	Rising edge trigger enable x (x = 42…73) 0: Rising edge of Linex is invalid 1: Rising edge of Linex is valid as an interrupt/event request

6.5.16. Falling edge trigger enable register 2 (EXTI_FTEN2)

Address offset: 0x3C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FTEN73	FTEN72	FTEN71	FTEN70	FTEN69	FTEN68	FTEN67	FTEN66	FTEN65	FTEN64	FTEN63	FTEN62	FTEN61	FTEN60	FTEN59	FTEN58
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FTEN57	FTEN56	FTEN55	FTEN54	FTEN53	FTEN52	FTEN51	FTEN50	FTEN49	FTEN48	FTEN47	FTEN46	FTEN45	FTEN44	FTEN43	FTEN42
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:0	FTENx	Falling edge trigger enable x (x = 42…73) 0: Falling edge of Linex is invalid 1: Falling edge of Linex is valid as an interrupt/event request

6.5.17. Software interrupt event register 2 (EXTI_SWIEV2)

Address offset: 0x40

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SWIEV73	SWIEV72	SWIEV71	SWIEV70	SWIEV69	SWIEV68	SWIEV67	SWIEV66	SWIEV65	SWIEV64	SWIEV63	SWIEV62	SWIEV61	SWIEV60	SWIEV59	SWIEV58
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWIEV57	SWIEV56	SWIEV55	SWIEV54	SWIEV53	SWIEV52	SWIEV51	SWIEV50	SWIEV49	SWIEV48	SWIEV47	SWIEV46	SWIEV45	SWIEV44	SWIE43	SWIEV42
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:0	SWIEVx	Interrupt/Event software trigger x (x = 42…73) 0: Deactivate the EXTIx software interrupt/event request. 1: Activate the EXTIx software interrupt/event request.

6.5.18. Pending register 2 (EXTI_PD2)

Address offset: 0x44

Reset value: undefined

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PD73	PD72	PD71	PD70	PD69	PD68	PD67	PD66	PD65	PD64	PD63	PD62	PD61	PD60	PD59	PD58
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD57	PD56	PD55	PD54	PD53	PD52	PD51	PD50	PD49	PD48	PD47	PD46	PD45	PD44	PD43	PD42
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions
31:0	PDx	Interrupt pending status x (x = 42…73) 0: EXTI Linex is not triggered 1: EXTI Linex is triggered. This bit is cleared to 0 by writing 1 to it.

6.5.19. Interrupt enable register 3 (EXTI_INTEN3)

Address offset: 0x48

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				INTEN101	INTEN100	INTEN99	INTEN98	INTEN97	INTEN96	INTEN95	INTEN94	INTEN93	INTEN92	INTEN91	INTEN90
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTEN89	INTEN88	INTEN87	INTEN86	INTEN85	INTEN84	INTEN83	INTEN82	INTEN81	INTEN80	INTEN79	INTEN78	INTEN77	INTEN76	INTEN75	INTEN74
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:0	INTENx	Interrupt enable bit x (x = 74…101) 0: Interrupt from Linex is disabled. 1: Interrupt from Linex is enabled.

6.5.20. Event enable register 3 (EXTI_EVEN3)

Address offset: 0x4c

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				EVEN101	EVEN100	EVEN99	EVEN98	EVEN97	EVEN96	EVEN95	EVEN94	EVEN93	EVEN92	EVEN91	EVEN90
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVEN89	EVEN88	EVEN87	EVEN86	EVEN85	EVEN84	EVEN83	EVEN82	EVEN81	EVEN80	EVEN79	EVEN78	EVEN77	EVEN76	EVEN75	EVEN74
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:0	EVENx	Interrupt enable bit x (x = 74…101) 0: Event from Linex is disabled. 1: Event from Linex is enabled.

6.5.21. Rising edge trigger enable register 3 (EXTI_RTEN3)

Address offset: 0x50

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				RTEN101	RTEN100	RTEN99	RTEN98	RTEN97	RTEN96	RTEN95	RTEN94	RTEN93	RTEN92	RTEN91	RTEN90
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTEN89	RTEN88	RTEN87	RTEN86	RTEN85	RTEN84	RTEN83	RTEN82	RTEN81	RTEN80	RTEN79	RTEN78	RTEN77	RTEN76	RTEN75	RTEN74
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:0	RTENx	Rising edge trigger enable x (x = 74…101) 0: Rising edge of Linex is invalid 1: Rising edge of Linex is valid as an interrupt/event request.

6.5.22. Falling edge trigger enable register 3 (EXTI_FTEN3)

Address offset: 0x54

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				FTEN101	FTEN100	FTEN99	FTEN98	FTEN97	FTEN96	FTEN95	FTEN94	FTEN93	FTEN92	FTEN91	FTEN90
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FTEN89	FTEN88	FTEN87	FTEN86	FTEN85	FTEN84	FTEN83	FTEN82	FTEN81	FTEN80	FTEN79	FTEN78	FTEN77	FTEN76	FTEN75	FTEN74
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:0	FTENx	Falling edge trigger enable x (x = 74…101) 0: Falling edge of Linex is invalid 1: Falling edge of Linex is valid as an interrupt/event request.

6.5.23. Software interrupt event register 3 (EXTI_SWIEV3)

Address offset: 0x58

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				SWIEV101	SWIEV100	SWIEV99	SWIEV98	SWIEV97	SWIEV96	SWIEV95	SWIEV94	SWIEV93	SWIEV92	SWIEV91	SWIEV90
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWIEV89	SWIEV88	SWIEV87	SWIEV86	SWIEV85	SWIEV84	SWIEV83	SWIEV82	SWIEV81	SWIEV80	SWIEV79	SWIEV78	SWIEV77	SWIEV76	SWIEV75	SWIEV74

rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:0	SWIEVx	Interrupt/Event software trigger x (x = 74…101) 0: Deactivate the EXTIx software interrupt/event request 1: Activate the EXTIx software interrupt/event request

6.5.24. Pending register 3 (EXTI_PD3)

Address offset: 0x5C

Reset value: undefined

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				PD101	PD100	PD99	PD98	PD97	PD96	PD95	PD94	PD93	PD92	PD91	PD90
				rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD89	PD88	PD87	PD86	PD85	PD84	PD83	PD82	PD81	PD80	PD79	PD78	PD77	PD76	PD75	PD74
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:0	PDx	Interrupt pending status x (x = 74…101) 0: EXTI Linex is not triggered 1: EXTI Linex is triggered. This bit is cleared to 0 by writing 1 to it.

7. Trigger selection controller (TRIGSEL)

7.1. Overview

The trigger selection controller (TRIGSEL) allows software to select the trigger input signal for various peripherals. TRIGSEL provides a flexible mechanism for a peripheral to select different trigger inputs.

With TRIGSEL, Each register can configure up to 3 TRIGSEL outputs as trigger input signals for the peripheral. Configure the corresponding register to select the different trigger signal for the specified trigger input of each peripheral.

7.2. Characteristics

- Support up to 144 trigger input signals.
- Each peripheral has its corresponding registers to select trigger input signal.
- Trigger input source could be external input signal or output of peripheral.
- Trigger selection output could be for external output or peripheral input.

7.3. Function overview

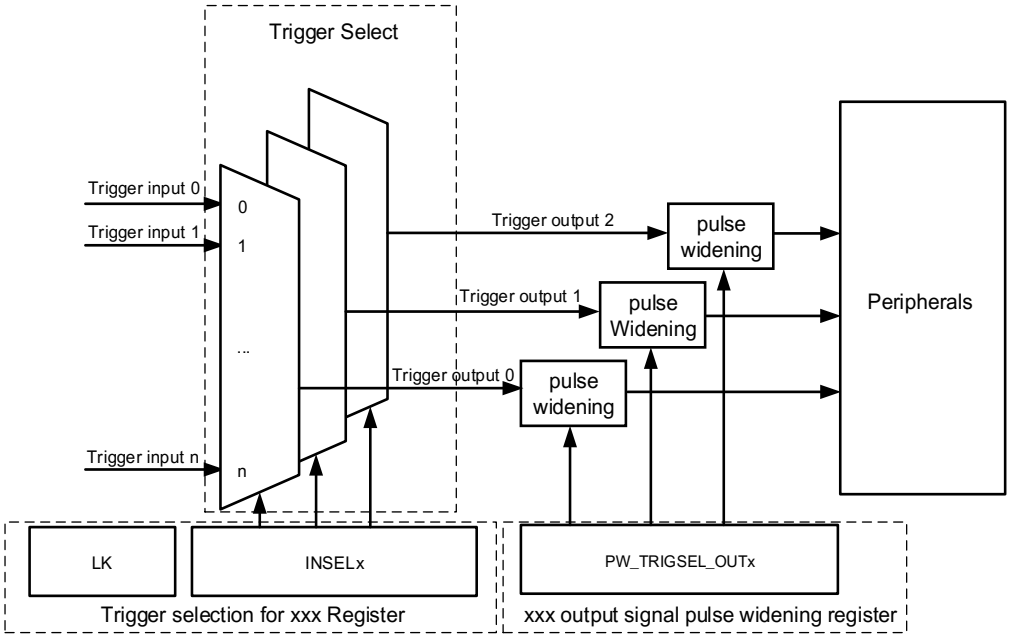
With TRIGSEL, peripherals that support trigger source selection have dedicated registers to select the trigger input source. Each register can be configured with 3 outputs, which are connected to the trigger input of the peripheral. Each output can select different trigger input sources.

TRIGSEL supports software-configurable output signal pulse widening, with a maximum of 32 HCLK clock cycles.

Note: If the time interval between multiple TRIGSEL signals is less than the configured pulse widening, only the first TRIGSEL signal is valid and subsequent TRIGSEL signals will be lost. Configure this feature properly.

The ***Figure 7-1. TRIGSEL main composition example*** shows the main composition of TRIGSEL.

Figure 7-1. TRIGSEL main composition example



7.4. Internal connect

The TRIGSEL allows software to select the trigger input for peripherals. The *Table 7-1. Trigger input bit fields selection* show the trigger input register selection.

Table 7-1. Trigger input bit fields selection

Fields	Bits value	Trigger input selection	Description
INSELx	0x00	LOGIC 0	
	0x01	LOGIC 1	
	0x02	TRIGSEL_IN0	Signals from the TRIGSEL_INx pins
	0x03	TRIGSEL_IN1	
	0x04	TRIGSEL_IN2	
	0x05	TRIGSEL_IN3	
	0x06	TRIGSEL_IN4	
	0x07	TRIGSEL_IN5	
	0x08	TRIGSEL_IN6	
	0x09	TRIGSEL_IN7	
	0x0a	TRIGSEL_IN8	
	0x0b	TRIGSEL_IN9	
	0x0c	TRIGSEL_IN10	
	0x0d	TRIGSEL_IN11	
	0x0e	TRIGSEL_IN12	
	0x0f	TRIGSEL_IN13	
	0x10	TRIGSEL_IN14	

Fields	Bits value	Trigger input selection	Description
	0x11	TRIGSEL_IN15	
	0x12	TIMER0_TRGO0	TIMERx output
	0x13	TIMER0_CH0	
	0x14	TIMER0_CH1	
	0x15	TIMER0_CH2	
	0x16	TIMER0_CH3	
	0x17	TIMER0_MCH0	
	0x18	TIMER0_MCH1	
	0x19	TIMER0_MCH2	
	0x1a	TIMER0_MCH3	
	0x1b	TIMER7_TRGO0	
	0x1c	TIMER7_CH0	
	0x1d	TIMER7_CH1	
	0x1e	TIMER7_CH2	
	0x1f	TIMER7_CH3	
	0x20	TIMER7_MCH0	
	0x21	TIMER7_MCH1	
	0x22	TIMER7_MCH2	
	0x23	TIMER7_MCH3	
	0x24	TIMER19_TRGO0	
	0x25	TIMER19_CH0	
	0x26	TIMER19_CH1	
	0x27	TIMER19_CH2	
	0x28	TIMER19_CH3	
	0x29	TIMER19_MCH0	
	0x2a	TIMER19_MCH1	
	0x2b	TIMER19_MCH2	
	0x2c	TIMER19_MCH3	
	0x2d	TIMER20_TRGO0	
	0x2e	TIMER20_CH0	
	0x2f	TIMER20_CH1	
	0x30	TIMER20_CH2	
	0x31	TIMER20_CH3	
	0x32	TIMER20_MCH0	
	0x33	TIMER20_MCH1	
	0x34	TIMER20_MCH2	
	0x35	TIMER20_MCH3	
	0x36	TIMER60_TRGO0	
	0x37	TIMER60_CH0	
	0x38	TIMER60_CH1	
	0x39	TIMER60_CH2	
	0x3a	TIMER60_CH3	

Fields	Bits value	Trigger input selection	Description
	0x3b	TIMER60_MCH0	
	0x3c	TIMER60_MCH1	
	0x3d	TIMER60_MCH2	
	0x3e	TIMER60_MCH3	
	0x3f	TIMER61_TRGO0	
	0x40	TIMER61_CH0	
	0x41	TIMER61_CH1	
	0x42	TIMER61_CH2	
	0x43	TIMER61_CH3	
	0x44	TIMER61_MCH0	
	0x45	TIMER61_MCH1	
	0x46	TIMER61_MCH2	
	0x47	TIMER61_MCH3	
	0x48	TIMER62_TRGO0	
	0x49	TIMER62_CH0	
	0x4a	TIMER62_CH1	
	0x4b	TIMER62_CH2	
	0x4c	TIMER62_CH3	
	0x4d	TIMER62_MCH0	
	0x4e	TIMER62_MCH1	
	0x4f	TIMER62_MCH2	
	0x50	TIMER62_MCH3	
	0x51	TIMER63_TRGO0	
	0x52	TIMER63_CH0	
	0x53	TIMER63_CH1	
	0x54	TIMER63_CH2	
	0x55	TIMER63_CH3	
	0x56	TIMER63_MCH0	
	0x57	TIMER63_MCH1	
	0x58	TIMER63_MCH2	
	0x59	TIMER63_MCH3	
	0x5a	TIMER1_TRGO0	
	0x5b	TIMER1_CH0	
	0x5c	TIMER1_CH1	
	0x5d	TIMER1_CH2	
	0x5e	TIMER1_CH3	
	0x5f	TIMER2_TRGO0	
	0x60	TIMER2_CH0	
	0x61	TIMER2_CH1	
	0x62	TIMER2_CH2	
	0x63	TIMER2_CH3	
	0x64	TIMER5_TRGO0	

Fields	Bits value	Trigger input selection	Description
	0x65	TIMER5_TRGO1	
	0x66	TIMER5_TRGO2	
	0x67	TIMER5_TRGO3	
	0x68	Reserved	
	0x69	TIMER6_TRGO0	
	0x6a	TIMER6_TRGO1	
	0x6b	TIMER6_TRGO2	
	0x6c	TIMER6_TRGO3	
	0x6d	Reserved	Reserved
	0x6e	CMP0_OUT	CMPx output
	0x6f	CMP1_OUT	
	0x70	BKP_Tamper	Tamper interrupt flag
	0x71	RTC_Alarm	RTC Alarm output
	0x72~0x75	Reserved	Reserved
	0x76	CM7_0_TXEV	the EVENTOUT from CM7_0
	0x77	Reserved	Reserved
	0x78	CK_OUT0	clock output
	0x79	TIMER_ETI0	Signals from TIMER_ETIx pin
	0x7a	TIMER_ETI1	
	0x7b	TIMER_ETI2	
	0x7c	CK_OUT1	clock output
	0x7d~0xa2	Reserved	Reserved
	0xa3	DAC0_eoc	DACx_eoc signals
	0xa4	DAC1_eoc	
	0xa5	ADC0_EORC	End of ADCx routine sequence conversion
	0xa6	ADC1_EORC	
	0xa7	Reserved	
	0xa8	ADC0_ECH	End of ADCx channel conversion
	0xa9	ADC1_ECH	
	0xaa	Reserved	
	0xab	ADC0_WD0_OUT	ADCx watchdog output
	0xac	ADC0_WD1_OUT	
	0xad	ADC0_WD2_OUT	
	0xae	ADC0_WD3_OUT	
	0xaf	ADC1_WD0_OUT	
	0xb0	ADC1_WD1_OUT	
	0xb1	ADC1_WD2_OUT	
	0xb2	ADC1_WD3_OUT	
	0xb3~0xb6	Reserved	
	0xb7	ADC0_EOI0C	End of ADCx inserted sequence conversion
	0xb8	ADC0_EOI1C	

Fields	Bits value	Trigger input selection	Description
	0xb9	ADC0_EOI2C	
	0xba	ADC0_EOI3C	
	0xbb	ADC0_EOI4C	
	0xbc	ADC0_EOI5C	
	0xbd	ADC1_EOI0C	
	0xbe	ADC1_EOI1C	
	0xbf	ADC1_EOI2C	
	0xc0	ADC1_EOI3C	
	0xc1	ADC1_EOI4C	
	0xc2	ADC1_EOI5C	

Table 7-2. TRIGSEL input and output mapping show the connection relationship between TRIGSEL input and output. Through the INSELx[7:0] bits of TRIGSEL register, an input trigger source can be selected for the output of TRIGSEL. Each TRIGSEL register can configure with up to 3 outputs, which are connected to the corresponding peripherals.

Table 7-2. TRIGSEL input and output mapping

Trigger Source	Trigger select	TRIGSEL register	TRIGSEL output	Peripherals
0	INSELx[7:0]	TRIGSEL_EXTOUT_0	Output0	TRIGSEL_OUT0
1			Output1	TRIGSEL_OUT1
TRIGSEL_IN0		TRIGSEL_EXTOUT_1	Output0	TRIGSEL_OUT2
TRIGSEL_IN1			Output1	TRIGSEL_OUT3
TRIGSEL_IN2		TRIGSEL_EXTOUT_2	Output0	TRIGSEL_OUT4
TRIGSEL_IN3			Output1	TRIGSEL_OUT5
TRIGSEL_IN4		TRIGSEL_EXTOUT_3	Output0	TRIGSEL_OUT6
TRIGSEL_IN5			Output1	TRIGSEL_OUT7
TRIGSEL_IN6		TRIGSEL_EXTOUT_4	Output0	TRIGSEL_OUT8
TRIGSEL_IN7			Output1	TRIGSEL_OUT9
TRIGSEL_IN8		TRIGSEL_EXTOUT_5	Output0	TRIGSEL_OUT10
TRIGSEL_IN9			Output1	TRIGSEL_OUT11
TRIGSEL_IN10		TRIGSEL_EXTOUT_6	Output0	TRIGSEL_OUT12
TRIGSEL_IN11			Output1	TRIGSEL_OUT13
TRIGSEL_IN12		TRIGSEL_EXTOUT_7	Output0	TRIGSEL_OUT14
TRIGSEL_IN13			Output1	TRIGSEL_OUT15
TRIGSEL_IN14		TRIGSEL_TIMER0ITI0	Output0	TIMER0_ITI0
TRIGSEL_IN15				
TIMER0_TRGO0		TRIGSEL_TIMER1ITI0	Output0	TIMER1_ITI0
TIMER0_CH0				
TIMER0_CH1		TRIGSEL_TIMER2ITI0	Output0	TIMER2_ITI0
TIMER0_CH2				
TIMER0_CH3		TRIGSEL_TIMER7ITI0	Output0	TIMER7_ITI0
TIMER0_MCH0				

Trigger Source	Trigger select	TRIGSEL register	TRIGSEL output	Peripherals
TIMER0_MCH1 TIMER0_MCH2 TIMER0_MCH3 TIMER7_TRGO0 TIMER7_CH0 TIMER7_CH1 TIMER7_CH2 TIMER7_CH3 TIMER7_MCH0 TIMER7_MCH1 TIMER7_MCH2 TIMER7_MCH3 TIMER19_TRGO0 TIMER19_CH0 TIMER19_CH1 TIMER19_CH2 TIMER19_CH3 TIMER19_MCH0 TIMER19_MCH1 TIMER19_MCH2 TIMER19_MCH3 TIMER20_TRGO0 TIMER20_CH0 TIMER20_CH1 TIMER20_CH2 TIMER20_CH3 TIMER20_MCH0 TIMER20_MCH1 TIMER20_MCH2 TIMER20_MCH3 TIMER60_TRGO0 TIMER60_CH0 TIMER60_CH1 TIMER60_CH2 TIMER60_CH3 TIMER60_MCH0 TIMER60_MCH1 TIMER60_MCH2 TIMER60_MCH3 TIMER61_TRGO0 TIMER61_CH0 TIMER61_CH1		TRIGSEL_TIMER19 ITI0	Output0	TIMER19_ITI0
		TRIGSEL_TIMER20 ITI0	Output0	TIMER20_ITI0
		TRIGSEL_TIMER60 ITI0	Output0	TIMER60_ITI0
		TRIGSEL_TIMER61 ITI0	Output0	TIMER61_ITI0
		TRIGSEL_TIMER62 ITI0	Output0	TIMER62_ITI0
		TRIGSEL_TIMER63 ITI0	Output0	TIMER63_ITI0
		TRIGSEL_TIMER0 BRKIN_0	Output0 Output1	TIMER0_BRKIN0 TIMER0_BRKIN1
		TRIGSEL_TIMER0 BRKIN_1	Output0 Output1	TIMER0_BRKIN2 TIMER0_BRKIN3
		TRIGSEL_TIMER7 BRKIN_0	Output0 Output1	TIMER7_BRKIN0 TIMER7_BRKIN1
		TRIGSEL_TIMER7 BRKIN_1	Output0 Output1	TIMER7_BRKIN2 TIMER7_BRKIN3
		TRIGSEL_TIMER19 BRKIN_0	Output0 Output1	TIMER19_BRKIN0 TIMER19_BRKIN1
		TRIGSEL_TIMER19 BRKIN_1	Output0 Output1	TIMER19_BRKIN2 TIMER19_BRKIN3
		TRIGSEL_TIMER20 BRKIN_0	Output0 Output1	TIMER20_BRKIN0 TIMER20_BRKIN1
		TRIGSEL_TIMER20 BRKIN_1	Output0 Output1	TIMER20_BRKIN2 TIMER20_BRKIN3
		TRIGSEL_TIMER60 BRKIN_0	Output0 Output1	TIMER60_BRKIN0 TIMER60_BRKIN1
		TRIGSEL_TIMER60 BRKIN_1	Output0 Output1	TIMER60_BRKIN2 TIMER60_BRKIN3
		TRIGSEL_TIMER61 BRKIN_0	Output0 Output1	TIMER61_BRKIN0 TIMER61_BRKIN1
		TRIGSEL_TIMER61 BRKIN_1	Output0 Output1	TIMER61_BRKIN2 TIMER61_BRKIN3
		TRIGSEL_TIMER62 BRKIN_0	Output0 Output1	TIMER62_BRKIN0 TIMER62_BRKIN1
		TRIGSEL_TIMER62 BRKIN_1	Output0 Output1	TIMER62_BRKIN2 TIMER62_BRKIN3
		TRIGSEL_TIMER63 BRKIN_0	Output0 Output1	TIMER63_BRKIN0 TIMER63_BRKIN1

Trigger Source	Trigger select	TRIGSEL register	TRIGSEL output	Peripherals
TIMER61_CH2 TIMER61_CH3		TRIGSEL_TIMER63BRKIN_1	Output0 Output1	TIMER63_BRKIN2 TIMER63_BRKIN3
TIMER61_MCH0		TRIGSEL_CM7	Output0	CM7_0_RXEV
TIMER61_MCH1 TIMER61_MCH2		TRIGSEL_DAC	Output0 Output1	DAC0_OUT0_EXTRG DAC1_OUT0_EXTRG
TIMER61_MCH3 TIMER62_TRGO0		TRIGSEL_ADC0ROUTRG	Output0	ADC0_ROUTRG
TIMER62_CH0 TIMER62_CH1 TIMER62_CH2		TRIGSEL_ADC0INSTRG_0	Output0 Output1 Output2	ADC0_INS0TRG ADC0_INS1TRG ADC0_INS2TRG
TIMER62_CH3 TIMER62_MCH0 TIMER62_MCH1		TRIGSEL_ADC0INSTRG_1	Output0 Output1 Output2	ADC0_INS3TRG ADC0_INS4TRG ADC0_INS5TRG
TIMER62_MCH2 TIMER62_MCH3 TIMER63_TRGO0		TRIGSEL_ADC1ROUTRG	Output0	ADC1_ROUTRG
TIMER63_CH0 TIMER63_CH1 TIMER63_CH2		TRIGSEL_ADC1INSTRG_0	Output0 Output1 Output2	ADC1_INS0TRG ADC1_INS1TRG ADC1_INS2TRG
TIMER63_CH3 TIMER63_MCH0 TIMER63_MCH1		TRIGSEL_ADC1INSTRG_1	Output0 Output1 Output2	ADC1_INS3TRG ADC1_INS4TRG ADC1_INS5TRG
TIMER63_MCH2 TIMER63_MCH3 TIMER1_TRGO0		TRIGSEL_SPI_0	Output0 Output1	SPI0_TRIG SPI1_TRIG
TIMER1_CH0 TIMER1_CH1		TRIGSEL_SPI_1	Output0 Output1	SPI2_TRIG SPI3_TRIG
TIMER1_CH2 TIMER1_CH3 TIMER2_TRGO0		TRIGSEL_SPI_2	Output0 Output1	SPI4_TRIG SPI5_TRIG
TIMER2_CH0 TIMER2_CH1 TIMER2_CH2		TRIGSEL_TIMERETI	Output0 Output1 Output2	TIMER_ETI0 TIMER_ETI1 TIMER_ETI2
TIMER2_CH3 TIMER5_TRGO0		TRIGSEL_LINFlexD_0	Output0 Output1	LINFlexD0_TRIG LINFlexD1_TRIG
TIMER5_TRGO1 TIMER5_TRGO2		TRIGSEL_LINFlexD_1	Output0 Output1	LINFlexD2_TRIG LINFlexD3_TRIG
TIMER5_TRGO3 TIMER6_TRGO0 TIMER6_TRGO1 TIMER6_TRGO2 TIMER6_TRGO3		TRIGSEL_LINFlexD_2	Output0 Output1	LINFlexD4_TRIG LINFlexD5_TRIG

Trigger Source	Trigger select	TRIGSEL register	TRIGSEL output	Peripherals
CMP0_OUT				
CMP1_OUT				
BKP_Tamper				
RTC_Alarm				
CM7_0_TXEV				
CK_OUT0				
TIMER_ETI0				
TIMER_ETI1				
TIMER_ETI2				
CK_OUT1				
DAC0_eoc				
DAC1_eoc				
ADC0_EORC				
ADC1_EORC				
ADC0_ECH				
ADC1_ECH				
ADC0_WD0_OUT				
ADC0_WD1_OUT				
ADC0_WD2_OUT				
ADC0_WD3_OUT				
ADC1_WD0_OUT				
ADC1_WD1_OUT				
ADC1_WD2_OUT				
ADC1_WD3_OUT				
ADC0_EOI0C				
ADC0_EOI1C				
ADC0_EOI2C				
ADC0_EOI3C				
ADC0_EOI4C				
ADC0_EOI5C				
ADC1_EOI0C				
ADC1_EOI1C				
ADC1_EOI2C				
ADC1_EOI3C				
ADC1_EOI4C				
ADC1_EOI5C				

Note: Refer to the file "GD32A711_A712 TRIGSEL connectivity.xlsx" for configuration rules.

7.5. Register definition

TRIGSEL base address: 0x4001 2000

Table 7-3. TRIGSEL register overview

Register name	Symbol	Address offset	Reset value
Trigger selection for EXTOUT register 0	TRIGSEL_EXTOUT_0	0x00	0x0000 0000
Trigger selection for EXTOUT register 1	TRIGSEL_EXTOUT_1	0x04	0x0000 0000
Trigger selection for EXTOUT register 2	TRIGSEL_EXTOUT_2	0x08	0x0000 0000
Trigger selection for EXTOUT register 3	TRIGSEL_EXTOUT_3	0x0C	0x0000 0000
Trigger selection for EXTOUT register 4	TRIGSEL_EXTOUT_4	0x10	0x0000 0000
Trigger selection for EXTOUT register 5	TRIGSEL_EXTOUT_5	0x14	0x0000 0000
Trigger selection for EXTOUT register 6	TRIGSEL_EXTOUT_6	0x18	0x0000 0000
Trigger selection for EXTOUT register 7	TRIGSEL_EXTOUT_7	0x1C	0x0000 0000
Trigger selection for TIMER0_ITI0 register	TRIGSEL_TIMER0ITI0	0x5C	0x0000 0000
Trigger selection for TIMER1_ITI0 register	TRIGSEL_TIMER1ITI0	0x60	0x0000 0000
Trigger selection for TIMER2_ITI0 register	TRIGSEL_TIMER2ITI0	0x64	0x0000 0000
Trigger selection for TIMER7_ITI0 register	TRIGSEL_TIMER7ITI0	0x68	0x0000 0000
Trigger selection for TIMER19_ITI0 register	TRIGSEL_TIMER19ITI0	0x6C	0x0000 0000
Trigger selection for TIMER20_ITI0register	TRIGSEL_TIMER20ITI0	0x70	0x0000 0000
Trigger selection for TIMER60_ITI0 register	TRIGSEL_TIMER60ITI0	0x74	0x0000 0000
Trigger selection for TIMER61_ITI0 register	TRIGSEL_TIMER61ITI0	0x78	0x0000 0000
Trigger selection for TIMER62_ITI0 register	TRIGSEL_TIMER62ITI0	0x7C	0x0000 0000
Trigger selection for TIMER63_ITI0 register	TRIGSEL_TIMER63ITI0	0x80	0x0000 0000
Trigger selection for TIMER0_BRKIN	TRIGSEL_TIMER0BRKIN	0x84	0x0000 0000

Register name	Symbol	Address offset	Reset value
register 0	_0		
Trigger selection for TIMER0_BRKIN register 1	TRIGSEL_TIMER0BRKIN_1	0x88	0x0000 0000
Trigger selection for TIMER7_BRKIN register 0	TRIGSEL_TIMER7BRKIN_0	0x8C	0x0000 0000
Trigger selection for TIMER7_BRKIN register 1	TRIGSEL_TIMER7BRKIN_1	0x90	0x0000 0000
Trigger selection for TIMER19_BRKIN register 0	TRIGSEL_TIMER19BRKIN_0	0x94	0x0000 0000
Trigger selection for TIMER19_BRKIN register 1	TRIGSEL_TIMER19BRKIN_1	0x98	0x0000 0000
Trigger selection for TIMER20_BRKIN register 0	TRIGSEL_TIMER20BRKIN_0	0x9C	0x0000 0000
Trigger selection for TIMER20_BRKIN register 1	TRIGSEL_TIMER20BRKIN_1	0xA0	0x0000 0000
Trigger selection for TIMER60_BRKIN register 0	TRIGSEL_TIMER60BRKIN_0	0xA4	0x0000 0000
Trigger selection for TIMER60_BRKIN register 1	TRIGSEL_TIMER60BRKIN_1	0xA8	0x0000 0000
Trigger selection for TIMER61_BRKIN register 0	TRIGSEL_TIMER61BRKIN_0	0xAC	0x0000 0000
Trigger selection for TIMER61_BRKIN register 1	TRIGSEL_TIMER61BRKIN_1	0xB0	0x0000 0000
Trigger selection for TIMER62_BRKIN register 0	TRIGSEL_TIMER62BRKIN_0	0xB4	0x0000 0000
Trigger selection for TIMER62_BRKIN register 1	TRIGSEL_TIMER62BRKIN_1	0xB8	0x0000 0000
Trigger selection for TIMER63_BRKIN register 0	TRIGSEL_TIMER63BRKIN_0	0xBC	0x0000 0000
Trigger selection for TIMER63_BRKIN register 1	TRIGSEL_TIMER63BRKIN_1	0xC0	0x0000 0000
Trigger selection for CM7 register	TRIGSEL_CM7	0xCC	0x0000 0000
Trigger selection for DAC register	TRIGSEL_DAC	0xD0	0x0000 0000
Trigger selection for ADC0_ROUTRG register	TRIGSEL_ADC0ROUTRG	0xD4	0x0000 0000
Trigger selection for ADC0_INSTRG register 0	TRIGSEL_ADC0INSTRG_0	0xD8	0x0000 0000
Trigger selection for ADC0_INSTRG register 1	TRIGSEL_ADC0INSTRG_1	0xDC	0x0000 0000
Trigger selection for ADC1_ROUTRG register	TRIGSEL_ADC1ROUTRG	0xE0	0x0000 0000
Trigger selection for ADC1_INSTRG	TRIGSEL_ADC1INSTRG_	0xE4	0x0000 0000

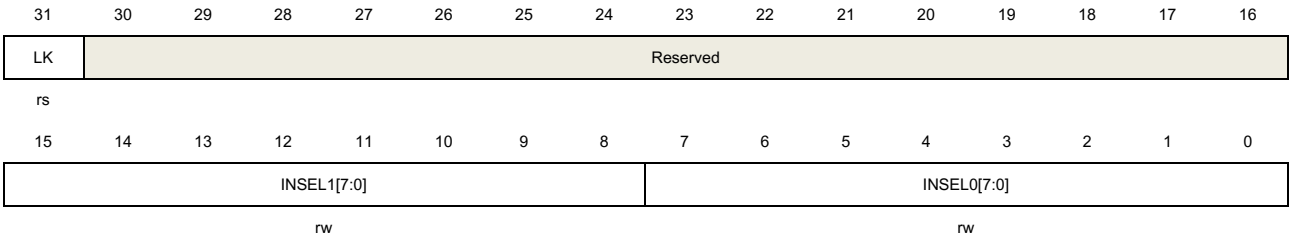
Register name	Symbol	Address offset	Reset value
register 0	0		
Trigger selection for ADC1_INSTRG register 1	TRIGSEL_ADC1INSTRG_1	0xE8	0x0000 0000
Trigger selection for SPI register 0	TRIGSEL_SPI_0	0xF8	0x0000 0000
Trigger selection for SPI register 1	TRIGSEL_SPI_1	0xFC	0x0000 0000
Trigger selection for SPI register 2	TRIGSEL_SPI_2	0x100	0x0000 0000
Trigger selection for TIMER_ETI register	TRIGSEL_TIMERETI	0x108	0x0000 0000
Trigger selection for LINFlexD register 0	TRIGSEL_LINFlexD_0	0x10C	0x0000 0000
Trigger selection for LINFlexD register 1	TRIGSEL_LINFlexD_1	0x110	0x0000 0000
Trigger selection for LINFlexD register 2	TRIGSEL_LINFlexD_2	0x114	0x0000 0000

7.5.1. Trigger selection for EXTOUT register 0 (TRIGSEL_EXTOUT_0)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_EXTOUT_0 register. 0: TRIGSEL_EXTOUT_0 register write is enabled. 1: TRIGSEL_EXTOUT_0 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output can be as the source of TRIGSEL_OUT1 (external output1 signal).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output

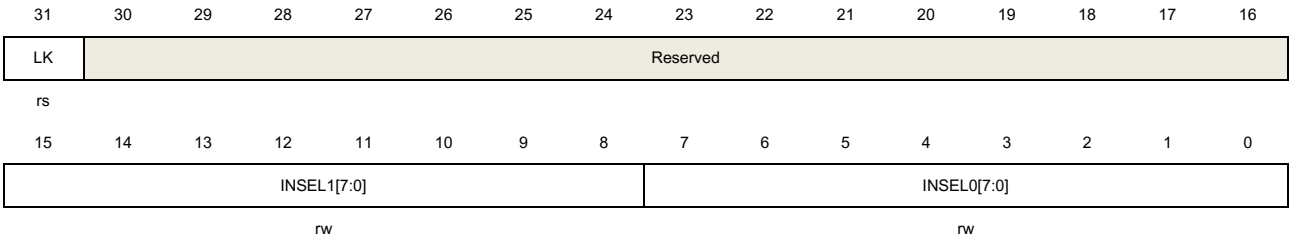
is used as the source of TRIGSEL_OUT0 (external output0 signal).

7.5.2. Trigger selection for EXTOUT register 1 (TRIGSEL_EXTOUT_1)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



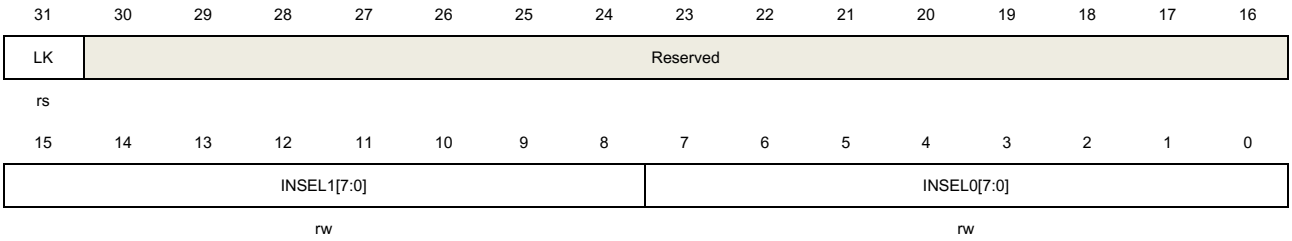
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_EXTOUT_1 register. 0: TRIGSEL_EXTOUT_1 register write is enabled. 1: TRIGSEL_EXTOUT_1 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output can be as the source of TRIGSEL_OUT3 (external output3 signal).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TRIGSEL_OUT2 (external output2 signal).

7.5.3. Trigger selection for EXTOUT register 2 (TRIGSEL_EXTOUT_2)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



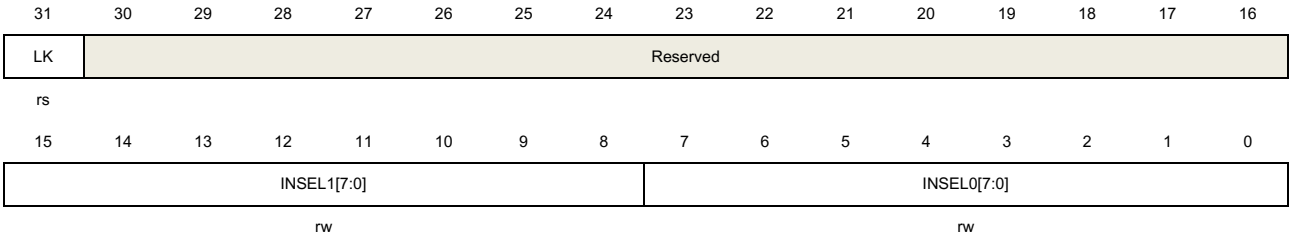
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_EXTOUT_2 register. 0: TRIGSEL_EXTOUT_2 register write is enabled. 1: TRIGSEL_EXTOUT_2 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TRIGSEL_OUT5 (external output5 signal).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TRIGSEL_OUT4 (external output4 signal).

7.5.4. Trigger selection for EXTOUT register 3 (TRIGSEL_EXTOUT_3)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



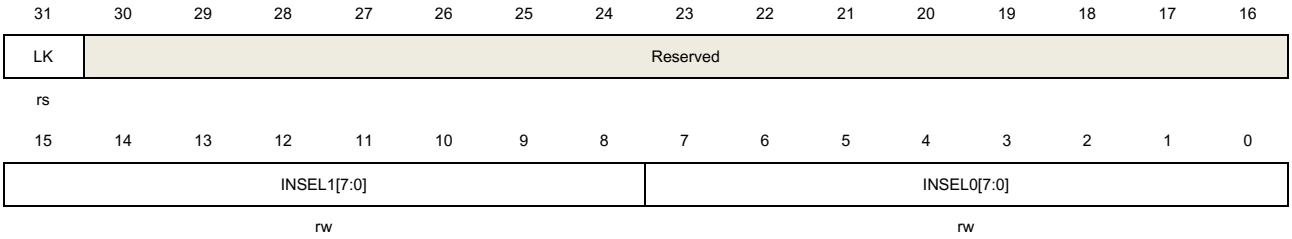
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_EXTOUT_3 register. 0: TRIGSEL_EXTOUT_3 register write is enabled. 1: TRIGSEL_EXTOUT_3 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TRIGSEL_OUT7 (external output7 signal).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TRIGSEL_OUT6 (external output6 signal).

7.5.5. Trigger selection for EXTOUT register 4 (TRIGSEL_EXTOUT_4)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



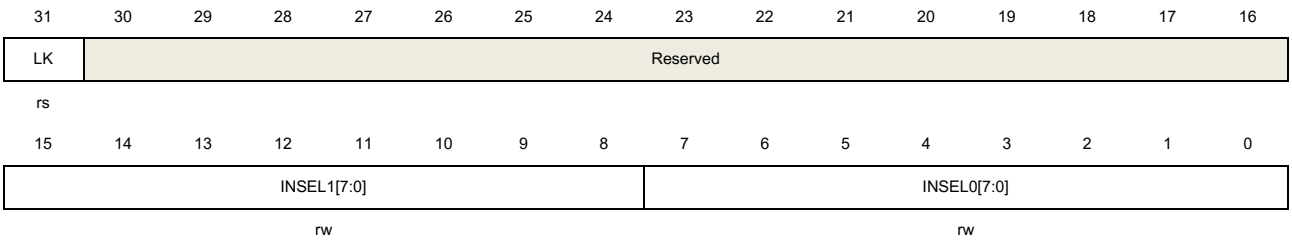
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_EXTOUT_4 register. 0: TRIGSEL_EXTOUT_4 register write is enabled. 1: TRIGSEL_EXTOUT_4 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output can be as the source of TRIGSEL_OUT9 (external output9 signal).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TRIGSEL_OUT8 (external output8 signal).

7.5.6. Trigger selection for EXTOUT register 5 (TRIGSEL_EXTOUT_5)

Address offset: 0x14

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it

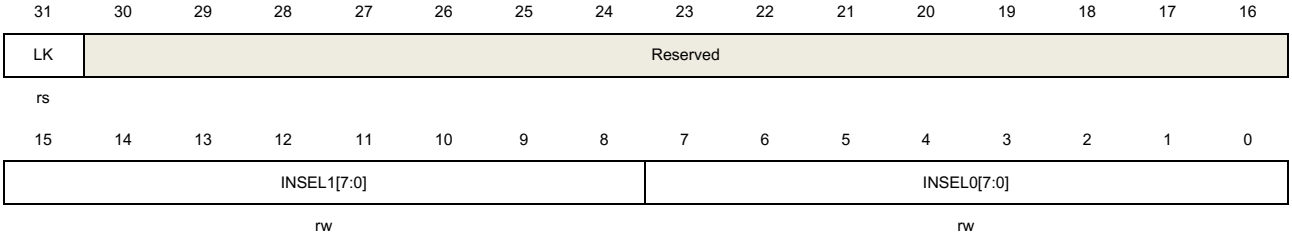
		disables write access to TRIGSEL_EXTOUT_5 register. 0: TRIGSEL_EXTOUT_5 register write is enabled. 1: TRIGSEL_EXTOUT_5 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output can be as the source of TRIGSEL_OUT11 (external output11 signal).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TRIGSEL_OUT10 (external output10 signal).

7.5.7. Trigger selection for EXTOUT register 6 (TRIGSEL_EXTOUT_6)

Address offset: 0x18

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



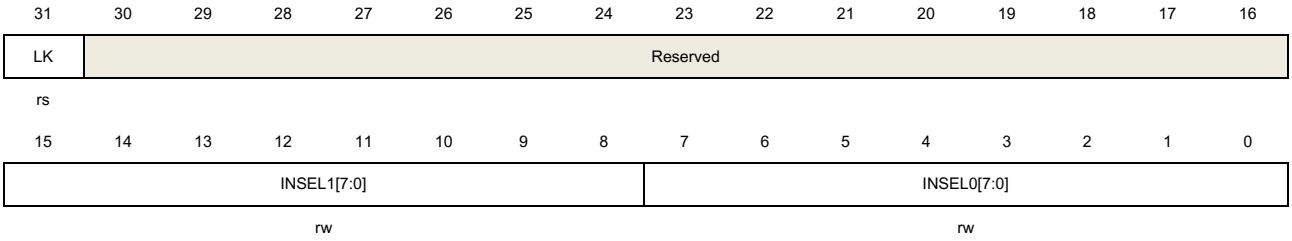
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_EXTOUT_6 register. 0: TRIGSEL_EXTOUT_6 register write is enabled. 1: TRIGSEL_EXTOUT_6 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TRIGSEL_OUT13 (external output13 signal).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TRIGSEL_OUT12 (external output12 signal).

7.5.8. Trigger selection for EXTOUT register 7 (TRIGSEL_EXTOUT_7)

Address offset: 0x1C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



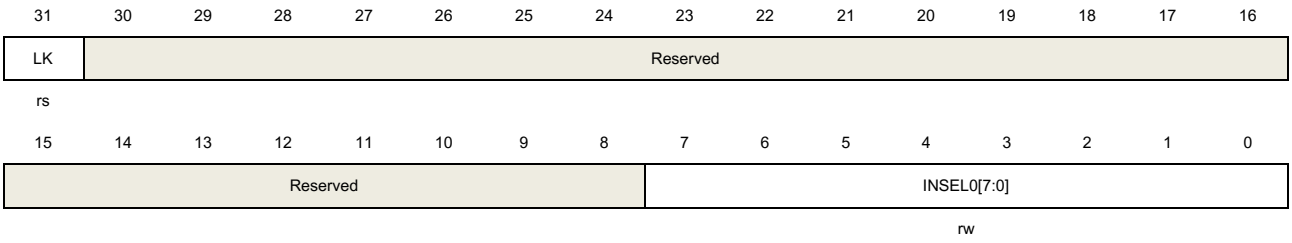
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_EXTOUT_7 register. 0: TRIGSEL_EXTOUT_7 register write is enabled. 1: TRIGSEL_EXTOUT_7 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TRIGSEL_OUT15 (external output15 signal).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TRIGSEL_OUT14 (external output14 signal).

7.5.9. Trigger selection for TIMER0_ITI0 register (TRIGSEL_TIMER0ITI0)

Address offset: 0x5C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER0ITI0 register. 0: TRIGSEL_TIMER0ITI0 register write is enabled.

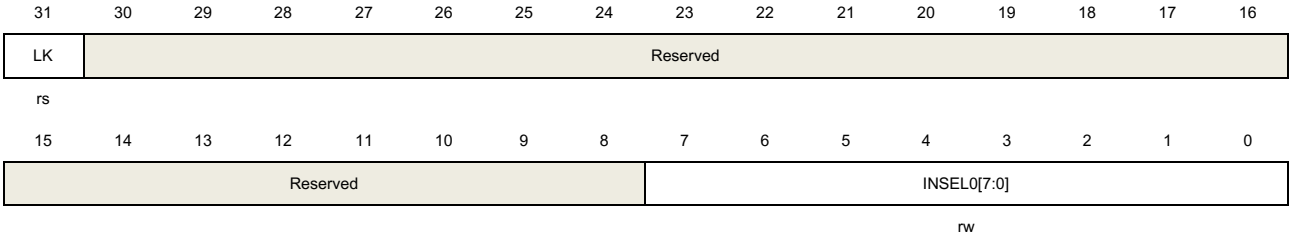
		1: TRIGSEL_TIMER0ITI0 register write is disabled.
30:8	Reserved	Must be kept at reset value.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER0_ITI0 trigger input.

7.5.10. Trigger selection for TIMER1_ITI0 register (TRIGSEL_TIMER1ITI0)

Address offset: 0x60

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



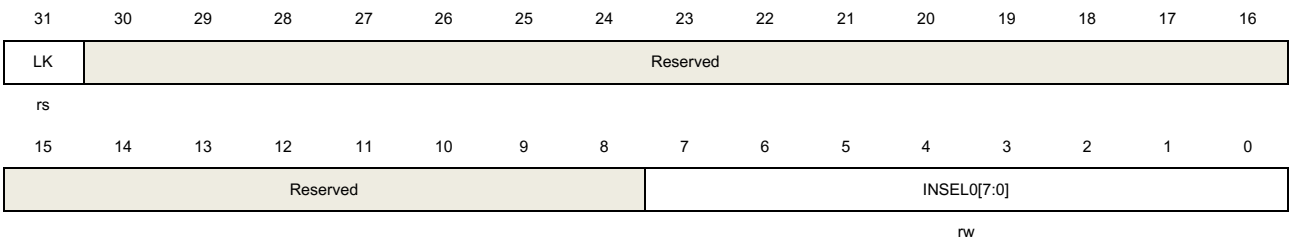
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER1ITI0 register. 0: TRIGSEL_TIMER1ITI0 register write is enabled. 1: TRIGSEL_TIMER1ITI0 register write is disabled.
30:8	Reserved	Must be kept at reset value.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER1_ITI0 trigger input.

7.5.11. Trigger selection for TIMER2_ITI0 register (TRIGSEL_TIMER2ITI0)

Address offset: 0x64

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

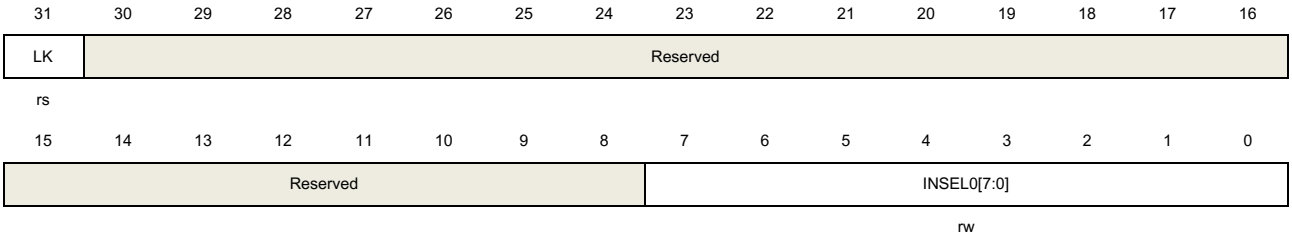


Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER2ITI0 register. 0: TRIGSEL_TIMER2ITI0 register write is enabled. 1: TRIGSEL_TIMER2ITI0 register write is disabled.
30:8	Reserved	Must be kept at reset value.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER2_ITI0 trigger input.

7.5.12. Trigger selection for TIMER7_ITI0 register (TRIGSEL_TIMER7ITI0)

Address offset: 0x68
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

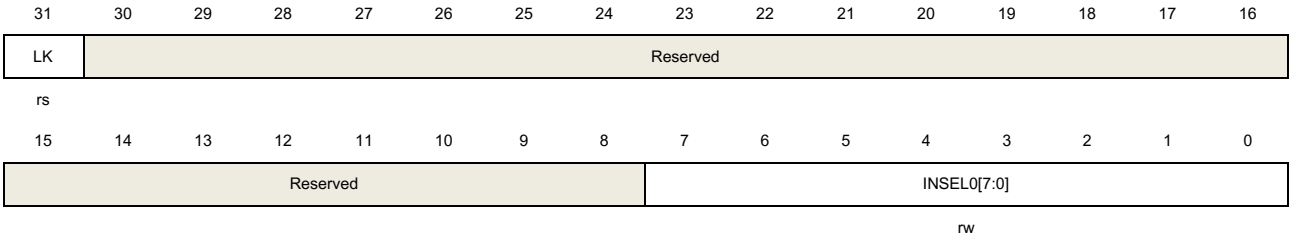


Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER7ITI0 register. 0: TRIGSEL_TIMER7ITI0 register write is enabled. 1: TRIGSEL_TIMER7ITI0 register write is disabled.
30:8	Reserved	Must be kept at reset value.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER7_ITI0 trigger input.

7.5.13. Trigger selection for TIMER19_ITI0 register (TRIGSEL_TIMER19ITI0)

Address offset: 0x6C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



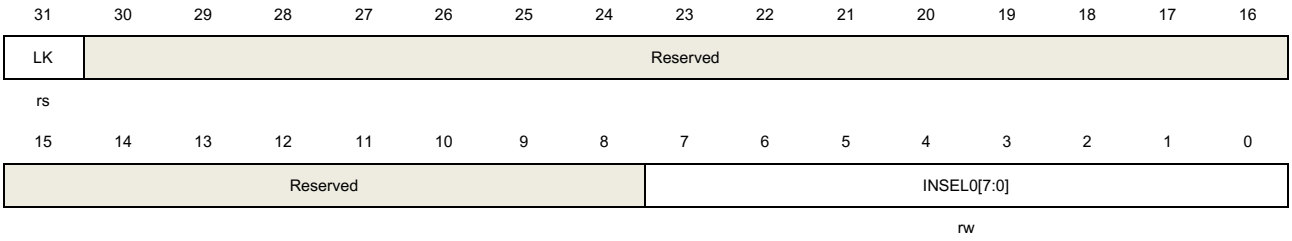
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER19ITI0 register. 0: TRIGSEL_TIMER19ITI0 register write is enabled. 1: TRIGSEL_TIMER19ITI0register write is disabled.
30:8	Reserved	Must be kept at reset value.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER19_ITI0trigger input.

7.5.14. Trigger selection for TIMER20_ITI0register (TRIGSEL_TIMER20ITI0)

Address offset: 0x70

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

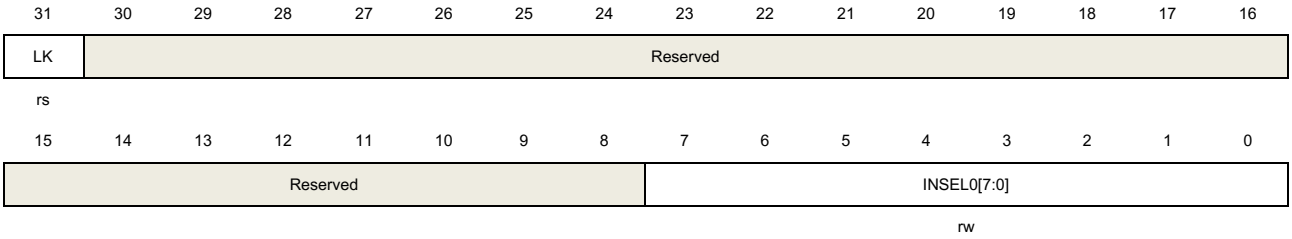


Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER20ITI0 register. 0: TRIGSEL_TIMER20ITI0 register write is enabled. 1: TRIGSEL_TIMER20ITI0 register write is disabled.
30:8	Reserved	Must be kept at reset value.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER20_ITI0 trigger input.

7.5.15. Trigger selection for TIMER60_ITI0 register (TRIGSEL_TIMER60ITI0)

Address offset: 0x74
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

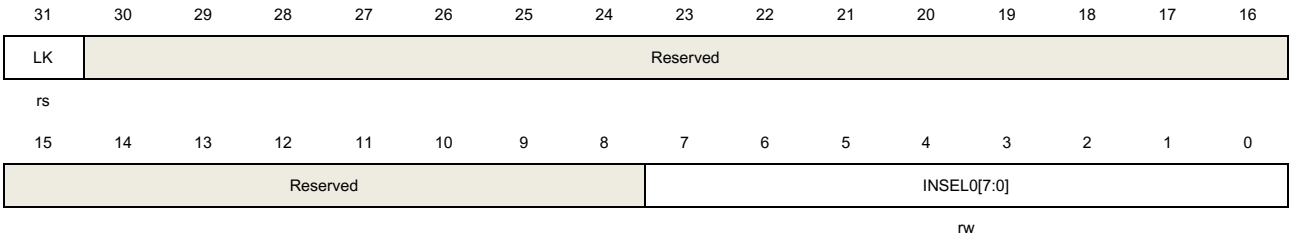


Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER60ITI0 register. 0: TRIGSEL_TIMER60ITI0 register write is enabled. 1: TRIGSEL_TIMER60ITI0 register write is disabled.
30:8	Reserved	Must be kept at reset value.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER60_ITI0 trigger input.

7.5.16. Trigger selection for TIMER61_ITI0 register (TRIGSEL_TIMER61ITI0)

Address offset: 0x78
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



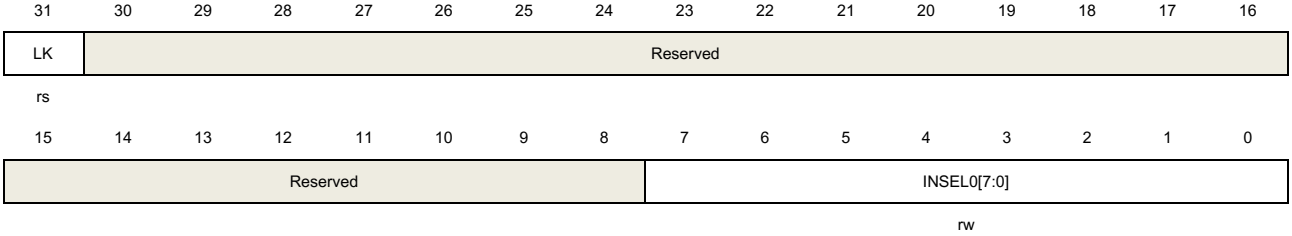
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER61ITI0 register. 0: TRIGSEL_TIMER61ITI0 register write is enabled. 1: TRIGSEL_TIMER61ITI0 register write is disabled.

30:8	Reserved	Must be kept at reset value.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER61_ITI0 trigger input.

7.5.17. Trigger selection for TIMER62_ITI0 register (TRIGSEL_TIMER62ITI0)

Address offset: 0x7C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

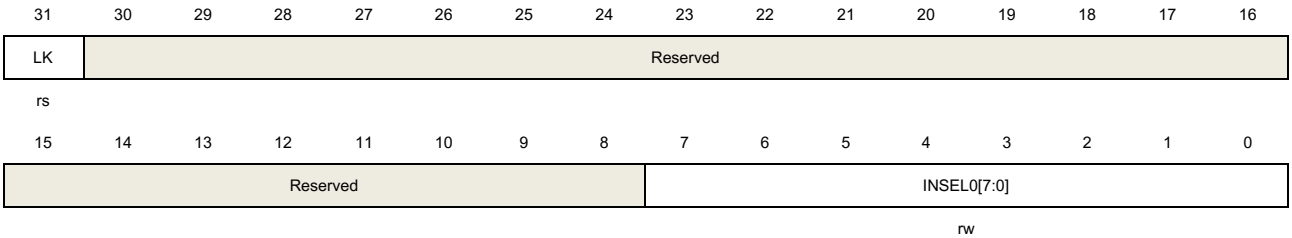


Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER62ITI0 register. 0: TRIGSEL_TIMER62ITI0 register write is enabled. 1: TRIGSEL_TIMER62ITI0 register write is disabled.
30:8	Reserved	Must be kept at reset value.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER62_ITI0 trigger input.

7.5.18. Trigger selection for TIMER63_ITI0 register (TRIGSEL_TIMER63ITI0)

Address offset: 0x80
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



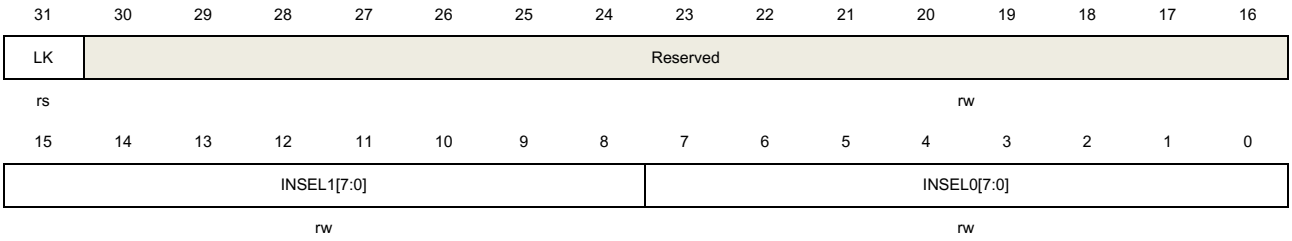
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER63ITI0 register. 0: TRIGSEL_TIMER63ITI0 register write is enabled. 1: TRIGSEL_TIMER63ITI0 register write is disabled.
30:8	Reserved	Must be kept at reset value.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER63_ITI0 trigger input.

7.5.19. Trigger selection for TIMER0_BRKIN register 0 (TRIGSEL_TIMER0BRKIN_0)

Address offset: 0x84

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER0BRKIN_0 register. 0: TRIGSEL_TIMER0BRKIN_0 register write is enabled. 1: TRIGSEL_TIMER0BRKIN_0 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER0_BRKIN1 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER0_BRKIN0 trigger input.

7.5.20. Trigger selection for TIMER0_BRKIN register 1 (TRIGSEL_TIMER0BRKIN_1)

Address offset: 0x88
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LK	Reserved														
rs	rw														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INSEL1[7:0]								INSEL0[7:0]							
rw								rw							

Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER0BRKIN_1 register. 0: TRIGSEL_TIMER0BRKIN_1 register write is enabled. 1: TRIGSEL_TIMER0BRKIN_1 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER0_BRKIN3 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER0_BRKIN2 trigger input.

7.5.21. Trigger selection for TIMER7_BRKIN register 0 (TRIGSEL_TIMER7BRKIN_0)

Address offset: 0x8C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LK	Reserved														
rs	rw														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INSEL1[7:0]								INSEL0[7:0]							
rw								rw							

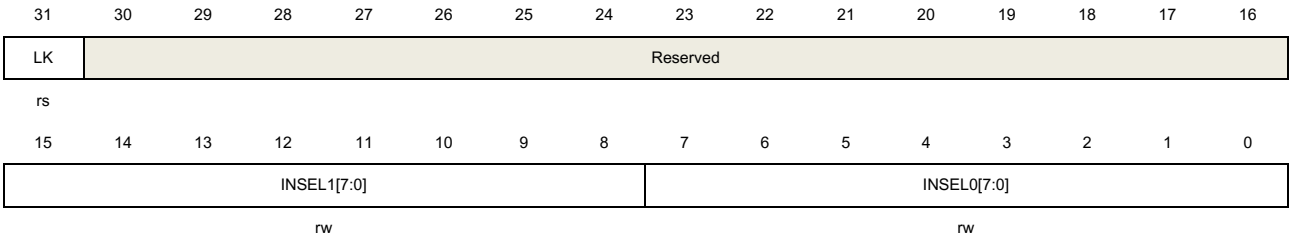
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER7BRKIN_0 register. 0: TRIGSEL_TIMER7BRKIN_0 register write is enabled. 1: TRIGSEL_TIMER7BRKIN_0 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER7_BRKIN1 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER7_BRKIN0 trigger input.

7.5.22. Trigger selection for TIMER7_BRKIN register 1 (TRIGSEL_TIMER7BRKIN_1)

Address offset: 0x90

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER7BRKIN_1 register. 0: TRIGSEL_TIMER7BRKIN_1 register write is enabled. 1: TRIGSEL_TIMER7BRKIN_1 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER7_BRKIN3 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0

These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER7_BRKIN2 trigger input.

7.5.23. Trigger selection for TIMER19_BRKIN register 0 (TRIGSEL_TIMER19BRKIN_0)

Address offset: 0x94
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LK	Reserved														
rs	rw														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INSEL1[7:0]								INSEL0[7:0]							
rw								rw							

Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER19BRKIN_0 register. 0: TRIGSEL_TIMER19BRKIN_0 register write is enabled. 1: TRIGSEL_TIMER19BRKIN_0 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER19_BRKIN1 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER19_BRKIN0 trigger input.

7.5.24. Trigger selection for TIMER19_BRKIN register 1 (TRIGSEL_TIMER19BRKIN_1)

Address offset: 0x98
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LK	Reserved														
rs	rw														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INSEL1[7:0]								INSEL0[7:0]							
rw								rw							

Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER19BRKIN_1 register. 0: TRIGSEL_TIMER19BRKIN_1 register write is enabled. 1: TRIGSEL_TIMER19BRKIN_1 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER19_BRKIN3 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER19_BRKIN2 trigger input.

7.5.25. Trigger selection for TIMER20_BRKIN register 0 (TRIGSEL_TIMER20BRKIN_0)

Address offset: 0x9C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LK	Reserved														
rs	rw														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INSEL1[7:0]								INSEL0[7:0]							
rw								rw							

Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER20BRKIN_0 register. 0: TRIGSEL_TIMER20BRKIN_0 register write is enabled. 1: TRIGSEL_TIMER20BRKIN_0 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output

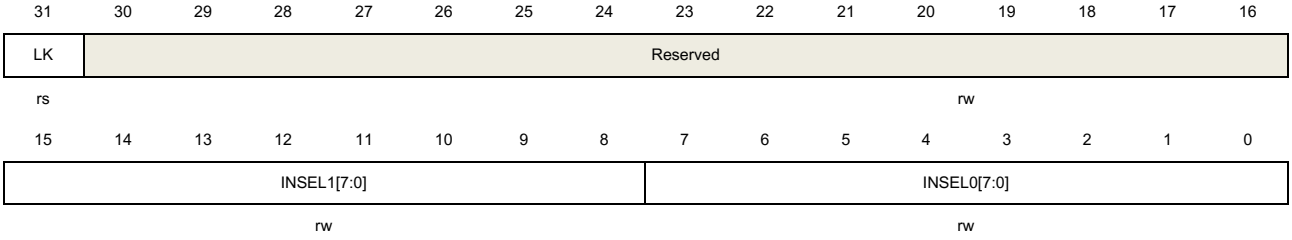
		is used as the source of TIMER20_BRKIN1 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER20_BRKIN0 trigger input.

7.5.26. Trigger selection for TIMER20_BRKIN register 1 (TRIGSEL_TIMER20BRKIN_1)

Address offset: 0xA0

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



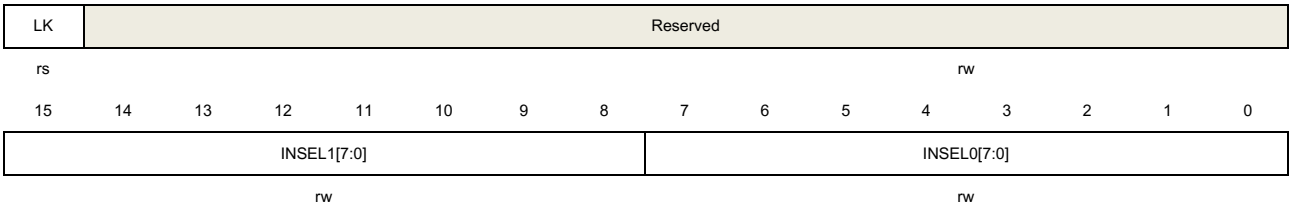
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER20BRKIN_1 register. 0: TRIGSEL_TIMER20BRKIN_1 register write is enabled. 1: TRIGSEL_TIMER20BRKIN_1 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER20_BRKIN3 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER20_BRKIN2 trigger input.

7.5.27. Trigger selection for TIMER60_BRKIN register 0 (TRIGSEL_TIMER60BRKIN_0)

Address offset: 0xA4

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



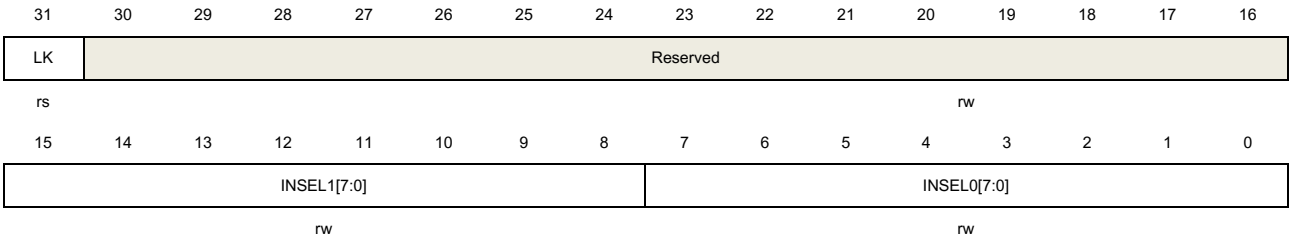
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER60BRKIN_0 register. 0: TRIGSEL_TIMER60BRKIN_0 register write is enabled. 1: TRIGSEL_TIMER60BRKIN_0 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER60_BRKIN1 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER60_BRKIN0 trigger input.

7.5.28. Trigger selection for TIMER60_BRKIN register 1 (TRIGSEL_TIMER60BRKIN_1)

Address offset: 0xA8

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



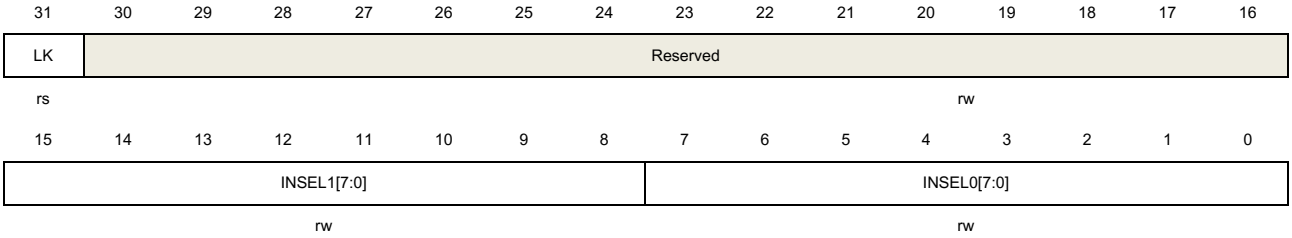
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER60BRKIN_1 register. 0: TRIGSEL_TIMER60BRKIN_1 register write is enabled. 1: TRIGSEL_TIMER60BRKIN_1 register write is disabled.
30:16	Reserved	Must be kept at reset value.

15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER60_BRKIN3 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER60_BRKIN2 trigger input.

7.5.29. Trigger selection for TIMER61_BRKIN register 0 (TRIGSEL_TIMER61BRKIN_0)

Address offset: 0xAC
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER61BRKIN_0 register. 0: TRIGSEL_TIMER61BRKIN_0 register write is enabled. 1: TRIGSEL_TIMER61BRKIN_0 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER61_BRKIN1 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER61_BRKIN0 trigger input.

7.5.30. Trigger selection for TIMER61_BRKIN register 1 (TRIGSEL_TIMER61BRKIN_1)

Address offset: 0xB0
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LK	Reserved														
rs	rw														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INSEL1[7:0]								INSEL0[7:0]							
rw								rw							

Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER61BRKIN_1 register. 0: TRIGSEL_TIMER61BRKIN_1 register write is enabled. 1: TRIGSEL_TIMER61BRKIN_1 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER61_BRKIN3 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER61_BRKIN2 trigger input.

7.5.31. Trigger selection for TIMER62_BRKIN register 0 (TRIGSEL_TIMER62BRKIN_0)

Address offset: 0xB4

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LK	Reserved														
rs	rw														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INSEL1[7:0]								INSEL0[7:0]							
rw								rw							

Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER62BRKIN_0 register. 0: TRIGSEL_TIMER62BRKIN_0 register write is enabled.

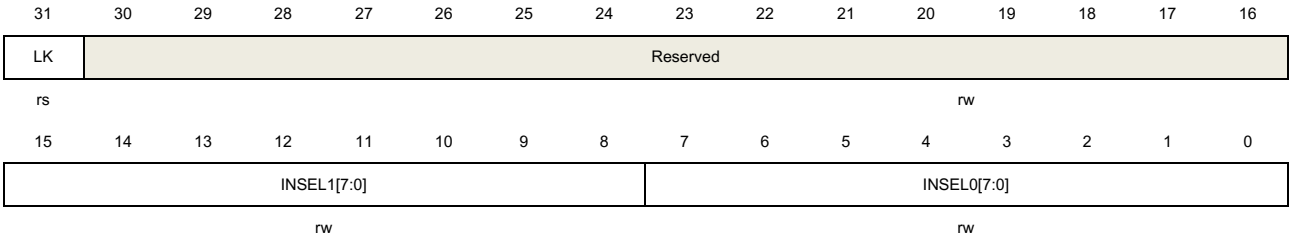
		1: TRIGSEL_TIMER62BRKIN_0 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER62_BRKIN1 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER62_BRKIN0 trigger input.

7.5.32. Trigger selection for TIMER62_BRKIN register 1 (TRIGSEL_TIMER62BRKIN_1)

Address offset: 0xB8

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER62BRKIN_1 register. 0: TRIGSEL_TIMER62BRKIN_1 register write is enabled. 1: TRIGSEL_TIMER62BRKIN_1 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER62_BRKIN3 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER62_BRKIN2 trigger input.

7.5.33. Trigger selection for TIMER63_BRKIN register 0 (TRIGSEL_TIMER63BRKIN_0)

Address offset: 0xBC

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LK	Reserved														
rs	rw														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INSEL1[7:0]								INSEL0[7:0]							
rw								rw							

Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER63BRKIN_0 register. 0: TRIGSEL_TIMER63BRKIN_0 register write is enabled. 1: TRIGSEL_TIMER63BRKIN_0 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER63_BRKIN1 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER63_BRKIN0 trigger input.

7.5.34. Trigger selection for TIMER63_BRKIN register 1 (TRIGSEL_TIMER63BRKIN_1)

Address offset: 0xC0

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LK	Reserved														
rs	rw														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INSEL1[7:0]								INSEL0[7:0]							
rw								rw							

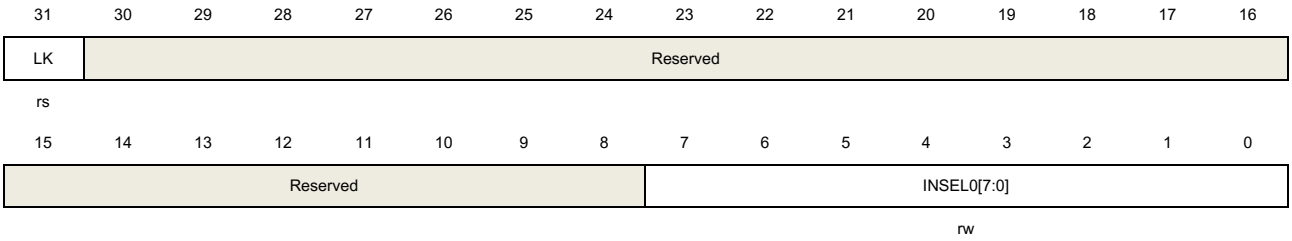
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMER63BRKIN_1 register. 0: TRIGSEL_TIMER63BRKIN_1 register write is enabled. 1: TRIGSEL_TIMER63BRKIN_1 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER63_BRKIN3 trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER63_BRKIN2 trigger input.

7.5.35. Trigger selection for CM7 register (TRIGSEL_CM7)

Address offset: 0xCC

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

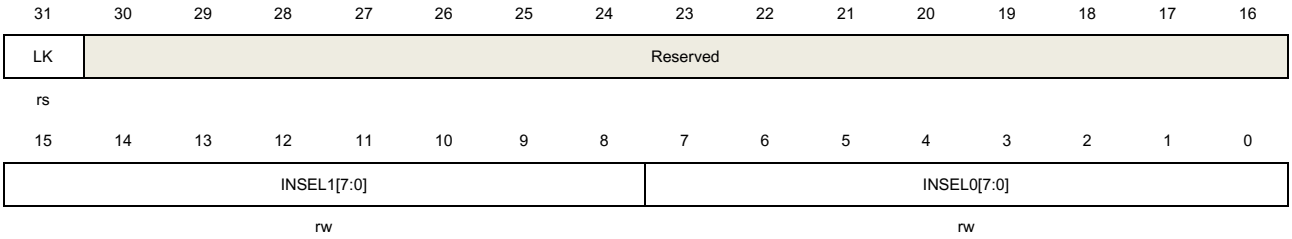


Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_CM7 register. 0: TRIGSEL_CM7 register write is enabled. 1: TRIGSEL_CM7 register write is disabled.
30:8	Reserved	Must be kept at reset value.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of CM7_0_RXEV.

7.5.36. Trigger selection for DAC register (TRIGSEL_DAC)

Address offset: 0xD0
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

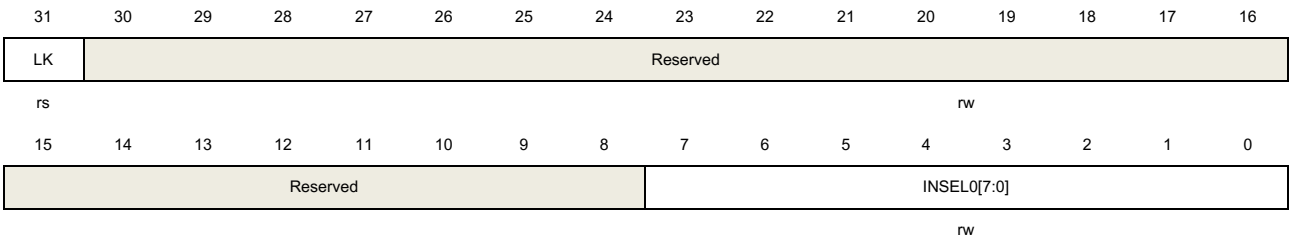


Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_DAC register. 0: TRIGSEL_DAC register write is enabled. 1: TRIGSEL_DAC register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of DAC1_OUT0_EXTRIG (DAC1_OUT0 external trigger input).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of DAC0_OUT0_EXTRIG (DAC0_OUT0 external trigger input).

7.5.37. Trigger selection for ADC0_ROUTRG register (TRIGSEL_ADC0ROUTRG)

Address offset: 0xD4
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



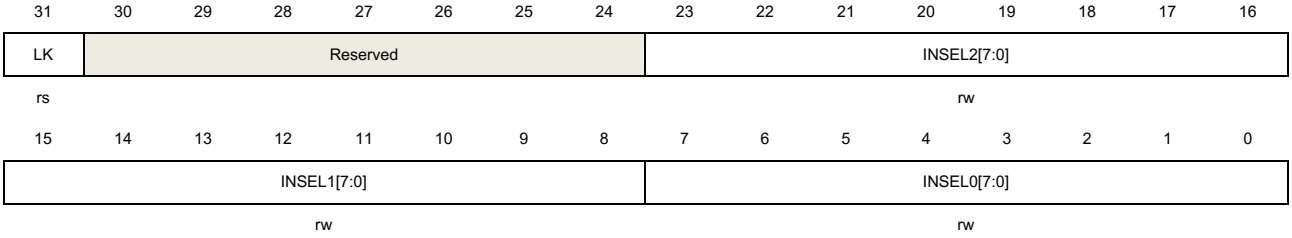
Bits	Fields	Descriptions

31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_ADC0ROUTRG register. 0: TRIGSEL_ADC0ROUTRG register write is enabled. 1: TRIGSEL_ADC0ROUTRG register write is disabled.
30:8	Reserved	Must be kept at reset value.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of ADC0_ROUTRG (ADC0 routine sequence external trigger input).

7.5.38. Trigger selection for ADC0_INSTRG register 0 (TRIGSEL_ADC0INSTRG_0)

Address offset: 0xD8
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



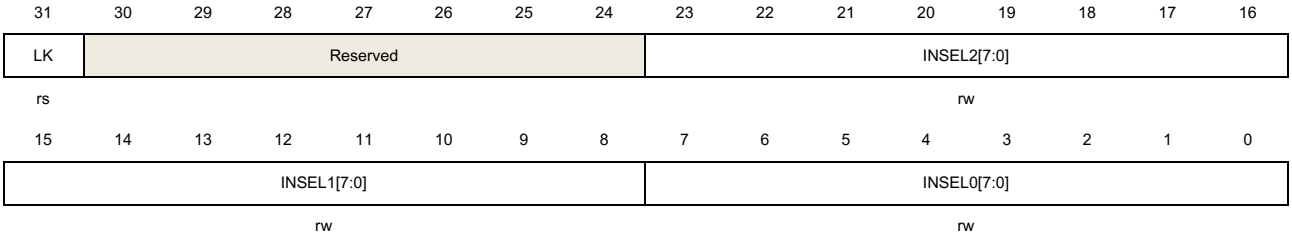
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_ADC0INSTRG_0 register. 0: TRIGSEL_ADC0INSTRG_0 register write is enabled. 1: TRIGSEL_ADC0INSTRG_0 register write is disabled.
30:24	Reserved	Must be kept at reset value.
23:16	INSEL2[7:0]	Trigger input source selection for output2 These bits are used to select trigger input signal connected to output2. The output is used as the source of ADC0_INS2TRG (ADC0 inserted sequence 2 external trigger input).
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of ADC0_INS1TRG (ADC0 inserted sequence 1 external trigger input).

7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of ADC0_INS0TRG (ADC0 inserted sequence 0 external trigger input).

7.5.39. Trigger selection for ADC0_INSTRG register 1 (TRIGSEL_ADC0INSTRG_1)

Address offset: 0xDC
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

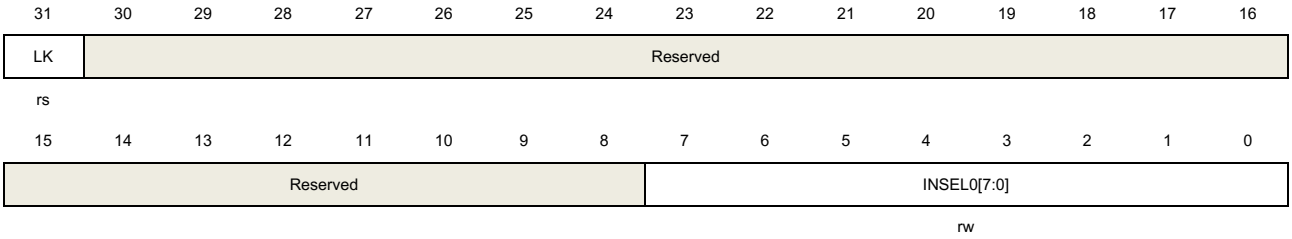


Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_ADC0INSTRG_1 register. 0: TRIGSEL_ADC0INSTRG_1 register write is enabled. 1: TRIGSEL_ADC0INSTRG_1 register write is disabled.
30:24	Reserved	Must be kept at reset value.
23:16	INSEL2[7:0]	Trigger input source selection for output2 These bits are used to select trigger input signal connected to output2. The output is used as the source of ADC0_INS5TRG (ADC0 inserted sequence 5 external trigger input).
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of ADC0_INS4TRG (ADC0 inserted sequence 4 external trigger input).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of ADC0_INS3TRG (ADC0 inserted sequence 3 external trigger input).

7.5.40. Trigger selection for ADC1_ROUTRG register (TRIGSEL_ADC1ROUTRG)

Address offset: 0xE0
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

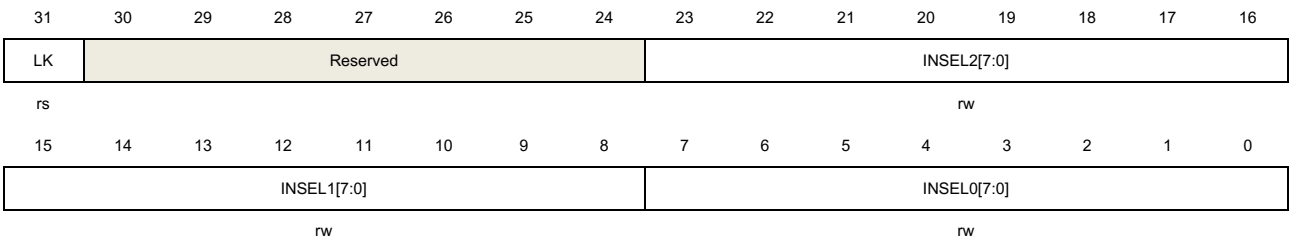


Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_ADC1ROUTRG register. 0: TRIGSEL_ADC1ROUTRG register write is enabled. 1: TRIGSEL_ADC1ROUTRG register write is disabled.
30:8	Reserved	Must be kept at reset value.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of ADC1_ROUTRG (ADC1 routine sequence external trigger input).

7.5.41. Trigger selection for ADC1_INSTRG register 0 (TRIGSEL_ADC1INSTRG_0)

Address offset: 0xE4
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it

		disables write access to TRIGSEL_ADC1INSTRG_0 register. 0: TRIGSEL_ADC1INSTRG_0 register write is enabled. 1: TRIGSEL_ADC1INSTRG_0 register write is disabled.
30:24	Reserved	Must be kept at reset value.
23:16	INSEL2[7:0]	Trigger input source selection for output2 These bits are used to select trigger input signal connected to output2. The output is used as the source of ADC1_INS2TRG (ADC1 inserted sequence 2 external trigger input).
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of ADC1_INS1TRG (ADC1 inserted sequence 1 external trigger input).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of ADC1_INS0TRG (ADC1 inserted sequence 0 external trigger input).

7.5.42. Trigger selection for ADC1_INSTRG register 1 (TRIGSEL_ADC1INSTRG_1)

Address offset: 0xE8

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LK	Reserved							INSEL2[7:0]							
rs								rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INSEL1[7:0]								INSEL0[7:0]							
rw								rw							

Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_ADC1INSTRG_1 register. 0: TRIGSEL_ADC1INSTRG_1 register write is enabled. 1: TRIGSEL_ADC1INSTRG_1 register write is disabled.
30:24	Reserved	Must be kept at reset value.
23:16	INSEL2[7:0]	Trigger input source selection for output2 These bits are used to select trigger input signal connected to output2. The output

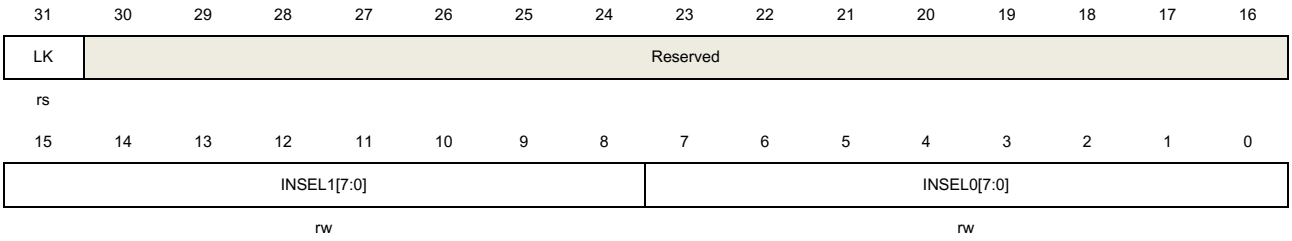
		is used as the source of ADC1_INS5TRG (ADC1 inserted sequence 5 external trigger input).
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of ADC1_INS4TRG (ADC1 inserted sequence 4 external trigger input).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of ADC1_INS3TRG (ADC1 inserted sequence 3 external trigger input).

7.5.43. Trigger selection for SPI register 0 (TRIGSEL_SPI_0)

Address offset: 0xF8

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



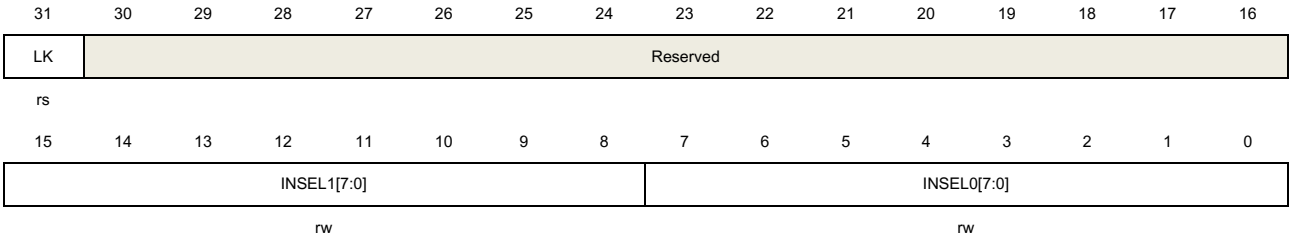
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_SPI_0 register. 0: TRIGSEL_SPI_0 register write is enabled. 1: TRIGSEL_SPI_0 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of SPI1_TRIG (SPI1 trigger input).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of SPI0_TRIG (SPI0 trigger input).

7.5.44. Trigger selection for SPI register 1 (TRIGSEL_SPI_1)

Address offset: 0xFC

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



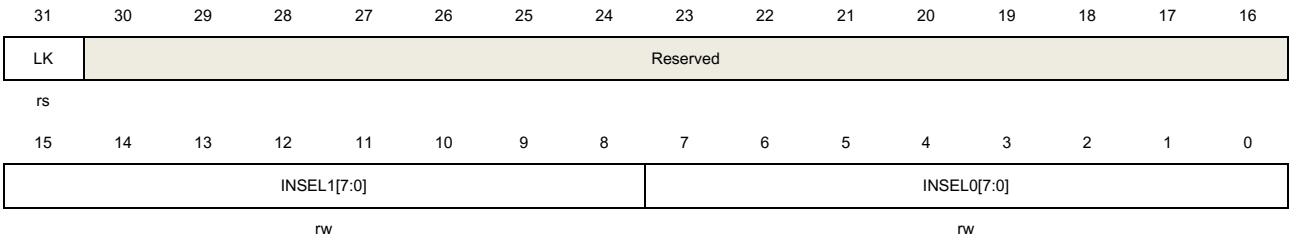
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_SPI_1 register. 0: TRIGSEL_SPI_1 register write is enabled. 1: TRIGSEL_SPI_1 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of SPI3_TRIG (SPI3 trigger input).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of SPI2_TRIG (SPI2 trigger input).

7.5.45. Trigger selection for SPI register 2 (TRIGSEL_SPI_2)

Address offset: 0x100

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



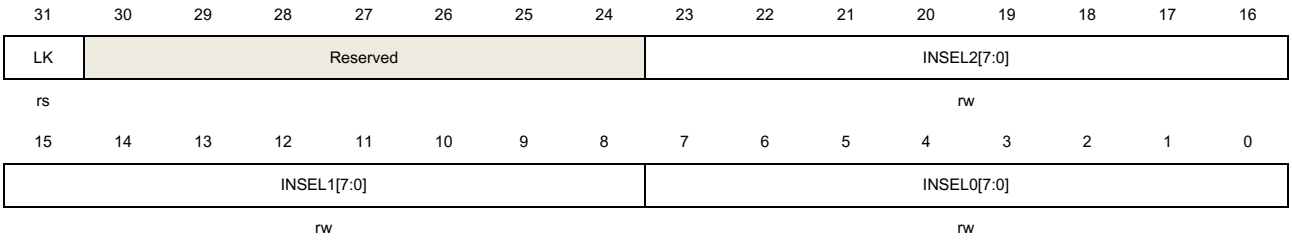
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_SPI_2 register. 0: TRIGSEL_SPI_2 register write is enabled.

		1: TRIGSEL_SPI_2 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of SPI5_TRIG (SPI5 trigger input).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of SPI4_TRIG (SPI4 trigger input).

7.5.46. Trigger selection for TIMER_ETI register (TRIGSEL_TIMERETI)

Address offset: 0x108
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

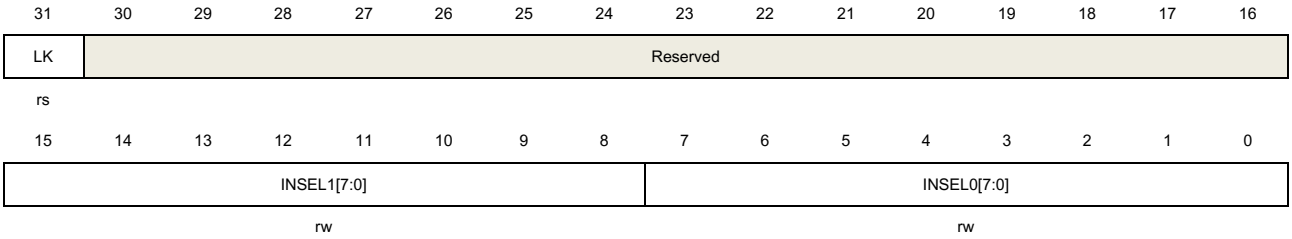


Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_TIMERETI register. 0: TRIGSEL_TIMERETI register write is enabled. 1: TRIGSEL_TIMERETI register write is disabled.
30:24	Reserved	Must be kept at reset value.
23:16	INSEL2[7:0]	Trigger input source selection for output2 These bits are used to select trigger input signal connected to output2. The output is used as the source of TIMER_ETI2(TIMER external trigger input 2).
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of TIMER_ETI1(TIMER external trigger input.
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of TIMER_ETI0(TIMER external trigger input.

7.5.47. Trigger selection for LINFlexD register 0 (TRIGSEL_LINFlexD_0)

Address offset: 0x10C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

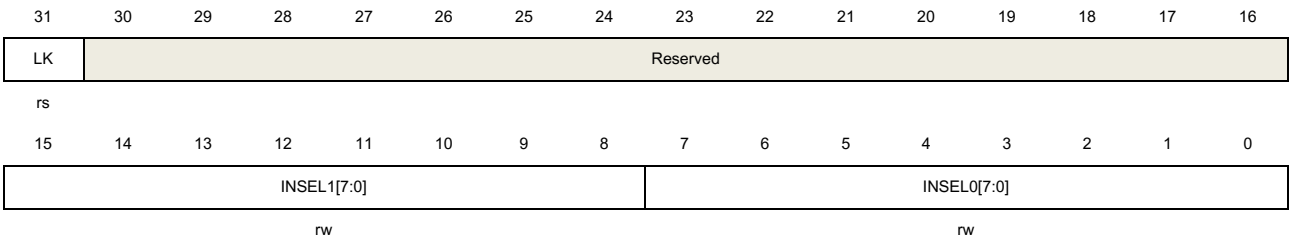


Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_LINFlexD_0 register. 0: TRIGSEL_LINFlexD_0 register write is enabled. 1: TRIGSEL_LINFlexD_0 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of LINFlexD1_TRIG (LINFlexD1 trigger input).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of LINFlexD0_TRIG (LINFlexD0 trigger input).

7.5.48. Trigger selection for LINFlexD register 1 (TRIGSEL_LINFlexD_1)

Address offset: 0x110
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



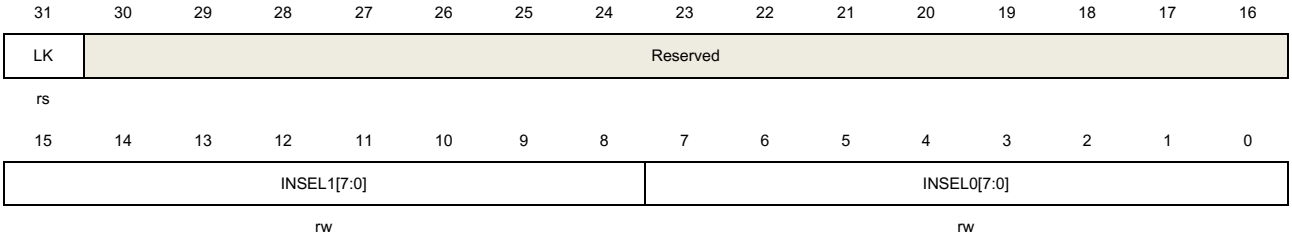
Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it

		disables write access to TRIGSEL_LINFlexD_1 register. 0: TRIGSEL_LINFlexD_1 register write is enabled. 1: TRIGSEL_LINFlexD_1 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of LINFlexD3_TRIG (LINFlexD3 trigger input).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of LINFlexD2_TRIG (LINFlexD2 trigger input).

7.5.49. Trigger selection for LINFlexD register 2 (TRIGSEL_LINFlexD_2)

Address offset: 0x114
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31	LK	TRIGSEL register lock. This bit is set by software and cleared only by a system reset. When it is set, it disables write access to TRIGSEL_LINFlexD_2 register. 0: TRIGSEL_LINFlexD_2 register write is enabled. 1: TRIGSEL_LINFlexD_2 register write is disabled.
30:16	Reserved	Must be kept at reset value.
15:8	INSEL1[7:0]	Trigger input source selection for output1 These bits are used to select trigger input signal connected to output1. The output is used as the source of LINFlexD5_TRIG (LINFlexD5 trigger input).
7:0	INSEL0[7:0]	Trigger input source selection for output0 These bits are used to select trigger input signal connected to output0. The output is used as the source of LINFlexD4_TRIG (LINFlexD4 trigger input).

8. General-purpose and alternate-function I/Os (GPIO and AFIO)

8.1. Overview

There are up to 144 general purpose I/O pins (GPIO), named PA0 ~ PA15, PB0 ~ PB7, PB10 ~ PB15, PC0 ~ PC10, PC13, PD0, PD8 ~ PD15, PE2 ~ PE15, PF0 ~ PF15, PG0 ~ PG8, PH2 ~ PH14, PI1, PI8 ~ PI15, PJ0 ~ PJ11, PK0 ~ PK2, PL0 ~ PL5, PM0 ~ PM5, PN0 ~ PN4. for the device to implement logic input / output functions. Each GPIO port has related control and configuration registers to satisfy the requirements of specific applications. The external interrupts on the GPIO pins of the device have related control and configuration registers in the Interrupt / Event Controller Unit (EXTI).

The GPIO ports are pin-shared with other alternative functions (AFs) to obtain maximum flexibility on the package pins. The GPIO pins can be used as alternative functional pins by configuring the corresponding registers regardless of the AF input or output pins.

Each of the GPIO pins can be configured by software as output (push-pull or open-drain), input, peripheral alternate function or analog mode. Each GPIO pin can be configured as pull-up, pull-down or no pull-up/pull-down. All GPIOs are high-current capable except for analog mode.

In addition, PC14 and PC15 only support digital input and analog functions.

8.2. Characteristics

- Input / output direction control.
- Schmitt trigger input function enable control.
- Each pin weak pull-up / pull-down function.
- Output push-pull / open drain enable control.
- Output set / reset control.
- External interrupt with programmable trigger edge-using EXTI configuration registers.
- Analog input / output configuration.
- Alternate function input / output configuration.
- Port configuration lock.
- Single cycle toggle output capability.
- Support external interrupt signal filtering.
- Supports pad retain function.
- Support analog filter function.

8.3. Function overview

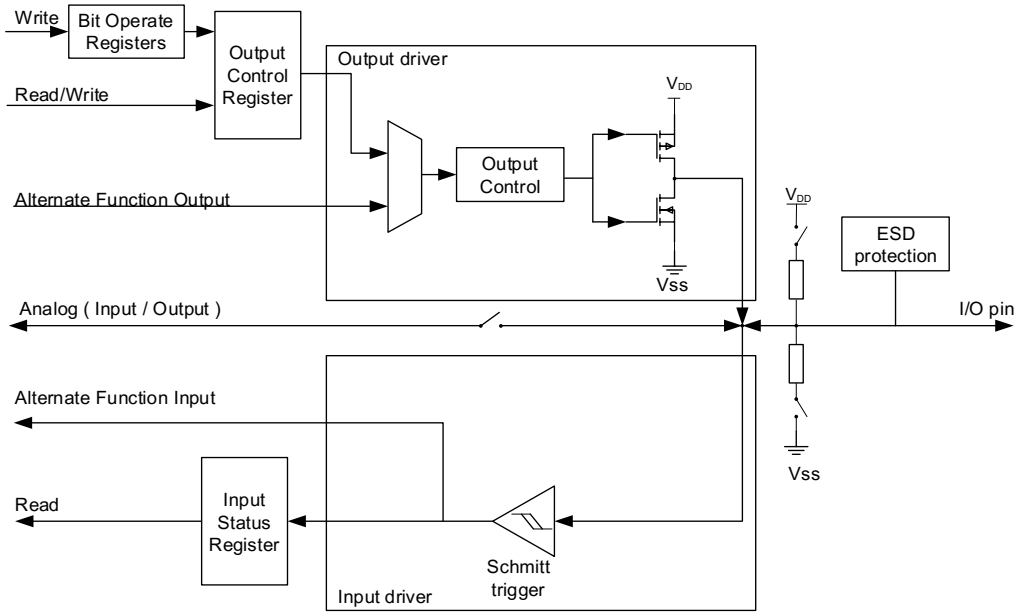
Each of the general-purpose I/O ports can be configured as GPIO inputs, GPIO outputs, AF function or analog mode by GPIO 32-bit configuration registers (GPIOx_CTL). When select AF function, the pad input or output is decided by selected AF function output enable. When the port is output (GPIO output or AFIO output), it can be configured as push-pull or open drain mode by GPIO output mode registers (GPIOx_OMODE). And the port max speed can be configured by GPIO output speed registers (GPIOx_OSPD). Each port can be configured as floating (no pull-up and pull-down), pull-up or pull-down function by GPIO pull-up / pull-down registers (GPIOx_PUD).

Table 8-1. GPIO configuration table

PAD TYPE			CTLy	OMy	PUDy
GPIO INPUT	X	Floating	00	X	00
		pull-up			01
		pull-down			10
GPIO OUTPUT	push-pull	Floating	01	0	00
		pull-up			01
		pull-down			10
	open-drain	Floating		1	00
		pull-up			01
		pull-down			10
AFIO INPUT	X	Floating	10	X	00
		pull-up			01
		pull-down			10
AFIO OUTPUT	push-pull	Floating	10	0	00
		pull-up			01
		pull-down			10
	open-drain	Floating		1	00
		pull-up			01
		pull-down			10
ANALOG	X	X	11	X	XX

Figure 8-1. Basic structure of a standard I/O port bit shows the basic structure of an I/O port bit.

Figure 8-1. Basic structure of a standard I/O port bit



8.3.1. GPIO pin configuration

During or just after the reset period, the alternative functions are all inactive and the most GPIO ports are configured in the analog mode. But the JTAG / Serial-Wired Debug pins are in input PU / PD mode after reset:

PA15: JTDI in PU mode

PA14: JTCK / SWCLK in PD mode

PA13: JTMS / SWDIO in PU mode

PB3: JTDO in output mode

The GPIO pins can be configured as inputs or outputs. When the GPIO pins are configured as input pins, all GPIO pins have an internal weak pull-up and weak pull-down which can be chosen. And the data on the external pins can be captured at every AHB clock cycle to the port input status register (GPIOx_ISTAT).

When the GPIO pins are configured as output pins, user can configure the speed of the ports. And chooses the output driver mode: Push-Pull or Open-Drain mode. The value of the port output control register (GPIOx_OCTL) is output on the I/O pin.

There is no need to read-then-write when programming the GPIOx_OCTL at bit level, the user can modify only one or several bits in a single atomic AHB write access by programming 1 to the bit operate register (GPIOx_BOP, or for clearing only GPIOx_BC, or for toggle only GPIOx_TG). The other bits will not be affected.

8.3.2. External interrupt / event lines

All ports have external interrupt capability. To use external interrupt lines, the port must be configured as input mode.

8.3.3. Alternate functions (AF)

When the port is configured as AFIO (set CTLy bits to "0b10", which is in GPIOx_CTL registers), the port is used as peripheral alternate functions. Each port has sixteen alternate functions can be configured by GPIO alternate functions selected registers (GPIOx_AFSELy (y = 0, 1)). The detail alternate function assignments for each port are in the device datasheet. When using the peripheral multiplexing function, it is necessary to allocate and initialize the corresponding GPIO first.

8.3.4. Additional functions

Some pins have additional functions, which have priority over the configuration in the standard GPIO registers. When for ADC, DAC, CMP, VRC_CTRL additional functions, the port must be configured as analog mode. When for WKUPx and NMI additional functions, the port must be configured as input mode. When for RTC and OSC32IN/OSC32OUT functions, the port type is set automatically by hardware. These ports can be used as normal GPIO when the additional functions disabled.

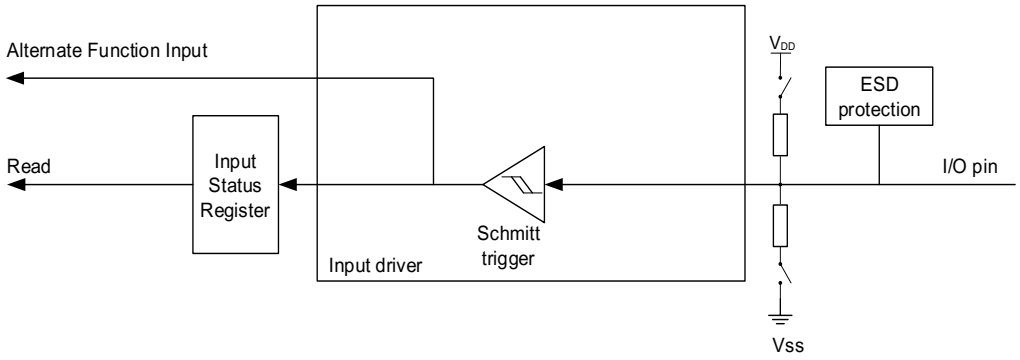
8.3.5. Input configuration

When GPIO pin is configured as input:

- The schmitt trigger input is enabled.
- The weak pull-up and pull-down resistors could be chosen.
- Every AHB clock cycle the data present on the I/O pin is got to the port input status register.
- The output buffer is disabled.

Figure 8-2. Input configuration shows the input configuration.

Figure 8-2. Input configuration



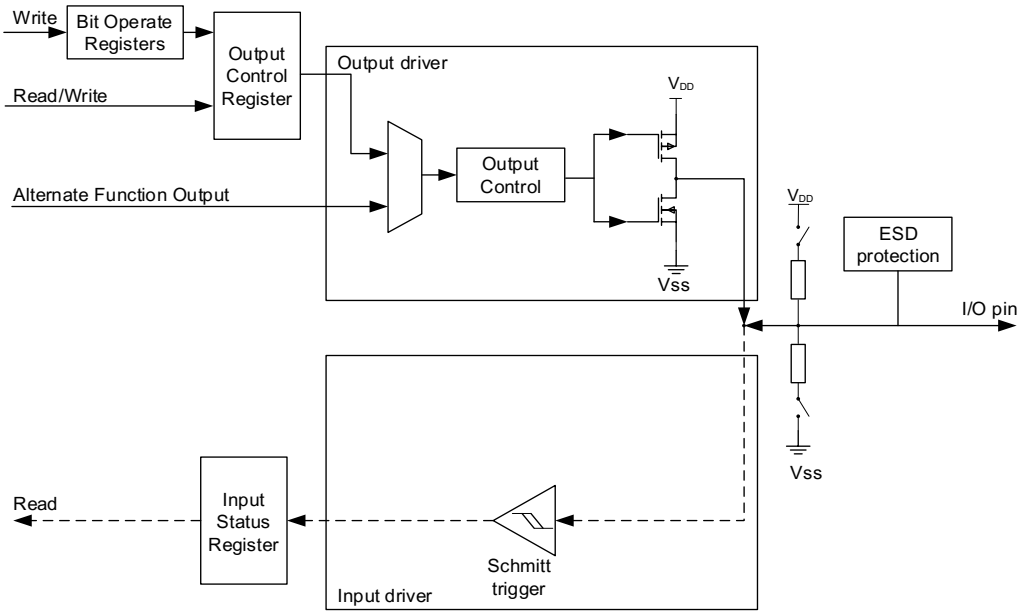
8.3.6. Output configuration

When GPIO pin is configured as output:

- The schmitt trigger input is enabled.
- The weak pull-up and pull-down resistors could be chosen.
- The output buffer is enabled.
- Open Drain Mode: The pad output low level when a "0" in the output control register; while the pad leaves Hi-Z when a "1" in the output control register.
- Push-Pull Mode: The pad output low level when a "0" in the output control register; while the pad output high level when a "1" in the output control register.
- A read access to the port output control register gets the last written value.
- A read access to the port input status register gets the I/O state.

Figure 8-3. Output configuration shows the output configuration.

Figure 8-3. Output configuration



8.3.7. Analog configuration

When GPIO pin is used as analog configuration:

- The weak pull-up and pull-down resistors are disabled.
- The output buffer is disabled.
- The schmitt trigger input is disabled.
- The port input status register of this I/O port bit is "0".

Figure 8-4. Analog configuration shows the analog configuration.

Figure 8-4. Analog configuration



8.3.8. Alternate function (AF) configuration

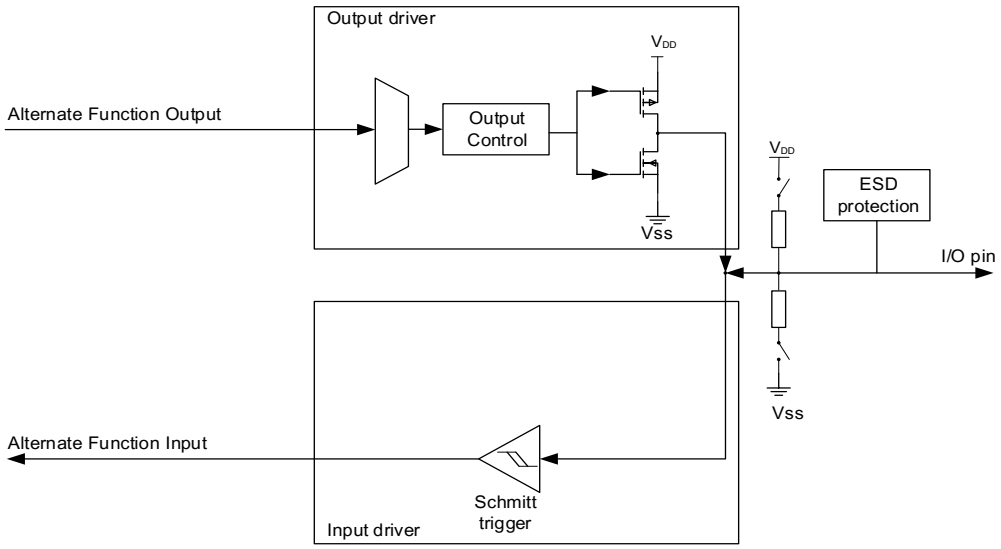
To suit for different device packages, the GPIO supports some alternate functions mapped to some other pins by software.

When be configured as alternate function:

- The output buffer is enabled in open-drain or push-pull configuration.
- The output buffer is driven by the peripheral.
- The schmitt trigger input is enabled.
- The weak pull-up and pull-down resistors could be chosen.
- The I/O pin data is stored into the port input status register every AHB clock.
- A read access to the port input status register gets the I/O state.
- A read access to the port output control register gets the last written value.

Figure 8-5. Alternate function configuration shows the alternate function configuration.

Figure 8-5. Alternate function configuration



8.3.9. GPIO locking function

The locking mechanism allows the IO configuration to be protected.

The protected registers are GPIOx_CTL, GPIOx_OMODE, GPIOx_OSPD, GPIOx_PUD GPIOx_AFSELy (y=0, 1), GPIOx_IFEN, GPIOx_PREN and GPIOx_SREN. It allows the I/O configuration to be frozen by the 32-bit locking register (GPIOx_LOCK). When the special LOCK sequence has occurred on LKK bit in GPIOx_LOCK register and the LKy bit is set in GPIOx_LOCK register, the corresponding port is locked and the corresponding port configuration cannot be modified until the next reset. It recommended to be used in the configuration of driving a power module.

8.3.10. GPIO single cycle toggle function

GPIO could toggle the I/O output level in single AHB cycle by writing 1 to the corresponding bit of GPIOx_TG register. The output signal frequency could up to the half of the AHB clock.

8.3.11. Pad input filter

The pad input filter is an analog filter. It can be enabled by configuring GPIOx_IFEN register. For details please refer to relevant registers.

8.3.12. External interrupt filter

External interrupt filter is a digital filter. It can be enabled by set EIFEx bit in GPIOx_IFCTL register. Clock of interrupt filtering can be configured by IQSP bit fields in GPIOx_IFCTL register, and the scope of filtering can be configured by GPIOx_IFSCCTL0 / GPIOx_IFSCCTL1 registers. If the function of interrupt filtering is enabled, the external

interrupt should have internal synchronous function. Therefore, the width of the interrupt signals should be at least 2.5 ~ 3 times the IRC48M clock cycles to correctly capture the interrupts.

Note: The clock of external interrupt filter must be provided by IRC48M, otherwise the interrupt will be disabled.

8.3.13. Pad retain

The I/O configuration is retained when the MCU from standby mode to run mode if pad retain is enabled. The advantage of the feature is that the MCU returns to run mode without reconfiguring. Pad retain can be configured by PRENy bit in GPIO_PREN register. Refer to the PMU chapter for details of pad retain.

Note: After the MCU exits Standby mode, if you want the GPIO to retain the IO mode it had before entering Standby mode when re-entering Standby, you need to set the corresponding bit of the GPIOx_PREN register to 1 and the PAD_RET_CFG bit in the CLTCLFG register to 0. Additionally, the GPIO retains the configuration not as a register value but as the IO mode in the GPIO Pad circuit.

The pad include run pins and standby pins, the standby pins are capable of outputting in standby mode. The standby pins refer to ***Table 8-2. Standby pins***. For run pins, regardless of whether pad retain is enabled or not, the state of the pad will be kept when entering standby mode, including the output state. When pad retain is enabled and the MCU exits standby, it can maintain the state before entering standby. If pad retain is disabled, the MCU cannot maintain the state before entering standby when it exits standby mode and will return to the default state. For standby pins, if pad retain is disabled, the pins can maintain a dynamic output state in standby mode. When pad retain is enabled, entering standby mode will retain the output state, and it is not guaranteed that dynamic output can be performed in standby mode. After exiting standby, it can maintain the state before entering standby.

Table 8-2. Standby pins

MCU	Pin	Function
GD32A711 / A712	PA13 / PL5 / PL4	CMP0_OUT
	PF7 / PM5 / PF3	CMP0_RROUT
	PM4 / PH13	CMP1_OUT
	PE6 / PA15	CMP1_RROUT
	PL0 / PM4	CK_OUT1
	PB7	NRST

8.3.14. Triple-modular redundancy (TMR)

When GPIO is configured as FFIO0 / FFIO1 / GFIO / NRST function, the corresponding GPIO registers need to be set triple-modular redundancy which can correct a fault. Each identified register is replaced by three registers and its voting logic, if one of the registers experience a failure, the functionality through the two replicated registers can prevail, negating the effects

of the fault. The GPIOx_TMRALARMSTAT register can record fail state of FFIO0 / FFIO1 / GFIO / NRST.

8.3.15. Pin States

The table in the following chapters mentioned various conditions / events device pin states, such as reset, system reset function, power on state, selftest phase etc. Reset event details can be found in the reset overview section.

For GD32A711/A712 devices

- NRST: PB7. The pad state tables below assume NRST configured as reset pin. (PB7 is dedicated reset pin when RSTPIN_MOD configured as 1, the signal is configured in the CTLCFG module)
- JTMS: PA13
- JTCK: PA14
- JTDI: PA15
- JTDO: PB3
- CK_OUT1: PM4
- FFIO[0:1]: PJ11, PJ10

Note: When the FFIOMOD bit in the CTLCFG_SR_STAT2 register is set to 1, PJ10 and PJ11 are used as FFIO pins. It is necessary to ensure that PJ10 and PJ11 are configured in alternate function mode and select the AF0 function.

Table 8-3. Pin state on power up condition of the MCU

Pin	State
NRST	Low
JTMS, JTCK, JTDI, JTDO	Pulled values: JTMS = High, JTCK = Low, JTDI = High) JTDO: High-Z
FFIO[0:1]	High-Z
CK_OUT1	High-Z
GPIOs	High-Z
OSCIN	High-Z before connecting OSCIN
OSCOUT	High-Z before connecting OSCOUT

Table 8-4. Pin state when any system reset is triggered

Pin	State
NRST	Low
JTMS, JTCK, JTDI, JTDO	Pulled values: JTMS = High, JTCK = Low, JTDI = High JTDO: High-Z
FFIO[0:1]	High-Z
CK_OUT1	Clkout expose CKOUT1_CFG is configured to be enabled else High-Z

Pin	State
GPIOs	High-Z

Table 8-5. Pin behavior when subsystem reset including subsystem entry sequence

Pin	State (subsystem reset entry)	State (Reset until SW comes up)
NRST	Low	Low
JTMS, JTCK, JTDI, JTDO	Pulled values: JTMS = High, JTCK = Low, JTDI = High if configured as JTAG, High-Z if configured as GPIO JTDO: High-Z	JTMS = High, JTCK = Low, JTDI = High JTDO: High-Z
FFIO[0:1]	ERR from FMU if FFIOMOD is configured to be enabled else High-Z	ERR from FMU if FFIOMOD is configured to be enabled else High-Z
CK_OUT1	Clkout expose if CKOUT1_CFG is configured to be enabled else High-Z	Clkout expose if CKOUT1_CFG is configured to be enabled else High-Z
GPIOs	High-Z	High-Z

Table 8-6. Pin behavior when selftest (ERPAETEN bit in RCU_ERPACFG register configured as 1)

Pin	State (subsystem reset entry)	State (Reset until SW comes up)
NRST	Low if configured as reset pin High-Z if configured as GPIO	Low if configured as reset pin High-Z if configured as GPIO
JTMS, JTCK, JTDI, JTDO as JTAG	Pulled values: JTMS = High, JTCK = Low, JTDI = High if configured as JTAG JTDO: High-Z	Pulled values: JTMS = High, JTCK = Low, JTDI = High if configured as JTAG JTDO: High-Z
FFIO[0:1]	ERR from FMU if FFIOMOD is configured to be enabled else High-Z.	ERR from FMU if FFIOMOD is configured to be enabled else High-Z.
CK_OUT1	Clkout expose if CKOUT1_CFG is configured to be enabled else High-Z.	Clkout expose if CKOUT1_CFG is configured to be enabled else High-Z.
GPIOs	High-Z	High-Z

Table 8-7. Pin behavior when selftest (ERPAETEN bit in RCU_ERPACFG register and ST_FINISH_RPAEN bit in RCU_SUBSRPACTL register are configured as 0)

Pin	State (subsystem reset entry)	State (Reset until SW comes up)
NRST	Same state as before selftest	Same state as before selftest
JTMS, JTCK, JTDI, JTDO as JTAG	Same state as before selftest	Pulled values (TMS = High, TCK = Low, TDI = High)

Pin	State (subsystem reset entry)	State (Reset until SW comes up)
		TDO: No change
FFIO[0:1]	FMU error state if FFIOMOD is enabled and FRCFG is enabled else no change	FMU error state if FFIOMOD is enabled and FRCFG is enabled else no change
CK_OUT1	Same state as before selftest	Same state as before selftest
GPIOs	Same state as before selftest	Same state as before selftest

Table 8-8. Pin behavior when FMU fault in run mode when the fault reaction is not reset

Pin	State(Standby until SW comes up)
NRST	If NRST is designated as special PAD, PAD status will not change. If NRST is configured to GPIO, then it will act according to SMC configuration. (1) If RSTPIN_MOD configured to 1, pad according to the reset state machine. (2) If RSTPIN_MOD configured to 0, SMC is configured to 1, pad according to the GPIOx_AFSEL0/1 configuration. (3) if RSTPIN_MOD is configured to 0, SMC is configured to 0, the pin state is High-Z
JTMS, JTCK, JTDI, JTDO as JTAG	If the PAD is configured to the JTAG, then there is no change, PAD will continue to perform operations; If the PAD is configured to GPIO, then it will act according to SMC configuration. (1) If the pad is configured to the JTAG mode, and the corresponding SMC bit of GPIOx_SMCCTL register is configured to 1, pad work according to the JTAG state machine or agreement. (2) If the pad is not configured to JTAG mode, and the corresponding SMC bit of GPIOx_SMCCTL register is configured to 0, the pins states are High-Z
FFIO[0:1]	If the PAD is configured as a dedicated PAD by programming FFIOMOD and the Eout indication is enabled, it will indicate an error status. If it is a GPIO PAD, it will act according to the SMC configuration. (1) If FFIOMOD is configured to 1, then the FFP pin indicates an error state. (2) If FFIOMOD is configured to 0, and the corresponding SMC bit of GPIOx_SMCCTL register is configured to 1, the pin state will not change. (3) If FFIOMOD is configured to 0, and the corresponding SMC bit of GPIOx_SMCCTL register is configured to 0, the pin state is High-Z.
CK_OUT1	If the PAD is configured to the special PAD, then there is no change, PAD will continue to perform operations; If the PAD is configured to GPIO, then it will act according to SMC configuration. (1) If the pad is configured to CK_OUT1 function, and the

Pin	State(Standby until SW comes up)
	corresponding SMC bit of the GPIO GPIOx_SMCCTL registers of is configured to 1, the pad work according to the CK_OUT1 function. (2) If the corresponding SMC bit of GPIOx_SMCCTL register is configured to 0, the pin state is High-Z.
GPIOs	If the PAD is configured to the special PAD, then there is no change, PAD will continue to perform operations; If the PAD is configured to GPIO, then it will act according to SMC configuration. (1) If the corresponding SMC bit of the GPIO GPIOx_SMCCTL registers of is configured to 1, the pin state will not change. (2) If the corresponding SMC bit of the GPIO GPIOx_SMCCTL registers of is configured to 0, the pin state is High-Z.

Table 8-9. Pin behavior when standby (PAD_RET_CFG bit field in CTLCFG_SUBSR_CTL0 register must be configured as 1)

Pin function	State(Standby until SW comes up)
NRST	According to the configuration before entering standby.
JTMS, JTCK, JTDI, JTDO as JTAG	Depends on software configuration before entering standby
FFIO[0:1]	Depends on software configuration before entering standby
CK_OUT1	Depends on software configuration before entering standby
GPIOs	Depends on software configuration before entering standby

8.4. Register definition

GPIOA base address: 0x4800 0000

GPIOB base address: 0x4800 0400

GPIOC base address: 0x4800 0800

GPIOD base address: 0x4800 0C00

GPIOE base address: 0x4800 1000

GPIOF base address: 0x4800 1400

GPIOG base address: 0x4800 1800

GPIOH base address: 0x4800 1C00

GPIOI base address: 0x4800 2000

GPIOJ base address: 0x4800 2400

GPIOK base address: 0x4800 2800

GPIOL base address: 0x4800 2C00

GPIOM base address: 0x4800 3000

GPION base address: 0x4800 3400

Table 8-10. GPIO register overview

Register name	Symbol	Address offset	Reset value
Port control register	GPIOx_CTL	0x00	GPIOA: 0xABFF FFFF; GPIOB: 0xFFFF FFBF; Others: 0xFFFF FFFF
Port output mode register	GPIOx_OMODE	0x04	0x0000 0000
Port output speed register	GPIOx_OSPD	0x08	GPIOB: 0x0000 00C0; Others: 0x0000 0000
Port pull-up / down register	GPIOx_PUD	0x0C	GPIOA: 0x6400 0000; Others: 0x0000 0000
Port input status register	GPIOx_ISTAT	0x10	0x0000 XXXX
Port output control register	GPIOx_OCTL	0x14	0x0000 0000
Port bit operate register	GPIOx_BOP	0x18	0x0000 0000
Port configuration lock register	GPIOx_LOCK	0x1C	0x0000 0000
Alternate function selected register 0	GPIOx_AFSEL0	0x20	0x0000 0000
Alternate function selected register 1	GPIOx_AFSEL1	0x24	0x0000 0000
Bit clear register	GPIOx_BC	0x28	0x0000 0000
Port bit toggle register	GPIOx_TG	0x2C	0x0000 0000
Port safe mode control register	GPIOx_SMCCTL	0x38	GPIOA: 0x0000 2000; GPIOB: 0x0000 0008; Others: 0x0000 0000
Port filtering enable register	GPIOx_IFEN	0x3C	0x0000 0000
Port retain enable register	GPIOx_PREN	0x40	0x0000 0000
Port slew rate enable register	GPIOx_SREN	0x44	GPIOA: 0x0000 001E; GPIOB: 0x0000 0001; GPIOC: 0x0000 0004;

Register name	Symbol	Address offset	Reset value
			GPIOG: 0x0000 0002; GPIOH: 0x0000 003C; Others: 0x0000 0000
Interrupt filtering control Register	GPIOx_IFCTL	0x48	0x0000 0000
Interrupt filtering sample counter control register 0	GPIOx_IFSCCTL0	0x4C	0x0000 0000
Interrupt filtering sample counter control register 1	GPIOx_IFSCCTL1	0x50	0x0000 0000
TMR alarm status register (Port B, E, J, K only)	GPIOx_TMRASTAT	0x54	GPIOB: 0x0000 0040; GPIOE / GPIOJ: 0x0000 0015; GPIOK: 0x0000 0005

8.4.1. Port control register (GPIOx_CTL, x=A···N)

Address offset: 0x00

Reset value: 0xABFF FFFF for port A; 0xFFFF FFBF for port B; others 0xFFFF FFFF

This register can be written by byte (8-bit), half-word (16-bit) and word (32-bit) and can be read by word (32-bit).

PC14 and PC15 only have digital input and analog functions.

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16
CTL15[1:0]	CTL14[1:0]	CTL13[1:0]	CTL12[1:0]	CTL11[1:0]	CTL10[1:0]	CTL9[1:0]	CTL8[1:0]
rw	rw	rw	rw	rw	rw	rw	rw
15 14	**13 12**	**11 10**	**9 8**	**7 6**	**5 4**	**3 2**	**1 0**
CTL7[1:0]	CTL6[1:0]	CTL5[1:0]	CTL4[1:0]	CTL3[1:0]	CTL2[1:0]	CTL1[1:0]	CTL0[1:0]
rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:30	CTL15[1:0]	Pin 15 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
29:28	CTL14[1:0]	Pin 14 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description

27:26	CTL13[1:0]	Pin 13 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
25:24	CTL12[1:0]	Pin 12 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
23:22	CTL11[1:0]	Pin 11 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
21:20	CTL10[1:0]	Pin 10 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
19:18	CTL9[1:0]	Pin 9 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
17:16	CTL8[1:0]	Pin 8 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
15:14	CTL7[1:0]	Pin 7 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
13:12	CTL6[1:0]	Pin 6 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
11:10	CTL5[1:0]	Pin 5 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
9:8	CTL4[1:0]	Pin 4 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
7:6	CTL3[1:0]	Pin 3 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
5:4	CTL2[1:0]	Pin 2 configuration bits These bits are set and cleared by software. Refer to CTL0[1:0] description
3:2	CTL1[1:0]	Pin 1 configuration bits These bits are set and cleared by software.

		Refer to CTL0[1:0] description
1:0	CTL0[1:0]	Pin 0 configuration bits These bits are set and cleared by software. 00: Input mode 01: GPIO output mode 10: Alternate function mode 11: Analog mode (Input and Output)

8.4.2. Port output mode register (GPIOx_OMODE, x=A...N)

Address offset: 0x04
Reset value: 0x0000 0000

This register can be written by byte (8-bit), half-word (16-bit) and word (32-bit) and can be read by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OM15	OM14	OM13	OM12	OM11	OM10	OM9	OM8	OM7	OM6	OM5	OM4	OM3	OM2	OM1	OM0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15	OM15	Pin 15 output mode bit These bits are set and cleared by software. Refer to OM0 description
14	OM14	Pin 14 output mode bit These bits are set and cleared by software. Refer to OM0 description
13	OM13	Pin 13 output mode bit These bits are set and cleared by software. Refer to OM0 description
12	OM12	Pin 12 output mode bit These bits are set and cleared by software. Refer to OM0 description
11	OM11	Pin 11 output mode bit These bits are set and cleared by software. Refer to OM0 description
10	OM10	Pin 10 output mode bit These bits are set and cleared by software.

		Refer to OM0 description
9	OM9	Pin 9 output mode bit These bits are set and cleared by software. Refer to OM0 description
8	OM8	Pin 8 output mode bit These bits are set and cleared by software. Refer to OM0 description
7	OM7	Pin 7 output mode bit These bits are set and cleared by software. Refer to OM0 description
6	OM6	Pin 6 output mode bit These bits are set and cleared by software. Refer to OM0 description
5	OM5	Pin 5 output mode bit These bits are set and cleared by software. Refer to OM0 description
4	OM4	Pin 4 output mode bit These bits are set and cleared by software. Refer to OM0 description
3	OM3	Pin 3 output mode bit These bits are set and cleared by software. Refer to OM0 description
2	OM2	Pin 2 output mode bit These bits are set and cleared by software. Refer to OM0 description
1	OM1	Pin 1 output mode bit These bits are set and cleared by software. Refer to OM0 description
0	OM0	Pin 0 output mode bit These bits are set and cleared by software. 0: Output push-pull mode (reset value) 1: Output open-drain mode

8.4.3. Port output speed register (GPIOx_OSPD, x=A...N)

Address offset: 0x08

Reset value: 0x0000 00C0 for port B; Others 0x0000 0000

This register can be written by byte (8-bit), half-word (16-bit) and word (32-bit) and can be read by word (32-bit).

Table 8-11. Pin type(1)

Port/Pin	Pin 0	Pin 1	Pin 2	Pin 3	Pin 4	Pin 5	Pin 6	Pin 7	Pin 8	Pin 9	Pin 10	Pin 11	Pin 12	Pin 13	Pin 14(2)	Pin 15(2)
GPIOA	2	3	4	4	4	2	2	2	2	1	1	1	1	2	2	2
GPIOB	3	2	2	3	1	1	1	2	-	-	2	2	1	1	1	1
GPIOC	1	1	3	2	2	2	1	1	1	2	2	-	-	1	-	-
GPIOD	1	-	-	-	-	-	-	-	1	1	1	1	1	1	2	1
GPIOE	-	-	1	1	1	1	1	2	1	1	1	1	1	1	1	2
GPIOF	2	2	1	2	2	2	1	2	2	1	1	1	1	2	2	2
GPIOG	2	3	1	1	1	1	1	1	1	-	-	-	-	-	-	-
GPIOH	-	-	4	4	3	4	2	2	2	2	2	2	1	1	1	-
GPIOI	-	1	-	-	-	-	-	-	1	1	1	1	1	1	1	1
GPIOJ	1	1	1	1	2	2	1	1	1	1	1	1	-	-	-	-
GPIOK	1	1	1	-	-	-	-	-	-	-	-	-	-	-	-	-
GPIOL	1	1	1	1	1	2	-	-	-	-	-	-	-	-	-	-
GPIOM	1	1	2	1	1	2	-	-	-	-	-	-	-	-	-	-
GPION	1	1	1	1	1	-	-	-	-	-	-	-	-	-	-	-

Note:

(1) Pin type
1: Standard
2: Standard plus
3: Medium
4: Fast

(2) PC14 and PC15 cannot be used as digital output functions

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	OSPD15	Reserved	OSPD14	Reserved	OSPD13	Reserved	OSPD12	Reserved	OSPD11	Reserved	OSPD10[	Reserved	OSPD9	Reserved	OSPD8
	rw		rw		rw		rw		rw		rw		rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	OSPD7	Reserved	OSPD6	OSPD5[1:0]		Reserved	OSPD4	OSPD3[1:0]		Reserved	OSPD2	Reserved	OSPD1	OSPD0[1:0]	
	rw		rw	rw			rw	rw			rw		rw	rw	

Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30	OSPD15	Pin 15 output max speed bits These bits are set and cleared by software. Refer to OSPD1 description
29	Reserved	Must be kept at reset value.
28	OSPD14	Pin 14 output max speed bits These bits are set and cleared by software. Refer to OSPD1 description
27	Reserved	Must be kept at reset value.
26	OSPD13	Pin 13 output max speed bits These bits are set and cleared by software.

		Refer to OSPD1 description
25	Reserved	Must be kept at reset value.
24	OSPD12	Pin 12 output max speed bits These bits are set and cleared by software. Refer to OSPD1 description
23	Reserved	Must be kept at reset value.
22	OSPD11	Pin 11 output max speed bits These bits are set and cleared by software. Refer to OSPD1 description
21	Reserved	Must be kept at reset value.
20	OSPD10	Pin 10 output max speed bits These bits are set and cleared by software. Refer to OSPD1 description
19	Reserved	Must be kept at reset value.
18	OSPD9	Pin 9 output max speed bits These bits are set and cleared by software. Refer to OSPD1 description
17	Reserved	Must be kept at reset value.
16	OSPD8	Pin 8 output max speed bits These bits are set and cleared by software. Refer to OSPD1 description
15	Reserved	Must be kept at reset value.
14	OSPD7	Pin 7 output max speed bits These bits are set and cleared by software. Refer to OSPD1 description
13	Reserved	Must be kept at reset value.
12	OSPD6	Pin 6 output max speed bits These bits are set and cleared by software. Refer to OSPD1 description
11:10	OSPD5[1:0]	Pin 5 output max speed bits **For PF5:** 00: Output max speed 12.5M 01: Output max speed 25M 1x: Output max speed 50M **For other pins, bit11 reserved:** GPIO standard

		0/1: Output max speed 10M GPIO standard plus 0: Output max speed 12.5M 1: Output max speed 25M GPIO medium 0: Output max speed 25M 1: Output max speed 50M GPIO fast 0: Output max speed 50M 1: Output max speed 120M
9	Reserved	Must be kept at reset value.
8	OSPD4	Pin 4 output max speed bits These bits are set and cleared by software. Refer to OSPD1 description
7:6	OSPD3[1:0]	Pin 3 output max speed bits These bits are set and cleared by software. **For PC3:** 00: Output max speed 12.5M 01: Output max speed 25M 1x: Output max speed 50M **For other pins, bit7 reserved:** GPIO standard 0/1: Output max speed 10M GPIO standard plus 0: Output max speed 12.5M 1: Output max speed 25M GPIO medium 0: Output max speed 25M 1: Output max speed 50M GPIO fast 0: Output max speed 50M 1: Output max speed 120M
5	Reserved	Must be kept at reset value.
4	OSPD2	Pin 2 output max speed bits These bits are set and cleared by software. Refer to OSPD1 description
3	Reserved	Must be kept at reset value.

2	OSPD1	Pin 1 output max speed bits These bits are set and cleared by software. GPIO standard 0/1: Output max speed 10M GPIO standard plus 0: Output max speed 12.5M 1: Output max speed 25M GPIO medium 0: Output max speed 25M 1: Output max speed 50M GPIO fast 0: Output max speed 50M 1: Output max speed 120M
1:0	OSPD0[1:0]	Pin 0 output max speed bits These bits are set and cleared by software. **For PA0:** 00: Output max speed 12.5M 01: Output max speed 25M 1x: Output max speed 50M **For other pins, bit1 reserved:** GPIO standard 0/1: Output max speed 10M GPIO standard plus 0: Output max speed 12.5M 1: Output max speed 25M GPIO medium 0: Output max speed 25M 1: Output max speed 50M GPIO fast 0: Output max speed 50M 1: Output max speed 120M

8.4.4. Port pull-up / down register (GPIOx_PUD, x=A...N)

Address offset: 0x0C

Reset value: 0x6400 0000 for port A; Others 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) and word (32-bit).

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16
PUD15[1:0]	PUD14[1:0]	PUD13[1:0]	PUD12[1:0]	PUD11[1:0]	PUD10[1:0]	PUD9[1:0]	PUD8[1:0]

rw	rw	rw	rw	rw	rw	rw	rw
15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
PUD7[1:0]	PUD6[1:0]	PUD5[1:0]	PUD4[1:0]	PUD3[1:0]	PUD2[1:0]	PUD1[1:0]	PUD0[1:0]
rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:30	PUD15[1:0]	Pin 15 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
29:28	PUD14[1:0]	Pin 14 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
27:26	PUD13[1:0]	Pin 13 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
25:24	PUD12[1:0]	Pin 12 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
23:22	PUD11[1:0]	Pin 11 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
21:20	PUD10[1:0]	Pin 10 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
19:18	PUD9[1:0]	Pin 9 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
17:16	PUD8[1:0]	Pin 8 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
15:14	PUD7[1:0]	Pin 7 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
13:12	PUD6[1:0]	Pin 6 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
11:10	PUD5[1:0]	Pin 5 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description

9:8	PUD4[1:0]	Pin 4 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
7:6	PUD3[1:0]	Pin 3 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
5:4	PUD2[1:0]	Pin 2 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
3:2	PUD1[1:0]	Pin 1 pull-up or pull-down bits These bits are set and cleared by software. Refer to PUD0[1:0] description
1:0	PUD0[1:0]	Pin 0 pull-up or pull-down bits These bits are set and cleared by software. 00: Floating mode, no pull-up and pull-down (reset value) 01: With pull-up mode 10: With pull-down mode 11: Reserved

8.4.5. Port input status register (GPIOx_ISTAT, x=A...N)

Address offset: 0x10

Reset value: 0x0000 XXXX

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ISTAT15	ISTAT14	ISTAT13	ISTAT12	ISTAT11	ISTAT10	ISTAT9	ISTAT8	ISTAT7	ISTAT6	ISTAT5	ISTAT4	ISTAT3	ISTAT2	ISTAT1	ISTAT0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	ISTATy	Port input status (y=0…15) These bits are set and cleared by hardware. 0: Input signal low 1: Input signal high

8.4.6. Port output control register (GPIOx_OCTL, x=A...N)

Address offset: 0x14

Reset value: 0x0000 0000

This register can be written by byte (8-bit), half-word (16-bit) and word (32-bit) and can be read by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OCTL15	OCTL14	OCTL13	OCTL12	OCTL11	OCTL10	OCTL9	OCTL8	OCTL7	OCTL6	OCTL5	OCTL4	OCTL3	OCTL2	OCTL1	OCTL0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	OCTLy	Port output control (y=0..15) These bits are set and cleared by software. 0: Pin output low 1: Pin output high

8.4.7. Port bit operate register (GPIOx_BOP, x=A...N)

Address offset: 0x18

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) and word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CR15	CR14	CR13	CR12	CR11	CR10	CR9	CR8	CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOP15	BOP14	BOP13	BOP12	BOP11	BOP10	BOP9	BOP8	BOP7	BOP6	BOP5	BOP4	BOP3	BOP2	BOP1	BOP0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bits	Fields	Descriptions
31:16	CRy	Port clear bit y (y=0..15) These bits are set and cleared by software. 0: No action on the corresponding OCTLy bit 1: Clear the corresponding OCTLy bit to 0
15:0	BOPy	Port set bit y (y=0..15) These bits are set and cleared by software. 0: No action on the corresponding OCTLy bit 1: Set the corresponding OCTLy bit to 1

8.4.8. Port configuration lock register (GPIOx_LOCK, x=A...N)

Address offset: 0x1C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															LKK
															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LK15	LK14	LK13	LK12	LK11	LK10	LK9	LK8	LK7	LK6	LK5	LK4	LK3	LK2	LK1	LK0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value
16	LKK	Lock key It can only be set by using the lock key writing sequence. And is always readable. 0: GPIOx_LOCK register and the port configuration are not locked 1: GPIOx_LOCK register is locked until an MCU reset LOCK key writing sequence: Write 1->Write 0->Write 1-> Read 0-> Read 1 Note: The value of LKy (y=0...15) must be held during the LOCK Key writing sequence.
15:0	LKy	Port lock bit y (y=0...15) These bits are set and cleared by software. 0: Port configuration not locked 1: Port configuration locked when LKK bit is "1"

8.4.9. Alternate function selected register 0 (GPIOx_AFSEL0, x=A...N)

Address offset: 0x20

Reset value: 0x0000 0000

This register can be written by byte (8-bit), half-word (16-bit) and word (32-bit) and can be read by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL7[3:0]				SEL6[3:0]				SEL5[3:0]				SEL4[3:0]			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL3[3:0]				SEL2[3:0]				SEL1[3:0]				SEL0[3:0]			
rw				rw				rw				rw			

Bits	Fields	Descriptions
31:28	SEL7[3:0]	Pin 7 alternate function selected These bits are set and cleared by software. Refer to SEL0[3:0] description

27:24	SEL6[3:0]	Pin 6 alternate function selected These bits are set and cleared by software. Refer to SEL0[3:0] description
23:20	SEL5[3:0]	Pin 5 alternate function selected These bits are set and cleared by software. Refer to SEL0[3:0] description
19:16	SEL4[3:0]	Pin 4 alternate function selected These bits are set and cleared by software. Refer to SEL0[3:0] description
15:12	SEL3[3:0]	Pin 3 alternate function selected These bits are set and cleared by software. Refer to SEL0[3:0] description
11:8	SEL2[3:0]	Pin 2 alternate function selected These bits are set and cleared by software. Refer to SEL0[3:0] description
7:4	SEL1[3:0]	Pin 1 alternate function selected These bits are set and cleared by software. Refer to SEL0[3:0] description
3:0	SEL0[3:0]	Pin 0 alternate function selected These bits are set and cleared by software. 0000: AF0 selected (reset value) 0001: AF1 selected 0010: AF2 selected 0011: AF3 selected 0100: AF4 selected 0101: AF5 selected 0110: AF6 selected 0111: AF7 selected 1000: AF8 selected 1001: AF9 selected 1010: AF10 selected 1011: AF11 selected 1100: AF12 selected 1101: AF13 selected 1110: AF14 selected 1111: AF15 selected

8.4.10. Alternate function selected register 1 (GPIOx_AFSEL1, x=A...N)

Address offset: 0x24

Reset value: 0x0000 0000

This register can be written by byte (8-bit), half-word (16-bit) and word (32-bit) and can be read by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15[3:0]				SEL14[3:0]				SEL13[3:0]				SEL12[3:0]			
rw				rw				rw				rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL11[3:0]				SEL10[3:0]				SEL9[3:0]				SEL8[3:0]			
rw				rw				rw				rw			

Bits	Fields	Descriptions
31:28	SEL15[3:0]	Pin 15 alternate function selected These bits are set and cleared by software. Refer to SEL8[3:0] description
27:24	SEL14[3:0]	Pin 14 alternate function selected These bits are set and cleared by software. Refer to SEL8[3:0] description
23:20	SEL13[3:0]	Pin 13 alternate function selected These bits are set and cleared by software. Refer to SEL8[3:0] description
19:16	SEL12[3:0]	Pin 12 alternate function selected These bits are set and cleared by software. Refer to SEL8[3:0] description
15:12	SEL11[3:0]	Pin 11 alternate function selected These bits are set and cleared by software. Refer to SEL8[3:0] description
11:8	SEL10[3:0]	Pin 10 alternate function selected These bits are set and cleared by software. Refer to SEL8[3:0] description
7:4	SEL9[3:0]	Pin 9 alternate function selected These bits are set and cleared by software. Refer to SEL8[3:0] description
3:0	SEL8[3:0]	Pin 8 alternate function selected These bits are set and cleared by software. 0000: AF0 selected (reset value) 0001: AF1 selected 0010: AF2 selected 0011: AF3 selected 0100: AF4 selected 0101: AF5 selected 0110: AF6 selected 0111: AF7 selected

1000: AF8 selected
1001: AF9 selected
1010: AF10 selected
1011: AF11 selected
1100: AF12 selected
1101: AF13 selected
1110: AF14 selected
1111: AF15 selected

8.4.11. Bit clear register (GPIOx_BC, x=A..N)

Address offset: 0x28
Reset value: 0x0000 0000

This register can be written by byte (8-bit), half-word (16-bit) and word (32-bit) and can be read by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CR15	CR14	CR13	CR12	CR11	CR10	CR9	CR8	CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CRy	Port clear bit y (y=0..15) These bits are set and cleared by software. 0: No action on the corresponding OCTLy bit 1: Clear the corresponding OCTLy bit

8.4.12. Port bit toggle register (GPIOx_TG, x=A...N)

Address offset: 0x2C
Reset value: 0x0000 0000

This register can be written by byte (8-bit), half-word (16-bit) and word (32-bit) and can be read by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TG15	TG14	TG13	TG12	TG11	TG10	TG9	TG8	TG7	TG6	TG5	TG4	TG3	TG2	TG1	TG0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	TGy	Port toggle bit y (y=0..15) These bits are set and cleared by software. 0: No action on the corresponding OCTLy bit 1: Toggle the corresponding OCTLy bit

8.4.13. Port safe mode control register (GPIOx_SMCCTL, x=A...N)

Address offset: 0x38
Reset value: 0x0000 2000 for port A; 0x0000 0008 for port B; Others 0x0000 0000

This register can be written by byte (8-bit), half-word (16-bit) and word (32-bit) and can be read by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMC15	SMC14	SMC13	SMC12	SMC11	SMC10	SMC9	SMC8	SMC7	SMC6	SMC5	SMC4	SMC3	SMC2	SMC1	SMC0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	SMCy	Pin y safe mode control (y = 0...15) Used only when the associated destination is a chip pin. Specifies whether the chip disables the pin's output buffer when the chip enters safe mode. 0: Disable safe mode control. (The output buffer returns to its previous state when the chip leaves safe mode.) 1: Enable safe mode control.

8.4.14. Port filtering enable register (GPIOx_IFEN, x=A...N)

Address offset: 0x3C
Reset value: 0x0000 0000

This register can be written by byte (8-bit), half-word (16-bit) and word (32-bit) and can be read by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IFEN15	IFEN14	IFEN13	IFEN12	IFEN11	IFEN10	IFEN9	IFEN8	IFEN7	IFEN6	IFEN5	IFEN4	IFEN3	IFEN2	IFEN1	IFEN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	IFENy	Pin y input filter enable (y = 0...15) 0: Disabled input filter. 1: Enabled input filter.

8.4.15. Port retain enable register (GPIOx_PREN, x=A...N)

Address offset: 0x40
Reset value: 0x0000 0000

This register can be written by byte (8-bit), half-word (16-bit) and word (32-bit) and can be read by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PREN15	PREN14	PREN13	PREN12	PREN11	PREN10	PREN9	PREN8	PREN7	PREN6	PREN5	PREN4	PREN3	PREN2	PREN1	PREN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	PRENy	Pin y (y = 0...15) pad retain enable. 0: Disable pad retain. 1: Enable pad retain.

8.4.16. Port slew rate enable register (GPIOx_SREN, x=A...N)

Address offset: 0x44
Reset value: 0x0000 001E for port A; 0x0000 0001 for port B; 0x0000 0004 for port C; 0x0000 0002 for port G; 0x0000 003C for port H; Others 0x0000 0000

This register can be written by byte (8-bit), half-word (16-bit) and word (32-bit) and can be read by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SREN15	SREN14	SREN13	SREN12	SREN11	SREN10	SREN9	SREN8	SREN7	SREN6	SREN5	SREN4	SREN3	SREN2	SREN1	SREN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions

31:16	Reserved	Must be kept at reset value
15:0	SRENy	Pin y slew rate enable(y = 0...15) 0: Disable slew rate. 1: Enable slew rate (PAD_IO waveform of rising and falling along slowly).

8.4.17. Interrupt filtering control Register (GPIOx_IFCTL, x=A...N)

Address offset: 0x48

Reset value: 0x0000 0000

This register can be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IQSP[3:0]				Reserved											
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EIFE15	EIFE14	EIFE13	EIFE12	EIFE11	EIFE10	EIFE9	EIFE8	EIFE7	EIFE6	EIFE5	EIFE4	EIFE3	EIFE2	EIFE1	EIFE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	IQSP [3:0]	External interrupt qualification sampling period for pin 0 to pin15: Prescaled filter Clock period is IRC48M × (1+ IQSP).
27:16	Reserved	Must be kept at reset value
15:0	EIFEy	Pin y enables digital glitch filter on the interrupt pad input. 0: Disabled digital glitch filter. 1: Enabled digital glitch filter.

8.4.18. Interrupt filtering sample counter control register 0(GPIOx_IFSCCTL0, x=A...N)

Address offset: 0x4C

Reset value: 0x0000 0000

This register can be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IFSC7[3:0]				IFSC6[3:0]				IFSC5[3:0]				IFSC4[3:0]			
rw				rw				rw				rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IFSC3[3:0]				IFSC2[3:0]				IFSC1[3:0]				IFSC0[3:0]			
rw				rw				rw				rw			

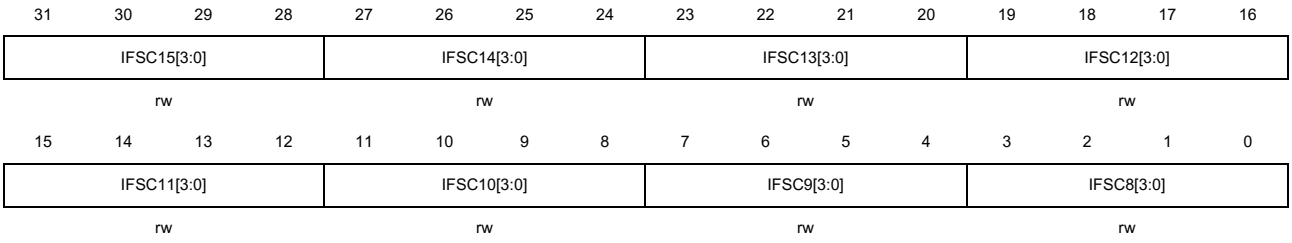
Bits	Fields	Descriptions
31:28	IFSC7[3:0]	Pin 7 maximum interrupt filter counter setting

		Refer to IFSC0[3:0] description
27:24	IFSC6[3:0]	Pin 6 maximum interrupt filter counter setting Refer to IFSC0[3:0] description
23:20	IFSC5[3:0]	Pin 5 maximum interrupt filter counter setting Refer to IFSC0[3:0] description
19:16	IFSC4[3:0]	Pin 4 maximum interrupt filter counter setting Refer to IFSC0[3:0] description
15:12	IFSC3[3:0]	Pin 3 maximum interrupt filter counter setting Refer to IFSC0[3:0] description
11:8	IFSC2[3:0]	Pin 2 maximum interrupt filter counter setting Refer to IFSC0[3:0] description
7:4	IFSC1[3:0]	Pin 1 maximum interrupt filter counter setting Refer to IFSC0[3:0] description
3:0	IFSC0[3:0]	Pin 0 maximum interrupt filter counter setting 0000-0010: all pass filter. 0011-1111: set the filter period to TCK ×IFSC0[3:0] + N * IRC48M (N = 0...4, N accounts for the uncertainty factor in filter period calculation. TCK is the prescaled filter clock period, which is the IRC48M clock prescaled to the IQSP value specified in GPIO_IFCTL)

8.4.19. Interrupt filtering sample counter control register 1(GPIOx_IFSCCTL1, x=A...N)

Address offset: 0x50
Reset value: 0x0000 0000

This register can be accessed by word (32-bit).



Bits	Fields	Descriptions
31:28	IFSC15[3:0]	Pin 15 maximum interrupt filter counter setting Refer to IFSC0[3:0] description
27:24	IFSC14[3:0]	Pin 14 maximum interrupt filter counter setting Refer to IFSC0[3:0] description

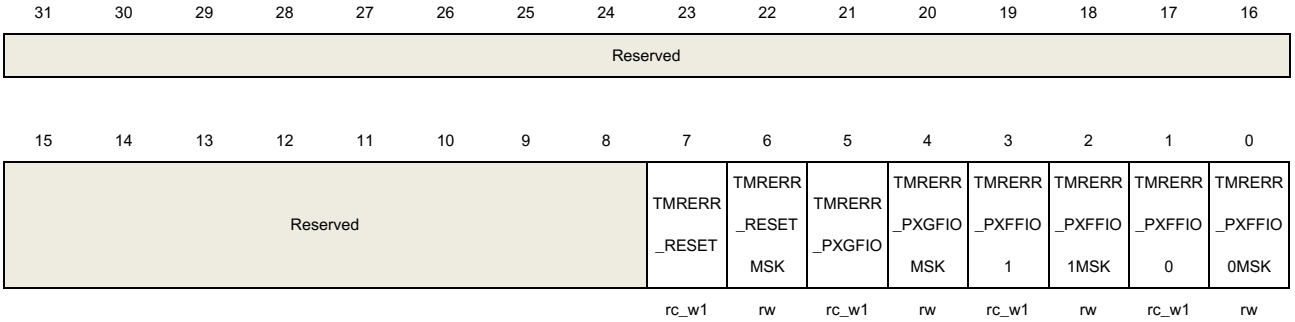
23:20	IFSC13[3:0]	Pin 13 maximum interrupt filter counter setting Refer to IFSC0[3:0] description
19:16	IFSC12[3:0]	Pin 12 maximum interrupt filter counter setting Refer to IFSC0[3:0] description
15:12	IFSC11[3:0]	Pin 11 maximum interrupt filter counter setting Refer to IFSC0[3:0] description
11:8	IFSC10[3:0]	Pin 10 maximum interrupt filter counter setting Refer to IFSC0[3:0] description
7:4	IFSC9[3:0]	Pin 9 maximum interrupt filter counter setting Refer to IFSC0[3:0] description
3:0	IFSC8[3:0]	Pin 8 maximum interrupt filter counter setting Refer to IFSC0[3:0] description

8.4.20. TMR alarm status register (GPIOx_TMRASTAT, x=B, E, J, K)

Address offset: 0x54

Reset value: 0x0000 0040 for port B; 0x0000 0015 for port E / J; 0x0000 0005 for port K;

This register can be accessed by word (32-bit).



Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value
7	TMRERR_RESET	RESET port mode configuration TMR error alarm RESET port reference registers contain GPIOB_CTL, GPIOB_AFSEL0. This bit is cleared by write 1. 0: No RESET port reference registers TMR error alarm detected 1: RESET port reference registers TMR error alarm detected Note: GPIOE_TMRASTAT / GPIOJ_TMRASTAT / GPIOK_TMRASTAT this bit is reserved.
6	TMRERR_RESETMSK	RESET port mode configuration TMR error alarm mask 0: Mask the TMR error alarm to FMU 1: Do not mask the TMR error alarm to FMU Note: GPIOE_TMRASTAT / GPIOJ_TMRASTAT / GPIOK_TMRASTAT this bit is

		reserved.
5	TMRERR_PXGFIO	GFIO port mode configuration TMR error alarm GFIO port reference registers contain GPIOx_CTL, GPIOx_AFSEL0, GPIOx_AFSEL1, GPIOx_OMODE, GPIOx_PUD, GPIOx_OSPD, GPIOx_SMCCTL, GPIOx_SREN, GPIOx_IFEN. x = E, J. This bit is cleared by write 1. 0: No GFIO port reference registers TMR error alarm detected 1: GFIO port reference registers TMR error alarm detected Note: GPIOB_TMRASTAT / GPIOK_TMRASTAT this bit is reserved.
4	TMRERR_PXGFIOMSK	GFIO port mode configuration TMR alarm error mask 0: Mask the TMR error alarm to FMU 1: Do not mask the TMR error alarm to FMU Note: GPIOB_TMRASTAT / GPIOK_TMRASTAT this bit is reserved.
3	TMRERR_PXFFIO1	FFIO1 port mode configuration TMR error alarm FFIO1 port reference registers contain GPIOx_CTL, GPIOx_AFSEL0, GPIOx_AFSEL1, GPIOx_OMODE, GPIOx_PUD, GPIOx_OSPD, GPIOx_SMCCTL, GPIOx_SREN, GPIOx_IFEN. x = E, J and K. This bit is cleared by write 1. 0: No FFIO1 port reference registers TMR error alarm detected 1: FFIO1 port reference registers TMR error alarm detected Note: GPIOB_TMRASTAT this bit is reserved.
2	TMRERR_PXFFIO1MSK	FFIO1 port mode configuration TMR alarm error mask 0: Mask the TMR error alarm to FMU 1: Do not mask the TMR error alarm to FMU Note: GPIOB_TMRASTAT this bit is reserved.
1	TMRERR_PXFFIO0	FFIO0 port mode configuration TMR error alarm FFIO0 port reference registers contain GPIOx_CTL, GPIOx_AFSEL0, GPIOx_AFSEL1, GPIOx_OMODE, GPIOx_PUD, GPIOx_OSPD, GPIOx_SMCCTL, GPIOx_SREN, GPIOx_IFEN. x = E, J and K. This bit is cleared by write 1. 0: No FFIO0 port reference registers TMR error alarm detected 1: FFIO0 port reference registers TMR error alarm detected Note: GPIOB_TMRASTAT this bit is reserved.
0	TMRERR_PXFFIO0MSK	FFIO0 port mode configuration TMR alarm mask 0: Mask the TMR error alarm to FMU 1: Do not mask the TMR error alarm to FMU Note: GPIOB_TMRASTAT this bit is reserved.

9. Cyclic redundancy checks management unit (CRC)

9.1. Overview

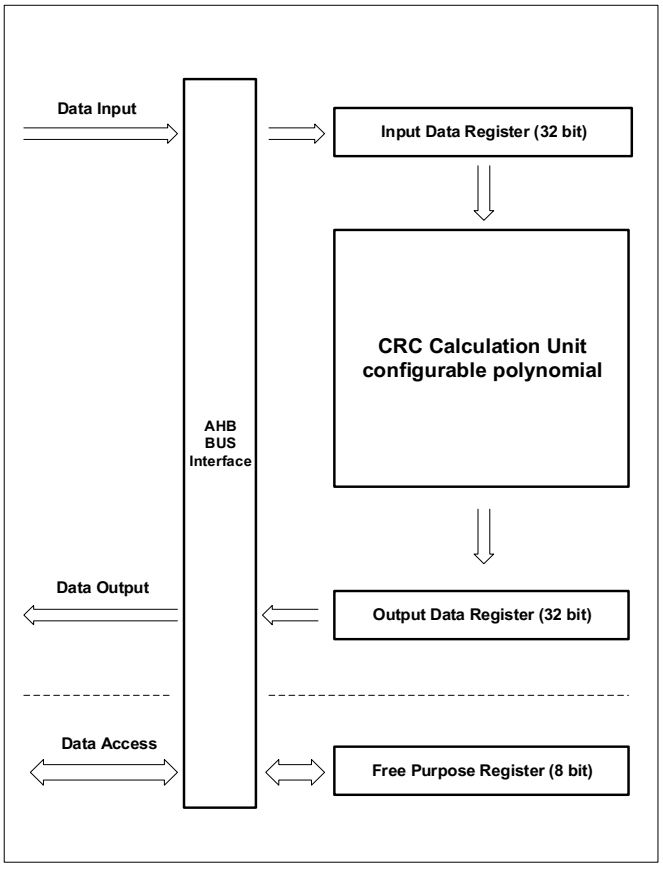
A cyclic redundancy check (CRC) is an error-detecting code commonly used in digital networks and storage devices to detect accidental changes to raw data.

This CRC management unit can be used to calculate 8 / 16 / 32 bits CRC code within user configurable polynomial.

9.2. Characteristics

- Supports 8 / 16 / 32 bits data input.
- For 8 / 16 / 32 bits input data length, the calculation cycles are 1 / 2 / 4 AHB clock cycles.
- User configurable polynomial value and size (7 / 8 / 16 / 32).
- User can configure CRC initial value.
- Free 8-bit register is unrelated to calculation and can be used for any other goals by any other peripheral devices.

Figure 9-1. Block diagram of CRC calculation unit



9.3. Function overview

- CRC calculation unit is used to calculate the 32-bit raw data, and CRC_DATA register will receive the raw data and store the calculation result.

  If the CRC_DATA register has not been cleared by setting the CRC_CTL register, the new input raw data will be calculated based on the result of previous value of CRC_DATA.

  CRC calculation will spend 4 / 2 / 1 AHB clock cycles for 32 / 16 / 8 bits data size. During this period, AHB will not be hanged because of the existence of the 32-bit input buffer.

- This module supplies an 8-bit free register CRC_FDATA.

  CRC_FDATA is unrelated to the CRC calculation. Independent read and write operations can be performed at any time.

- Reversible function can reverse the input data and output data.

  For input data, 3 reverse types can be selected.

  Original data is 0x3456CDEF:

  1) byte reverse:
  32-bit data is divided into 4 groups and reverse implement in group inside. Reversed data: 0x2C6AB3F7.

  2) half-word reverse:
  32-bit data is divided into 2 groups and reverse implement in group inside. Reversed data: 0x6A2CF7B3.

  3) word reverse:
  32-bit data is divided into 1 groups and reverse implement in group inside. Reversed data: 0xF7B36A2C.

  For output data, reverse type is word reverse.

  For example: when REV_O=1, calculation result 0x3344CCDD will be converted to 0xBB3322CC.

- User configurable initial calculation data is available.

  When RST bit is set or write operation to CRC_IDATA register, the CRC_DATA register will be automatically initialized to the value in CRC_IDATA.

- User configurable polynomial.

- Depends on PS[1:0] bits, the valid polynomial and output bit width can be selected by user. It is strongly recommended to reset the CRC calculation unit after changing the PS[1:0] bits or polynomial.

9.4. Register definition

CRC base address: 0x4900 3000

Table 9-1. CRC register overview

Register name	Symbol	Address offset	Reset value
Data register	CRC_DATA	0x00	0xFFFF FFFF
Free data register	CRC_FDATA	0x04	0x0000 0000
Control register	CRC_CTL	0x08	0x0000 0000
Initialization data register	CRC_IDATA	0x10	0xFFFF FFFF
Polynomial register	CRC_POLY	0x14	0x04C1 1DB7

9.4.1. Data register (CRC_DATA)

Address offset: 0x00

Reset value: 0xFFFF FFFF

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA[15:0]															
rw															

Bits	Fields	Descriptions
31:0	DATA[31:0]	CRC calculation result bits Software writes and reads. This register is used to calculate new data, and the register can be written the new data directly. Write value cannot be read because the read value is the previous CRC calculation result.

9.4.2. Free data register (CRC_FDATA)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FDATA[7:0]							

rw

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	FDATA[7:0]	Free data register bits Software writes and reads. These bits are unrelated with CRC calculation. This byte can be used for any goal by any other peripheral. The CRC_CTL register will generate no effect to the byte.

9.4.3. Control register (CRC_CTL)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								REV_O	REV_I[1:0]		PS[1:0]		Reserved		RST
								rw	rw		rw				rs

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	REV_O	Reverse output data value in bit order 0:Not bit reversed for output data 1:Bit reversed for output data
6:5	REV_I[1:0]	Reverse type for input data 0: Dot not use reverse for input data 1: Reverse input data with every 8-bit length 2: Reverse input data with every 16-bit length 3: Reverse input data with whole 32-bit length
4:3	PS[1:0]	Size of polynomial 0: 32 bit 1: 16 bit (POLY [15:0] is used for calculation.) 2: 8 bit (POLY [7:0] is used for calculation.) 3: 7 bit (POLY [6:0] is used for calculation.)
2:1	Reserved	Must be kept at reset value.
0	RST	Software writes and reads. Set this bit can reset the CRC_DATA register. When set, the value of the CRC_DATA register is automatically initialized to the

value in the CRC_IDATA register and then automatically cleared by hardware. This bit will take no effect to CRC_FDATA.

9.4.4. Initialization data register (CRC_IDATA)

Address offset: 0x10

Reset value: 0xFFFF FFFF

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IDATA[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDATA[15:0]															
rw															

Bits	Fields	Descriptions
31:0	IDATA[31:0]	Configurable initial CRC data value When RST bit in CRC_CTL asserted, CRC_DATA will be programmed to this value.

9.4.5. Polynomial register (CRC_POLY)

Address offset: 0x14

Reset value: 0x04C1 1DB7

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
POLY[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POLY[15:0]															
rw															

Bits	Fields	Description
31:0	POLY[31:0]	User configurable polynomial value This value is used together with PS[1:0] bits.

10. Direct memory access controller (DMA)

10.1. Overview

The direct memory access (DMA) controller provides a hardware method of transferring data between peripherals and/or memory without intervention from the MCU, thereby increasing system performance by off-loading the MCU from copying large amounts of data and avoiding frequent interrupts to serve peripherals needing more data or having available data.

Two AHB master interfaces and 8 FIFOs (FIFO with 16*64-bit depth for channel 0~2 and FIFO with 4*64-bit depth for channel 3~7) are presented in each DMA controller, which achieves a high DMA transmission performance. There are 16 independent channels in the DMA controller (8 for DMA0 and 8 for DMA1). Each channel is assigned a specific or multiple target peripheral devices for memory access request management. Two arbiters respectively for memory and peripheral are implemented inside to handle the priority among DMA requests.

Both the DMA controller and the Cortex®-M7 core implement data access through the system bus. An arbitration mechanism is implemented to solve the competition between these two masters. When the same peripheral is targeted, the MCU access will be suspended for some specific bus cycles. A round-robin scheduling algorithm is utilized in the bus matrix to guaranty at least half the bandwidth to the MCU.

10.2. Characteristics

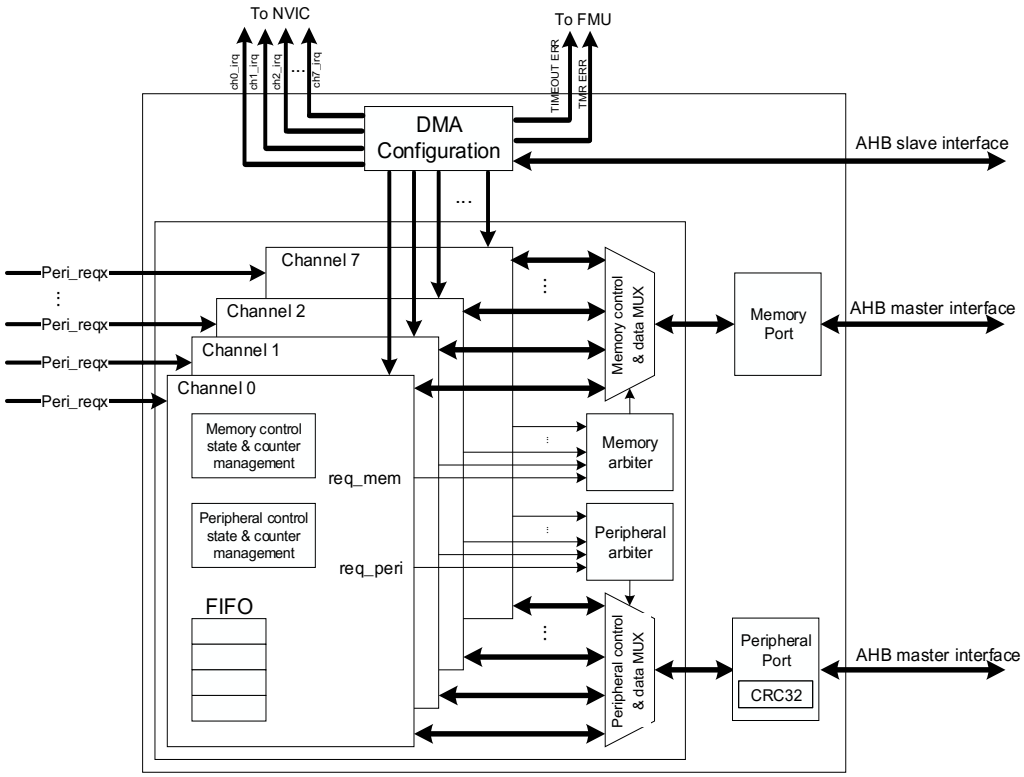
- Two AHB master interface for transferring data, and one AHB slave interface for programming DMA.
- 16 channels (8 for DMA0 and 8 for DMA1) and each channel are configurable.
- Support independent single, 4, 8, 16-beat incrementing burst memory and peripheral transfer.
- Each channel is associated either with a DMA request signal coming from a peripheral, or with a software trigger in memory-to-memory transfers. This configuration is done by software.
- Support switch-buffer transmission between peripheral and memory.
- Software DMA channel priority (low, medium, high, ultra-high) and hardware DMA channel priority (DMA channel 0 has the highest priority and DMA channel 15 has the lowest priority) or round-robin channel arbitration.
- Support independent 8, 16, 32, 64-bit memory and peripheral transfer.
- Support independent fixed and increasing address generation algorithm of memory and peripheral.
- Support circular transfer mode, programmable number of circular times (max to 256) and support independent memory address and peripheral address circular.
- Support three transfer modes:Read from memory and write to peripheral, Read from peripheral and write to memory, Read from memory and write to memory.

- Support multi-data transfer mode: Pack/Unpack data when memory transfer width are different from peripheral transfer width.
- Support single-data transfer mode: Read data from source when FIFO is empty and write data to destination when one data has been pushed into FIFO.
- Programmable length of data to be transferred, max to 65535.
- One separate interrupt per channel with five types of event flags.
- Support interrupt enable and clear.
- Support channel transfer timeout monitor.
- Support TMR for channel control register.
- Support CRC engine.

10.3. Function overview

10.3.1. Block diagram

Figure 10-1. DMA block diagram



As shown in ***Figure 10-1. DMA block diagram***, a DMA controller consists of four main parts:

- DMA configuration through AHB slave interface.
- Data access through two AHB master interfaces respectively for memory access and peripheral access.
- Two arbiters inside to manage multiple peripheral requests coming at the same time.
- Channel data management to control data packing/unpacking and counting.

The DMA controller transfers data from one address to another without CPU intervention. It supports multiple data sizes, burst types, address generation algorithm, priority levels and several transfer modes to allow for flexible application by configuring the corresponding bits in DMA registers. All the DMA registers can be 32-bit configured through AHB slave interface.

Three transfer modes are supported, including peripheral-to-memory, memory-to-peripheral and memory-to-memory, which is determined by the TM bits in the DMA_CHxCTL register, as listed in ***Table 10-1. Transfer mode***.

Table 10-1. Transfer mode

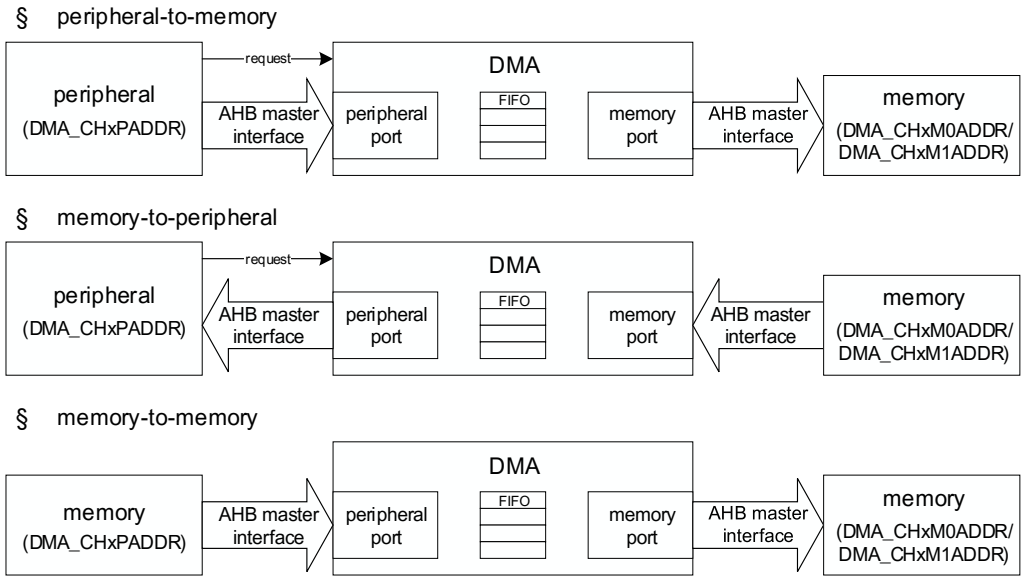
Transfer mode	TM[1:0]	Source	Destination
Peripheral to memory	00	DMA_CHxPADDR	DMA_CHxM0ADDR/ DMA_CHxM1ADDR
Memory to peripheral	01	DMA_CHxM0ADDR/ DMA_CHxM1ADDR	DMA_CHxPADDR
Memory to memory	10	DMA_CHxPADDR	DMA_CHxM0ADDR/ DMA_CHxM1ADDR

Note:

1. The MBS bit in DMA_CHxCTL register determines which is selected as the memory buffer address in DMA_CHxM0ADDR and DMA_CHxM1ADDR register. For more information, refer to section ***Switch-buffer mode***.

2. The TM bits in DMA_CHxCTL register are forbidden to configure to 0b11, or the channel will be automatically disabled.

Figure 10-2. Data stream for three transfer modes



As shown in ***Figure 10-2. Data stream for three transfer modes***, two AHB master interfaces are implemented in each DMA respectively for memory and peripheral.

- Memory to peripheral: read data from memory through AHB master interface for memory,

and write data to peripheral through AHB master interface for peripheral.

- Peripheral to memory: read data from peripheral through AHB master interface for peripheral, and write data to memory through AHB master interface for memory.
- Memory to memory: read data from memory through AHB master interface for peripheral, and write data to another memory through AHB master interface for memory.

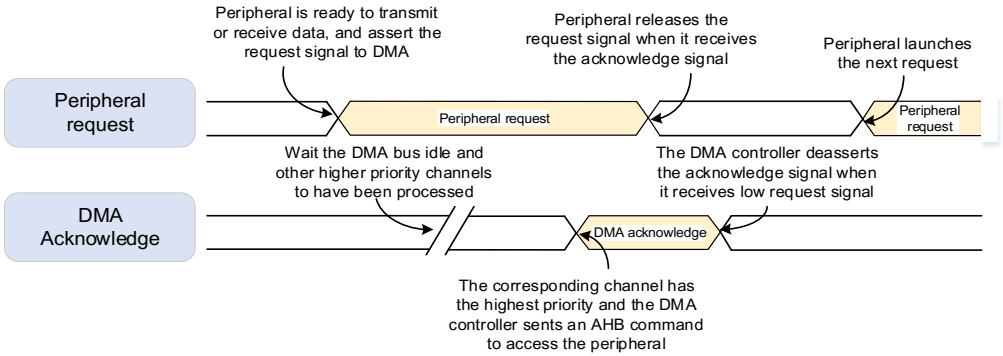
10.3.2. Peripheral handshake

To ensure a well-organized and efficient data transfer, a handshake mechanism is introduced between the DMA and peripherals, including a request signal and a acknowledge signal:

- Request signal asserted by peripheral to DMA controller, indicating that the peripheral is ready to transmit or receive data.
- Acknowledge signal responded by DMA to peripheral, indicating that the DMA controller has initiated an AHB command to access the peripheral.

Figure 10-3. Handshake mechanism shows how the handshake mechanism works between the DMA controller and peripherals.

Figure 10-3. Handshake mechanism



10.3.3. Data process

Arbitration

Two arbiters are implemented in each DMA respectively for memory and peripheral port.Peripheral request checking is performed on all enabled channels when the DMA is idle or after completing the data transfer for one peripheral request. When two or more requests are received at the same time, the arbiter determines which channel is selected to respond according to the following priority rules:

Arbitration uses a two-tier priority system; Software priority and channel priority. The DMA assigns each channel to a priority level by configuring the PRIO bits in the DMA_CHxCTL register. Software arbitration level is fixed-priority and cannot be changed. Channel arbitration uses fixed priority, the lower channel number with the higher priority; or can be configured to use a selective round-robin scheme for specified channels within each software priority level.

- Software priority: Four levels, including low, medium, high and ultra high by configuring the PRIO bits in the DMA_CHxCTL register.
- Channel priority: For channels with equal software priority level, fixed priority or round robin can be chosen:
  - Hardware priority: For channels with equal software priority level, priority is given to the channel with lower channel number if the round-robin arbitration is not enabled. For example, when channel 0 and channel 2 are configured with the same software priority, the priority of channel 0 is higher than that of channel 2.
  - Round Robin: When enable round-robin arbitration, for channels with equal software priority level, the round-robin channel arbitration can provide.

Transfer width, burst and counter

Transfer width

PWIDTH and MWIDTH in the DMA_CHxCTL register indicate the data width of a peripheral and memory transfer separately. The DMA supports 8-bit, 16-bit, 32-bit and 64-bit transfer width. In multi-data mode, if PWIDTH is not equal to MWIDTH, the DMA can automatically packs/unpacks data to achieve an integrated and correct data transfer operation. In single-data mode, MWIDTH is automatically locked as PWIDTH by hardware immediately after enable the DMA channel.

Transfer burst type

PBURST and MBURST in the DMA_CHxCTL register indicate the burst type of a peripheral and memory transfer separately. The DMA supports single burst, 4-beat, 8-beat and 16-beat incrementing burst for peripheral port and memory port. In single-data mode, only single burst type is supported and PBURST and MBURST are automatically locked as '00' by hardware immediately after enable the DMA channel.

In peripheral-to-memory or memory-to-peripheral mode, if PBURST is different from '00', DMA responses an increasing burst transfer of 4, 8, 16-beat based on the PBURST bits for each peripheral request. If the remaining bytes number of data item to be transferred is less than the bytes number needed for a burst transfer, the remaining data items are transferred in single transaction.

AMBA protocol specifies that bursts must not cross a 1KB address boundary, or a transfer error will be responded to the master. In each DMA, the peripheral burst transfer crossing a 1KB address boundary is decomposed to 4, 8 or 16 single transactions depend on the PBURST bits, as the same as the memory burst transfer.

Transfer counter

The CNT bits in the DMA_CHxCNT register specifies how many data to be transmitted on the channel and must be configured before the CHEN bit is enabled. During the transmission, the CNT bits indicate the remaining number of data items to be transferred.

The CNT bits are related to peripheral transfer width, the number of data bytes to be transferred is the CNT bits multiplied by the byte number of the peripheral transfer width. For example, if the PWIDTH bits are equal to '10', and the number of data bytes to be transferred is CNT×4. The CNT bits is decreased by 1 when a single or a beat of the burst peripheral transfer (the source memory transfer in the memory-to-memory mode) has been completed even if the transfer mode is peripheral-to-memory or memory-to-memory.

When configuring the CNT bits, the following rules must be respected to guarantee a good DMA operation:

1. If the circular mode is disabled by clearing the CMEN bit in the DMA_CHxCTL register, the rules to configure the CNT bits in the DMA_CHxCNT register based on the transfer width are listed in the *Table 10-2. CNT configuration*. The number of data bytes must be an integer multiple of the memory transfer width to guarantee an integrated single memory transfer.

Note: The number of data bytes does not need to be an integer multiple of the bytes number of a memory burst transfer or a peripheral burst number if the PBURST or/and MBURST bits are not equal to '00'. The remaining data that not enough for a burst transfer can be divided into single transfer automatically.

Table 10-2. CNT configuration

PWIDTH	MWIDTH	CNT
8-bit	16-bit	Multiple of 2
8-bit	32-bit	Multiple of 4
8-bit	64-bit	Multiple of 8
16-bit	32-bit	Multiple of 2
16-bit	64-bit	Multiple of 4
32-bit	64-bit	Multiple of 2
Others		Any value

2. If the circular mode is enabled by setting the CMEN bit in the DMA_CHxCTL register. The number of data bytes must be an integer multiple of the byte number of a peripheral burst transfer and a memory burst transfer to guarantee an integrated memory and peripheral burst transfer:

- $CNT / PBURST_beats$ must be an integer.
- $(CNT \times PWIDTH_bytes) / (MBURST_beats \times MWIDTH_bytes)$ must be an integer.
  - PWIDTH_bytes is the byte number of the peripheral transfer width, 1 for 8-bit, 2 for 16-bit, 4 for 32-bit and 8 for 64-bit.
  - PBURST_beats is the beat number of a peripheral burst transfer, 1 for single burst, 4 for INCR4 (4-beat incrementing burst), 8 for INCR8 (8-beat incrementing burst) and 16 for INCR16 (16-beat incrementing burst).
  - MWIDTH_bytes is the byte number of the memory transfer width, 1 for 8-bit, 2 for 16-bit, 4 for 32-bit and 8 for 64-bit.
  - MBURST_beats is the beat number of a memory burst transfer, 1 for single burst, 4

for INCR4, 8 for INCR8 and 16 for INCR16.

For example:

1. If PWIDTH is 16-bit, PBURST is INCR4, MWIDTH is 8-bit and MBURST is INCR16, CNT/4 and (CNT×2)/(1×16) must be an integer, so the CNT bits must be configured to the multiple of 8.
2. If PWIDTH is 8-bit, PBURST is INCR16, MWIDTH is 16-bit and MBURST is INCR4, CNT/16 and (CNT×1)/(2×4) must be an integer, so the CNT bits must be configured to the multiple of 16.

Note: When the switch-buffer mode is enabled by setting the SBMEN bit in the DMA_CHxCTL register, the circular mode is enabled automatically by hardware, and the above rules must also be respected.

FIFO

A FIFO is implemented as a data buffer for each DMA channel. Data reading from the source address is stored in the FIFO temporarily and transmitted to the destination through the destination port. Two data processing modes are supported depend on the FIFO configuration, including single-data mode and multi-data mode. When the transfer mode is memory-to-memory, only multi-data mode is supported to implement the DMA data processing.

Multi-data mode

The multi-data mode is selected by configuring the MDMEN bit in the DMA_CHxFCTL register to '1'.

In this mode, the DMA responds the source request when there is enough FIFO space for a source transfer, pushing the data reading from the source address into the FIFO. If the destination is a peripheral, the DMA responds the peripheral request when there is enough FIFO data for a peripheral burst transfer. If the memory is configured as the destination, the FIFO counter critical value configured in the FCCV bits of the DMA_CHxFCTL register controls the memory data processing. Only when the FIFO counter is reached the critical value, the data in the FIFO are entirely popped and written into the memory address.

To guarantee a good DMA behavior, the FIFO counter critical value (FCCV bits in the DMA_CHxFCTL register) must be an integer multiple of a memory burst transfer to ensure there is enough data for memory burst transfers. The configuration rules of the FIFO counter critical value depending on memory transfer width and memory burst types are listed in ***Table 10-3. FIFO counter critical value configuration rules for channel 3~7*** and ***Table 10-4. FIFO counter critical value configuration rules for channel 0~2***

Table 10-3. FIFO counter critical value configuration rules for channel 3~7

MWIDTH	MBURST	FIFO counter critical value			
		1 dual-word	2 dual-word	3 dual-word	4 dual-word
8-bit	single	8 single transaction	16 single transaction	24 single transaction	32 single transaction

MWIDTH	MBURST	FIFO counter critical value			
		1 dual-word	2 dual-word	3 dual-word	4 dual-word
	INCR4	2 burst transaction	4 burst transaction	6 burst transaction	8 burst transaction
	INCR8	1 burst transaction	2 burst transaction	3 burst transaction	4 burst transaction
	INCR16	ERROR	1 burst transaction	ERROR	2 burst transaction
16-bit	single	4 single transaction	8 single transaction	12 single transaction	16 single transaction
	INCR4	1 burst transaction	2 burst transaction	3 burst transaction	4 burst transaction
	INCR8	ERROR	1 burst transaction	ERRO ERROR R	2 burst transaction
	INCR16	ERROR	ERROR	ERROR	1 burst transaction
32-bit	single	2 single transaction	4 single transaction	6 single transaction	8 single transaction
	INCR4	ERROR	1 burst transaction	ERROR	2 burst transaction
	INCR8	ERROR	ERROR	ERROR	1 burst transaction
	INCR16	ERROR	ERROR	ERROR	ERROR
64-bit	single	1 single transaction	2 single transaction	3 single transaction	4 single transaction
	INCR4	ERROR	ERROR	ERROR	1 burst transaction
	INCR8	ERROR	ERROR	ERROR	ERROR
	INCR16	ERROR	ERROR	ERROR	ERROR

Table 10-4. FIFO counter critical value configuration rules for channel 0~2

MWIDTH	MBURST	FIFO counter critical value			
		4dual-word	8 dual-word	12 dual-word	16 dual-word
8-bit	single	32 single transaction	64 single transaction	96 single transaction	128 single transaction
	INCR4	8 burst transaction	16 burst transaction	24 burst transaction	32 burst transaction
	INCR8	4 burst transaction	8 burst transaction	12 burst transaction	16 burst transaction
	INCR16	2 burst transaction	4 burst transaction	6 burst transaction	8 burst transaction
16-bit	single	16 single transaction	32 single transaction	48 single transaction	64 single transaction
	INCR4	4 burst transaction	8 burst transaction	12 burst transaction	16 burst transaction
	INCR8	2 burst transaction	4 burst transaction	6 burst transaction	8 burst transaction
	INCR16	1 burst transaction	2 burst transaction	3 burst transaction	4 burst transaction
32-bit	single	8 single transaction	16 single transaction	24 single transaction	32 single transaction
	INCR4	2 burst transaction	4 burst transaction	6 burst transaction	8 burst transaction
	INCR8	1 burst transaction	2 burst transaction	3 burst transaction	4 burst transaction
	INCR16	ERROR	1 burst transaction	ERROR	2 burst transaction
64-bit	single	4 single transaction	8 single transaction	12 single transaction	16 single transaction

MWIDTH	MBURST	FIFO counter critical value			
		4dual-word	8 dual-word	12 dual-word	16 dual-word
	INCR4	1 burst transaction	2 burst transaction	3 burst transaction	4 burst transaction
	INCR8	ERROR	1 burst transaction	ERROR	2 burst transaction
	INCR16	ERROR	ERROR	ERROR	1 burst transaction

Note: The example configurations in the table are not exhaustive and only illustrate the configuration rules.

Note: When the transfer mode is peripheral-to-memory, if the PBURST_beats×PWIDTH_bytes = 32, the FIFO counter critical value must not be equal to '2b10'. When receiving a peripheral request, DMA initiates a peripheral burst transfer to entirely fill the FIFO. Then DMA lunches memory burst transfers to pop three words from the FIFO depending on the FIFO counter critical value and a word is still remained in the FIFO. There is no enough space for a peripheral burst transfer and the FIFO counter critical value is not reached, which makes DMA transfer frozen.

Single-data mode

The single-data mode is selected by configuring the MDMEN bit in the DMA_CHxFCTL register to '0'. In this mode, only single transfer is supported to implement the DMA data access, and the FIFO counter critical value configured in the FCCV bits of the DMA_CHxFCTL register has no meaning.

In single-data mode, DMA responds the source request only when the FIFO is empty, pushing the data reading from the source address into the FIFO whatever the source transfer width is. When the FIFO is not empty, DMA responds the destination request, popping the data from the FIFO and writing it to the destination address.

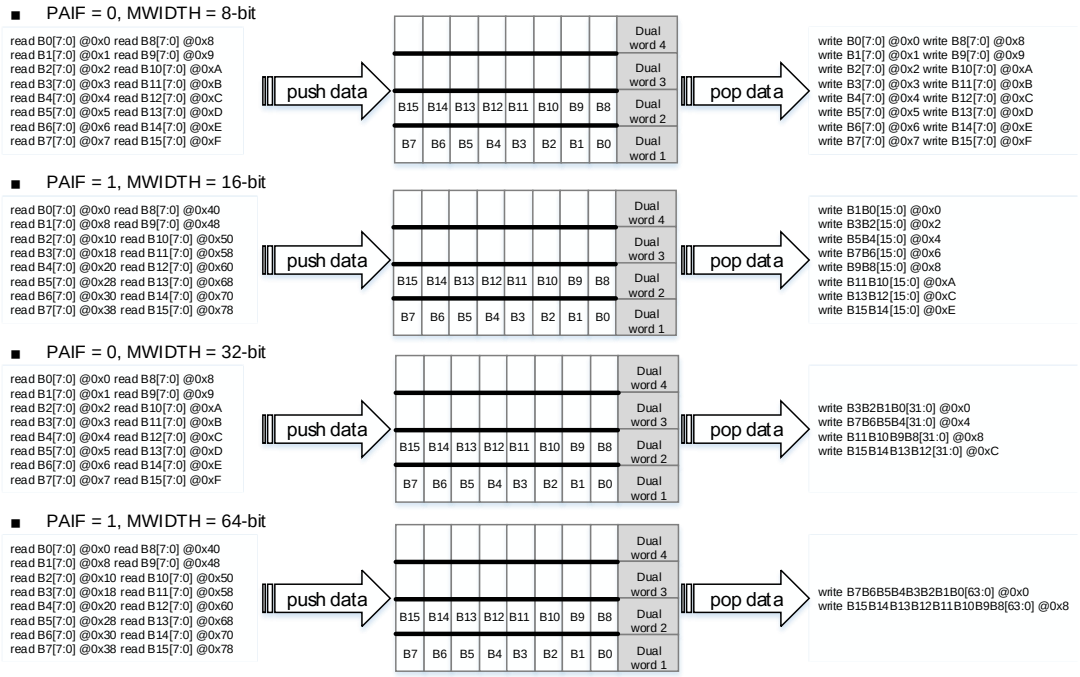
Pack/Unpack

In single-data mode, the MWIDTH bits are equal to the PWIDTH bits by force, data packing/unpacking is not needed.

In multi-data mode, the independent PWIDTH and MWIDTH bits configuration are supported for flexible DMA transfer. When the PWIDTH bits and the MWIDTH bits are not equal, DMA reading access and writing access are executed in different transfer width, and DMA packs/unpacks the data automatically. In DMA transfer operation, only little-endian addressing for both memory and peripheral is supported.

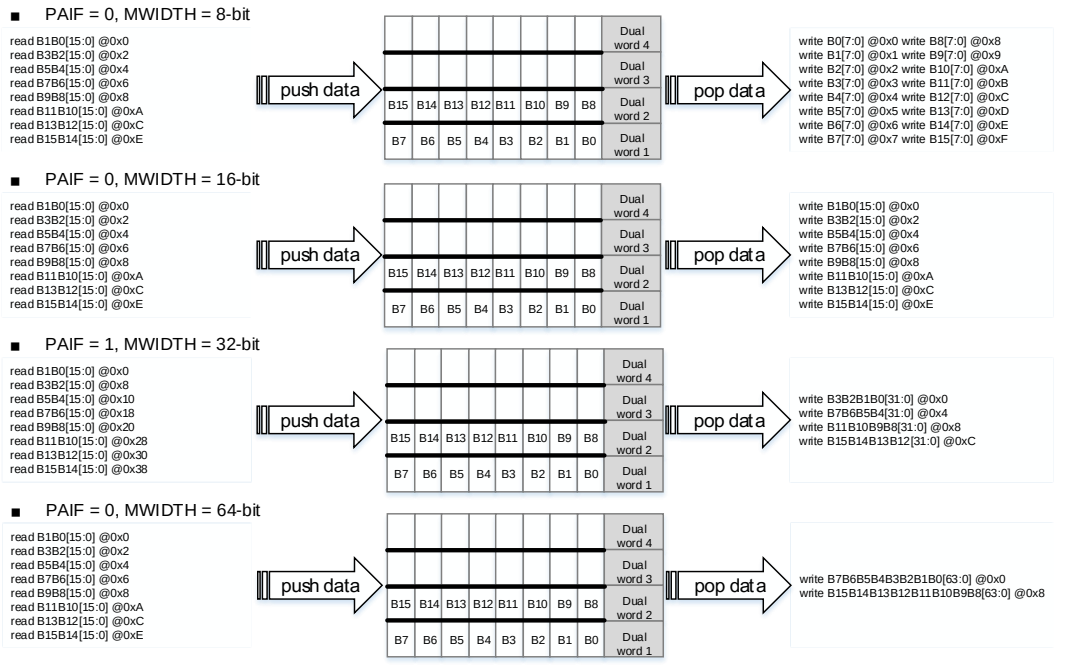
Suppose the CNT bits are 16, the PWIDTH bits are equal to '00', and both PNAGA and MNAGA are set. The DMA transfer operations for different MWIDTH are shown in the ***Figure 10-4. Data packing/unpacking when PWIDTH = '00'***.

Figure 10-4. Data packing/unpacking when PWIDTH = '00'



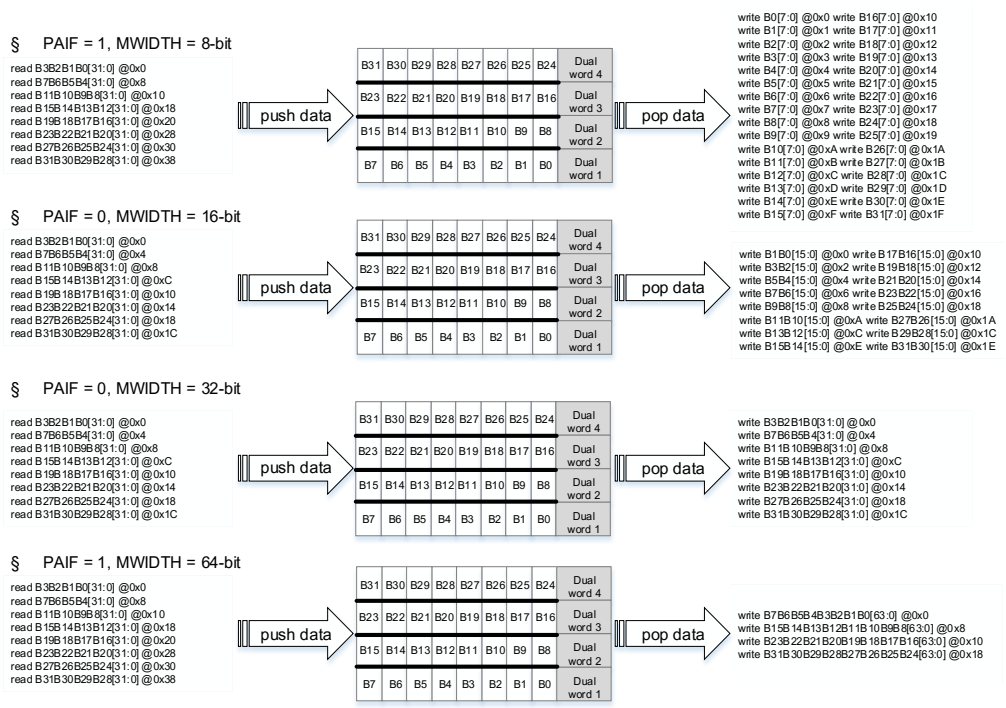
Suppose the CNT bits are 8, the PWIDTH bits are equal to '01', and both PNAGA and MNAGA are set. The DMA transfer operations for different MWIDTH are shown in the ***Figure 10-5. Data packing/unpacking when PWIDTH = '01'***.

Figure 10-5. Data packing/unpacking when PWIDTH = '01'



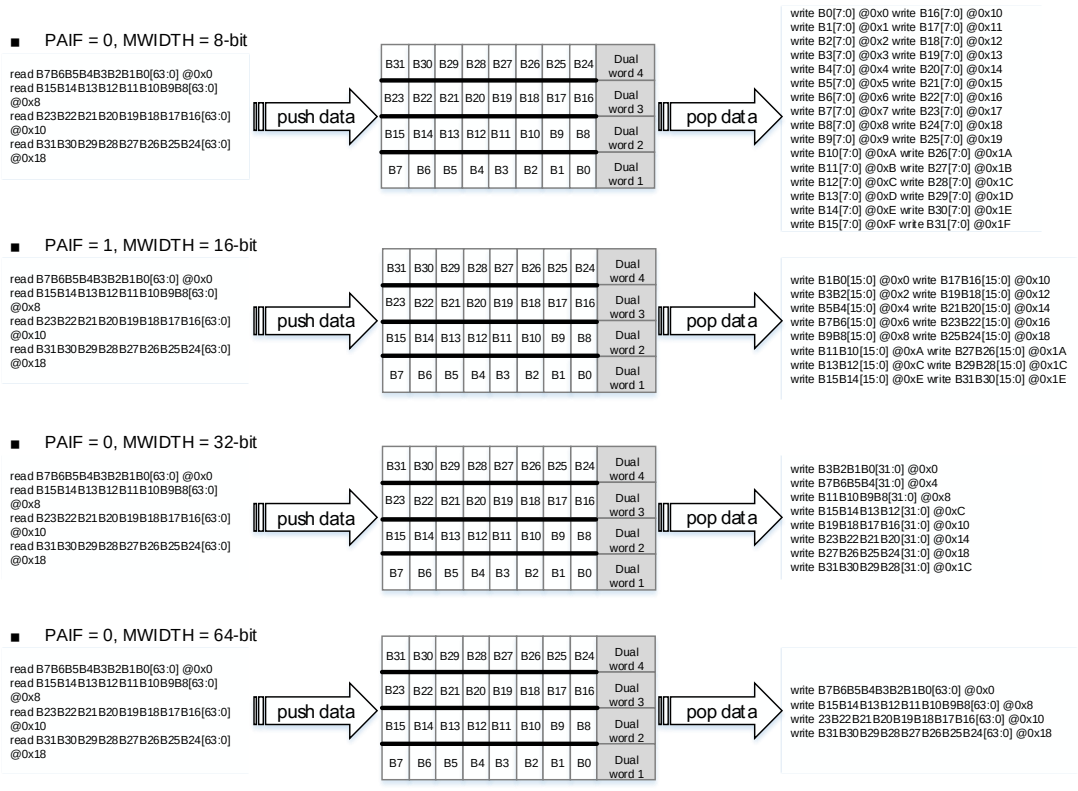
Suppose CNT is 8, the PWIDTH bits are equal to '10', and both PNAGA and MNAGA are set. The DMA transfer operations for different MWIDTH are shown in the ***Figure 10-6. Data packing/unpacking when PWIDTH = '10'***.

Figure 10-6. Data packing/unpacking when PWIDTH = '10'



Suppose CNT is 4, the PWIDTH bits are equal to '11', and both PNAGA and MNAGA are set. The DMA transfer operations for different MWIDTH are shown in the ***Figure 10-7. Data packing/unpacking when PWIDTH = '11'***.

Figure 10-7. Data packing/unpacking when PWIDTH = '11'



10.3.4. Address generation

Two kinds of address generation algorithm are implemented independently for memory and peripheral, including the fixed mode and the increased mode. The PNAGA and MNAGA bit in the DMA_CHxCTL register are used to configure the next address generation algorithm of peripheral and memory.

In the fixed mode, the next address is always equal to the base address configured in the base address registers (DMA_CHxPADDR, DMA_CHxM0ADDR, and DMA_CHxM1ADDR).

In the increasing mode, the next address is equal to the current address plus 1 or 2, 4 or 8 bytes, depending on the transfer data width. In Multi-data mode with PBURST in the DMA_CHxCTL register is '00', if PAIF in the DMA_CHxCTL register is enabled, the next peripheral address increment is fixed to 8 bytes, and has nothing to do with the peripheral transfer data width. The PAIF has no meaning to the memory address generation.

Note: If PAIF in the DMA_CHxCTL register is enable, the peripheral base address configured in the DMA_CHxPADDR register must be 64-bit alignment.

10.3.5. Circular mode

Circular mode is implemented to handle continue peripheral requests. The CMEN bit in DMA_CHxCTL register is used to enable/disable the circular mode. Circular mode is available only when DMA controls the transfer flow.

In circular mode, the CNT bits are automatically reloaded with the pre-programmed value and the full transfer finish flag is asserted at the end of every DMA transfer. DMA can always respond the peripheral request until a transfer error is detected or the CHEN bit in the DMA_CHxCTL register is cleared.

When the CMEN in the DMA_CHxCTL register and the CCNTEN bit in the DMA_CHxCNT register are enabled , the circular counts can be configured by CCNT[7:0] bits field.

The memory and peripheral address circular mode can be enable / disable by configuring the MECN bit and PCEN bit in DMA_CHxCTL register.

10.3.6. Switch-buffer mode

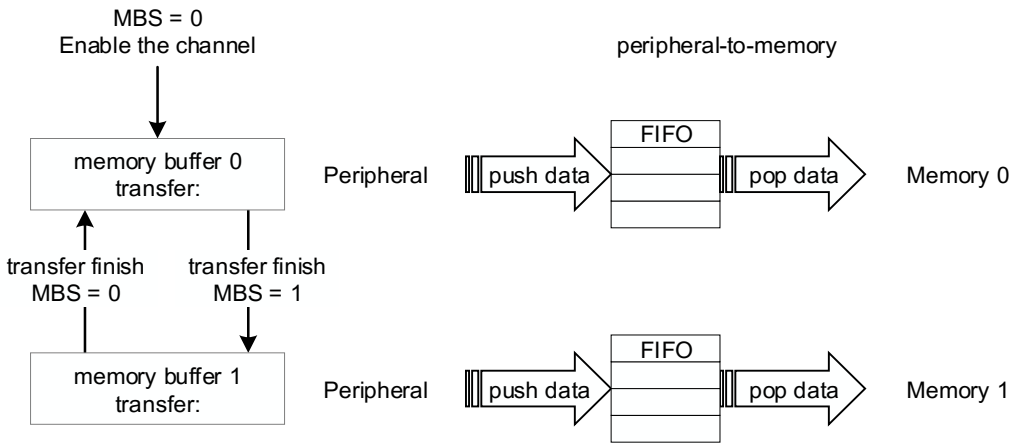
Similar to circular mode, switch-buffer mode is also implemented to handle continues peripheral requests. The SBMEN bit in the DMA_CHxCTL register is used to enable/disable the switch-buffer mode. When the switch-buffer mode is enabled, the circular mode is automatically enabled immediately after the channel is enabled. Switch-buffer mode is only available when the transfer mode is peripheral-to-memory or memory-to-peripheral. When the transfer mode is memory-to-memory, the switch-buffer mode is automatically disabled immediately after the channel is enabled.

Switch-buffer mode is supported with two memory buffers and the base address of the two memory buffers are separately configured in the DMA_CHxM0ADDR and

DMA_CHxM1ADDR register. In switch-buffer mode, the DMA memory pointer switches from the current memory buffer to another at the end of every DMA transfer. During the DMA transmission, the memory buffer not being processed by DMA can be accessed by other AHB masters. In switch-buffer mode, the base address of the memory buffer not accessed by DMA can be updated even if the channel is enabled.

The MBS bit in the DMA_CHxCTL register is configured to select which memory buffer is accessed by DMA at the first DMA transfer before the channel is enabled. In switch-buffer mode, this bit switches automatically between '0' and '1' at the end of every DMA transfer, and can be used as a flag indicating the current memory buffer accessed by DMA during the transmission. The DMA operation of switch-buffer mode are shown in ***Figure 10-8. DMA operation of switch-buffer mode***.

Figure 10-8. DMA operation of switch-buffer mode



10.3.7. Transfer flow controller

The transfer flow controller controls the number of data items to be transferred. The TFCS bit in the DMA_CHxCTL register determines which of DMA and peripheral is selected to control the transfer flow.

- DMA as transfer flow controller: The CNT bits in the DMA_CHxCNT register determine the number of data items to be transferred. The CNT bits must be configured before the channel is enabled.
- Peripheral as transfer flow controller: The CNT bits configured in the DMA_CHxCNT register before the channel is enabled have no meaning and these bits are force to '0xFFFF' immediately after the channel is enabled. The peripheral determines when to finish the DMA transfer by informing a last request signal to DMA.

Note: When the transfer mode is memory-to-memory, the transfer flow controller is fixed to be DMA whatever the TFCS bit is configured to.

10.3.8. Transfer operation

Three transfer modes are supported to implement the data transfer, including peripheral-to-memory, memory-to-peripheral and memory-to-memory. Memory and peripheral can be configured as source and destination relatively.

Memory transfer

- Peripheral-to-memory mode:
  - In single-data mode, when the FIFO is not empty, DMA initiates a single memory transfer and writes data into the corresponding memory address.
  - In multi-data mode, when the FIFO counter reaches the critical value, DMA starts single or burst memory transfers to entirely fetch the FIFO data and write to the memory.
- Memory-to-peripheral mode:
  - In single-data mode, when the channel is enabled, DMA starts a single memory transfer and pushes the reading data into the FIFO immediately. During the transmission, the memory transfer is initiated only when the FIFO is empty.
  - In multi-data mode, when the channel is enabled, DMA starts several single or burst transfers to fill up the FIFO whether the peripheral request is asserted or not. During the transmission, the memory transfer is initiated once when there is enough space for it in the FIFO.
- Memory-to-memory mode: When the FIFO counter reaches the critical value, DMA starts single or burst memory transfers to entirely fetch the FIFO data and write to the memory.

Peripheral transfer

- Peripheral-to-memory mode: When receiving a peripheral request and there is enough space in the FIFO for a peripheral transfer, DMA starts a peripheral transfer and pushes the reading data into the FIFO.
- Memory-to-peripheral mode: When receiving a peripheral request and there is enough data in the FIFO for a peripheral transfer, DMA starts a peripheral transfer to fetch the FIFO data and write to the peripheral.

10.3.9. Transfer finish

The DMA transfer is finished automatically and the FTFIFx bit in the DMA_INTF0, DMA_INTF1, DMA_INTF2 or DMA_INTF3 register is set when one of the following situations occurs:

- Transfer completion.
- Software clear.
- Error detection.

Transfer completion

When enabled, the DMA begins to transfer data between peripheral and memory. After the

pre-programmed number of data items has been transferred successfully, the DMA transfer is completed and the CHEN bit is automatically cleared in the DMA_CHxCTL register.

- Peripheral-to-memory mode: If DMA is the transfer flow controller, when the CNT bits reach to zero and the contents of the FIFO have been entirely transferred into the memory, an end of transfer is generated. If peripheral is the transfer flow controller, the DMA transfer is completed when the last peripheral request has been responded and the contents of the FIFO have been entirely transferred into the memory.
- Memory-to-peripheral mode: If DMA is the transfer flow controller, when the CNT bits reach to zero, an end of transfer is achieved. If peripheral is the transfer flow controller, the DMA transfer is completed when the last peripheral request has been responded.
- Memory-to-memory: only DMA can be the transfer flow controller. When the CNT bits reach to zero and the contents of the FIFO have been entirely transferred into the memory, an end of transfer is generated.

Software clear

The DMA transfer can be stopped by clearing the CHEN bit in the DMA_CHxCTL register by software. After the software cleared operation, the CHEN bit is still read as 1 to indicate that there are memory or peripheral transfers still active or the remaining data in the FIFO need to be transferred.

- Peripheral-to-memory: After the software cleared operation, the peripheral transfer is stopped when the current single or burst transfer is completed. To ensure that the data had been read from peripheral can be entirely transferred into the memory, the memory transfer continues to be active until the FIFO is empty. If the remaining byte number in the FIFO is not enough for a burst memory transfer, these data items are transferred in single transaction. If the remaining byte number is less than the memory transfer width, these data items are still written in memory transfer width with MSBs filled with zero. The software can read the CNT bits to calculate the number of valid data items in the memory. After the contents of the FIFO has been entirely transferred into the memory, the CHEN bit is cleared automatically by hardware and the FTFIFx bit in the DMA_INTF0, DMA_INTF1, DMA_INTF2 or DMA_INTF3 register is set.
- Memory-to-peripheral: After the software cleared operation, the DMA transfer is stopped when the current memory and peripheral transfer are completed. Then the CHEN bit is cleared automatically by hardware and the FTFIFx bit in the DMA_INTF0, DMA_INTF1, DMA_INTF2 or DMA_INTF3 register is set.
- Memory-to-memory: The same as the peripheral-to-memory mode with the source memory transfer is implemented through the peripheral port.

Error detection

Three types error can disable the DMA transfer:

- FIFO error: When a wrong FIFO configuration is detected, the DMA channel is disabled immediately without starting any transfers. In this situation, the FTFIFx is not asserted. For more information about the FIFO error, refer to section ***Error***.

- Bus error: When the memory or peripheral port attempts to access an address beyond the access scope, a bus error is detected and the DMA transfer is stopped immediately without setting the FTFIFx. If this error is aroused by the peripheral port, the CNT bits are still decreased by 1. For more information about the bus error, refer to section ***Error***.
- Register access error: In switch-buffer mode, an access error is detected when a write command is active on the memory base address register which is being accessed by DMA. When this error occurs, the DMA operation is the same as it after the CHEN bit software cleared. For more information about the register access error, refer to section ***Error***.

10.3.10. Channel configuration

When starting a new DMA transfer, it is recommended to respect the following steps:

1. Read the CHEN bit and judge whether the channel is enabled or not. If the channel is enabled, clear the CHEN bit by software or wait the current DMA transfer finished. When the CHEN bit is read as '0', configuring and starting a new DMA transfer is allowed.
2. Clear the FTFIFx bit in the DMA_INTF0, DMA_INTF1, DMA_INTF2 or DMA_INTF3 register, or a new DMA transfer can not be re-enabled.
3. Configure the TM bits in the DMA_CHxCTL register to set the transfer mode.
4. Configure the memory and peripheral burst types, the target memory buffer, switch-buffer mode, priority of the channel, memory and peripheral transfer width, memory and peripheral address generation algorithm, circular mode in the DMA_CHxCTL register, etc.
5. Configure multi-data mode, and the FCCV bits to set the FIFO counter critical value if multi-data mode is enabled in the DMA_CHxFCTL register.
6. Configure the enable bit for full transfer finish interrupt, half transfer finish interrupt, transfer access error interrupt, single-data mode exception interrupt in the DMA_CHxCTL register and the enable bit for FIFO error and exception interrupt in the DMA_CHxFCTL register.
7. Configure the DMA_CHxPADDR register for setting the peripheral base address.
8. If the switch-buffer mode is enabled, configure the memory base address in DMA_CHxM0ADDR and DMA_CHxM1ADDR register . If only one memory buffer is to be used, configure the DMA_CHxM0ADDR or DMA_CHxM1ADDR corresponding with the MBS bit in the DMA_CHxCTL register.
9. Configure the DMA_CHxCNT register to set the total transfer data number. If circular count mode is enabled, configure CCNT.
10. Configure the CHEN bit with '1' in the DMA_CHxCTL register to enable the channel.

When restarting the suspended DMA transfer, it is recommended to respect the following steps:

1. Read the CHEN bit and ensure the DMA suspend operation has been completed. When the CHEN bit is read as '0', DMA is idle, restarting the DMA transfer is allowed.
2. Clear the FTFIFx bit in the DMA_INTF0, DMA_INTF1, DMA_INTF2 or DMA_INTF3 register, or the DMA transfer cannot be re-enabled.
3. Read the DMA_CHxCNT register to obtain the number of the remaining data items and calculate the number of the data items had already been transferred.
4. Configure the DMA_CHxPADDR register to update the peripheral address pointer.
5. Configure the DMA_CHxM0ADDR or the DMA_CHxM1ADDR register to update the memory address pointer.
6. Configure the DMA_CHxCNT with the number of the remaining data items.
7. Configure the CHEN bit with '1' in the DMA_CHxCTL register to restart the channel.

10.4. Interrupts

Each DMA channel has a dedicated interrupt. There are five interrupt events connected to each interrupt, including full transfer finish interrupt, half transfer finish interrupt, transfer access error interrupt, single-data mode exception interrupt, and FIFO error and exception interrupt. A DMA channel interrupt may be produced when any interrupt event occurs on the channel.

Each interrupt event has a dedicated flag bit in the DMA_INTF0, DMA_INTF1, DMA_INTF2 or DMA_INTF3 register, a dedicated clear bit in the DMA_INTC0, DMA_INTC1, DMA_INTC2 and DMA_INTC3 register, and a dedicated enable bit in the DMA_CHxCTL and CHxFCTL register, as described in the *Table 10-5. DMA interrupt events*..

Table 10-5. DMA interrupt events

Interrupt event	Flag bit	Enable bit	Clear bit
	DMA_INTFx	DMA_CHxCTL or DMA_CHxFCTL	DMA_INTCx
Full transfer finish	FTFIF	FTFIE	FTFIFC
Half transfer finish	HTFIF	HTFIE	HTFIFC
Transfer access error	TAEIF	TAEIE	TAEIFC
Single-data mode exception	SDEIF	SDEIE	SDEIFC
FIFO error and exception	FEEIF	FEEIE	FEEIFC

These five events can be divided into three types:

- Flag: Full transfer finish flag and half transfer finish flag.
- Exception: Single-data mode exception and FIFO exception.
- Error: Transfer access error and FIFO error.

When the exception events occur, the DMA transmission is not affected and continues transferring normally. When the error events are detected, the DMA transmission is stopped. These three types of event are described in detail in the following sections.

10.4.1. Flag

Two flag events are supported, including full transfer finish flag and half transfer finish flag.

The full transfer finish flag is asserted, when one of the following situations occurs:

- The CNT bits reach to zero when DMA is the transfer flow controller.
- When peripheral is the transfer flow controller, the last request is responded completely and the contents of the FIFO are entirely written into the memory in peripheral-to-memory mode.
- When the channel is disabled by software before the end of the transfer, the current memory and peripheral is completed and the contents of the FIFO are entirely written into the memory in peripheral-to-memory or memory-to-memory mode.
- When the channel is disabled because of register access error before the end of the transfer, the current memory and peripheral is completed and the contents of the FIFO are entirely written into the memory in peripheral-to-memory or memory-to-memory mode.

When the full transfer finish flag is asserted and the enabled bit for the full transfer finish interrupt is set, an interrupt is generated.

The half transfer finish flag is asserted, only when DMA is the transfer flow controller and half of the CNT bits are transferred. If peripheral is the transfer flow controller, DMA does not know when half of data items has been transferred and the half transfer finish flag will stay zero.

When the half transfer finish flag is asserted and the enabled bit for the half transfer finish interrupt is set, an interrupt is generated.

10.4.2. Exception

Two exception events are supported, including single-data mode exception and FIFO exception. These exceptions have no effect on the DMA transmission.

Single-data mode exception

This exception can be detected only when the single-data mode is enabled and the transfer mode is peripheral-to-memory. When a peripheral request is valid and the FIFO is not empty, there are two or more data items stored in the FIFO after responding the peripheral request, which could be a problem for the subsequent processing of the data and the single-data mode exception bit SDEIFx will be set.

When the single-data mode exception is asserted and the enabled bit for the single-data mode exception interrupt is set, an interrupt is generated.

FIFO exception

When a FIFO underrun or a FIFO overrun condition occurs, the FIFO exception is asserted. This exception can be detected only when the transmission is between peripheral and

memory.

In peripheral-to-memory mode, when a peripheral request is valid and there is not enough space in the FIFO for the single or burst peripheral transfer, a FIFO overrun condition is detected. This peripheral request is not responded until the FIFO space is enough, and the accuracy of the data transmission will not be destroyed.

In memory-to-peripheral mode, when a peripheral request is valid and there is not enough data in the FIFO for the single or burst peripheral, a FIFO underrun condition is detected. This peripheral request is not responded until the data number in the FIFO is enough, and the accuracy of the data transmission will not be destroyed.

Note: In memory-to-peripheral mode, The DMA channel must be enabled before the peripheral. If the peripheral is enabled first, it will immediately generate a DMA transfer request. At that point the DMA channel is not yet enabled and the FIFO contains no valid data, which triggers the FIFO error flag.

Note: In memory-to-peripheral mode, During SPI communication, the peripheral generates multiple DMA transfer requests in rapid succession, which may falsely trigger the FIFO error flag. It is recommended that software ignore this flag and not enable the FIFO error interrupt.

When the FIFO exception is asserted and the enabled bit for the FIFO error and exception interrupt is set, an interrupt is generated.

10.4.3. Error

FIFO error and transfer access error (including the register access error and bus error) can be detected during the DMA transmission, and the transmission can be stopped when one of the errors occurs.

FIFO error

For a DMA operation, when the multi-data mode is enabled, the right and wrong configurations of the FIFO counter critical value corresponding with the memory transfer width and memory burst types are listed in ***Table 10-3. FIFO counter critical value configuration rules for channel 3~7*** and ***Table 10-4. FIFO counter critical value configuration rules for channel 0~2.***

If a wrong configuration is detected after enable the channel, a FIFO error is generated and the channel is disabled immediately without starting any transfers.

When the FIFO error is asserted and the enabled bit for the FIFO error and exception interrupt is set, an interrupt is generated.

Register access error

The register access error is detected only when the switch-buffer is enabled. If the software attempts to update a memory address register currently accessed by the DMA controller, a register access error is detected. For example, when the memory 0 buffer is the current

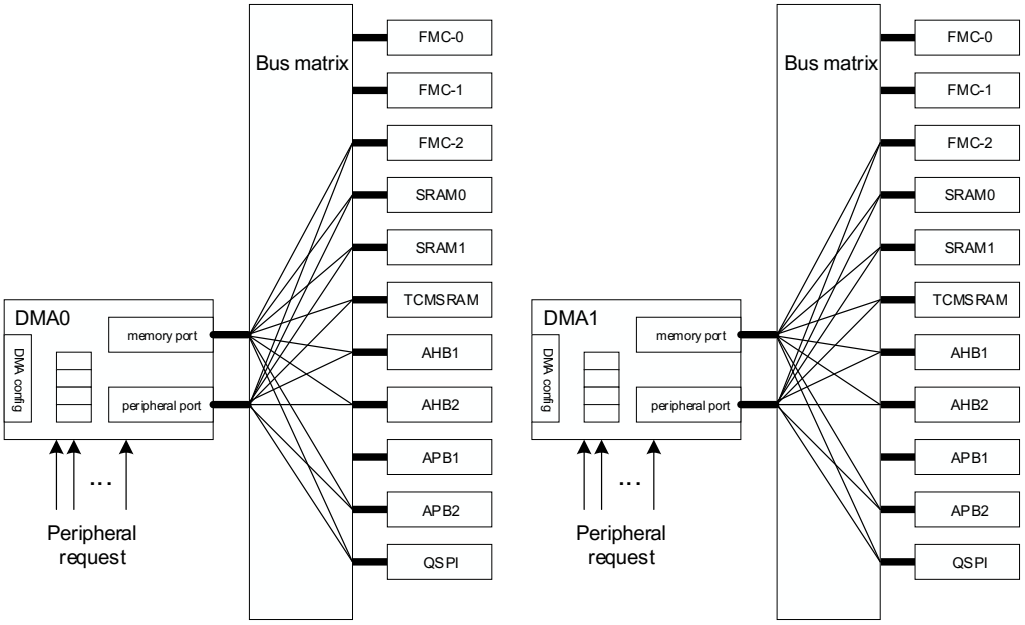
source or destination, a write access on the DMA_CHxM0ADDR register could produce a register access error. When a register access error occurs, the DMA transmission is stopped when the current memory and peripheral transfer are completed and the valid FIFO data are entirely drained into the memory if needed.

When the register access error is asserted and the enabled bit for the transfer access error and exception interrupt is set, an interrupt is generated.

Bus error

When the address accessed by the DMA controller is beyond the allowed area, a response error will be received and the channel is disabled immediately. The allowed and forbidden access region for DMA0 and DMA1 are shown in ***Figure 10-9. System connection of DMA0 and DMA1***. When the bus error is asserted and the enabled bit for the transfer access error and exception interrupt is set, an interrupt is generated.

Figure 10-9. System connection of DMA0 and DMA1



10.4.4. DMA request mapping

The DMA requests of a channel are coming from the AHB/APB peripherals through the corresponding channel output of DMAMUX request multiplexer, refers to ***Table 11-2. Request multiplexer input mapping***.

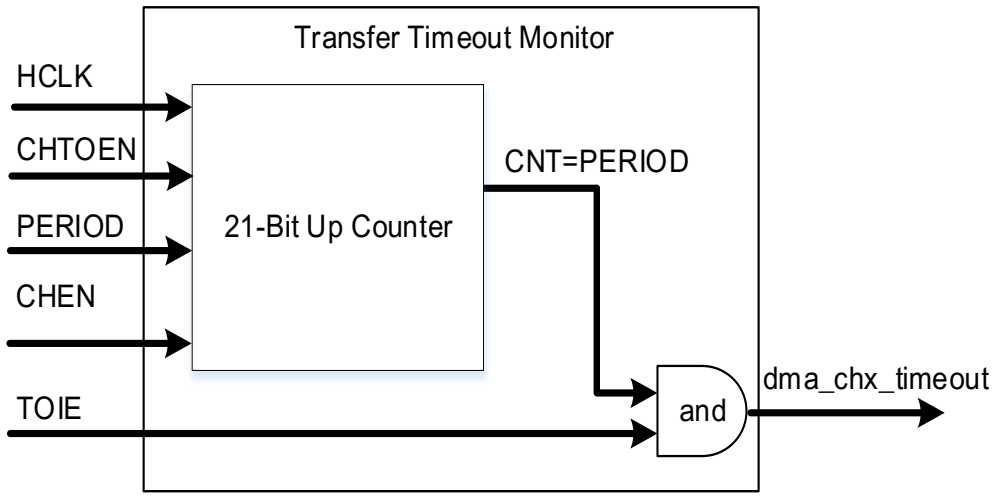
10.5. Fault interrupt management

10.5.1. Channel x transfer timeout monitor

The transfer timer is used to detect timeout for the channel started by setting the CHEN bit in DMA_CHxCTL register to the channel transfer finish.

There is a built-in counter for each channel transfer timer, when the channel is associated, the counter counts up if the CHTOEN bit in the DMA_CHxTOCTL is set until the transfer finish flag occurred. The timeout threshold value is determined by the PERIOD[20:0] field in the DMA_CHxTOCTL register that can be configured by software. If the counter reaches the PERIOD value before the transfer finish, the timeout flag will be generated. If the transfer timeout interrupt is enabled (set the TOIEx bit in the DMA_TOINTEN), the timeout interrupt will be generated to FMU. Refer to ***Figure 10-10. DMA channel x transfer timeout monitor interrupt logic.***

Figure 10-10. DMA channel x transfer timeout monitor interrupt logic



Calculate the channel x transfer timeout by using the formula below.

$$t_{TO}=t_{hclk}\times PERIOD[20:0]\ (ms) \quad (11\text{-}1)$$

where:

t_{TO}: channel x transfer timeout

t_{hclk}: AHB clock period measured in ms

10.5.2. Channel x control register enhanced

DMA support enhanced registers for control registers by TMR. If the DMA_CHxCTL register occurs a fault, the corresponding fault flag bit in the TFIFx bit in DMA_TFINTF register will be

set. A fault interrupt will be generated to FMU if TFIEx bit is enabled in the DMA_TFINTEN register.

10.6. CRC engine

10.6.1. Description

The DMA provides a built-in CRC engine on peripheral port for the safety-related data transferred between the Peripheral and Memory.

The CRC calculation unit is a parallel CRC 32-bit engine which implements the IEEE 802.3 polynomial: P(x) = x32 + x26 + x23 + x22 + x16 + x12 + x11 + x10 + x8 + x7 + x5 + x4 + x2 + x + 1. The DMA engine computes a CRC value over the data on the peripheral port. This CRC is stored in the register DMA_CHxPCRCDATA.

The verification of the CRC is done by the application. If the DMA CRC mechanism for data peripheral port is used, it shall be tested by the system integrator. This can be done by computing a CRC over the same data on the memory port and compared with the result of the DMA CRC mechanism.

10.6.2. Function

CRC calculation unit is used to calculate the 64-bit data on the peripheral port, and DMA_CHxPCRCDATA register will store the calculation result.

The CRC calculator can be initialized to a programmable value using the RTSx bit in the DMA_PCRCCTL register (the default value is 0xFFFFFFFF).

The initial CRC value can be programmed with the DMA_CRCIDATA register. The DMA_CHxPCRCDATA register is automatically initialized upon DMA_CRCIDATA register write access.

10.7. Register definition

DMA0 base address: 0x4900 0000

DMA1 base address: 0x4900 0400

Table 10-6. DMA register overview

Register name	Symbol	Address offset	Reset value
Interrupt flag register 0	DMA_INTF0	0x00	0x0000 0000
Interrupt flag register 1	DMA_INTF1	0x04	0x0000 0000
Interrupt flag clear register 0	DMA_INTC0	0x10	0x0000 0000
Interrupt flag clear register 1	DMA_INTC1	0x14	0x0000 0000
Channel x control register	DMA_CHxCTL	0x20	0x0600 0000
Channel x counter register	DMA_CHxCNT	0x24	0x0000 0000
Channel x peripheral base address register	DMA_CHxPADDR	0x28	0x0000 0000
Channel x memory 0 base address register	DMA_CHxM0ADDR	0x2C	0x0000 0000
Channel x memory 1 base address register	DMA_CHxM1ADDR	0x30	0x0000 0000
Channel x FIFO control register	DMA_CHxFCTL	0x34	0x0000 0021
Channel x peripheral CRC data register	DMA_CHxPCRCDATA	0x200	0xFFFF FFFF
CRC initialization data register	DMA_CRCIDATA	0x240	0xFFFF FFFF
Peripheral port CRC control register	DMA_PCRCCTL	0x280	0x0000 0000
Global control register	DMA_GCTL	0x2C0	0x0000 0000
Channel x transfer timeout monitor control register	DMA_CHxTOCTL	0x300	0x0000 0000
Timeout interrupt flag register	DMA_TOINTF	0x340	0x0000 0000
Timeout interrupt enable register	DMA_TOINTEN	0x344	0x0000 0000
TMR fault interrupt flag register	DMA_TFINTF	0x348	0x0000 0000
TMR fault interrupt enable register	DMA_TFINTEN	0x34C	0x0000 FFFF
Channel x peripheral port current address register	DMA_CHxPCADDR	0x350	0x0000 0000
Channel x memory port current address register	DMA_CHxMCADDR	0x390	0x0000 0000

10.7.1. Interrupt flag register 0 (DMA_INTF0)

Address offset: 0x00
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved				FTFIF3	HTFIF3	TAEIF3	SDEIF3	Reserved	FEEIF3	FTFIF2	HTFIF2	TAEIF2	SDEIF2	Reserved	FEEIF2
				r	r	r	r		r	r	r	r	r		r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				FTFIF1	HTFIF1	TAEIF1	SDEIF1	Reserved	FEEIF1	FTFIF0	HTFIF0	TAEIF0	SDEIF0	Reserved	FEEIF0
				r	r	r	r		r	r	r	r	r		r

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27/21/11/5	FTFIFx	Full Transfer finish flag of channel x (x=0…3) Hardware set and software cleared by writing 1 to the corresponding bit in DMA_INTC0 register. 0: Transfer has not finished on channel x 1: Transfer has finished on channel x
26/20/10/4	HTFIFx	Half transfer finish flag of channel x (x=0…3) Hardware set and software cleared by writing 1 to the corresponding bit in DMA_INTC0 register. 0: Half number of transfer has not finished on channel x 1: Half number of transfer has finished on channel x
25/19/9/3	TAEIFx	Transfer access error flag of channel x (x=0…3) Hardware set and software cleared by writing 1 to the corresponding bit in DMA_INTC0 register. 0: Transfer access error has not occurred on channel x 1: Transfer access error has occurred on channel x
24/18/8/2	SDEIFx	Single data mode exception of channel x (x=0…3) Hardware set and software cleared by writing 1 to the corresponding bit in DMA_INTC0 register. 0: Single data mode exception has not occurred on channel x 1: Single data mode exception has occurred on channel x
23/17/7/1	Reserved	Must be kept at reset value.
22/16/6/0	FEEIFx	FIFO error and exception of channel x (x=0…3) Hardware set and software cleared by writing 1 to the corresponding bit in DMA_INTC0 register. 0: FIFO error or exception has not occurred on channel x 1: FIFO error or exception has occurred on channel x

10.7.2. Interrupt flag register 1 (DMA_INTF1)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				FTFIF7	HTFIF7	TAEIF7	SDEIF7	Reserved	FEEIF7	FTFIF6	HTFIF6	TAEIF6	SDEIF6	Reserved	FEEIF6
				r	r	r	r		r	r	r	r	r		r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				FTFIF5	HTFIF5	TAEIF5	SDEIF5	Reserved	FEEIF5	FTFIF4	HTFIF4	TAEIF4	SDEIF4	Reserved	FEEIF4
				r	r	r	r		r	r	r	r	r		r

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27/21/11/5	FTFIFx	Full Transfer finish flag of channel x (x=4…7) Hardware set and software cleared by writing 1 to the corresponding bit in DMA_INTC1 register. 0: Transfer has not finished on channel x 1: Transfer has finished on channel x
26/20/10/4	HTFIFx	Half transfer finish flag of channel x (x=4…7) Hardware set and software cleared by writing 1 to the corresponding bit in DMA_INTC1 register. 0: Half number of transfer has not finished on channel x 1: Half number of transfer has finished on channel x
25/19/9/3	TAEIFx	Transfer access error flag of channel x (x=4…7) Hardware set and software cleared by writing 1 to the corresponding bit in DMA_INTC1 register. 0: Transfer access error has not occurred on channel x 1: Transfer access error has occurred on channel x
24/18/8/2	SDEIFx	Single data mode exception of channel x (x=4…7) Hardware set and software cleared by writing 1 to the corresponding bit in DMA_INTC1 register. 0: Single data mode exception has not occurred on channel x 1: Single data mode exception has occurred on channel x
23/17/7/1	Reserved	Must be kept at reset value.
22/16/6/0	FEEIFx	FIFO error and exception of channel x (x=4…7) Hardware set and software cleared by writing 1 to the corresponding bit in DMA_INTC1 register. 0: FIFO error or exception has not occurred on channel x 1: FIFO error or exception has occurred on channel x

10.7.3. Interrupt flag clear register 0 (DMA_INTC0)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				FTFIFC3	HTFIFC3	TAEIFC3	SDEIFC3	Reserved	FEEIFC3	FTFIFC2	HTFIFC2	TAEIFC2	SDEIFC2	Reserved	FEEIFC2
				w	w	w	w		w	w	w	w	w		w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				FTFIFC1	HTFIFC1	TAEIFC1	SDEIFC1	Reserved	FEEIFC1	FTFIFC0	HTFIFC0	TAEIFC0	SDEIFC0	Reserved	FEEIFC0
				w	w	w	w		w	w	w	w	w		w

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27/21/11/5	FTFIFCx	Clear bit for Full transfer finish flag of channel x (x=0…3) 0: No effect 1: Clear full transfer finish flag
26/20/10/4	HTFIFCx	Clear bit for half transfer finish flag of channel x (x=0…3) 0: No effect 1: Clear half transfer finish flag
25/19/9/3	TAEIFCx	Clear bit for transfer access error flag of channel x (x=0…3) 0: No effect 1: Clear transfer access error flag
24/18/8/2	SDEIFCx	Clear bit for single data mode exception of channel x (x=0…3) 0: No effect 1: Clear single data mode exception flag
23/17/7/1	Reserved	Must be kept at reset value.
22/16/6/0	FEEIFCx	Clear bit for FIFO error and exception of channel x (x=0…3) 0: No effect 1: Clear FIFO error and exception flag

10.7.4. Interrupt flag clear register 1 (DMA_INTC1)

Address offset: 0x14

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				FTFIFC7	HTFIFC7	TAEIFC7	SDEIFC7	Reserved	FEEIFC7	FTFIFC6	HTFIFC6	TAEIFC6	SDEIFC6	Reserved	FEEIFC6
				w	w	w	w		w	w	w	w	w		w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				FTFIFC5	HTFIFC5	TAEIFC5	SDEIFC5	Reserved	FEEIFC5	FTFIFC4	HTFIFC4	TAEIFC4	SDEIFC4	Reserved	FEEIFC4
				w	w	w	w		w	w	w	w	w		w

Bits	Fields	Descriptions

31:28	Reserved	Must be kept at reset value.
27/21/11/5	FTFIFCx	Clear bit for full transfer finish flag of channel x (x=4…7) 0: No effect 1: Clear full transfer finish flag
26/20/10/4	HTFIFCx	Clear bit for half transfer finish flag of channel x (x=4…7) 0: No effect 1: Clear half transfer finish flag
25/19/9/3	TAEIFCx	Clear bit for transfer access error flag of channel x (x=4…7) 0: No effect 1: Clear transfer access error flag
24/18/8/2	SDEIFCx	Clear bit for single data mode exception of channel x (x=4…7) 0: No effect 1: Clear single data mode exception flag
23/17/7/1	Reserved	Must be kept at reset value.
22/16/6/0	FEEIFCx	Clear bit for FIFO error and exception of channel x (x=4…7) 0: No effect 1: Clear FIFO error and exception flag

10.7.5. Channel x control register (DMA_CHxCTL)

x = 0...7, where x is a channel number

Address offset: 0x20 + 0x18 * x

Reset value: 0x0600 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					MCEN	PCEN	MBURST[1:0]		PBURST[1:0]		Reserved	MBS	SBMEN	PRIO[1:0]	
			rw	rw	rw	rw	rw		rw			rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAIF	MWIDTH[1:0]		PWIDTH[1:0]		MNAGA	PNAGA	CMEN	TM[1:0]		TFCS	FTFIE	HTFIE	TAEIE	SDEIE	CHEN
rw	rw		rw		rw	rw	rw	rw		rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:27	Reserved	Must be kept at reset value.
26	MCEN	Memory address circular enable Software set and clear. When this bit is enabled in the circular mode, the memory address is automatically reloaded with the initial base address programmed during the configuration phase. This bit makes no effect when the circular mode is disabled 0: Disable memory address circular in the circular mode. 1: Enable memory address circular in the circular mode

		This bit can not be written when CHEN is '1'. This bit is automatically locked as '1' by hardware immediately after enable CHEN if SBMEN is configured to '1'.
25	PCEN	Peripheral address circular enable Software set and clear. When this bit is enabled in the circular mode, the peripheral address is automatically reloaded with the initial base address programmed during the configuration phase. This bit makes no effect when the circular mode is disabled. 0: Disable eripheral address circular in the circular mode. 1: Enable peripheral address circular in the circular mode This bit can not be written when CHEN is '1'.
24:23	MBURST[1:0]	Transfer burst type of memory Software set and clear. 00: single burst 01: INCR4 (4-beat incrementing burst) 10: INCR8 (8-beat incrementing burst) 11: INCR16 (16-beat incrementing burst) These bits cannot be written when CHEN is '1'. These bits are automatically locked as '00' by hardware immediately after enable CHEN if MDMEN in the DMA_CHxFCTL register is configured to '0'.
22:21	PBURST[1:0]	Transfer burst type of peripheral Software set and clear 00: single burst 01: INCR4 (4-beat incrementing burst) 10: INCR8 (8-beat incrementing burst) 11: INCR16 (16-beat incrementing burst) These bits cannot be written when CHEN is '1'. These bits are automatically locked as '00' by hardware immediately after enable CHEN if MDMEN in the DMA_CHxFCTL register is configured to '0'.
20	Reserved	Must be kept at reset value.
19	MBS	Memory buffer select Hardware and software set, hardware and software clear. 0: Memory 0 is selected as memory transfer area 1: Memory 1 is selected as memory transfer area This bit cannot be written when CHEN is '1'. During the transmission, this bit can be set and cleared by hardware at the end of transfer to indicate which memory buffer is being accessed by DMA.
18	SBMEN	Switch-buffer mode enable Software set and clear. 0: Disable switch-buffer mode 1: Enable switch-buffer mode

		This bit cannot be written when CHEN is '1'.
17:16	PRIO[1:0]	Priority level Software set and clear. 00: Low 01: Medium 10: High 11: Ultra high These bits cannot be written when CHEN is '1'.
15	PAIF	Peripheral address increment fixed Software set and clear. 0: The peripheral address increment is determined by PWIDTH 1: The peripheral address increment is fixed to 8 This bit cannot be written when CHEN is '1'. During the transmission, when PNAGA is configured to '0', this bit has no effect. These bits are automatically locked as '0' by hardware immediately after enable CHEN if MDMEN in the DMA_CHxFCTL register is configured to '0' or PBURST are not equal to '00'.
14:13	MWIDTH[1:0]	Transfer width of memory Software set and clear. 00: 8-bit 01: 16-bit 10: 32-bit 11: 64-bit These bits cannot be written when CHEN is '1'. These bits are automatically locked as PWIDTH by hardware immediately after enable CHEN if MDMEN in the DMA_CHxFCTL register is configured to '0'.
12:11	PWIDTH[1:0]	Transfer width of peripheral Software set and clear. 00: 8-bit 01: 16-bit 10: 32-bit 11: 64-bit These bits cannot be written when CHEN is '1'.
10	MNAGA	Next address generation algorithm of memory Software set and clear 0: Fixed address mode 1: Increasing address mode This bit cannot be written when CHEN is '1'.
9	PNAGA	Next address generation algorithm of peripheral Software set and clear 0: Fixed address mode

		1: Increasing address mode This bit can not be written when CHEN is '1'.
8	CMEN	Circular mode enable Software set and clear. 0: Disable circular mode. 1: Enable circular mode This bit cannot be written when CHEN is '1'. This bit is automatically locked as '0' by hardware immediately after enable CHEN if TFCS is configured to '1'. This bit is automatically locked as '1' by hardware immediately after enable CHEN if SBMEN is configured to '1'.
7:6	TM[1:0]	Transfer mode Software set and clear. 00: Read from peripheral and write to memory 01: Read from memory and write to peripheral 10: Read from memory and write to memory 11: Reserved These bits cannot be written when CHEN is '1'.
5	TFCS	Transfer flow controller select Software set and clear. 0: DMA is selected as the transfer flow controller 1: Peripheral is selected as the transfer flow controller This bit cannot be written when CHEN is '1'.
4	FTFIE	Enable bit for full transfer finish interrupt Software set and clear. 0:Disable full transfer finish interrupt 1:Enable full transfer finish interrupt
3	HTFIE	Enable bit for half transfer finish interrupt Software set and clear. 0: Disable half transfer finish interrupt 1: Enable half transfer finish interrupt
2	TAEIE	Enable bit for transfer access error interrupt Software set and clear. 0: Disable transfer access error interrupt 1: Enable transfer access error interrupt
1	SDEIE	Enable bit for single data mode exception interrupt Software set and clear. 0: Disable single data mode exception interrupt 1: Enable single data mode exception interrupt
0	CHEN	Channel enable

Software set, hardware clear.

0: Disable channel

1: Enable channel

When this bit is asserted, the DMA transfer is started. This bit is automatically cleared when one of the following situations occurs:

When the transfer of channel is fully finished.

When a wrong FIFO configuration or a transfer access error is detected.

After a software clear operation, this bit is still read as 1 to indicate that there are memory or peripheral transfers still active until hardware has terminated all activity, at which point this bit is read as 0. Software can therefore poll this bit to determine when this channel is free for a new DMA transfer.

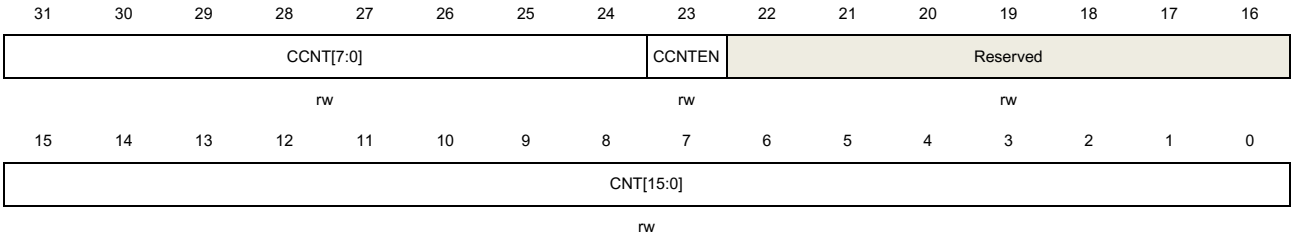
10.7.6. Channel x counter register (DMA_CHxCNT)

x = 0...7, where x is a channel number

Address offset: 0x24 + 0x18 * x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:24	CCNT[7:0]	Circular mode count When using circular mode counting, CCNT is decremented by 1 each time CNT is decremented to 0. These bits can not be written when CHEN in the DMA_CHxCTL register is '1'. These bits make no effect when the circular mode is disabled.
23	CCNTEN	Circular mode count enable 0: Disable circular mode count 1: Enable circular mode count This bit can not be written when CHEN in the DMA_CHxCTL register is '1'. This bit make no effect when the circular mode is disabled.
22:16	Reserved	Must be kept at reset value.
15:0	CNT[15:0]	Transfer counter These bits can not be written when CHEN in the DMA_CHxCTL register is '1'. These bits are related to PWIDTH. During the transmission, these bits signify the number of remaining data to be transferred. After each DMA peripheral transfer, CNT is decremented by 1. If CMEN or SBMEN in the DMA_CHxCTL register is

configured to '1', CNT can be reloaded automatically to the original value at the end of transfer.

10.7.7. Channel x peripheral base address register (DMA_CHxPADDR)

x = 0...7, where x is a channel number

Address offset: 0x28 + 0x18 * x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PADDR[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PADDR[15:0]															
rw															

Bits	Fields	Descriptions
31:0	PADDR[31:0]	Peripheral base address These bits cannot be written when CHEN in the DMA_CHxCTL register is '1'. When PWIDTH is 01 (16-bit), the LSB of these bits is ignored. Access is automatically aligned to a half word address. When PWIDTH is 10 (32-bit), the two LSBs of these bits are ignored. Access is automatically aligned to a word address. When PWIDTH is 11 (64-bit), the three LSBs of these bits are ignored. Access is automatically aligned to double word address. **Note:** If PAIF in the DMA_CHxCTL register is enable, these bits must be configured to 64-bit alignment.

10.7.8. Channel x memory 0 base address register (DMA_CHxM0ADDR)

x = 0...7, where x is a channel number

Address offset: 0x2C + 0x18 * x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
M0ADDR[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M0ADDR[15:0]															
rw															

Bits	Fields	Descriptions

31:0	M0ADDR[31:0]	Memory 0 base address When MBS in the DMA_CHxCTL register is read as to '0', these bits specific the memory base address accessed by DMA during the transmission. These bits cannot be written when CHEN in the DMA_CHxCTL register is '1' and MBS in the DMA_CHxCTL register is read as '0'. When memory 0 is selected as memory transfer area and MWIDTH in the DMA_CHxCTL register is 01 (16-bit), the LSB of these bits is ignored. Access is automatically aligned to a half word address. When memory 0 is selected as memory transfer area and MWIDTH in the DMA_CHxCTL register is 10 (32-bit), the two LSBs of these bits are ignored. Access is automatically aligned to a word address. When memory 0 is selected as memory transfer area and MWIDTH in the DMA_CHxCTL register is 11 (64-bit), the three LSBs of these bits are ignored. Access is automatically aligned to double word address.

10.7.9. Channel x memory 1 base address register (DMA_CHxM1ADDR)

x = 0...7, where x is a channel number

Address offset: 0x30 + 0x18 * x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
M1ADDR[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M1ADDR[15:0]															
rw															

Bits	Fields	Descriptions
31:0	M1ADDR[31:0]	Memory 1 base address When MBS in the DMA_CHxCTL register is read as to '1', these bits specific the memory base address accessed by DMA during the transmission. These bits cannot be written when CHEN in the DMA_CHxCTL register is '1' and MBS in the DMA_CHxCTL register is read as '1'. When memory 1 is selected as memory transfer area and MWIDTH in the DMA_CHxCTL register is 01 (16-bit), the LSB of these bits is ignored. Access is automatically aligned to a half word address. When memory 1 is selected as memory transfer area and MWIDTH in the DMA_CHxCTL register is 10 (32-bit), the two LSBs of these bits are ignored. Access is automatically aligned to a word address. When memory 1 is selected as memory transfer area and MWIDTH in the DMA_CHxCTL register is 11 (64-bit), the three LSBs of these bits are ignored. Access is automatically aligned to double word address.

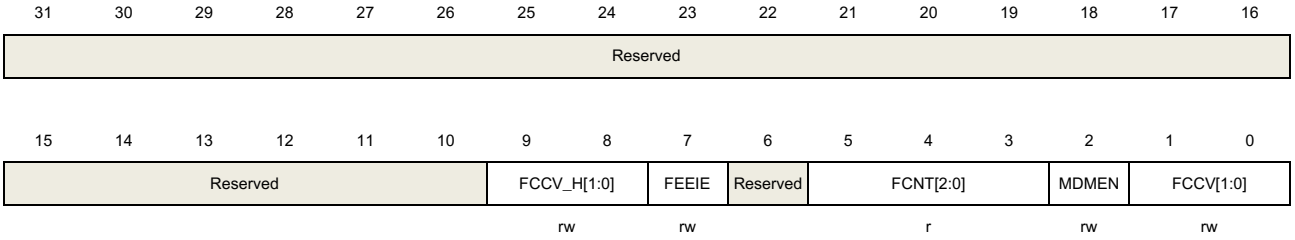
10.7.10. Channel x FIFO control register (DMA_CHxFCTL)

x = 0...7, where x is a channel number

Address offset: 0x34 + 0x18 * x

Reset value: 0x0000 0021

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:8	FCCV_H	FIFO counter critical value Software set and clear. These bits are used in combination with FCCV[1:0] For channel 0~2 {FCCV_H,FCCV}: 0000: 1*2 Words 0001: 2*2 Words 0010: 3*2 Words 0011: 4*2 Words 0100: 5*2 Words 0101: 6*2 Words 0110: 7*2 Words 0111: 8*2 Words 1000: 9*2 Words … 1110: 15*2 Words 1111: 16*2 Words These bits can not be written when CHEN in the DMA_CHxCTL register is '1'. When MDMEN is configured to '0', these bits has no meaning. These bits are reserved for channel 3~7.
7	FEEIE	Enable bit for FIFO error and exception interrupt Software set and clear. 0: Disable FIFO error and exception interrupt 1: Enable FIFO error and exception interrupt
6	Reserved	Must be kept at reset value.
5:3	FCNT[2:0]	FIFO counter Hardware set and clear. 000: Channel 0-2: 0 < FIFO data number < 8 words

Channel 3-7: No data
001: Channel 0-2: 8 ≤ FIFO data number < 16 words
Channel 3-7: 2 words
010: Channel 0-2: 16 ≤ FIFO data number < 24 words
Channel 3-7: 4 words
011: Channel 0-2: 24 ≤ FIFO data number < 32 words
Channel 3-7: 6 words
100: Empty
101: Full
110~111: Reserved
These bits specific the number of data stored in FIFO during the transmission. When MDMEN is configured to '0', these bits has no meaning.

2	MDMEN	Multi-data mode enable Software set and clear. 0: Disable Multi-data mode 1: Enable Multi-data mode These bits cannot be written when CHEN in the DMA_CHxCTL register is '1'. These bits are automatically locked as '1' by hardware immediately after enable CHEN in the DMA_CHxCTL register if TM in the DMA_CHxCTL register is configured to '10'.
1:0	FCCV[1:0]	FIFO counter critical value Software set and clear 00: Two word 01: Four Words 10: Six Words 11: Eight Words These bits can not be written when CHEN in the DMA_CHxCTL register is '1'. When MDMEN is configured to '0', these bits has no meaning.

10.7.11. Channel x peripheral CRC data register (DMA_CHxPCRCDATA)

x = 0...7, where x is a channel number
Address offset: 0x200 + 0x04 * x
Reset value: 0xFFFF FFFF

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRC_PDATA[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC_PDATA[15:0]															
rw															

Bits	Fields	Descriptions
31:0	CRC_PDATA[31:0]	Peripheral port CRC data This register is used to store the result of the working CRC-32 Ethernet polynomial checksum for read data. It holds the previous CRC calculation result when it is read. In order to start a CRC-32 sequence, this register must be initialized (e.g. write this register with a desired value or write 1 to RSTx to load the initial value programmed in the DMA_CRCIDATA register).

10.7.12. CRC initialization data register (DMA_CRCIDATA)

Address offset: 0x240
Reset value: 0xFFFF FFFF

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRC_IDATA[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC_IDATA[15:0]															
rw															

Bits	Fields	Descriptions
31:0	CRC_IDATA[31:0]	CRC initialization data value When RSTx bit in DMA_PCRCCTL is set, DMA_CHxPCRCDATA will be programmed to this value.

10.7.13. Peripheral port CRC control register (DMA_PCRCCTL)

Address offset: 0x280
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RST7	RST6	RST5	RST4	RST3	RST2	RST1	RST0
								w	w	w	w	w	w	w	w

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	RSTx	Reset bit Set this bit can reset the peripheral data CRC register of channel x (x=0…7).

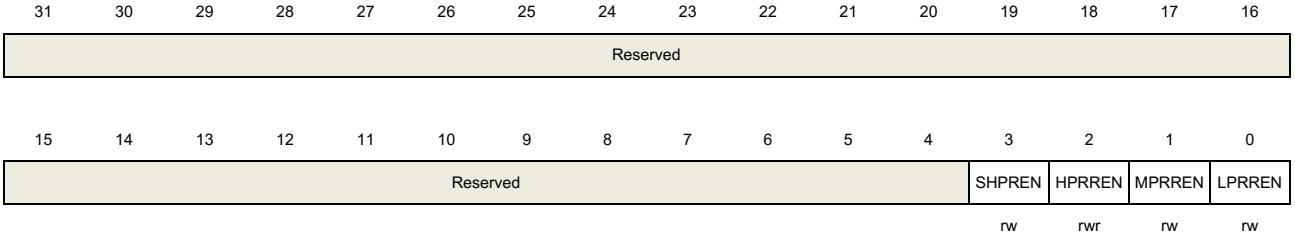
When set, the value of the DMA_CHxPCRCDATA register is automatically initialized to the value in the DMA_CRCIDATA register and then automatically cleared by hardware.

10.7.14. Global control register (DMA_GCTL)

Address offset: 0x2C0

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3	SHPREN	Enable round-robin for channels with ultra high software priority level Software set and clear. 0: Round-robin channel arbitration disabled. Fixed priority arbitration used for channel selection within ultra high software priority level 1: Round-robin channel arbitration enabled. Round-robin arbitration used for channel selection within ultra high software priority level This bit can not be written when CHEN in the DMA_CHxCTL register is '1'.
2	HPRREN	Enable round-robin for channels with high software priority level Software set and clear. 0: Round-robin channel arbitration disabled. Fixed priority arbitration used for channel selection within high software priority level 1: Round-robin channel arbitration enabled. Round-robin arbitration used for channel selection within high software priority level This bit can not be written when CHEN in the DMA_CHxCTL register is '1'.
1	MPRREN	Enable round-robin for channels with medium software priority level Software set and clear. 0: Round-robin channel arbitration disabled. Fixed priority arbitration used for channel selection within medium software priority level 1: Round-robin channel arbitration enabled. Round-robin arbitration used for channel selection within medium software priority level This bit can not be written when CHEN in the DMA_CHxCTL register is '1'.
0	LPRREN	Enable round-robin for channels with low software priority level Software set and clear.

0: Round-robin channel arbitration disabled. Fixed priority arbitration used for channel selection within low software priority level

1: Round-robin channel arbitration enabled. Round-robin arbitration used for channel selection within low software priority level

This bit can not be written when CHEN in the DMA_CHxCTL register is '1'.

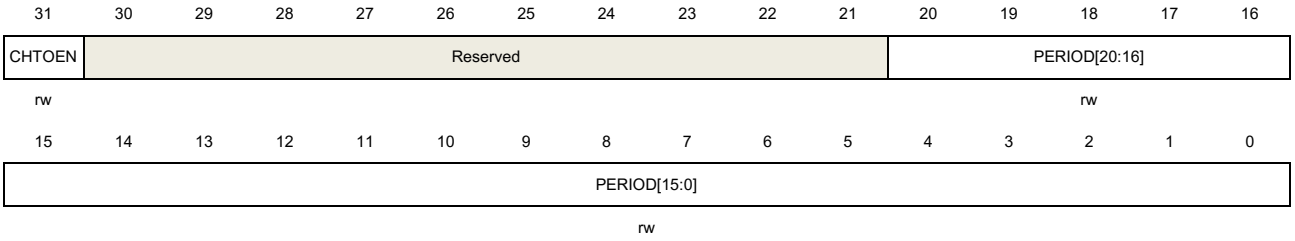
10.7.15. Channel x transfer timeout monitor control register (DMA_CHxTOCTL)

x = 0...7, where x is a channel number

Address offset: 0x300 + 0x04 * x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



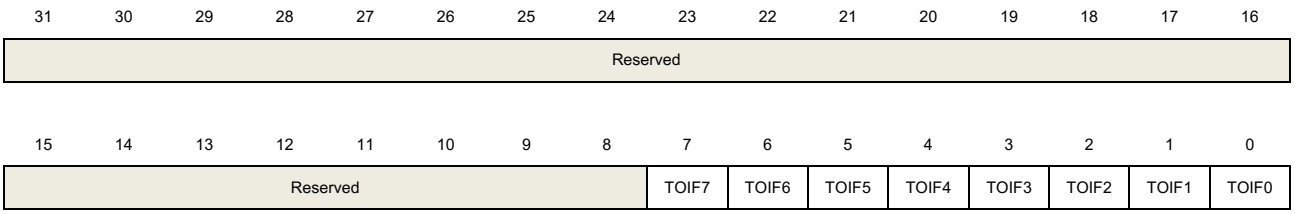
Bits	Fields	Descriptions
31	CHTOEN	Channel transfer timeout monitor enable This bit is set and cleared by software. 0: Transfer timeout monitor disabled 1: Transfer timeout monitor enabled This bit cannot be written when CHEN is '1'.
30:21	Reserved	Keep at reset value.
20:0	PERIOD[20:0]	The timeout threshold value. This bit is set and cleared by software. This field is used to define the timeout value of the transfer. A timeout interrupt occur when the value of the built-in counter is greater than this value. These bits cannot be written when CHTOEN is '1'.

10.7.16. Timeout interrupt flag register (DMA_TOINTF)

Address offset: 0x340

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



								rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions
31:8	Reserved	Keep at reset value.
7:0	TOIFx	Transfer timeout flag of channel x (x = 0…7) Hardware set and software can clear it by writing 1. 0: Channel x transfer timeout has not occurred 1: Channel x transfer timeout has occurred

10.7.17. Timeout interrupt enable register (DMA_TOINTEN)

Address offset: 0x344

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TOIE7	TOIE6	TOIE5	TOIE4	TOIE3	TOIE2	TOIE1	TOIE0
								rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:8	Reserved	Keep at reset value.
7:0	TOIEx	Transfer timeout interrupt of channel x enable (x = 0…7) These bits are set and cleared by software. 0: Channel x transfer timeout interrupt disabled 1: Channel x transfer timeout interrupt enabled

10.7.18. TMR fault interrupt flag register (DMA_TFINTF)

Address offset: 0x348

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TFIF7	TFIF6	TFIF5	TFIF4	TFIF3	TFIF2	TFIF1	TFIF0
								rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions

31:8	Reserved	Keep at reset value.
7:0	TFIFx	TMR fault flag of channel x control register (x=0…7) Hardware set and software can clear these fields by writing 1. 0: Channel x control register has not occurred a TMR fault 1: Channel x control register has occurred a TMR fault

10.7.19. TMR fault interrupt enable register (DMA_TFINTEN)

Address offset: 0x34C

Reset value: 0x0000 00FF

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TFIE7	TFIE6	TFIE5	TFIE4	TFIE3	TFIE2	TFIE1	TFIE0
								rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:8	Reserved	Keep at reset value.
7:0	TFIEx	TMR fault interrupt enable of channel x control register (x=0…7) These bits are set and cleared by software. 0: TMR fault interrupt of channel x control register disabled 1: TMR fault interrupt of channel x control register enabled

10.7.20. Channel x peripheral port current address register (DMA_CHxPCADDR)

x = 0...7, where x is a channel number

Address offset: 0x350 + 0x04 * x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PADDR[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PADDR[15:0]															
r															

Bits	Fields	Descriptions
31:0	PADDR[31:0]	Peripheral port current address pointing to the peripheral port data.

These bits cannot be written. During the transmission, these bits signify the updated peripheral port access address.

In address increment mode, the read address equals the current actual address plus 1, 2, 4, or 8, depending on the peripheral transfer data width. (If PAIF in the DMA_CHxCTL register is enabled, the read address equals the current actual address plus 8.)

10.7.21. Channel x memory port current address register (DMA_CHxMCADDR)

x = 0...7, where x is a channel number

Address offset: 0x390 + 0x04 * x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MADDR[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MADDR[15:0]															
r															

Bits	Fields	Descriptions
31:0	MADDR[31:0]	Memory port current address pointing to the memory port data. These bits cannot be written. During the transmission, these bits signify the updated actual address accessed by memory port. In address increment mode, the read address equals the current actual address plus 1, 2, 4, or 8, depending on the memory transfer data width.

11. DMA request multiplexer (DMAMUX)

11.1. Overview

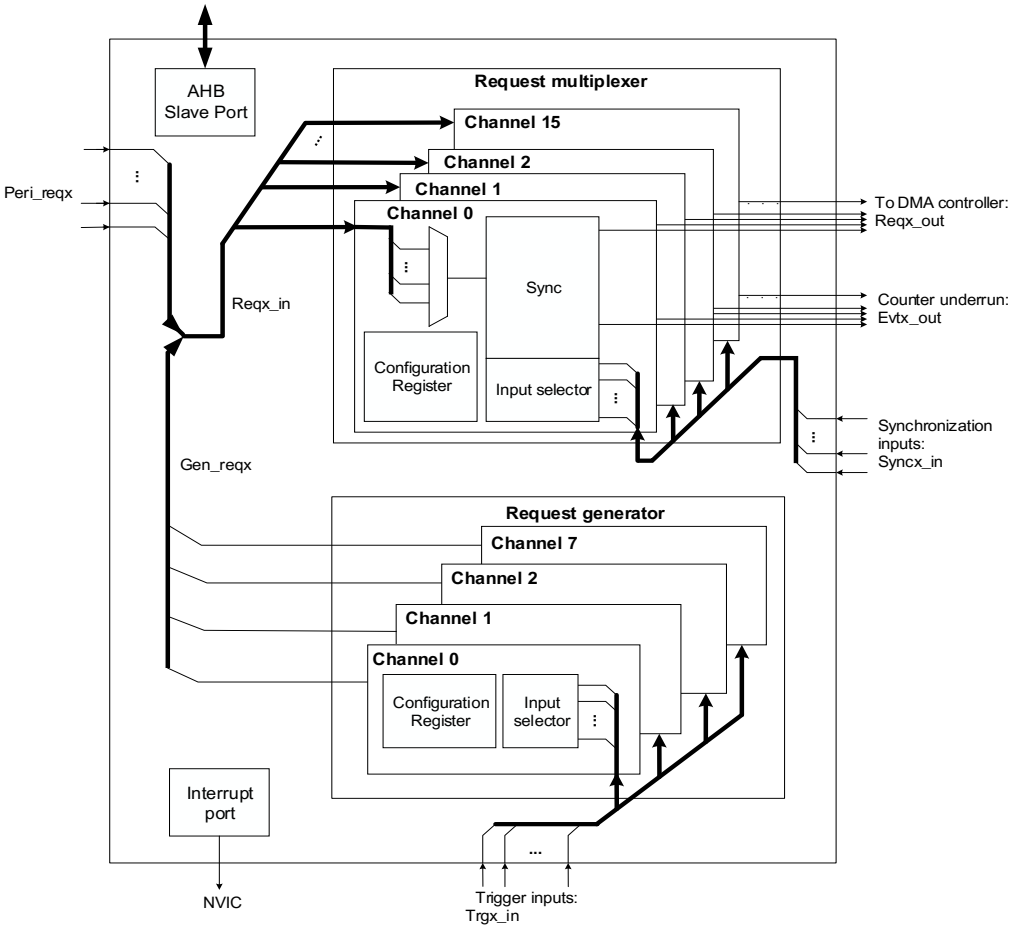
DMAMUX is a transmission scheduler for DMA requests. The DMAMUX request multiplexer is used for routing a DMA request line between the peripherals / generated DMA request (from the DMAMUX request generator) and the DMA controller. Each DMAMUX request multiplexer channel selects a unique DMA request line, unconditionally or synchronously with events from its DMAMUX synchronization inputs. The DMA request is pending until it is served by the DMA controller which generates a DMA acknowledge signal (the DMA request signal is de-asserted).

11.2. Characteristics

- 16 channels for DMAMUX request multiplexer.
- 8 channels for DMAMUX request generator.
- Support 25 trigger inputs.
- Support 25 synchronization inputs.
- Each DMAMUX request generator channel has a DMA request trigger input selector, a DMA request generator counter, and the trigger overrun flag.
- Each DMAMUX request multiplexer channel has 185 input DMA request lines from peripherals, a synchronization input selector, one DMA request line output, one channel event output for DMA request chaining, a DMA request multiplexer counter, and the synchronization overrun flag.

11.3. Block diagram

Figure 11-1. DMAMUX block diagram



11.4. Signal description

The DMAMUX signals are described as follows:

- Reqx_in: DMAMUX request multiplexer inputs from peripheral requests and request generator channels.
- Peri_reqx: DMAMUX DMA request line inputs from peripherals.
- Gen_reqx: DMAMUX generated DMA request from request generator.
- Reqx_out: DMAMUX requests outputs to DMA controller.
- Trgx_in: DMAMUX DMA request triggers inputs to request generator.
- Syncx_in: DMAMUX synchronization inputs to request multiplexer.
- Evtx_out: DMAMUX request multiplexer counter underrun event outputs.

11.5. Function overview

As shown in *Figure 11-1. DMAMUX block diagram*, DMAMUX includes two sub-blocks:

- DMAMUX request multiplexer.
  DMAMUX request multiplexer inputs (Reqx_in) source from:
  - Peripherals (Peri_reqx).
  - DMAMUX request generator outputs (Gen_reqx).

  DMAMUX request multiplexer outputs (Reqx_out) is connected to channels of DMA controller.
  Synchronization inputs (Syncx_in) source from internal or external signals.
- DMAMUX request generator.
  Trigger inputs (Trgx_in) source from internal or external signals.

11.5.1. DMAMUX request multiplexer

The DMAMUX request multiplexer enables routing a DMA request line between the peripherals / generated DMA request and the DMA controllers of the product. Its component unit is the request multiplexer channels. DMA request lines are connected in parallel to all request multiplexer channels. There is a synchronization unit for each request multiplexer channel. The synchronization inputs are connected in parallel to all synchronization unit of request multiplexer channels. And there is a built-in DMAMUX request multiplexer counter for each request multiplexer channel.

Request multiplexer channel

A DMA request input for the DMAMUX request multiplexer channel x is configured by the MUXID[7:0] bits in the DMAMUX_RM_CHxCFG register, sourced either from the peripherals or from the DMAMUX request generator, the sources can refer to *Table 11-2. Request multiplexer input mapping*. A DMAMUX request multiplexer channel is connected and dedicated to one single channel of the DMA controller.

Note: The value 0 of MUXID[7:0] bits corresponds to no DMA request line is selected. It is not allowed to configure the same DMA request line (same non-null MUXID[7:0]) to two different request multiplexer channels.

When synchronization mode is disabled

Each time the connected DMAMUX request is served by the DMA controller, the served DMA request is de-asserted, and the built-in DMAMUX request multiplexer counter is decremented. At the request multiplexer counter underrun, the built-in DMAMUX request multiplexer counter is automatically loaded with the value in NBR[4:0] bits of the DMAMUX_RM_CHxCFG register. If the channel event generation is enabled by setting EVGEN bit, the number of DMA requests before an output event generation is NBR[4:0] + 1.

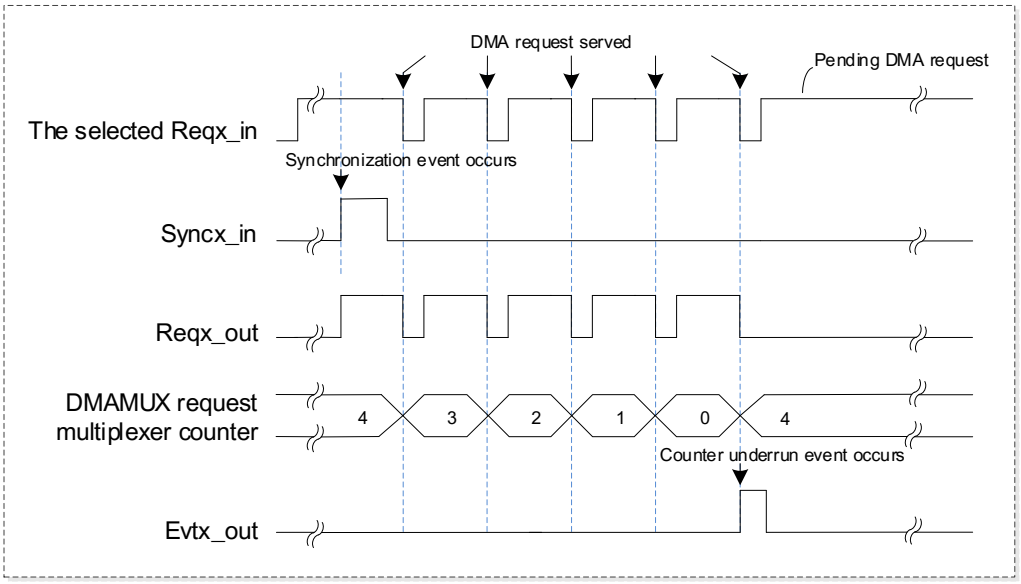
Note: The NBR[4:0] bits value shall only be written by software when both synchronization enable bit SYNCEN and event generation enable EVGEN bit of the corresponding request multiplexer channel x are disabled.

When synchronization mode is enabled

A channel x in synchronization mode, when a rising/falling edge on the selected synchronization input is detected, the pending selected input DMA request line is routed to the multiplexer channel x output. Each time the connected DMAMUX request is served by the DMA controller, the served DMA request is de-asserted, and the built-in DMAMUX request multiplexer counter is decremented. At the request multiplexer counter underrun, the input DMA request line is disconnected from the request multiplexer channel x output, and the built-in DMAMUX request multiplexer counter is automatically loaded with the value in NBR[4:0] bits of the DMAMUX_RM_CHxCFG register. The number of DMA requests transferred to the request multiplexer channel x output following a detected synchronization event is NBR[4:0] + 1.

Figure 11-2. Synchronization mode shows an example when NBR[4:0]=4, SYNCEN=1, EVGEN=1, SYNCP[1:0]=01.

Figure 11-2. Synchronization mode



DMAMUX request multiplexer channel x can be synchronized by setting the synchronization enable bit SYNCEN in the DMAMUX_RM_CHxCFG register. The synchronization input is selected by SYNCID[4:0] bits in the DMAMUX_RM_CHxCFG register, the sources can refer to *Table 11-4. Synchronization input mapping*. The synchronization input valid edge is configured by the SYNCP[1:0] bits of the DMAMUX_RM_CHxCFG register.

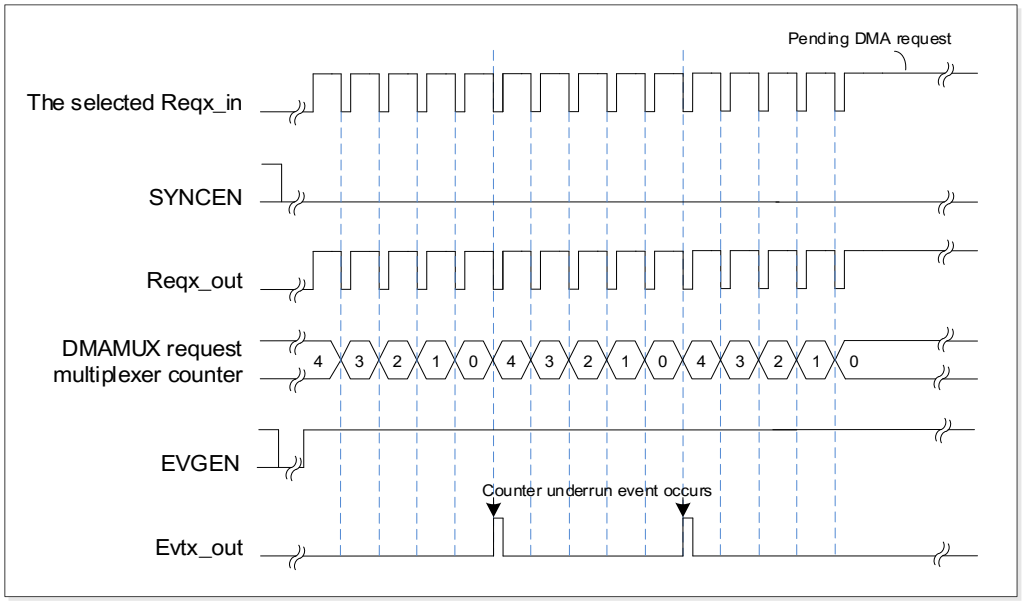
Note: If a synchronization input event occurs when there is no pending selected input DMA request line, the input event is discarded. The following asserted input request lines will not be routed to the DMAMUX multiplexer channel output until a synchronization input event occurs again.

Channel event generation

Each DMA request line multiplexer channel has an event output called Evtx_out, which is the DMA request multiplexer counter underrun event. Signals Evt0_out ~ Evt15_out can be used for DMA request chaining. If event generation bit EVGEN in the DMAMUX_RM_CHxCFG register is enabled on the channel x output, when its DMA request multiplexer counter is automatically reloaded with the value of the programmed NBR[4:0] field, the multiplexer channel generates a channel event, as a pulse of one AHB clock cycle.

Figure 11-3. Event generation shows an example when NBR[4:0]=4, SYNCEN=0, EVGEN=1.

Figure 11-3. Event generation



Note: If EVGEN = 1 and NBR[4:0] = 0, an event is generated after each served DMA request.

Synchronization overrun

If a new synchronization event occurs before the built-in DMAMUX request multiplexer counter underrun, the synchronization overrun flag bit SOIFx is set in the DMAMUX_RM_INTF register.

Note: The synchronization mode of request multiplexer channel x shall be disabled by resetting SYNCEN bit in DMAMUX_RM_CHxCFG register at the completion of the use of the related channel of the DMA controller. Otherwise, when a new synchronization event occurs, there will be a synchronization overrun due to the absence of a DMA acknowledge (that is, no served request) received from the DMA controller.

11.5.2. DMAMUX request generator

The DMAMUX request generator produces DMA requests upon trigger input event. Its component unit is the request generator channels. DMA request trigger inputs are connected in parallel to all request generator channels. And there is a built-in DMAMUX request generator counter for each request generator channel.

The active edge of trigger input events is selected through the RGTP[1:0] bits in DMAMUX_RG_CHxCFG register. The DMA request trigger input for the DMAMUX request generator channel x is selected through the TID[4:0] bits in DMAMUX_RG_CHxCFG register, the sources can refer to ***Table 11-3. Trigger input mapping***. DMAMUX request generator channel x can be enabled by setting RGEN to 1 in DMAMUX_RG_CHxCFG register.

Request generator channel

Upon the trigger input event, the corresponding request generator channel starts generating DMA requests on its output, and the output goes to the input of the DMAMUX request multiplexer. Each time the DMAMUX generated request is served by the connected DMA controller, the served request will be de-asserted, and the built-in DMAMUX request generator counter of the request generator channel is decremented. At the request generator counter underrun, the request generator channel stops generating DMA requests. The built-in DMAMUX request generator counter will be automatically reloaded to its programmed value upon the next trigger input event, the built-in counter is programmed by the NBRG[4:0] bits of the DMAMUX_RG_CHxCFG register.

Note: The number of generated DMA requests after the trigger input event is NBRG[4:0] + 1. The NBRG[4:0] value shall only be written by software when the RGEN bit of the corresponding generator channel x is disabled.

Trigger overrun

If a request generator channel x was enabled by RGEN bit, when a new DMA request trigger event for the request generator channel x occurs before the DMAMUX request generator counter underrun, then the request trigger overrun event flag bit TOIFx is set by hardware in the DMAMUX_RG_INTF register.

Note: The request generator channel x shall be disabled by resetting RGEN bit in DMAMUX_RG_CHxCFG register at the completion of the usage of the related channel of the DMA controller. Otherwise, when a new detected trigger event occurs, there will be a trigger overrun due to the absence of an acknowledge (that is, no served request) received from the DMA.

11.5.3. Channel configurations

The following sequence should be followed to configure a DMAMUX channel y and the related DMA channel x:

1. Set and configure the DMA channel x completely, except enabling the channel x.
2. Set and configure the related DMAMUX channel y completely.
3. Configure the CHEN bit with '1' in the DMA_CHxCTL register to enable the DMA channel x.

11.5.4. Interrupt

There are two types of interrupt event, including synchronization overrun event on each DMAMUX request multiplexer channel, and trigger overrun event on each DMAMUX request generator channel.

Each interrupt event has a dedicated flag bit, a dedicated clear bit, and a dedicated enable bit. The relationship is described in the following ***Table 11-1. Interrupt events***.

Table 11-1. Interrupt events

Interrupt event	Flag bit	Clear bit	Enable bit
Synchronization overrun event on DMAMUX request multiplexer channel x	SOIFx in DMAMUX_RM_INTF register	SOIFCx in DMAMUX_RM_INTC register	SOIE in DMAMUX_RM_CHxCFG register
Trigger overrun event on DMAMUX request generator channel y	TOIFy in DMAMUX_RG_INTF register	TOIFCy in DMAMUX_RG_INTC register	TOIE in DMAMUX_RG_CHxCFG register

Trigger overrun interrupt

When the DMAMUX request trigger overrun flag TOIFx is set, and the trigger overrun interrupt is enabled by setting TOIE bit, a trigger overrun interrupt will be generated. The overrun flag TOIFx is reset by writing 1 to the corresponding clear bit of overrun flag TOIFCx in the DMAMUX_RG_INTC register.

Synchronization overrun interrupt

When the synchronization overrun flag SOIFx is set, and the synchronization overrun interrupt is enabled by setting SOIE bit, a synchronization overrun interrupt will be generated. The overrun flag SOIFx is reset by writing 1 to the corresponding clear bit of synchronization overrun flag bit SOIFCx in the DMAMUX_RM_INTC register.

11.5.5. DMAMUX mapping

DMAMUX is used with DMA0 and DMA1. Channel 0 to 7 of DMAMUX are connected to channel 0 to 7 of DMA0, channel 8 to 15 of DMAMUX are connected to channel 0 to 7 of DMA1.

Request multiplexer input mapping

A DMA request is sourced either from the peripherals or from the DMAMUX request generator, the sources can refer to ***Table 11-2. Request multiplexer input mapping***, configured by the

MUXID[7:0] bits in the DMAMUX_RM_CHxCFG register for the DMAMUX request multiplexer channel x.

Table 11-2. Request multiplexer input mapping

Request multiplexer channel input identification MUXID[7:0]	Source
1	Gen_req0
2	Gen_req1
3	Gen_req2
4	Gen_req3
5	Gen_req4
6	Gen_req5
7	Gen_req6
8	Gen_req7
9	Reserved
10	Reserved
11	Reserved
12	DAC0
13	DAC1
14	TIMER5_UP0
15	TIMER6_UP0
16	SPI0_RX
17	SPI0_TX
18	SPI1_RX
19	SPI1_TX
20	SPI2_RX
21	SPI2_TX
22	SPI3_RX
23	SPI3_TX
24	SPI4_I2S_RX
25	SPI4_I2S_TX
26	SPI5_I2S_RX
27	SPI5_I2S_TX
28	I2C0_RX
29	I2C0_TX
30	I2C1_RX
31	I2C1_TX
32	Reserved
33	Reserved
34	Reserved
35	Reserved
36	Reserved

Request multiplexer channel input identification MUXID[7:0]	Source
37	Reserved
38	Reserved
39	Reserved
40	Reserved
41	Reserved
42	Reserved
43	Reserved
44	Reserved
45	Reserved
46	Reserved
47	Reserved
48	LINFlexD0_RX
49	LINFlexD0_TX
50	LINFlexD1_RX
51	LINFlexD1_TX
52	LINFlexD2_RX
53	LINFlexD2_TX
54	LINFlexD3_RX
55	LINFlexD3_TX
56	LINFlexD4_RX
57	LINFlexD4_TX
58	LINFlexD5_RX
59	LINFlexD5_TX
60	Reserved
61	Reserved
62	Reserved
63	Reserved
64	Reserved
65	Reserved
66	Reserved
67	Reserved
68	Reserved
69	Reserved
70	Reserved
71	Reserved
72	TIMER0_CH0
73	TIMER0_CH1
74	TIMER0_CH2
75	TIMER0_CH3
76	TIMER0_CH0N

Request multiplexer channel input identification MUXID[7:0]	Source
77	TIMER0_CH1N
78	TIMER0_CH2N
79	TIMER0_CH3N
80	TIMER0_UP
81	TIMER0_TRG
82	TIMER0_CMT
83	TIMER7_CH0
84	TIMER7_CH1
85	TIMER7_CH2
86	TIMER7_CH3
87	TIMER7_CH0N
88	TIMER7_CH1N
89	TIMER7_CH2N
90	TIMER7_CH3N
91	TIMER7_UP
92	TIMER7_TRG
93	TIMER7_CMT
94	TIMER19_CH0
95	TIMER19_CH1
96	TIMER19_CH2
97	TIMER19_CH3
98	TIMER19_CH0N
99	TIMER19_CH1N
100	TIMER19_CH2N
101	TIMER19_CH3N
102	TIMER19_UP
103	TIMER19_TRG
104	TIMER19_CMT
105	TIMER20_CH0
106	TIMER20_CH1
107	TIMER20_CH2
108	TIMER20_CH3
109	TIMER20_CH0N
110	TIMER20_CH1N
111	TIMER20_CH2N
112	TIMER20_CH3N
113	TIMER20_UP
114	TIMER20_TRG
115	TIMER20_CMT
116	TIMER60_CH0

Request multiplexer channel input identification MUXID[7:0]	Source
117	TIMER60_CH1
118	TIMER60_CH2
119	TIMER60_CH3
120	TIMER60_CH0N
121	TIMER60_CH1N
122	TIMER60_CH2N
123	TIMER60_CH3N
124	TIMER60_UP
125	TIMER60_TRG
126	TIMER60_CMT
127	TIMER61_CH0
128	TIMER61_CH1
129	TIMER61_CH2
130	TIMER61_CH3
131	TIMER61_CH0N
132	TIMER61_CH1N
133	TIMER61_CH2N
134	TIMER61_CH3N
135	TIMER61_UP
136	TIMER61_TRG
137	TIMER61_CMT
138	TIMER62_CH0
139	TIMER62_CH1
140	TIMER62_CH2
141	TIMER62_CH3
142	TIMER62_CH0N
143	TIMER62_CH1N
144	TIMER62_CH2N
145	TIMER62_CH3N
146	TIMER62_UP
147	TIMER62_TRG
148	TIMER62_CMT
149	TIMER63_CH0
150	TIMER63_CH1
151	TIMER63_CH2
152	TIMER63_CH3
153	TIMER63_CH0N
154	TIMER63_CH1N
155	TIMER63_CH2N
156	TIMER63_CH3N

Request multiplexer channel input identification MUXID[7:0]	Source
157	TIMER63_UP
158	TIMER63_TRG
159	TIMER63_CMT
160	TIMER1_CH0
161	TIMER1_CH1
162	TIMER1_CH2
163	TIMER1_CH3
164	TIMER1_UP
165	TIMER1_TRG
166	TIMER2_CH0
167	TIMER2_CH1
168	TIMER2_CH2
169	TIMER2_CH3
170	TIMER2_UP
171	TIMER2_TRG
172	QSPI
173	Reserved
174	Reserved
175	Reserved
176	Reserved
177	DTM_CAN0_RX0
178	DTM_CAN0_RX1
179	DTM_CAN0_TX
180	DTM_CAN0_TXE
181	DTM_CAN1_RX0
182	DTM_CAN1_RX1
183	DTM_CAN1_TX
184	DTM_CAN1_TXE
185	ADC0_ROUTINE
186	ADC0_INSERTED0
187	ADC0_INSERTED1
188	ADC0_INSERTED2
189	ADC0_INSERTED3
190	ADC0_INSERTED4
191	ADC0_INSERTED5
192	ADC1_ROUTINE
193	ADC1_INSERTED0
194	ADC1_INSERTED1
195	ADC1_INSERTED2
196	ADC1_INSERTED3

Request multiplexer channel input identification MUXID[7:0]	Source
197	ADC1_INSERTED4
198	ADC1_INSERTED5
199	Reserved
200	Reserved
201	Reserved
202	Reserved
203	Reserved
204	Reserved
205	Reserved
206	TIMER5_UP1
207	TIMER5_UP2
208	TIMER5_UP3
209	TIMER6_UP1
210	TIMER6_UP2
211	TIMER6_UP3
212	DTM_CAN2_RX0
213	DTM_CAN2_RX1
214	DTM_CAN2_TX
215	DTM_CAN2_TXE
216	DTM_CAN3_RX0
217	DTM_CAN3_RX1
218	DTM_CAN3_TX
219	DTM_CAN3_TXE
220	DTM_CAN4_RX0
221	DTM_CAN4_RX1
222	DTM_CAN4_TX
223	DTM_CAN4_TXE
224	DTM_CAN5_RX0
225	DTM_CAN5_RX1
226	DTM_CAN5_TX
227	DTM_CAN5_TXE
228	DTM_CAN6_RX0
229	DTM_CAN6_RX1
230	DTM_CAN6_TX
231	DTM_CAN6_TXE
232	DTM_CAN7_RX0
233	DTM_CAN7_RX1
234	DTM_CAN7_TX
235	DTM_CAN7_TXE

Trigger input mapping

The DMA request trigger input for the DMAMUX request generator channel x is selected through the TID[4:0] bits in DMAMUX_RG_CHxCFG register, the sources can refer to ***Table 11-3. Trigger input mapping***.

Table 11-3. Trigger input mapping

Trigger input identification TID[4:0]	Source
0	EXTI_0
1	EXTI_1
2	EXTI_2
3	EXTI_3
4	EXTI_4
5	EXTI_5
6	EXTI_6
7	EXTI_7
8	EXTI_8
9	EXTI_9
10	EXTI_10
11	EXTI_11
12	EXTI_12
13	EXTI_13
14	EXTI_14
15	EXTI_15
16	Evt0_out
17	Evt1_out
18	Evt2_out
19	Evt3_out
20	Evt4_out
21	Evt5_out
22	Evt6_out
23	Evt7_out
24	TIMER1_TRGO

Note: *When using EXTI as the trigger input source, the EXTI pending flag must be cleared manually before the next trigger input can be performed.*

Synchronization input mapping

The synchronization input is selected by SYNCID[4:0] bits in the DMAMUX_RM_CHxCFG register, the sources can refer to ***Table 11-4. Synchronization input mapping***.

Table 11-4. Synchronization input mapping

Synchronization input identification SYNCID[4:0]	Source
0	EXTI_0
1	EXTI_1
2	EXTI_2
3	EXTI_3
4	EXTI_4
5	EXTI_5
6	EXTI_6
7	EXTI_7
8	EXTI_8
9	EXTI_9
10	EXTI_10
11	EXTI_11
12	EXTI_12
13	EXTI_13
14	EXTI_14
15	EXTI_15
16	Evt0_out
17	Evt1_out
18	Evt2_out
19	Evt3_out
20	Evt4_out
21	Evt5_out
22	Evt6_out
23	Evt7_out
24	TIMER1_TRGO

Note: *When using EXTI as the synchronization input source, the EXTI pending flag must be cleared manually before the next synchronization input can be performed.*

11.6. Register definition

DMAMUX base address: 0x4900 0800

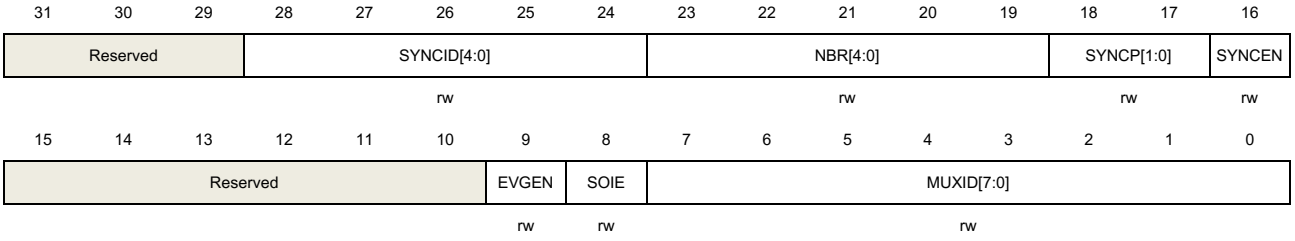
Table 11-5. DMAMUX register overview

Register name	Symbol	Address offset	Reset value
Request multiplexer channel x configuration register	DMAMUX_RM_CHxCFG	0x00	0x0000 0000
Request multiplexer channel interrupt flag register	DMAMUX_RM_INTF	0x80	0x0000 0000
Request multiplexer channel interrupt flag clear register	DMAMUX_RM_INTC	0x84	0x0000 0000
Request generator channel x configuration register	DMAMUX_RG_CHxCFG	0x100	0x0000 0000
Request generator channel interrupt flag register	DMAMUX_RG_INTF	0x140	0x0000 0000
Request generator channel interrupt flag clear register	DMAMUX_RG_INTC	0x144	0x0000 0000

11.6.1. Request multiplexer channel x configuration register (DMAMUX_RM_CHxCFG)

x = 0...15, where x is a channel number
Address offset: 0x00 + 0x04 * x
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:29	Reserved	Must be kept at reset value.
28:24	SYNCID[4:0]	Synchronization input identification Selects the synchronization input source.
23:19	NBR[4:0]	Number of DMA requests to forward The number of DMA requests to forward to the DMA controller after a synchronization event / before an output event is generated equals to NBR[4:0] + 1. These bits shall only be written when both SYNCEN and EVGEN bits are disabled.

18:17	SYNCP[1:0]	Synchronization input polarity 00: No event detection 01: Rising edge 10: Falling edge 11: Rising and falling edges
16	SYNCEN	Synchronization enable 0: Disable synchronization 1: Enable synchronization
15:10	Reserved	Must be kept at reset value.
9	EVGEN	Event generation enable 0: Disable event generation 1: Enable event generation
8	SOIE	Synchronization overrun interrupt enable 0: Disable interrupt 1: Enable interrupt
7:0	MUXID[7:0]	Multiplexer input identification Selects the input DMA request in multiplexer input sources.

11.6.2. Request multiplexer channel interrupt flag register (DMAMUX_RM_INTF)

Address offset: 0x80

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOIF15	SOIF14	SOIF13	SOIF12	SOIF11	SOIF10	SOIF9	SOIF8	SOIF7	SOIF6	SOIF5	SOIF4	SOIF3	SOIF2	SOIF1	SOIF0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	SOIFx	Synchronization overrun event flag of request multiplexer channel x (x = 0…15) If a synchronization event occurs when the DMAMUX request counter value is lower than NBR[4:0], the flag is set. It is cleared by writing 1 to the corresponding SOIFCx bit in DMAMUX_RM_INTC register.

11.6.3. Request multiplexer channel interrupt flag clear register (DMAMUX_RM_INTC)

Address offset: 0x84
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOIFC15	SOIFC14	SOIFC13	SOIFC12	SOIFC11	SOIFC10	SOIFC9	SOIFC8	SOIFC7	SOIFC6	SOIFC5	SOIFC4	SOIFC3	SOIFC2	SOIFC1	SOIFC0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
31:0	SOIFCx	Clear bit for synchronization overrun event flag of request multiplexer channel x (x = 0…15) Writing 1 clears the corresponding overrun flag SOIFx in the DMAMUX_RM_INTF register.

11.6.4. Request generator channel x configuration register (DMAMUX_RG_CHxCFG)

x = 0...7, where x is a channel number

Address offset: 0x100 + 0x04 * x
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								NBRG[4:0]					RGTP[1:0]		RGEN
										rw			rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TOIE	Reserved			TID[4:0]				
							rw				rw				

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:19	NBRG[4:0]	Number of DMA requests to be generated The number of DMA requests to be generated after a trigger event equals to NBRG[4:0] + 1.

Note: These bits shall only be written when RGEN bit is disabled.

18:17	RGTP[1:0]	DMAMUX request generator trigger polarity 00: No event trigger detection 01: Rising edge 10: Falling edge 11: Rising and falling edges
16	RGEN	DMAMUX request generator channel x enable 0: Disable DMAMUX request generator channel x 1: Enable DMAMUX request generator channel x
15:9	Reserved	Must be kept at reset value.
8	TOIE	Trigger overrun interrupt enable 0: Disable interrupt 1: Enable interrupt
7:5	Reserved	Must be kept at reset value.
4:0	TID[4:0]	Trigger input identification Selects the DMA request trigger input source.

11.6.5. Request generator channel interrupt flag register (DMAMUX_RG_INTF)

Address offset: 0x140

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TOIF7	TOIF6	TOIF5	TOIF4	TOIF3	TOIF2	TOIF1	TOIF0
								r	r	r	r	r	r	r	r

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	TOIFx	Trigger overrun event flag of request generator channel x (x = 0…7) If a new trigger event occurs before the request generator counter underrun, the flag is set. It is cleared by writing 1 to the corresponding TOIFCx bit in the DMAMUX_RG_INTC register.

11.6.6. Request generator channel interrupt flag clear register (DMAMUX_RG_INTC)

Address offset: 0x144

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TOIFC7	TOIFC6	TOIFC5	TOIFC4	TOIFC3	TOIFC2	TOIFC1	TOIFC0
								w	w	w	w	w	w	w	w

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	TOIFCx	Clear bit for trigger overrun event flag of request generator channel x (x = 0…7) Writing 1 clears the corresponding trigger overrun flag TOIFx in the DMAMUX_RG_INTF register.

12. Debug (DBG)

12.1. Overview

The GD32A711/GD32A712 series provide a large variety of debug, trace and test features. They are implemented with a standard configuration of the Arm® CoreSight™ module together with a daisy chained standard TAP controller. Debug and trace functions are integrated into the ARM® Cortex®-M7. The debug system supports serial wire debug (SWD) and trace functions in addition to standard JTAG debug. The debug and trace functions refer to the following documents:

- Cortex®-M7 Technical Reference Manual
- ARM® Debug Interface v5 Architecture Specification

The debug system provides debuggers with a function to keep certain peripherals in their current state for TIMER, WWDGT, FWDGT, I2C or MFCOM.

12.2. JTAG / SW function overview

Debug capabilities can be accessed by a debug tool via serial wire (SW - Debug Port) or JTAG interface (JTAG - Debug Port).

12.2.1. Pin assignment

The JTAG interface provides 4-pin standard JTAG, known as JTAG clock (JTCK), JTAG mode selection (JTMS), JTAG data input (JTDI) and JTAG data output (JTDO). The serial wire debug (SWD) provide 2-pin SW interface, known as SW data input/output (SWDIO) and SW clock (SWCLK). The two SW pins are multiplexed with two of four JTAG pins, which is SWDIO multiplexed with JTMS, SWCLK multiplexed with JTCK. The JTDO is also used as trace async data output (TRACESWO) when async trace enabled.

Table 12-1. Pin assignment

Pin	Debug interface
PA15	JTDI
PA14	JTCK/SWCLK
PA13	JTMS/SWDIO
PB3	JTDO

By default, 4-pin standard JTAG debug mode is chosen after reset. If switch to SW debug mode, the PA15/PB3 are released to other GPIO functions. When the debug mode switches from JTAG mode to SWD mode, both PA15 and PB3 will switch to alternate function mode, with functions remaining as JTDI and JTDO, and the ports are left floating. If JTAG and SW not used, all 4-pin can be released to other GPIO functions. When these pins are released as general-purpose GPIOs, they are all set to alternate mode, left floating, and will not affect

other functions.

12.2.2. JTAG daisy chained structure

The Cortex®-M7 JTAG TAP (CPU JTAG) is connected to a boundary-scan (BSD) JTAG TAP (MCU JTAG). The BSD JTAG IR is 5-bit width, while the Cortex®-M7 JTAG IR is 4-bit width. So when JTAG in IR shift step, it first shift 5-bit BYPASS instruction (5'b 11111) for BSD JTAG, and then shift normal 4-bit instruction for Cortext-M7 JTAG. Because of the data shift under BSD JTAG BYPASS mode, adding 1 extra bit to the data chain is needed.

The BSD JTAG IDCODE is 0x790007A3.

The method to enter BSD mode can refer to the ARM® Debug Interface v5 Architecture Specification.

12.2.3. Debug reset

The JTAG-DP and SW-DP registers are in the power on reset domain. The system reset initializes the majority of the Cortex®-M7, excluding NVIC and debug logic (FPB, DWT, and ITM).

12.2.4. JEDEC-106 ID code

The Cortex®-M7 integrates JEDEC-106 ID code, which is located in ROM table and mapped on the address of 0xE00FF000_0xE00FFFFF.

12.3. Debug hold function overview

12.3.1. Debug support for TIMER, I2C, WWDGT, FWDGT and MFCOM

When the core halted and the corresponding bit in DBG control register x (DBG_CTLx, x=0,1,2) is set, the following behaved.

For TIMER, the timer counters stopped and hold for debug.

For I2C, SMBUS timeout hold for debug.

For WWDGT or FWDGT, the counter clock stopped for debug.

For MFCOM, the timer counters stopped and hold for debug.

12.4. Register definition

DBG base address: 0xE00E1000

Table 12-2. DBG register overview

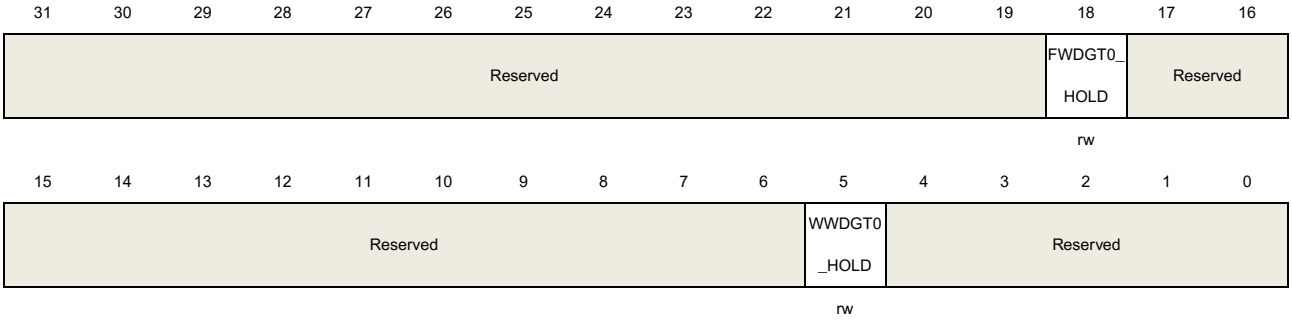
Register Name	Symbol	Address Offset	Reset Value
Control register 0	DBG_CTL0	0x00	0x0000 0000
Control register 1	DBG_CTL1	0x04	0x0000 0000
Control register 2	DBG_CTL2	0x08	0x0000 0000

12.4.1. Control register0 (DBG_CTL0)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18	FWDGT0_HOLD	FWDGT0 hold bit This bit is set and reset by software. 0: no effect 1: Hold the FWDGT0 counter clock for debug when core M7 halted.
17:6	Reserved	Must be kept at reset value.
5	WWDGT0_HOLD	WWDGT0 hold bit This bit is set and reset by software. 0: no effect 1: Hold the WWDGT0 counter clock for debug when core M7 halted.
4:0	Reserved	Must be kept at reset value.

12.4.2. Control register1 (DBG_CTL1)

Address offset: 0x004

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									I2C1_HOLD	I2C0_HOLD	Reserved				
									rw	rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										TIMER6_HOLD	TIMER5_HOLD	Reserved		TIMER2_HOLD	TIMER1_HOLD
										rw	rw			rw	rw

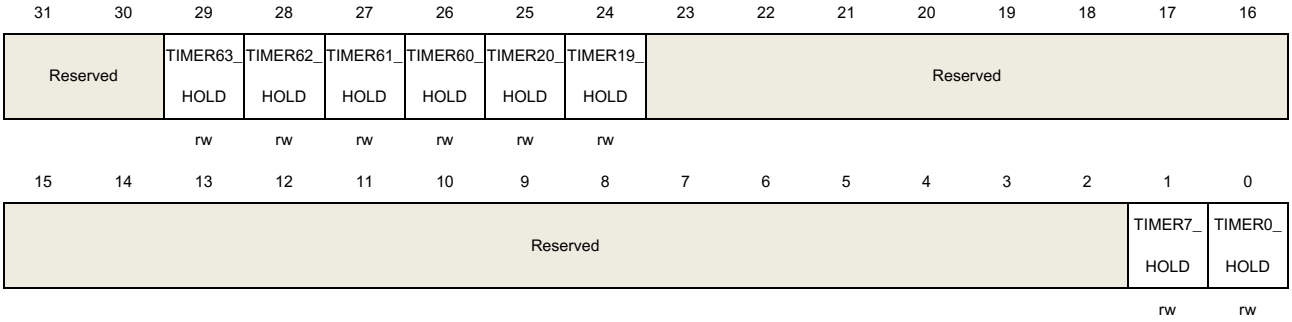
Bits	Fields	Descriptions
31:23	Reserved	Must be kept at reset value.
22	I2C1_HOLD	I2C1 hold bit This bit is set and reset by software. 0: no effect 1: Hold the I2C1 SMBUS timeout for debug when core M7 halted.
21	I2C0_HOLD	I2C0 hold bit This bit is set and reset by software. 0: no effect 1: Hold the I2C0 SMBUS timeout for debug when core M7 halted.
20:6	Reserved	Must be kept at reset value.
5	TIMER6_HOLD	TIMER6 hold bit This bit is set and reset by software. 0: no effect 1: Hold the TIMER6 counter for debug when core M7 halted.
4	TIMER5_HOLD	TIMER5 hold bit This bit is set and reset by software. 0: no effect 1: Hold the TIMER5 counter for debug when core M7 halted.
3:2	Reserved	Keep as reserved value
1	TIMER2_HOLD	TIMER2 hold bit This bit is set and reset by software. 0: no effect 1: Hold the TIMER2 counter for debug when core M7 halted.
0	TIMER1_HOLD	TIMER1 hold bit This bit is set and reset by software. 0: no effect 1: Hold the TIMER1 counter for debug when core M7 halted.

12.4.3. Control register2 (DBG_CTL2)

Address offset: 0x008

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29	TIMER63_HOLD	TIMER63 hold bit This bit is set and reset by software. 0: no effect 1: Hold the TIMER63 counter for debug when core M7 halted.
28	TIMER62_HOLD	TIMER62 hold bit This bit is set and reset by software. 0: no effect 1: Hold the TIMER62 counter for debug when core M7 halted.
27	TIMER61_HOLD	TIMER61 hold bit This bit is set and reset by software. 0: no effect 1: Hold the TIMER61 counter for debug when core M7 halted.
26	TIMER60_HOLD	TIMER60 hold bit This bit is set and reset by software. 0: no effect 1: Hold the TIMER60 counter for debug when core M7 halted.
25	TIMER20_HOLD	TIMER20 hold bit This bit is set and reset by software. 0: no effect 1: Hold the TIMER20 counter for debug when core M7 halted.
24	TIMER19_HOLD	TIMER19 hold bit This bit is set and reset by software. 0: no effect 1: Hold the TIMER19 counter for debug when core M7 halted.

23:2	Reserved	Keep as reserved value
1	TIMER7_HOLD	TIMER7 hold bit This bit is set and reset by software. 0: no effect 1: Hold the TIMER7 counter for debug when core M7 halted.
0	TIMER0_HOLD	TIMER0 hold bit This bit is set and reset by software. 0: no effect 1: Hold the TIMER0 counter for debug when core M7 halted.

13. Analog-to-digital converter (ADC)

13.1. Overview

The 12-bit ADC is an analog-to-digital converter using the successive approximation method. ADC0 and ADC1 both have 34 multiplexed channels making the ADC convert analog signals from 24 external channels, 4 external extended channels with 32 channels, and 6 internal channels. The analog watchdog allows the application to detect whether the input voltage goes outside the user-defined higher or lower thresholds. The analog signals of the channels can be converted by the ADC in one-shot mode, continuous mode. A left-aligned or right-aligned 16-bit data register holds the output of the ADC. An on-chip hardware average scheme improves performances while off-loading the related computational burden from the MCU.

13.2. Characteristics

- High performance:
  - 12-bit, 10-bit, 8-bit or 6-bit configurable resolution.
  - Self-calibration.
  - Programmable sampling time.
  - Data alignment with built-in data coherency.
  - DMA support.
- Analog input channels:
  - 28 external analog inputs (the last 4 channels are external extend channels for 32 channels) in ADC0 and ADC1.
  - 2 channels for DAC (V_{DAC0_OUT0} and V_{DAC1_OUT0}).
  - 1 channel for internal temperature sensor (V_{SENSE}).
  - 1 channel for internal reference voltage (V_{REFINT}).
  - 2 channels for presample and self test (V_{REFP}, V_{REFN}).
- Start-of-conversion can be initiated:
  - By software.
  - By TRIGSEL.
- Conversion modes:
  - One-shot mode converts the selected channels of sequence once per trigger.
  - Continuous mode converts the sequence of selected inputs continuously.
  - SYNC mode (the device with two ADCs).
- Analog watchdog: Up to 4 analog watchdogs to monitor conversion data for each input channel and issue an interrupt if the converted data does not meet the limit of watchdog events.
- Interrupt generation:
  - at the end of routine and inserted sequence conversions.
  - at end of per channel conversion.

- analog watchdog event.
- self test error event.

- Averaging: Option to improve accuracy via averaging, which calculates conversion data by averaging the data of up to 256 conversions.
- Auto turn-off of the conversion clock when ADC is idle.
- Self test functions validate ADC structural integrity during functional operation and create events with different severities on any finding.
- ADC input range: $V_{REFN} \leq V_{IN} \leq V_{REFP}$.

13.3. Pins and internal signals

Figure 13-1. ADC module block diagram shows the ADC block diagram and *Table 13-2. ADC pins definition* gives the ADC pin description. *Table 13-3. ADC channel remapping* gives the mapping relationship of ADC channels.

Table 13-1. ADC internal signals

Internal signal name	Signal type	Description
V_{SENSE}	Input	Internal temperature sensor output voltage
V_{REFINT}	Input	Internal voltage reference output voltage

Table 13-2. ADC pins definition

Name	Signal type	Remarks
V_{DDA}	Input, analog power supply	Main I/O and analog power supply
V_{SS}	Input, power supply ground	Ground for power supply
V_{REFP}	Input, analog reference positive	The positive reference voltage for the ADC
V_{REFN}	Input, analog reference negative	The negative reference voltage for the ADC, $V_{REFN} = V_{SS}$
ADCx_IN[23:0] (x=0/1) ADCx_EECIN[3:0] (x=0/1)	Input, Analog signals	Up to 28 external channels(x=0/1),

Table 13-3. ADC channel remapping

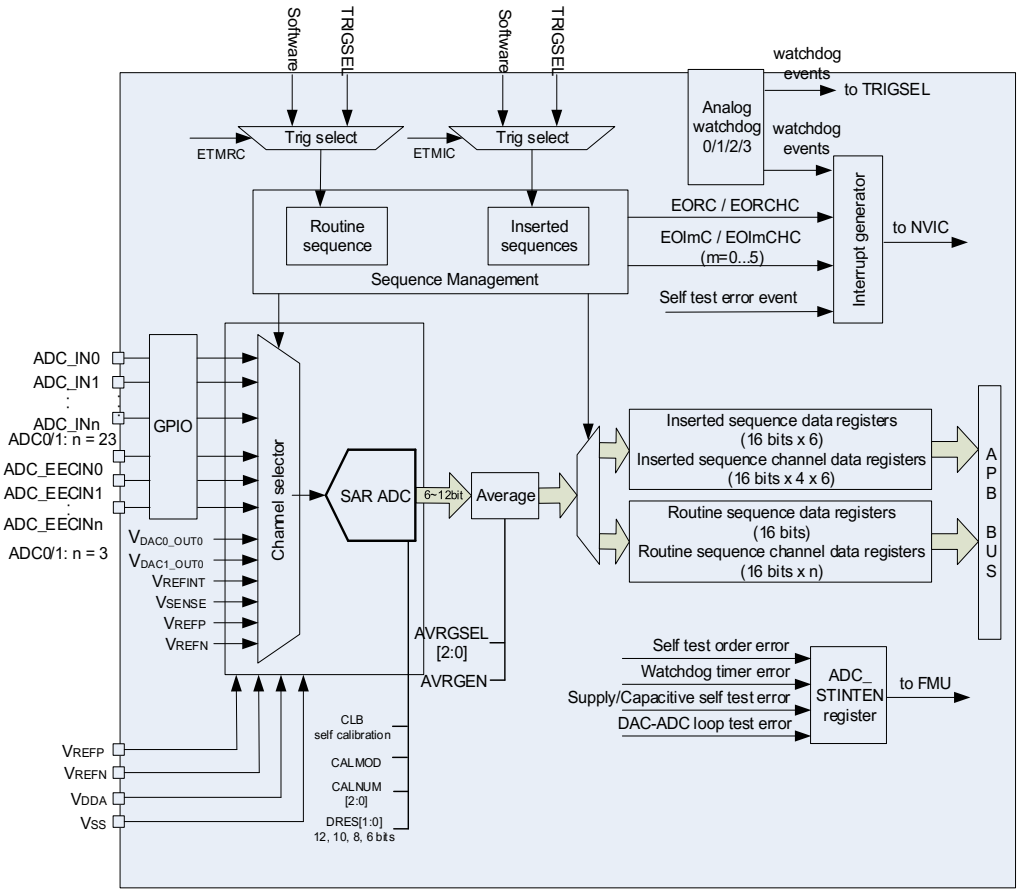
ADC0/ADC1	
ADC channel	**ADC pin / input**
ADC channel 0	ADC_IN0
ADC channel 1	ADC_IN1
ADC channel 2	ADC_IN2
ADC channel 3	ADC_IN3
ADC channel 4	ADC_IN4
ADC channel 5	ADC_IN5

ADC0/ADC1	
ADC channel	**ADC pin / input**
ADC channel 6	ADC_IN6
ADC channel 7	ADC_IN7
ADC channel 8	ADC_IN8
ADC channel 9	ADC_IN9
ADC channel 10	ADC_IN10
ADC channel 11	ADC_IN11
ADC channel 12	ADC_IN12
ADC channel 13	ADC_IN13
ADC channel 14	ADC_IN14
ADC channel 15	ADC_IN15
ADC channel 16	ADC_IN16
ADC channel 17	ADC_IN17
ADC channel 18	ADC_IN18
ADC channel 19	ADC_IN19
ADC channel 20	ADC_IN20
ADC channel 21	ADC_IN21
ADC channel 22	ADC_IN22
ADC channel 23	ADC_IN23
ADC channel 24	ADC_EECIN0 (When this channel is used, the ADCx_EECSEL0, ADCx_EECSEL1 and ADCx_EECSEL2 pins need to be configured for the selection signal of the external analog multiplexer)
ADC channel 25	
ADC channel 26	
ADC channel 27	
ADC channel 28	
ADC channel 29	
ADC channel 30	
ADC channel 31	
ADC channel 32	ADC_EECIN1 (When this channel is used, the ADCx_EECSEL0, ADCx_EECSEL1 and ADCx_EECSEL2 pins need to be configured for the selection signal of the external analog multiplexer)
ADC channel 33	
ADC channel 34	
ADC channel 35	
ADC channel 36	
ADC channel 37	
ADC channel 38	
ADC channel 39	
ADC channel 40	ADC_EECIN2 (When this channel is used, the ADCx_EECSEL0, ADCx_EECSEL1 and ADCx_EECSEL2 pins need to be configured for the selection signal of the external analog multiplexer)
ADC channel 41	
ADC channel 42	
ADC channel 43	
ADC channel 44	
ADC channel 45	
ADCchannel 46	

ADC0/ADC1	
ADC channel	**ADC pin / input**
ADC channel 47	
ADC channel 48	ADC_EECIN3 (When this channel is used, the ADCx_EECSEL0, ADCx_EECSEL1 and ADCx_EECSEL2 pins need to be configured for the selection signal of the external analog multiplexer)
ADC channel 49	
ADC channel 50	
ADC channel 51	
ADC channel 52	
ADC channel 53	
ADC channel 54	
ADC channel 55	
ADC channel 56	V_{DAC0_OUT0}
ADC channel 57	V_{DAC1_OUT0}
ADC channel 58	V_{REFINT}
ADC channel 59	V_{SENSE}
ADC channel 60	V_{REFP}
ADC channel 61	V_{REFN}

13.4. Function Description

Figure 13-1. ADC module block diagram



13.4.1. Foreground calibration function

The ADC has a foreground calibration feature. During the procedure, the ADC calculates a calibration factor which is internally applied to the ADC until the next ADC power-off. The application must not use the ADC during calibration and must wait until it is completed. The calibration should be performed before starting A/D conversion. The calibration is initiated by software by setting bit CLB=1. CLB bit stays at 1 during all the calibration sequence. It is then cleared by hardware as soon as the calibration is completed.

The calibration process can be repeated by a configurable number of times by setting the CALNUM[2:0] bits in ADC_CTL register, and the results will be averaged to get a more precise calibration result.

When the ADC operating conditions change (such as supply power voltage V_{DDA}, positive reference voltage V_{REFP}, temperature and so on), it is recommended to re-run a calibration cycle.

The internal analog calibration can be reset by setting the RSTCLB bit in ADC_CTL register.

Calibration software procedure:

1. Ensure that ADCON = 1.
2. Delay 14 CK_ADC to wait for ADC stability.
3. Set CALNUM.
4. Set RSTCLB (optional)
5. Set CLB=1.
6. Wait until CLB=0.

Note: The CKAUTO bit can only be set to 1 after ADC calibration is completed.

13.4.2. Dual clock domain architecture

The ADC sub-module, with exception of the APB interface block, is feed by an ADC clock, which can be asynchronous and independent from the APB clock. The maximum ADC frequency is 16MHz.

The RCU controller has a dedicated programmable prescaler for the ADC clock.Refer to RCU Section for more information on generating this clock source.

Note: No matter what clock, the timing sequence in adc is always regarded as async clock.

13.4.3. Low power mode

Power-down

The ADCON bit on the ADC_CTL register is the enable switch of the ADC module. The ADC module will keep in reset state if this bit is 0. For power saving, when this bit is reset, the analog sub-module will be put into power-down mode. The ADCON bit is 0 by default value after reset, so ADC analog module is kept in power-down mode. This state must be exited before starting any operation by setting the ADCON field. In power-down mode, no conversion can be started.

The ADCON bit can be written at any time, but is generally recommended for use in continuous mode: The ongoing operation should quit continuous mode first by setting RMODE to 0, the ongoing operation be aborted manually by setting the ABORT field of routine sequence to 1 before resetting the ADCON to 0.

ADC power-down mode state (the state is idle) is indicated by ADCSTAT bit in ADC_STAT register when ADC enters into power-down mode.

After ADC is enabled, you need delay t_{su} time for sampling, the value of t_{su} please refer to the device datasheet.

Clock automatic off

Through setting the CKAUTO bit in the ADC_CTL register to 1, can enable the clock automatic off mode to reduce the ADC power consumption. After this mode is enabled, the clock

CK_ADC automatically shuts down when the ADC module is in idle state.

Note: The CKAUTO bit can only be set to 1 after ADC calibration is completed.

13.4.4. Sequence management

Conversion sequence

Each ADC has seven sequences, a routine sequence and six inserted sequences. Inserted sequences are numbered from 0 to 5. Every sequence has a lot of triggers and any one of these triggers can start the sequence conversion.

Sequence priority

Inserted sequence priority is based on the sequence number. The higher the number, the higher the priority associated with it. The inserted sequence 0 is the lowest priority.

The channels in inserted sequence can be converted only once, while the routine sequence channels can be converted continuously a loop. An inserted sequence can suspend the routine sequence which is converting.

When the routine sequence channel is converting, the trigger of the inserted sequence m (m = 0... 5) arrives, the routine sequence conversion will be suspended and the inserted sequence m (m = 0... 5) channels will begin the conversion. After all channels of the inserted sequence m (m = 0... 5) are converted once, the routine sequence will resume the conversion. By configuring the RESTART bit in the ADC_RCTL register, the routine sequence will resume the sequence conversion from the suspended channel (RRESTART =1'b0) or restart the sequence conversion from the first channel (RRESTART =1'b1).

Figure 13-2. Restart procedure, RRESTART=1, RL=3, IL=1

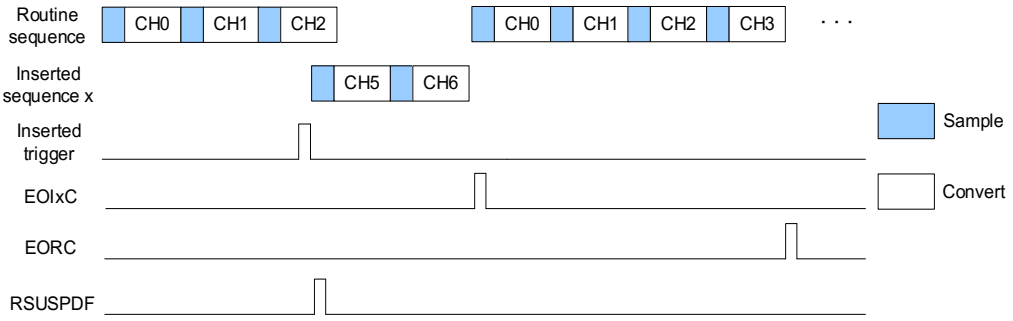
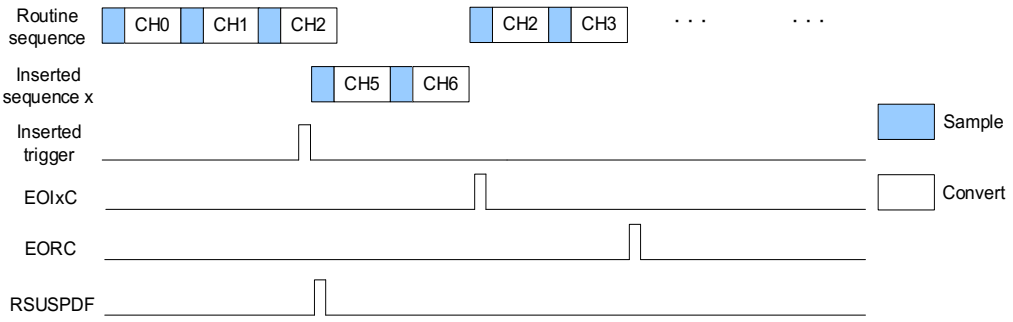


Figure 13-3. Resume procedure, RRESTART=0, RL=3, IL=1



The inserted sequence with higher priority can interrupt the inserted sequence with lower priority. For example, the inserted sequence 5 can interrupt any one sequence of inserted sequences 0 to inserted sequence 4.

The routine sequence and inserted sequence m (m = 0…4) can be suspend by higher priority sequences. There are corresponding suspended flags to check whether the sequence is suspended or not. And there also have the RRESTART/IRESTART bits to configure the restart modes.

13.4.5. Operation mode

Routine sequence operation mode

In the routine sequence, a sequence of up to 28 conversions can be organized in a specific sequence. There are two modes for routine sequence conversion, and the mode can be selected by the RMODE bit in ADC_CTL register.

The channels are configured by bits RSQ[5:0] in ADC_RCFGx (x = 0…27) registers, and the total conversion length is specified by RL[4:0] bit-filed in ADC_RCTL register.

When RMODE bit is 1'b0, the routine sequence works in one-shot mode, all routine channels can be converted only once.

When RMODE bit is 1'b1, the routine sequence works in continuous mode, all routine channels are continuously performed. This sequence repeats itself until the RMODE field is reset to 0 by software.

After the sequence conversion is completed, the end of sequence conversion flag (EORC bit) is set and an interrupt will be generated if the EORCIE bit (in the ADC_RCTL register) is set.

At the end of each channel conversion, the end of channel conversion flag (EORCHC bit) is set to 1 and an interrupt will be generated if the RECHIE bit (in the ADC_RCFGx (x = 0…27) register) is set to 1.

If the RL[4:0] bit-field is 5'b00000 and there is only 1 channel in the routine sequence, after the sequence conversion is completed, the end of sequence conversion flag EORC bit and the end of channel conversion flag EORCHC bit are set at the same time.

Inserted sequence operation mode

In inserted sequence, a sequence of up to 4 conversions can be organized in a specific sequence. The ISQ[5:0] bit-field in the ADC_ImCFGx (m = 0...5, x = 0...3) register is used to select the channels. The IL[1:0] bit-filed in the (ADC_ImCTL) (m = 0...5) register is used to configure the total conversion length.

The inserted sequence works in one-shot mode, and all inserted channels can be converted only once.

The end of sequence conversion flag (EOImC bit, m = 0…5) and the end of channel conversion flag (EOImCHC bit, m = 0…5) are as same as the routine sequence.

Stream conversion mode

Both routine and inserted sequences support stream conversion mode. The following describes the routine sequence as an example, and the inserted sequence is similar to the routine sequence.

The stream conversion mode of routine sequence is configured in ADC_RSTREAM register and can work in both one-shot and continuous modes. RSTREAMEN bit is used to enable the stream conversion mode, RSTREAMCNT[15:0] bit-field specify how many sequence conversions repeat (this bit-fields must be configured before the RSTREAMEN bit is enabled). The repeat number is equal to (RSTREAMCNT[15:0] + 1).

After routine sequence has completed the (RSTREAMCNT[15:0] + 1) times conversion, the flag RSTREAMIF is set to 1, and an interrupt is generated if an interrupt is enabled (if the RSTREAMIE bit is enabled).

When RSTREAMAREN = 1'b1, the RSTREAMCNT[15:0] bit-field can be reloaded automatically by hardware with the previously programmed value after the routine sequence has completed the set number of samples and conversions.

If the RSTREAMCNT[15:0] bit-field is 0, the routine sequence will convert once, and after this sequence conversion is completed, the RSTREAMIF bit will be set.

13.4.6. Abort a sequence

Setting the RABORT/IABORT bit (in ADC_RCTL/ADC_ImCTL (m = 0...5) registers) to 1 can abort an ongoing channel conversion in sequences. The current channel conversion is aborted and all conversions of the corresponding sequence will be ended. During an abort operation, the RABORT/IABORT field is reset to zero when the channel conversion is aborted. The EORC/EOImC (m = 0...5) flag to the aborted sequence is not generated. This behavior applies to routine and 6 inserted sequences.

The behavior of aborting routine sequence depends on the value of RMODE bit. When RMODE bit is 1'b0, the routine sequence works in one-shot mode, the routine sequence conversions are immediately aborted; When RMODE bit is 1'b1, the routine sequence works in continuous mode, the current routine sequence conversions are aborted, and the new

routine sequence conversions are started (start from the first channel). The EORC flag of the current routine sequence conversion is not generated.

When routine sequence works in continuous mode (RMODE = 1'b1), the method, which is configuring the RMODE bit to 0 and then setting RABORT bit to 1, could abort the conversions immediately.

Figure 13-4. Abort the routine sequence, RMODE=0

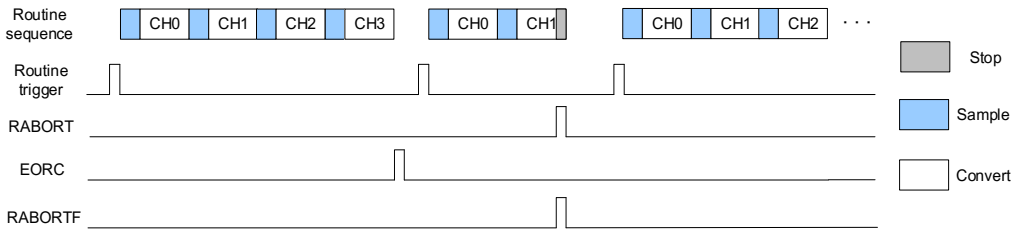
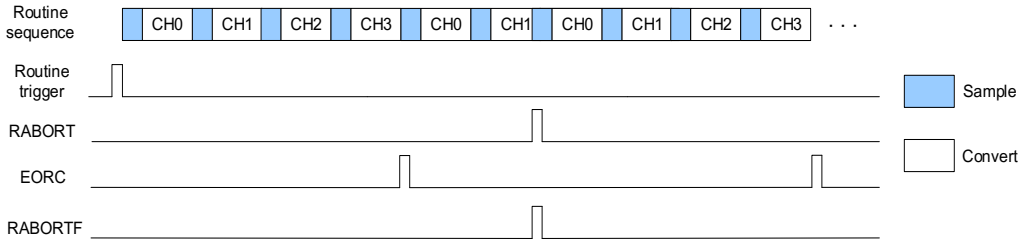


Figure 13-5. Abort the routine sequence, RMODE=1



When the IABORT bit in ADC_ImCTL (m = 0...5) register is set to 1, the inserted sequence m which is already suspended a lower priority sequence (an inserted sequence or routine sequence) will be aborted. And the suspended sequence will resume the conversion. In this case, the end of sequence conversion flag EOImC is not generated.

Every inserted sequence and routine sequence have individual RABORT/IABORT bit (in ADC_RCTL/ADC_ImCTL (m = 0...5) registers) and RABORTF/ImABORTF (m = 0...5) bit (in ADC_ESTAT register).

13.4.7. Programmable presample and sample time

The presampling function precharges or discharges the ADC's internal sample capacitor node before the analog input channel n (n = 0...55, presampling does not apply for internal channels) starts sampling.

The presampling function is implemented by inserting the presampling process during the normal sampling process. During presampling process, ADC samples the internal reference voltage (V_{REFP} or V_{REFN}); And during the sampling process, ADC samples analog signals coming from external channels.

Setting the PSMPEN bit in ADC_CHCFGn (n = 0...55) registers to 1 can enable the presampling function of channel n (n = 0...55). The sampling steps of the channel n (n = 0...55) are: presampling-> sampling -> conversion. The presampling time is programmed in

PSMP[9:0] bit-field in the ADC_PSMPCFG register.

Setting the DISSMP bit in ADC_CHCFGn (n = 0...55)) registers to 1 can disable the sampling process of channel n (n = 0...55). In this case, only the internal reference voltage is sampled and converted in presampling and converting process, and the conversion data is stored in the corresponding data register.

Figure 13-6. Presampling and sampling sequence, PSMPEN = 1, DISSMP=0

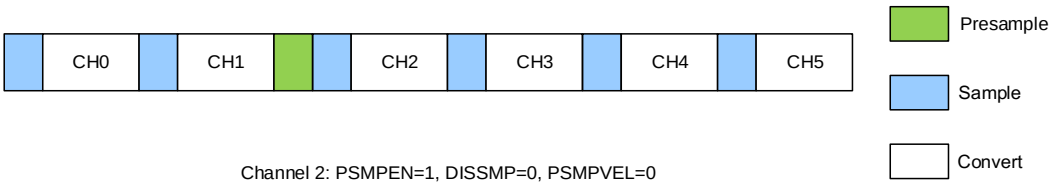
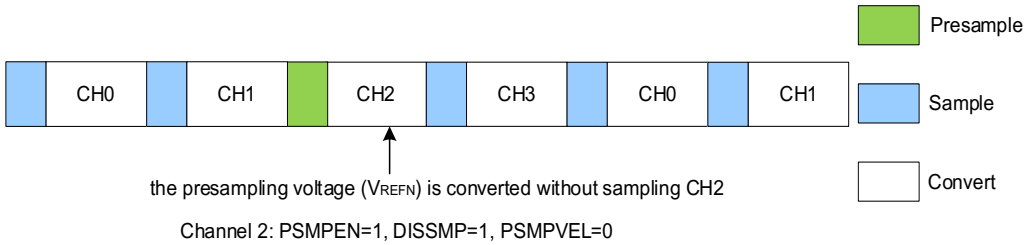


Figure 13-7. Presampling and sampling sequence, PSMPEN = 1, DISSMP=1



13.4.8. Analog multiplexer channel

ADC0 and ADC1 both have 4 external extended channels, and these four channels are used to connect the 32 external channels. Each external extended channel uses the external analog multiplexer to select one channel from 8 channels.

Setting the EDDLY[15:0] bit-field in the ADC_EDDLY register, can configure the delay time between the external analog multiplexers decode signals and the actual start of channel conversion. If presampling mode is enabled, the delay time of the EDDLY[15:0] bit-field starts after the presample phase.

For example, to use channel 27 of ADC0, the following configuration is required: Configure the corresponding pins of ADC0_EECSEL0, ADC0_EECSEL1, and ADC0_EECSEL2 to the corresponding alternate function (AF) configuration (set CTLx[1:0] = 2'b10 in the GPIOx_CTL register). Configure the alternate function of the three corresponding pins to the respective function bits (set SELx[3:0] in the GPIOx_AFSELx register). Configure the three corresponding pins to output push-pull mode (set OMx = 1'b0 in the GPIOx_OMODE register).

The remaining configurations are the same as for normal channels.

Channel 27 of the ADC corresponds to the fourth channel of the ADC_EECIN0 pin. Therefore, the ADC0_EECSEL2 pin outputs a low level, while the ADC0_EECSEL1 outputs high level and ADC0_EECSEL0 pins outputs low levels.

Figure 13-8. Schematic diagram of analog multiplexing channel EECIN0

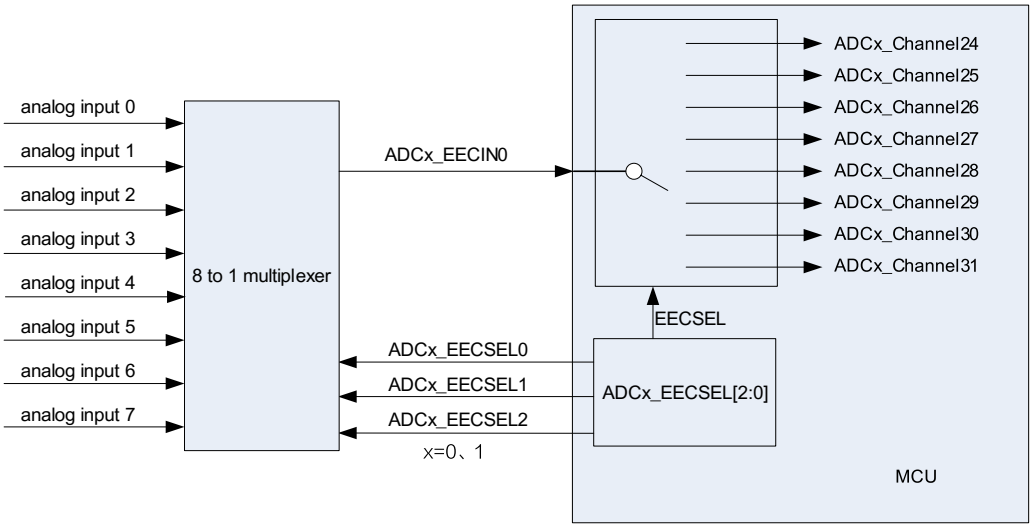


Table 13-4. ADC Analog multiplexing channel signal

ADC channel	ADC pin / input	ADC0_EECSEL0	ADC0_EECSEL1	ADC0_EECSEL2
ADC channel 24	ADC_EECIN0	0	0	0
ADC channel 25		1	0	0
ADC channel 26		0	1	0
ADC channel 27		1	1	0
ADC channel 28		0	0	1
ADC channel 29		1	0	1
ADC channel 30		0	1	1
ADC channel 31		1	1	1
ADC channel 32	ADC_EECIN1	0	0	0
ADC channel 33		1	0	0
ADC channel 34		0	1	0
ADC channel 35		1	1	0
ADC channel 36		0	0	1
ADC channel 37		1	0	1
ADC channel 38		0	1	1
ADC channel 39		1	1	1
ADC channel 40	ADC_EECIN2	0	0	0

ADC channel	ADC pin / input	ADC0_EECSEL0	ADC0_EECSEL1	ADC0_EECSEL2
ADC channel 41		1	0	0
ADC channel 42		0	1	0
ADC channel 43		1	1	0
ADC channel 44		0	0	1
ADC channel 45		1	0	1
ADC channel 46		0	1	1
ADC channel 47		1	1	1
ADC channel 48	ADC_EECIN3	0	0	0
ADC channel 49		1	0	0
ADC channel 50		0	1	0
ADC channel 51		1	1	0
ADC channel 52		0	0	1
ADC channel 53		1	0	1
ADC channel 54		0	1	1
ADC channel 55		1	1	1

13.4.9. Analog multiplexer decode signals delay

ADC0 and ADC1 both have 4 external extended channels, and these four channels are used to connect the 32 external channels. Each external extended channel uses the external analog multiplexer to select one channel from 8 channels.

Setting the EDDLY[15:0] bit-field in the ADC_EDDLY register, can configure the delay time between the external analog multiplexers decode signals and the actual start of channel conversion. If presampling mode is enabled, the delay time of the EDDLY[15:0] bit-field starts after the presample phase.

13.4.10. Conversion start delay

Setting the CSDEN[6:0] bit-field in the ADC_CSDLY register, can enable conversion start delay function of the sequence. A delay time between the trigger signal and the actual start of channel conversion is configured by the CSDLY[15:0] bit-field in the ADC_CSDLY register. If conversion start delay function of the sequence is enabled, and a trigger is coming to ADC, the ADC will delay this trigger with time of the CSDLY[15:0] bit-field to start conversion.

13.4.11. Analog watchdog

ADC can have up to 4 analog watchdogs, and each watchdog has one watchdog threshold register x (ADC_WDTx, x = 0...3). The threshold values (high threshold and low threshold) for each channel can be selected independently in WDHT[11:0] bit-field and WDLT[11:0] bit-field.

The analog watchdog x (x = 0...3) are very flexible, can configure the watchdog x (x = 0...3) of channels by programming WDSEL[1:0] bit-field in ADC_CHCFGn (n = 0….61) registers.

Every watchdog has 6 watchdog modes, and these modes can be configured in WDMSEL[2:0] bit-field in ADC_CHCFGn (n = 0….61) registers.

For example, Result is the conversion result of ADC channel n (n = 0...61), WDHT is the watchdog high threshold, and WDLT is the low threshold. Six analog watchdog modes are:

- Result <= WDLT;
- WDLT < Result <= WDHT;
- Result > WDHT;
- Result > WDLT;
- Result > WDHT or Result <= WDLT;
- Result <= WDHT.

When the selected analog watchdog mode event occurs, the channel analog watchdog flags will set in WDIFy[31:0] bit-field of ADC_WDINTFy (y = 0, 1) registers and the WDE bit of ADC_ESTAT register will set too. An interrupt of the analog watchdog x (x = 0...3) will be generated if the WDEIE bit in ADC_WDTx (x = 0…3) register is set.

13.4.12. Data management

ADC conversion results access

Users can access ADC results using interrupt or DMA requests. Interrupt requests are based on the EORCHC or EOImCHC (m = 0…5) flags.

Every channel has corresponding channel data registers (ADC_RCHDATAx (x = 0…27) and ADC_ImCHDATAx (m = 0…5, x = 0…3) registers) to save conversion result. And per sequence has individual sequence data registers (ADC_RDATA and ADC_ImDATA (m = 0…5) registers) to save conversion result.

The channel data register corresponds one-to-one to the conversion sequence; every completed conversion result can be stored into a corresponding channel data register. ADC_RCHDATAx (x = 0…27) and ADC_ImCHDATAx (m = 0…5, x = 0…3) registers provide the following information:

- The channel conversion results are stored in the corresponding bit-field according to the conversion sequence: RCHDATA[15:0] and ICHDATA[15:0] bit-field;
- The channel that was used for the conversion sequence: RCH[5:0] and ICH[5:0] bit-field;

- The valid flag of result: RCHAVLB and ICHAVLB bit, the flag will be cleared if data is read;
- The overwrite flag: RCHOVWF and ICHOVWF bit. The overwrite flag will be set if RCHDATA[15:0] and ICHDATA[15:0] bit-field are valid and a new conversion result need to be record in RCHDATA[15:0] and ICHDATA[15:0] bit-field (if OVWEN bit in the ADC_CTL register is set to 1). An interrupt of the channel overwrite will be generated if the CHOVWIE bit in ADC_CTL register is set.

The channel conversion data can also be stored in the sequence data registers (ADC_RDATA and ADC_ImDATA (m = 0...5) registers). Every conversion channel can route the conversion result to an available sequence data registers based on sequence. Each sequence data register has a buffer register to record conversion result, and they can store up to two valid conversion data. ADC_RDATA and ADC_ImDATA (m = 0...5) registers provide the following information:

- Each channel conversion result data of sequence: RDATA[15:0] and IDATA[15:0] bit-field;
- Channel number of corresponding conversion: RSQCH[5:0] and ISQCH[5:0] bit-field;
- The valid flag of result: RAVLB and IAVLB bit, the valid value will be cleared if data is read.
- The overwrite flag: ROVWF and IOVWF bit, the overwrite flag will be set if RDATA[15:0] and IDATA[15:0] bit-field and their buffer registers are all valid and a new conversion result need to be record in RDATA[15:0] and IDATA[15:0] bit-field (if OVWEN bit in the ADC_CTL register is set to 1). An interrupt of the sequence overwrite will be generated if the OVWIE bit in ADC_CTL register is set.

Gain and offset

ADCs support both gain and offset calculation, when these two functions are enabled at the same time, the gain is calculated firstly, and then calculate the offset.

Gain and offset function chips are not configured at the factory; customers need to configure them themselves if high-precision sampling is required.

The analog watchdog function is performed after offset and gain calculation with unsigned values. For correct watchdog operation, the data after offset calculation must be in unsigned format.

Gain

When the GAINEN bit is set in the ADC_CTL register, the gain is activated on all ADC converted data. After each conversion, calculate the data using the following formula:

$$DATA(\text{new ADC data}) = DATA(\text{original ADC data}) \times (GAIN / 4096) \quad (13\text{-}1)$$

The programmable range of GAIN is 0 to 16383, and the actual gain factor (GAIN / 4096) is 0 to 3.999756.

Offset

When the OFFEN bit is set in the ADC_OFFCFGx (x = 0…3) register, the offset calculation is activated on all converted data. After each conversion, calculate the data using the following formula:

DATA(new ADC data) = DATA (original ADC data) - offset value (13-2)

Table 13-5. Algorithm between raw converted data and offset describes the relationship between raw data and offset with different resolutions.

Table 13-5. Algorithm between raw converted data and offset

DRES [1:0] bits	Algorithm between raw converted data and offset		Result	Remarks
	Raw result converted data, left aligned	Offset		
00	DATA[11:0]	OFFSET[11:0]	12-bit data	-
01	DATA[11:2], DATA[1:0] = 2'b00	OFFSET[11:0]	10-bit data	The user must configure OFFSET[1:0] to 00.
10	DATA[11:4], DATA[3:0] = 4'b0000	OFFSET[11:0]	8-bit data	The user must configure OFFSET[3:0] to 0000.
11	DATA[11:6], DATA[5:0] = 6'b000000	OFFSET[11:0]	6-bit data	The user must configure OFFSET[5:0] to 000000.

Date alignment

The alignment of data stored after conversion can be specified by DAL bit in the ADC_CTL register. The routine and inserted sequence both support alignment function.

Figure 13-9. 12-bit Data storage mode

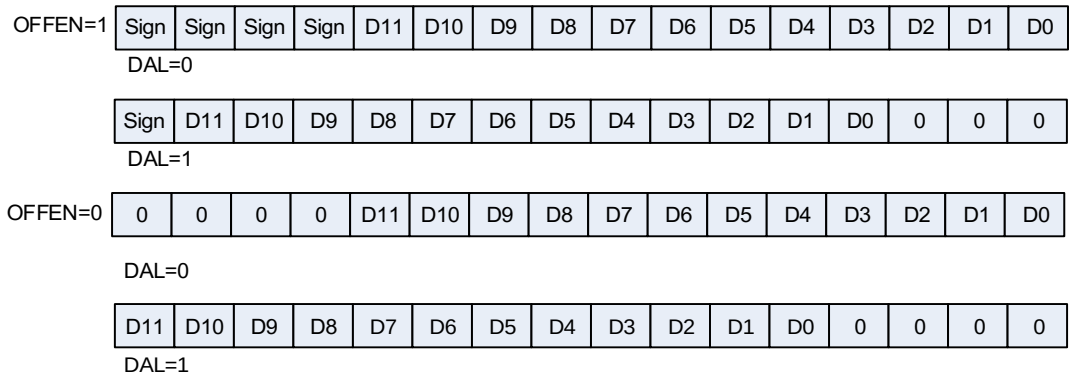
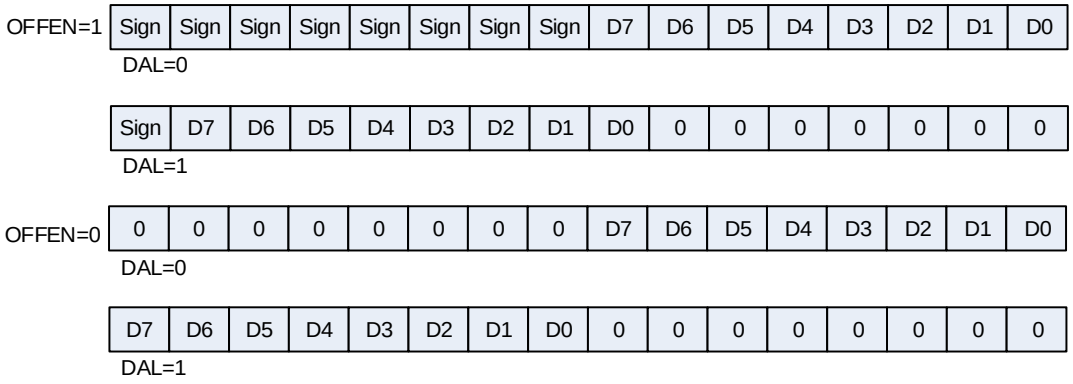


Figure 13-10. 10-bit Data storage mode

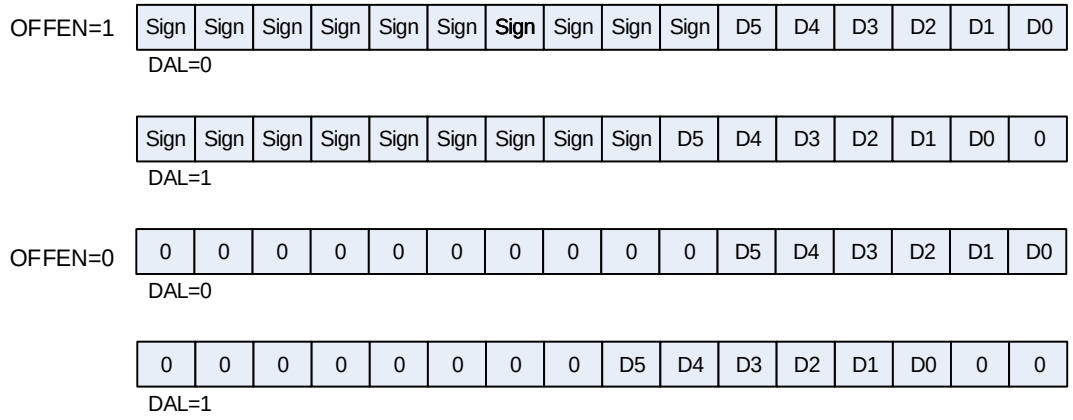


Figure 13-11. 8-bit Data storage mode



6-bit resolution data alignment is different from 12-bit / 10-bit / 8-bit resolution data alignment, shown as ***Figure 13-12. 6-bit Data storage mode***.

Figure 13-12. 6-bit Data storage mode



Average

The on-chip hardware average circuit performs data preprocessing to offload the CPU. When the AVRGEN bit (in the ADC_CTL register) is set to 1, the average function is enabled. It can handle multiple conversions and average them into a single data with increased data width, up to 256 conversions are averaged to result register or their FIFO register.

$$Result = \frac{1}{N} * \sum_{n=0}^{N-1} D_{out}(n) \quad (13\text{-}3)$$

The on-chip hardware average circuit performs the following functions: summing and averaging. The averaging ratio N is defined by the AVRGSEL[2:0] bits in the ADC_CTL register.

13.4.13. Trigger source

The conversion of routine or inserted sequence can be triggered by TRIGSEL or software. The trigger source of routine sequence is controlled by the ETMRC[1:0] bits in the ADC_RCTL register, and the external trigger source of inserted sequence is controlled by the ETMIC[1:0] bits in the ADC_ImCTL (m = 0...5) register

ETMRC[1:0] and ETMIC[1:0] control bits are used to specify, which event of two events can trigger conversion for the routine and inserted channels.

Table 13-6. Trigger source for routine channels

ETMRC[1:0]	Trigger Source	Trigger Type
01、10、11	TRIGSEL	Signal from TRIGSEL
00	RSWCST	Software trigger

Table 13-7. Trigger source for inserted channels

ETMIC[1:0]	Trigger Source	Trigger Type
01、10、11	TRIGSEL	Signal from TRIGSEL
00	ISWCST	Software trigger

13.4.14. DMA request

Conversion result data from any channel or sequence can be transferred from a register to system memory via Direct Memory Access (DMA). DMA transfers function is enabled by setting the DMA bit in ADC_CTL register to 1.

There are two DMA transfer methods to choose:

1) Transmission by channel: the on-chip DMA controller can get a DMA request after the conversion of every channel in the sequence by setting the RCHDMAEN bit in ADC_RCFGx (x = 0...27) registers or the ICHDMAEN bit in ADC_ImCFGx (m = 0...5, x = 0...3) registers. The data of ADC_RCHDATAx (x = 0…27) and ADC_ImCHDATAx (m = 0…5, x = 0…3) registers can be transferred by DMA interface.

2) Transmission by sequence: the on-chip DMA controller can get a DMA request after the conversion of every channel in the sequence by setting the RDMAEN bit in ADC_RDATA register or the IDMAEN bits in ADC_ImDATA (m = 0…5) registers. The data of ADC_RDATA and ADC_ImDATA (m = 0…5) registers can be transferred by DMA interface.

The corresponding DMA enable bits must be initialized before starting any conversion.

13.4.15. Programmable resolution - fast conversion mode

It is possible to obtain faster conversion time (t_{ADC}) by reducing the ADC resolution.

The resolution can be configured to be either 12, 10, 8 or 6 bits by programming the DRES[1:0] bits in the ADC_CTL register.

Lower resolution allows faster conversion time for applications where high data precision is not required. The DRES[1:0] bits must only be changed when the ADCON bit is reset. Lower resolution reduces the conversion time needed for the successive approximation steps as shown in *Table 13-8. tCONV timings depending on resolution*.

Table 13-8. t_{CONV} timings depending on resolution

DRES[1:0] bits	t_{CONV} (ADC clock cycles)	t_{CONV}(ns) at f_{ADC} = 16MHz	t_{SMPL}(min) (ADC clock cycles)	t_{ADC} (ADC clock cycles)	t_{ADC}(ns) at f_{ADC} = 16MHz
00	12.5	781.3ns	2.5	15	937.5ns
01	10.5	656.3ns	2.5	13	812.5ns
10	8.5	531.3ns	2.5	11	687.5ns
11	6.5	406.3ns	2.5	9	562.5ns

13.4.16. ADC internal channels

When the HPTSVEN bit in ADC_CTL register is set, the high-precision temperature sensor channel (channel 59 for ADC0 and ADC1) is enabled. The temperature sensor can be used to measure the ambient temperature of the device. The sensor output voltage can be converted into a digital value by ADC. The sampling time for the temperature sensor is recommended to be set to at least 10μs. When this sensor is not in use, it can be set in power down mode by resetting the HPTSVEN bit.

The output voltage of the temperature sensor changes linearly with temperature. Due to the diversification of production process, the deviation of the temperature change curve is different on chip to chip (refer to the device datasheet for more information).

To use the temperature sensor:

1. Configure the ADC clock.
2. Configure the conversion sequence (channel 59 for ADC0 and ADC1) and the sampling time (at least 10μs) for the channel.
3. Enable the temperature sensor by setting the HPTSVEN bit in the ADC control register (ADC_CTL).
4. Start the ADC conversion by setting the ADCON bit (or by trigger).
5. Read the resulting temperature data($V_{temperature}$) in the ADC data register, and get the temperature using the following formula:

$$\text{Temperature (°C)} = ((V_{temperature} - V_{25}) / \text{Avg_Slope}) + 25 \quad (14\text{-}4)$$

$V_{temperature}$: The output voltage of temperature sensor.

V_{25}: The output voltage of temperature sensor at 25°C. The ADC conversion results of the

temperature sensor corresponding to 25°C were calibrated during chip production(VDDA=5.0V). The factory-calibrated offset value is stored in the read-only area of flash, and refers to electrical characteristics section of device datasheet.

Avg_Slope: Average slope for curve between temperatures vs. the output voltage of temperature sensor, refers to electrical characteristics section of device datasheet.

Note:

1) After the temperature sensor is enabled, it is necessary to wait for at least 3 ADC sampling cycles before the ADC conversion code value is considered valid, and the first 3 conversion data should be discarded;
2) The sampling accuracy of temperature sensor can be improved by means of hardware on-chip over-sampling or software averaging.

When the INREFEN bit in ADC_CTL register is set, the VREFINT channel (channel 58 for ADC0 and ADC1) is enabled. The internal reference voltage (V_{REFINT}) provides a stable (bandgap) voltage output for the ADC and comparators. V_{REFINT} is internally connected to the ADC input channel.

13.5. ADC sync mode

In devices with two ADCs, ADC sync mode can be used.

In ADC sync mode, the conversion of ADC1 are synchronized by the triggers of ADC0. The two or three ADCs convert parallelly or rotately, according to the mode selected by the SYNCM[3:0] bits in ADC_SYNCCTL register.

In ADC sync mode, when the conversion is configured to be triggered by an external event, the external trigger must be disabled for ADC1.

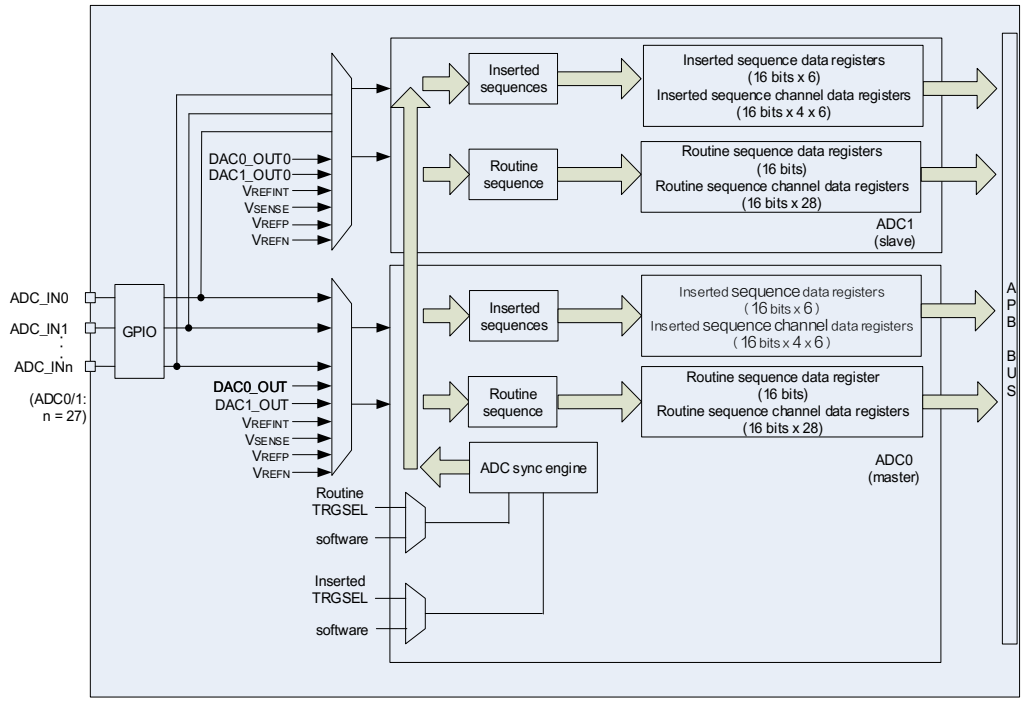
The following modes can be configured:

- Free mode.
- Routine parallel mode.
- Inserted parallel mode.
- Follow-up mode.
- Trigger rotation mode.
- Routine parallel mode & inserted parallel mode.
- Routine parallel mode & trigger rotation mode.

When the ADCs are in a sync mode other than free mode, they should be configured to free mode before being configured to another sync mode.

The ADC sync scheme is shown in ***Figure 13-13. ADC sync block diagram***.

Figure 13-13. ADC sync block diagram



13.5.1. Free mode

In this mode, the ADC synchronization is bypassed, and each ADC works freely.

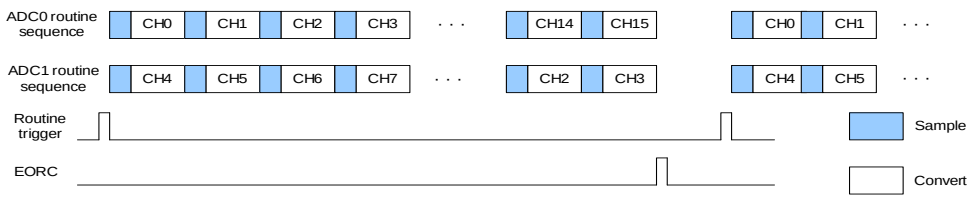
13.5.2. Routine parallel mode

The routine parallel mode is enabled by setting the SYNCM[3:0] bits in the ADC_SYNCCTL register to 4'b0110. In the routine parallel mode, all ADCs convert the routine sequence channels parallelly at the selected external trigger of ADC0. The trigger is selected by configuring the ETMRC[1:0] bit-field in the ADC_RCTL register of ADC0.

EORC interrupts (if enabled on the ADC interfaces) are generated at the end of routine sequences.

The converted data is stored in the ADC_RDATA registers and ADC_RCHDATAx (x = 0…27) registers of each ADC interface. The behavior of routine parallel mode is shown in the ***Figure 13-14. Routine parallel mode on 16 channels***.

Figure 13-14. Routine parallel mode on 16 channels



Note:

1. Do not convert the same channel on the two ADCs at a given time (no overlapping sampling times for the ADCs when converting the same channel).
2. Make sure to trigger the ADCs when none of them is converting (do not trigger ADC0 when some of the conversions are not finished).

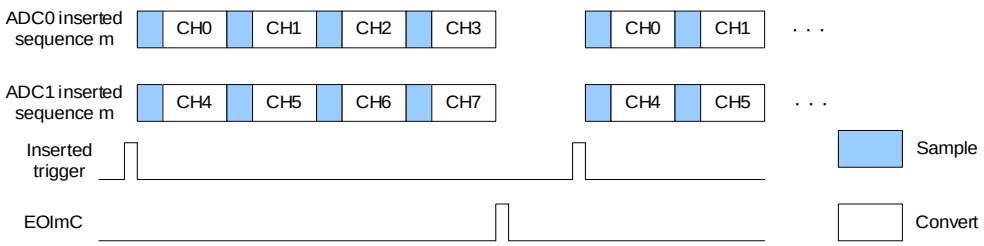
13.5.3. Inserted parallel mode

The inserted parallel mode is enabled by setting the SYNCM[3:0] bits in the ADC_SYNCCTL register to 4'b0101. In the inserted parallel mode, all ADCs convert the inserted sequence channels parallelly at the selected external trigger of ADC0. The trigger is selected by configuring the ETMIC[1:0] bits in the ADC_ImCTL (m = 0...5) registers of ADC0.

EOImC (m = 0...5) interrupts (if enabled on the ADC interfaces) are generated at the end of the inserted sequences.

The converted data is stored in the ADC_ImDATA (m = 0...5) registers and ADC_ImCHDATAx (m = 0…5, x = 0…3) of each ADC interface. The behavior of inserted parallel mode is shown in the *Figure 13-15. Inserted parallel mode on 4 channels*.

Figure 13-15. Inserted parallel mode on 4 channels



Note:

1. Do not convert the same channel on two ADCs at a given time (no overlapping sampling times for the two ADCs when converting the same channel).
2. Make sure to trigger the ADCs when none of them is converting (do not trigger ADC0 when some of the conversions are not finished).

13.5.4. Follow-up mode

The follow-up mode is enabled by setting the SYNCM[3:0] bits in the ADC_SYNCCTL register to 4'b0111. In the follow-up mode, ADC0 converts the routine sequence at the selected external trigger. The trigger is selected by configuring the ETMRC[1:0] bits in the ADC_RCTL register of ADC0. After a delay time, ADC1 converts the routine sequence. The routine sequence in above descriptions only includes one channel.

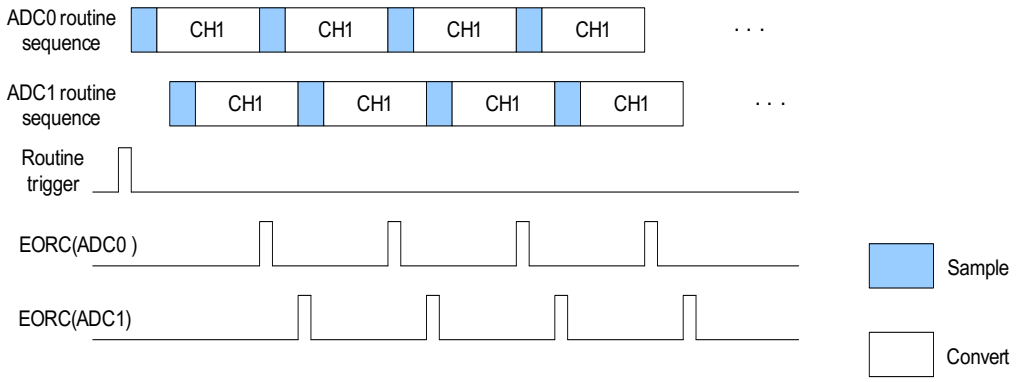
The delay time is calculated from the end of the previous sequence sampling. To prevent more than one ADCs from sampling the same channel at a given time, the delay time between

two consecutive sample phase is 2 CK_ADC cycles.

If the RMODE bit in ADC_RCTL register is set, the selected routine sequences are continuously converted. EORC interrupts (if enabled on the ADC interfaces) are generated at the end of the routine sequences.

The converted data are stored in the ADC_RDATA registers and ADC_RCHDATAx (x = 0…27) registers of each ADC interface. The behavior of follow-up mode is shown in the *Figure 13-16. Follow-up mode on 1 channel in continuous mode*.

Figure 13-16. Follow-up mode on 1 channel in continuous mode



Note: Make sure to trigger the ADCs when none of them is converting (do not trigger ADC0 when some of the conversions are not finished).

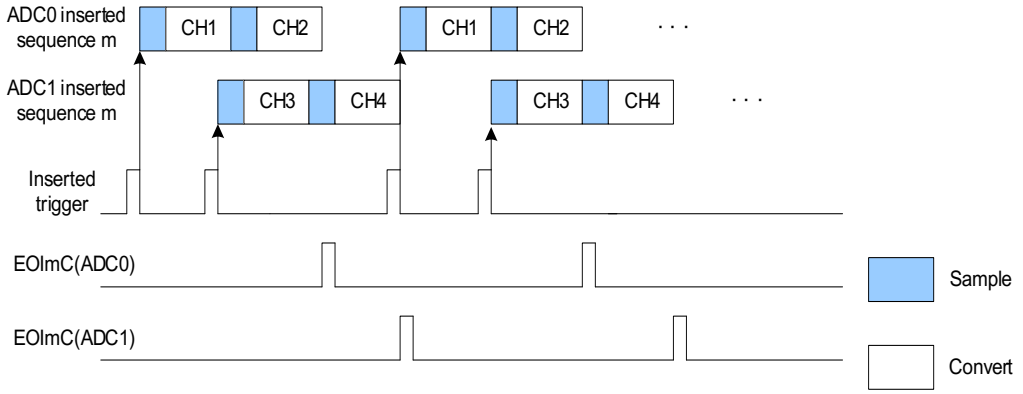
13.5.5. Trigger rotation mode

The trigger rotation mode is enabled by setting the SYNCM[3:0] bits in the ADC_SYNCCTL register to 4'b1001. In the trigger rotation mode, the inserted sequences of the ADCs are triggered in turn by the selected external trigger of ADC0. The trigger is selected by configuring the ETMIC[1:0] bits in the ADC_ImCTL (m = 0...5) registers of ADC0.

When SYNCM[3:0] = 5'b01001, the selected external trigger triggers the ADCs in the sequence: ADC0 -> ADC1 -> ADC0 -> ADC1, and etc.

EOImC (m = 0...5) interrupts (if enabled on the ADC interfaces) are generated at the end of the inserted sequences. The behavior of trigger rotation mode is shown in the *Figure 13-17. Trigger rotation: inserted sequence*.

Figure 13-17. Trigger rotation: inserted sequence



Note: Do not convert the same channel on two ADCs (no overlapping sampling times for the two ADCs when converting the same channel).

13.5.6. Combined routine parallel & inserted parallel mode

The combined routine parallel and inserted parallel mode is enabled by setting the SYNCM[3:0] bits in the ADC_SYNCCTL register to 4'b0001. In the combined routine parallel and inserted parallel mode, it is also possible to interrupt parallel conversion of the routine sequence to inserted parallel conversion of an inserted sequence.

EORC interrupts (if enabled on the ADC interfaces) are generated at the end of routine sequences. EOImC (m = 0...5) interrupts (if enabled on the ADC interfaces) are generated at the end of the inserted sequences.

Note:

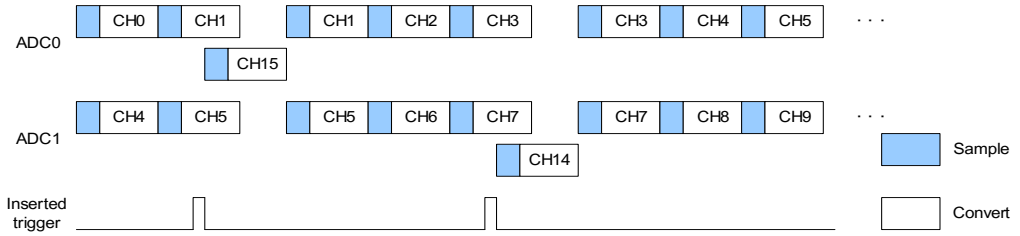
1. Do not convert the same channel on two ADCs at a given time (no overlapping sampling times for the two ADCs when converting the same channel).
2. The inserted sequence length of the ADCs should be configured the same.

13.5.7. Combined routine parallel & trigger rotation mode

The combined routine parallel and trigger rotation mode is enabled by setting the SYNCM[3:0] bits in the ADC_SYNCCTL register to 4'b0010. In the combined routine parallel and trigger rotation mode, it is possible to interrupt routine sequence parallel conversion by triggering conversion of an inserted sequence rotately. When the inserted trigger occurs, the inserted conversion is immediately started. When the routine sequence conversions are interrupted, the routine sequence conversion is stopped at the inserted trigger and resumed parallelly at the end of the inserted sequence conversion. The behavior is shown in ***Figure 13-18. Combined routine parallel & trigger rotation mode: SYNCM = 4'b0010***.

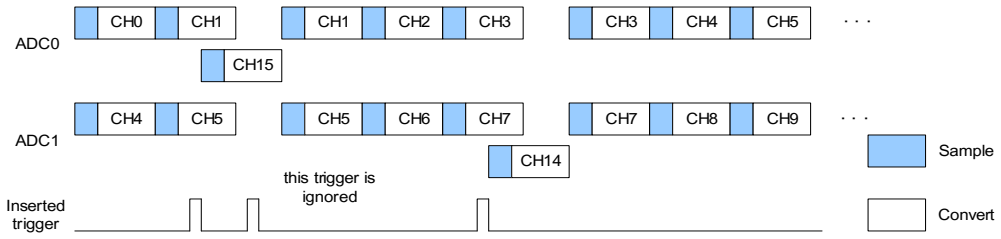
EORC interrupts (if enabled on the ADC interfaces) are generated at the end of routine sequence. EOImC (m = 0...5) interrupts (if enabled on the ADC interfaces) are generated at the end of the inserted sequences.

Figure 13-18. Combined routine parallel & trigger rotation mode: SYNCM = 4'b0010



If another inserted sequence trigger occurs during an inserted sequence conversion, the latter trigger will be ignored, as shown in ***Figure 13-19. Trigger occurs during inserted sequence conversion: SYNCM = 4'b0010***.

Figure 13-19. Trigger occurs during inserted sequence conversion: SYNCM = 4'b0010



Note:

1. Do not convert the same channel on two ADCs at a given time (no overlapping sampling times for the two ADCs when converting the same channel).
2. The inserted sequence length of the ADCs should be configured the same.

13.6. Self test

For the safe operation of ADC, the ADC has a self test function to regularly check whether the running environment of ADC is normal. The self test function is enabled by setting the STEN bit in the ADC_STCFG register to 1, when the STEN bit is set, the ADC detects whether an error event has occurred automatically, and if so, the corresponding error flag bit is set.

13.6.1. Self test modes

ADC supports three self test modes: supply self test, capacitive self test and supply + capacitive self test. The modes can be configured in STMSEL[1: 0] bit-field in ADC_STCFG register. When the self test function is enabled, the ADC will to check the internal reference voltages (V_{REFN}, V_{REFINT} and V_{REFP}) and internal sample capacitor values.

Supply self test

The supply self test mode is divided into three steps from step 0 to step 2, which are the samples and conversions of V_{REFN}, V_{REFINT} and V_{REFP}. In continuous conversion mode, these

three steps must be executed one after another without any other conversions in between. Only in oneshot conversion mode is the self-test step allowed to be configured, as follows:

- Step 0: Negative reference voltage (V_{REFN}).
- Step 1: Internal reference voltage (V_{REFINT}).
- Step 2: Positive reference voltage (V_{REFP}).

Capacitive self test

The capacitive self test mode is divided into 8 steps from step 0 to step 7, which are the samples and conversions of the capacitance matrix. Each step of capacitive self test takes up to 0.938μs with 16MHz ADC clock frequency, the specific step voltages are as follows:

- Step 0: $V_{REFP}/2^1$;
- Step 1: $V_{REFP}/2^2$;
- Step 2: $V_{REFP}/2^3$;
- Step 3: $V_{REFP}/2^4$;
- Step 4: $V_{REFP}/2^5$;
- Step 5: $V_{REFP}/2^6$;
- Step 6: $V_{REFP}/2^7$;
- Step 7: $V_{REFP}/2^8$.

The expected conversion result of capacitive self test is zero. If ADC conversion result is higher than 0, the corresponding error flags which indicate capacitive self test error occurred are set, then ADC should be re-calibrated for handling the error. If the error still exists after re-calibration, ADC maybe has suffered a permanent damage.

Supply + capacitive self test

In this mode, the supply self test is performed first (from step 0 to step 2), and then the capacitive self test (from step 3 to step 10). The operation is exactly the same as if they are run separately.

13.6.2. Self test in routine sequence

Self test conversions are similar to routine sequence conversions and only operate in routine sequence. Self test conversions are performed at the end of routine sequence. The operation mode of the routine sequence determines the conversion way of the self test steps.

In one-shot mode

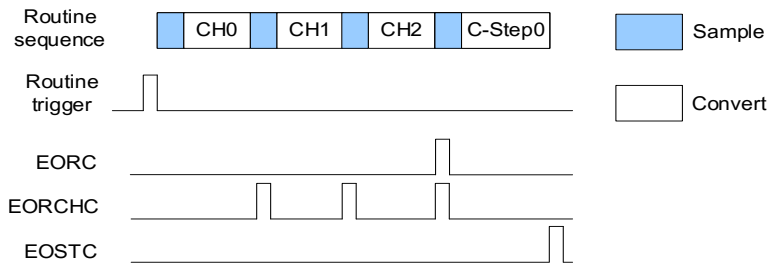
When self test function is enabled, only one self test step is performed at the end of routine sequence conversion. Self test mode and self test steps can be selected in ADC_STCFG register.

Therefore, the configuration procedure for routine sequence with self test conversion in one-shot mode is as follows:

1. Configure RL[4:0] bit-field in ADC_RCTL register and RSQ[5:0] bit-field in ADC_RCFGx (x=0...27) registers to select the sequence length and channels to converted for routine sequence.
2. Set MODE=0 to select one-shot mode.
3. Program self test sampling time by CSPT[9:0] or SSPT[9:0] bit-field in ADC_STSAMPT register.
4. Select the self test mode and step by STMSEL[1:0] and STSSEL[2:0] bit-field in ADC_STCFG register.
5. Enable self test function by setting STEN bit in ADC_STCFG register.
6. Start the routine sequence conversion by setting SWRCST (or triggered by external trigger).
7. All routine channel conversions are performed, EORC bit and EORCHC bit are set in ADC_STAT register at the end of conversions.
8. After the end of all the channel conversions in the routine sequence, the selected step number of the selected self test mode is performed similar to a routine channel. If self test step is running, ACTST bit (in ADC_STAT register) will be 1.
9. At the end of self test conversion, the self test conversion result is saved in ADC_STDATA register and STAVLB bit is set. Also, EOSTC bit is set in ADC_STSTAT register.
10. ADC becomes idle state.

For instance, assume that the CH0, CH1 and CH2 are selected to convert in the routine sequence, and one-shot mode is applied in the routine sequence. If STMSEL[1:0] is set to 2'b10 and STSSEL[2:0] is set to 3'b000, capacitive self test step 0 will be performed after CH2 conversion. So, the conversion order is CH0 -> CH1 -> CH2 -> C-Step 0. EORC, EORCHC and EOSTC bits are set to 1 at the end of C-Step 0.

Figure 13-20. Capacitive self test in one-shot mode



In continuous mode

When self test function is enabled, all steps of the configured self test mode can be performed. For supply self test mode, the three supply self test steps are performed at once at the end of routine sequence conversion. For capacitive self test mode, only one self test step is performed at the end of routine sequence conversion. So, capacitive self test step 0 is performed at the end of first time routine sequence conversion, then, capacitive self test step

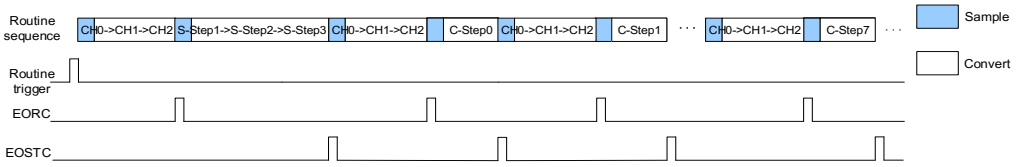
1 is performed at the end of second time routine sequence conversion, and so on. After the end of capacitive self test step 7, start from capacitive self test step 0 again.

Therefore, the configuration procedure for routine sequence with self test conversion in continuous mode is as follows:

1. Configure RL[4:0] bit-field in ADC_RCTL register and RSQ[5:0] bit-field in ADC_RCFGx (x=0...27) registers to select the sequence length and channels to converted for routine sequence.
2. Set MODE=1 to select continuous mode.
3. Program self test sampling time by CSPT[9:0] or SSPT[9:0] bit-field in ADC_STSAMPT register.
4. Select the self test mode by STMSEL[1:0] bit-field. The STSSEL[2:0] bit-field will be not used.
5. Enable self test function by setting STEN bit in ADC_STCFG register.
6. Start the routine sequence conversion by setting SWRCST (or triggered by external trigger).
7. All routine channel conversions are performed, EORC bit and EORCHC bit are set in ADC_STAT register at the end of conversions.
8. At the end of all the channel conversions in the routine sequence, assuming STMSEL[1:0] is set to 2'b11 (supply + capacitive self test), three supply self test steps will be performed by order. If self test step is running, ACTST bit will be 1.
9. At the end of supply self test conversion, the self test conversion result is saved in ADC_STDATA register and STAVLB bit is set. Also, EOSTC is set in ADC_STSTAT register.
10. Then the next routine sequence conversion starts, EORC bit and EORCHC bit are set at the end of conversions.
11. At end of the routine sequence conversion, capacitive self test step 0 is performed.
12. At the end of capacitive self test conversion, the self test conversion result is saved in ADC_STDATA register and STAVLB bit is set. Also, EOSTC is set in ADC_STSTAT register.
13. Then the next routine sequence conversion starts, EORC bit and EORCHC bit are set at the end of conversions.
14. At end of the routine sequence conversion, capacitive self test step 1 is performed.
15. This capacitive self test process continues until the end of capacitive self test step 7. Then, a new round of routine sequence conversion with supply self test steps starts.
16. When routine sequence conversion stops, self test process also stops.
17. ADC becomes idle state.

For instance, assume that the CH0, CH1 and CH2 are selected to convert in the routine sequence, and continuous mode is applied in the routine sequence. If STMSEL[1:0] is set to 2'b11, the conversion order is CH0 -> CH1 -> CH2 -> S-Step 0 -> S-Step 1 -> S-Step 2 -> CH0 -> CH1 -> CH2 -> C-Step 0 -> CH0 -> CH1 -> CH2 -> C-Step 1 -> …… -> CH0 -> CH1 -> CH2 -> C-Step 7 -> CH0 -> CH1 -> CH2 -> S-Step 0 -> S-Step 1 -> S-Step 2….

Figure 13-21. Supply + capacitive self test in continuous mode



13.6.3. Analog watchdog for self test

The ADC provides analog watchdogs to monitor the conversion data of each self test step. For this purpose, separate self test analog watchdogs have been provided for each self test mode.

After the conversion of each self test step, if the analog watchdog function is enabled by setting the SSTnWDEN bit in ADC_SSTWDCFGn (n=0...2) register, a comparison is performed between the converted value and the threshold values.

If the converted value is not between the high and lower threshold values specified by the ADC_SSTWDCFGn (n=0...2) register, the corresponding error flags (CSTE / SST0E / SST1E / SST2E) in ADC_STSTAT register is set. For capacitive self test mode, the step number in which error occurred is updated in the CSTESTP[2:0] bit-field (in ADC_STSTAT register). Also, the wrong data is saved in ADC_STEDATAn (n=0,1) register. An interrupt will be generated if corresponding interrupt enable bit is set to 1 in ADC_STINTEN register. The interrupt signals are also given to FMU module, so that the necessary measures can be taken at the chip level.

Three supply self test steps have three analog watchdogs, and a separate error flag is provided in ADC_STSTAT register for each supply self test step. Hence, there are separate interrupt enable bits for each supply self test step in ADC_STINTEN register. For the same reason, separate fields exist in ADC_STEDATAn (n=0,1) register to store wrong data for each supply self test step.

In supply self test step 1, the internal reference voltage V_{REFINT} is sampled and converted by ADC. If supply self test step 1 watchdog is enabled, the conversion result of supply self test step 1 is compared with high threshold value set in SST1WDHT[11:0] bit-field and low threshold value set in SST1WDLT[11:0] bit-field (in ADC_SSTWDCFG1 register). The SST1E bit in ADC_STSTAT register is set if high or low thresholds is exceeded. SST1E bit is cleared by writing 0 to it.

The conversion results of supply self test step 1 can be obtained by the following formula:

$$Result = V_{REFINT} * 4096 / (V_{REFP} - V_{REFN}) \quad (13\text{-}5)$$

The high and low thresholds can be determined by following formula.

$$\text{High threshold} = \max(V_{REFINT}) * 4096 / (\min(V_{REFP}) - \max(V_{REFN})) \quad (13\text{-}6)$$

$$\text{Low threshold} = \min(V_{REFINT}) * 4096 / (\max(V_{REFP}) - \min(V_{REFN})) \quad (13\text{-}7)$$

Where $\max(\bullet)$ and $\min(\cdot)$ indicate the maximum and minimum values. And these maximum and minimum reference voltages can be found in device datasheet.

For supply self test step 0 and 2, the V_{REFP} and V_{REFN} are sampled and converted by ADC. Because the ideal converted value is 0x000 for supply self test step 0, only the high threshold value is required in ADC_SSTWDCFG0 register. And the ideal converted value is 0xFFF for supply self test step 2, so the low threshold value is required in ADC_SSTWDCFG2 register.

13.6.4. Watchdog timer for self test

The watchdog timer is used to check whether the self test steps are completed within the configured safe time. Self test watchdog timer only supports continuous mode.

Supply and capacitive self test modes have different watchdog timer which runs independently of the other. The watchdog timer for supply self test mode can be enabled by setting the SSTWDTEN bit to 1 in the ADC_SSTWDCFG0 register, and the watchdog timer for capacitive self test mode can be enabled by setting the CSTWDTEN bit to 1 in the ADC_CSTWDCFG3 register. The safe time can be set by STWDT[2:0] bit-field in ADC_STCFG register (the default value is 0.1 ms at ADC clock frequency with 16 MHz).

The safe time is calculated from the routine sequence conversion before self test step 0 until the end of the last step of the same self test mode.

The configuration procedure is as follows:

1. Configure ADC_RCTL and ADC_RCFGx (x=0...27) registers to select the sequence length and channels to converted for routine sequence.
2. Set MODE=1 to select continuous mode.
3. Program self test sampling time by CSPT[9:0] or SSPT[9:0] bit-field in ADC_STSAMPT register.
4. Select the self test mode by STMSEL[1:0] bit-field. And STSSEL[2:0] bit-field will be not used.
5. Enable self test function by setting STEN bit.
6. Configure the safe time by STWDT[2:0] bit-field in ADC_STCFG register.
7. The supply and capacitive self test watchdog timers are enabled by setting SSTWDTEN bit and CSTWDTEN bit. It is necessary to complete all configuration before the watchdog timers are enabled. If one self test mode is selected, watchdog timers for self test mode that are not performed need to be disabled (When STMSEL[1:0] bit-field is set to 2'b11, both the supply and capacitive self test watchdog timers are enabled).
8. Start the routine sequence conversion by setting SWRCST (or triggered by external trigger). And the watchdog timers start counting, this time is t0.
9. All routine channel conversions are performed.
10. At the end of all the channel conversions in the routine sequence (assuming STMSEL[1:0] bit-field is set to 2'b11, supply and capacitive self test mode), three supply self test steps

will be performed by order.

11. After three supply self test steps conversion, then the next routine sequence conversion starts.
12. At end of the routine sequence conversion, capacitive self test step 0 is performed.
13. Then the next routine sequence conversion starts.
14. At end of the routine sequence conversion, capacitive self test step1 is performed.
15. This capacitive self test process continues until the end of capacitive self test step 7. After the capacitive self test process is complete, the watchdog timers stop counting (this time is t1) and begin to re-count. Then, a new round of routine sequence conversion with supply and capacitive self test steps starts and the entire self test process is repeated.

If Δt (t1-t0) is above the safe time which is set in STWDT[2:0] bit-field, the watchdog timer error flag WDTE bit (in ADC_STSTAT register) can be set to 1. An interrupt will generate if WDTEIE bit (in ADC_STINTEN register) is set to 1. The interrupt signal is also given to FMU module if the FMUWDTEIE bit (in ADC_STINTEN register) is set to 1. Otherwise, the watchdog timer reset and restarts to check the next self test process.

Figure 13-22. Watchdog timer for supply self test

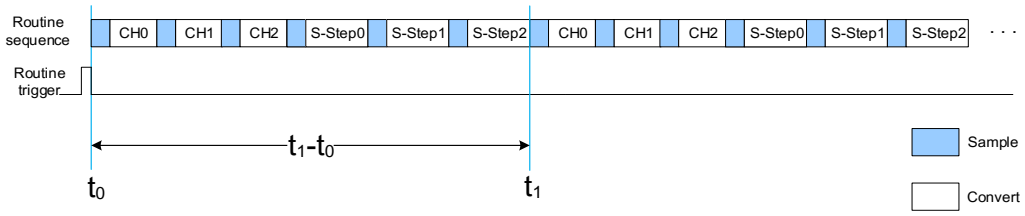


Figure 13-23. Watchdog timer for capacitive self test

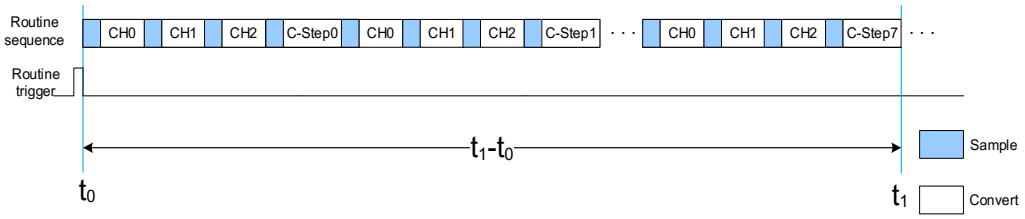
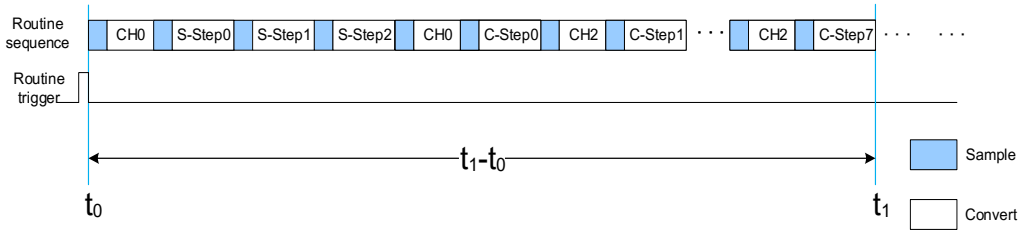


Figure 13-24. Watchdog timer for supply and capacitive self test



13.6.5. Self test steps execution rate control

Self test execution rate control function only supports continuous mode. Self test execution rate control is used to determine that how to arrange self test step in the successive routine sequences. The execution rate is configured in STEXER[7:0] bit-field in ADC_STCFG register.

If STEXER[7:0] bit-field is set to default value 0, a capacitive self test step (or three supply self test steps) is executed after every routine sequence conversion. In this case, if the number of channels in the routine sequence is much more, then the proportion of time taken by one self test step in the routine sequence is small. When the number of channels in the routine sequence is fewer, or even only one, then the time used by a self test step in the routine sequence is a large proportion. So, self test execution rate control is necessary.

When STEXER[7:0] bit-field is set to nonzero value, the self test step to be executed not at the end of each routine sequence conversion, but at the end of (STEXER[7:0] + 1) time of routine sequence conversion. For example, if STEXER[7:0] bit-field is set to 8'b00000100, a capacitive self test step (or three supply self test steps) is executed after five routine sequence conversions, then the next self test step is executed at the end of the next five routine sequence conversions and so on.

13.6.6. Abort self test

Abort for self test steps

In one-shot mode, if RABORT bit is set to 1 when the self test step is ongoing, the self test step is aborted. The EOSTC bit in ADC_STSTAT register will not generated.

In continuous mode (with STEXER[7:0] = 8'b00000000), for capacitive self test mode, if RABORT bit is set to 1 when capacitive self test step n (n=0…7) is ongoing, the capacitive self test step n (n=0…7) is aborted and next routine sequence conversion starts. At the end of this routine sequence conversion, capacitive self test step n (n=0…7) is performed again. In case of supply self test, three supply self test steps are performed again after the end of the next routine sequence conversion. When STEXER[7:0] bit-field is set to a nonzero value, if the abort occurs during the self test step conversions, the self test step is executed after next (STEXER[7:0] + 1) time of routine sequence conversions.

Abort for routine sequence conversions

When STEXER[7:0] bit-field is set to a nonzero value, if the abort occurs during the routine sequence conversions, the self test step is executed at the end of (STEXER[7:0] + 1) time of the routine sequence conversion.

For example, if STEXER[7:0] bit-field is set to 8'b00000011, four routine sequence conversions are followed by a self-test step conversion. If the abort occurs during the third routine sequence conversion, the third routine sequence conversion is aborted, and then this conversion is restarted from the first channel of the routine sequence. The self test step is

executed after the end of the fourth routine sequence conversion.

13.6.7. Self test steps order checking

The self test steps order checking function only supports continuous mode. This function is used to verify that the self test steps are executed in the correct order. If the steps are not in correct order, an error is flagged by setting the STODE bit (in ADC_STSTAT register) to 1. An interrupt will generate if the STODEIE bit (in ADC_STINTEN register) is set to 1. The interrupt signal is also given to FMU module if the FMUSTODEIE bit (in ADC_STINTEN register) is set to 1.

A self test order error will occur in the following conditions:

- Self test steps are not executed in correct order.
- If an abort occurs while a self test step is being performed, this self test step (three supply self test step for supply self test mode) will be performed again at end of next routine sequence. When the self test step starts again, a self test order error will occur and the STODE bit is set to 1.

If an inserted sequence aborts a self test step conversion, when the aborted self test step conversion starts again, the self test order error will not occur and the STODE bit is not set to 1.

13.7. DAC-ADC loop test

ADC and DAC modules can be used to loop test, and the loop test steps are:

1. Configure internal channels (internal channel 56 for DAC0_OUT0 and internal channel 57 for DAC1_OUT0) in inserted sequence 5.
2. Configure DAC0SEL and DAC1SEL bits to select the channels, if DAC0_OUT0 or DAC1_OUT0 need to check.
3. Configure the LTHT[7:0] and LTLT[7:0] bit-fields to set threshold values with given value, the given value is configured by DAC0 / DAC1 register.
4. Set LTEN and LTEIE to 1.
5. Wait DACx (x = 0, 1) conversion end signals to trigger ADC (need to select DAC conversion signals to trigger ADC in TRGSEL).
6. A FMU interrupt will be generated if ADC conversion result exceeds the threshold values of the given value.
7. Repeat step 1 to step 6.

Refer to the DAC module for more details.

13.8. ADC interrupts

The interrupt can be produced on one of the events:

- End of conversion for routine and inserted sequences.
- End of channel conversion.
- The analog watchdog event.
- Self test error event.
- Stream conversion completed interrupt for routine and inserted sequences.
- Channel/Sequence data register overrun interrupt

Separate interrupt enable bits are available for flexibility.

The interrupts of ADC0, ADC1 are mapped to different interrupt vectors IRQ.

13.9. FMU interrupts

The FMU interrupt can be produced on one of the events:

- Self test order error.
- Watchdog timer error.
- Capacitive self test error.
- Supply self test error.
- DAC-ADC loop test error.

13.10. ADC registers

ADC0 base address: 0x4000 F000

ADC1 base address: 0x4000 F800

Table 14-9. ADC register overview

Register name	Symbol	Address offset	Reset value
Control register	ADC_CTL	0x00	0x0000 0000
Status register	ADC_STAT	0x04	0x0000 3F00
Error status register	ADC_ESTAT	0x08	0x0000 0000
Watchdog threshold register	ADC_WDTx	0x0C + 0x04 * x (x = 0...3, x is the watchdog number)	0x0000 0000
Presampling time configure register	ADC_PSMPCFG	0x1C	0x0000 0000
Channel configure register	ADC_CHCFGn	0x20 + 0x04 * n (n = 0...61, n is the channel number)	0x0000 0000
Watchdog interrupt flag register 0	ADC_WDINTF0	0x118	0x0000 0000
Watchdog interrupt flag register 1	ADC_WDINTF1	0x11C	0x0000 0000
External decode delay register	ADC_EDDLY	0x120	0x0000 0000
Conversionstartdelayregister	ADC_CSDLY	0x124	0x0000 0000
Gain configureregister	ADC_GAINCFG	0x128	0x0000 0000
Offset configure register x	ADC_OFFCFGx	0x12C + 0x04 * x (x = 0...3, x is the number of the offset register)	0x0000 0000
Synccontrolregister	ADC_SYNCCTL	0x13C	0x0000 0000
Routinesequencecontrol register	ADC_RCTL	0x200	0x0000 0000
Routinesequencestream register	ADC_RSTREAM	0x204	0x0000 0000
Routinesequence configure register x	ADC_RCFGx	0x208 + 0x04 * x (x = 0...27, where x is the regular sequence channel number)	0x0000 0000
Routine sequence channel conversion end interrupt flag register	ADC_RCHINTF	0x278	0x0000 0000
Routine sequence channel data register x	ADC_RCHDATAx	0x27C + 0x04 * x (x = 0...27, where x is the regular sequence channel number)	0x00000000
Routinesequencedataregister	ADC_RDATA	0x2EC	0x0000 0000
Inserted sequence m control register	ADC_ImCTL	0x300 + m × 0x050 (m = 0...5, m is the	0x0000 0000

		injection sequence number)	
Inserted sequence m stream register	ADC_ImSTREAM	0X304 + m × 0x050 (m = 0... 5, m is the injection sequence number)	0x0000 0000
Inserted sequence m configure register x	ADC_ImCFGx	0X308 + m × 0x050 + 0x04 × x (m = 0... 5, m is the injection sequence number; x = 0... 3, x is the injection sequence channel number)	0x0000 0000
Inserted sequence m channel conversion end interrupt flag register	ADC_ImCHINTF	0X318 + m × 0x050 (m = 0... 5, m is the injection sequence number)	0x0000 0000
Inserted sequence m channel data register x	ADC_ImCHDATAx	0X31C + m × 0x050 + 0x04 × x (m = 0... 5, m is the injection sequence number; x = 0... 3, x is the injection sequence channel number)	0x00000000
Inserted sequence m data register	ADC_ImDATA	0X32C + 0x050 * m (m = 0... 5, m is the injection sequence number)	0x0000 0000
Selftestsampletimeregister	ADC_STSAMPT	0x600	0x0000 0000
Selftest interrupt enable register	ADC_STINTEN	0x604	0x0000 0000
Selftestconfigureregister	ADC_STCFG	0x608	0x0000 0000
Selfteststatusregister	ADC_STSTAT	0x60C	0x0000 0000
Selftesterrordataregister0	ADC_STEDATA0	0x610	0x0000 0000
Self test error data register 1	ADC_STEDATA1	0x614	0x0000 0000
Self test conversion data register	ADC_STDATA	0x618	0x0000 0000
Watchdog for supply self test configure register 0	ADC_SSTWDCFG0	0x61C	0x0010 0000
Watchdog for supply self test configure register 1	ADC_SSTWDCFG1	0x620	0x0768 0348
Watchdog for supply self test configure register 2	ADC_SSTWDCFG2	0x624	0x0000 0FEF
Watchdog for capacitive self test configure register 0	ADC_CSTWDCFG0	0x628	0x0030 0030
Watchdog for capacitive self test	ADC_CSTWDCFG1	0x62C	0x0030 0030

configure register 1			
Watchdog for capacitive self test configure register 2	ADC_CSTWDCFG2	0x630	0x0030 0030
Watchdog for capacitive self test configure register 3	ADC_CSTWDCFG3	0x634	0x000C 0018
DAC - ADC loop test configure register	ADC_DTALTCFG	0x638	0x0000 0000

13.10.1. Control register (ADC_CTL)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INREFEN	HPTSVEN	Reserved					STLK	CALNUM[2:0]			RSTCLB	CLB	CALMOD	Reserved	CKAUTO
rw	rw						rw	rw			rw	rw	rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GAINEN	DRES[1:0]		OVWEN	CHOVWIE	OVWIE	DAL	DMA	Reserved		AVRGSEL[2:0]			AVRGEN	Reserved	ADCON
rw	rw		rw	rw	rw	rw	rw			rw			rw		rw

Bits	Fields	Descriptions
31	INREFEN	Internal reference voltage enable This bit can be set or cleared by software in ADCx (x = 0, 1). When this bit is set, Channel 58 is enabled in ADC0 and ADC1. 0: Internal reference voltage channel disable 1: Internal reference voltage channel enable
30	HPTSVEN	High-precision temperature sensor enable This bit can be set or cleared by software in ADCx (x = 0, 1). Channel 59 is enabled in ADC0 and ADC1. 0: High-precision temperature sensor channel disable 1: High-precision temperature sensor channel enable
29:25	Reserved	Must be kept at reset value.
24	STLK	Self test register lock This bit is used to protects the following self test registers (from ADC_STSAMPT register to ADC_DTALTCFG register) from writing. 0: Self test registers can be modified (are read-write) 1: Self test registers cannot be modified (are read-only)
23:21	CALNUM[2:0]	Calibration Times These bits define the calibration times for ADC. 000: Calibration count=1, Total calibration duration=536 ADC clock cycles (33.5us，f_{ADC} =16MHz)

001: Calibration count=3, Total calibration duration=770 ADC clock cycles (48.13us，f_{ADC} =16MHz)
010: Calibration count=7, Total calibration duration=1238 ADC clock cycles (77.38us，f_{ADC} =16MHz)
011: Calibration count=15, Total calibration duration=2174 ADC clock cycles (135.88us，f_{ADC} =16MHz)
100: Calibration count=31, Total calibration duration=4046 ADC clock cycles (252.88us，f_{ADC}=16MHz)
others: reserved.

20	RSTCLB	Reset calibration This bit is set by software and cleared by hardware after the calibration registers are initialized. 0: Calibration register initialize done. 1: Initialize calibration register start.
19	CLB	ADC calibration This bit is cleared by hardware. 0: Calibration done 1: Calibration start
18	CALMOD	ADC calibration mode 0: Offset, mismatch mode 1: Offset mode **Note**: 1) The offset mode is used to calibrate the offset voltage of the ADC internal comparator to reduce the influence caused by the comparator offset voltage. 2) The mismatch mode is used to calibrate the capacitor adaptation of the ADC internal to reduce the influence caused by capacitor adaptation.
17	Reserved	Must be kept at reset value.
16	CKAUTO	Clock automatic off This bit used to turn off the analog part clock when the ADC is idle. 0: Clock on 1: Clock automatic off when the ADC in Idle state
15	GAINEN	Gain mode enable This bit can be set or cleared by software to the gain mode. 0: Gain mode disabled 1: Gain mode enabled and effect on all channels **Note**: Software is allowed to write this bit must ensure that no conversion is ongoing.
14:13	DRES[1:0]	ADC data resolution 00: 12bit 01: 10bit 10: 8bit

		11: 6bit
12	OVWEN	Overwrite enable This bit determines whether the data register can acquire new conversion data before reading the current converted data. When overwrite is enabled, new conversion result will replace the older conversion data regardless of its validity. 0: Overwrite disabled. The data register can't acquire new conversion data before reading the current converted data. 1: Overwrite enabled The data register can acquire new conversion data before reading the current converted data.
11	CHOVWIE	Channel overwrite interrupt enable 0: Disable overwrite interrupt 1: Enable overwrite interrupt
10	OVWIE	Sequence overwrite interrupt enable 0: Disable overwrite interrupt 1: Enable overwrite interrupt
9	DAL	Data alignment 0: LSB alignment 1: MSB alignment
8	DMA	DMA request enable 0: DMA request disable 1: DMA request enable
7:6	Reserved	Must be kept at reset value
5:3	AVRGSEL[2:0]	Averaging conversion number selection These bits are used to select the number of conversions to average to obtain the conversion data. 000: 2 conversions 001: 4 conversions 010: 8 conversions 011: 16 conversions 100: 32 conversions 101: 64 conversions 110: 128 conversions 111: 256 conversions
2	AVRGEN	Averaging function enable This bit is used to control whether conversion averaging function is enabled. 0: Disabled 1: Enabled
1	Reserved	Must be kept at reset value
0	ADCON	ADC ON The ADC will be wake up when this bit is changed from low to high and take a

stabilization time. For power saving, when this bit is reset, the analog submodule will be put into power down mode.
0: ADC disable and power down
1: ADC enable

13.10.2. Status register (ADC_STAT)

Address offset: 0x04
Reset value: 0x0000 3F00

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	EOI5CHC	EOI4CHC	EOI3CHC	EOI2CHC	EOI1CHC	EOI0CHC	EORCHC	Reserved	EOI5C	EOI4C	EOI3C	EOI2C	EOI1C	EOI0C	EORC
	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ACTCH[5:0]							Reserved	ACTST	ACTSQ[2:0]			ADCSTAT[2:0]		
	r								r	r			r		

Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30	EOI5CHC	End of inserted sequence 5 channel conversion This bit indicates whether conversion of inserted sequence 5 channel has completed. 0: No end of inserted sequence 5 channel conversion 1: End of inserted sequence 5 channel conversion
29	EOI4CHC	End of inserted sequence 4 channel conversion This bit indicates whether conversion of inserted sequence 4 channel has completed. 0: No end of inserted sequence4 channel conversion 1: End of inserted sequence4 channel conversion
28	EOI3CHC	End of inserted sequence 3 channel conversion This bit indicates whether conversion of inserted sequence 3 channel has completed. 0: No end of inserted sequence 3 channel conversion 1: End of inserted sequence 3 channel conversion
27	EOI2CHC	End of inserted sequence 2 channel conversion This bit indicates whether conversion of inserted sequence 2 channel has completed. 0: No end of inserted sequence 2 channel conversion 1: End of inserted sequence 2 channel conversion
26	EOI1CHC	End of inserted sequence 1 channel conversion This bit indicates whether conversion of inserted sequence 1 channel has

completed.
0: No end of inserted sequence 1 channel conversion
1: End of inserted sequence 1 channel conversion

25	EOI0CHC	End of inserted sequence 0 channel conversion This bit indicates whether conversion of inserted sequence 0 channel has completed. 0: No end of inserted sequence 0 channel conversion 1: End of inserted sequence 0 channel conversion
24	EORCHC	End of routine sequence channel conversion This bit indicates whether conversion of routine sequence channel has completed. 0: No end of routine sequence channel conversion 1: End of routine sequence channel conversion
23	Reserved	Must be kept at reset value.
22	EOI5C	End of inserted sequence 5 conversion This bit indicates whether conversion of inserted sequence 5 has completed. 0: No end of inserted sequence 5 conversion 1: End of inserted sequence 5 conversion
21	EOI4C	End of inserted sequence 4 conversion This bit indicates whether conversion of inserted sequence 4 has completed. 0: No end of inserted sequence 4 conversion 1: End of inserted sequence 4 conversion
20	EOI3C	End of inserted sequence 3 conversion This bit indicates whether conversion of inserted sequence 3 has completed. 0: No end of inserted sequence 3 conversion 1: End of inserted sequence 3 conversion
19	EOI2C	End of inserted sequence 2 conversion This bit indicates whether conversion of inserted sequence 2 has completed. 0: No end of inserted sequence 2 conversion 1: End of inserted sequence 2 conversion
18	EOI1C	End of inserted sequence 1 conversion This bit indicates whether conversion of inserted sequence 1 has completed. 0: No end of inserted sequence 1 conversion 1: End of inserted sequence 1 conversion
17	EOI0C	End of inserted sequence 0 conversion This bit indicates whether conversion of inserted sequence 0 has completed. 0: No end of inserted sequence 0 conversion 1: End of inserted sequence 0 conversion
16	EORC	End of routine sequence conversion This bit indicates whether conversion of routine sequence has completed.

		0: No end of routine sequence conversion 1: End of routine sequence conversion
15:14	Reserved	Must be kept at reset value.
13:8	ACTCH[5:0]	Active channel status These bits indicate which channel is converting. 000000: ADC channel 0 000001: ADC channel 1 000010: ADC channel 2 000011: ADC channel 3 000100: ADC channel 4 …… 111101: ADC channel 61 Other values are reserved.
7	Reserved	Must be kept at reset value.
6	ACTST	Active self test status This bit indicates whether an ongoing conversion is for self test. 0: No self test ongoing 1: Self test ongoing
5:3	ACTSQ[2:0]	Active sequence status These bits indicate which sequence in current conversion. 000: No conversion 001: Routine sequence 010: Inserted sequence 0 011: Inserted sequence 1 100: Inserted sequence 2 101: Inserted sequence 3 110: Inserted sequence 4 111: Inserted sequence 5
2:0	ADCSTAT[2:0]	ADC State These bits indicate the current state of the ADC. 000: IDLE (ADC is not working) 001: WAIT (conversion start delay state) 010: PSAMP (presampling state) 011: EDDLY (external decode signals delay state) 100: SAMP (sampling state) 101: CONV (ADC is converting)

13.10.3. Error status register (ADC_ESTAT)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WDE	Reserved				I5TRGE	I5ABORTF	Reserved		I4TRGE	I4ABORTF	I4SUSPDF	Reserved	I3TRGE	I3ABORTF	I3SUSPDF
rc_w0					rc_w0	rc_w0			rc_w0	rc_w0	r		rc_w0	rc_w0	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	I2TRGE	I2ABORTF	I2SUSPDF	Reserved	I1TRGE	I1ABORTF	I1SUSPDF	Reserved	I0TRGE	I0ABORTF	I0SUSPDF	Reserved	RTRGE	RABORTF	RSUSPDF
	rc_w0	rc_w0	r		rc_w0	rc_w0	r		rc_w0	rc_w0	r		rc_w0	rc_w0	r

Bits	Fields	Descriptions
31	WDE	Analog Watchdog event flag This bit indicates whether the analog watchdog event occurred. 0: No analog watchdog event occurs 1: Analog watchdog event occurs
30:27	Reserved	Must be kept at reset value.
26	I5TRGE	Inserted sequence 5 trigger error This bit indicates whether the inserted sequence 5 trigger come when the inserted sequence 5 is converting. If the trigger come, a trigger error will occur. 0: No trigger error 1: Trigger error
25	I5ABORTF	Inserted sequence 5 aborted flag This bit indicates inserted sequence 5 whether or not be aborted. 0: Not aborted 1: Aborted
24:23	Reserved	Must be kept at reset value.
22	I4TRGE	Inserted sequence 4 trigger error This bit indicates whether the inserted sequence 4 trigger come when the inserted sequence 4 is converting. If the trigger come, a trigger error will occur. 0: No trigger error 1: Trigger error
21	I4ABORTF	Inserted sequence 4 aborted flag This bit indicates inserted sequence 4 whether or not be aborted. 0: Not aborted 1: Aborted
20	I4SUSPDF	Inserted sequence 4 suspended flag This bit is read-only and indicates inserted sequence 4 whether or not be suspended. When inserted sequence 4 is suspended, the bit is set until the conversion resumes. 0: Not suspended 1: Suspended Note: If, at the moment when the current sequence conversion is completed, a

		trigger signal for a high-priority sequence arrives simultaneously to interrupt it, the suspension flag of the current sequence will be set. However, after the high-priority conversion is completed, the current sequence will not be re-converted.
19	Reserved	Must be kept at reset value.
18	I3TRGE	Inserted sequence 3 trigger error This bit indicates whether the inserted sequence 3 trigger come when the inserted sequence 3 is converting. If the trigger come, a trigger error will occur. 0: No trigger error 1: Trigger error
17	I3ABORTF	Inserted sequence 3 aborted flag This bit indicates inserted sequence 3 whether or not be aborted. 0: Not aborted 1: Aborted
16	I3SUSPDF	Inserted sequence 3 suspended flag This bit is read-only and indicates inserted sequence 3 whether or not be suspended. When inserted sequence 3 is suspended, the bit is set until the conversion resumes. 0: Not suspended 1: Suspended Note: If, at the moment when the current sequence conversion is completed, a trigger signal for a high-priority sequence arrives simultaneously to interrupt it, the suspension flag of the current sequence will be set. However, after the high-priority conversion is completed, the current sequence will not be re-converted.
15	Reserved	Must be kept at reset value.
14	I2TRGE	Inserted sequence 2 trigger error This bit indicates whether the inserted sequence 2 trigger come when the inserted sequence 2 is converting. If the trigger come, a trigger error will occur. 0: No trigger error 1: Trigger error
13	I2ABORTF	Inserted sequence 2 aborted flag This bit indicates inserted sequence 2 whether or not be aborted. 0: Not aborted 1: Aborted
12	I2SUSPDF	Inserted sequence 2 suspended flag This bit is read-only and indicates inserted sequence 2 whether or not be suspended. When inserted sequence 2 is suspended, the bit is set until the conversion resumes. 0: Not suspended 1: Suspended

Note: If, at the moment when the current sequence conversion is completed, a trigger signal for a high-priority sequence arrives simultaneously to interrupt it, the suspension flag of the current sequence will be set. However, after the high-priority conversion is completed, the current sequence will not be re-converted.

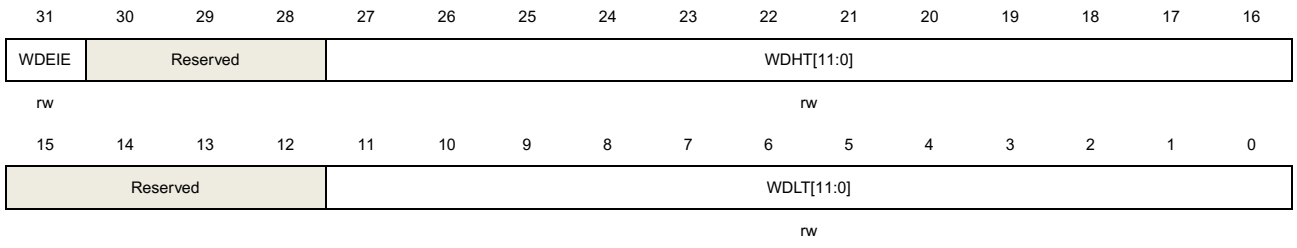
11	Reserved	Must be kept at reset value.
10	I1TRGE	Inserted sequence 1 trigger error This bit indicates whether the inserted sequence 1 trigger come when the inserted sequence 1 is converting. If the trigger come, a trigger error will occur. 0: No trigger error 1: Trigger error
9	I1ABORTF	Inserted sequence 1 aborted flag This bit indicates inserted sequence 1 whether or not be aborted. 0: Not aborted 1: Aborted
8	I1SUSPDF	Inserted sequence 1 suspended flag This bit is read-only and indicates inserted sequence 1 whether or not be suspended. When inserted sequence 1 is suspended, the bit is set until the conversion resumes. 0: Not suspended 1: Suspended Note: If, at the moment when the current sequence conversion is completed, a trigger signal for a high-priority sequence arrives simultaneously to interrupt it, the suspension flag of the current sequence will be set. However, after the high-priority conversion is completed, the current sequence will not be re-converted.
7	Reserved	Must be kept at reset value.
6	I0TRGE	Inserted sequence 0 trigger error This bit indicates whether the inserted sequence 0 trigger come when the inserted sequence 0 is converting. If the trigger come, a trigger error will occur. 0: No trigger error 1: Trigger error
5	I0ABORTF	Inserted sequence 0 aborted flag This bit indicates inserted sequence 0 whether or not be aborted. 0: Not aborted 1: Aborted
4	I0SUSPD	Inserted sequence 0 suspended flag This bit is read-only and indicates inserted sequence 0 whether or not be suspended. When inserted sequence 0 is suspended, the bit is set until the conversion resumes. 0: Not suspended

		1: Suspended Note: If, at the moment when the current sequence conversion is completed, a trigger signal for a high-priority sequence arrives simultaneously to interrupt it, the suspension flag of the current sequence will be set. However, after the high-priority conversion is completed, the current sequence will not be re-converted.
3	Reserved	Must be kept at reset value.
2	RTRGE	Routine sequence trigger error This bit indicates whether the routine sequence trigger come when the routine sequence is converting. If the trigger come, a trigger error will occur. 0: No trigger error 1: Trigger error
1	RABORTF	Routine sequence aborted flag This bit indicates routine sequence whether or not aborted. 0: Not aborted 1: Aborted
0	RSUSPDF	Routine sequence suspended flag This bit is read-only and indicates routine sequence whether or not be suspended. When routine sequence is suspended, the bit is set until the conversion resumes. 0: Not suspended 1: Suspended

13.10.4. Watchdog threshold register x (ADC_WDTx) (x = 0…3)

Address offset: 0x0C + 0x04 * x (x = 0...3, where x is the number of watchdog)
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



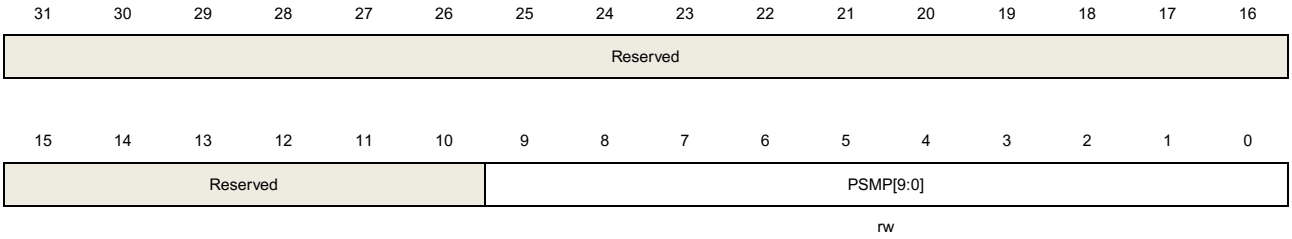
Bits	Fields	Descriptions
31	WDEIE	Analog watchdog x interrupt enable 0: Interrupt disable 1: Interrupt enable
30:28	Reserved	Must be kept at reset value.
27:16	WDHT[11:0]	High threshold for analog watchdog x

		These bits define the high threshold for the analog watchdog x.
15:12	Reserved	Must be kept at reset value.
11:0	WDLT[11:0]	Low threshold for analog watchdog x These bits define the low threshold for the analog watchdog x.

13.10.5. Presampling time configure register(ADC_PSMPCFG)

Address offset: 0x1C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

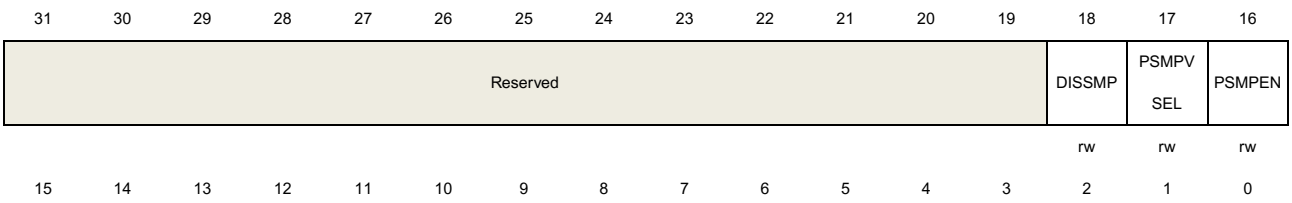


Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:0	PSMP[9:0]	ADC channel n pre-sample time 00 0000 0000: 2.5 cycles 00 0000 0001: 3.5 cycles 00 0000 0010: 4.5 cycles 00 0000 0011: 5.5 cycles 00 0000 0100: 6.5 cycles …… 10 0111 1110: 640.5 cycles All other values are reserved. **Note**: if PSMP[9:0]>638, the work behavior in ADC will not be guaranteed.

13.10.6. Channel configure register n (ADC_CHCFGn) (n = 0...61)

Address offset: 0x20 + 0x04 * n (n = 0...61, where n is a channel number)
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Reserved	WDSEL[1:0]	WDMSEL[2:0]
	rw	rw

Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18	DISSMP	Disable channel n sampling This bit determines whether to convert the presampled value after presampling. If the bit is set to 1, convert the presampled value after presampling and this conversion data is stored in the corresponding data register. 0: Channel n's sampling is not disabled, no conversion for the presampled value after presampling 1: Channel n's sampling is disabled, conversion for the presampled value after presampling **Note**: This bit is just used with n = 0..55, presampling does not apply for internal channels.
17	PSMPVSEL	ADC channel n pre-sample voltage select 0: V_{REFN} (Internal channel 61) 1: V_{REFP} (Internal channel 60) **Note**: This bit is just used with n = 0..55, presampling does not apply for internal channels.
16	PSMPEN	ADC channel n pre-sample enable 0: Disable 1: Enable **Note**: This bit is just used with n = 0..55, presampling does not apply for internal channels.
15:5	Reserved	Must be kept at reset value.
4:3	WDSEL[1:0]	ADC channel n analog watchdog selection 00: Analog watchdog 0 01: Analog watchdog 1 10: Analog watchdog 2 11: Analog watchdog 3
2:0	WDMSEL[2:0]	Analog watchdog mode select Result is the conversion result of ADC channel n, WDHT is the watchdog high threshold, and WDLT is the low threshold. 000: Analog watchdog disable 001: Result <=WDLT 010: WDLT < Result <= WDHT 011: Result > WDHT 100: Result > WDLT 101: Result > WDHT or Result<= WDLT 110: Result <= WDHT

111: Reserved

13.10.7. Watchdog interrupt flag register 0 (ADC_WDINTF0)

Address offset: 0x118

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WDIF0[31:16]															
rc_w0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDIF0[15:0]															
rc_w0															

Bits	Fields	Descriptions
31:0	WDIF0[31:0]	Analog watchdog interrupt flag 0 Each bit corresponds to channel n (n is from 0 to 31) analog watchdog event flag. The corresponding bits are set to 1 by hardware when the conversion result of channel n (n is from 0 to 31) meets analog watchdog event. This bit-field can be cleared by software writing 0 to it, and also be cleared by software writing 0 to WDE bit. WDIF0[n] = 0: No channel n analog watchdog event occurred WDIF0[n] = 1: Channel n analog watchdog event occurred

13.10.8. Watchdog interrupt flag register 1 (ADC_WDINTF1)

Address offset: 0x11C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		WDIF1[29:16]													
		rc_w0													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDIF1[15:0]															
rc_w0															

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29:0	WDIF1[29:0]	Analog watchdog interrupt flag 1 Each bit corresponds to channel n (n is from 32 to 61) analog watchdog event flag. The corresponding bits are set to 1 by hardware when the conversion result of

channel n (n is from 32 to 61) meets analog watchdog event.

This bit-field can be cleared by software writing 0 to it, and also be cleared by software writing 0 to WDE bit.

WDIF1[n-32] = 0: No channel n analog watchdog event occurred

WDIF1[n-32] = 1: Channel n analog watchdog event occurred

13.10.9. External decode delay register (ADC_EDDLY)

Address offset: 0x120

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EDDLY[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	EDDLY[15:0]	External analog multiplexers decode signals delay time The value of this bit-field is expressed as the number of ADC clock cycles. 0000 0000 0000 0000: 0 ADC clock cycles 0000 0000 0000 0001: 1 ADC clock cycles 0000 0000 0000 0010: 2 ADC clock cycles 0000 0000 0000 0011: 3 ADC clock cycles 0000 0000 0000 0100: 4 ADC clock cycles …… 1111 1111 1111 1111: 65 535 ADC clock cycles

13.10.10. Conversion start delay register (ADC_CSDLY)

Address offset: 0x124

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									CSDEN[6:0]						
									rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSDLY[15:0]															
rw															

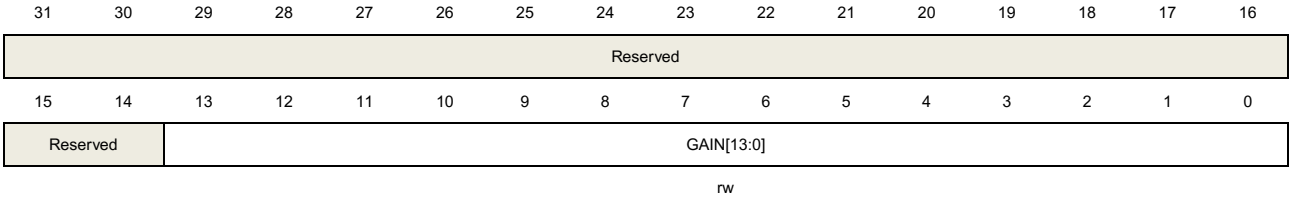
Bits	Fields	Descriptions
31:23	Reserved	Must be kept at reset value.
22:16	CSDEN[6:0]	The delay time from trigger to start conversion enable 0000000: All sequences trigger delay disable 0000001: Routine sequence trigger delay enable 0000010: Inserted sequence 0 trigger delay enable 0000100: Inserted sequence 1 trigger delay enable 0001000: Inserted sequence 2 trigger delay enable 0010000: Inserted sequence 3 trigger delay enable 0100000: Inserted sequence 4 trigger delay enable 1000000: Inserted sequence 5 trigger delay enable
15:0	CSDLY[15:0]	The delay time from trigger to start conversion The value of this bit-field is expressed as the number of ADC clock cycles. 0000 0000 0000 0000: 1 ADC clock cycles 0000 0000 0000 0001: 2 ADC clock cycles 0000 0000 0000 0010: 3 ADC clock cycles 0000 0000 0000 0011: 4 ADC clock cycles 0000 0000 0000 0100: 5 ADC clock cycles …… 1111 1111 1111 1111: 65536 ADC clock cycles

13.10.11. Gain configure register (ADC_GAINCFG)

Address offset: 0x128

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13:0	GAIN[13:0]	Gain factor These bits can be set or cleared by software to program the gain factor. The programmable range of GAIN is 0 to 16383, and the actual gain calculation factor (GAIN / 4096) is 0 to 3.999756. 00 0000 0000 0000: The gain is 0, the value of (GAIN /4096) is 0. ... 01 0000 0000 0000: The gain is 4096, the value of (GAIN / 4096) is 1.

...

10 0000 0000 0000: The gain is 8192, the value of (GAIN / 4096) is 2.

...

11 0000 0000 0000: The gain is 12288, the value of (GAIN / 4096) is 3.

...

11 1111 1111 1111: The gain is 16383, the value of (GAIN / 4096) is 3.999756

Note: The GAINEN bit of ADC_CTL register must set 1 and then the gain factor can be effected.

13.10.12. Offset configure register x (ADC_OFFCFGx) (x = 0…3)

Address offset: 0x12C + 0x04 * x (x = 0...3, where x is offset register number)

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OFFEN	OFFPOS	SIGNEN	Reserved							OFFCH[5:0]					
rw	rw	rw								rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				OFFSET[11:0]											
				rw											

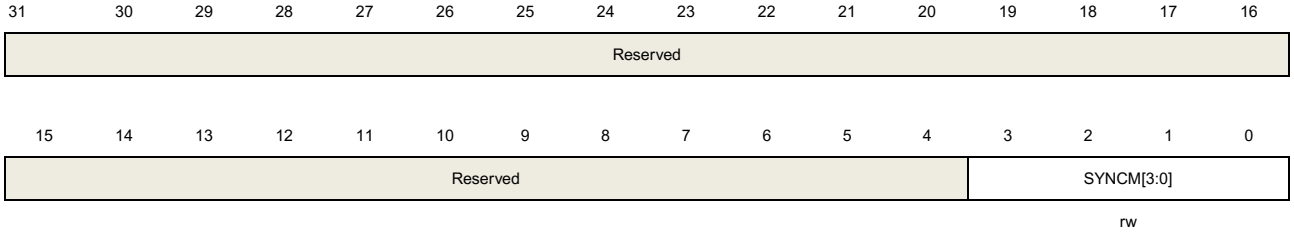
Bits	Fields	Descriptions
31	OFFEN	Offset x enable This bit is written by software to enable or disable the offset calculation. 0: Disable the offset 1: Enable the offset
30	OFFPOS	Positive offset This bit is written by software to select the negative offset or the positive offset. 0: Negative offset. OFFSET[11:0] bits will be subtracted from the raw converted data of per channel. 1: Positive offset. OFFSET[11:0] bits will be added from the raw converted data of per channel.
29	SIGNEN	Signed data enable This bit is written by software to enable the data signed function. 0: Offset result be signed, and data saturated at 0x000 and 0xFFF 1: Signed enabled
28:22	Reserved	Must be kept at reset value.
21:16	OFFCH[5:0]	Channel selection for the offset x These bits are written by software to define the channel to which the offset programmed into bits OFFSET[11:0] applies. 000000: ADC channel 0

		000001: ADC channel 1 000010: ADC channel 2 000011: ADC channel 3 000100: ADC channel 4 …… 111101: ADC channel 61 Other values are reserved.
15:12	Reserved	Must be kept at reset value.
11:0	OFFSET[11:0]	Offset factor These bits will be used to calculate the conversion result with the raw converted data of the selected channel.

13.10.13. Sync control register (ADC_SYNCCTL)

Address offset: 0x13C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

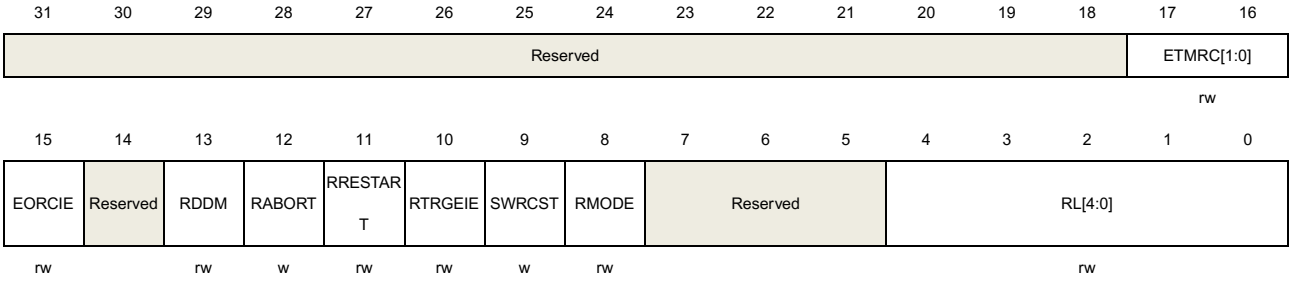


Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
3:0	SYNCM[3:0]	ADC sync mode When ADC sync mode is enabled these bits should be set to 0000 firstly before change to another value. 0000: ADC sync mode disabled. All the ADCs work independently. 0001: ADC0 and ADC1 work in combined routine parallel & inserted parallel mode. 0010: ADC0 and ADC1 work in combined routine parallel & trigger rotation mode. 0101: ADC0 and ADC1 work in inserted parallel mode. 0110: ADC0 and ADC1 work in routine parallel mode. 0111: ADC0 and ADC1 work in follow-up mode. 1001: ADC0 and ADC1 work in trigger rotation mode. All other values are reserved.

13.10.14. Routine sequence control register (ADC_RCTL)

Address offset: 0x200
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



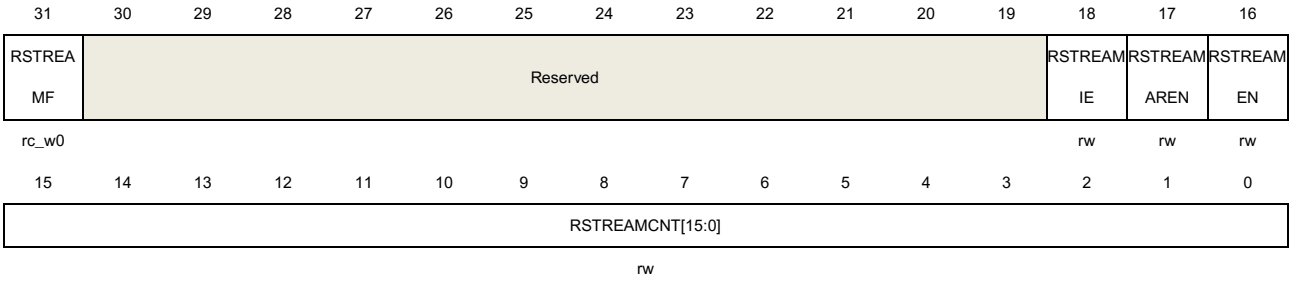
Bits	Fields	Descriptions
31:18	Reserved	Must be kept at reset value.
17:16	ETMRC[1:0]	External trigger mode for routine sequence 00: External trigger for routine sequence disable 01: Rising edge of external trigger for routine sequence enable 10: Falling edge of external trigger for routine sequence enable 11: Rising and falling edge of external trigger for routine sequence enable
15	EORCIE	Interrupt enable for EORC 0: Interrupt disable 1: Interrupt enable
14	Reserved	Must be kept at reset value.
13	RDDM	Routine sequence DMA disable mode 0: The DMA engine is disabled after the end of transfer signal from DMA controller is detected. 1: When DMA=1, the DMA engine issues a request at end of each routine conversion.
12	RABORT	Abort routine sequence current conversion 0: Routine sequence conversion proceeds 1: Routine sequence conversion aborted
11	RRESTART	Restart routine sequence when the routine sequence be suspended 0: Resume the sequence conversion from the suspended channel 1: Restart the sequence conversion from the first channel
10	RTRGEIE	Interrupt enable for RTRGE 0: Interrupt disable 1: Interrupt enable
9	SWRCST	Software start on routine channel Setting 1 on this bit starts a conversion of a sequence of routine channels. It is set by software and cleared by hardware immediately after the conversion starts.
8	RMODE	Routine sequence conversion mode 0: One-shot mode

		1: Continuous mode
7:5	Reserved	Must be kept at reset value.
4:0	RL[4:0]	Routine sequence length The total number of conversion in routine sequence equals to RL[4:0] + 1.

13.10.15. Routine sequence stream register (ADC_RSTREAM)

Address offset: 0x204
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

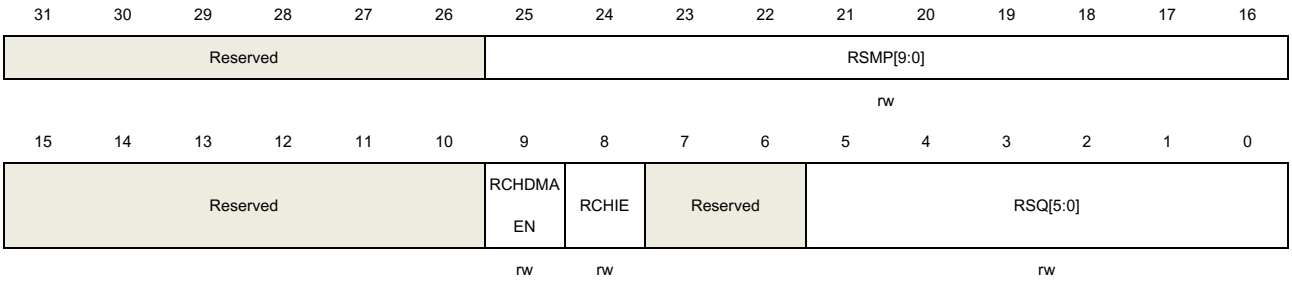


Bits	Fields	Descriptions
31	RSTREAMF	Routine sequence stream flag This bit indicates whether a routine sequence stream conversion has completed. 0: Stream conversion not complete 1: Stream conversion complete
30:19	Reserved	Must be kept at reset value.
18	RSTREAMIE	Routine sequence stream interrupt enable 0: Disabled 1: Enabled
17	RSTREAMAREN	Routine sequence stream counter value reload enable 0: Disabled 1: Enabled
16	RSTREAMEN	Routine sequence stream conversion enable 0: Disabled 1: Enabled
15:0	RSTREAMCNT[15:0]	Routine sequence stream counter value The stream conversion number is equal to RSTREAMCNT[15:0] + 1. This bit-field can't be written when RSTREAMEN = 1'b1. When RSTREAMAREN = 1'b1, the RSTREAMCNT[15:0] bit-field can be reloaded automatically by hardware with the previously programmed value after the routine sequence has completed the set number of samples and conversions.

13.10.16. Routine sequence configure register x (ADC_RCFGx) (x = 0…27)

Address offset: 0x208 + 0x04 * x (x = 0...27, where x is routine sequence channel number)
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

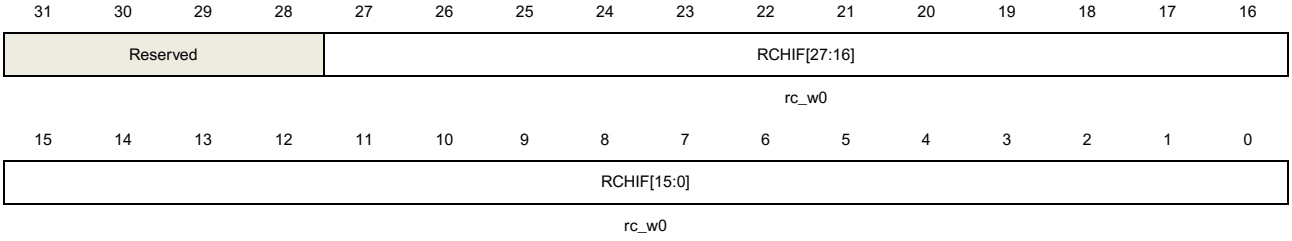


Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25:16	RSMP[9:0]	Routine sequence channel x sample time 00 0000 0000: 2.5 cycles 00 0000 0001: 3.5 cycles 00 0000 0010: 4.5 cycles 00 0000 0011: 5.5 cycles 00 0000 0100: 6.5 cycles …… 10 0111 1110: 640.5 cycles All other values are reserved. **Note**: if RSMP[9:0] > 638, the work behavior in ADC will not be guaranteed.
15:10	Reserved	Must be kept at reset value.
9	RCHDMAEN	Routine sequence channel x DMA request enable This bit is used to enable the DMA request of routine sequence channel x when the end of channel (selected by RSQ[5:0] bit-field) conversion. 0: DMA request disable 1: DMA request enable
8	RCHIE	Routine sequence channel x interrupt enables for RCHIF or EORCHC This bit is used to enable an interrupt request of routine sequence channel x when the end of channel (selected by RSQ[5:0] bit-field) conversion. 0: Interrupt disable 1: Interrupt enable
7:6	Reserved	Must be kept at reset value.
5:0	RSQ[5:0]	The channel number n (n is from 0 to 61) is written to these bits to select a channel as the xth conversion in the routine sequence. **Note**: The channel number can't over than 61 in ADC0 and ADC1.

13.10.17. Routine sequence channel conversion end interrupt flag register (ADC_RCHINTF)

Address offset: 0x278
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

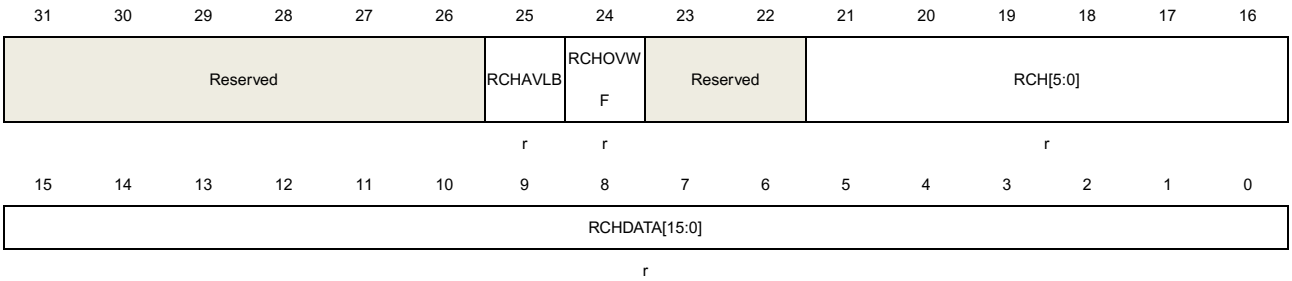


Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:0	RCHIF[27:0]	Routine sequence channel conversion end interrupt flag These bits indicate whether conversion of routine channel selected by RSQ[5:0] bit-field has completed. Each bit corresponds to an routine channel conversion end flag. Set by hardware at the end of routine channel, and cleared by software writing 0 to it. The software clears the EORCHC bit and can also clear this bit-field. RCHIF[x] = 0: Routine sequence channel x conversion not end RCHIF[x] = 1: Routine sequence channel x conversion end

13.10.18. Routine sequence channel data register x (ADC_RCHDATAx) (x = 0…27)

Address offset: 0x27C + 0x04 * x (x = 0...27, where x is routine sequence channel number)
Reset value: 0x00000000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25	RCHAVLB	Routine sequence channel x conversion data available This bit indicates whether the new conversion data of routine sequence channel x

		is available. This bit is reset to 0 by hardware when the data is read. 0: Unavailable 1: Available
24	RCHOVWF	Routine sequence channel x conversion data overwritten flag This bit indicates whether the previous conversion data of routine sequence channel x was overwritten in the case that the previous conversion data having not been read. Overwrite conversion data function is enabled by OVWEN bit in ADC_CTL register. 0: No overwritten 1: Overwritten Note: Read ADC_RCHDATAx will clean the RCHOVWF flag.
23:22	Reserved	Must be kept at reset value.
21:16	RCH[5:0]	Routine sequence channel x These bits indicate which channel is selected by RSQ[5:0] bit-field, whose conversion data is stored in RCHDATA[15:0] bit-field. 000000: ADC channel 0 000001: ADC channel 1 000010: ADC channel 2 000011: ADC channel 3 000100: ADC channel 4 …… 111101: ADC channel 61 Other values are reserved.
15:0	RCHDATA[15:0]	Routine sequence channel x conversion data These bits contain conversion result from routine channel selected by RSQ[5:0] bit-field. The conversion result is always 16-bit wide, regardless of the conversion resolution selected (DRES[1:0] bit-field in ADC_CTL register).

13.10.19. Routine sequence data register (ADC_RDATA)

Address offset: 0x2EC

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDMAEN	Reserved					RAVLB	ROVWF	Reserved		RSQCH[5:0]					
rw						r	r			r					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA15:0]															
r															

Bits	Fields	Descriptions
31	RDMAEN	Routine sequence DMA request enable This bit is used to enable a DMA request at the end of conversion of a routine channel. 0: DMA request disable 1: DMA request enable
30:26	Reserved	Must be kept at reset value.
25	RAVLB	Routine sequence conversion data available This bit indicates whether the new conversion data in routine sequence is available. This bit is reset to 0 by hardware when the data is read. 0: No available 1: Available
24	ROVWF	Routine sequence conversion data overwritten flag This bit indicates whether the previous conversion data in routine sequence was overwritten in the case that the previous conversion data having not been read. Overwrite conversion data function is enabled by OVWEN bit in ADC_CTL register. 0: No overwritten 1: Overwritten Note: Read ADC_RDATA will clean the ROVWF flag.
23:22	Reserved	Must be kept at reset value.
21:16	RSQCH[5:0]	Routine sequence channel These bits indicate which channel is selected by RSQ[5:0] bit-field in ADC_RCFGx (x = 0…27) registers, whose conversion data is stored in RDATA[15:0] bit-field. 000000: ADC channel 0 000001: ADC channel 1 000010: ADC channel 2 000011: ADC channel 3 000100: ADC channel 4 …… 111101: ADC channel 61 Other values are reserved.
15:0	RDATA[15:0]	Routine sequence conversion data These bits contain conversion result from routine sequence. RDATA[15:0] bit-field has a buffer register to record conversion result, and they can store up to two valid conversion data. The conversion result is always 16-bit wide, regardless of the conversion resolution selected (DRES[1:0] bit-field in ADC_CTL register).

13.10.20. Inserted sequence m control register (ADC_ImCTL) (m = 0...5)

Address offset: 0x300 + m × 0x050 (m = 0...5, where m is inserted sequence number)

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														ETMIC[1:0]	
														rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IEOCIE	Reserved	IDDM	IABORT	IRESTART	ITRGEIE	SWICST	Reserved							IL[1:0]	
rw		rw	w	rw	rw	w								rw	

Bits	Fields	Descriptions
31:18	Reserved	Must be kept at reset value.
17:16	ETMIC	External trigger mode for inserted sequence 00: External trigger for inserted sequence disable 01: Rising edge of external trigger for inserted sequence enable 10: Falling edge of external trigger for inserted sequence enable 11: Rising and falling edge of external trigger for inserted sequence enable
15	EOICIE	Interrupt enable for EOIC 0: Interrupt disable 1: Interrupt enable
14	Reserved	Must be kept at reset value.
13	IDDM	Inserted sequence DMA disable mode 0: The DMA engine is disabled after the end of transfer signal from DMA controller is detected. 1: When DMA=1, the DMA engine issues a request at end of each inserted conversion.
12	IABORT	Abort inserted sequence m current conversion 0: Inserted sequence m current conversion proceeds 1: Inserted sequence m current conversion aborted
11	IRESTART	Restart inserted sequence m when inserted sequence m be suspended 0: Resume the sequence conversion from the suspended channel 1: Restart the sequence conversion from the first channel **Note**: This bit is just used with m = 0…4. When m = 5, this bit is reserved.
10	ITRGEIE	Interrupt enable for ITRGE 0: Interrupt disable 1: Interrupt enable
9	SWICST	Software start on inserted channel. Setting 1 on this bit starts a conversion of a sequence of inserted channels. It is set by software and cleared by hardware immediately after the conversion starts.
8:2	Reserved	Must be kept at reset value.

1:0	IL[1:0]	Inserted sequence length The total number of conversion in inserted sequence equals to IL[1:0] + 1.

13.10.21. Inserted sequence m stream register (ADC_ImSTREAM) (m = 0...5)

Address offset: 0x304 + m × 0x050 (m = 0...5, where m is inserted sequence number)
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISTREAMF	Reserved												ISTREAMIE	ISTREAMAREN	ISTREAMEN
rc_w0													rw	rw	rw

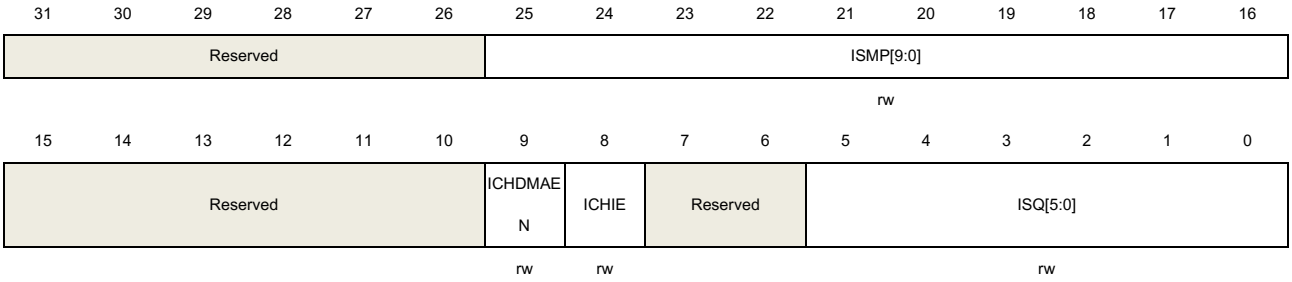
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ISTREAMCNT[15:0]															
rw															

Bits	Fields	Descriptions
31	ISTREAMF	Inserted sequence stream flag This bit indicates whether an inserted sequence stream conversion has completed. 0: Stream conversion not complete 1: Stream conversion complete
30:19	Reserved	Must be kept at reset value.
18	ISTREAMIE	Inserted sequence stream interrupt enable 0: Disabled 1: Enabled
17	ISTREAMAREN	Inserted sequence stream counter value reload enable 0: Disabled 1: Enabled
16	ISTREAMEN	Inserted sequence stream conversion enable 0: Disabled 1: Enabled
15:0	ISTREAMCNT[15:0]	Inserted sequence stream counter value The stream conversion number is equal to ISTREAMCNT[15:0] + 1. This bit-field can't be written when ISTREAMEN = 1'b1. When ISTREAMAREN = 1'b1, the ISTREAMCNT[15:0] bit-field can be reloaded automatically by hardware with the previously programmed value after the inserted sequence has completed the set number of samples and conversions.

13.10.22. Inserted sequence m configure register x (ADC_ImCFGx) (m = 0...5, x = 0...3)

Address offset: 0x308 + m × 0x050 + 0x04 × x
(m = 0...5, where m is inserted sequence number; x = 0...3, where x is inserted sequence channel number)
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



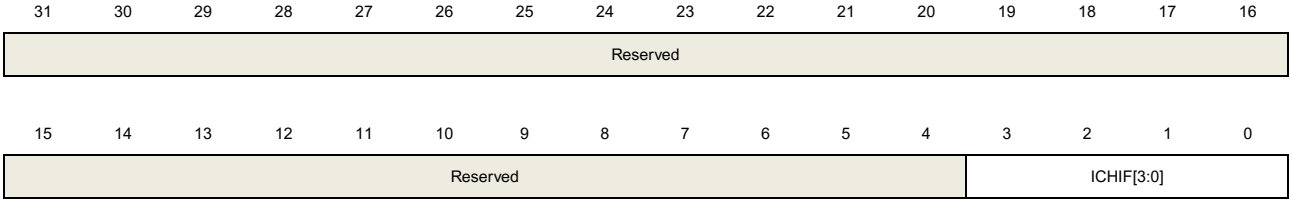
Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25:16	ISMP[9:0]	Inserted sequence m channel x sample time 00 0000 0000: 2.5 cycles 00 0000 0001: 3.5 cycles 00 0000 0010: 4.5 cycles 00 0000 0011: 5.5 cycles 00 0000 0100: 6.5 cycles …… 10 0111 1110: 640.5 cycles All other values are reserved. **Note**: if ISMP[9:0]>638, the work behavior in adc will not be guaranteed.
15:10	Reserved	Must be kept at reset value.
9	ICHDMAEN	Inserted sequence m channel x DMA request enable This bit is used to enable the DMA request of inserted sequence x when the end of channel (selected by ISQ[5:0] bit-field) conversion. 0: DMA request disable 1: DMA request enable
8	ICHIE	Inserted sequence m channel x interrupt enable for ICHIF or EOImCHC (m = 0...5) This bit is used to enable an interrupt request of inserted sequence x when the end of channel (selected by ISQ[5:0] bit-field) conversion. 0: Interrupt disable 1: Interrupt enable
7:6	Reserved	Must be kept at reset value.

5:0	ISQ[5:0]	The channel number n (n is from 0 to 61) is written to these bits to select a channel as the xth conversion in the inserted sequence. **Note**: The channel number can't over than 61 in ADC0 and ADC1.

13.10.23. Inserted sequence m channel conversion end interrupt flag register (ADC_ImCHINTF) (m = 0...5)

Address offset: 0x318 + m × 0x050 (m = 0...5, where m is inserted sequence number)
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

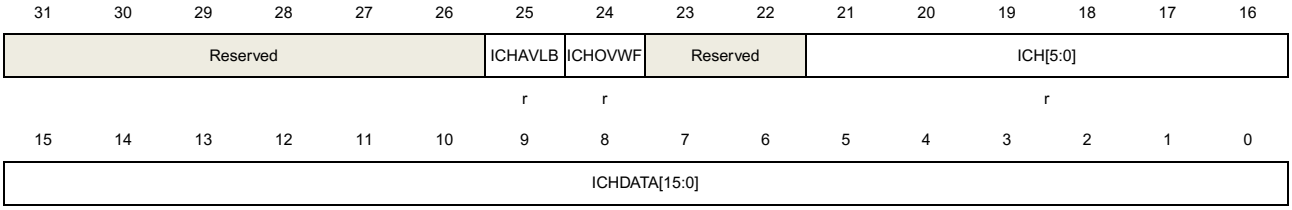


Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3:0	ICHIF[3:0]	Inserted sequence m channel conversion end interrupt flag This bit indicates whether conversion of inserted channel selected by ISQ[5:0] bit-field (in ADC_ImCFGx register, m = 0...5) has completed. Each bit corresponds to an inserted channel conversion end flag. Set by hardware at the end of inserted channel, and cleared by software writing 0 to it. The software clears the EOImCHC bit and can also clear this bit-field. ICHIF[x] = 0: Inserted sequence m channel x conversion not end. ICHIF[x] = 1: Inserted sequence m channel x conversion end.

13.10.24. Inserted sequence m channel data register x (ADC_ImCHDATAx) (m = 0...5, x = 0...3)

Address offset: 0x31C + m × 0x050+ 0x04 × x
(m = 0...5, where m is inserted sequence number; x = 0...3, where x is inserted sequence channel number)
Reset value: 0x00000000

This register has to be accessed by word (32-bit).



r

Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25	ICHAVLB	Inserted sequence m channel x conversion data available This bit indicates whether new conversion data of inserted sequence m channel x is available. This bit is reset to 0 by hardware when the data is read. 0: Unavailable 1: Available
24	ICHOVWF	Inserted sequence m channel x conversion data overwritten flag This bit indicates whether the previous conversion data of inserted sequence m channel x was overwritten in the case that the previous conversion data having not been read. Overwrite conversion data function is enabled by OVWEN bit in ADC_CTL register. 0: No overwritten 1: Overwritten Note: Read ADC_ImCHDATAx will clean the ICHOVWF flag.
23:22	Reserved	Must be kept at reset value.
21:16	ICH[5:0]	Inserted sequence m channel x These bits indicate which channel is selected by ISQ[5:0] bit-field (in ADC_ImCFGx register), whose conversion data is stored in IxCHDATA[15:0] bit-field. 000000: ADC channel 0 000001: ADC channel 1 000010: ADC channel 2 000011: ADC channel 3 000100: ADC channel 4 …… 111101: ADC channel 61 Other values are reserved.
15:0	ICHDATA[15:0]	Inserted sequence m channel x conversion data These bits contain conversion result from inserted channel selected by ISQ[5:0] bit-field (in ADC_ImCFGx register). The conversion result is always 16-bit wide, regardless of the conversion resolution selected (DRES[1:0] bit-field in ADC_CTL register).

13.10.25. Inserted sequence m data register (ADC_ImDATA) (m = 0...5)

Address offset: 0x32C + 0x050 * m (m = 0...5, where m is inserted sequence number)
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IDMAEN	Reserved					IAVLB	IOVWF	Reserved		ISQCH[5:0]					
rw						r	r			r					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDATA[15:0]															
r															

Bits	Fields	Descriptions
31	IDMAEN	Inserted sequence m DMA request enable This bit is used to enable a DMA request at the end of conversion of a channel of inserted sequence m. 0: DMA request disable 1: DMA request enable
30:26	Reserved	Must be kept at reset value.
25	IAVLB	Inserted sequence m conversion data available This bit indicates whether the new conversion data in inserted sequence m is available. This bit is reset to 0 by hardware when the data is read. 0: Unavailable 1: Available
24	IOVWF	Inserted sequence m conversion data overwritten flag This bit indicates whether the previous conversion data in inserted sequence m was overwritten in the case that the previous conversion data having not been read. Overwrite conversion data function is enabled by OVWEN bit in ADC_CTL register. 0: No overwritten 1: Overwritten Note: Read ADC_ImDATA will clean the IOVWF flag.
23:22	Reserved	Must be kept at reset value.
21:16	ISQCH[4:0]	Inserted sequence m channel These bits indicate which channel is selected by ISQ[5:0] bit-field in ADC_ImCFGx (m = 0...5, x = 0…3) registers, whose conversion data is stored in IDATA[15:0] bit-field. 000000: ADC channel 0 000001: ADC channel 1 000010: ADC channel 2 000011: ADC channel 3 000100: ADC channel 4 …… 111101: ADC channel 61 Other values are reserved.
15:0	IDATA[15:0]	Inserted sequence m conversion data These bits contain conversion result from inserted sequence. IDATA[15:0] bit-field

has a buffer register to record conversion result, and they can store up to two valid conversion data.

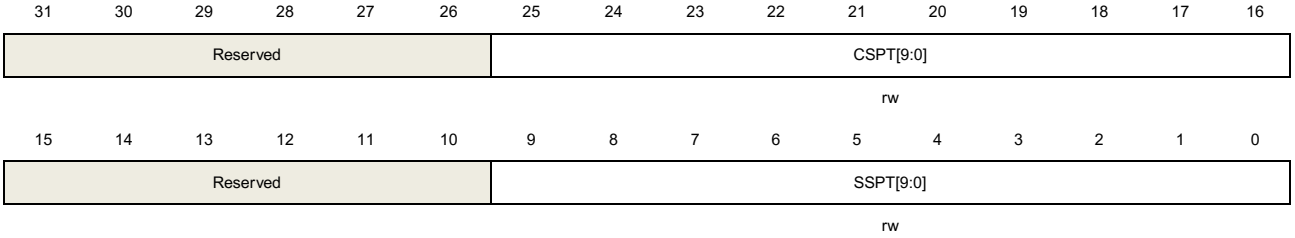
The conversion result is always 16-bit wide, regardless of the conversion resolution selected (DRES[1:0] bit-field in ADC_CTL register).

13.10.26. Self test sample time register (ADC_STSAMPT)

Address offset: 0x600

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



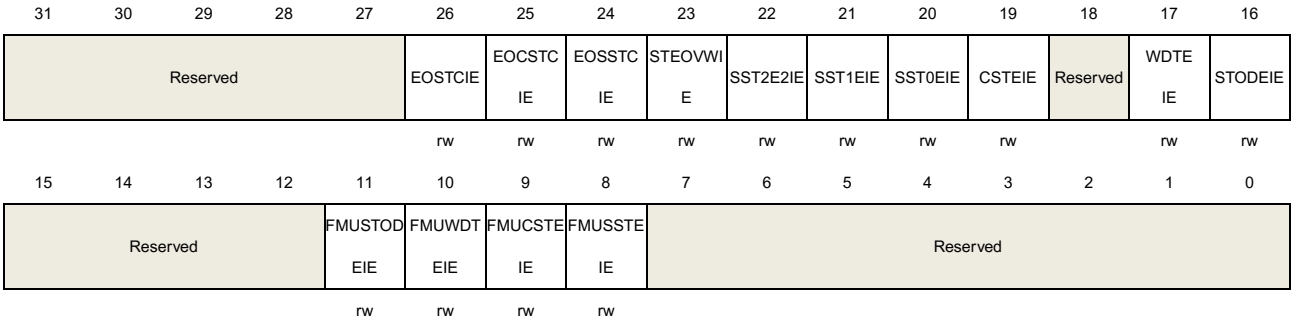
Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25:16	CSPT[9:0]	Sampling time for capacitive self test step 00 0000 0000: 2.5 cycles 00 0000 0001: 3.5 cycles 00 0000 0010: 4.5 cycles 00 0000 0011: 5.5 cycles 00 0000 0100: 6.5 cycles …… 10 0111 1110: 640.5 cycles All other values are reserved. **Note**: if CSPT[9:0] > 638, the work behavior in adc will not be guaranteed.
15:10	Reserved	Must be kept at reset value.
9:0	SSPT[9:0]	Sampling time for supply self test step 00 0000 0000: 2.5 cycles 00 0000 0001: 3.5 cycles 00 0000 0010: 4.5 cycles 00 0000 0011: 5.5 cycles 00 0000 0100: 6.5 cycles …… 10 0111 1110: 640.5 cycles All other values are reserved. **Note**: if SSPT[9:0] > 638, the work behavior in ADC will not be guaranteed.

13.10.27. Self test interrupt enable register (ADC_STINTEN)

Address offset: 0x604

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:27	Reserved	Must be kept at reset value.
26	EOSTCIE	Interrupt enable for End of self test conversion 0: Interrupt disable 1: Interrupt enable
25	EOCSTCIE	Interrupt enable for End of all capacitive self test steps conversion 0: Interrupt disable 1: Interrupt enable
24	EOSSTCIE	Interrupt enable for End of all supply self test steps conversion 0: Interrupt disable 1: Interrupt enable
23	STEOVWIE	Interrupt enable for Self test error overwrite 0: Interrupt disable 1: Interrupt enable
22	SST2EIE	Interrupt enable for Supply self test step 2 error 0: Interrupt disable 1: Interrupt enable
21	SST1EIE	Supply self test step 1 error 0: Interrupt disable 1: Interrupt enable
20	SST0EIE	Supply self test step 0 error 0: Interrupt disable 1: Interrupt enable
19	CSTEIE	Interrupt enable for Capacitive self test error 0: Interrupt disable

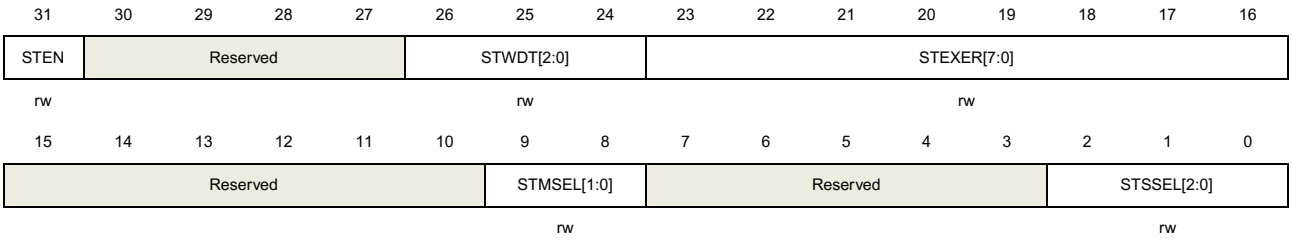
		1: Interrupt enable
18	Reserved	Must be kept at reset value.
17	WDTEIE	Interrupt enable for Watchdog timer error 0: Interrupt disable 1: Interrupt enable
16	STODEIE	Interrupt enable for Self test step order error 0: Interrupt disable 1: Interrupt enable
15:12	Reserved	Must be kept at reset value.
11	FMUSTODEIE	Interrupt enable for self test order error STODE to FMU 0: Interrupt disable 1: Interrupt enable
10	FMUWDTEIE	Interrupt enable for watchdog timer error WDTE to FMU 0: Interrupt disable 1: Interrupt enable
9	FMUCSTEIE	Interrupt enable for capacitive self test error CSTE to FMU 0: Interrupt disable 1: Interrupt enable
8	FMUSSTEIE	Interrupt enable for supply self test error SSTxE to FMU 0: Interrupt disable 1: Interrupt enable
7:0	Reserved	Must be kept at reset value.

13.10.28. Self test configure register (ADC_STCFG)

Address offset: 0x608

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31	STEN	Self test enable When this bit is set, the self test channels are enabled. 0: Disable

		1: Enable
30:27	Reserved	Must be kept at reset value.
26:24	STWDT[2:0]	Self test watchdog timer These bits define a safety limit time period for self test. The safe time period is the maximum duration until the running self test is expected to finish. 000: 0.1 ms at 15 MHz ADC clock frequency 001: 0.5 ms at 15 MHz ADC clock frequency 010: 1 ms at 15 MHz ADC clock frequency 011: 2 ms at 15 MHz ADC clock frequency 100: 5 ms at 15 MHz ADC clock frequency 101: 10 ms at 15 MHz ADC clock frequency 110: 20 ms at 15 MHz ADC clock frequency 111: 50 ms at 15 MHz ADC clock frequency
23:16	STEXER[7:0]	Self test execution rate These bits define the number of conversions of the routine sequence, and then the next self test step is executed. These bits should be configured before self test is enabled. 00000000: A self test step is executed after the routine sequence has been converted every time. 00000001: A self test step is executed after the routine sequence has been converted 2 times. …… 11111111: A self test step is executed after the routine sequence has been converted 256 times.
15:10	Reserved	Must be kept at reset value.
9:8	STMSEL[1:0]	Self test mode select 00: Supply self test 01: Reserved 10: Capacitive self test 11: Supply and capacitive self test (First the supply self test, then the capacitor self test)
7:3	Reserved	Must be kept at reset value.
2:0	STSSEL[2:0]	Capacitive and supply self test steps select in one shot mode 000: Supply self test step 0 or capacitive self test step 0 001: Supply self test step 1 or capacitive self test step 1 010: Supply self test step 2 or capacitive self test step 2 011: Capacitive self test step 3 100: Capacitive self test step 4 101: Capacitive self test step 5 110: Capacitive self test step 6 111: Capacitive self test step 7

Note: These bits are not used and should be set to 0 in continuous mode.

13.10.29. Self test status register (ADC_STSTAT)

Address offset: 0x60C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					EOSTC	EOCSTC	EOSSTC	STEOVW	SST2E	SST1E	SST0E	CSTE	Reserved	WDTE	STODE
					rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0		rc_w0	rc_w0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													CSTESTP[2:0]		
													r		

Bits	Fields	Descriptions
31:27	Reserved	Must be kept at reset value.
26	EOSTC	End of self test conversion This bit indicates whether a self test conversion has completed. 0: Not end of self test conversion 1: End of self test conversion
25	EOCSTC	End of all capacitive self test steps conversion This bit indicates whether all capacitive self test steps conversion have completed. 0: Not end of all capacitive self test conversion 1: End of all capacitive self test conversion
24	EOSSTC	End of all supply self test steps conversion This bit indicates whether all supply self test steps conversion have completed. 0: Not end of all supply self test conversion 1: End of all supply self test conversion
23	STEOVW	Self test error overwrite This bit indicates whether a responding error occurred when the error flag (SST2E, SST1E, SST0E, CERR bit) was set. 0: No error overwritten occurred 1: Error overwritten occurred
22	SST2E	Supply self test step 2 error This bit indicates whether an error occurred during conversion of supply self test step 2. 0: No error occurred 1: Error occurred
21	SST1E	Supply self test step 1 error This bit indicates whether an error occurred during conversion of supply self test

		step 1. 0: No error occurred 1: Error occurred
20	SST0E	Supply self test step 0 error This bit indicates whether an error occurred during conversion of supply self test step 0. 0: No error occurred 1: Error occurred
19	CSTE	Capacitive self test error This bit indicates whether an error occurred during conversion of capacitive self test. 0: No error occurred 1: Error occurred
18	Reserved	Must be kept at reset value.
17	WDTE	Watchdog timer error This bit indicates whether the self test has completed within the safety limit time. 0: Self test completed within the safety limit time. 1: Self test did not complete within safety limit time.
16	STODE	Self test step order error This bit indicates whether the self test step is converted in the correct order. 0: Self test step converted in correct order 1: Self test step did not convert in correct order
15:3	Reserved	Must be kept at reset value.
2:0	CSTESTP[2:0]	Error step of capacitive self test This bit is read only and indicates the capacitive self test step where an error occurred. 000: An error occurred at capacitive self test step 0 001: An error occurred at capacitive self test step 1 010: An error occurred at capacitive self test step 2 011: An error occurred at capacitive self test step 3 100: An error occurred at capacitive self test step 4 101: An error occurred at capacitive self test step 5 110: An error occurred at capacitive self test step 6 111: An error occurred at capacitive self test step 7

13.10.30. Self test error data register 0 (ADC_STEDATA0)

Address offset: 0x610
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved	SST2EDATA[11:0]
	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SST1EDATA[11:0]											
				r											

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:16	SST2EDATA[11:0]	Error data of supply self test step 2 conversion result These bits define the error converted data of supply self test step 2.
15:12	Reserved	Must be kept at reset value.
11:0	SST1EDATA[11:0]	Error data of supply self test step 1 conversion result These bits define the error converted data of supply self test step 1.

13.10.31. Self test error data register 1 (ADC_STEDATA1)

Address offset: 0x614
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				SST0EDAT[11:0]											
				r											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				CSTEDATA[11:0]											
				r											

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:16	SST0EDATA[11:0]	Error data of supply self test step 0 conversion result These bits define the error converted data of supply self test step 0.
15:12	Reserved	Must be kept at reset value.
11:0	CSTEDATA[11:0]	Error data of capacitive self test step conversion result These bits define the error converted data of capacitive self test steps.

13.10.32. Self test conversion data register (ADC_STDATA)

Address offset: 0x618
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						STAVLB	STOVWF	Reserved			STSTP[4:0]				
						r	r				r				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				STDATA[11:0]											
				r											

Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25	STAVLB	Self test conversion data available This bit indicates whether the new conversion data in self test is available. This bit is reset to 0 by hardware when the data is read. 0: No available 1: Available
24	STOVWF	Self test conversion data overwritten flag This bit indicates whether the previous conversion data in self test was overwritten in the case that the previous conversion data having not been read. Overwrite conversion data function is enabled by OVWEN bit in ADC_CTL register. 0: No overwritten 1: Overwritten
23:21	Reserved	Must be kept at reset value.
20:16	STSTP[4:0]	Self test step This bit-field contains which self test step converted this result data in STDATA[11:0]. 0000~0010: for supply self test. 0000~0111: for capacitive self test 0000~1010: for supply and capacitive self test. 0000~0010 is for supply self test, 0011~1010 is for capacitive self test.
15:12	Reserved	Must be kept at reset value.
11:0	STDATA[11:0]	Self test conversion data These bits contain conversion result from self test step.

13.10.33. Watchdog for supply self test configure register 0 (ADC_SSTWDCFG0)

Address offset: 0x61C
Reset value: 0x0010 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SST0WDEN	SSTWDTEN	Reserved		SST0WDHT[11:0]											

rw	rw														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits	Fields	Descriptions
31	SST0WDEN	Supply self test step 0 watchdog enable This bit is used to enable the watchdog for supply self test step 0. 0: Disable 1: Enable
30	SSTWDTEN	Supply self test watchdog timer enable This bit is used to enable the self test watchdog timer for supply self test. The self test watchdog timer has two functions. One is to check whether the conversion order of the steps of the supply self test is correct. The other one is to check whether the supply self test was completed within the safety limit time. 0: Disable 1: Enable **Note**: This bit just used in continuous mode (The MODE bit is 1).
29:28	Reserved	Must be kept at reset value.
27:16	SST0WDHT[11:0]	Supply test step 0 watchdog high threshold value These bits define the high threshold for the supply test step 0 watchdog.
15:0	Reserved	Must be kept at reset value.

13.10.34. Watchdog for supply self test configure register 1 (ADC_SSTWDCFG1)

Address offset: 0x620

Reset value: 0x0768 0348

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SST1WDEN	Reserved			SST1WDHT[11:0]											
rw				rw											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SST1WDLT[11:0]											
				rw											

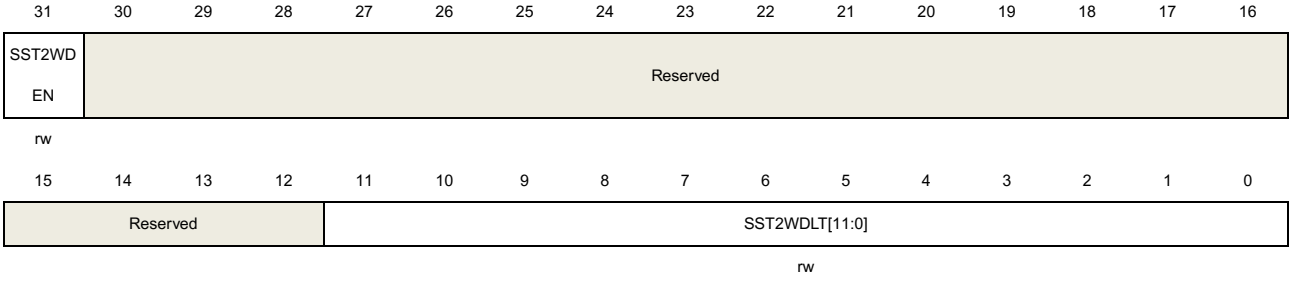
Bits	Fields	Descriptions
31	SST1WDEN	Supply self test step 1 watchdog enable This bit is used to enable the watchdog for supply self test step 1. 0: Disable 1: Enable

30:28	Reserved	Must be kept at reset value.
27:16	SST1WDHT[11:0]	Supply test step 1 watchdog high threshold value These bits define the high threshold for the supply test step 1 watchdog.
15:12	Reserved	Must be kept at reset value.
11:0	SST1WDLT[11:0]	Supply test step 1 watchdog low threshold value These bits define the low threshold for the supply test step 1 watchdog.

13.10.35. Watchdog for supply self test configure register 2 (ADC_SSTWDCFG2)

Address offset: 0x624
Reset value: 0x0000 0FEF

This register has to be accessed by word (32-bit).

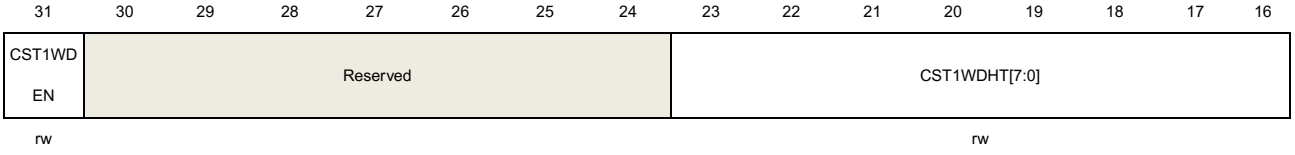


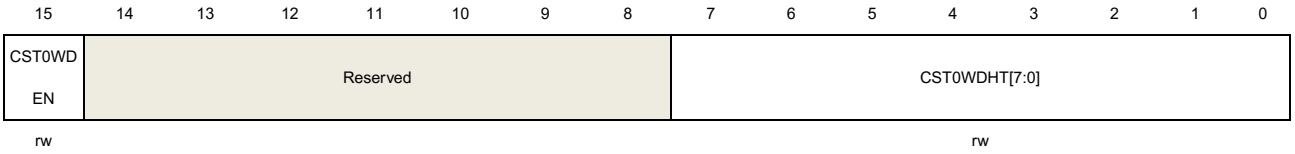
Bits	Fields	Descriptions
31	SST2WDEN	Supply self test step 2 watchdog enable This bit is used to enable the watchdog for supply self test step 2. 0: Disable 1: Enable
30:12	Reserved	Must be kept at reset value.
11:0	SST2WDLT[11:0]	Supply test step 2 watchdog low threshold value These bits define the low threshold for the supply test step 2 watchdog.

13.10.36. Watchdog for capacitive self test configure register 0 (ADC_CSTWDCFG0)

Address offset: 0x628
Reset value: 0x0030 0030

This register has to be accessed by word (32-bit).



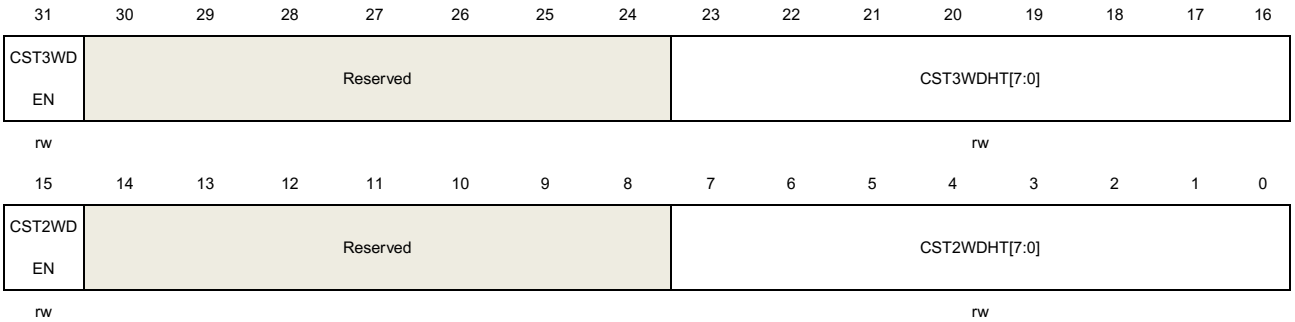


Bits	Fields	Descriptions
31	CST1WDEN	Capacitive self test step 1 watchdog enable This bit is used to enable the watchdog for capacitive self test step 1. 0: Disable 1: Enable
30:24	Reserved	Must be kept at reset value.
23:16	CST1WDHT[7:0]	Capacitive test step 1 analog watchdog high threshold value These bits define the high threshold for the capacitive test step 1 analog watchdog.
15	CST0WDEN	Capacitive self test step 0 watchdog enable This bit is used to enable the watchdog for capacitive self test step 0. 0: Disable 1: Enable
14:8	Reserved	Must be kept at reset value.
7:0	CST0WDHT[7:0]	Capacitive test step 0 analog watchdog high threshold value These bits define the high threshold for the capacitive test step 0 analog watchdog.

13.10.37. Watchdog for capacitive self test configure register 1 (ADC_CSTWDCFG1)

Address offset: 0x62C
Reset value: 0x0030 0030

This register has to be accessed by word (32-bit).



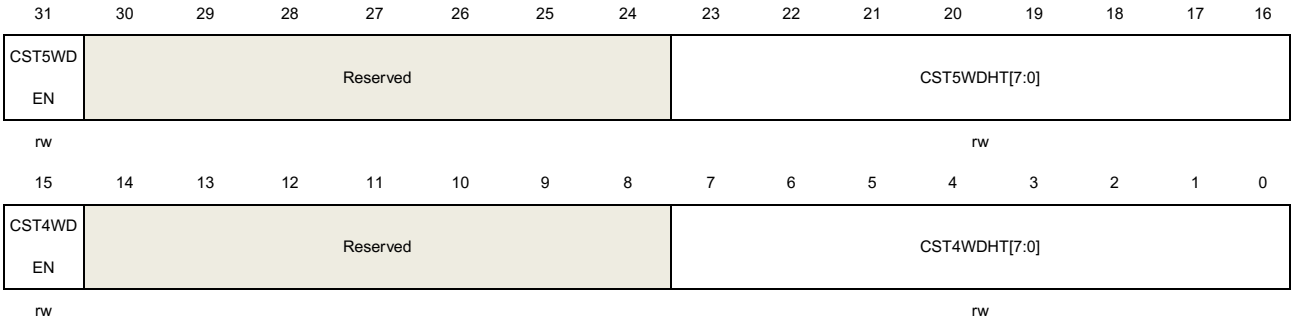
Bits	Fields	Descriptions
31	CST3WDEN	Capacitive self test step 3 watchdog enable This bit is used to enable the watchdog for capacitive self test step 3. 0: Disable

		1: Enable
30:24	Reserved	Must be kept at reset value.
23:16	CST3WDHT[7:0]	Capacitive test step 3 analog watchdog high threshold value These bits define the high threshold for the capacitive test step 3 analog watchdog.
15	CST2WDEN	Capacitive self test step 2 watchdog enable This bit is used to enable the watchdog for capacitive self test step 2. 0: Disable 1: Enable
14:8	Reserved	Must be kept at reset value.
7:0	CST2WDHT[7:0]	Capacitive test step 2 analog watchdog high threshold value These bits define the high threshold for the capacitive test step 2 analog watchdog.

13.10.38. Watchdog for capacitive self test configure register 2 (ADC_CSTWDCFG2)

Address offset: 0x630
Reset value: 0x0030 0030

This register has to be accessed by word (32-bit).



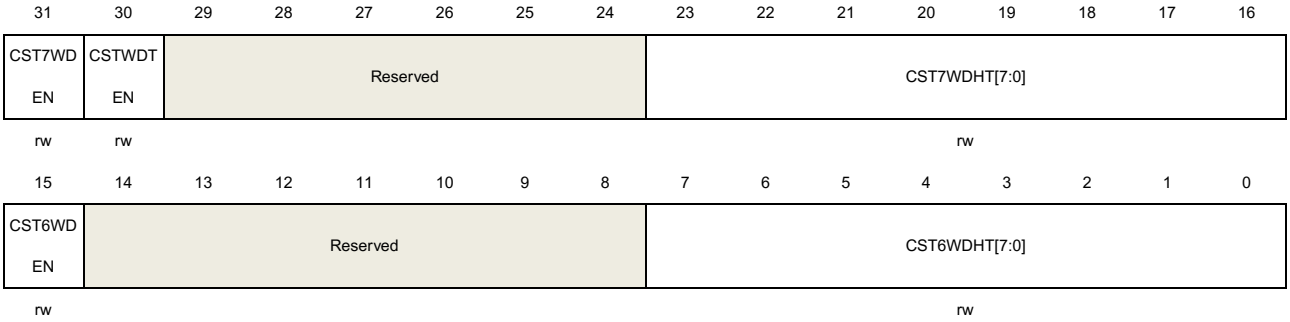
Bits	Fields	Descriptions
31	CST5WDEN	Capacitive self test step 5 watchdog enable This bit is used to enable the watchdog for capacitive self test step 5. 0: Disable 1: Enable
30:24	Reserved	Must be kept at reset value.
23:16	CST5WDHT[7:0]	Capacitive test step 5 analog watchdog high threshold value These bits define the high threshold for the capacitive test step 5 analog watchdog.
15	CST4WDEN	Capacitive self test step 4 watchdog enable This bit is used to enable the watchdog for capacitive self test step 4. 0: Disable

		1: Enable
14:8	Reserved	Must be kept at reset value.
7:0	CST4WDHT[7:0]	Capacitive test step 4 analog watchdog high threshold value These bits define the high threshold for the capacitive test step 4 analog watchdog.

13.10.39. Watchdog for capacitive self test configure register 3 (ADC_CSTWDCFG3)

Address offset: 0x634
Reset value: 0x000C 0018

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31	CST7WDEN	Capacitive self test step 7 watchdog enable This bit is used to enable the watchdog for capacitive self test step 7. 0: Disable 1: Enable
30	CSTWDTEN	Capacitive self test watchdog timer enable This bit is used to enable the self test watchdog timer for capacitive self test. The self test watchdog timer has two functions. One is to check whether the conversion order of the steps of the capacitive self test is correct. The other one is to check whether the capacitive self test was completed within the safety limit time. 0: Disable 1: Enable **Note**: This bit just used in continuous mode (The MODE bit is 1).
29:24	Reserved	Must be kept at reset value.
23:16	CST7WDHT[7:0]	Capacitive test step 7 analog watchdog high threshold value These bits define the high threshold for the capacitive test step 7 analog watchdog.
15	CST6WDEN	Capacitive self test step 6 watchdog enable This bit is used to enable the watchdog for capacitive self test step 6. 0: Disable

		1: Enable
14:8	Reserved	Must be kept at reset value.
7:0	CST6WDHT[7:0]	Capacitive test step 6 analog watchdog high threshold value These bits define the high threshold for the capacitive test step 6 analog watchdog.

13.10.40. DAC - ADC loop test configure register (ADC_DTALTCFG)

Address offset: 0x638
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LTEN[1:0]		LTEIE	LTE	Reserved				LTHT[7:0]							
rw		rw	RO					rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DAC1SEL	DAC0SEL	Reserved						LTLT[7:0]							
rw	rw							rw							

Bits	Fields	Descriptions
31:30	LTEN[1:0]	DAC - ADC loop test enable 00/10/11: Loop test disable 01: Loop test enable
29	LTEIE	Interrupt enable for LTE to FMU 0: Interrupt disable 1: Interrupt enable
28	LTE	DAC - ADC loop test error This bit indicates whether an error occurred in the DAC to ADC loop test. 0: Loop test no error 1: Loop test error Note: Writing 0 to LTEIE will clear the LTE signal
27:24	Reserved	Must be kept at reset value.
23:16	LTHT[7:0]	DAC - ADC loop test high threshold value These bits define the high threshold for DAC to ADC loop test. If ADC conversion value higher than the sum of the given value (from DAC register) and LTHT, the LTE bit is set to 1 by hardware.
15	DAC1SEL	Select DAC1_OUT0 as ADC input in DAC to ADC loop test mode 0: DAC1_OUT0 not be selected 1: DAC1_OUT0 be selected
14	DAC0SEL	Select DAC0_OUT0 as ADC input in DAC to ADC loop test mode 0: DAC0_OUT0 not be selected

		1: DAC0_OUT0 be selected
13:8	Reserved	Must be kept at reset value.
7:0	LTLT[7:0]	DAC - ADC loop test low threshold value These bits define the low threshold for DAC to ADC loop test. If ADC conversion value lower than the difference of the given value (from DAC register) and LTLT, the LTE bit is set to 1 by hardware.

14. Digital-to-analog converter (DAC)

14.1. Overview

The Digital-to-analog converter converts 12-bit digital data to a voltage on the external pins. The digital data can be configured in 8-bit or 12-bit mode, left-aligned or right-aligned mode. DMA can be used to update the digital data on external triggers.

The output voltage can be optionally buffered for higher drive capability.

The DAC supports DAC-CMP loop test and DAC-ADC loop test for safety mechanism.

14.2. Characteristics

The main features of DAC are as follows:

- 8-bit or 12-bit resolution.
- Left or right data alignment.
- DMA capability for each channel and underrun function.
- Conversion update synchronously.
- Conversion trigged by external triggers.
- Configurable internal buffer.
- Extern voltage reference, V_{REFP}.
- Noise wave (LFSR noise mode and triangle noise mode).
- Loop test: DAC-ADC and DAC-CMP.

Figure 14-1. DAC block diagram and *Table 14-1. DAC I / O description* show the block diagram of DAC and the pin description of DAC, respectively.

Figure 14-1. DAC block diagram

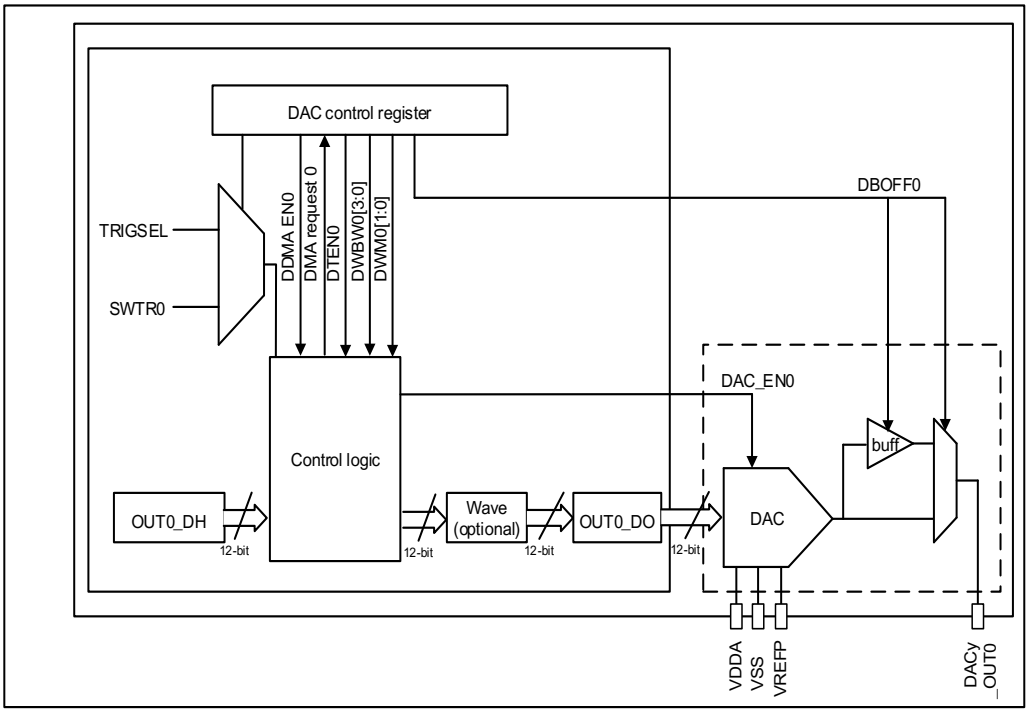


Table 14-1. DAC I / O description

Name	Description	Signal type
V_{DDA}	Analog power supply	Input, analog supply
V_{SS}	Ground for power supply	Input, supply ground
V_{REFP}	Positive reference voltage of DAC	Input, analog positive reference
DACy_OUT0	DAC analog output	Analog output

The *Table 14-2. DAC triggers and outputs summary* details the triggers and outputs of the DAC.

Table 14-2. DAC triggers and outputs summary

	DAC0	DAC1
Channel	**Channel0**	**Channel0**
DAC outputs connected to I / Os	PI9	PI10
DAC output buffer	•	
DAC trigger signals from TRIGSEL	•	
DAC software trigger	•	

Note: The GPIO pins should be configured to analog mode before enable the DAC module.

14.3. Function overview

14.3.1. DAC enable

The DAC can be turned on by setting the DEN0 bit in the DAC_CTL0 register. A t_{WAKEUP} time is needed to startup the analog DAC submodule.

14.3.2. DAC output buffer

For reducing output impedance and driving external loads without an external operational amplifier, an output buffer is integrated inside each DAC module.

The output buffer, which is turned on by default to reduce the output impedance and improve the driving capability, can be turned off by setting the DBOFF0 bits in the DAC_CTL0 register.

When DAC output buffer turns off, DAC can connect to on chip peripherals (CMP) independently by setting the DDISC0 bit in the DAC_CTL0 register.

Note: In high temperature environments, it is recommended to turn on the output buffer by setting the DBOFF0 bit to 0.

14.3.3. DAC data configuration

The 12-bit DAC holding data (OUT0_DH) can be configured by writing any one of the DAC_OUT0_R12DH, DAC_OUT0_L12DH and DAC_OUT0_R8DH registers. When the data is loaded by DAC_OUT0_R8DH register, only the MSB 8 bits are configurable, the LSB 4 bits are forced to 4'b0000.

14.3.4. DAC trigger

The DAC conversion can be triggered by software or rising edge of external trigger source. The DAC external trigger is enabled by setting the DTEN0 bits in the DAC_CTL0 register. The DAC external triggers are selected by the DTSEL0[1:0] bits in the DAC_CTL0 register, which is shown as ***Table 14-3. Triggers of DAC***.

Table 14-3. Triggers of DAC

DTSEL0[1:0]	Trigger Source	Trigger Type
2'b00	TRIGSEL	Hardware trigger
2'b01	SWTR	Software trigger
2'b10	Reserved	Reserved
2'b11		

The external trigger is generated from the TRIGSEL, while the software trigger can be generated by setting the SWTR0 bits in the DAC_SWT register.

14.3.5. DAC conversion

If the external trigger is enabled by setting the DTEN0 bit in DAC_CTL0 register, the DAC holding data is transferred to the DAC output data (DAC_OUT0_DO) register when the selected trigger event happened. When the external trigger is disabled, the transfer is performed automatically.

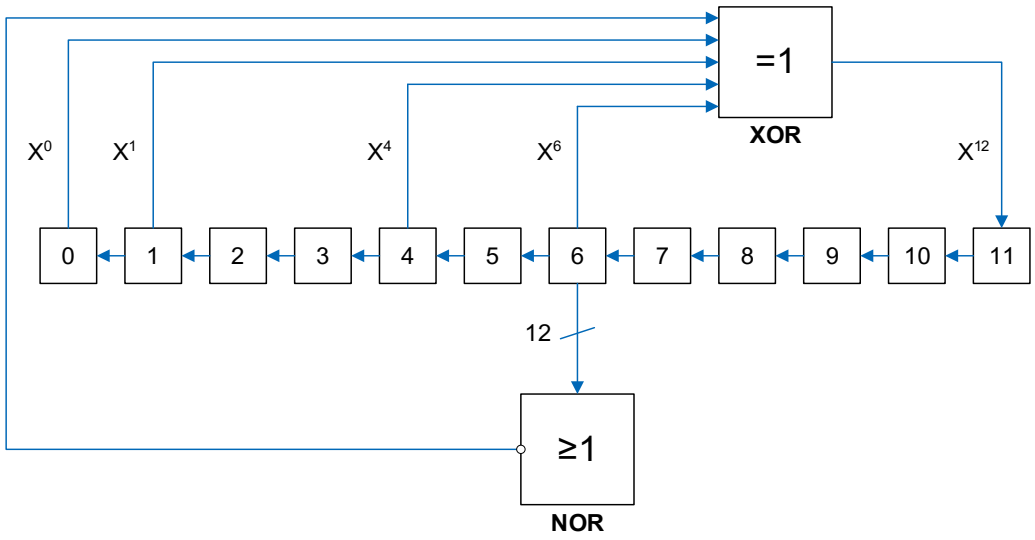
When the DAC holding data (OUT0_DH) is loaded into the DAC_OUT0_DO register, after the time $t_{SETTLING}$ which is determined by the analog output load and the power supply voltage, the analog output is valid.

14.3.6. DAC noise wave

There are two methods of adding noise wave to the DAC output data: LFSR noise wave mode and Triangle wave mode. The noise wave mode can be selected by the DWM0[1:0] bits in the DAC_CTL0 register. The amplitude of the noise can be configured by the DAC noise wave bit width (DWBW0[3:0]) bits in the DAC_CTL0 register.

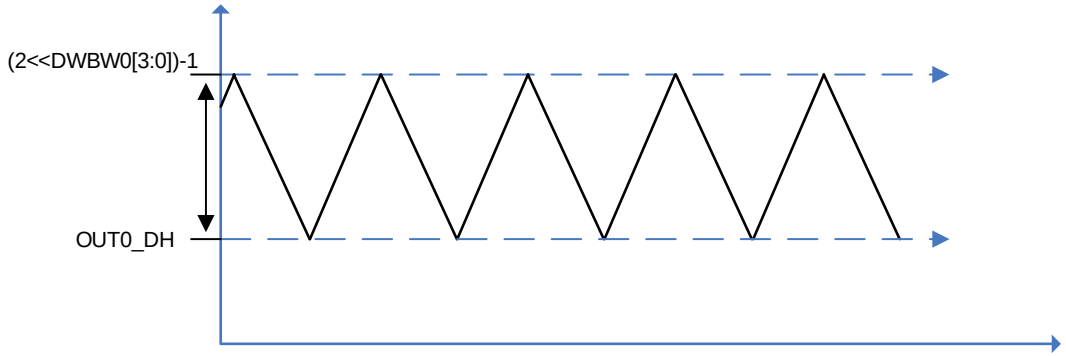
LFSR noise wave mode: there is a Linear Feedback Shift Register (LFSR) in the DAC control logic, it controls the LFSR noise signal which is added to the OUT0_DH value, and then the result is stored into the DAC_OUT0_DO register. When the configured DAC noise wave bit width is less than 12, the noise signal equals to the LSB DWBW0[3:0] bits of the LFSR register, while the MSB bits are masked.

Figure 14-2. DAC LFSR algorithm



Triangle noise mode: a triangle signal is added to the OUT0_DH value, and then the result is stored into the DAC_OUT0_DO register. The minimum value of the triangle signal is 0, while the maximum value of the triangle signal is (2 << DWBW0[3:0]) - 1.

Figure 14-3. DAC triangle noise wave



14.3.7. DAC output calculate

The analog output voltages on the DAC pin are determined by the following equation:

$$V_{DAC_OUT}=V_{REFP}*OUT0_DO/4096 \quad (15\text{-}1)$$

The digital input is linearly converted to an analog output voltage, and its range is 0 to V_{REFP}.

14.3.8. DMA request

When the external trigger is enabled, the DMA request is enabled by setting the DDMAEN0 bits of the DAC_CTL0 register. A DMA request will be generated when an external hardware trigger (not a software trigger) occurs.

If the second external trigger arrives before confirming the previous request, the new request will not be serviced, and an underrun error event occurs. The DDUDR0 bit in the DAC_STAT0 register is set, an interrupt will be generated if the DDUDRIE0 bit in the DAC_CTL0 register is set. The DMA request will be stalled until the DDUDR0 bit is cleared.

14.3.9. Loop test

The DAC supports DAC-CMP loop test and DAC-ADC loop test for safety mechanism.

DAC-ADC loop test

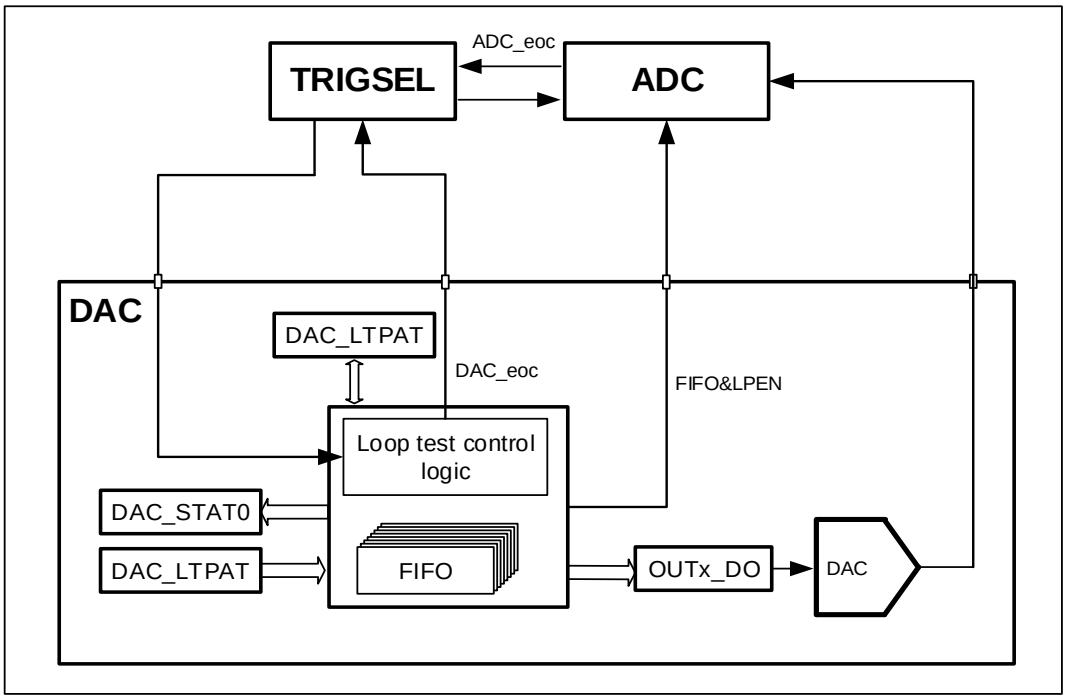
The DAC-ADC loop test is shown in ***Figure 14-4. DAC-ADC Loop test diagram***. Prepare the test patterns, and loop test supports a maximum of 8 test patterns. Writing test patterns into DAC_LTPAT register, and these patterns are transferred to the internal FIFO. The LTCNUM[3:0] bits in DAC_LTCTL register are the remaining number of loop test pattern to be converted. Setting the LTEN bit in DAC_LTCTL register enable the loop test. DAC_eoc signal is generated when DAC conversion completed, and the LTWT[7:0] bits in DAC_LTCTL register configure the waiting time for each DAC conversion. DAC_eoc signal is sent to the TRIGSEL module which triggers ADC to convert the output voltage of the DAC. After the ADC conversion is completed, the ADC_eoc signal is sent to TRIGSEL, which triggers DAC conversion for the next loop test pattern. When all loop test patterns have been executed, the

LTDIF bit in DAC_STAT0 register is set to 1 and the LTEN bit in DAC_LTCTL register is cleared. An interrupt will be generated if the LTDIE bit in the DAC_CTL0 register is set.

Setting LTFF bit in DAC_LTCTL register can flush loop test FIFO. Hardware automatically clears LTFF bit after the flush process completes.

If the number of patterns writing into DAC_LTPAT register are more than 8, only the first 8 test patterns will be executed, and the subsequent test patterns will be discarded. Simultaneously the LTOVIF bit in DAC_STAT0 register is set to 1, and an interrupt will be generated if the LTOVIE bit in the DAC_CTL0 register is set.

Figure 14-4. DAC-ADC Loop test diagram



DAC-ADC loop test configuration steps:

1. Configure ADC loop test function.
2. Write 1 to LTFF bit in DAC_LTCTL register, and wait for LTFF bit to clear to 0.
3. Write test patterns into DAC_LTPAT register.
4. Read LTCNUM[3:0] bit in DAC_LTCTL register, and check the number of test patterns. (This step is not mandatory.)
5. Configure the LTWT[7:0] bits in DAC_LTCTL register.
6. Write 1 into LTEN bit in DAC_LTCTL register to begin loop test.

DAC-CMP loop test

Enable the CMP circular mode. The DAC output is used as one input of CMP to compare with another input with different analog patterns, comparing CMPx_IP and CMPx_IM with two cases: CMPx_IM > CMPx_IP, CMPx_IM < CMPx_IP, and the CMP output indicates the comparison result.

For example, test the comparator external pin input using the DAC-CMP loop:

1. Enable the CMP circular mode by RREN bit in the CMP_RRCTL register and CMPxEN bit in the CMPx_CS register.
2. CMP_IM selects DAC as input, and the selected value should cover the case: CMP_IM > CMP_IP, CMP_IM < CMP_IP;
3. Comparator CMP_IP selects the external pin input;
4. Read CMPx_RRCOMF register to judge whether the comparison result is expected. CMP_IP and CMP_IM are known in advance. For example, the DAC outputs 0.5V while the external pin inputs 0.6V, the expected comparing result is 1.
5. If the comparing result is not expected, the software takes follow-up measures to make the chip return to the safe state.
6. Jump back to 1 for the next test.

14.4. Registers definition

DAC0 base address: 0x4000 7400

DAC1 base address: 0x4000 CC00

Table 23-39. DAC register overview

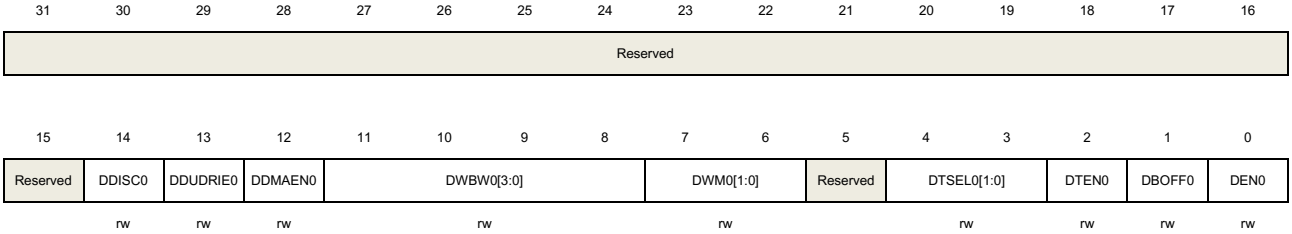
Register name	Symbol	Address offset	Reset value
DACx control register 0	DAC_CTL0	0x00	0x0000 0000
DACx software trigger register	DAC_SWT	0x04	0x0000 0000
DACx_OUT0 12-bit right-aligned data holding register	DAC_OUT0_R12DH	0x08	0x0000 0000
DACx_OUT0 12-bit left-aligned data holding register	DAC_OUT0_L12DH	0x0C	0x0000 0000
DACx_OUT0 8-bit right-aligned data holding register	DAC_OUT0_R8DH	0x10	0x0000 0000
DACx_OUT0 data output register	DAC_OUT0_DO	0x2C	0x0000 0000
DACx status register 0	DAC_STAT0	0x34	0x0000 0000
DACx Loop test control register	DAC_LTCTL	0x90	0x0000 0000
DACx Loop test pattern register	DAC_LTPAT	0x94	0x0000 0000

14.4.1. DACx control register 0 (DAC_CTL0)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:15	Reserved	Must be kept at reset value.
14	DDISC0	DACx_OUT0 connect GPIO selection 0: DAC is connected to the external pin and to on chip peripherals (CMP). 1: DAC is connected to on chip peripherals (CMP) independently only if DACx_OUT0 output buffer turns off (DBOFF=1). Otherwise DAC is connected to the external pin and to on chip peripherals (CMP).
13	DDUDRIE0	DACx_OUT0 DMA underrun interrupt enable 0: DACx_OUT0 DMA underrun interrupt disabled 1: DACx_OUT0 DMA underrun interrupt enabled

12	DDMAEN0	DACx_OUT0 DMA enable 0: DACx_OUT0 DMA mode disabled 1: DACx_OUT0 DMA mode enabled
11:8	DWBW0[3:0]	DACx_OUT0 noise wave bit width These bits specify bit width of the noise wave signal of DACx_OUT0. These bits indicate that unmask LFSR bit[n-1, 0] in LFSR noise mode or the amplitude of the triangle is ((2 << (n-1)) - 1) in triangle noise mode, where n is the bit width of wave. 0000: The bit width of the wave signal is 1 0001: The bit width of the wave signal is 2 0010: The bit width of the wave signal is 3 0011: The bit width of the wave signal is 4 0100: The bit width of the wave signal is 5 0101: The bit width of the wave signal is 6 0110: The bit width of the wave signal is 7 0111: The bit width of the wave signal is 8 1000: The bit width of the wave signal is 9 1001: The bit width of the wave signal is 10 1010: The bit width of the wave signal is 11 ≥1011: The bit width of the wave signal is 12
7:6	DWM0[1:0]	DACx_OUT0 noise wave mode These bits specify the mode selection of the noise wave signal of DACx_OUT0 when trigger of DACx_OUT0 is enabled (DTEN0 = 1). 00: Wave disabled 01: LFSR noise mode 1x: Triangle noise mode
5	Reserved	Must be kept at reset value.
4:3	DTSEL0[1:0]	DACx_OUT0 trigger selection These bits are only used if bit DTEN0 = 1 and select the external event used to trigger DAC. 00: EXTRIG (external trigger from TRIGSEL) 01: Software trigger All other values: Reserved.
2	DTEN0	DACx_OUT0 trigger enable 0: DACx_OUT0 trigger disabled 1: DACx_OUT0 trigger enabled
1	DBOFF0	DACx_OUT0 output buffer turn off 0: DACx_OUT0 output buffer turns on to reduce the output impedance and improve the driving capability 1: DACx_OUT0 output buffer turns off
0	DEN0	DACx_OUT0 enable 0: DACx_OUT0 disabled

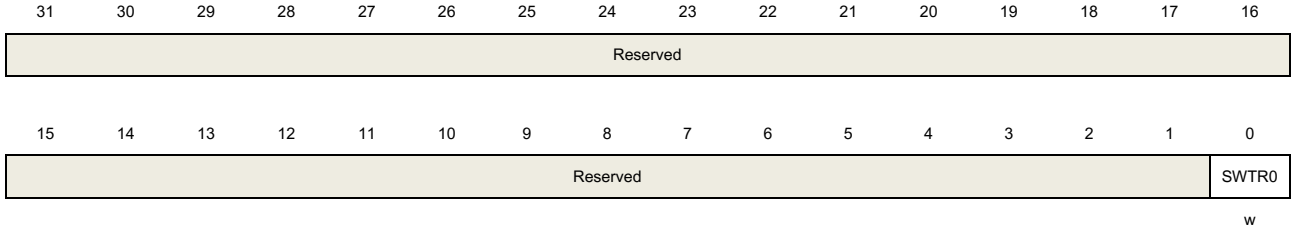
1: DACx_OUT0 enabled

14.4.2. DACx software trigger register (DAC_SWT)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



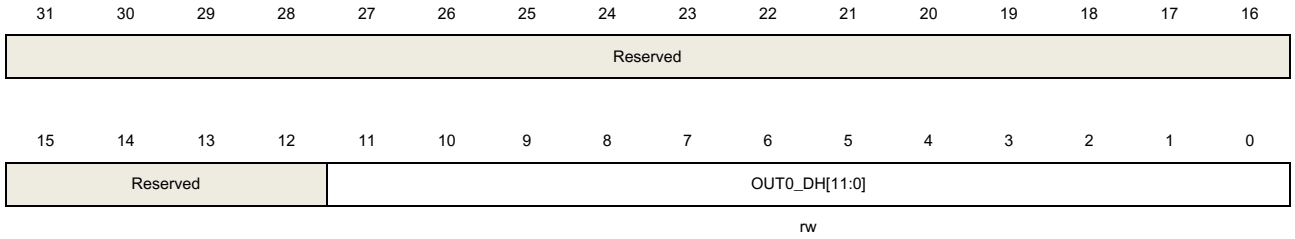
Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	SWTR0	DACx_OUT0 software trigger, cleared by hardware. 0: Software trigger disabled 1: Software trigger enabled

14.4.3. DACx_OUT0 12-bit right-aligned data holding register (DAC_OUT0_R12DH)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



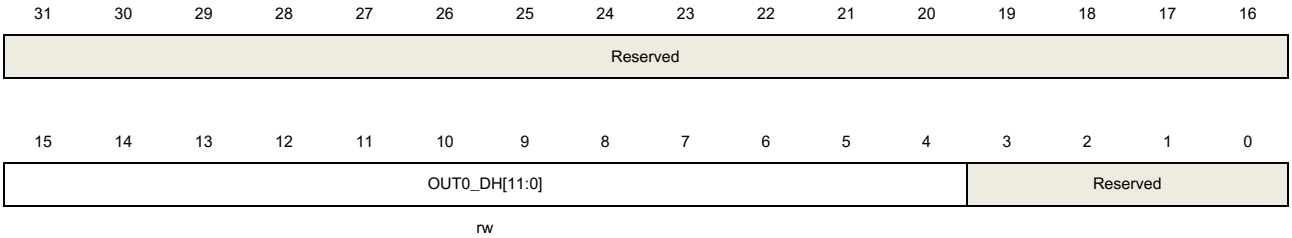
Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	OUT0_DH[11:0]	DACx_OUT0 12-bit right-aligned data. These bits specify the data that is to be converted by DACx_OUT0.

14.4.4. DACx_OUT0 12-bit left-aligned data holding register (DAC_OUT0_L12DH)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



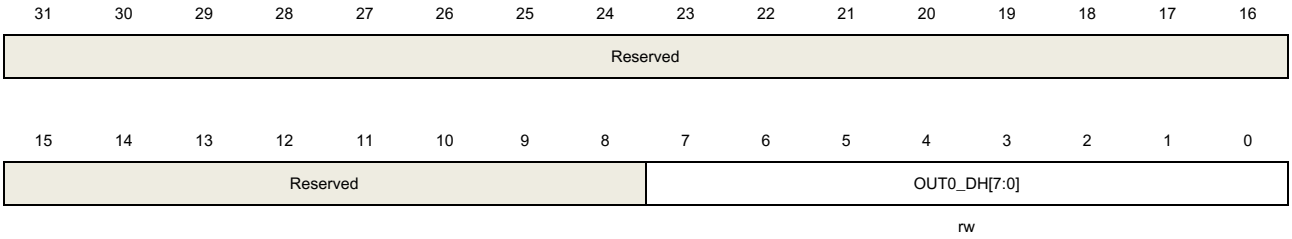
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:4	OUT0_DH[11:0]	DACx_OUT0 12-bit left-aligned data. These bits specify the data that is to be converted by DACx_OUT0.
3:0	Reserved	Must be kept at reset value.

14.4.5. DACx_OUT0 8-bit right-aligned data holding register (DAC_OUT0_R8DH)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



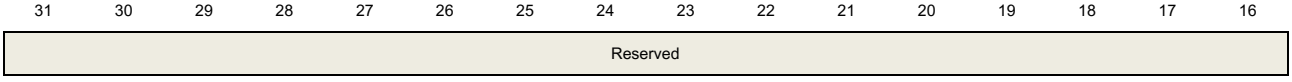
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	OUT0_DH[7:0]	DACx_OUT0 8-bit right-aligned data. These bits specify the MSB 8 bits of the data that is to be converted by DACx_OUT0.

14.4.6. DACx_OUT0 data output register (DAC_OUT0_DO)

Address offset: 0x2C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				OUT0_DO[11:0]											
				r											

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	OUT0_DO[11:0]	DACx_OUT0 data output These bits, which are read only, reflect the data that is being converted by DACx_OUT0.

14.4.7. DACx status register 0 (DAC_STAT0)

Address offset: 0x34

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DDUDR0	LTDIF	LTOVIF	Reserved										
		rc_w1	rc_w1	rc_w1											

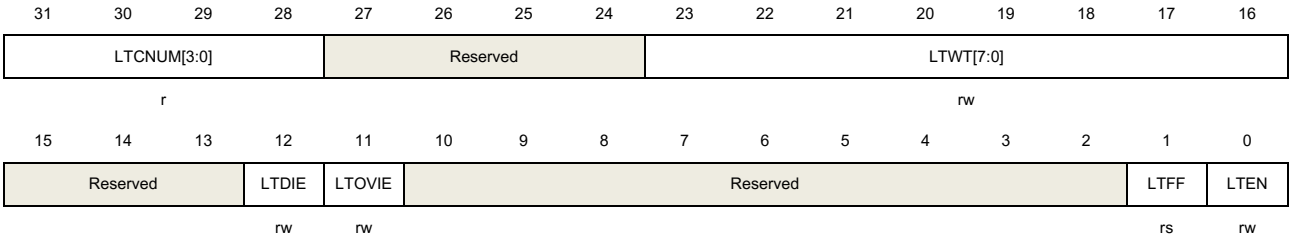
Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13	DDUDR0	DACx_OUT0 DMA underrun flag, set by hardware, cleared by software write 1. 0: No underrun occurred. 1: Underrun occurred (Speed of DAC trigger is high than the DMA transfer).
12	LTDIF	Loop test done interrupt flag, set by hardware, cleared by software write 1. 0: Loop test done not occurred 1: Loop test done occurred
11	LTOVIF	Loop test FIFO overflow interrupt flag, set by hardware, cleared by software write 1. 0: Loop test pattern FIFO overflow not occurred 1: Loop test pattern FIFO overflow occurred
10:0	Reserved	Must be kept at reset value.

14.4.8. DACx Loop test control register (DAC_LTCTL)

Address offset: 0x90

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



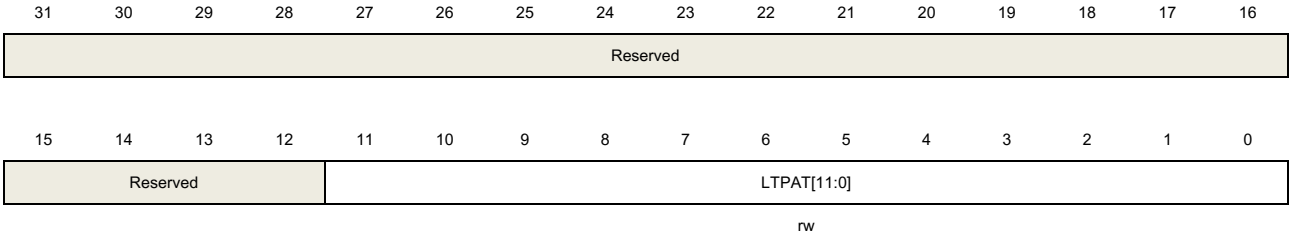
Bits	Fields	Descriptions
31:28	LTCNUM[3:0]	The remaining number of loop test pattern to be converted 0001~1000: The remaining number of loop test pattern to be converted. Others: Reserved.
27:24	Reserved	Must be kept at reset value.
23:16	LTWT[7:0]	Loop test wait time Specifies how many CK_APB1 cycles to wait before the signal of DAC_conv_done setting. 00000000: 200 CK_APB1 cycles 00000001~11111111: (LTWT[7:0] * 4) CK_APB1 cycles For example, if the wait time is 2us, and the CK_APB1 clock is 100MHz, the LTWT[7:0] should be 2us / 0.01us / 4=50(0x32)
15:13	Reserved	Must be kept at reset value.
12	LTDIE	Loop test done interrupt enable 0: Loop test done interrupt disabled 1: Loop test done interrupt enabled
11	LTOVIE	Loop test overflow interrupt enable 0: Loop test pattern FIFO overflow interrupt disabled 1: Loop test pattern FIFO overflow interrupt enabled
10:2	Reserved	Must be kept at reset value.
1	LTFF	Loop test FIFO flush Setting this bit can flush loop test FIFO. Hardware automatically clears this bit after the flush process completes. After setting this bit, application should wait until this bit is cleared before any other operation on DAC.
0	LTEN	Loop test enable 0: Loop test disable. 1: Loop test enable. When the loop test done, this bit will be cleared automatically.

14.4.9. DACx Loop test pattern register(DAC_LTPAT)

Address offset: 0x94

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	LTPAT[11:0]	Loop test pattern Every loop test pattern written to the register is automatically transferred to FIFO. Supports writing up to 8 loop test patterns at a time.

15. Watchdog timer (WDGT)

The watchdog timer (WDGT) is a hardware timing circuitry that can be used to detect system failures due to software malfunctions. There are two watchdog timer peripherals in the chip: free watchdog timer (FWDGT) and window watchdog timer (WWDGT). They offer a combination of a high safety level, flexibility of use and timing accuracy. Both watchdog timers are offered to resolve malfunctions of software.

The watchdog timer will generate a reset (or an interrupt in window watchdog timer) when the internal counter reaches a given value. The watchdog timer counter can be stopped while the processor is in the debug mode.

15.1. Free watchdog timer (FWDGT)

15.1.1. Overview

The free watchdog timer (FWDGT) has free clock source (IRC40K and LXTAL). Thereupon the FWDGT can operate even if the main clock fails. It's suitable for the situation that requires an independent environment and lower timing accuracy.

The free watchdog timer generates a reset when the internal down counter reaches 0 or the counter is refreshed when the value of the counter is greater than the window register value. The register write protection function in free watchdog can be enabled to prevent it from changing the configuration unexpectedly.

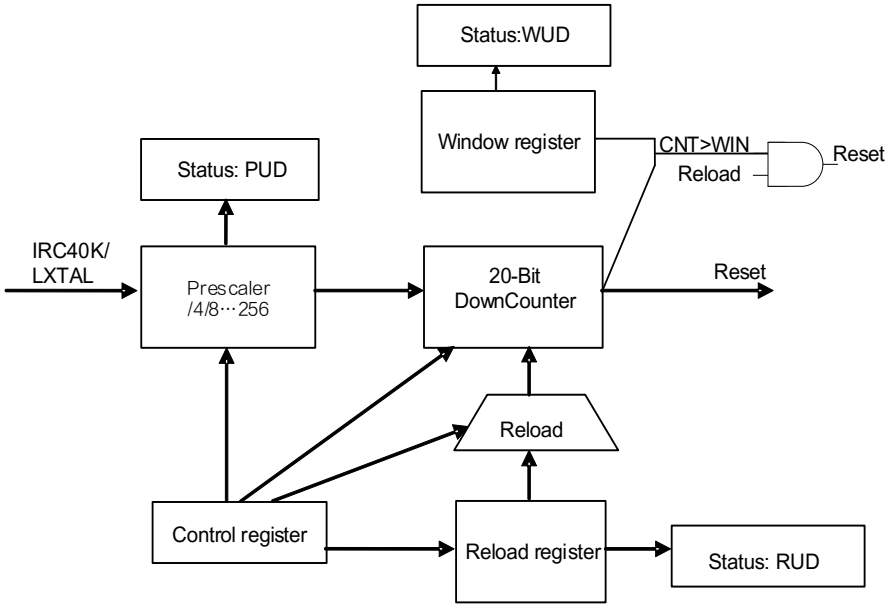
15.1.2. Characteristics

- Free-running 20-bit down counter.
- Generate reset in two conditions when FWDGT is enabled:
  - Reset when the counter reached 0.
  - The counter is refreshed when the value of the counter is greater than the window register value.
- Free clock source, FWDGT can operate even if the main clock fails such as in standby mode.
- Hardware free watchdog bit, automatically start the FWDGT or not when power on.
- FWDGT debug mode, the FWDGT can stop or continue to work in debug mode.

15.1.3. Function overview

The free watchdog consists of an 8-stage prescaler and a 20-bit down counter. ***Figure 15-1. Free watchdog block diagram*** shows the functional block of the free watchdog module.

Figure 15-1. Free watchdog block diagram



The free watchdog is enabled by writing the value (0xCCCC) to the control register (FWDGT_CTL), then counter starts counting down. When the counter reaches the value (0x0 0000), there will be a reset.

The counter can be reloaded by writing the value (0xAAAA) to the FWDGT_CTL register at any time. The reload value comes from the FWDGT_RLD register. The software can prevent the watchdog reset by reloading the counter before the counter reaches the value (0x0 0000).

By setting the appropriate window in the FWDGT_WND register, the FWDGT can also work as a window watchdog timer. A reset will occur if the reload operation is performed while the counter is greater than the value stored in the window register (FWDGT_WND). The default value of the FWDGT_WND is 0x000F FFFF, so if it is not updated, the window option is disabled. A reload operation is performed in order to reset the down counter to the FWDGT_RLD value and simplify cycle calculation to generate the next reload, as soon as the window value is changed.

The free watchdog can automatically start at power on when the hardware free watchdog bit in the device option bits is set. To avoid reset, the software should reload the counter before the counter reaches 0x0 0000.

The FWDGT_PSC register, the FWDGT_RLD register and the FWDGT_WND register are write protected. Before writing these registers, the software should write the value (0x5555) to the FWDGT_CTL register. These registers will be protected again by writing any other value to the FWDGT_CTL register. When an update operation of the prescaler register (FWDGT_PSC), window register (FWDGT_WND) or the reload value register (FWDGT_RLD) is ongoing, the status bits in the FWDGT_STAT register are set.

If the FWDGTx_HOLD (x = 0 / 1) bit in DBG module is cleared, the FWDGT continues to work even the Cortex®-M7 core halted (Debug mode). The FWDGT stops in Debug mode if the FWDGTx_HOLD (x = 0 / 1) bit is set.

Note:

1. For GD32A711/A712 devices, FWDGT1 is not included.

Table 15-1. Min/max FWDGT timeout period at 40k / 32.768kHz (IRC40K / LXTAL)

Prescaler divider	PSC[2:0] bits	Min timeout (ms) RLD[19:0]= 0x0 0000	Max timeout (s) RLD[19:0]= 0xF FFFF
1 / 4	000	0.1 / 0.122	104.8 / 128.0
1 / 8	001	0.2 / 0.244	209.7 / 256.0
1 / 16	010	0.4 / 0.488	419.4 / 512.0
1 / 32	011	0.8 / 0.976	838.8 / 1024.0
1 / 64	100	1.6 / 1.953	1677.7 / 2048.0
1 / 128	101	3.2 / 3.906	3355.4 / 4096.0
1 / 256	110 or 111	6.4 / 7.812	6710.8 / 8192.0

The FWDGT timeout can be more accurately by calibrating the IRC40K and LXTAL.

Note: When after the execution of watchdog reload operation, if the MCU needs enter the standby mode immediately, more than 3 IRC40K/LXTAL clock intervals must be inserted in the middle of reload and standby mode commands by software setting.

15.1.4. Register definition

FWDGT base address: 0x4000 3000

Table 15-1. FWDGT register overview

Register name	Symbol	Address offset	Reset value
Control register	FWDGT_CTL	0x00	0x0000 0000
Prescaler register	FWDGT_PSC	0x04	0x0000 0000
Reload register	FWDGT_RLD	0x08	0x0000 1FFF
Status register	FWDGT_STAT	0x0C	0x0000 0000
Window register	FWDGT_WND	0x10	0x000F FFFF

Control register (FWDGT_CTL)

Address offset: 0x00

Reset value: 0x0000 0000

This register can be accessed by half-word(16-bit) or word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMD[15:0]															
w															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CMD[15:0]	Write only. Several different functions are realized by writing these bits with different values. 0x5555: Disable the FWDGT_PSC, FWDGT_RLD and FWDGT_WND write protection. 0xCCCC: Start the free watchdog timer counter. When the counter reduces to 0, the free watchdog generates a reset 0xAAAA: Reload the counter

Prescaler register (FWDGT_PSC)

Address offset: 0x04

Reset value: 0x0000 0000

This register can be accessed by half-word(16-bit) or word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	PSC[2:0]
	rw

Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2:0	PSC[2:0]	Free watchdog timer prescaler selection. Write 0x5555 in the FWDGT_CTL register before writing these bits. During a write operation to this register, the PUD bit in the FWDGT_STAT register is set and the value read from this register is invalid. 000: 1 / 4 001: 1 / 8 010: 1 / 16 011: 1 / 32 100: 1 / 64 101: 1 / 128 110: 1 / 256 111: 1 / 256 If several prescaler values are used by the application, it is mandatory to wait until PUD bit has been reset before changing the prescaler value. If the prescaler value has been updated, it is not necessary to wait until PUD has been reset before continuing code execution (Before entering low-power mode, it is necessary to wait until PUD is reset).

Reload register (FWDGT_RLD)

Address offset: 0x08

Reset value: 0x0000 1FFF

This register can be accessed by half-word(16-bit) or word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												RLD[19:16]			
												rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RLD [15:0]															
rw															

Bits	Fields	Descriptions
31:20	Reserved	Must be kept at reset value.
19:0	RLD[19:0]	Free watchdog timer counter reload value. Write 0xAAAA in the FWDGT_CTL register will reload the FWDGT counter with the RLD value. These bits are write-protected. Write 0x5555 to the FWDGT_CTL register before writing these bits. During a write operation to this register, the RUD bit in the FWDGT_STAT register is set and the value read from this register is invalid. If several reload values are used by the application, it is mandatory to wait until RUD bit has been reset before changing the reload value and wait until RLD_BSY has

been reset before writing 0xAAAA to reload the counter. If the reload value has been updated, it is not necessary to wait until RUD has been reset before continuing code execution (Before entering low-power mode, it is necessary to wait until RUD is reset).

Status register (FWDGT_STAT)

Address offset: 0x0C

Reset value: 0x0000 0000

This register can be accessed by half-word(16-bit) or word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RLD_BSY	Reserved														
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													WUD	RUD	PUD
													r	r	r

Bits	Fields	Descriptions
31	RLD_BSY	The flag bit indicating that the reload command is being executed This bit is set by hardware. When this bit is set, it means the reload command is being executed. If the software needs to initiate a new reload command (i.e., write 0xAAAA to CMD), it must wait for this bit to reset to 0; otherwise, the command will be ignored.
30:3	Reserved	Must be kept at reset value.
2	WUD	Watchdog counter window value update When a write operation to FWDGT_WND register ongoing, this bit is set and the value read from FWDGT_WND register is invalid.
1	RUD	Free watchdog timer counter reload value update During a write operation to FWDGT_RLD register, this bit is set and the value read from FWDGT_RLD register is invalid.
0	PUD	Free watchdog timer prescaler value update During a write operation to FWDGT_PSC register, this bit is set and the value read from FWDGT_PSC register is invalid.

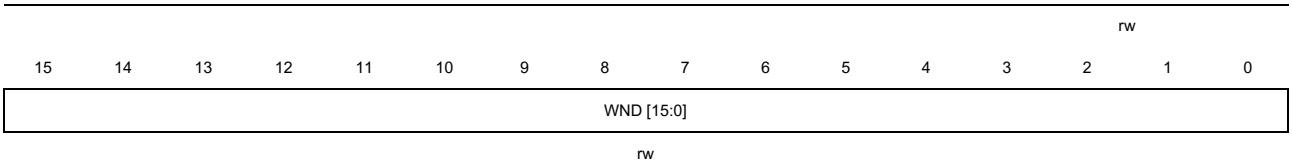
Window register (FWDGT_WND)

Address offset: 0x10

Reset value: 0x000F FFFF

This register can be accessed by half-word(16-bit) or word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												WND [19:16]			



Bits	Fields	Descriptions
31:20	Reserved	Must be kept at reset value.
19:0	WND[19:0]	Watchdog counter window value. These bits are used to contain the high limit of the window value to be compared to the downcounter. A reset will occur if the reload operation is performed while the counter is greater than the value stored in this register. The WUD bit in the FWDGT_STAT register must be reset in order to be able to change the reload value. These bits are write protected. Write 0x5555 in the FWDGT_CTL register before writing these bits. If several window values are used by the application, it is mandatory to wait until WUD bit has been reset before changing the window value. However, after updating the window value it is not necessary to wait until WUD is reset before continuing code execution except in case of low-power mode entry(Before entering low-power mode, it is necessary to wait until WUD is reset).

15.2. Window watchdog timer (WWDGT)

15.2.1. Overview

The window watchdog timer (WWDGT) is used to detect system failures due to software malfunctions. After the window watchdog timer starts, the value of down counter reduces progressively. The watchdog timer causes a reset when the counter reached 0x3F (the CNT [6] bit has been cleared). The watchdog timer also causes a reset when the counter is refreshed before the counter reached the window register value. So the software should refresh the counter in a limited window. The window watchdog timer generates an early wakeup status flag when the counter reaches 0x40, interrupt occurs if it is enabled.

The window watchdog timer clock is prescaled from the WWDGT kernel clock. The window watchdog timer is suitable for the situation that requires an accurate timing.

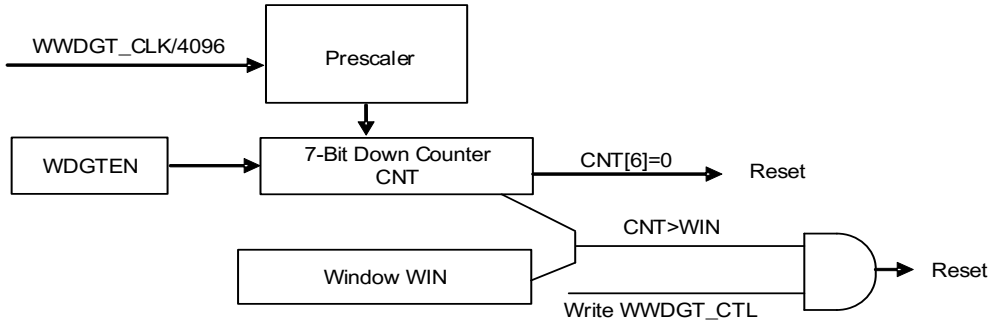
15.2.2. Characteristics

- Programmable free-running 7-bit down counter.
- Generate reset in two conditions when WWDGT is enabled:
  - Reset when the counter reached 0x3F.
  - The counter is refreshed when the value of the counter is greater than the window register value.
- Early wakeup interrupt (EWI): if the watchdog is started and the interrupt is enabled, the interrupt occurs when the counter reaches 0x40 or refreshes before it reaches the window value.
- WWDGT debug mode, the WWDGT can stop or continue to work in debug mode.
- Support clock source from IRC40K, LXTAL, HXTAL, IRC48M ,SYSTEM and APB1 clock.

15.2.3. Function overview

If the window watchdog timer is enabled (set the WDGTEN bit in the WWDGT_CTL), the watchdog timer cause a reset when the counter reaches 0x3F (the CNT[6] bit has been cleared), or the counter is refreshed before the counter reaches the window register value.

Figure 15-2. Window watchdog timer block diagram



The watchdog is always disabled after power on reset. The software starts the watchdog by setting the WDGTEN bit in the WWDGT_CTL register. When window watchdog timer is enabled, the counter counts down all the time, the configured value of the counter should be greater than 0x3F (it implies that the CNT[6] bit should be set). The CNT[5:0] determine the maximum time interval between two reloading. The count down speed depends on the WWDGT kernel clock and the prescaler (PSC[4:0] bits in the WWDGT_CFG register).

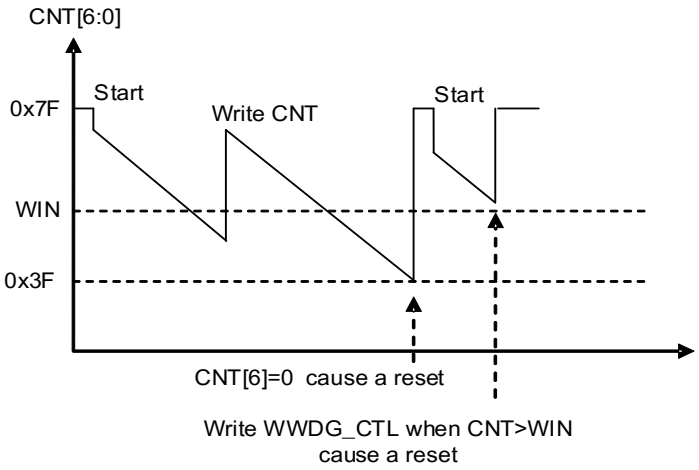
The WIN[6:0] bits in the configuration register (WWDGT_CFG) specifies the window value. The software can prevent the reset event by reloading the down counter. The counter value is less than the window value and greater than 0x3F, otherwise the watchdog causes a reset.

The early wakeup interrupt (EWI) is enabled by setting the EWIE bit in the WWDGT_CFG register, and the interrupt will be generated when the counter reaches 0x40 or the counter is refreshed before it reaches the window value. The software can do something such as communication or data logging in the interrupt service routine (ISR) in order to analyze the reason of software malfunctions or save the important data before resetting the device. Moreover, the software can reload the counter in ISR to manage a software system check and so on. In this case, the WWDGT will never generate a WWDGT reset but can be used for other things.

The EWI interrupt flag is cleared by writing '0' to the EWIF bit in the WWDGT_STAT register.

If the EWI interrupt cannot be generated normally, the EWIERR bit will be set in the WWDGT_ALMSTAT register. This error can be masked when EWIEM is set in WWDGT_ALMSTATM register. If the WWDGT cannot be active normally, the NAERR bit will be set in the WWDGT_ALMSTAT register. This error can be masked when NAEM is set in WWDGT_ALMSTATM register.

Figure 15-3. Window watchdog timing diagram



Calculate the WWDGT timeout by using the formula below.

$$t_{WWDGT}=t_{WWDGT_CLK} \times 4096 \times 2^{PSC} \times (CNT[5:0]+1) \quad (ms) \qquad (16\text{-}1)$$

where:

t_{WWDGT}: WWDGT timeout

t_{WWDGT_CLK}: WWDGT clock period measured in ms

The ***Table* 15-2. Min-max timeout value at 80 MHz (fWWDGT_CLK)** shows the minimum and maximum values of the t_{WWDGT}.

Table 15-2. Min-max timeout value at 80 MHz (f_{WWDGT_CLK})

Prescaler divider	PSC[4:0]	Min timeout value CNT[6:0] =0x40	Max timeout value CNT[6:0]=0x7F
1/1	00000	51.2μs	3.2768ms
1/2	00001	102.4μs	6.5536ms
1/4	00010	204.8μs	13.1072ms
1/8	00011	409.6μs	26.2144ms
1/16	00100	819.2μs	52.4288ms
1/32	00101	1638.4μs	104.8576ms
1/64	00110	3276.8μs	209.7152ms
1/128	00111	6553.6μs	419.4304ms
1/256	01000	13107.2μs	838.8608ms
1/512	01001	26214.4μs	1677.7216ms
1/1024	01010	52428.8μs	3355.4432ms
1/2048	01011	104857.6μs	6710.8864ms
1/4096	01100	209715.2μs	13421.7728ms
1/8192	01101	419430.4μs	26843.5456ms
1/16384	01110	838860.8μs	53687.0912ms
1/32768	01111	1677721.6μs	107374.1824ms
1/65536	10000	3355443.2μs	214748.3648ms
1/131072	10001	6710886.4μs	429496.7296ms

If the WWDGTx_HOLD (x = 0 / 1) bit in DBG module is cleared, the WWDGT continues to work even the Cortex®-M7 core halted (Debug mode). While the WWDGTx_HOLD (x = 0 / 1) bit is set, the WWDGT stops in Debug mode.

Note: For GD32A711/A712 devices, WWDGT1 is not included.

15.2.4. Register definition

WWDGT base address: 0x4000 2C00

Table 15-2. WWDGT register overview

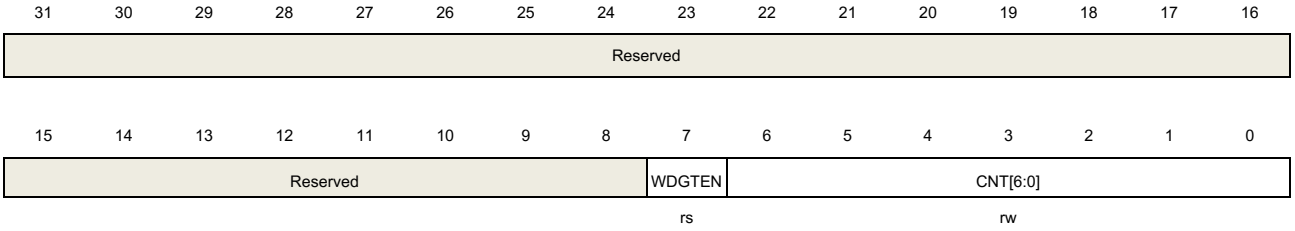
Register name	Symbol	Address offset	Reset value
Control register	WWDGT_CTL	0x00	0x0000 007F
Configuration register	WWDGT_CFG	0x04	0x0000 007F
Status register	WWDGT_STAT	0x08	0x0000 0000
Alarm status register	WWDGT_ALMSTAT	0x0C	0x0000 0000
Alarm status mask register	WWDGT_ALMSTATM	0x10	0x0000 0003

Control register (WWDGT_CTL)

Address offset: 0x00

Reset value: 0x0000 007F

This register can be accessed by half-word(16-bit) or word(32-bit).



Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	WDGTEN	Start the Window watchdog timer. Cleared by a hardware reset. Writing 0 has no effect. 0: Disable window watchdog timer 1: Enable window watchdog timer
6:0	CNT[6:0]	The value of the watchdog timer counter. A reset occurs when the value of this counter decreases from 0x40 to 0x3F. When the value of this counter is greater than the window value, writing this counter also causes a reset.

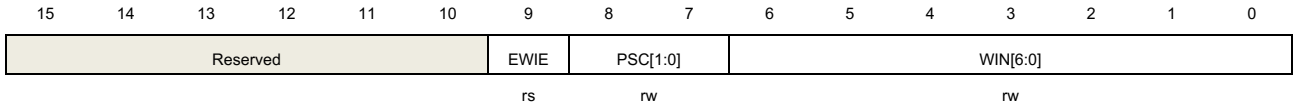
Configuration register (WWDGT_CFG)

Address offset: 0x04

Reset value: 0x0000 007F

This register can be accessed by half-word(16-bit) or word(32-bit).



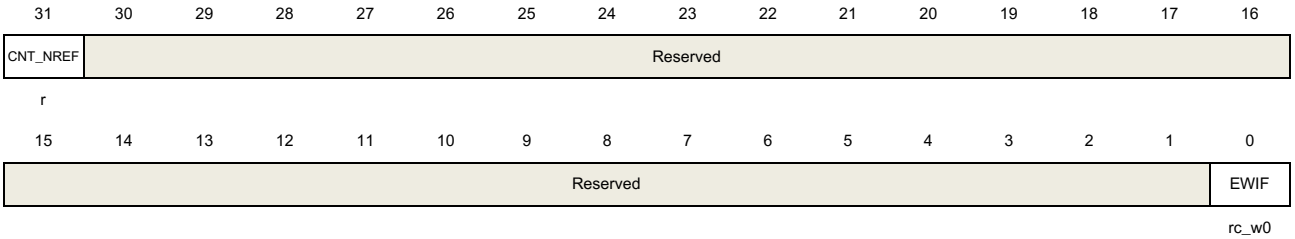


Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18:16	PSC[4:2]	PSC[4:2], refer to PSC[1:0] description.
15:10	Reserved	Must be kept at reset value.
9	EWIE	Early wakeup interrupt enable. If the bit is set, an interrupt occurs when the counter reaches 0x40 or refreshes before it reaches the window value. It can be cleared by a hardware reset or software clock reset in RCU module. A write operation of 0 has no effect.
8:7	PSC[1:0]	Prescaler. Combined with PSC[4:2] determine the time base of the watchdog timer counter. 00000: (WWDGT_CLK / 4096) / 1 00001: (WWDGT_CLK / 4096) / 2 00010: (WWDGT_CLK / 4096) / 4 00011: (WWDGT_CLK / 4096) / 8 00100: (WWDGT_CLK / 4096) / 16 00101: (WWDGT_CLK / 4096) / 32 00110: (WWDGT_CLK / 4096) / 64 00111: (WWDGT_CLK / 4096) / 128 01000: (WWDGT_CLK / 4096) / 256 01001: (WWDGT_CLK / 4096) / 512 01010: (WWDGT_CLK / 4096) / 1024 01011: (WWDGT_CLK / 4096) / 2048 01100: (WWDGT_CLK / 4096) / 4096 01101: (WWDGT_CLK / 4096) / 8192 01110: (WWDGT_CLK / 4096) / 16384 01111: (WWDGT_CLK / 4096) / 32768 10000: (WWDGT_CLK / 4096) / 65536 10001: (WWDGT_CLK / 4096) / 131072 10010~11111: (WWDGT_CLK / 4096) / 1
6:0	WIN[6:0]	The Window value. A reset occurs if the watchdog counter (CNT bits in WWDGT_CTL) is written when the value of the watchdog counter is greater than the Window value.

Status register (WWDGT_STAT)

Address offset: 0x08
Reset value: 0x0000 0000

This register can be accessed by half-word(16-bit) or word(32-bit).



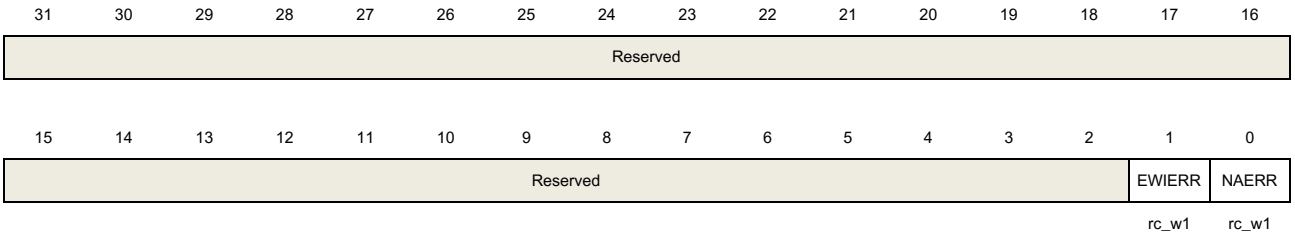
Bits	Fields	Descriptions
31	CNT_NREF	This flag is set by hardware. It indicates that it is recommended not to refresh the CNT[6:0] value in WWDGT_CTL register when this flag is set, otherwise some errors may occur.
30:1	Reserved	Must be kept at reset value.
0	EWIF	Early wakeup interrupt flag. When the counter reaches 0x40 or refreshes before it reaches the window value, this bit is set by hardware even the interrupt is not enabled (EWIE in WWDGT_CFG is cleared). This bit is cleared by writing 0. There is no effect when writing 1.

Alarm status register (WWDGT_ALMSTAT)

Address offset: 0x0C

Reset value: 0x0000 0000

This register can be accessed by half-word(16-bit) or word(32-bit).



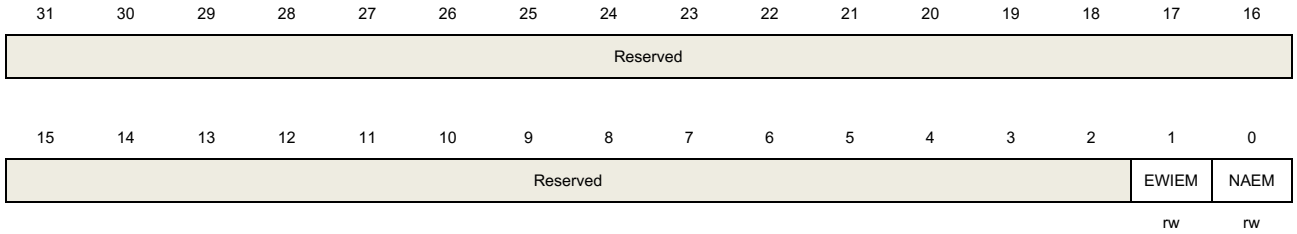
Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	EWIERR	For the key register WWDGT_CFG, the bit9:EWIE TMR monitoring status. This bit is cleared by writing 1. 0: The EWIE bit has no TMR error. 1: The EWIE bit has a TMR error.
0	NAERR	For the key register WWDGT_CTL, the bit7:WDGTEN TMR monitoring status. This bit is cleared by writing 1. 0: The WDGTEN bit has no TMR error. 1: The WDGTEN bit has a TMR error.

Alarm status mask register (WWDGT_ALMSTATM)

Address offset: 0x10

Reset value: 0x0000 0003

This register can be accessed by half-word(16-bit) or word(32-bit).



Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	EWIEM	EWIERR status mask bit If this bit is set, EWIERR cannot be transmitted to FMU. Write 1 to mask EWIERR bit in WWDGT_ALMSTAT register.
0	NAEM	NAERR status mask bit If this bit is set, NAERR cannot be transmitted to FMU. Write 1 to mask NAERR bit in WWDGT_ALMSTAT register.

16. Real-time clock (RTC)

16.1. Overview

The RTC is usually used as a clock-calendar. The RTC circuits are located in two power supply domains. The ones in the Backup Domain consist of a 32-bit up-counter, an alarm, a prescaler, a divider and the RTC clock configuration register. That means the RTC settings and time are kept when the device resets or wakes up from Standby mode. While the circuits in the V11 domain only include the APB interface and a control register. In the following sections, the details of the RTC function will be described.

16.2. Characteristics

- 32-bit programmable counter for counting elapsed time.
- Programmable prescaler: Max division factor is up to 2^{20}.
- Separate clock domains:
  - PCLK1 clock domain.
  - RTC clock domain (this clock must be at least 4 times slower than the PCLK1 clock).
- RTC clock source:
  - HXTAL clock divided by 128.
  - LXTAL oscillator clock.
  - IRC40K oscillator clock.
- Maskable interrupt source:
  - Alarm interrupt.
  - Second interrupt.
  - Overflow interrupt.

16.3. Function overview

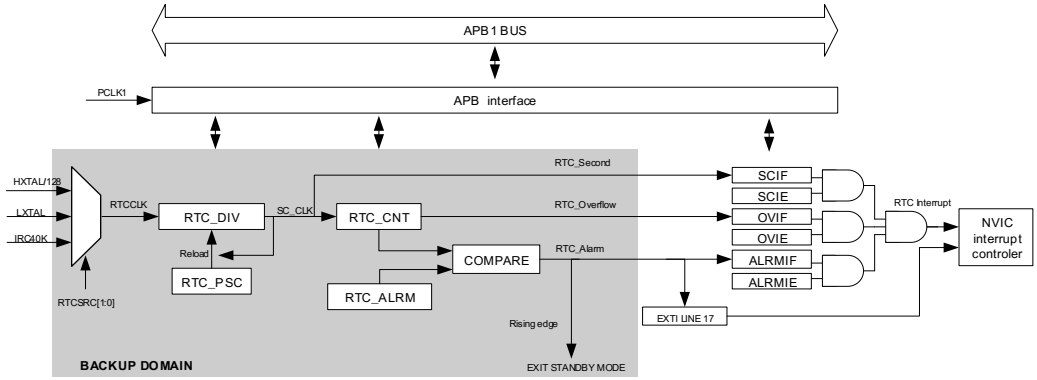
The RTC circuits consist of two major units: APB interface located in PCLK1 clock domain and RTC core located in RTC clock domain.

APB Interface is connected with the APB1 bus. It includes a set of registers, can be accessed by APB1 bus.

RTC core includes two major blocks. One is the RTC prescaler block, which generates the RTC time base clock SC_CLK. RTC prescaler block includes a 20-bit programmable divider (RTC prescaler) which can make SC_CLK is divided from RTC source clock. If second interrupt is enabled in the RTC_INTEN register, the RTC will generate an interrupt at every SC_CLK rising edge. Another block is a 32-bit programmable counter, which can be initialized with the value of current system time. If alarm interrupt is enabled in the RTC_INTEN register, the RTC will generate an alarm interrupt when the system time equals to the alarm time

(stored in the RTC_ALRMH / L register).

Figure 16-1. Block diagram of RTC



16.3.1. RTC reset

The APB interface, RTC_INTEN register and RTC_CTL register are reset by system reset. The RTC core (prescaler, divider, counter and alarm) is reset only by a backup domain reset.

Steps to enable access to the backup registers and the RTC after reset are as follows:

1. Set the PMUEN and BKPEN bits in the RCU_APB1EN register to enable the power and backup interface clocks;
2. Enable access to the backup registers and RTC by setting the BKPWEN bit in the PMU_CTL.

16.3.2. RTC reading

The APB interface and RTC core are located in two different power supply domains.

In the RTC core, only counter and divider registers are readable registers. And the values in the two registers and the RTC flags are internally updated at each rising edge of the RTC clock, which is resynchronized by the APB1 clock.

When the APB interface is immediately enabled from a disable state, the read operation is not recommended because the first internal update of the registers has not finished. That means, when a system reset or power reset, the APB interface was in disabled state, but the RTC core has been kept running. In these cases, the correct read operation should first clear the RSYNF bit in the RTC_CTL register and wait for it to be set by hardware. While WFI and WFE have no effects on the RTC APB interface.

16.3.3. RTC configuration

The RTC_PSC, RTC_CNT and RTC_ALRM registers in the RTC core are writable. These registers' value can be set only when the peripheral enters configuration mode. And the CMF bit in the RTC_CTL register is used to indicate the configuration mode status. The write

operation executes when the peripheral exit configuration mode, and it takes at least three RTCCLK cycles to complete. The value of the LWOFF bit in the RTC_CTL register sets to '1', if the write operation finished. The new write operation should wait for the previous one finished.

Disable the interrupt function in RTC_INTEN register before the configuration. After exiting the configuration, check the flags and enable the interrupt function in RTC_INTEN register.

The configuration steps are as follows:

1. Wait until the value of LWOFF bit in the RTC_CTL register sets to '1';
2. Enter configuration mode by setting the CMF bit in the RTC_CTL register;
3. Write to the RTC registers;
4. Exit configuration mode by clearing the CMF bit in the RTC_CTL register;
5. Wait until the value of LWOFF bit in the RTC_CTL register sets to '1'.

16.3.4. RTC flag assertion

Before the update of the RTC counter, the RTC second interrupt flag (SCIF) is asserted on the last RTCCLK cycle.

Before the counter equals the RTC alarm value stored in the alarm register plus one, the RTC alarm interrupt flag (ALRMIF) is asserted on the last RTCCLK cycle.

Before the counter equals to 0x00, the RTC overflow interrupt flag (OVIF) is asserted on the last RTCCLK cycle.

The RTC alarm write operation and second interrupt flag must be synchronized by using either of the following sequences:

- Use the RTC alarm interrupt and update the RTC alarm and / or RTC counter registers inside the RTC interrupt routine.
- Update the RTC alarm and / or the RTC counter registers after the SCIF bit to be set in the RTC control register.

Figure 16-2. RTC second and alarm waveform example (RTC_PSC = 3, RTC_ALRM = 2)

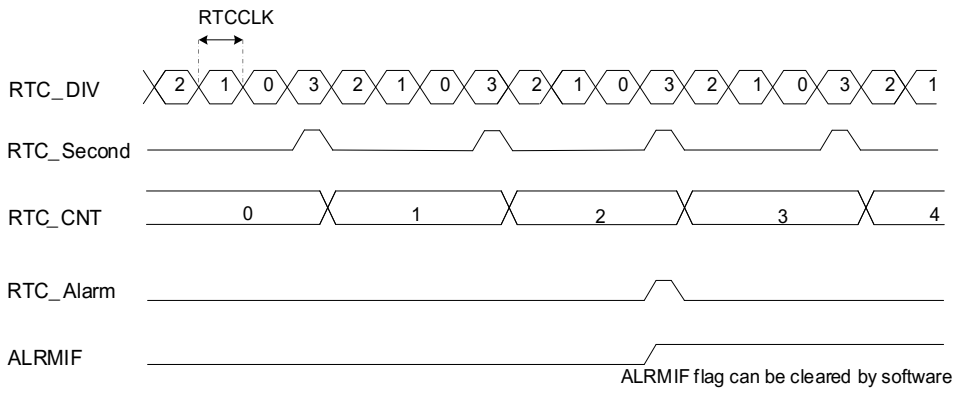
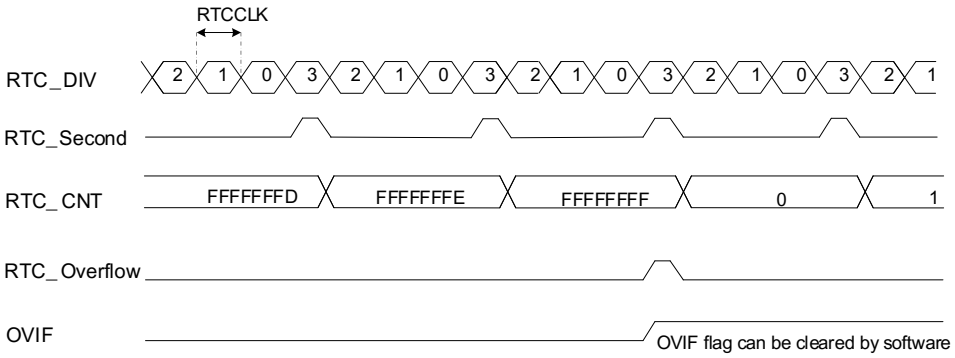


Figure 16-3. RTC second and overflow waveform example (RTC_PSC= 3)



16.4. Register definition

RTC base address: 0x4000 2800

Table 16-1. RTC register overview

Register Name	Symbol	Address Offset	Reset Value
interrupt enable register	RTC_INTEN	0x00	0x0000 0000
Control register	RTC_CTL	0x04	0x0000 0020
prescaler high register	RTC_PSCH	0x08	0x0000 0000
prescaler low register	RTC_PSCL	0x0C	0x0000 8000
divider high register	RTC_DIVH	0x10	0x0000 0000
divider low register	RTC_DIVL	0x14	0x0000 8000
counter high register	RTC_CNTH	0x18	0x0000 0001
counter low register	RTC_CNTL	0x1C	0x0000 0000
alarm high register	RTC_ALRMH	0x20	0x0000 FFFF
alarm low register	RTC_ALRML	0x24	0x0000 FFFF

16.4.1. RTC interrupt enable register (RTC_INTEN)

Address offset: 0x00

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													OVIE	ALRMIE	SCIE
													rw	rw	rw

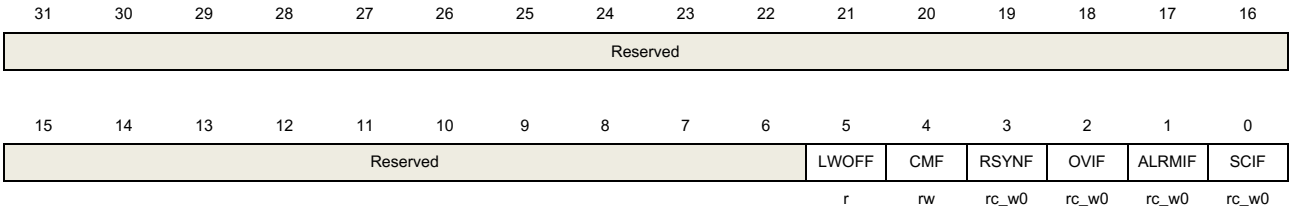
Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2	OVIE	Overflow interrupt enable 0: Disable overflow interrupt 1: Enable overflow interrupt
1	ALRMIE	Alarm interrupt enable 0: Disable alarm interrupt 1: Enable alarm interrupt
0	SCIE	Second interrupt enable 0: Disable second interrupt 1: Enable second interrupt

16.4.2. RTC control register (RTC_CTL)

Address offset: 0x04

Reset value: 0x0000 0020

This register can be accessed by half-word (16-bit) or word (32-bit).



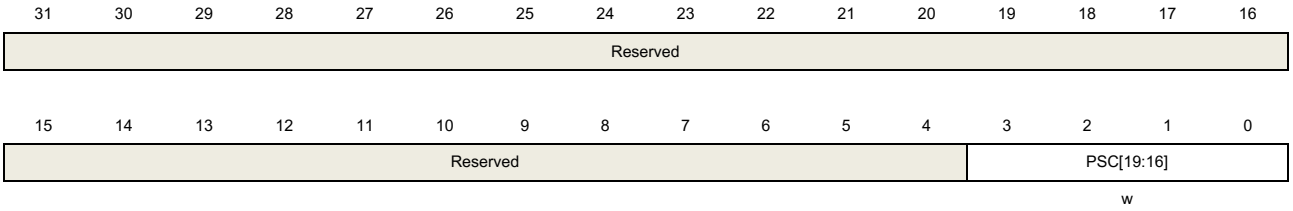
Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5	LWOFF	Last write operation finished flag 0: Last write operation on RTC registers did not finished 1: Last write operation on RTC registers finished
4	CMF	Configuration mode flag 0: Exit configuration mode 1: Enter configuration mode
3	RSYNF	Registers synchronized flag 0: Registers not yet synchronized with the APB1 clock 1: Registers synchronized with the APB1 clock
2	OVIF	Overflow interrupt flag 0: Overflow event not detected 1: Overflow event detected. An interrupt will occur if the OVIE bit is set in RTC_INTEN.
1	ALRMIF	Alarm interrupt flag 0: Alarm event not detected 1: Alarm event detected. An interrupt named RTC global interrupt will occur if the ALRMIE bit is set in RTC_INTEN. And another interrupt named the RTC alarm interrupt will occur if the EXTI 17 is enabled in interrupt mode.
0	SCIF	Second interrupt flag 0: Second event not detected 1: Second event detected. An interrupt will occur if the SCIE bit is set in RTC_INTEN. Set by hardware when the divider reloads the value in RTC_PSCH / L, thus incrementing the RTC counter.

16.4.3. RTC prescaler high register (RTC_PSCH)

Address offset: 0x08

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).



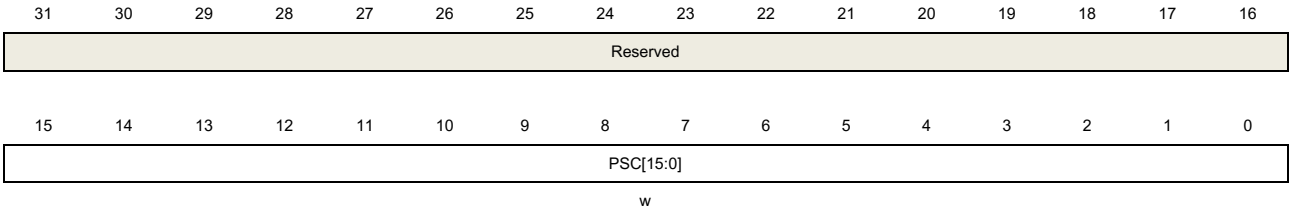
Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3:0	PSC[19:16]	RTC prescaler value high **Note:** PSC[19:0] bits cannot be set to 0.

16.4.4. RTC prescaler low register (RTC_PSCL)

Address offset: 0x0C

Reset value: 0x0000 8000

This register can be accessed by half-word (16-bit) or word (32-bit).



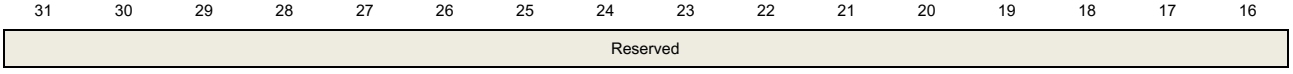
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	PSC[15:0]	RTC prescaler value low The frequency of SC_CLK is the RTCCLK frequency divided by (PSC[19:0]+1). **Note:** PSC[19:0] bits cannot be set to 0.

16.4.5. RTC divider high register (RTC_DIVH)

Address offset: 0x10

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												DIV[19:16]			
r															

Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3:0	DIV[19:16]	RTC divider value high

16.4.6. RTC divider low register (RTC_DIVL)

Address offset: 0x14

Reset value: 0x0000 8000

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIV[15:0]															
r															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	DIV[15:0]	RTC divider value low The RTC divider register is reloaded by hardware when the RTC prescaler or RTC counter register is updated.

16.4.7. RTC counter high register (RTC_CNTH)

Address offset: 0x18

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[31:16]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CNT[31:16]	RTC counter value high

16.4.8. RTC counter low register (RTC_CNTL)

Address offset: 0x1C

Reset value: 0x0000 0000

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CNT[15:0]	RTC counter value low

16.4.9. RTC alarm high register (RTC_ALRMH)

Address offset: 0x20

Reset value: 0x0000 FFFF

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALRM[31:16]															
w															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	ALRM[31:16]	RTC alarm value high

16.4.10. RTC alarm low register (RTC_ALRML)

Address offset: 0x24

Reset value: 0x0000 FFFF

This register can be accessed by half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALRM[15:0]															
w															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	ALRM[15:0]	RTC alarm value low

17. TIMER

Table 17-1. Timers (Timer) are divided into three sorts

TIMER	TIMER0/7/19/20/60/61/62/63	TIMER1/2	TIMER5/6
TYPE	Advanced	General-L0	Basic
Presale	16-bit	16-bit	16-bit
Counter	16-bit	16-bit	32-bit
Count mode	UP, DOWN, Center-aligned	UP, DOWN, Center-aligned	UP ONLY
Repetition	●	×	×
Channel Capture/ Compare	8	4	0
Complementary & Dead-time	●	×	×
Break	●	×	×
Single Pulse	●	●	●
Quadrature Decoder	●	●	×
Master-slave management	●	●	×
Inter Connection	● (1)	● (1)	TRIO TO DAC
DMA	●	●	● (2)
Debug Mode	●	●	●

●: Support.

×: Not support.

(1) Please refer to TRIGSEL for more details.

(2) Only update events will generate a DMA request. TIMER5/6 do not have DMAS bit (DMA request source selection).

TIMERx	ITI0	ITI1	ITI2	ITI3	ITI4	ITI5	ITI6	ITI7	ITI8	ITI9
TIMER0	TRGSEL	TIMER7_TRGO0	TIMER19_TRGO0	TIMER20_TRGO0	TIMER60_TRGO0	TIMER61_TRGO0	TIMER62_TRGO0	TIMER63_CH0	TIMER1_CH0	TIMER2_TRGO0
TIMER7		TIMER0_TRGO0	TIMER19_TRGO0	TIMER20_TRGO0	TIMER60_TRGO0	TIMER61_TRGO0	TIMER62_TRGO0	TIMER63_CH0	TIMER1_CH0	TIMER2_TRGO0
TIMER19		TIMER0_TRGO0	TIMER7_TRGO0	TIMER20_TRGO0	TIMER60_TRGO0	TIMER61_TRGO0	TIMER62_TRGO0	TIMER63_CH0	TIMER1_CH0	TIMER2_TRGO0

TIMERx	ITI0	ITI1	ITI2	ITI3	ITI4	ITI5	ITI6	ITI7	ITI8	ITI9
TIMER20		TIMER0_TRGO0	TIMER7_TRGO0	TIMER19_TRGO0	TIMER60_TRGO0	TIMER61_TRGO0	TIMER62_TRGO0	TIMER63_CH0	TIMER1_CH0	TIMER2_TRGO0
TIMER60		TIMER0_TRGO0	TIMER7_TRGO0	TIMER19_TRGO0	TIMER20_TRGO0	TIMER61_TRGO0	TIMER62_TRGO0	TIMER63_CH0	TIMER1_CH0	TIMER2_TRGO0
TIMER61		TIMER0_TRGO0	TIMER7_TRGO0	TIMER19_TRGO0	TIMER20_TRGO0	TIMER60_TRGO0	TIMER62_TRGO0	TIMER63_CH0	TIMER1_CH0	TIMER2_TRGO0
TIMER62		TIMER0_TRGO0	TIMER7_TRGO0	TIMER19_TRGO0	TIMER20_TRGO0	TIMER60_TRGO0	TIMER61_TRGO0	TIMER63_CH0	TIMER1_CH0	TIMER2_TRGO0
TIMER63		TIMER0_TRGO0	TIMER7_TRGO0	TIMER19_TRGO0	TIMER20_TRGO0	TIMER60_TRGO0	TIMER61_TRGO0	TIMER62_CH0	TIMER1_CH0	TIMER2_TRGO0
TIMER1		TIMER0_TRGO0	TIMER7_TRGO0	TIMER19_TRGO0	TIMER20_TRGO0	TIMER60_TRGO0	TIMER61_TRGO0	TIMER62_CH0	TIMER63_CH0	TIMER2_TRGO0
TIMER2		TIMER0_TRGO0	TIMER7_TRGO0	TIMER19_TRGO0	TIMER20_TRGO0	TIMER60_TRGO0	TIMER61_TRGO0	TIMER62_CH0	TIMER63_CH0	TIMER1_TRGO0

17.1. Advanced timer (TIMERx, x=0, 7, 19, 20, 60, 61, 62, 63)

17.1.1. Overview

The advanced timer module (TIMER0/7/19/20/60/61/62/63) is an eight-channel timer that supports both input capture and output compare. They can generate PWM signals to control motor or be used for power management applications. The advanced timer has a 16-bit counter that can be used as an unsigned counter.

In addition, the advanced timers can be programmed and be used for counting. their external events can be used to drive other timers.

Timer also includes a dead-time insertion module which is suitable for motor control applications.

Timers are completely independent with each other, but they may be synchronized to provide a larger timer with their counter value increasing in unison.

17.1.2. Characteristics

- Total channel num: 8.
- Counter width: 16 bits.
- Selectable clock source: internal clock, internal trigger, external input, external trigger.
- Multiple counter modes: up counting, down counting and center-aligned counting.
- Quadrature decoder: used for motion tracking and determination of both rotation direction and position.
- Hall sensor function: used for 3-phase motor control.
- Programmable prescaler: 16 bits. The factor can be changed ongoing.
- Each channel is independent and user-configurable: input capture mode, output compare mode, programmable PWM mode, composite PWM mode, asymmetric PWM mode, single pulse mode and trigger out.
- Input capture mode supports multi edge capture trigger, and trigger edge can be configured by software.
- Programmable dead time insertion and separated dead time insertion. Dead time can be changed when timer is running.
- Auto reload function.
- Programmable counter repetition function.
- Break input.
- The initial counter of timer can be refresh.
- Interrupt output or DMA request: update event, trigger event, compare/capture event, commutation event and break input.
- Daisy chaining of timer module allows a single timer to start multiple timers.
- Timer synchronization allows the selected timers to start counting on the same clock cycle.
- Timer master-slave management.

■ The counter is frozen when break is enabled.

17.1.3. Block diagram

Figure 17-1. Advanced timer block diagram provides details of the internal configuration of the advanced timer, and *Table 17-2. Advanced timer channel description* introduces the input and output of the channels.

Figure 17-1. Advanced timer block diagram

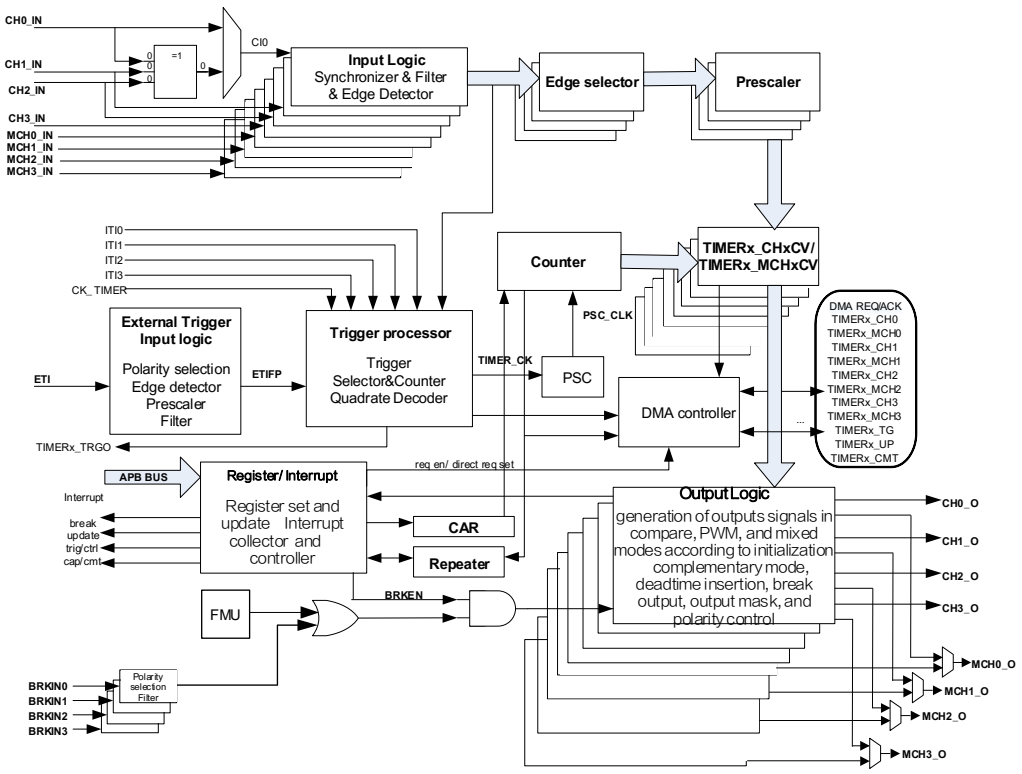


Table 17-2. Advanced timer channel description

Channel name (x=0..3)	MCHxMSEL[1:0]=00 independent mode	MCHxMSEL[1:0]=01 mirrored mode	MCHxMSEL[1:0]=11 complementary mode
CHx (Channel x)	CHx and MCHx can independently input capture and compare output	CHx output is the same as MCHx output (just used for output).	only the CHx is valid for input, and the outputs of MCHx and CHx are complementary
MCHx (Multi mode channel x)			

17.1.4. Function overview

Clock source configuration

The clock source of the advanced timer can be either the CK_TIMER or an alternate clock source controlled by TSCFGy[4:0] (y=0..11,15) in SYSCFG_TIMERxCFG

(x=0,7,19,20,60,61,62,63)

if TSCFGy[4:0] (y=0..2,6,8,9,10,11,15) !=5'b00000 in SYSCFG_TIMERxCFG (x=0,7,19,20,60,61,62,63), the prescaler is clocked by other clock sources selected in the TSCFGy[4:0] register, more details will be introduced later. When the TSCFGy[4:0] (y=3,4,5,7) are setting to an available value, the internal clock TIMER_CK is the counter prescaler driving clock source.

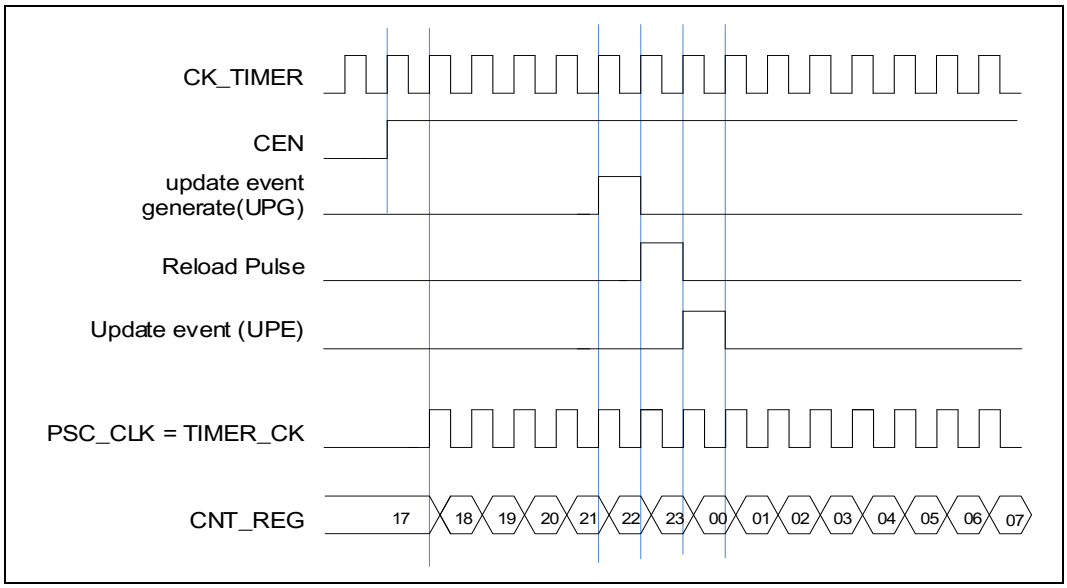
Internal clock mode

- TSCFGy[4:0] = 5'b00000 in SYSCFG_TIMER0CFG(y=0..11,15) Internal clock CK_TIMER is selected as timer clock source which is from module RCU.

The default clock source is the CK_TIMER for driving the counter prescaler when TSCFGy[4:0] (y=0..11,15).= 5'b00000 in SYSCFG_TIMERxCFG(x=0,7,19,20,60,61,62,63) When the CEN is set, the CK_TIMER will be divided by PSC value to generate PSC_CLK.

In this mode, the TIMER_CK which drives counter's prescaler to count is equal to CK_TIMER which is from RCU module.

Figure 17-2. Timing chart of internal clock divided by 1



External clock mode

- TSCFG6[4:0] are setting to an available value (external clock mode 0). External input pin is selected as timer clock source.

The TIMER_CK, which drives counter's prescaler to count, can be triggered by the event of rising or falling edge on the external pin TIMERx_CH0/TIMERx_CH1. This mode can be selected by setting TSCFG6[4:0] to 0x15, 0x16, 0x17.

And, the counter prescaler can also be driven by rising edge on the internal trigger input pin ITI0..9. This mode can be selected by setting TSCFG6[4:0] to 0x1~0x9.

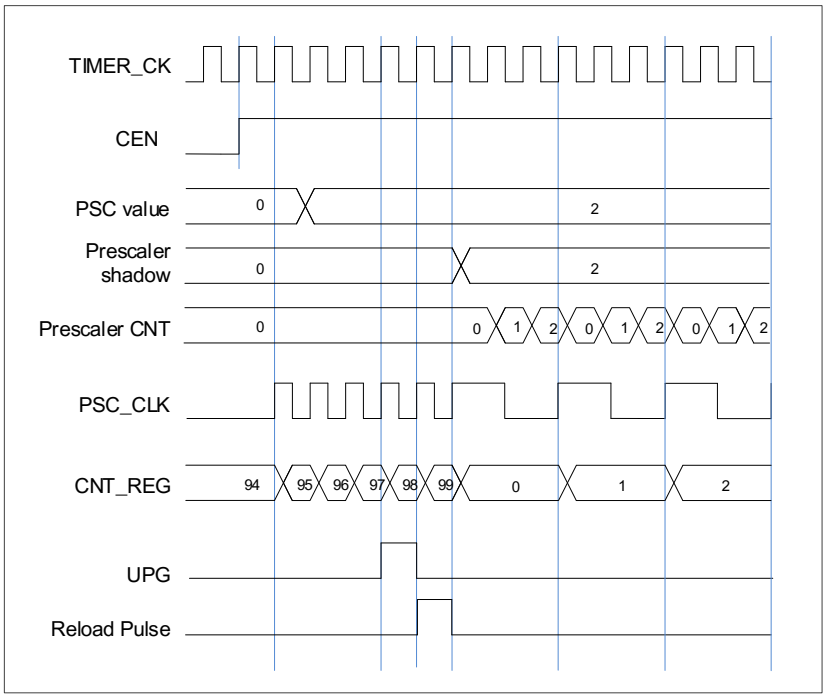
- SMC1= 1'b1 (external clock mode 1). External input ETI is selected as timer clock source.

The TIMER_CK, which drives counter's prescaler to count, can be triggered by the event of rising or falling edge on the external pin ETI. This mode can be selected by setting the SMC1 bit in the TIMERx_SMCFG register to 1. The other way to select the ETI signal as the clock source is setting TSCFG6[4:0] to 0x18. Note that the ETI signal is derived from the ETI pin sampled by a digital filter. When the ETI signal is selected as the clock source, the trigger controller including the edge detection circuitry will generate a clock pulse on each ETI signal rising edge to clock the counter prescaler.

Clock prescaler

The counter clock (PSC_CK) is obtained by the TIMER_CK through the prescaler, and the prescale factor can be configured from 1 to 65536 through the prescaler register (TIMERx_PSC). The new written prescaler value will not take effect until the next update event.

Figure 17-3. Timing chart of PSC value change from 0 to 2



Counter up counting

In this mode, the counter counts up continuously from 0 to the counter-reload value, which is defined in the TIMERx_CAR register, in a count-up direction. Once the counter reaches the counter reload value, the counter will start counting up from 0 again and an overflow event will be generated. In addition, the update events will be generated after (TIMERx_CREP+1) times of overflow events. The counting direction bit DIR in the TIMERx_CTL0 register should be set to 0 for the up counting mode.

Whenever, if the update event software trigger is enabled by setting the UPG bit in the TIMERx_SWEVG register, the counter value will be initialized to 0 and an update event will

be generated.

If the UPDIS bit in TIMERx_CTL0 register is set, the update event is disabled.

When an update event occurs, all the shadow registers (repetition counter register, counter auto reload register, prescaler register) are updated.

Figure 17-4. Timing chart of up counting mode, PSC=0/2 and *Figure 17-5. Timing chart of up counting mode, change TIMERx_CAR on the go* show some examples of the counter behavior for different clock prescaler factors when TIMERx_CAR=0x63.

Figure 17-4. Timing chart of up counting mode, PSC=0/2

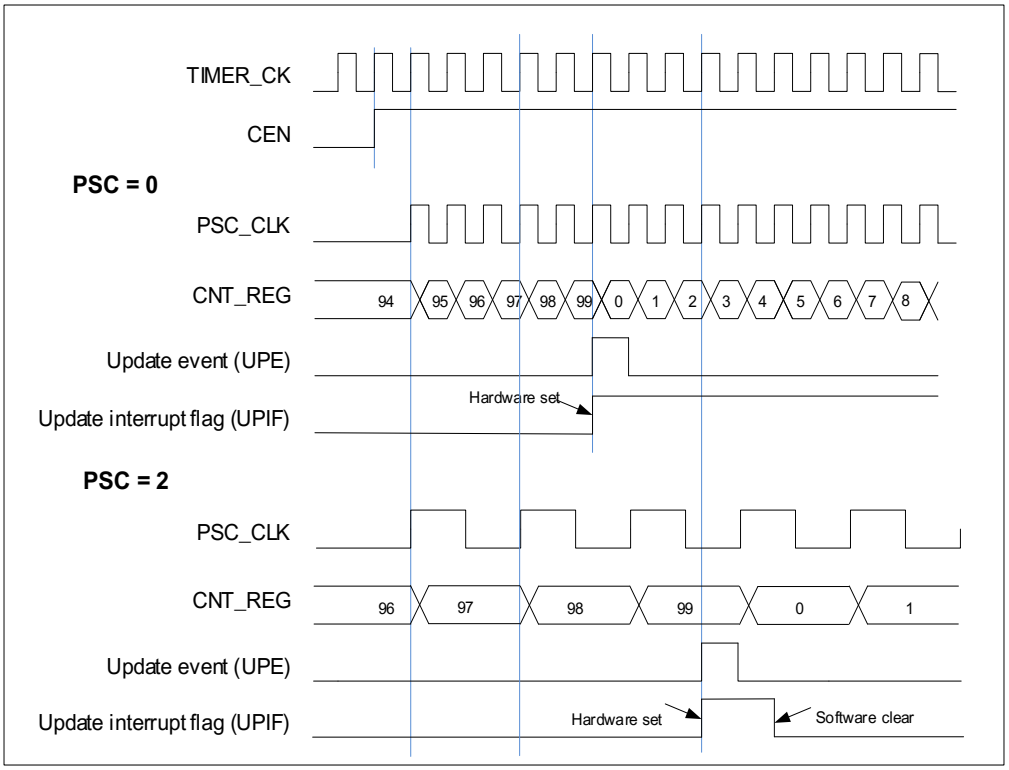
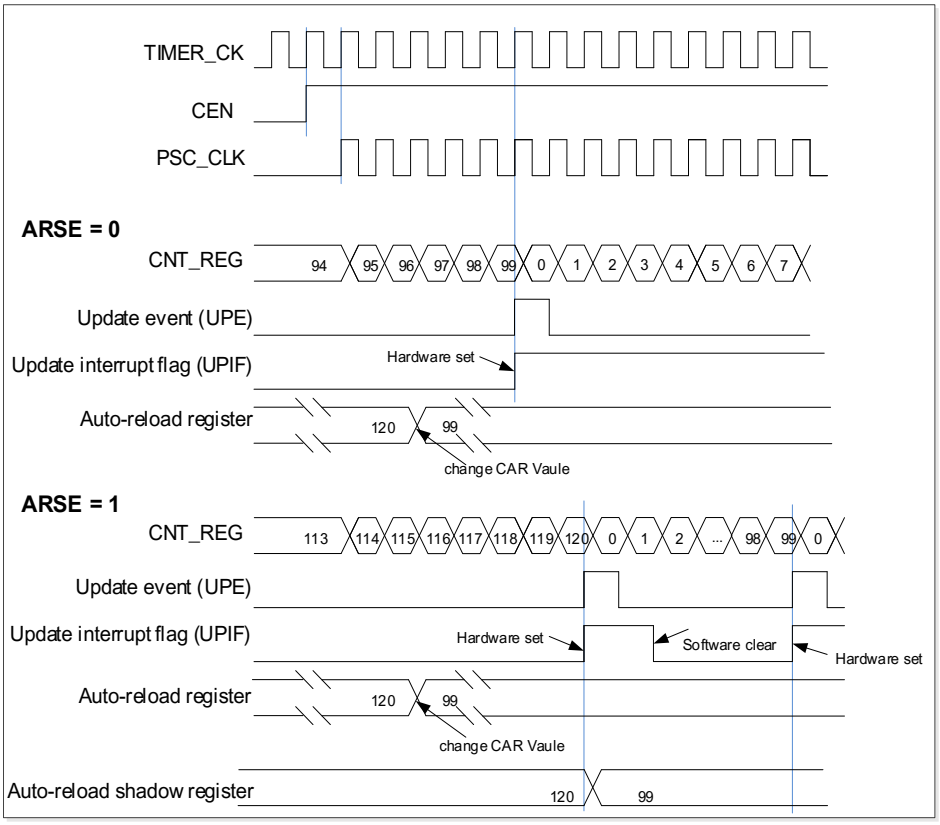


Figure 17-5. Timing chart of up counting mode, change TIMERx_CAR on the go



Counter down counting

In this mode, the counter counts down continuously from the counter-reload value, which is defined in the TIMERx_CAR register, to 0 in a count-down direction. Once the counter reaches to 0, the counter will start counting down from the counter-reload value again and an underflow event will be generated. In addition, the update event will be generated after (TIMERx_CREP+1) times of underflow. The counting direction bit DIR in the TIMERx_CTL0 register should be set to 1 for the down-counting mode.

When the update event is set by the UPG bit in the TIMERx_SWEVG register, the counter value will be initialized to the counter reload value and an update event will be generated.

If the UPDIS bit in TIMERx_CTL0 register is set, the update event is disabled.

When an update event occurs, all the shadow registers (repetition counter register, counter auto reload register, prescaler register) are updated.

Figure 17-6. Timing chart of down counting mode, PSC=0/2 and *Figure 17-7. Timing chart of down counting mode, change TIMERx_CAR on the go* show some examples of

the counter behavior in different clock frequencies when TIMERx_CAR = 0x63.

Figure 17-6. Timing chart of down counting mode, PSC=0/2

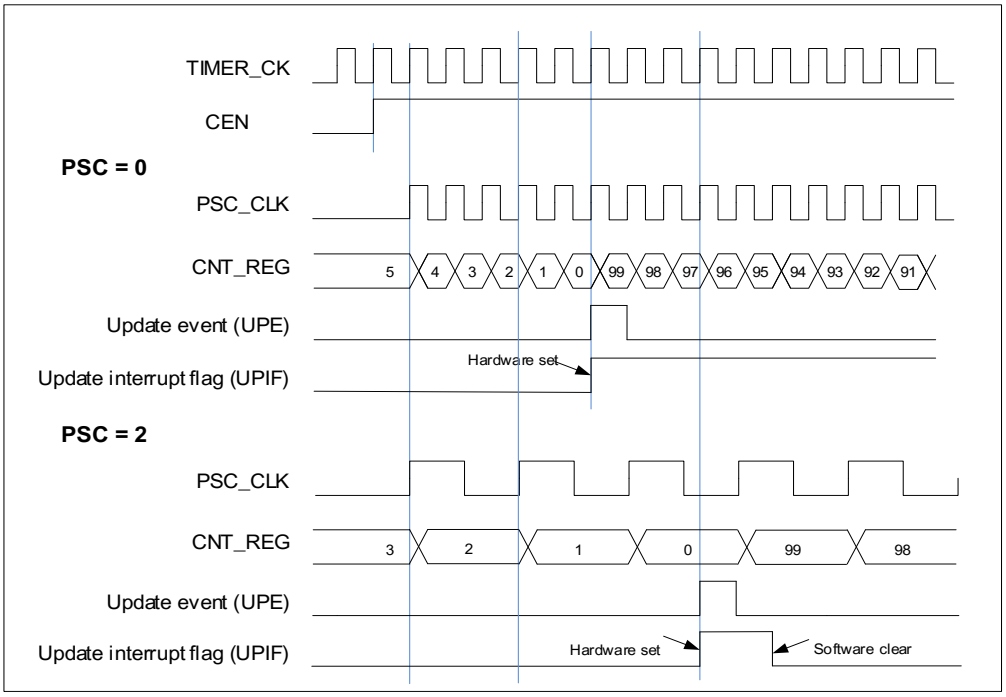
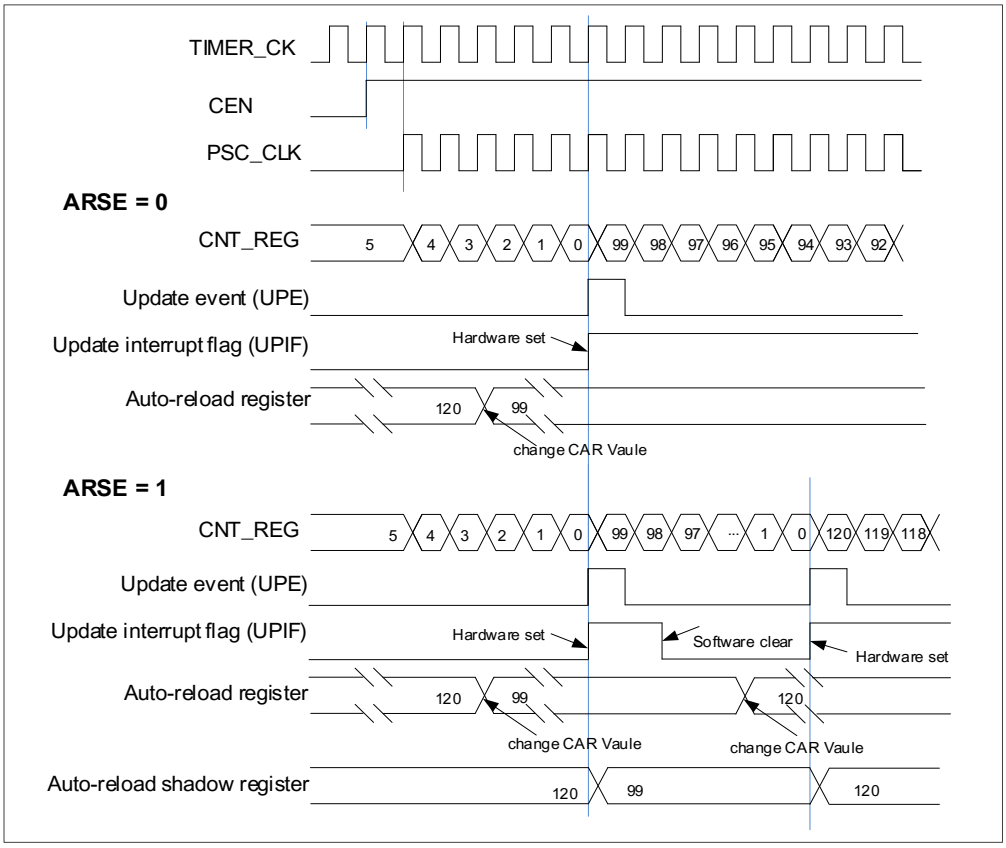


Figure 17-7. Timing chart of down counting mode, change TIMERx_CAR on the go



Counter center-aligned counting

In this mode, the counter counts up from 0 to the counter-reload value and then counts down to 0 alternatively. The Timer module generates an overflow event when the counter counts to the counter-reload value subtract 1 in the up-counting direction and generates an underflow event when the counter counts to 1 in the down-counting direction. The counting direction bit DIR in the TIMERx_CTL0 register is read-only and indicates the counting direction when in the center-aligned mode.

Setting the UPG bit in the TIMERx_SWEVG register will initialize the counter value to 0 and generate an update event irrespective of whether the counter is counting up or down in the center-aligned counting mode.

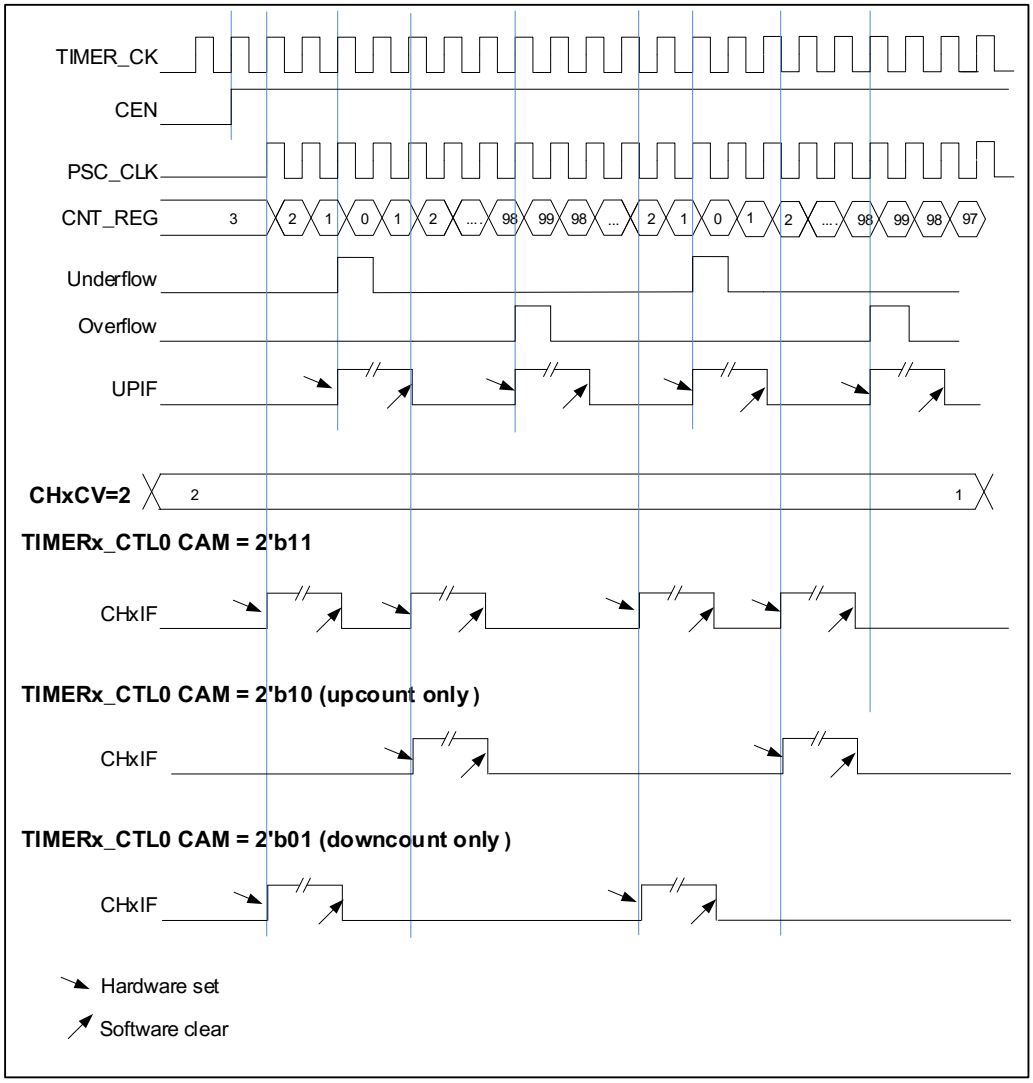
The UPIF bit in the TIMERx_INTF register will be set to 1 either when an underflow event or an overflow event occurs. While the CHxIF bit is associated with the value of CAM[1:0] in TIMERx_CTL0. The details refer to ***Figure 17-8. Timing chart of center-aligned counting mode***.

If the UPDIS bit in the TIMERx_CTL0 register is set, the update event is disabled.

When an update event occurs, all the shadow registers (repetition counter register, counter auto-reload register, prescaler register) are updated.

Figure 17-8. Timing chart of center-aligned counting mode shows some examples of the counter behavior when TIMERx_CAR=0x99. TIMERx_PSC=0x0.

Figure 17-8. Timing chart of center-aligned counting mode



Update event (from overflow/underflow) rate configuration

The rate of update events generation (from overflow and underflow events) can be configured by the TIMERx_CREP register. Counter repetition is used to generator update event or updates the timer registers only after a given number (N+1) of cycles of the counter, where N is CREP in TIMERx_CREP register. The repetition counter is decremented at each counter overflow (does not exist in down counting mode) and underflow (does not exist in up counting mode).

Setting the UPG bit in the TIMERx_SWEVG register will reload the content of CREP in TIMERx_CREP register and generate an update event.

The new written CREP value will not take effect until the next update event. When the value of CREP is odd, and the counter is counting in center-aligned mode, the update event is generated (on overflow or underflow) depending on when the written CREP value takes effect. If an update event is generated by software after writing an odd number to CREP, the update

events will be generated on the underflow. If the next update event occurs on overflow after writing an odd number to CREP, then the subsequent update events will be generated on the overflow.

Figure 17-9. Repetition counter timing chart of center-aligned counting mode

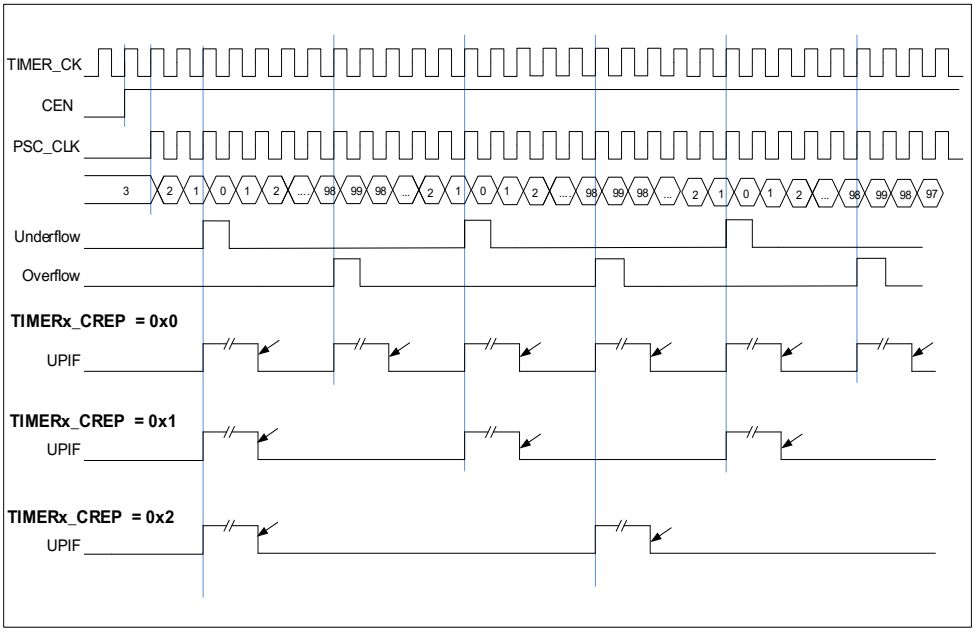


Figure 17-10. Repetition counter timing chart of up counting mode

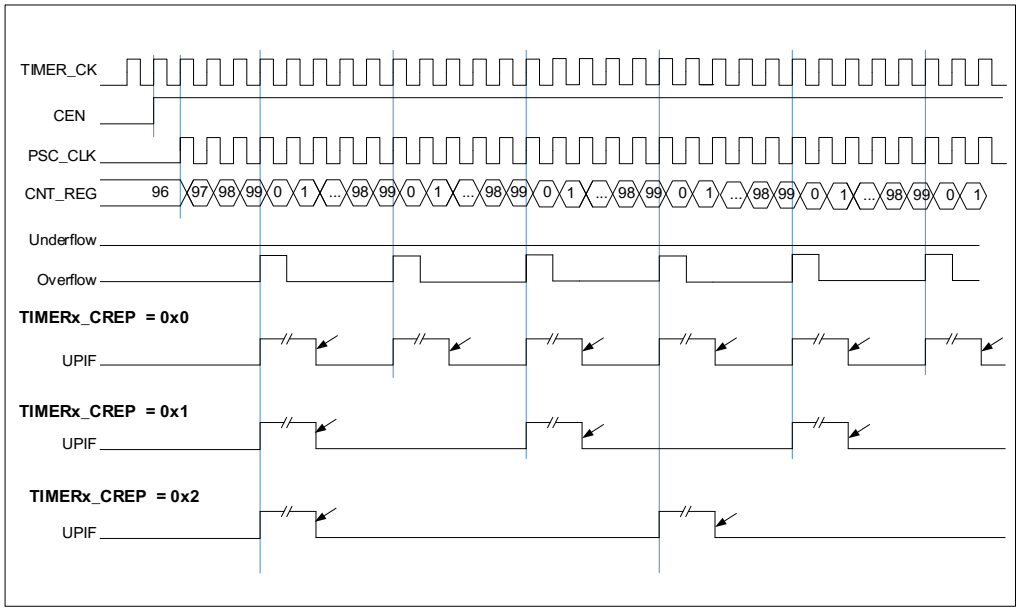
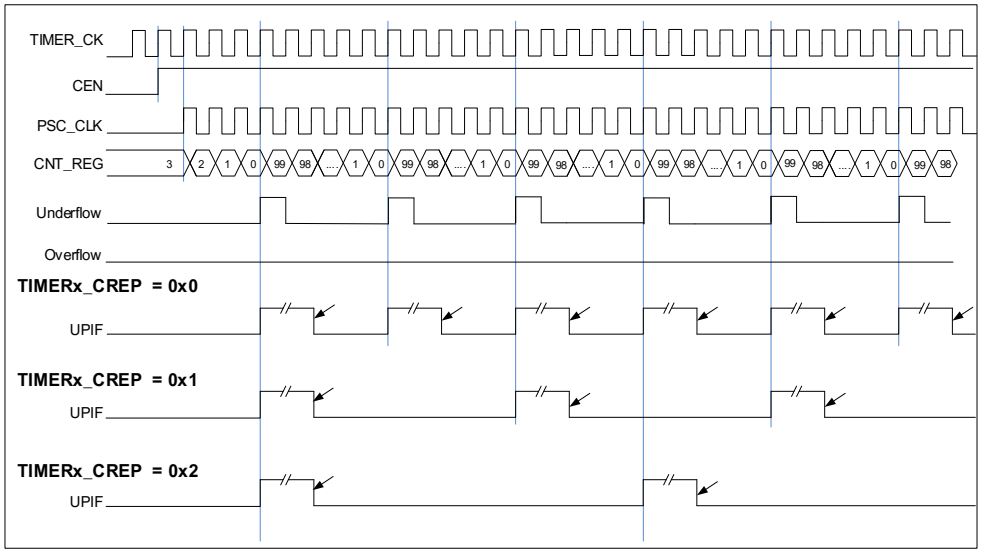


Figure 17-11. Repetition counter timing chart of down counting mode



Input capture and output compare channels

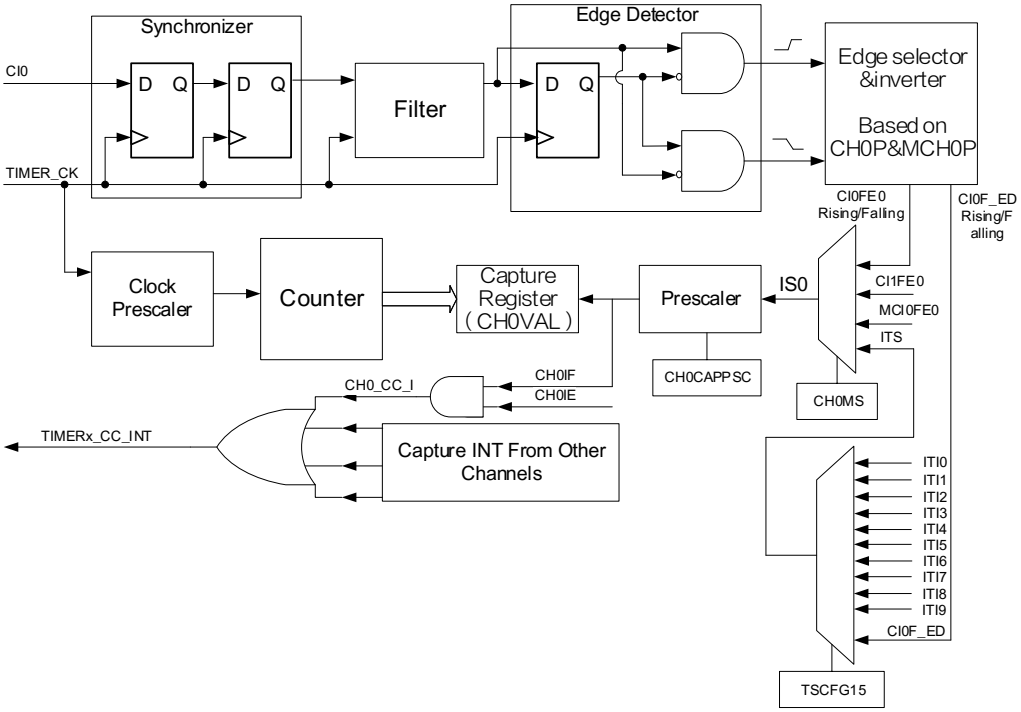
The advanced timer has eight independent channels which can be used as capture inputs or compare outputs. Each channel is built around a channel capture compare register including an input stage, a channel controller and an output stage.

When the channels are used for input, channel x and multi mode channel x can perform input capture independently; when the channels are used for comparison output, the channel x and multi mode channel x can output independent, mirrored, and complementary outputs.

■ Channel input capture function

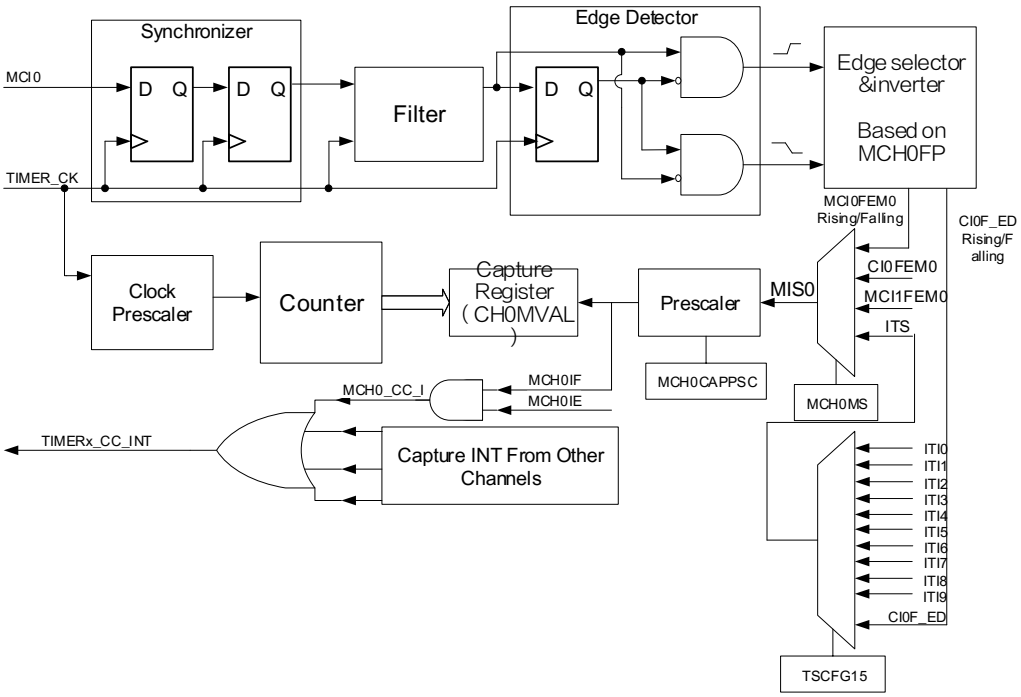
Channel input capture function allows the channel to perform measurements such as pulse timing, frequency, period, duty cycle and so on. The input stage consists of a digital filter, a channel polarity selection, edge detection and a channel prescaler. When a selected edge occurs on the channel input, the current value of the counter is captured into the TIMERx_CHxCV/ TIMERx_MCHxCV(x=0..3) registers, at the same time the CHxIF/ MCHxIF (x=0..3) bits are set and the channel interrupt is generated if it is enabled when CHxIE/ MCHxIE =1(x=0..3).

Figure 17-12. Channel 0 input capture principle



In the diagram, CI0 represents the input signal of channel 0, CI0FE0 represents the valid signal of channel 0 after passing through the edge selector, CI1FE0 represents the valid signal of channel 1 after passing through the edge selector, MCI0FE0 represents the valid signal of multi-mode channel 0 after passing through the edge selector, and ITS represents the internal trigger source input.

Figure 17-13. Multi mode channel 0 input capture principle



The input signals of channelx (CIx/ MCIx) can be the TIMERx_CHx/ TIMERx_MCHxCV signal or the XOR signal of the TIMERx_CH0, TIMERx_CH1 and TIMERx_CH2 signals(just for CI0).

First, the input signal of channel (CIx/ MCIx) is synchronized to TIMER_CK signal, and then sampled by a digital filter to generate a filtered input signal. Then through the edge detector, the rising or falling edge is detected by configuring CHxP/ MCHxP or MCHxFP bits. The input capture signal can also be selected from the input signal of other channel or the internal trigger signal by configuring CHxMS/ MCHxMS bits. The IC prescaler makes several input events generate one effective capture event. On the capture event, TIMERx_CHxCV/ TIMERx_MCHxCV will store the value of counter.

So, the process can be divided into several steps as below:

Step1: Filter configuration (CHxCAPFLT bit in TIMERx_CHCTL0 register and MCHxCAPFLT bit in TIMERx_MCHCTL0 register).

Based on the input signal and quality of requested signal, configure compatible CHxCAPFLT or MCHxCAPFLT bit.

Step2: Edge selection (CHxP and MCHxP bits in TIMERx_CHCTL2 register, MCHxFP[1:0] bits in TIMERx_MCHCTL2 register).

Rising edge or falling edge, choose one by configuring CHxP and MCHxP bits or MCHxFP[1:0] bits.

Step3: Capture source selection (CHxMS bit in TIMERx_CHCTL0 register, MCHxMS bit in TIMERx_MCHCTL0 register).

As soon as selecting one input capture source by CHxMS, the channel must be set to input mode (CHxMS! =0x000 or MCHxMS!=0x000) and TIMERx_CHxCV/ TIMERx_MCHxCV cannot be written any more.

Step4: Interrupt enable (CHxIE and CHxDEN bits, MCHxIE and MCHxDEN bits in TIMERx_DMAINTEN).

Enable the related interrupt to get the interrupt and DMA request.

Step5: Capture enable (CHxEN and MCHxEN bits in TIMERx_CHCTL2).

Result: When the wanted input signal is captured, TIMERx_CHxCV/ TIMERx_MCHxCV will be set by counter's value and CHxIF/ MCHxIF bit is asserted. If the CHxIF/ MCHxIF bit is already 1, the CHxOF/ MCHxOF bit will also be asserted. The interrupt and DMA request will be asserted or not based on the configuration of CHxIE and CHxDEN bits, MCHxIE and MCHxDEN bits in TIMERx_DMAINTEN.

Direct generation: A DMA or interrupt request is generated by setting CHxG directly.

The channel input capture function can be also used for pulse width measurement from signals on the TIMERx_CHx and TIMERx_MCHx pins. For example, PWM signal connects to CI0 input. Select CI0 as channel 0 capture signals by setting CH0MS to 3'b001 in the channel control register (TIMERx_CHCTL0) and set capture on rising edge. Select CI0 as channel 1 capture signal by setting CH1MS to 3'b010 in the channel control register (TIMERx_CHCTL0) and set capture on falling edge. The counter is set to restart mode and is restarted on channel 0 rising edge. Then the TIMERX_CH0CV can measure the PWM period and the TIMERx_CH1CV can measure the PWM duty cycle.

■ **Channel multi-edge input capture function**

In some input capture scenarios, it is necessary to capture the multi-edge of input signals, such as the measurement of input signal period and input signal duty cycle. The eight channels of advanced timer provide multi edge capture in order to meet these application scenarios.

The prerequisite for enabling multi-edge capture is that the channel is in input mode, (CHxMS/MCHxMS != 3 'b000, x=0,1,2,3), The CHxMECEN and CHxMECP bits of the multi-edge capture configuration register TIMERx_CHMECAPCFG are configured to enable the corresponding channel (TIMERx_CHx , x=0,1,2,3) and edge capture polarity of the input signal. The MCHxMECEN and MCHxMECP bits of the multi-edge capture configuration register TIMERx_MCHMECAPCFG are configured to enable the corresponding channel (TIMERx_MCHx , x=0,1,2,3) and edge capture polarity of the input signal. Configuring CHxMEBL determines the level of buffer captured by the current channel.The TIMERx_CHxCV and TIMERx_CHxCV_ADD registers record the counter value captured for the first time and the counter value captured for the second time.The third and fourth captured counters are recorded in TIMERx_CHxMECAP. Configuring MCHxMEBL determines the level of buffer captured by the current channel of multi channel.The TIMERx_MCHxCV and TIMERx_MCHxCV_ADD registers record the counter value captured for the first time and the counter value captured for the second time.The third and fourth captured counters are recorded in TIMERx_MCHxMECAP.

■ **Channel output compare function**

Figure 17-14. Channel output compare principle (when MCHxMSEL = 2'00, x=0, 1, 2, 3),

Figure 17-15. Channel output compare principle (when MCHxMSEL = 2'01, x=0, 1, 2, 3) and *Figure 17-16. Channel output compare principle (with complementary output when MCHxMSEL = 2'11, x=0,1,2,3)* show the principle circuit of channels output compare function.

Figure 17-14. Channel output compare principle (when MCHxMSEL = 2'00, x=0, 1, 2, 3)

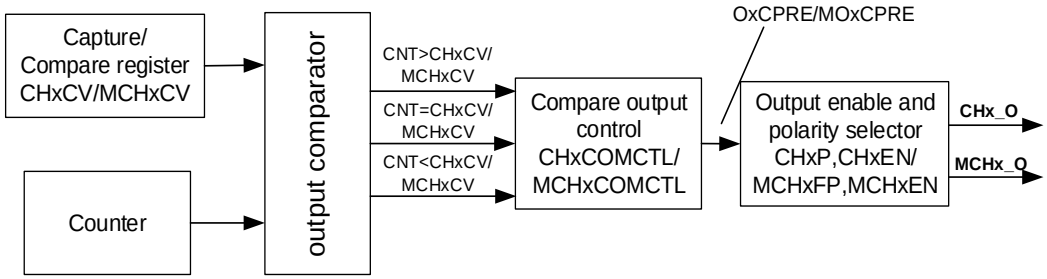


Figure 17-15. Channel output compare principle (when MCHxMSEL = 2'01, x=0, 1, 2, 3)

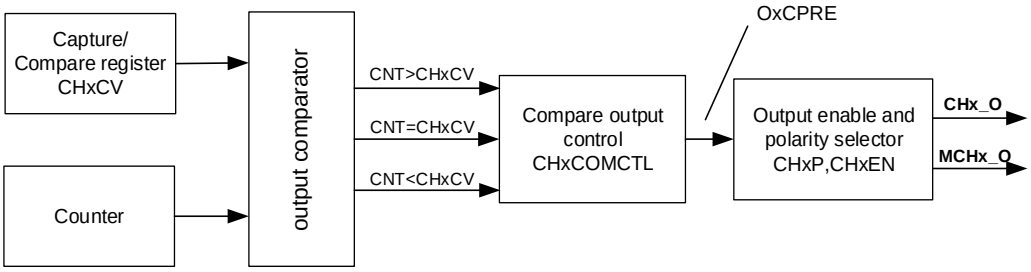
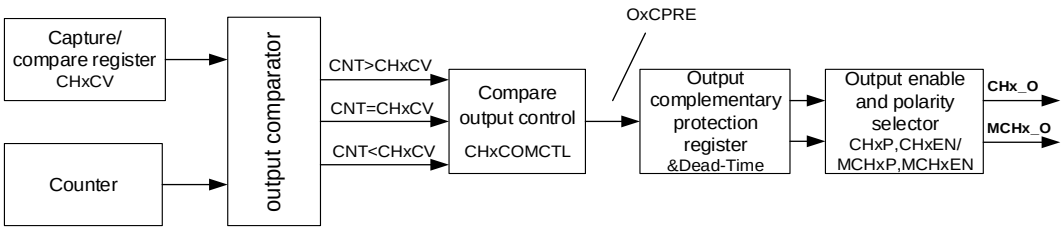


Figure 17-16. Channel output compare principle (with complementary output when MCHxMSEL = 2'11, x=0,1,2,3)



The relationship between the channel output signal CHx_O/MCHx_O and the OxCPRE/ MOxCPRE signal (more details refer to *Channel output prepare signal*) is described as blew(the active level of OxCPRE is high and the active level of MOxCPRE is high).

- When MCHxMSEL=2'b00 (in TIMERx_CTL2 register), the MCHx_O output is independent from the CHx_O output. The output level of CHx_O depends on OxCPRE signal, CHxP bit and CHxEN bit (please refer to the TIMERx_CHCTL2 register for more details). The output level of MCHx_O depends on MOxCPRE signal, MCHxFP[1:0] bits and MCHxEN bit (please refer to the TIMERx_MCHCTL2 for more details). Please refer to *Figure 17-14. Channel output compare principle (when MCHxMSEL = 2'00, x=0, 1, 2, 3)*.

- When MCHxMSEL=2'b01, the MCHx_O output is the same as the CHx_O output. The output level of CHx_O and MCHx_O depends on OxCPRE signal, CHxP bit and CHxEN bit. It is invalid for the configuration of MCHx_O. Please refer to ***Figure 17-15. Channel output compare principle (when MCHxMSEL = 2'01, x=0, 1, 2, 3)***.

- When MCHxMSEL=2'b11, the MCHx_O output is the inverse of the CHx_O output. The output level of CHx_O/MCHx_O depends on OxCPRE signal, CHxP/ MCHxP bits and CHxEN/MCHxEN bits. Please refer to ***Figure 17-16. Channel output compare principle (with complementary output when MCHxMSEL = 2'11, x=0,1,2,3)***.

For examples (the MCHx_O output is independent from the CHx_O output):

1) Configure CHxP=0 (the active level of CHx_O is high, the same as OxCPRE), CHxEN=1 (the output of CHx_O is enabled):
   If the output of OxCPRE is active(high) level, the output of CHx_O is active(high) level;
   If the output of OxCPRE is inactive(low) level, the output of CHx_O is active(low) level.

2) Configure MCHxP=1 (the active level of MCHx_O is low, contrary to MOxCPRE), MCHxEN=1 (the output of MCHx_O is enabled):
   If the output of MOxCPRE is active(high) level, the output of MCHx_O is active(low) level;
   If the output of MOxCPRE is inactive(low) level, the output of MCHx_O is active(high) level.

When MCHxMSEL=2'b11 and CHx_O and MCHx_O are output at the same time, the specific outputs of CHx_O and MCHx_O are related to the relevant bits (ROS, IOS, POEN and DTCFG bits) in the TIMERx_CCHP register. Please refer to ***Channel output complementary PWM*** for more details.

In channel output compare function, the TIMERx can generate timed pulses with programmable position, polarity, duration and frequency. When the counter matches the value in the TIMERx_CHxCV/ TIMERx_MCHxCV register of an output compare channel, the channel output can be set, cleared, or toggled based on CHxCOMCTL/ MCHxCOMCTL. When the counter reaches the value in the TIMERx_CHxCV/ TIMERx_MCHxCV register, the CHxIF/ MCHxIF bit will be set and the channel interrupt is generated if CHxIE/ MCHxIE = 1. And the DMA request will be asserted, if CHxDEN/ MCHxDEN =1.

So, the process can be divided into several steps as below:

Step1: Clock Configuration. Such as clock source, clock prescaler and so on.
Step2: Compare mode configuration.
- Set the shadow enable mode by CHxCOMSEN/ MCHxCOMSEN .
- Set the output mode (set/clear/toggle) by CHxCOMCTL/ MCHxCOMCTL.
- Select the active polarity by CHxP/MCHxP/ MCHxFP.
- Enable the output by CHxEN/ MCHxEN.

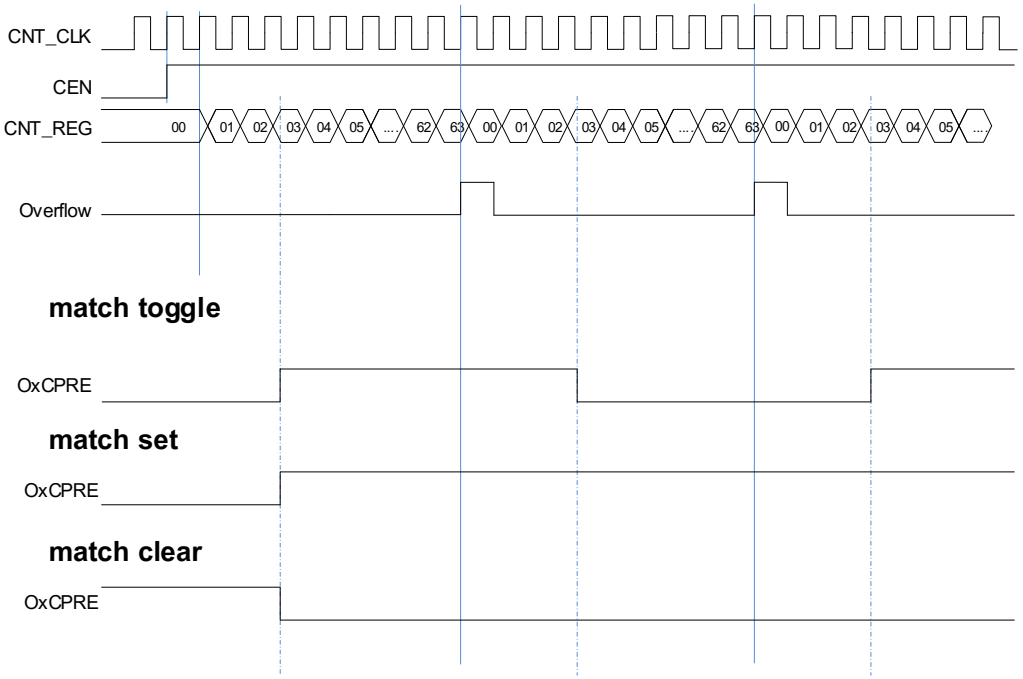
Step3: Interrupt/DMA request enable configuration by CHxIE/ MCHxIE /CHxDEN/ MCHxDEN.
Step4: Compare output timing configuration by TIMERx_CAR and TIMERx_CHxCV/ TIMERx_MCHxCV.
The TIMERx_CHxCV/ TIMERx_MCHxCV can be changed onging to meet the

expected waveform.

Step5: Start the counter by configuring CEN to 1.

Figure 17-17. Output-compare under three modes shows the three compare modes: toggle/set/clear. CARL=0x63, CHxVAL=0x3.

Figure 17-17. Output-compare under three modes



Output PWM function

In the PWM output function (by setting the CHxCOMCTL/ MCHxCOMCTL bit to 4'b0110 (PWM mode 0) or to 4'b0111(PWM mode 1)), the channel can generate PWM waveform according to the TIMERx_CAR registers and TIMERx_CHxCV/ TIMERx_MCHxCV registers.

Based on the counter mode, PWM can also be divided into EAPWM (Edge-aligned PWM) and CAPWM (Center-aligned PWM).

The EAPWM's period is determined by TIMERx_CAR and the duty cycle is determined by TIMERx_CHxCV/ TIMERx_MCHxCV. *Figure 17-18. EAPWM timechart* shows the EAPWM output and interrupts waveform.

The CAPWM's period is determined by 2*TIMERx_CAR, and the duty cycle is determined by 2*TIMERx_CHxCV/ TIMERx_MCHxCV. *Figure 17-19. CAPWM timechart* shows the CAPWM output and interrupts waveform.

In up counting mode, if the value of TIMERx_CHxCV/ TIMERx_MCHxCV is greater than the value of TIMERx_CAR, the output will be always active in PWM mode 0 (CHxCOMCTL/ MCHxCOMCTL =4'b0110). And if the value of TIMERx_CHxCV/ TIMERx_MCHxCV is greater

than the value of TIMERx_CAR, the output will be always inactive in PWM mode 1 (CHxCOMCTL/ MCHxCOMCTL =4'b0111).

Figure 17-18. EAPWM timechart

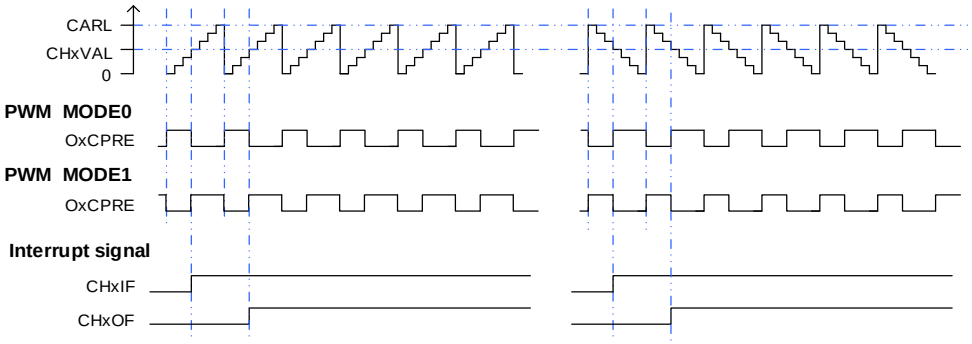
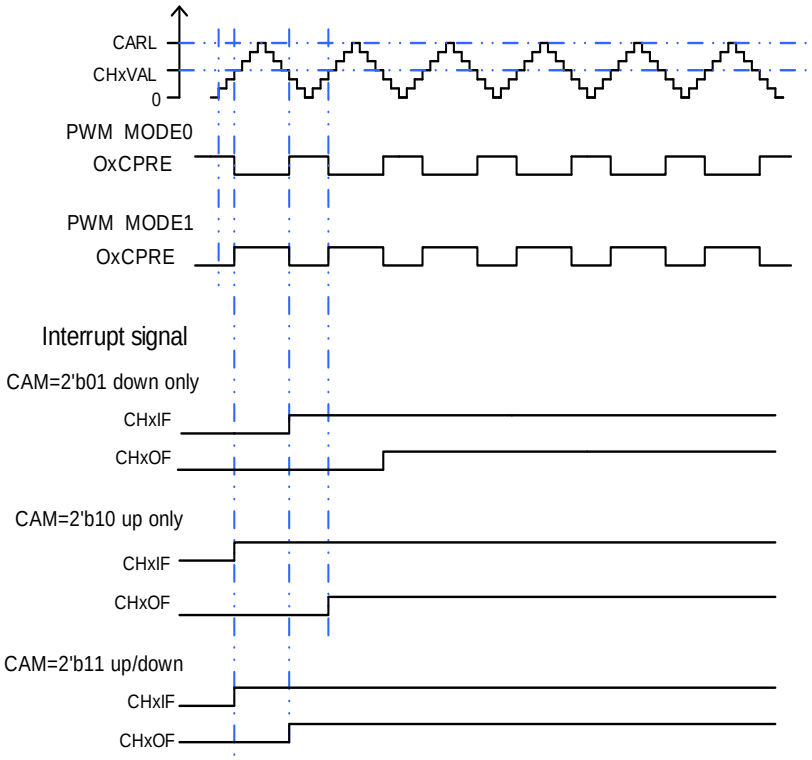


Figure 17-19. CAPWM timechart



Composite PWM mode

In the composite PWM mode (CHxMS[2:0] = 3'b000 and CHxCOMCTL = 4'b1000or 4'b1001), the PWM signal output in channel x (x=0..3) is composited by CHxVAL and CHxVAL_ADD bits.

If CHxCOMCTL = 4'b1000 (PWM mode 0) and DIR = 1'b0 (up counting mode), or

CHxCOMCTL = 4'b1001 (PWM mode 1) and DIR = 1'b1 (Down counting mode), the OxCPRE is forced low when the counter matches the value of CHxVAL. The OxCPRE is forced high when the counter matches the value of CHxVAL_ADD.

If CHxCOMCTL = 4'b1001 (PWM mode 1) and DIR = 1'b0 (up counting mode), or CHxCOMCTL = 4'b1000 (PWM mode 0) and DIR = 1'b1 (down counting mode) , the OxCPRE is forced high when the counter matches the value of CHxVAL. The OxCPRE is forced low when the counter matches the value of CHxVAL_ADD.

When CHxVAL or CHxVAL_ADD = 0 / CARL, the channel x output is specially processed, and by setting the CHxPERFOREN bit in the TIMERx_CHCTL2 register, the OxCPRE output is forced to either high or low level (determined according to the selected composite PWM mode).

The PWM period is determined by (CARL + 0x0001) and the PWM pulse width is determined by the following ***Table 17-3.The composite PWM pulse width***.

Table 17-3.The composite PWM pulse width

Condition	Mode	PWM pulse width
CHxVAL < CHxVAL_ADD ≤ CARL	Composite PWM mode 0	(CARL + 0x0001) + (CHxVAL - CHxVAL_ADD)
	Composite PWM mode 1	(CHxVAL_ADD - CHxVAL)
CHxVAL_ADD < CHxVAL ≤ CARL	Composite PWM mode 0	(CHxVAL - CHxVAL_ADD)
	Composite PWM mode 1	(CARL + 0x0001) + (CHxVAL_ADD - CHxVAL)
(CHxVAL = CHxVAL_ADD ≤ CARL) or (CHxVAL > CARL > CHxVAL_ADD)	Composite PWM mode 0 (up counting) or PWM mode 1 (down counting)	100%
	Composite PWM mode 0 (down counting) or PWM mode 1 (up counting)	0%
CHxVAL_ADD > CARL > CHxVAL	composite PWM mode 0(up counting) or composite PWM mode 1(down counting)	0%
	Composite PWM mode 0(down counting) or composite PWM mode 1(up counting)	100%
(CHxVAL>CARL)and (CHxVAL_ADD > CARL)	-	The output of CHx_O is keeping

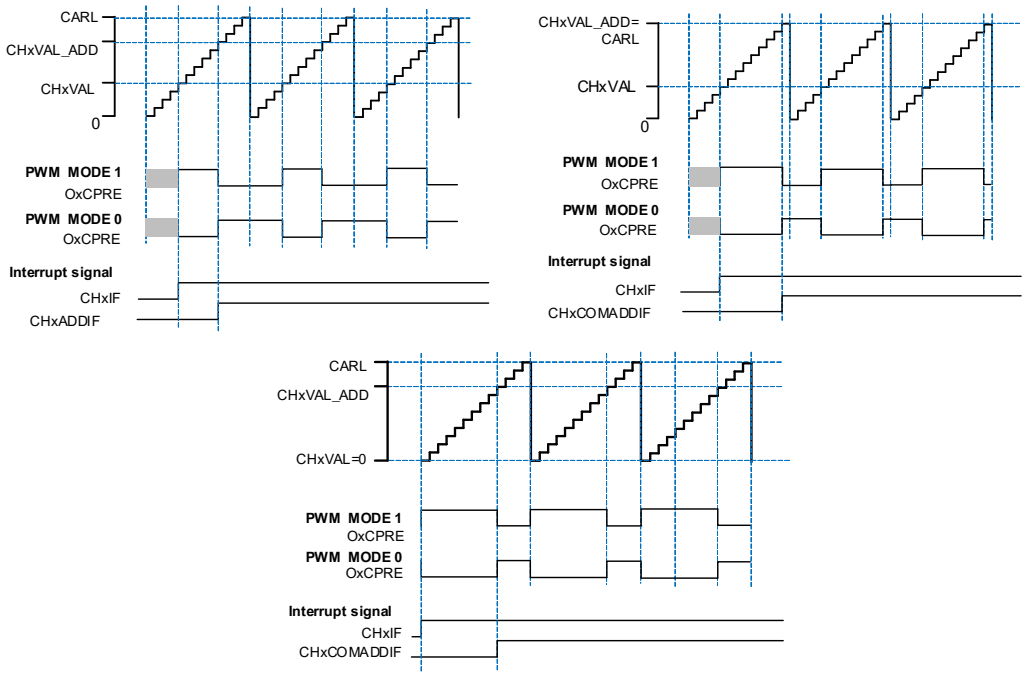
When the counter reaches the value of CHxVAL, the CHxIF bit is set and the channel x

interrupt is generated if CHxIE = 1, and the DMA request will be asserted, if CHxDEN=1. When the counter reaches the value of CHxVAL_ADD, the CHxCOMADDIF bit is set (This flag just used in composite PWM mode and asymmetric PWM mode, when CHxCOMCTL[3:2]=2'b10) and the channel x additional compare interrupt is generated if CHxCOMADDIE = 1 (Only interrupt is generated, no DMA request is generated).

According to the relationship among CHxVAL, CHxVAL_ADD and CARL, it can be divided into four situations:

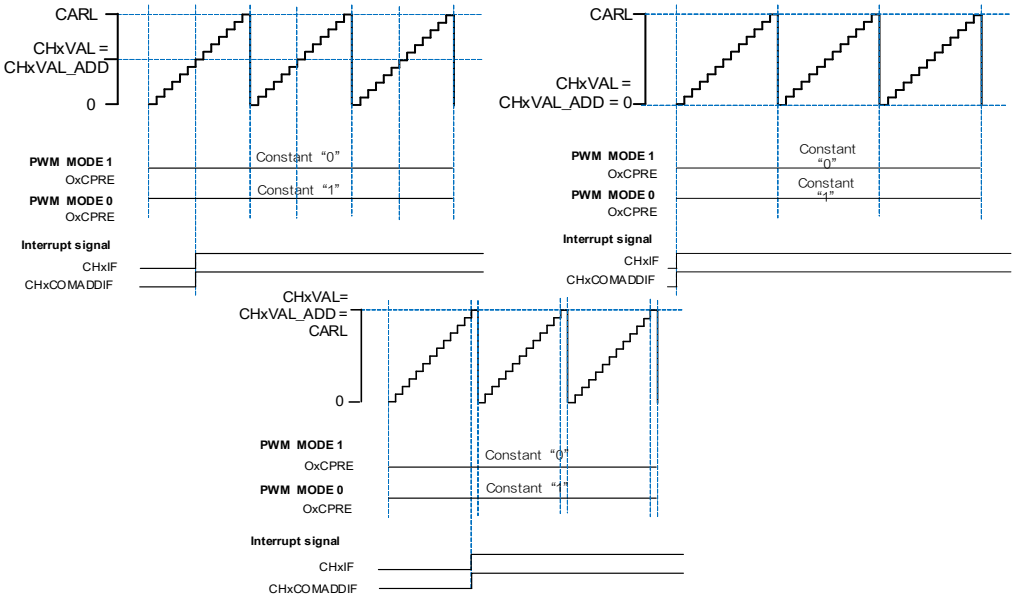
1) CHxVAL < CHxVAL_ADD, and the values of CHxVAL and CHxVAL_ADD between 0 and CARL.

Figure 17-20. Channel x output PWM with (CHxVAL < CHxVAL_ADD)



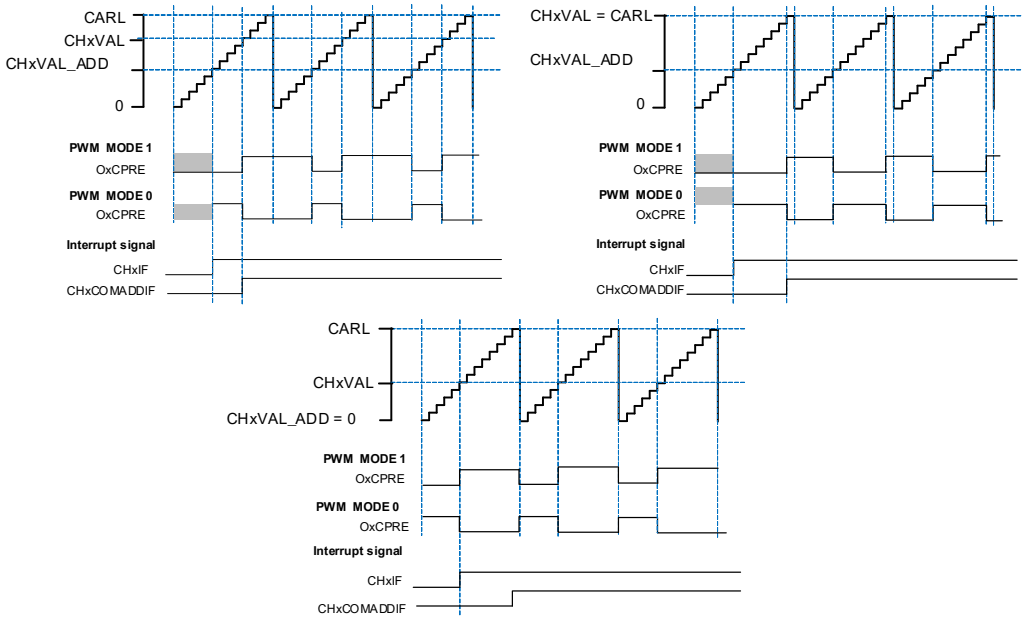
2) CHxVAL = CHxVAL_ADD, and the value of CHxVAL and CHxVAL_ADD between 0 and CARL.

Figure 17-21. Channel x output PWM with (CHxVAL = CHxVAL_ADD)



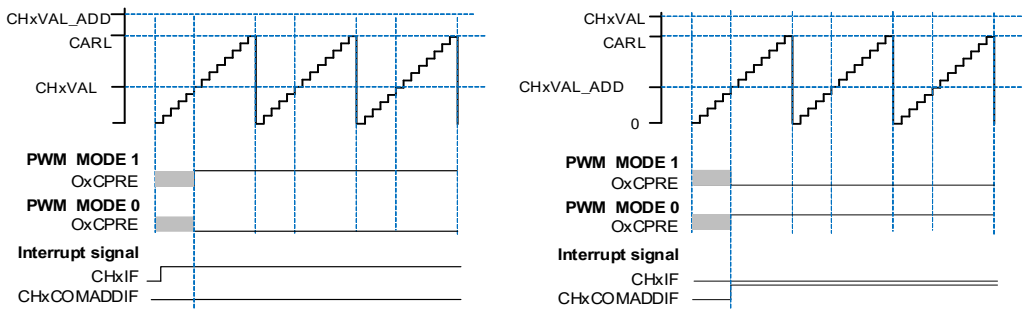
3) CHxVAL > CHxVAL_ADD, and the value of CHxVAL and CHxVAL_ADD between 0 and CARL.

Figure 17-22. Channel x output PWM with (CHxVAL > CHxVAL_ADD)



4) One of the value of CHxVAL and CHxVAL_ADD exceeds CARL.

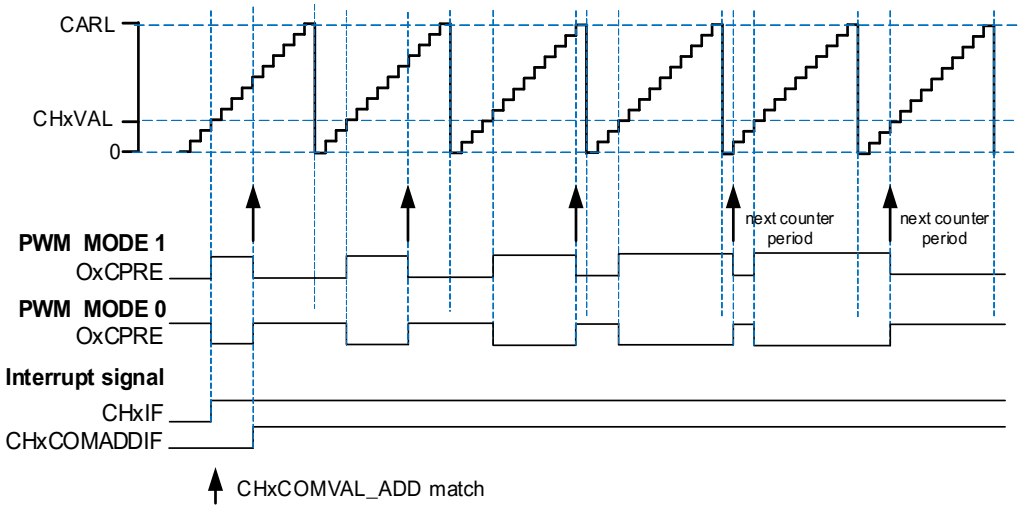
Figure 17-23. Channel x output PWM with CHxVAL or CHxVAL_ADD exceeds CARL



The composite PWM mode is intended to support the generation of PWM signals where the period is not modified while the signal is being generated, but the duty cycle will be varied. *Figure 17-24. Channel x output PWM duty cycle changing with CHxVAL_ADD* shows the PWM output and interrupts waveform.

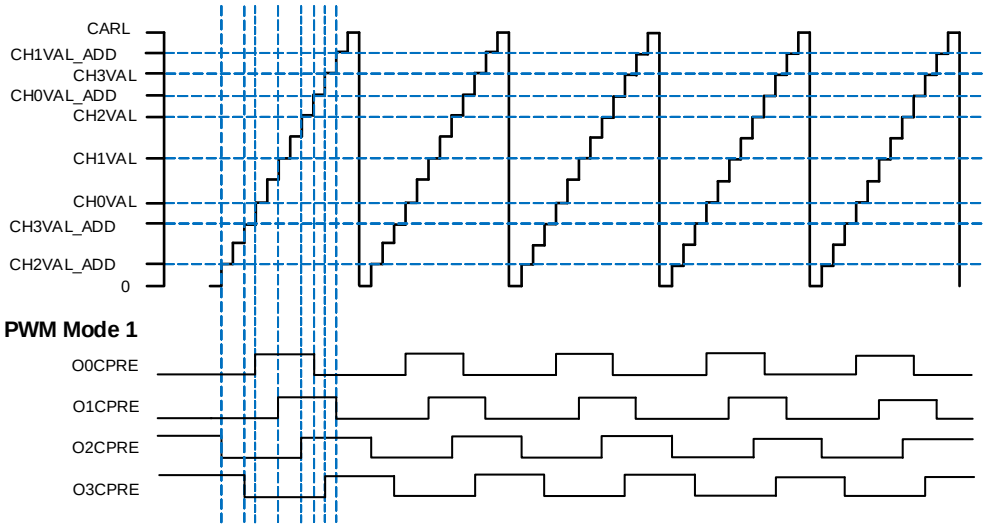
In some cases, the CHxVAL_ADD match can happen on the next counter period (the value of CHxVAL_ADD was written after the counter reaches the value of CHxVAL, and the value of CHxVAL_ADD was less than or equal to the CHxVAL).

Figure 17-24. Channel x output PWM duty cycle changing with CHxVAL_ADD



If more than one channels are configured in composite PWM mode, it is possible to fix an offset for the channel x match edge of each pair with respect to other channels. This behavior is useful in the generation of lighting PWM control signals where it is desirable that edges are not coincident with each other pair to help eliminate noise generation. The CHxVAL register value is the shift of the PWM pulse with respect to the beginning of counter period.

Figure 17-25. Four Channels outputs in composite PWM mode



Asymmetric PWM mode

Asymmetric PWM mode allows two center-aligned PWM signals to be generated with a programmable phase shift. In the asymmetric PWM mode(CHxCOMCTL[3:0]=4'b1010/4'b1011 and CHxMS[2:0] = 3'b000) , the PWM signal output in channel x (x=0..3) is composited by CHxVAL and CHxVAL_ADD bits. The asymmetric PWM mode is only available in the center-aligned counting mode.

If CHxCOMCTL = 4'b1010 (asymmetric PWM mode 0) and DIR = 1'b0) (up counting mode), the OxCPRE is forced low when the counter matches the value of CHxVAL.

If CHxCOMCTL = 4'b1010 (asymmetric PWM mode 0) and DIR = 1'b1 (down counting mode), the OxCPRE is forced high when the counter matches the value of CHxVAL_ADD.

If CHxCOMCTL = 4'b1011 (asymmetric PWM mode 1) and DIR = 1'b0 (up counting mode), the OxCPRE is forced high when the counter matches the value of CHxVAL.

If CHxCOMCTL = 4'b1011 (asymmetric PWM mode 1) and DIR = 1'b1 (down counting mode), the OxCPRE is forced low when the counter matches the value of CHxVAL_ADD.

Compared to composite PWM mode, asymmetric PWM mode is only valid with center-aligned mode, and the CHxVAL is only valid when counting up and CHxVAL_ADD is only valid when counting down.

When CHxVAL or CHxVAL_ADD = 0 / CARL, the channel x output is specially processed, and by setting the CHxPERFOREN bit in the TIMERx_CHCTL2 register, the OxCPRE output is forced to either high or low level (determined according to the selected asymmetric PWM mode).

When CHxVAL =0, the OxCPRE is forced low in asymmetric PWM mode 0. When CHxVAL =0, the OxCPRE is forced high in asymmetric PWM mode 1.

When CHxVAL_ADD =CARL, the OxCPRE is forced high in asymmetric PWM mode 0. When CHxVAL_ADD = CARL, the OxCPRE is forced low in asymmetric PWM mode 1.

When CHxVAL = CARL, the OxCPRE is forced low in asymmetric PWM mode 0. When

CHxVAL = CARL, the OxCPRE is forced high in asymmetric PWM mode 1.

When CHxVAL_ADD = 0, the OxCPRE is forced high in asymmetric PWM mode 0. When CHxVAL_ADD = 0, the OxCPRE is forced low in asymmetric PWM mode 1.

The PWM period is determined by (2CARL + 0x0001) and the PWM pulse width is determined by the following ***Table 17-4.The asymmetric PWM pulse width***.

Table 17-4.The asymmetric PWM pulse width

Condition	Mode	PWM pulse width
CHxVAL < CHxVAL_ADD ≤ CARL or CHxVAL_ADD < CHxVAL ≤ CARL or CHxVAL = CHxVAL_ADD	Asymmetric PWM mode 0	CHxVAL_ADD + CHxVAL
	Asymmetric PWM mode 1	(2CARL + 0x0001) - CHxVAL_ADD - CHxVAL
CHxVAL > CARL > CHxVAL_ADD	Asymmetric PWM mode 0	100%
	Asymmetric PWM mode 1	0%
CHxVAL_ADD > CARL > CHxVAL	Asymmetric PWM mode 0	0%
	Asymmetric PWM mode 1	100%
(CHxVAL>CARL)and (CHxVAL_ADD > CARL)	-	The output of CHx_O is keeping

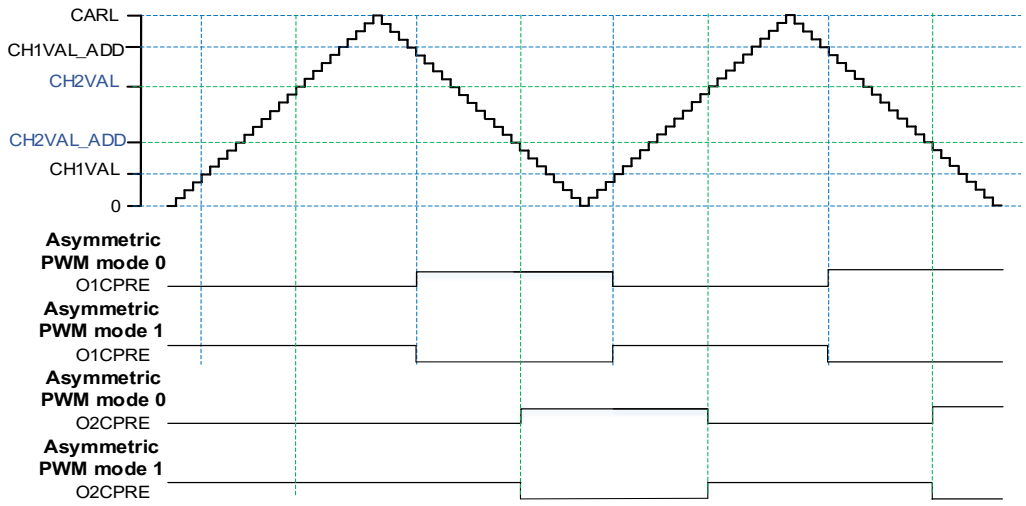
When the counter reaches the value of CHxVAL, the CHxIF bit is set and the channel x interrupt is generated if CHxIE = 1, and the DMA request will be asserted, if CHxDEN=1. When the counter reaches the value of CHxVAL_ADD, the CHxCOMADDIF bit is set (this flag just used in composite PWM mode and asymmetric PWM mode, when CHxCOMCTL[3:2]= 2'b10) and the channel x additional compare interrupt is generated if CHxCOMADDIE = 1 (Only interrupt is generated, no DMA request is generated).

According to the relationship among CHxVAL, CHxVAL_ADD and CARL, it can be divided into four situations:

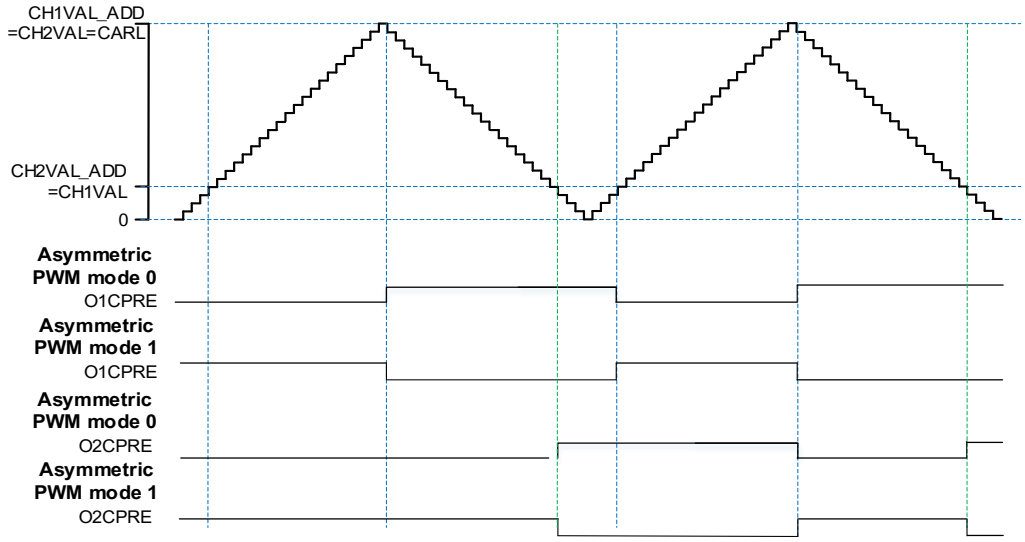
1）CHxVAL < CHxVAL_ADD, and the values of CHxVAL and CHxVAL_ADD between 0 and CARL.

Figure 17-26. Channel x output PWM with(CHxVAL < CHxVAL_ADD / CHxVAL_ADD < CHxVAL)

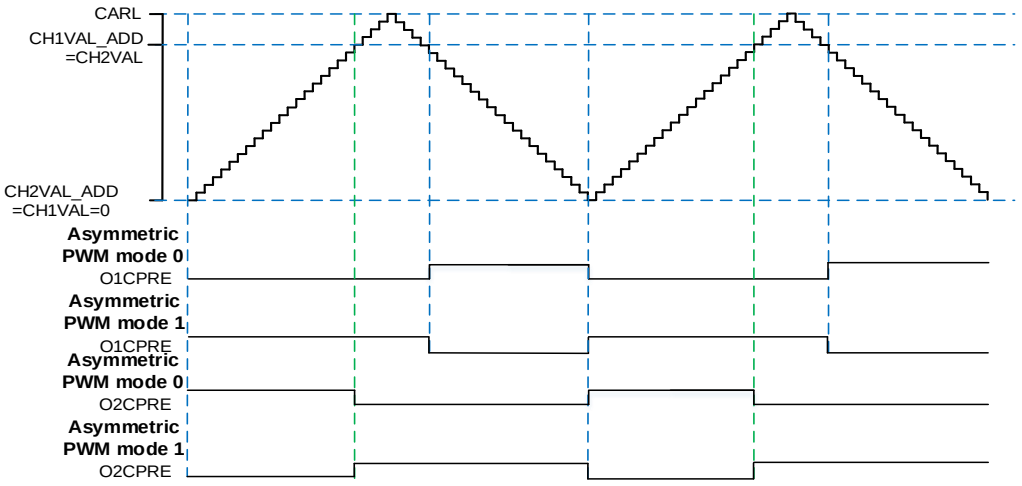
a. 0 < CH1VAL < CH1VAL_ADD < CARL / 0 < CH2VAL_ADD < CH2VAL < CARL



b. 0 < CH1VAL < CH1VAL_ADD = CARL / 0 < CH2VAL_ADD < CH2VAL = CARL, and CH1VAL = CH2VAL_ADD



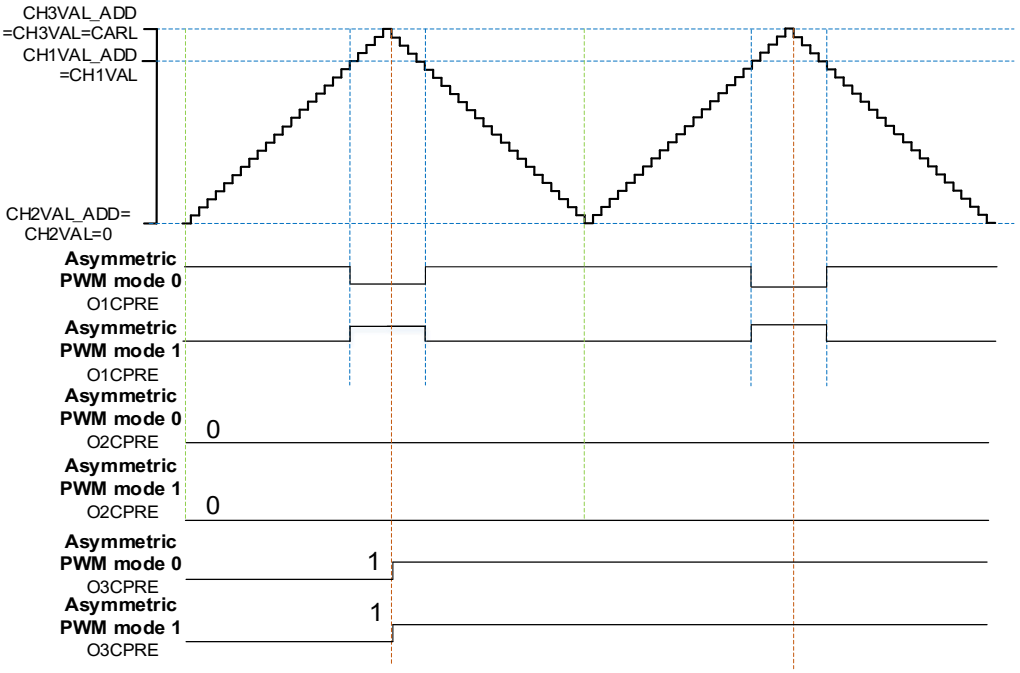
c. 0 = CH1VAL < CH1VAL_ADD < CARL / 0 = CH2VAL_ADD < CH2VAL < CARL, and CH1VAL_ADD = CH2VAL



2）CHxVAL = CHxVAL_ADD, and the value of CHxVAL and CHxVAL_ADD between 0 and CARL.

Figure 17-27. Channel x output PWM with(CHxVAL = CHxVAL_ADD)

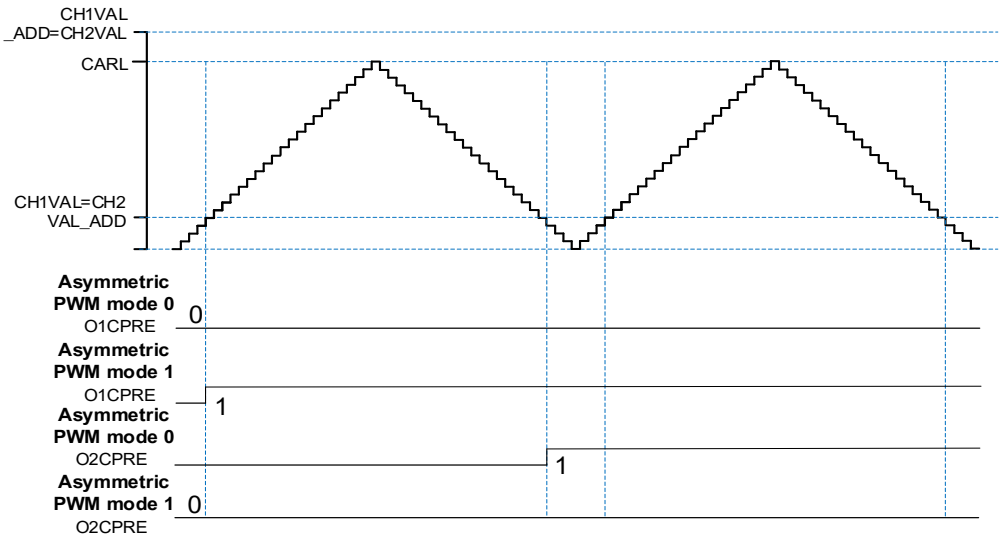
0 < CH1VAL = CH1VAL_ADD < CARL / CH2VAL = CH2VAL_ADD = 0 / CH3VAL = CH3VAL_ADD = CARL



3）One of the value of CHxVAL and CHxVAL_ADD exceeds CARL.

Figure 17-28. Channel x output PWM with CHxVAL or CHxVAL_ADD exceeds CARL

0 < CH1VAL < CARL < CH1VAL_ADD / 0 < CH2VAL_ADD < CARL < CH2VAL



Output match pulse select

Basing on that CHx_O(x=0..3) outputs are configured by CHxCOMCTL[3:0](x=0..3) bits when the match events occur, the output signal is configured by CHxOMPSEL[1:0](x=0..3) bit to be normal or a pulse.

When the match events occur, the CHxOMPSEL[1:0](x=0..3) bits are used to select the output of OxCPRE which drives CHx_O:

- CHxOMPSEL = 2'b00, the OxCPRE signal is output normally with the configuration of CHxCOMCTL[2:0] bits;
- CHxOMPSEL = 2'b01, only the counter is counting up, the OxCPRE signal is output a pulse when the match events occur, and the pulse width is one CK_TIMER clock cycle.
- CHxOMPSEL = 2'b10, only the counter is counting down, the OxCPRE signal is output a pulse when the match events occur, and the pulse width is one CK_TIMER clock cycle.
- CHxOMPSEL = 2'b11, both the counter is counting up and counting down, the OxCPRE signal is output a pulse when the match events occur, and the pulse width is one CK_TIMER clock cycle.

Figure 17-29. CHx_O output with a pulse in edge-aligned mode (CHxOMPSEL!=2'b00)

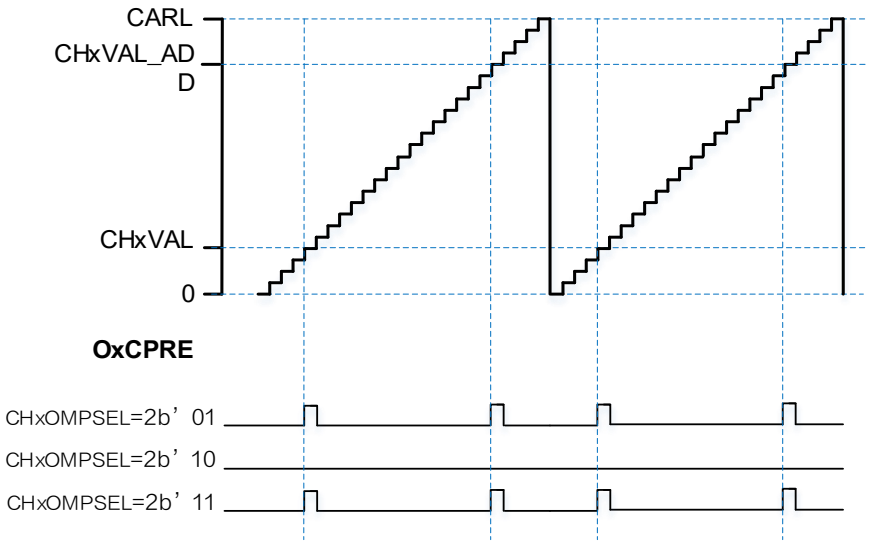
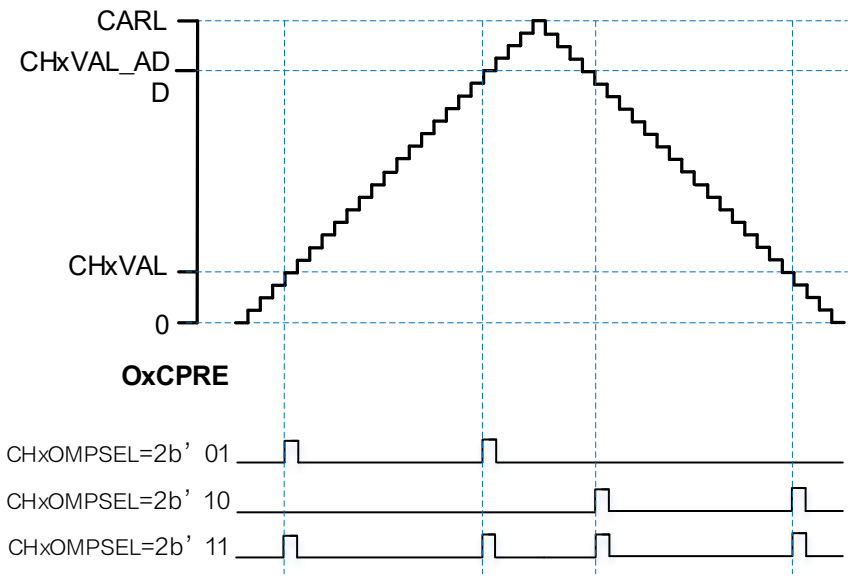


Figure 17-30. CHx_O output with a pulse in center-aligned mode (CHxOMPSEL!=2'b00)



Channel output prepare signal

As is shown in ***Figure 17-14. Channel output compare principle (when MCHxMSEL = 2'00, x=0, 1, 2, 3)*** and ***Figure 17-16. Channel output compare principle (with complementary output when MCHxMSEL = 2'11, x=0,1,2,3)***, when TIMERx is configured in compare match output mode, a middle signal named OxCPRE or MOxCPRE (channel x output or multi mode channel x output prepare signal) will be generated before the channel outputs signal.

The OxCPRE and MOxCPRE signal have several types of output function. The OxCPRE signal type is defined by configuring the CHxCOMCTL bit and the MOxCPRE signal type is defined by configuring the MCHxCOMCTL bit.

Take OxCPRE as an example for description below and these include keeping the original

level by configuring the CHxCOMCTL field to 0x00, setting to high by configuring the CHxCOMCTL field to 0x01, setting to low by configuring the CHxCOMCTL field to 0x02 or toggling signal by configuring the CHxCOMCTL field to 0x03 when the counter value matches the content of the TIMERx_CHxCV register.

The PWM mode 0/ PWM mode 1 output is another output type of OxCPRE which is setup by configuring the CHxCOMCTL field to 0x06/0x07. In these modes, the OxCPRE signal level is changed according to the counting direction and the relationship between the counter value and the TIMERx_CHxCV content. Refer to the definition of relative bit for more details.

Another special function of the OxCPRE signal is forced output which can be achieved by configuring the CHxCOMCTL field to 0x04/ 0x05. The output can be forced to an inactive/active level irrespective of the comparison condition between the values of the counter and the TIMERx_CHxCV.

Configure the CHxCOMCEN bit to 1 in the TIMERx_CHCTL0 register, the OxCPRE signal can be forced to inactive when the ETIFP signal derived from the external ETI pin is set to a active level. When the next comparison result changes after the ETIFP signal becomes invalid, the OxCPRE signal will immediately return to the active level.

Channel output complementary PWM

The outputs of CHx_O and MCHx_O have three situations:

- MCHxMSEL=2'b00: The MCHx_O output is independent from the CHx_O output;
- MCHxMSEL=2'b01: The MCHx_O output is the same as the CHx_O output and the MCHxOMCTL bits are not used in the generation of the MCHx_O output;
- MCHxMSEL=2'b11: The outputs of MCHx_O and CHx_O are complementary and the MCHxOMCTL bits are not used in the generation of the MCHx_O output.

Function of complementary is for a pair of channels, CHx_O and MCHx_O, the two output signals cannot be active at the same time. The TIMERx has 4 pairs of channels, all the four pairs have this function. The complementary signals CHx_O and MCHx_O are controlled by a group of parameters: the CHxEN and MCHxEN bits in the TIMERx_CHCTL2 register, the POEN, ROS and IOS bits in the TIMERx_CCHP register(when CHx_O and MCHx_O channels has separated deadtime value and break function, please refer to ***Table 17-5. Complementary outputs controlled by parameters (MCHxMSEL =2'b11)***), ISOx and ISOxN bits in the TIMERx_CTL1 register. The output polarity is determined by CHxP and MCHxP bits in the TIMERx_CHCTL2 register.

Table 17-5. Complementary outputs controlled by parameters (MCHxMSEL =2'b11)

Complementary Parameters					Output Status	
POEN	ROS	IOS	CHxEN	MCHxEN	CHx_O	MCHx_O
0	0/1	0	0	0	CHx_O / MCHx_O = LOW CHx_O / MCHx_O output disable(1).	
				1	CHx_O/ MCHx_O output "off-state" (2): the CHx_O/ MCHx_O output inactive level firstly: CHx_O = CHxP, MCHx_O = CHxNP; If the clock for deadtime generator is present, after a deadtime: CHx_O = ISOx, MCHx_O = ISOxN. (3)	
			1	0		
				1		
		1	x	x	CHx_O/ MCHx_O output "off-state": the CHx_O/ MCHx_O output inactive level firstly: CHx_O = CHxP, MCHx_O = CHxNP; If the clock for deadtime generator is present, after a deadtime: CHx_O = ISOx, MCHx_O = ISOxN.	
1	0	0/1	0	0	CHx_O/MCHx_O = LOW CHx_O/MCHx_O output disable.	
				1	CHx_O = LOW CHx_O output disable.	MCHx_O=OxCPRE⊕(4)MCHxP MCHx_O output enable.
			1	0	CHx_O=OxCPRE⊕CHxP CHx_O output enable.	MCHx_O = LOW MCHx_O output disable.
				1	CHx_O=OxCPRE⊕CHxP CHx_O output enable.	MCHx_O=(!OxCPRE)(5)⊕MCHxP MCHx_O output enable.
	1		0	0	CHx_O = CHxP CHx_O output disable.	MCHx_O = MCHxP MCHx_O output disable.
				1	CHx_O = CHxP CHx_O output enable	MCHx_O=OxCPRE⊕MCHxP MCHx_O output enable
			1	0	CHx_O=OxCPRE⊕CHxP CHx_O output enable	MCHx_O = MCHxP MCHx_O output enable.

				1	CHx_O=OxCPRE⊕CHxP CHx_O output enable	MCHx_O=(!OxCPRE)⊕MCH xP MCHx_O output enable.

Note:

(1) output disable: the CHx_O / MCHx_O are disconnected to corresponding pins, the pin is floating with GPIO pull up/down setting which will be Hi-Z if no pull.

(2) "off-state": CHx_O / MCHx_O output with inactive state (e.g., CHx_O = (0⊕CHxP) = CHxP).

(3) See Break mode section for more details.

(4) ⊕: Xor calculate.

(5) (!OxCPRE)： the complementary output of the OxCPRE signal.

Insertion dead time for complementary PWM

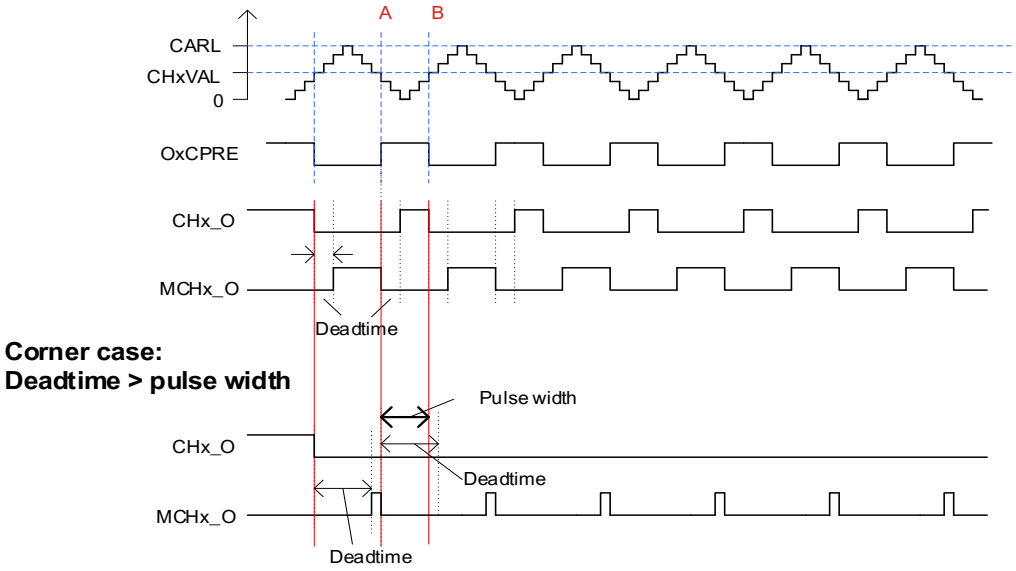
The dead time insertion is enabled when MCHxMSEL=2'b11 and both CHxEN and MCHxEN are configured to 1'b1, it is also necessary to configure POEN to 1. The field named DTCFG defines the dead time delay that can be used for all channels. Refer to the ***Complementary channel protection register(TIMERx_CCHP)*** for details about the delay time.

The dead time delay insertion ensures that two complementary signals are not active at the same time.

When the channel x match event (TIMERx_CNT = CHxVAL) occurs, OxCPRE will be toggled in PWM mode 0. At point A in ***Figure 17-31. Channel output complementary PWM with dead-time insertion***, CHx_O signal remains at the low level until the end of the dead time delay, while MCHx_O signal will be cleared at once. Similarly, at point B when the channelx match event (TIMERx_CNT = CHxVAL) occurs again, OxCPRE is cleared, and CHx_O signal will be cleared at once, while MCHx_O signal remains at the low level until the end of the dead time delay.

Sometimes, we can see corner cases about the dead time insertion. For example: the dead time delay is greater than or equal to the duty cycle of the CHx_O signal, then the CHx_O signal is always inactive (As shown in ***Figure 17-31. Channel output complementary PWM with dead-time insertion***).

Figure 17-31. Channel output complementary PWM with dead-time insertion



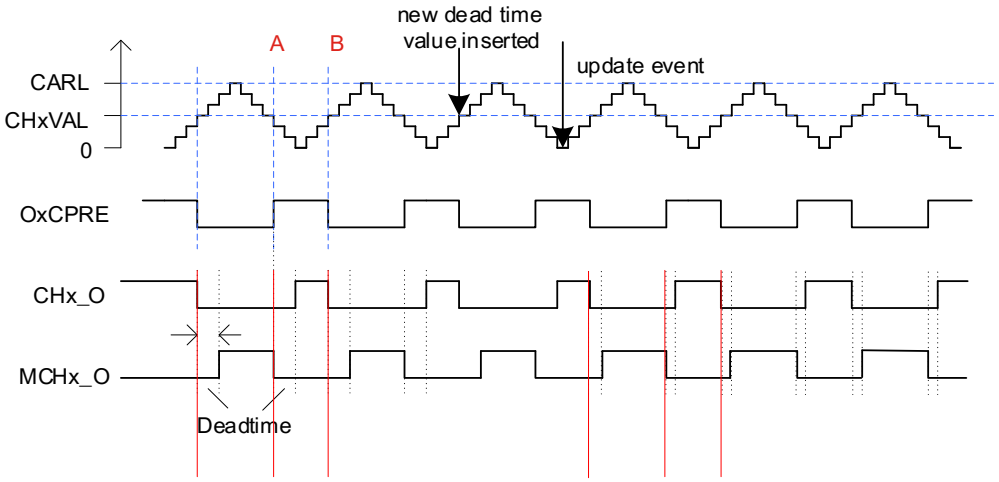
When CHx_O and MCHx_O channels has separated deadtime value, please refer to ***Separated dead time insertion and Break function***.

By configuring the DTIENCHx(x=0..3) bit in the TIMERx_CTL2 register to realize the independent control of dead-time insertion function for each pair of channels. When the DTIENCHx(x=0..3) bit is "0", the corresponding channels CHx_O and MCHx_O will not be inserted into the dead-time.

In order to support the dynamic variation of dead time during running, the dead time shadow function can be enabled by setting DTSREN in the TIMERx_CCHP register. The new dead time will take effect after the next update event occurs.

The channel independent dead time shadow function can be enabled by setting the FCCHPxEN and DTxSREN bits in the TIMEx_FCCHPx register. please refer to ***Figure 17-32. Dead time shadow function***.

Figure 17-32. Dead time shadow function



Break mode

In this function, CHx_O and MCHx_O are controlled by the POEN, OAEN, IOS and ROS bits in the TIMERx_CCHP register, ISOx and ISOxN bits in the TIMERx_CTL1 register.

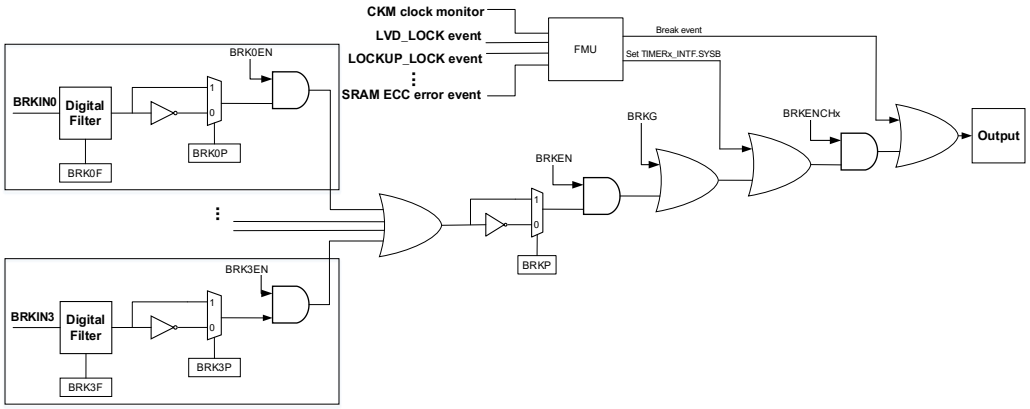
The MCHx_O output is the inverse of the CHx_O output when the MCHxMSEL=2'b11 (the MCHxOMCTL bits are not used in the generation of the MCHx_O output). In this case, CHx_O and MCHx_O signals cannot be set to active level at the same time. The break sources are system events and other events. When the break event occurs, the outputs is force at a predefined level (either active or inactive) after a deadtime duration.

The break event is the result of logic ORed of all sources. The break function can handle two types of event sources:

- External sources: coming from BRKINx(x=0..3) inputs, and with digital filters and polarity selection;
- Internal sources: system sources(such as: HXTAL stuck event which is generated by Clock Monitor CKM in RCU, LVD lock event, Cortex®-M7 core LOCKUP_LOCK event or SRAM ECC error event), and on-chip comparator events(configured in TRIGSEL module, input by BRKIN0 pin).

Break events can also be generated by software using BRKG bit in the TIMERx_SWEVG register.

Figure 17-33. Break function diagram



Note: The polarity of the abort signal from BRKINx (x=0..3) input is jointly determined by the BRKP bit in the TIMERx_CCHP register and the BRKxP bit in the TIMERx_BRKCFG register.

Refer to ***Figure 17-33. Break function diagram***, BRKINx(x=0..3) can from the TRIGSEL module or from the GPIO pins, refer to ***Timer break input select register (SYSCFG_TIMERBKINSEL)***.

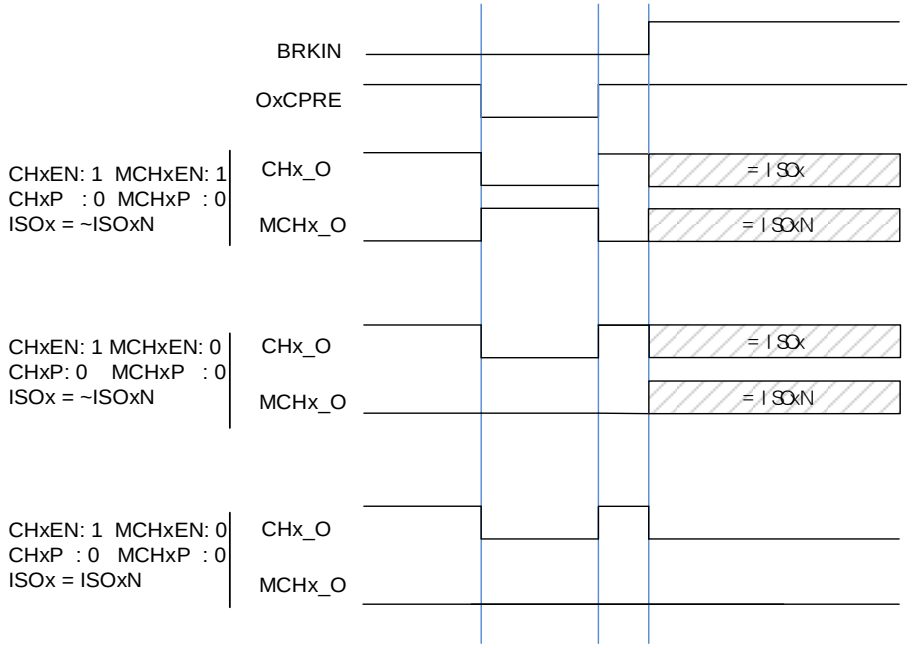
When the abort event source is an external signal source, the abort function is enabled by setting the BRKEN bit in the TIMERx_CCHP register, the BRKENCHx bit in the TIMERx_CTL2 register, and the BRKxEN bit in the TIMERx_BRKCFG register to 1.

When the MCHxMSEL = 2'b11 and a break occurs, the POEN bit is cleared asynchronously. As soon as POEN is 0, the level of the CHx_O and MCHx_O outputs are determined by the

ISOx and ISOxN bits in the TIMERx_CTL1 register. However, when the abort event is generated by an internal signal source, the output will not be controlled by the TIMERx_CTL1 register, and the channel output will be forced low. If IOS = 0, the timer releases the enable output, otherwise the output enable is still controlled by the TIMER.. The complementary outputs are first in the reset state, and then the dead time generator is reactivated to drive the outputs with the level programmed in the ISOx and ISOxN bits after a dead time.

When a break occurs, the BRKIF bit in the TIMERx_INTF register will be set. When the abort event originates from an internal signal source, SYSBIF will also be set in addition to BRKIF. If BRKIE is 1, an interrupt will be generated.

Figure 17-34. Output behavior of the channel in response to a break (the break high active)



When CHx_O and MCHx_O channels has separated break function, please refer to ***Separated dead time insertion and Break function***.

By configuring the BEKENCHx(x=0..3) bit in the TIMERx_CTL2 register to realize the independent control of break function for each pair of channels. When the BEKENCHx(x=0..3) bit is "0" and a break event occurs, the corresponding channels CHx_O and MCHx_O will not be changed and the outputs is keeping.

Separated dead time insertion and Break function

The separated dead time insertion and break function for CHx_O and MCHx_O allows that each pair of channels has its own deadtime value and break function. In this function, CHx_O and MCHx_O are actually controlled by the IOS bit、ROS bit and DTCFG[7:0] bits in TIMERx_FCCHPy(y=0..3) register.

By configuring the FCCHPyEN (y=0..3) bits in the TIMERx_FCCHPy(y=0..3) registers can

select whether each pair of channels uses the separated dead time insertion and break function. When the FCCHPyEN=0, the ROS、IOS and DTCFG[7:0] bits in TIMERx_CCHP register is active; When the FCCHPyEN=1, the ROS、IOS and DTCFG[7:0] bits in TIMERx_FCCHPy register is active.

Quadrature decoder

The quadrature decoder function uses two quadrature inputs CI0 and CI1 derived from the TIMERx_CH0 and TIMERx_CH1 pins respectively to interact with each other to generate the counter value.

Setting TSCFGy[4:0](y=0..2) != 5'b00000 to select that the counting direction of timer is determined only by the CI0, only by the CI1, or by the CI0 and the CI1.

The DIR bit is modified by hardware automatically during the voltage level change of each direction selection source. The mechanism of changing the counter direction is shown in *Table 17-6. Counting direction in different quadrature decoder signals*. The CI0FE0 and CI1FE1 are the signals of the CI0 and CI1 after the filering and polarity selection. The quadrature decoder can be regarded as an external clock with a direction selection. This means that the counter counts continuously from 0 to the counter-reload value. Therefore, users must configure the TIMERx_CAR register before the counter starts to count.

Table 17-6. Counting direction in different quadrature decoder signals

Counting mode	Level	CI0FE0		CI1FE1	
		Rising	Falling	Rising	Falling
Quadrature decoder mode 0 TSCFG0[4:0]!= 5'b00000	CI1FE1=1	Down	Up	-	-
	CI1FE1=0	Up	Down	-	-
Quadrature decoder mode 1 TSCFG1[4:0]!= 5'b00000	CI0FE0=1	-	-	Up	Down
	CI0FE0=0	-	-	Down	Up
Quadrature decoder mode 2 TSCFG2[4:0]!= 5'b00000	CI1FE1=1	Down	Up	X	X
	CI1FE1=0	Up	Down	X	X
	CI0FE0=1	X	X	Up	Down
	CI0FE0=0	X	X	Down	Up

Note: "-" means "no counting"; "X" means impossible. "0" means "low level", "1" means "high level".

Figure 17-35. Example of counter operation in decoder interface mode(Quadrature

decoder mode 2)

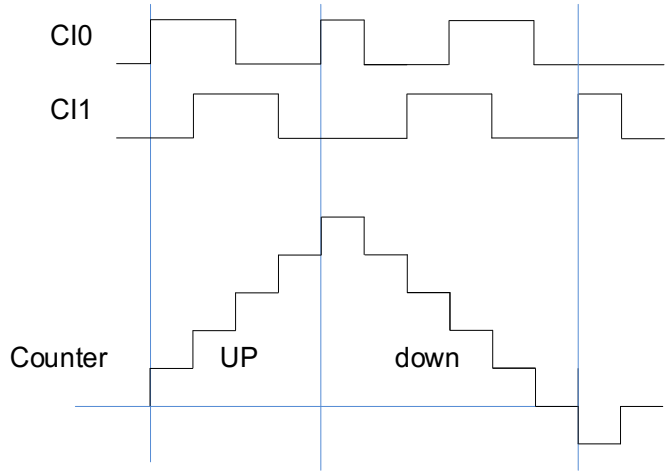
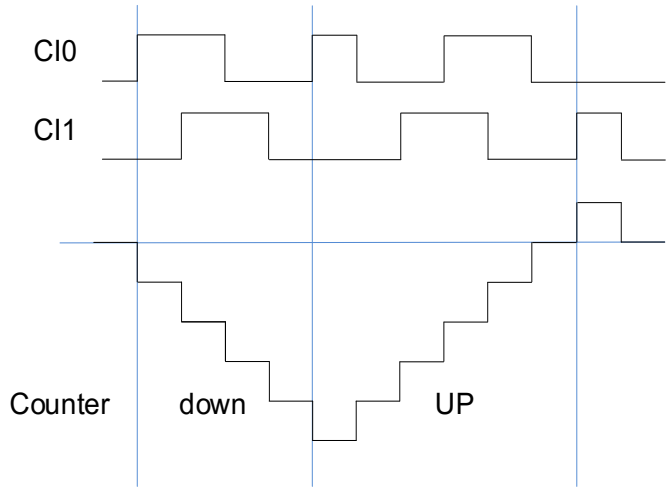


Figure 17-36. Example of decoder interface mode with CI0FE0 polarity inverted(Quadrature decoder mode 2)



Quadrature decoder signal detection

The quadrature decoder signal jump detection function can be enabled by setting the DECJDEN bit (in TIMERx_CTL2 register) to 1, which can be used to detect whether the level jump edges (rising or falling) of the CI0 and CI1 signals occur at the same time.

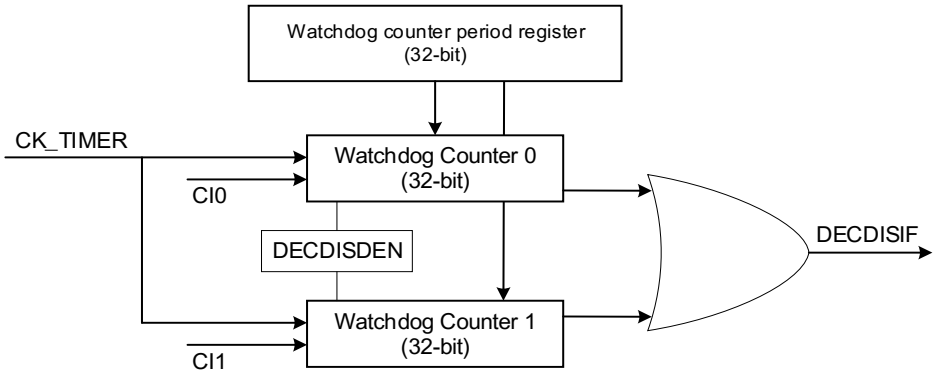
When DECJDEN =1, if the level transitions of the two quadrature signals CI0 and CI1 occur simultaneously, the interrupt flag DECJIF is set, if DECJIE=1, the corresponding interrupt is generated.

The quadrature decoder signal disconnection detection function can be enabled by setting the DECDISDEN bit (in TIMERx_CTL2 register) to 1, which can be used to detect the signal conditions of the CI0 and CI1 signals.

As shown in *Figure 17-37. Quadrature decoder signal disconnection detection block diagram*. The signal detection module includes two 32-bit watchdog counters and a period register. The CI0 and CI1 signals are used to reset the two watchdog counters respectively.

When DECDISDEN=1, two watchdog counters start counting up at the same time. If the counter countinues to count to the watchdog period value (this value is determined by the WDGPER[31:0] bit-field in the TIMERx_WDGPER register), the counter is timeout and the interrupt flag DECDISIF is set. If DECDISIE=1, the corresponding interrupt is generated.

Figure 17-37. Quadrature decoder signal disconnection detection block diagram



Decoder

The decoder function has four modes: decoder mode 0~3, which can be selected by setting TSCFGy[4:0] (y=8, 9, 10, 11) != 5'b00000.There are two input sources in these two modes: CI0 and CI1, and the CI0FE0 and CI1FE1 are the signals of the CI0 and CI1 after the filering and polarity selection.

When the decoder mode 0 and decoder mode 1 are enabled, the CI0FE0 is used as the count direction signal and the CI1FE1 signal is used as the count pulse.

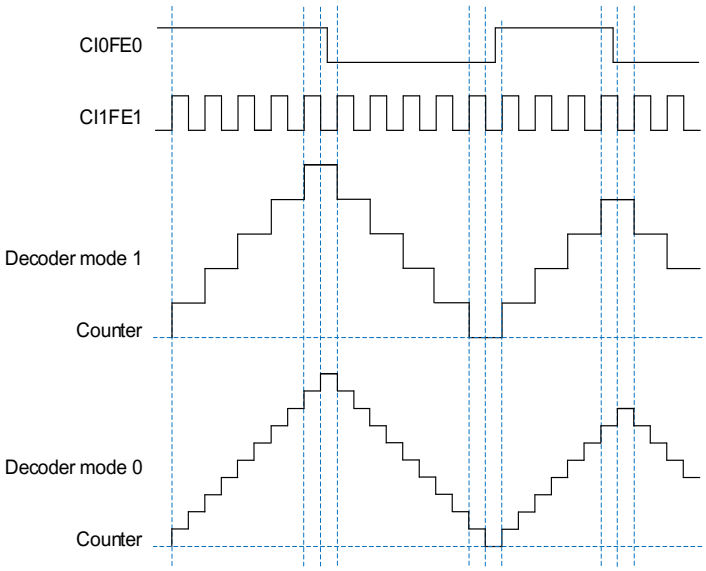
The CH0P is used to configure the count direction selection: When CH0P =0, the counter will count up when the CI0FE0 signal is high and will count down when the CI0FE0 signal is low; When CH0P =1, the counter will count down when the CI0FE0 signal is high and will count up when the CI0FE0 signal is low.

The CH1P is used to configure the count pulse edge selection: In decoder mode 0, the counter will count on both rising and falling edges of the CI1FE1 signal. In decoder mode 1, when CH1P=0, the counter will count on the rising edge of the CI1FE1 signal; When CH1P=1, the counter will count on the falling edge of the CI1FE1 signal. The more details for decoder mode 1 is shown in *Table 17-7. the counter operation in decoder mode 1*

Table 17-7. the counter operation in decoder mode 1

CH1P	level	counter operation
0	CI0FE0 is high	the counter will count up on the rising edge of the CI1FE1 input signal
	CI0FE0 is low	the counter will count down on the rising edge of the CI1FE1 input signal
1	CI0FE0 is high	the counter will count up on the falling edge of the CI1FE1 input signal
	CI0FE0 is low	the counter will count down on the falling edge of the CI1FE1 input signal

Figure 17-38. Example of counter operation in decoder mode 0 / 1 with CH1P=0



The decoder mode 2/3 uses two inputs CI0FE0 and CI1FE1 respectively to interact with each other to generate the counter value, and the counting direction DIR bit modified by hardware automatically.

When in decoder mode 2, the counter will count on both rising and falling edges of CI0FE0 and CI1FE1 signals. And the counting direction determined by the CH0P and CH1P bits.

When in decoder mode 3, the counter will count on rising or falling edge of CI0FE0 and CI1FE1 signals. When CHxP=0, the counter will counter on the high level or the falling edge or the CIxFEx signal; When CHxP=1, the counter will counter on the low level or the rising edge or the CIxFEx signal. The more details for decoder mode 2/3 is shown in ***Figure 17-39. Example of counter operation in decoder mode 2 / 3 (CH0P / CH1P=0)*** and ***Table 17-8. the counter operation in decoder mode 2 / 3***.

Figure 17-39. Example of counter operation in decoder mode 2 / 3 (CH0P / CH1P=0)

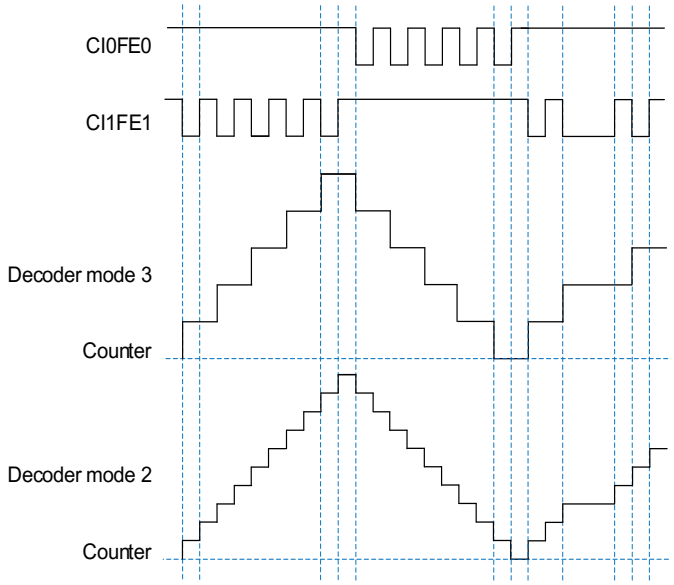


Table 17-8. the counter operation in decoder mode 2 / 3

Counting mode	Polarity	Level	CI0FE0		CI1FE1	
			Rising	Falling	Rising	Falling
decoder mode 2 TSCFG10[4:0]!= 5'b00000	CHxP=0 (x=0 or 1)	CI1FE1=1	Down	Down	x	x
		CI1FE1=0	-	-	x	x
		CI0FE0=1	x	x	Up	Up
		CI0FE0=0	x	x	-	-
	CHxP=1 (x=0 or 1)	CI1FE1=1	-	-	x	x
		CI1FE1=0	Down	Down	x	x
		CI0FE0=1	x	x	-	-
		CI0FE0=0	x	x	Up	Up
decoder mode 3 TSCFG11[4:0]!= 5'b00000	CHxP=0 (x=0 or 1)	CI1FE1=1	-	Down	x	x
		CI1FE1=0	-	-	x	x
		CI0FE0=1	x	x	-	Up
		CI0FE0=0	x	x	-	-
	CHxP=1 (x=0 or 1)	CI1FE1=1	-	-	x	x
		CI1FE1=0	Down	-	x	x
		CI0FE0=1	x	x	-	-
		CI0FE0=0	x	x	Up	-

Note: "-" means "no counting"; "X" means impossible. "0" means "low level", "1" means "high level".

When the counter direction changes in the decoder modes, the DIR bit in TIMERx_CTL0 register within a change, at the same time the DIRTRANIF bit in TIMERx_INTF is set to 1. And the corresponding interrupt is generated if the DIRTRANIE bit in TIMERx_DMAINTEN register is set to 1.

Hall sensor function

Hall sensor is generally used to control BLDC motor, the timers can support this function.

Figure 17-40. Hall sensor is used for BLDC motor shows how to connect the timer and the motor. And two timers are needed. TIMER_in(Advanced/General L0 TIMER)is used to accept three HALL sensor signals.

Each of the three input of HALL sensors provides a pulse which is applied to an input capture pin, then both the speed and position of rotor can be calculated by analyzing the hall sensor signals.

By the internal connection function (TRGO-ITIx), TIMER_in and TIMER_out can be connected. TIMER_out will generate PWM signals to control the speed of BLDC motor based on the ITIx. Then, the feedback circuit is finished, you can change the configuration to fit your request.

Because the advanced/general L0 TIMER has the input XOR function, they can be used as the TIMER_in timer. And the advanced timer has the functions of complementary output and dead time, so it can be used as the TIMER_out timer.

In addition, the interconnected timer can be seleted by TRIGSEL module, for example :

TIMER_in (TIMER0) -> TIMER_out (TIMER7 ITI0)

TIMER_in (TIMER1) -> TIMER_out (TIMER0 ITI1)

After appropriate interconnected timers are selected and wires are connected, the timers need to be configured. Some key settings are as follows:

- Enable XOR by setting TI0S, then, the change of each input signal will make the CI0 toggle. CH0VAL will record the current value of counter.
- Choose ITIx to trigger commutation by configuring CCUC and CCSE.
- Configure PWM parameters based on the requests.

Figure 17-40. Hall sensor is used for BLDC motor

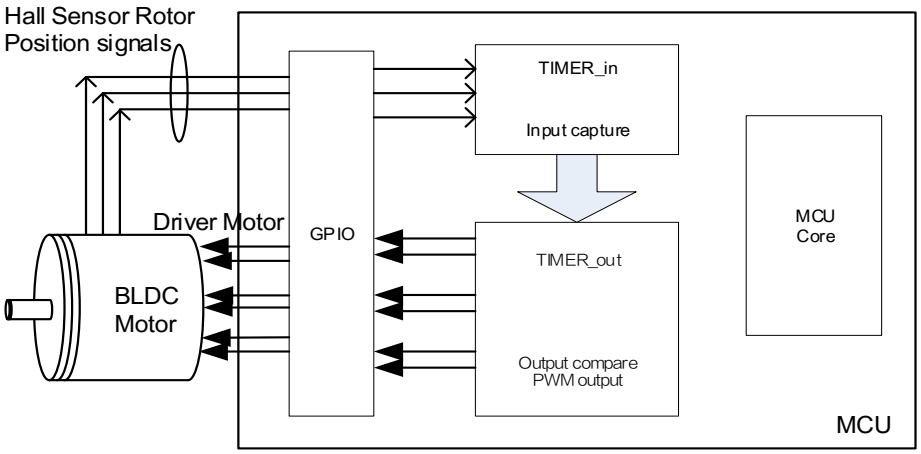
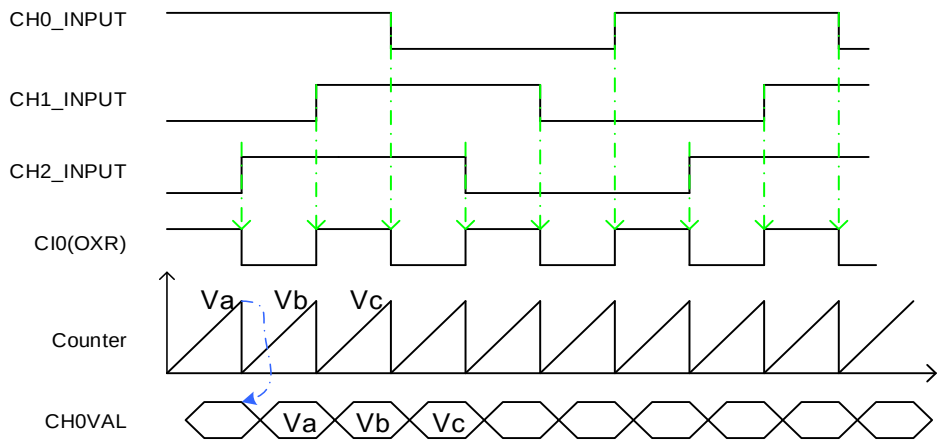
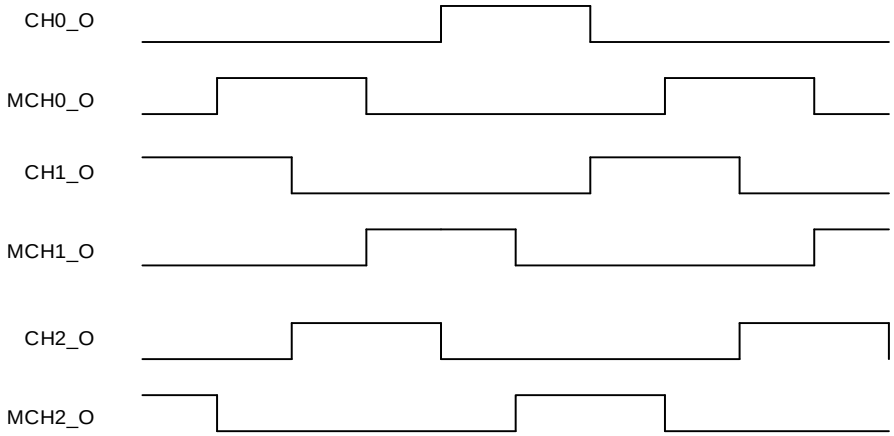


Figure 17-41. Hall sensor timing between two timers

Advanced/General L0 TIMER_in under input capture mode



Advanced TIMER_out under output compare mode(PWM with Dead-time)



Master-slave management

The TIMERx can be synchronized with a trigger in several modes including restart mode, pause mode and event mode which is selected by the TSCFGy[4:0] in SYSCFG_TIMERxCFG(x=3,4,5).

Table 17-9. Examples of slave mode

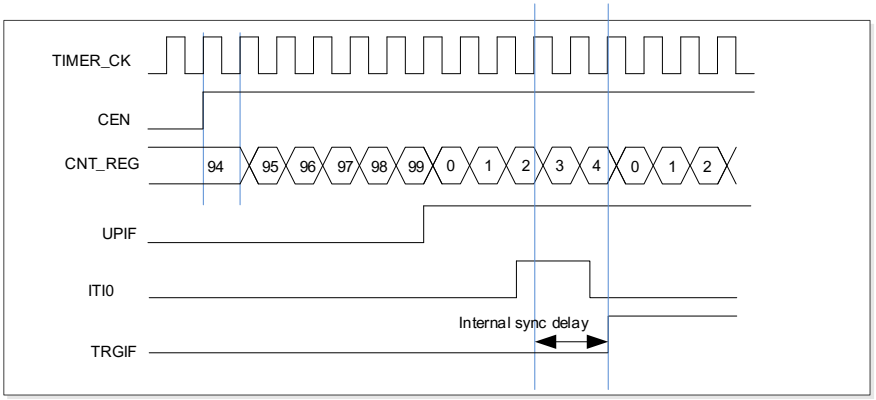
	Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
LIST	TSCFGy[4:0] y=3 (restart mode) y=4 (pause mode) y=5 (event mode)	TSCFGy[4:0] 00001: Internal trigger input 0 (ITI0) 00010: Internal trigger input 1 (ITI1) 00011: Internal trigger input 2 (ITI2) 00100: Internal trigger input 3 (ITI3)	If CI0FE0 or CI1FE1 is selected as the trigger source, configure the CHxP and MCHxP for the polarity selection and inversion. If ETIFP (the filtered output of external	For the ITIx, no filter and prescaler can be used. For the CIx, filter can be used by configuring CHxCAPFLT, no prescaler can be used. For the ETIFP, filter

Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
	00101: Internal trigger input 4 (ITI4) 00110: Internal trigger input 5 (ITI5) 00111: Internal trigger input 6 (ITI6) 01000: Internal trigger input 7 (ITI7) 01001: Internal trigger input 8 (ITI8) 01010: Internal trigger input 9 (ITI9) 10101: CI0 edge flag (CI0F_ED) 10110: The filtered output of channel 0 input (CI0FE0) 10111: The filtered output of channel 1 input (CI1FE1) 11000: The filtered output of external trigger input (ETIFP) 11001：The filtered output of channel 2 input (CI2FE2) 11010：The filtered output of channel 3 input (CI3FE3) 11011：The filtered output of multi mode channel 0 input (MCI0FEM0) 11100：The filtered output of multi mode channel 1 input (MCI1FEM1) 11101：The filtered output of multi mode channel 2 input (MCI2FEM2) 11110：The filtered output of multi mode	trigger input ETI) is selected as the trigger source, configure the ETP for polarity selection and inversion.	can be used by configuring ETFC and prescaler can be used by configuring ETPSC.

Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
	channel 3 input (MCI3FEM3)		
Restart mode The counter will be cleared and restart when a rising edge of trigger input comes.	TSCFG3[4:0] = 5'b 00001ITI0 is selected.	For ITI0, no polarity selector can be used.	For the ITI0, no filter and prescaler can be used.

Figure 17-42. Restart mode

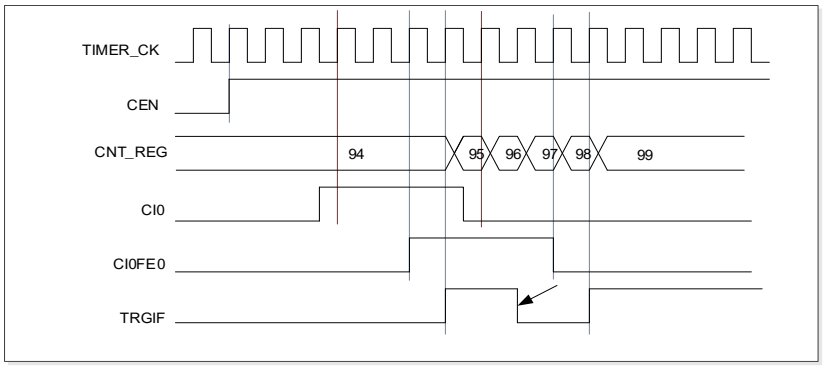
Exam1



Pause mode The counter will be paused when the trigger input is low, and it will start when the trigger input is high.	TSCFG4[4:0] = 5'b 10110 CI0FE0 is selected.	TI0S=0 (Non-xor) [MCH0P=0, CH0P=0] CI0FE0 does not invert. The capture event will occur on the rising edge only.	Filter is bypassed in this example.

Figure 17-43. Pause mode

Exam2

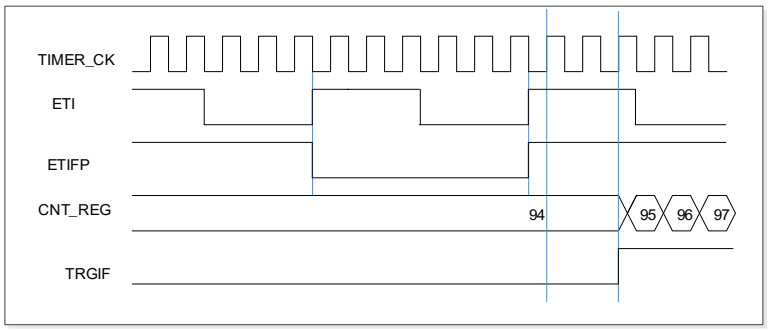


Event mode The counter will start to count when a rising edge of trigger input comes.	TSCFG5[4:0] = 5'b 11000 ETIFP is selected.	ETP = 0, the polarity of ETI does not change.	ETPSC = 1, ETI is divided by 2. ETFC = 0, ETI does not filter.

Exam3

Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler

Figure 17-44. Event mode



(1) The ETI pin can select from TIMER_ETIx(x=0,1,2,7,19,20,60,61,62,63) pins, and each advanced TIMER only can use one of them.

Single pulse mode

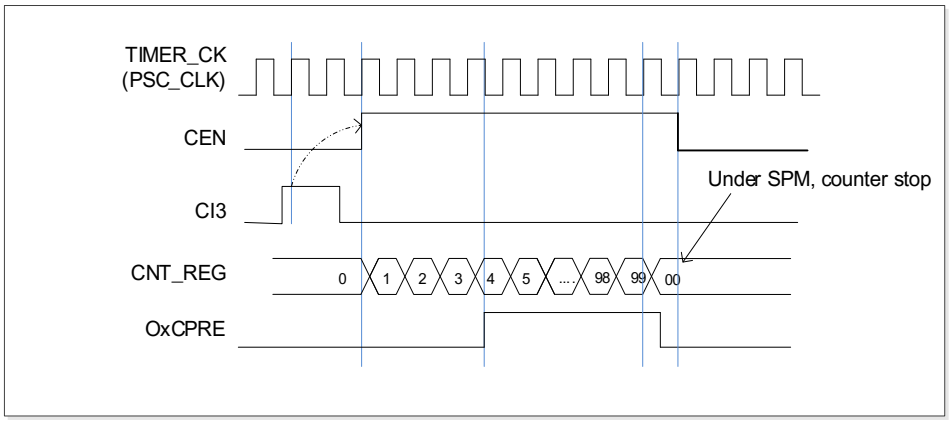
Single pulse mode is opposite to the repetitive mode, which can be enabled by setting SPM in TIMERx_CTL0. When you set SPM, the counter will be clear and stop when the next update event. In order to get pulse waveform, you can set the TIMERx to PWM mode or compare by CHxCOMCTL.

Once the timer is set to the single pulse mode, it is not necessary to configure the timer enable bit CEN in the TIMERx_CTL0 register to 1 to enable the counter. Setting the CEN bit to 1 or a trigger signal edge can generate a pulse and then keep the CEN bit at a high state until the update event occurs or the CEN bit is written to 0 by software. If the CEN bit is cleared to 0 by software, the counter will be stopped and its value will be held.

In the single pulse mode, the active edge of trigger which sets the CEN bit to 1 will enable the counter. However, there exists several clock delays to perform the comparison result between the counter value and the TIMERx_CHxCV value. After a trigger rising occurs in the single pulse mode, the OxCPRE signal will immediately be forced to the state which the OxCPRE/ MOxCPRE signals will change to, as the compare match event occurs without taking the comparison result into account.

Single pulse mode is also applicable to composite PWM mode (CHxCPWMEN = 1'b1 and CHxMS[2:0] = 3'b000).

Figure 17-45. Single pulse mode TIMERx_CHxCV=0x04, TIMERx_CAR=0x99

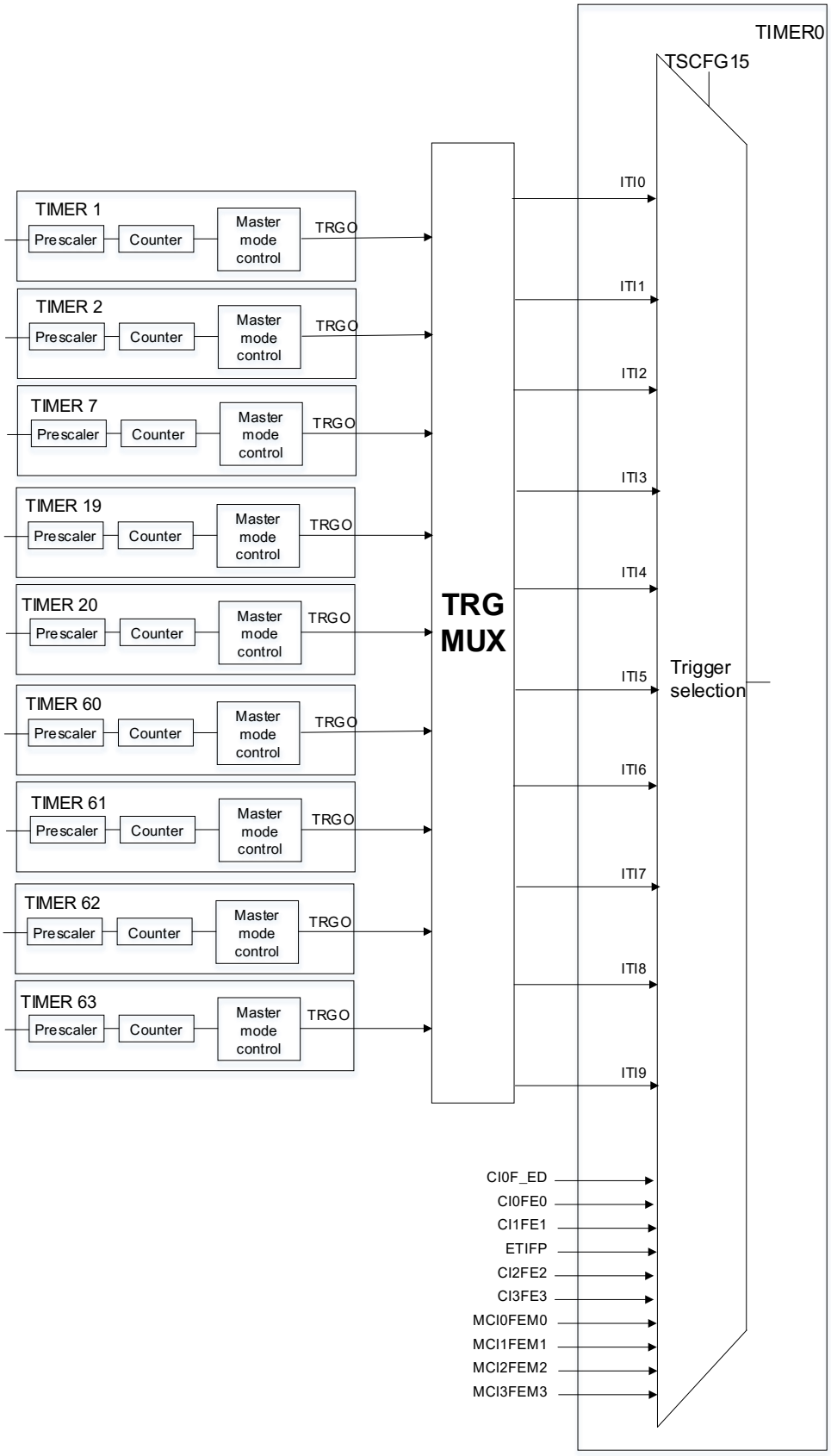


Timers interconnection

The timers can be internally connected for timer chaining or synchronization. This can be implemented by configuring one timer to operate in the master mode while configuring another timer to be in the slave mode. The following figures show several examples of trigger selection for the master mode and slave mode.

Figure 17-46. TIMER0 master/slave mode example shows the TIMER0 trigger selection when it is configured in slave mode.

Figure 17-46. TIMER0 master/slave mode example



Other interconnection examples:

■ TIMER1 as the prescaler for TIMER0

TIMER1 is configured as a prescaler for TIMER0. Refer to ***Figure 17-46. TIMER0 master/slave mode example*** for connections. Steps are shown as follows:

1. Configure TIMER1 in master mode and select its update event (UPE) as trigger output (MMC0=4'b0010 in the TIMER1_CTL1 register). Then TIMER1 drives a periodic signal on each counter overflow.
2. Configure TIMER1 period (TIMER1_CAR register).
3. Configure TIMER0 in external clock mode 0 and select the TIMER1 as TIMER0 input trigger source (TRCFG6[4:0] = 5b'01001 in the_SYSCFG_TIMER0CFG1 register).
4. Start TIMER0 by writing '1' to the CEN bit (TIMER0_CTL0 register).
5. Start TIMER1 by writing '1' to the CEN bit (TIMER1_CTL0 register).

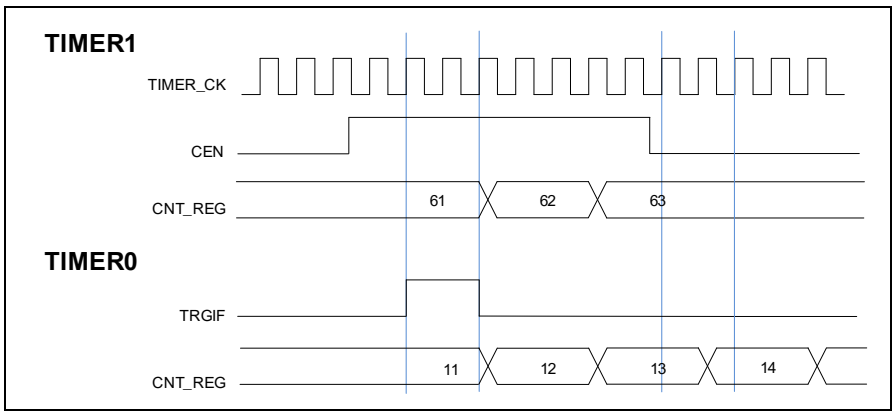
■ Start TIMER0 with TIMER1's enable signal

First, enable TIMER0 with the enable signal of TIMER1. Refer to ***Figure 17-47. Triggering TIMER0 with enable signal of TIMER1***. TIMER0 starts counting from its current value with the divided internal clock after being triggered by TIMER1 enable signal output.

When TIMER0 receives the trigger signal, its CEN bit is set and the counter counts until TIMER0 is disabled. Both clock frequency of the counters are divided by 3 from TIMER_CK ($f_{PSC_CLK} = f_{TIMER_CK}$ /3). Steps are shown as follows:

1. Configure TIMER1 in master mode to send its enable signal as trigger output (MMC0=4'b0001 in the TIMER1_CTL1 register).
2. Configure TIMER0 in event mode and select the TIMER1 as TIMER0 input trigger source (TRCFG5[4:0] = 5b'01001 in the_SYSCFG_TIMER0CFG0 register).
3. Start TIMER1 by writing 1 to the CEN bit (TIMER1_CTL0 register).

Figure 17-47. Triggering TIMER0 with enable signal of TIMER1



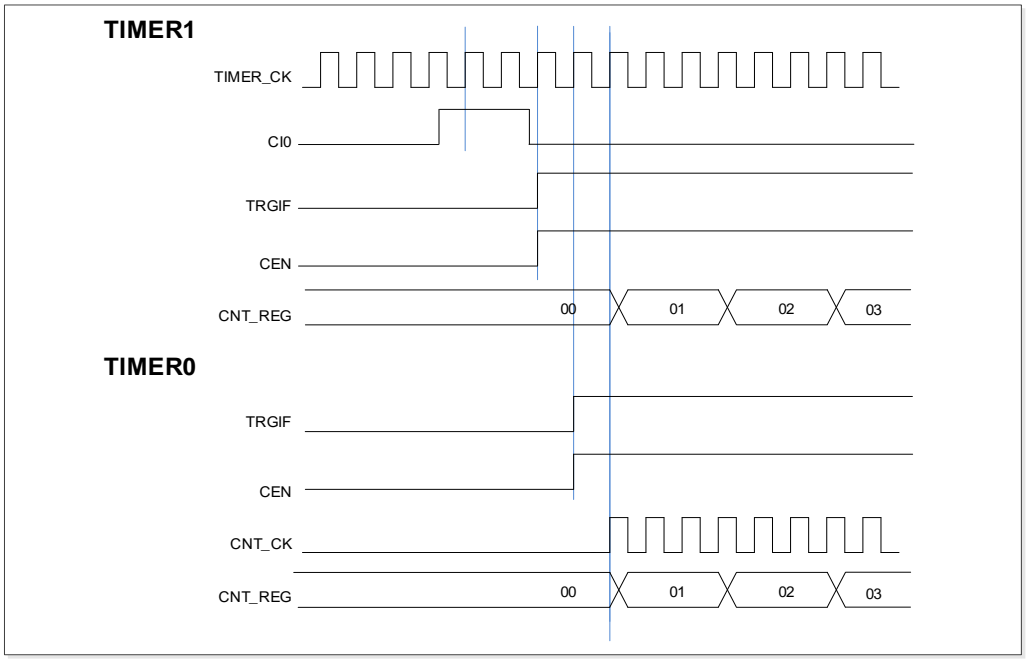
■ Using an external trigger to start two timers synchronously.

The start of TIMER0 is triggered by the enable signal of TIMER1, and TIMER1 is triggered by its CI0 input rising edge. To ensure that two timers start synchronously, TIMER1 must be configured in master/slave mode. Steps are shown as follows:

1. Configure TIMER1 in event mode and select the CI0F_ED as TIMER1 input trigger source (TRCFG5[4:0] = 5b' 10101 in the_SYSCFG_TIMER1CFG0 register).
2. Configure TIMER1 in master / slave mode by writing MSM=1 (TIMER1_SMCFG register).
3. Configure TIMER0 in event mode and select the TIMER2 as TIMER0 input trigger source (TRCFG5[4:0] = 5b' 10101 in the_SYSCFG_TIMER0CFG0 register).

When the CI0 signal of TIMER1 generates a rising edge, two timer counters start counting synchronously with the internal clock and both TRGIF flags are set.

Figure 17-48. Triggering TIMER0 and TIMER1 with TIMER1's CI0 input

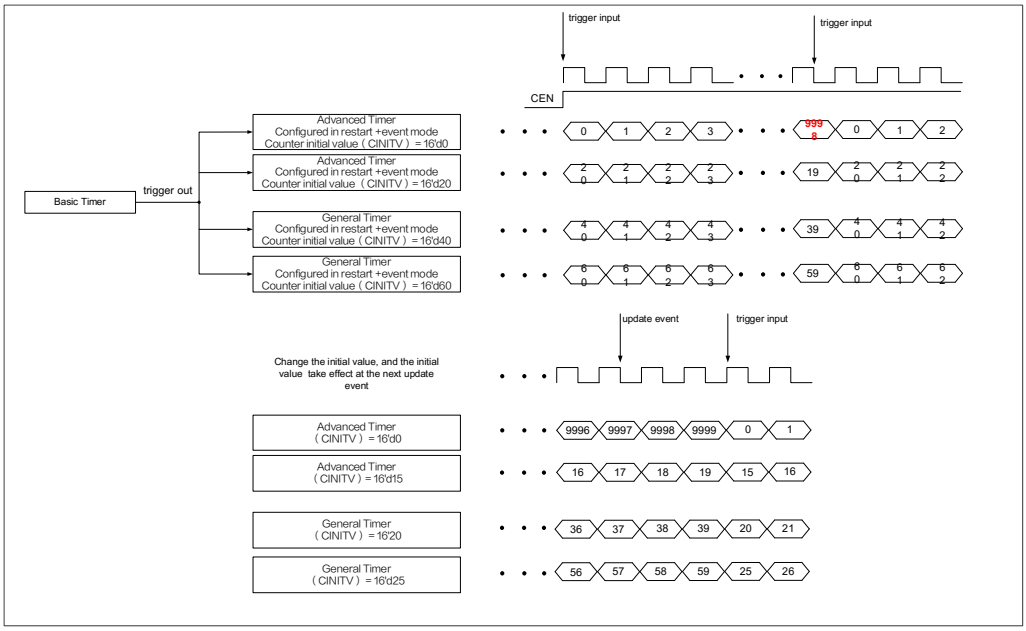


In some interconnection configurations, several timers are triggered and synchronized to start counting at the same time. several timers may have phase offsets after long counts and the phase offsets is eliminated by periodically refreshing the synchronization counter with an external trigger, it is possible to control the initial value of the phase between several timers by configuring counter initial control register(TIMERx_CINITV).

Four slave timers are configured in reset + event mode. the initial values of the timer can be preload by setting the CINITVEN bit of the TIMERx_CINITCTL register. In the following figure, the initial load of the four slave timers is 0,20,40,60 respectively.So there's an arithmetic shift of 20 CK_TIMER.

The 9998 count in the figure below shows the phase shift between the counter buses of different timers after a long count. The four timers count value is refreshed synchronization by trigger output of basic timer, and the phase offset is avoided.

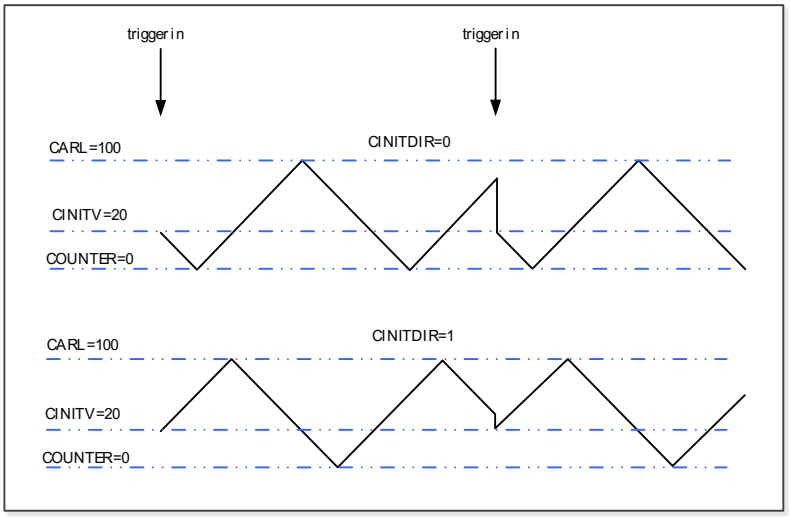
Figure 17-49. Triggering four timers by trigger out of basic timer



The counting direction after resetting the counter can be configured by CINITDIR bit in TIMERx_CINITCTL register when the counter is in center-aligned counting mode. CINITDIR only takes effect when CINITVEN in TIMERx_CINITCTL register is enabled, and CINITDIR does not take effect when the counter is in up/down counting mode.

When CINITDIR is 0, the counting direction is down after resetting the counter. When CINITDIR is 1, the count direction is up after resetting the counter.

Figure 17-50. Counting direction in CINITDIR is 0 or 1



Timer DMA mode

Timer DMA mode is the function that configures timer's register by DMA module. The relative registers are TIMERx_DMACFG and TIMERx_DMATB. Corresponding DMA request bit should be asserted to enable DMA request for internal interrupt event. TIMERx will send a request to DMA when the interrupt event occurs. DMA is configured to M2P (memory to

peripheral) mode and the address of TIMERx_DMATB is configured to PADDR (peripheral base address), then DMA will access the TIMERx_DMATB. In fact, TIMERx_DMATB register is only a buffer, timer will map the TIMERx_DMATB to an internal register, appointed by the field of DMATA in TIMERx_DMACFG. If the field of DMATC in TIMERx_DMACFG is 0 (1 transfer), the timer sends only one DMA request. While if TIMERx_DMATC is not 0, such as 3 (4 transfers), then timer will send 3 more requests to DMA, and DMA will access timer's registers DMATA+0x4, DMATA+0x8 and DMATA+0xC at the next 3 accesses to TIMERx_DMATB. In a word, one-time DMA internal interrupt event asserts, (DMATC+1) times request will be sent by TIMERx.

If one more DMA request event occurs, TIMERx will repeat the process above.

Timer debug mode

When the Cortex®-M7 core is halted, and the TIMERx_HOLD configuration bit in DBG_CTL register is set to 1, the TIMERx counter stops.

17.1.5. Registers definition (TIMERx, x=0, 7, 19, 20, 60 , 61 , 62 , 63)

TIMER0 base address: 0x4001 2C00

TIMER7 base address: 0x4001 3400

TIMER19 base address: 0x4001 5000

TIMER20 base address: 0x4000 0000

TIMER60 base address: 0x4000 1000

TIMER61 base address: 0x4000 D000

TIMER62 base address: 0x4001 6400

TIMER63 base address: 0x4000 D400

Table 17-10. TIMERx register overview

Register Name	Symbol	Address Offset	Reset Value
Control register 0	TIMERx_CTL0	0x00	0x0000 0000
Control register 1	TIMERx_CTL1	0x04	0x0000 0000
Slave mode configuration register	TIMERx_SMCFG	0x08	0x3000 0000
DMA and interrupt enable register 0	TIMERx_DMAINTEN0	0x0C	0x0000 0000
DMA and interrupt enable register 1	TIMERx_DMAINTEN1	0x8C	0x0000 0000
Interrupt flag register	TIMERx_INTF	0x10	0x0000 0000
Software event generation register	TIMERx_SWEVG	0x14	0x0000 0000
Channel control register 0	TIMERx_CHCTL0	0x18	0x0000 0000
Channel control register 1	TIMERx_CHCTL1	0x1C	0x0000 0000
Channel control register 2	TIMERx_CHCTL2	0x20	0x0000 0000
Counter register	TIMERx_CNT	0x24	0x0000 0000
Prescaler register	TIMERx_PSC	0x28	0x0000 0000
Counter auto reload register	TIMERx_CAR	0x2C	0x0000 FFFF
Counter repetition register	TIMERx_CREP	0x30	0x0000 0000
Channel 0 capture/compare value register	TIMERx_CH0CV	0x34	0x0000 0000
Channel 1 capture/compare value register	TIMERx_CH1CV	0x38	0x0000 0000
Channel 2 capture/compare value register	TIMERx_CH2CV	0x3C	0x0000 0000
Channel 3 capture/compare value register	TIMERx_CH3CV	0x40	0x0000 0000
Complementary channel protection register	TIMERx_CCHP	0x44	0x0000 0000
Multi mode channel control register 0	TIMERx_MCHCTL0	0x48	0x0000 0000
Multi mode channel control register 1	TIMERx_MCHCTL1	0x4C	0x0000 0000
Multi mode channel control register 3	TIMERx_MCHCTL2	0x50	0x0000 0000
Multi mode channel 0 capture/compare value register	TIMERx_MCH0CV	0x54	0x0000 0000
Multi mode channel 1 capture/compare value register	TIMERx_MCH1CV	0x58	0x0000 0000
Multi mode channel 2 capture/compare value register	TIMERx_MCH2CV	0x5C	0x0000 0000
Multi mode channel 3 capture/compare value register	TIMERx_MCH3CV	0x60	0x0000 0000
Channel 0 additional capture/compare value register	TIMERx_CH0CV_ADD	0x64	0x0000 0000
Channel 1 additional capture/compare value register	TIMERx_CH1CV_ADD	0x68	0x0000 0000
Channel 2 additional capture/compare	TIMERx_CH2CV_ADD	0x6C	0x0000 0000

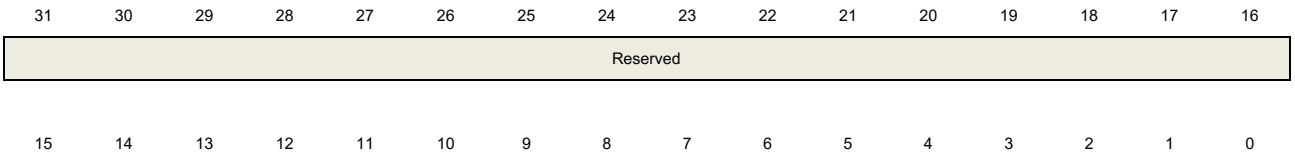
Register Name	Symbol	Address Offset	Reset Value
value register			
Channel 3 additional capture/compare value register	TIMERx_CH3CV_ADD	0x70	0x0000 0000
Control register 2	TIMERx_CTL2	0x74	0x0FF0 00FF
Break configuration register	TIMERx_BRKCFG	0x78	0x5500 0000
Free complementary channel protection register 0	TIMERx_FCCHP0	0x7C	0x0000 0000
Free complementary channel protection register 1	TIMERx_FCCHP1	0x80	0x0000 0000
Free complementary channel protection register 2	TIMERx_FCCHP2	0x84	0x0000 0000
Free complementary channel protection register 3	TIMERx_FCCHP3	0x88	0x0000 0000
Multi mode channel 0 additional capture/compare value register	TIMERx_MCH0CV_ADD	0x90	0x0000 0000
Multi mode channel 1 additional capture/compare value register	TIMERx_MCH1CV_ADD	0x94	0x0000 0000
Multi mode channel 2 additional capture/compare value register	TIMERx_MCH2CV_ADD	0x98	0x0000 0000
Multi mode channel 3 additional capture/compare value register	TIMERx_MCH3CV_ADD	0x9C	0x0000 0000
Watchdog counter period register	TIMERx_WDGPER	0xA0	0xFFFF FFFF
Counter initial control register	TIMERx_CINITCTL	0xA4	0x0000 0000
Counter initial value register	TIMERx_CINITV	0xA8	0x0000 0000
Channel multi-edge capture configuration register	TIMERx_CHMECAPCFG	0xAC	0x0000 0000
Multi mode channel multi-edge capture configuration register	TIMERx_MCHMECAPCFG	0xB0	0x0000 0000
Channel 0 multi-edge capture register	TIMERx_CH0MECAP	0xB4	0x0000 0000
Channel 1 multi-edge capture register	TIMERx_CH1MECAP	0xB8	0x0000 0000
Channel 2 multi-edge capture register	TIMERx_CH2MECAP	0xBC	0x0000 0000
Channel 3 multi-edge capture register	TIMERx_CH3MECAP	0xC0	0x0000 0000
Multi mode channel 0 multi-edge capture register	TIMERx_MCH0MECAP	0xC4	0x0000 0000
Multi mode channel 1 multi-edge capture register	TIMERx_MCH1MECAP	0xC8	0x0000 0000
Multi mode channel 2 multi-edge capture register	TIMERx_MCH2MECAP	0xCC	0x0000 0000
Multi mode channel 3 multi-edge capture register	TIMERx_MCH3MECAP	0xD0	0x0000 0000
Control register 3	TIMERx_CTL3	0xD4	0x0000 0000
DMA configuration register	TIMERx_DMACFG	0xE0	0x0000 0000
DMA transfer buffer register	TIMERx_DMATB	0xE4	0x0000 0000
Configuration register	TIMERx_CFG	0xFC	0x0000 0000

Control register 0 (TIMERx_CTL0)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Reserved	CFBEN	CKDIV[1:0]	ARSE	CAM[1:0]	DIR	SPM	UPS	UPDIS	CEN
	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:11	Reserved	Must be kept at reset value.
10	CFBEN	Counter freeze in break mode enable 0: The counter is not freeze in break mode when CFBEN is reset 1: The counter is freeze in break mode when CFBEN is set Note: This bit is valid when the BRKEN is placed in the TIMER x_CCHP.
9:8	CKDIV[1:0]	Clock division The CKDIV bits can be configured by software to specify division factor between the CK_TIMER and the dead-time and digital filter sample clock (DTS). 00: f_{DTS}= f_{CK_TIMER} 01: f_{DTS}= f_{CK_TIMER} /2 10: f_{DTS}= f_{CK_TIMER} /4 11: Reserved
7	ARSE	Auto-reload shadow enable 0: The shadow register for TIMERx_CAR register is disabled 1: The shadow register for TIMERx_CAR register is enabled
6:5	CAM[1:0]	Counter align mode selection 00: No center-aligned mode (edge-aligned mode). The direction of the counter is specified by the DIR bit. 01: Center-aligned and counting down assert mode. The counter counts in center-aligned mode and channel is configured in output mode (CHxMS = 3'b000 in TIMERx_CHCTL0 register). Only when counting down, CHxIF/MCHxIF bit and CHxCOMADDIF/MCHxADDCOMIF can be set. 10: Center-aligned and counting up assert mode. The counter counts in center-aligned mode and channel is configured in output mode (CHxMS = 3'b000 in TIMERx_CHCTL0 register). Only when counting up, CHxIF/MCHxIF bit and CHxCOMADDIF/MCHxADDCOMIF can be set. 11: Center-aligned and counting up/down assert mode. The counter counts in center-aligned mode and channel is configured in output mode (CHxMS = 3'b000 in TIMERx_CHCTL0 register). Both when counting up and counting down, CHxIF/MCHxIF bit and CHxCOMADDIF/MCHxADDCOMIF can be set. After the counter is enabled, these bits cannot be switched from 0x00 to non 0x00.
4	DIR	Direction 0: Count up 1: Count down If the timer work in center-aligned mode or decoder mode, this bit is read only.
3	SPM	Single pulse mode 0: Single pulse mode is disabled. The counter continues after an update event.

1: Single pulse mode is enabled. The counter counts until the next update event occurs.

2 UPS Update source

This bit is used to select the update event sources by software.

0: These events generate an update interrupt or a DMA request:

- The UPG bit is set.
- The counter generates an overflow or underflow event.
- The restart mode generates an update event.

1: This event generates update interrupts or DMA requests:

The counter generates an overflow or underflow event

1 UPDIS Update disable

This bit is used to enable or disable the update event generation.

0: Update event enable. When an update event occurs, the corresponding shadow registers are loaded with their preloaded values. These events generate update event:

- The UPG bit is set.
- The counter generates an overflow or underflow event.
- The restart mode generates an update event.

1: Update event disable.

Note: When this bit is set to 1, setting UPG bit or restart mode does not generate an update event, but the counter and prescaler are initialized.

0 CEN Counter enable

0: Counter disable

1: Counter enable

The CEN bit must be set by software when timer works in external clock mode, pause mode or decoder mode.

Control register 1 (TIMERx_CTL1)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCUC[2:1]		Reserved				MMC0[3]	Reserved								
rw						rw									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ISO3N	ISO3	ISO2N	ISO2	ISO1N	ISO1	ISO0N	ISO0	TI0S	MMC0[2:0]			DMAS	CCUC[0]	Reserved	CCSE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw			rw	rw		rw

Bits	Fields	Descriptions
31:30	CCUC[2:1]	Refer to CCUC[0] bit.

29:26	Reserved	Must be kept at reset value.
25	MMC0[3]	Refer to MMC0[2:0] description.
24:16	Reserved	Must be kept at reset value.
15	ISO3N	Idle state of multi mode channel 3 complementary output. Refer to ISO0N bit.
14	ISO3	Idle state of channel 3 output. Refer to ISO0 bit.
13	ISO2N	Idle state of multi mode channel 2 complementary output. Refer to ISO0N bit
12	ISO2	Idle state of channel 2 output Refer to ISO0 bit
11	ISO1N	Idle state of multi mode channel 1 complementary output Refer to ISO0N bit
10	ISO1	Idle state of channel 1 output Refer to ISO0 bit
9	ISO0N	Idle state of multi mode channel 0 complementary output 0: When POEN bit is reset, MCH0_O is set low. 1: When POEN bit is reset, MCH0_O is set high. This bit can be modified only when PROT[1:0] bits in TIMERx_CCHP register is 00.
8	ISO0	Idle state of channel 0 output 0: When POEN bit is reset, CH0_O is set low. 1: When POEN bit is reset, CH0_O is set high. The CH0_O output changes after a dead time if MCH0_O is implemented. This bit can be modified only when PROT[1:0] bits in TIMERx_CCHP register is 00.
7	TI0S	Channel 0 trigger input selection 0: The TIMERx_CH0 pin input is selected as channel 0 trigger input. 1: The result of combinational XOR of TIMERx_CH0, TIMERx_CH1 and TIMERx_CH2 pins is selected as channel 0 trigger input.
6:4	MMC0[2:0]	Master mode control This bit-field combine with MMC0[3] control the selection of TRGO signal, which is sent in master mode to slave timers for synchronization function. 0000: When a counter reset event occurs, a TRGO trigger signal is output. The counter resert source: Master timer generate a reset the UPG bit in the TIMERx_SWEVG register is set 0001: Enable. When a counter start event occurs, a TRGO trigger signal is output. The counter start source : CEN control bit is set

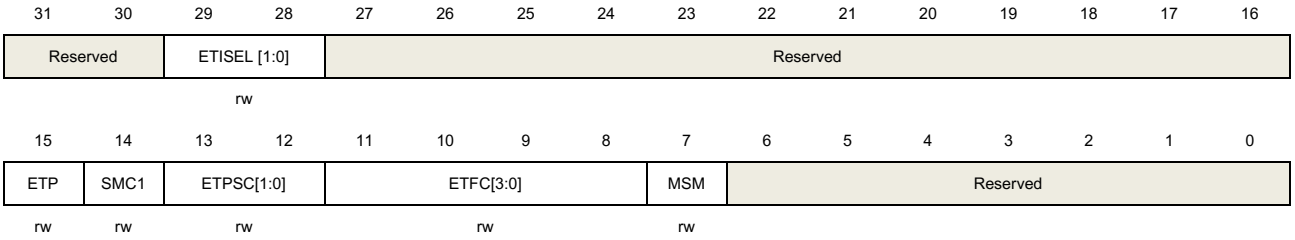
		The trigger input in pause mode is high 0010: When an update event occurs, a TRGO trigger signal is output. The update source depends on UPDIS bit and UPS bit. 0011: When a capture or compare pulse event occurs in channel0, a TRGO trigger signal is output. 0100: When a compare event occurs, a TRGO trigger signal is output. The compare source is from O0CPRE. 0101: When a compare event occurs, a TRGO trigger signal is output. The compare source is from O1CPRE. 0110: When a compare event occurs, a TRGO trigger signal is output. The compare source is from O2CPRE. 0111: When a compare event occurs, a TRGO trigger signal is output. The compare source is from O3CPRE. 1001: When a overflow event occurs, a TRGO trigger signal is output. 1010: When a underflow event occurs, a TRGO trigger signal is output. 1011: When a software event occurs, a TRGO trigger signal is output. Other: reserved.
3	DMAS	DMA request source selection 0: When capture or compare event occurs, the DMA request of channel x is sent 1: When update event occurs, the DMA request of channel x is sent.
2	CCUC[0]	Commutation control shadow register update control CCUC[2:1] and CCUC[0] are used to control the update of the shadow register. When the commutation control shadow registers (for CHxEN, MCHxEN and CHxCOMCTL bits) are enabled (CCSE=1), the update control of the shadow registers are shown as below: 000: The shadow registers update when CMTG bit is set. 001: The shadow registers update when CMTG bit is set or a rising edge of TRGI occurs. 100: The shadow registers update when an overflow event generate. 101: The shadow registers update when an underflow event generate. 110: The shadow registers update when an overflow or underflow event generate. Others: reserved. When a channel does not have a complementary output, this bit has no effect.
1	Reserved	Must be kept at reset value.
0	CCSE	Commutation control shadow enable 0: The shadow registers (for CHxEN, MCHxEN, CHxCOMCTL,CHxP,MCHxP bits) are disabled. 1: The shadow registers (for CHxEN, MCHxEN, CHxCOMCTL,CHxP,MCHxP bits) are enabled. After these bits have been written, they are updated when commutation event comes. When a channel does not have a complementary output, this bit has no effect.

Slave mode configuration register (TIMERx_SMCFG)

Address offset: 0x08

Reset value: 0x3000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29:28	ETISEL[1:0]	TIMER0 external trigger select 00: Timer external trigger 0 01: Timer external trigger 1 10: Timer external trigger 2 11: Reserved
27:16	Reserved	Must be kept at reset value.
15	ETP	External trigger polarity This bit specifies the polarity of ETI signal. 0: ETI is active at rising edge or high level. 1: ETI is active at falling edge or low level.
14	SMC1	Part of SMC is used to enable external clock mode 1 In external clock mode 1, the counter is clocked by any active edge of the ETIFP signal. 0: External clock mode 1 disabled 1: External clock mode 1 enabled When the slave mode is configured as restart mode, pause mode or event mode, the timer can still work in the external clock 1 mode by setting this bit. But the TRGS bits must not be 3'b111 in this case. The external clock input will be ETIFP if external clock mode 0 and external clock mode 1 are enabled at the same time. **Note:** External clock mode 0 enable is in SYSCFG_TIMERxCFG1 register's TSCFG6[4:0] bit field.
13:12	ETPSC[1:0]	The prescaler of external trigger The frequency of external trigger signal ETIFP must not be higher than 1/4 of TIMER_CK frequency. When the frequency of external trigger signal is high, the prescaler can be enabled to reduce ETIFP frequency. 00: Prescaler disabled

01: The prescaler is 2
10: The prescaler is 4
11: The prescaler is 8

11:8 ETFC[3:0] External trigger filter control

The external trigger can be filtered by digital filter and this bit-field configure the filtering capability.

Basic principle of digital filter: continuously sample the external trigger signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit-field, it is considered to be an effective level.

The filtering capability configuration is as follows:

EXTFC[3:0]	Times	f_{SAMP}
4'b0000	Filter disabled.	
4'b0001	2	f_{CK_TIMER}
4'b0010	4	
4'b0011	8	
4'b0100	6	$f_{DTS_CK}/2$
4'b0101	8	
4'b0110	6	$f_{DTS_CK}/4$
4'b0111	8	
4'b1000	6	$f_{DTS_CK}/8$
4'b1001	8	
4'b1010	5	$f_{DTS_CK}/16$
4'b1011	6	
4'b1100	8	
4'b1101	5	$f_{DTS_CK}/32$
4'b1110	6	
4'b1111	8	

7 MSM Master-slave mode

This bit can be used to synchronize the selected timers to begin counting at the same time. The TRGI is used as the start event, and through TRGO, timers are connected.

0: Master-slave mode disabled
1: Master-slave mode enabled

6:0 Reserved Must be kept at reset value.

DMA and interrupt enable register 0 (TIMERx_DMAINTEN0)

Address offset: 0x0C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

CH3COMADDIE	CH2COMADDIE	CH1COMADDIE	CH0COMADDIE	MCH3DEN	MCH2DEN	MCH1DEN	MCH0DEN	MCH3IE	MCH2IE	MCH1IE	MCH0IE	Reserved	DECDISIE	DECJIE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3 2	1	0

Reserved	TRGDEN	CMTDEN	CH3DEN	CH2DEN	CH1DEN	CH0DEN	UPDEN	BRKIE	TRGIE	CMTIE	CH3IE	CH2IE	CH1IE	CH0IE	UPIE
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31	CH3COMADDIE	Channel 3 additional compare interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used in composite PWM mode and asymmetric PWM mode(when CH3MS[2:0] = 3'b000 and CH3COMCTL[3:2]=2'b10).
30	CH2COMADDIE	Channel 2 additional compare interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used in composite PWM mode and asymmetric PWM mode(when CH3MS[2:0] = 3'b000 and CH3COMCTL[3:2]=2'b10).
29	CH1COMADDIE	Channel 1 additional compare interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used in composite PWM mode and asymmetric PWM mode(when CH3MS[2:0] = 3'b000 and CH3COMCTL[3:2]=2'b10).
28	CH0COMADDIE	Channel 0 additional compare interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used in composite PWM mode and asymmetric PWM mode(when CH3MS[2:0] = 3'b000 and CH3COMCTL[3:2]=2'b10).
27	MCH3DEN	Multi mode channel 3 capture/compare DMA request enable 0: Disabled 1: Enabled **Note:** This bit just used for channel input and output independent mode (when MCH3MSEL[1:0] = 2b'00).
26	MCH2DEN	Multi mode channel 2 capture/compare DMA request enable 0: Disabled 1: Enabled **Note:** This bit just used for channel input and output independent mode (when MCH2MSEL[1:0] = 2b'00).
25	MCH1DEN	Multi mode channel 1 capture/compare DMA request enable 0: Disabled 1: Enabled

		Note: This bit just used for channel input and output independent mode (when MCH1MSEL[1:0] = 2b'00).
24	MCH0DEN	Multi mode channel 0 capture/compare DMA request enable 0: Disabled 1: Enabled **Note:** This bit just used for channel input and output independent mode (when MCH0MSEL[1:0] = 2b'00).
23	MCH3IE	Multi mode channel 3 capture/compare interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used for channel input and output independent mode (when MCH3MSEL[1:0] = 2b'00).
22	MCH2IE	Multi mode channel 2 capture/compare interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used for channel input and output independent mode (when MCH2MSEL[1:0] = 2b'00).
21	MCH1IE	Multi mode channel 1 capture/compare interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used for channel input and output independent mode (when MMCH1SEL[1:0] = 2b'00).
20	MCH0IE	Multi mode channel 0 capture/compare interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used for channel input and output independent mode (when MMCH0SEL[1:0] = 2b'00).
19:18	Reserved	Must be kept at reset value.
17	DECDISIE	Quadrature decoder signal disconnection interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used for quadrature decoder signal disconnection detection is enabled (when DECDISDEN =1).
16	DECJIE	Quadrature decoder signal jump (the two signals jump at the same time) interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used for quadrature decoder signal jump detection is enabled (when DECJDEN =1).

15	Reserved	Must be kept at reset value.
14	TRGDEN	Trigger DMA request enable 0: Disabled 1: Enabled
13	CMTDEN	Commutation DMA request enable 0: Disabled 1: Enabled
12	CH3DEN	Channel 3 capture/compare DMA request enable 0: Disabled 1: Enabled
11	CH2DEN	Channel 2 capture/compare DMA request enable 0: Disabled 1: Enabled
10	CH1DEN	Channel 1 capture/compare DMA request enable 0: Disabled 1: Enabled
9	CH0DEN	Channel 0 capture/compare DMA request enable 0: Disabled 1: Enabled
8	UPDEN	Update DMA request enable 0: Disabled 1: Enabled
7	BRKIE	Break interrupt enable 0: Disabled 1: Enabled
6	TRGIE	Trigger interrupt enable 0: Disabled 1: Enabled
5	CMTIE	Commutation interrupt enable 0: Disabled 1: Enabled
4	CH3IE	Channel 3 capture/compare interrupt enable 0: Disabled 1: Enabled
3	CH2IE	Channel 2 capture/compare interrupt enable 0: Disabled 1: Enabled
2	CH1IE	Channel 1 capture/compare interrupt enable

		0: Disabled 1: Enabled
1	CH0IE	Channel 0 capture/compare interrupt enable 0: Disabled 1: Enabled
0	UPIE	Update interrupt enable 0: Disabled 1: Enabled

DMA and interrupt enable register1 (TIMERx_DMAINTEN1)

Address offset: 0x8C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												MCH3COMADDIE	MCH2COMADDIE	MCH1COMADDIE	MCH0COMADDIE
												rw	rw	rw	rw

Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3	MCH3COMADDIE	Refer to MCH0COMADDIE bit.
2	MCH2COMADDIE	Refer to MCH0COMADDIE bit.
1	MCH1COMADDIE	Refer to MCH0COMADDIE bit.
0	MCH0COMADDIE	Multi mode Channel 0 additional compare interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used in composite PWM mode and asymmetric PWM mode(when CH3MS[2:0] = 3'b000 and CH3COMCTL[3:2]=2'b10).

Interrupt flag register (TIMERx_INTF)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

CH3COMADDIF	CH2COMADDIF	CH1COMADDIF	CH0COMADDIF	MCH3OF	MCH2OF	MCH1OF	MCH0OF	MCH3IF	MCH2IF	MCH1IF	MCH0IF	MCH3COMADDIF	MCH2COMADDIF	DECDISIF	DECJIF
rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

MCH1COMADDIF	MCH0COMADDIF	SYSBIF	CH3OF	CH2OF	CH1OF	CH0OF	Reserved	BRKIF	TRGIF	CMTIF	CH3IF	CH2IF	CH1IF	CH0IF	UPIF
rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	.	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

Bits	Fields	Descriptions
31	CH3COMADDIF	Channel 3 additional compare interrupt flag. Refer to CH0COMADDIF description.
30	CH2COMADDIF	Channel 2 additional compare interrupt flag. Refer to CH0COMADDIF description.
29	CH1COMADDIF	Channel 1 additional compare interrupt flag. Refer to CH0COMADDIF description.
28	CH0COMADDIF	Channel 0 additional compare interrupt flag. This flag is set by hardware and cleared by software. If channel 0 is in output mode, this flag is set when a compare event occurs. 0: No channel 0 output compare interrupt occurred 1: Channel 0 output compare interrupt occurred **Note:** This bit just used in composite PWM mode and asymmetric PWM mode(when CH3MS[2:0] = 3'b000 and CH3COMCTL[3:2]=2'b10).
27	MCH3OF	Multi mode channel 3 over capture flag Refer to MCH0OF description.
26	MCH2OF	Multi mode channel 2 over capture flag Refer to MCH0OF description.
25	MCH1OF	Multi mode channel 1 over capture flag Refer to MCH0OF description.
24	MCH0OF	Multi mode channel 0 over capture flag When multi mode channel 0 is configured in input mode, this flag is set by hardware when a capture event occurs while MCH0IF flag has already been set. This flag is cleared by software. 0: No over capture interrupt occurred 1: Over capture interrupt occurred
23	MCH3IF	Multi mode channel 3 capture/compare interrupt flag Refer to MCH0IF description
22	MCH2IF	Multi mode channel 2 capture/compare interrupt flag Refer to MCH0IF description
21	MCH1IF	Multi mode channel 1 capture/compare interrupt flag

		Refer to MCH0IF description
20	MCH0IF	Multi mode channel 0 capture/compare interrupt flag This flag is set by hardware and cleared by software. If multi mode channel 0 is in input mode, this flag is set when a capture event occurs. If multi mode channel 0 is in output mode, this flag is set when a compare event occurs. If multi mode channel 0 is set to input mode, this bit will be reset by reading TIMERx_MCH0CV. 0: No multi mode channel 0 capture/compare interrupt occurred 1: Multi mode channel 0 capture/compare interrupt occurred
19	MCH3ADDCOMIF	Multi mode channel 3 additional compare interrupt flag. Refer to MCH0ADDCOMIF description.
18	MCH2ADDCOMIF	Multi mode channel 2 additional compare interrupt flag. Refer to MCH0ADDCOMIF description.
17	DECDISIF	Quadrature decoder signal disconnection interrupt flag 0: No quadrature decoder signal disconnection interrupt occurred 1: Quadrature decoder signal disconnection interrupt occurred **Note:** This bit just used for quadrature decoder signal disconnection detection is enabled (when DECDISDEN =1).
16	DECJIF	Quadrature decoder signal jump (the two signals jump at the same time) interrupt flag 0: No quadrature decoder signal jump interrupt occurred 1: Quadrature decoder signal jump interrupt occurred **Note:** This bit just used for quadrature decoder signal jump detection is enabled (when DECJDEN =1).
15	MCH1ADDCOMIF	Multi mode channel 1 additional compare interrupt flag. Refer to MCH0ADDCOMIF description.
14	MCH0ADDCOMIF	Multi mode channel 0 additional compare interrupt flag. This flag is set by hardware and cleared by software. If multi mode channel 0 is in output mode, this flag is set when a compare event occurs. 0: No multi mode channel 0 output compare interrupt occurred 1: multi mode channel 0 output compare interrupt occurred **Note:** This bit just used in composite PWM mode and asymmetric PWM mode(when CH3MS[2:0] = 3'b000 and CH3COMCTL[3:2]=2'b10).
13	SYSBIF	System source break interrupt flag This flag is set by hardware when the system sources are active, and cleared by software if the system sources are inactive. 0: No system source break interrupt occurred 1: System source break interrupt occurred

		Note: When this bit is set, this bit must be cleared by software before the channel outputs are restored.
12	CH3OF	Channel 3 over capture flag Refer to CH0OF description
11	CH2OF	Channel 2 over capture flag Refer to CH0OF description
10	CH1OF	Channel 1 over capture flag Refer to CH0OF description
9	CH0OF	Channel 0 over capture flag When channel 0 is configured in input mode, this flag is set by hardware when a capture event occurs while CH0IF flag has already been set. This flag is cleared by software. 0: No over capture interrupt occurred 1: Over capture interrupt occurred
8	Reserved	Must be kept at reset value.
7	BRKIF	Break interrupt flag When the break input is inactive, the bit is set by hardware. When the break input is inactive, the bit can be cleared by software. 0: No active level break has been detected. 1: An active level has been detected.
6	TRGIF	Trigger interrupt flag This flag is set on trigger event and cleared by software. When in pause mode, both edges on trigger input generates a trigger event, otherwise, only an active edge on trigger input can generates a trigger event. 0: No trigger event occurred 1: Trigger interrupt occurred
5	CMTIF	Channel commutation interrupt flag This flag is set by hardware when the commutation event of channel occurs, and cleared by software. 0: No channel commutation interrupt occurred 1: Channel commutation interrupt occurred
4	CH3IF	Channel 3 capture/compare interrupt flag Refer to CH0IF description
3	CH2IF	Channel 2 capture/compare interrupt flag Refer to CH0IF description
2	CH1IF	Channel 1 capture/compare interrupt flag Refer to CH0IF description
1	CH0IF	Channel 0 capture/compare interrupt flag

This flag is set by hardware and cleared by software.
If channel 0 is in input mode, this flag is set when a capture event occurs. If channel 0 is in output mode, this flag is set when a compare event occurs.
If channel 0 is set to input mode, this bit will be reset by reading TIMERx_CH0CV.
0: No channel 0 interrupt occurred
1: Channel 0 interrupt occurred

0 UPIF Update interrupt flag
This bit is set by hardware when an update event occurs and cleared by software.
0: No update interrupt occurred
1: Update interrupt occurred

Software event generation register (TIMERx_SWEVG)

Address offset: 0x14
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH3COMADDG	CH2COMADDG	CH1COMADDG	CH0COMADDG	MCH3COMADDG	MCH2COMADDG	MCH1COMADDG	MCH0COMADDG	MCH3G	MCH2G	MCH1G	MCH0G	Reserved			
w	w	w	w	w	w	w	w	w	w	w	w				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BRKG	TRGG	CMTG	CH3G	CH2G	CH1G	CH0G	UPG
								w	w	w	w	w	w	w	w

Bits	Fields	Descriptions
31	CH3COMADDG	Channel 3 additional compare event generation. Refer to CH0COMADDG description.
30	CH2COMADDG	Channel 2 additional compare event generation. Refer to CH0COMADDG description.
29	CH1COMADDG	Channel 1 additional compare event generation. Refer to CH0COMADDG description.
28	CH0COMADDG	Channel 0 additional compare event generation. This bit is set by software to generate a compare event in channel 0 additional , it is automatically cleared by hardware. When this bit is set, the CH0COMADDIF flag will be set, and the corresponding interrupt will be sent if enabled. 0: No generate a channel 0 additional compare event 1: Generate a channel 0 additional compare event **Note:** This bit just used in composite PWM mode (when CH0CPWMEN=1, CH0MS[2:0] = 3'b000 and CH0COMCTL=3'b110 or 3'b111).
27	MCH3COMADDG	Multi mode Channel 3 additional compare event generation.

		Refer to MCH0COMADDG description.
26	MCH2COMADDG	Multi mode Channel 2 additional compare event generation. Refer to MCH0COMADDG description.
25	MCH1COMADDG	Multi mode Channel 1 additional compare event generation. Refer to MCH0COMADDG description.
24	MCH0COMADDG	Multi mode Channel 0 additional compare event generation. This bit is set by software to generate a compare event in multi mode channel 0, it is automatically cleared by hardware. When this bit is set, the MCH0COMADDIF flag will be set, and the corresponding interrupt will be sent if enabled. 0: No generate a multi mode channel 0 additional compare event 1: Generate a multi mode channel 0 additional compare event **Note:** This bit just used in composite PWM mode and asymmetric PWM mode(when CH0MS[2:0] = 3'b000 and CH0COMCTL[3:2]=2'b10).
23	MCH3G	Multi mode channel 3 capture or compare event generation. Refer to MCH0G description.
22	MCH2G	Multi mode channel 2 capture or compare event generation. Refer to MCH0G description.
21	MCH1G	Multi mode channel 1 capture or compare event generation. Refer to MCH0G description.
20	MCH0G	Multi mode channel 0 capture or compare event generation. This bit is set by software to generate a capture or compare event in multi mode channel 0, it is automatically cleared by hardware. When this bit is set, the MCH0IF flag will be set, and the corresponding interrupt or DMA request will be sent if enabled. In addition, if multi mode channel 0 is configured in input mode, the current value of the counter is captured to TIMERx_MCH0CV register, and the MCH0OF flag is set if the MCH0IF flag has been set. 0: No generate a multi mode channel 0 capture or compare event 1: Generate a multi mode channel 0 capture or compare event
19:8	Reserved	Must be kept at reset value.
7	BRKG	Break event generation This bit is set by software to generate an event and cleared by hardware automatically. When this bit is set, the POEN bit will be cleared and BRKIF flag will be set, related interrupt or DMA transfer can occur if enabled. 0: No generate a break event 1: Generate a break event
6	TRGG	Trigger event generation This bit is set by software and cleared by hardware automatically. When this bit is set, the TRGIF flag in TIMERx_INTF register will be set, related interrupt or DMA

transfer can occur if enabled.
0: No generate a trigger event
1: Generate a trigger event

5 CMTG
Channel commutation event generation
This bit is set by software and cleared by hardware automatically. When this bit is set, channel's capture/compare control registers (CHxEN, MCHxEN and CHxCOMCTL bits) are updated based on the value of CCSE (in the TIMERx_CTL1).
0: No affect
1: Generate channel commutation update event

4 CH3G
Channel 3 capture or compare event generation
Refer to CH0G description

3 CH2G
Channel 2 capture or compare event generation
Refer to CH0G description

2 CH1G
Channel 1 capture or compare event generation
Refer to CH0G description

1 CH0G
Channel 0 capture or compare event generation
This bit is set by software to generate a capture or compare event in channel 0, it is automatically cleared by hardware. When this bit is set, the CH0IF flag will be set, and the corresponding interrupt or DMA request will be sent if enabled. In addition, if channel 0 is configured in input mode, the current value of the counter is captured to TIMERx_CH0CV register, and the CH0OF flag is set if the CH0IF flag has been set.
0: No generate a channel 0 capture or compare event
1: Generate a channel 0 capture or compare event

0 UPG
Update event generation
This bit can be set by software, and automatically cleared by hardware. When this bit is set, the counter is cleared if the center-aligned or up counting mode is selected, while in down counting mode it takes the auto-reload value. The prescaler counter is cleared at the same time.
0: No generate an update event
1: Generate an update event

Channel control register 0 (TIMERx_CHCTL0)

Address offset: 0x18
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH1MS [2]	CH0MS [2]	CH1COM ADDSEN	CH0COM ADDSEN			Reserved	CH1COM CTL[3]	Reserved							CH0COM CTL[3]

		Reserved	Reserved	CH1COMADDUPSEL	CH0COMADDUPSEL										
rw	rw	rw	rw	rw	rw		rw								rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1COMCEN	CH1COMCTL[2:0]			CH1COMSEN	CH1COMFEN	CH1MS[1:0]		CH0COMCEN	CH0COMCTL[2:0]			CH0COMSEN	CH0COMFEN	CH0MS[1:0]	
CH1CAPFLT[3:0]				CH1CAPPSC[1:0]				CH0CAPFLT[3:0]				CH0CAPPSC[1:0]			
rw				rw		rw		rw				rw		rw	

Output compare mode:

Bits	Fields	Descriptions
31	CH1MS[2]	Channel 1 I/O mode selection Refer to CH1MS[1:0]description
30	CH0MS[2]	Channel 0 I/O mode selection Refer to CH0MS[1:0] description
29	CH1COMADDSEN	Channel 1 additional compare output shadow enable Refer to CH0COMADDSEN description.
28	CH0COMADDSEN	Channel 0 additional compare output shadow enable When this bit is set, the shadow register of TIMERx_CH0COMV_ADD register which updates at each update event will be enabled. 0: Channel 0 additional compare output shadow disabled 1: Channel 0 additional compare output shadow enabled The PWM mode can be used without validating the shadow register only in single pulse mode (SPM bit in TIMERx_CTL0 register is set). This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 and CH0MS bit-field is 000.
27	CH1COMADDUPSEL	Refer to CH0COMADDUPSEL.
26	CH0COMADDUPSEL	Update selection of channel 0 additional compare value 0: Channel 0 additional compare value is update from update event. 1: Channel 0 additional compare value is update from compare event.
25	Reserved	Must be kept at reset value.
24	CH1COMCTL[3]	Refer to CH1COMCTL[2:0]
23:17	Reserved	Must be kept at reset value.
16	CH0COMCTL[3]	Refer to CH0COMCTL[2:0]
15	CH1COMCEN	Channel 1 output compare clear enable Refer to CH0COMCEN description
14:12	CH1COMCTL[2:0]	Channel 1 compare output control Refer to CH0COMCTL description

11	CH1COMSEN	Channel 1 output compare shadow enable Refer to CH0COMSEN description
10	CH1COMFEN	Channel 1 output compare fast enable Refer to CH0COMSEN description
9:8	CH1MS[1:0]	Channel 1 mode selection This bit-field combine with CH1MS[2] specifies the direction of the channel and the input signal selection. The CH1MS[2:0] bit-field is writable only when the channel is not active (When MCH1MSEL[1:0] = 2b'00, the CH1EN bit in TIMERx_CHCTL2 register is reset; when MCH1MSEL[1:0] = 2b'01 or 2b'11, the CH1EN and MCH1EN bits in TIMERx_CHCTL2 register are reset). 000: Channel 1 is programmed as output. 001: Channel 1 is programmed as input, IS1 is connected to CI1FE1. 010: Channel 1 is programmed as input, IS1 is connected to CI0FE1. 011: Channel 1 is programmed as input, IS1 is connected to ITS. This mode is working only if an internal trigger input is selected (through SYSCFG_TIMERxCFG2 register's TSCFG15[4:0]). 100: Channel 1 is programmed as input, IS1 is connected to MCI1FE1. 101~111: Reserved.
7	CH0COMCEN	Channel 0 output compare clear enable When this bit is set, if the ETIFP signal is detected as high level, the O0CPRE signal will be cleared. 0: Channel 0 output compare clear disabled 1: Channel 0 output compare clear enabled
6:4	CH0COMCTL[2:0]	Channel 0 compare output control The CH0COMCTL[3] and CH0COMCTL[2:0] bit-field control the behavior of O0CPRE which drives CH0_O. The active level of O0CPRE is high, while the active level of CH0_O depends on CH0P bit. **Note**: When multi mode channel 0 is configured in output mode, and the MCH0MSEL[1:0] = 2b'11, This bit-field controls the behavior of O0CPRE which drives CH0_O and MCH0_O. The active level of O0CPRE is high, while the active level of CH0_O and MCH0_O depends on CH0P and MCH0P bits. 0000: Timing mode. The O0CPRE signal keeps stable, independent of the comparison between the register TIMERx_CH0CV and the counter TIMERx_CNT. 0001: Set the channel output on match. O0CPRE signal is forced high when the counter matches the output compare register TIMERx_CH0CV. 0010: Clear the channel output on match. O0CPRE signal is forced low when the counter matches the output compare register TIMERx_CH0CV. 0011: Toggle on match. O0CPRE toggles when the counter matches the output compare register TIMERx_CH0CV. 0100: Force low. O0CPRE is forced low level. 0101: Force high. O0CPRE is forced high level. 0110: PWM mode 0. When counting up, O0CPRE is active as long as the counter

is smaller than TIMERx_CH0CV, otherwise it is inactive. When counting down, MO0CPRE is inactive as long as the counter is larger than TIMERx_CH0CV, otherwise it is active.

0111: PWM mode 1. When counting up, O0CPRE is inactive as long as the counter is smaller than TIMERx_CH0CV, otherwise it is active. When counting down, O0CPRE is active as long as the counter is larger than TIMERx_CH0CV, otherwise it is inactive.

1000: Composite PWM mode 0, When counting up, the channel output is forced low level when the counter is matched with the value of TIMERx_CH0CV, the channel output is forced high level when the counter matches the value of TIMERx_CH0CV_ADD. When counting down, the channel output is forced high level when the counter is matched with the value of TIMERx_CH0CV, the channel output is forced low level when the counter matches the value of TIMERx_CH0CV_ADD.

1001: Composite PWM mode 1. When counting up, the channel output is forced high level when the counter is matched with the value of TIMERx_CH0CV, the channel output is forced low level when the counter matches the value of TIMERx_CH0CV_ADD. When counting down, the channel output is forced low level when the counter is matched with the value of TIMERx_CH0CV, the channel output is forced high level when the counter matches the value of TIMERx_CH0CV_ADD.

1010: Asymmetric PWM mode 0. Only for Center aligned mode, when counting up, the channel output is forced low level when the counter is matched with the value of TIMERx_CH0CV, when counting down, the channel output is forced high level when the counter is matched with the value of TIMERx_CH0CV_ADD.

1011: Asymmetric PWM mode 1. Only for Center aligned mode, when counting up, the channel output is forced high level when the counter is matched with the value of TIMERx_CH0CV, when counting down, the channel output is forced low level when the counter is matched with the value of TIMERx_CH0CV_ADD.

1100~1111: Reserved.

Note: In the composite or asymmetric PWM mode (CH0COMCTL[3:2]=2'b10 and CH0MS = 3'b000), the PWM signal output in channel 0 is composited by TIMERx_CH0CV and TIMERx_CH0CV_ADD. Please refer to ***Composite PWM mode*** and ***Asymmetric PWM mode*** for more details.

If configured in PWM mode(PWM mode 0/1,composite PWM mode0/1, asymmetric PWM mode0/1), the O0CPRE level changes only when the output compare mode switches from "Timing" mode to "PWM" mode or the result of the comparison changes.

When the outputs of CH0 and MCH0 are complementary, this bit-field is preloaded. If CCSE =1, this bit-field will only be updated when an channel commutation event is generated.

This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 and CH0MS bit-field is 000 (compare mode).

3	CH0COMSEN	Channel 0 compare output shadow enable When this bit is set, the shadow register of TIMERx_CH0CV register which updates

		at each update event will be enabled. 0: Channel 0 output compare shadow disabled 1: Channel 0 output compare shadow enabled The PWM mode can be used without verifying the shadow register only in single pulse mode (when SPM=1). This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 and CH0MS bit-field is 000.
2	CH0COMFEN	Channel 0 output compare fast enable When this bit is set, the effect of an event on the trigger in input on the capture /compare output will be accelerated if the channel is configured in PWM0 or PWM1 mode. The output channel will treat an active edge on the trigger input as a compare match, and CH0_O is set to the compare level independently from the result of the comparison. 0: Channel 0 output quickly compare disable. The minimum delay from an edge on the trigger input to activate CH0_O output is 5 clock cycles. 1: Channel 0 output quickly compare enable. The minimum delay from an edge on the trigger input to activate CH0_O output is 3 clock cycles.
1:0	CH0MS[1:0]	Channel 0 I/O mode selection The CH0MS[2] and CH0COMCTL[1:0] bit-field specifies the work mode of the channel and the input signal selection. The CH0MS[2:0] bit-field is writable only when the channel is not active (When MCH0MSEL[1:0] = 2b'00, the CH1EN bit in TIMERx_CHCTL2 register is reset; when MCH0MSEL[1:0] = 2b'01 or 2b'11, the CH0EN and MCH0EN bits in TIMERx_CHCTL2 register are reset). 000: Channel 0 is programmed as output. 001: Channel 0 is programmed as input, IS0 is connected to CI0FE0. 010: Channel 0 is programmed as input, IS0 is connected to CI1FE0. 011: Channel 0 is programmed as input, IS0 is connected to ITS, this mode is working only if an internal trigger input is selected (through SYSCFG_TIMERxCFG2 register's TSCFG15[4:0]). 100: Channel 0 is programmed as input, IS0 is connected to MCI0FE0. 101~111: Reserved.

Input capture mode:

Bits	Fields	Descriptions
31	CH1MS[2]	Channel 1 I/O mode selection Same as output compare mode.
30	CH0MS[2]	Channel 0 I/O mode selection Same as output compare mode.
29:16	Reserved	Must be kept at reset value.
15:12	CH1CAPFLT[3:0]	Channel 1 input capture filter control Refer to CH0CAPFLT description.

11:10 CH1CAPPSC[1:0] Channel 1 input capture prescaler
Refer to CH0CAPPSC description.

9:8 CH1MS[1:0] Channel 1 I/O mode selection
Same as output compare mode.

7:4 CH0CAPFLT[3:0] Channel 0 input capture filter control
The CI0 input signal can be filtered by digital filter and this bit-field configure the filtering capability.
Basic principle of digital filter: continuously sample the CI0 input signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit, it is considered to be an effective level.
The filtering capability configuration is as follows:

CH0CAPFLT [3:0]	Times	f_{SAMP}
4'b0000	Filter disabled.	
4'b0001	2	
4'b0010	4	f_{CK_TIMER}
4'b0011	8	
4'b0100	6	$f_{DTS}/2$
4'b0101	8	
4'b0110	6	$f_{DTS}/4$
4'b0111	8	
4'b1000	6	$f_{DTS}/8$
4'b1001	8	
4'b1010	5	
4'b1011	6	$f_{DTS}/16$
4'b1100	8	
4'b1101	5	
4'b1110	6	$f_{DTS}/32$
4'b1111	8	

3:2 CH0CAPPSC[1:0] Channel 0 input capture prescaler
This bit-field specifies the factor of the prescaler on channel 0 input. The prescaler is reset when CH0EN bit in TIMERx_CHCTL2 register is cleared.
00: Prescaler disabled, input capture occurs on every channel input edge.
01: The input capture occurs on every 2 channel input edges
10: The input capture occurs on every 4 channel input edges
11: The input capture occurs on every 8 channel input edges

1:0 CH0MS[1:0] Channel 0 I/O mode selection
Same as output compare mode.

Channel control register 1 (TIMERx_CHCTL1)

Address offset: 0x1C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH3MS [2]	CH2MS [2]	CH3COMADDSEN / Reserved	CH2COMADDSEN / Reserved	CH3COMADDUPSEL	CH2COMADDUPSEL	Reserved	CH3COMCTL[3]	Reserved							CH2COMCTL[3]
rw	rw	rw	rw	rw	rw		rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3COMCEN	CH3COMCTL[2:0]			CH3COMSEN	CH3COMFEN	CH3MS[1:0]		CH2COMCEN	CH2COMCTL[2:0]			CH2COMSEN	CH2COMFEN	CH2MS[1:0]	
CH3CAPFLT[3:0]				CH3CAPPSC[1:0]				CH2CAPFLT[3:0]				CH2CAPPSC[1:0]			
rw				rw		rw		rw				rw		rw	

Output compare mode:

Bits	Fields	Descriptions
31	CH3MS[2]	Channel 3 I/O mode selection Refer to CH3MS[1:0]description.
30	CH2MS[2]	Channel 2 I/O mode selection Refer to CH2MS[1:0] description.
29	CH3COMADDSEN	Channel 3 additional compare output shadow enable Refer to CH2COMADDSEN description.
28	CH2COMADDSEN	Channel 2 additional compare output shadow enable When this bit is set, the shadow register of TIMERx_CH2COMV_ADD register which updates at each update event will be enabled. 0: Channel 2 additional compare shadow disabled 1: Channel 2 additional compare shadow enabled The PWM mode can be used without validating the shadow register only in single pulse mode (SPM bit in TIMERx_CTL0 register is set). This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 and CH2MS bit-field is 000.
27	CH3COMADDUPSEL	Refer to CH0COMADDUPSEL.
26	CH2COMADDUPSEL	Update selection of channel 2 additional compare value 0: Channel 2 additional compare value is update from update event. 1: Channel 2 additional compare value is update from compare event.
25	Reserved	Must be kept at reset value.
24	CH3COMCTL[3]	Refer to CH3COMCTL[2:0]
23:17	Reserved	Must be kept at reset value.
16	CH2COMCTL[3]	Refer to CH2COMCTL[2:0]
15	CH3COMCEN	Channel 3 output compare clear enable Refer to CH2COMCEN description

14:12	CH3COMCTL[2:0]	Channel 3 compare output control Refer to CH0COMCTL description
11	CH3COMSEN	Channel 3 output compare shadow enable Refer to CH2COMSEN description
10	CH3COMFEN	Channel 3 output compare fast enable Refer to CH2COMSEN description
9:8	CH3MS[1:0]	Channel 3 mode selection This bit-field combine with CH3MS[2] specifies the direction of the channel and the input signal selection. The CH3MS[2:0] bit-field is writable only when the channel is not active (When MCH3MSEL[1:0] = 2b'00, the CH3EN bit in TIMERx_CHCTL2 register is reset; when MCH3MSEL[1:0] = 2b'11, the CH3EN and MCH3EN bits in TIMERx_CHCTL2 register are reset). 000: Channel 3 is configured as output. 001: Channel 3 is configured as input, IS3 is connected to CI3FE3. 010: Channel 3 is configured as input, IS3 is connected to CI2FE3. 011: Channel 3 is configured as input, IS3 is connected to ITS, this mode is working only if an internal trigger input is selected (through TSCFG15[4:0] bit-field in SYSCFG_TIMERxCFG2(x=0,7) register). 100: Channel 3 is configured as input, IS3 is connected to MCI3FE3. 101~111: Reserved.
7	CH2COMCEN	Channel 2 output compare clear enable. When this bit is set, if the ETIFP signal is detected as high level, the O2CPRE signal will be cleared. 0: Channel 2 output compare clear disabled 1: Channel 2 output compare clear enabled
6:4	CH2COMCTL[2:0]	Channel 1 output compare shadow enable Refer to CH0COMSEN description
3	CH2COMSEN	Channel 2 output compare shadow enable Refer to CH0COMSEN description
2	CH2COMFEN	Channel 2 output compare fast enable When this bit is set, the effect of an event on the trigger in input on the capture/ compare output will be accelerated if the channel is configured in PWM0 or PWM1 mode. The output channel will treat an active edge on the trigger input as a compare match, and CH2_O is set to the compare level independently from the result of the comparison. 0: Channel 2 output quickly compare disable. The minimum delay from an edge on the trigger input to activate CH2_O output is 5 clock cycles. 1: Channel 2 output quickly compare enable. The minimum delay from an edge on the trigger input to activate CH2_O output is 3 clock cycles.
1:0	CH2MS[1:0]	Channel 2 I/O mode selection

This bit-field combine with CH2MS[2] specifies the work mode of the channel and the input signal selection. The CH2MS[2:0] bit-field is writable only when the channel is not active(When MCH2MSEL[1:0] = 2b'00, the CH2EN bit in TIMERx_CHCTL2 register is reset; when MCH2MSEL[1:0] = 2b'01 or 2b'11, the CH2EN and MCH2EN bits in TIMERx_CHCTL2 register are reset).
000: Channel 2 is programmed as output.
001: Channel 2 is programmed as input, IS2 is connected to CI2FE2.
010: Channel 2 is programmed as input, IS2 is connected to CI3FE2.
011: Channel 2 is programmed as input, IS2 is connected to ITS. This mode is working only if an internal trigger input is selected (through TSCFG15 bits in SYSCFG_TIMERxCFG2 register).
100: Channel 2 is programmed as input, IS2 is connected to MCI2FE2.
101~111: Reserved.

Input capture mode:

Bits	Fields	Descriptions
31	CH3MS[2]	Channel 3 I/O mode selection Same as output compare mode.
30	CH2MS[2]	Channel 2 I/O mode selection Same as output compare mode.
31:16	Reserved	Must be kept at reset value.
15:12	CH3CAPFLT[3:0]	Channel 3 input capture filter control Refer to CH2CAPFLT description.
11:10	CH3CAPPSC[1:0]	Channel 3 input capture prescaler Refer to CH2CAPPSC description.
9:8	CH3MS[1:0]	Channel 3 mode selection Same as output compare mode.
7:4	CH2CAPFLT[3:0]	Channel 2 input capture filter control The CI2 input signal can be filtered by digital filter and this bit-field configure the filtering capability. Basic principle of digital filter: continuously sample the CI2 input signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit, it is considered to be an effective level. The filtering capability configuration is as follows:

CH2CAPFLT [3:0]	Times	f_{SAMP}
4'b0000	Filter disabled.	
4'b0001	2	f_{CK_TIMER}
4'b0010	4	
4'b0011	8	
4'b0100	6	$f_{DTS}/2$
4'b0101	8	

4'b0110	6	$f_{DTS}/4$
4'b0111	8	
4'b1000	6	$f_{DTS}/8$
4'b1001	8	
4'b1010	5	
4'b1011	6	$f_{DTS}/16$
4'b1100	8	
4'b1101	5	
4'b1110	6	$f_{DTS}/32$
4'b1111	8	

3:2 CH2CAPPSC[1:0] Channel 2 input capture prescaler

This bit-field specifies the factor of the prescaler on channel 2 input. The prescaler is reset when CH2EN bit in TIMERx_CHCTL2 register is cleared.

00: Prescaler disabled, capture is done on each channel input edge.

01: The input capture occurs on every 2 channel input edges

10: The input capture occurs on every 4 channel input edges

11: The input capture occurs on every 8 channel input edges

1:0 CH2MS[1:0] Channel 2 mode selection

Same as output compare mode.

Channel control register 2 (TIMERx_CHCTL2)

Address offset: 0x20

Reset value: 0x0000 0000

This register can be accessed by half-word(16-bit) or word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH3PERFOREN	CH2PERFOREN	CH1PERFOREN	CH0PERFOREN	MCH3PERFOREN	MCH2PERFOREN	MCH1PERFOREN	MCH0PERFOREN	Reserved							
rw	rw	rw	rw	rw	rw	rw	rw								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCH3P	MCH3EN	CH3P	CH3EN	MCH2P	MCH2EN	CH2P	CH2EN	MCH1P	MCH1EN	CH1P	CH1EN	MCH0P	MCH0EN	CH0P	CH0EN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31	CH3PERFOREN	Channel 3 in the composite PWM or asymmetric PWM mode, the period point match moment O3CPRE level setting Refer to CH0PERFOREN description.
30	CH2PERFOREN	Channel 2 in the composite PWM or asymmetric PWM mode, the period point match moment O2CPRE level setting Refer to CH0PERFOREN description.
29	CH1PERFOREN	Channel 1 in the composite PWM or asymmetric PWM mode, the period point match moment O1CPRE level setting

		Refer to CH0PERFOREN description.
28	CH0PERFOREN	Channel 0 in the composite PWM or asymmetric PWM mode, the period point match moment O0CPRE level setting 0: disable 1: In the composite PWM0 or asymmetric PWM0 mode, the level will be forced to high at the time of period point matching; In the composite PWM1 or asymmetric PWM1 mode, the level at the matching time of the period point is forced to low. **Note:** In the central alignment mode, the period point is the counter underflow time; In the edge-aligned mode, the period point is the counter flow moment.
27	MCH3PERFOREN	Multi mode channel 3 in the composite PWM or asymmetric PWM mode, the period point match moment MO3CPRE level setting Refer to MCH0PERFOREN description.
26	MCH2PERFOREN	Multi mode channel 2 in the composite PWM or asymmetric PWM mode, the period point match moment MO2CPRE level setting Refer to MCH0PERFOREN description.
25	MCH1PERFOREN	Multi mode channel 1 in the composite PWM or asymmetric PWM mode, the period point match moment MO1CPRE level setting Refer to MCH0PERFOREN description.
24	MCH0PERFOREN	Multi mode channel 0 in the composite PWM or asymmetric PWM mode, the period point match moment MO0CPRE level setting 0: disable 1: In the composite PWM0 or asymmetric PWM0 mode, the level will be forced to high at the time of period point matching; In the composite PWM1 or asymmetric PWM1 mode, the level at the matching time of the period point is forced to low. **Note:** In the central alignment mode, the period point is the counter underflow time; In the edge-aligned mode, the period point is the counter flow moment.
23:16	Reserved	Must be kept at reset value.
15	MCH3P	Multi mode channel 3 capture/compare polarity Refer to MCH0P description.
14	MCH3EN	Multi mode channel 3 capture/compare enable Refer to MCH0EN description.
13	CH3P	Channel 3 capture/compare polarity Refer to CH0P description.
12	CH3EN	Channel 3 capture/compare enable Refer to CH0EN description.
11	MCH2P	Multi mode channel 2 output polarity Refer to MCH0P description.
10	MCH2EN	Multi mode channel 2 output enable

		Refer to MCH0EN description.
9	CH2P	Channel 2 capture/compare polarity Refer to CH0P description.
8	CH2EN	Channel 2 capture/compare enable Refer to CH0EN description.
7	MCH1P	Multi mode channel 1 output polarity Refer to MCH0P description.
6	MCH1EN	Multi mode channel 1 output enable Refer to MCH0EN description.
5	CH1P	Channel 1 capture/compare polarity Refer to CH0P description.
4	CH1EN	Channel 1 capture/compare enable Refer to CH0EN description.
3	MCH0P	Multi mode channel 0 output polarity When Multi mode channel 0 is configured in output mode, and the MCH0MSEL[1:0] = 2b'11, this bit specifies the MCH0_O output signal polarity. 0: Multi mode channel 0 output active high 1: Multi mode channel 0 output active low When CH0 is configured in input mode, in conjunction with CH0P, this bit is used to define the polarity of CH0. This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 or 10.
2	MCH0EN	Multi mode channel 0 capture/compare enable When multi mode channel 0 is configured in output mode, setting this bit enables MCH0_O signal in active state. When multi mode channel 0 is configured in input mode, setting this bit enables the capture event in multi mode channel 0. 0: Multi mode channel 0 disabled 1: Multi mode channel 0 enabled
1	CH0P	Channel 0 capture/compare polarity When channel 0 is configured in output mode, this bit specifies the output signal polarity. 0: Channel 0 active high 1: Channel 0 active low When channel 0 is configured in input mode, these bits specify the channel 0 input signal's polarity. [MCH0P, CH0P] will select the active trigger or capture polarity for channel 0 input signals. 00: channel 0 input signal's rising edge is the active signal for capture or trigger operation in slave mode. And channel 0 input signal will not be inverted. 01: channel 0 input signal's falling edge is the active signal for capture or trigger operation in slave mode. And channel 0 input signal will be inverted.

		10: Reserved. 11: Noninverting/both channel 0 input signal's edges. This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 or 10.
0	CH0EN	Channel 0 capture/compare enable When channel 0 is configured in output mode, setting this bit enables CH0_O signal in active state. When channel 0 is configured in input mode, setting this bit enables the capture event in channel 0. 0: Channel 0 disabled 1: Channel 0 enabled

Counter register (TIMERx_CNT)

Address offset: 0x24
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CNT[15:0]	This bit-field indicates the current counter value. Writing to this bit-field can change the value of the counter. Note: When a CNT is greater than the CAR, the CNT will be counted to 0xFFFF before clearing.

Prescaler register (TIMERx_PSC)

Address offset: 0x28
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	PSC[15:0]	Prescaler value of the counter clock The TIMER_CK clock is divided by (PSC+1) to generate the counter clock. The value of this bit-field will be loaded to the corresponding shadow register at every update event.

Counter auto reload register (TIMERx_CAR)

Address offset: 0x2C
Reset value: 0x0000 FFFF

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CARL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CARL[15:0]	Counter auto reload value This bit-field specifies the auto reload value of the counter. **Note:** When the timer is configured in input capture mode, this register must be configured a non-zero value (such as 0xFFFF) which is larger than user expected value. When the timer is configured in bit output mode, this register should avoid setting the CAR to 0.

Counter repetition register (TIMERx_CREP)

Address offset: 0x30
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CREP[7:0]							
								rw							

Bits	Fields	Descriptions

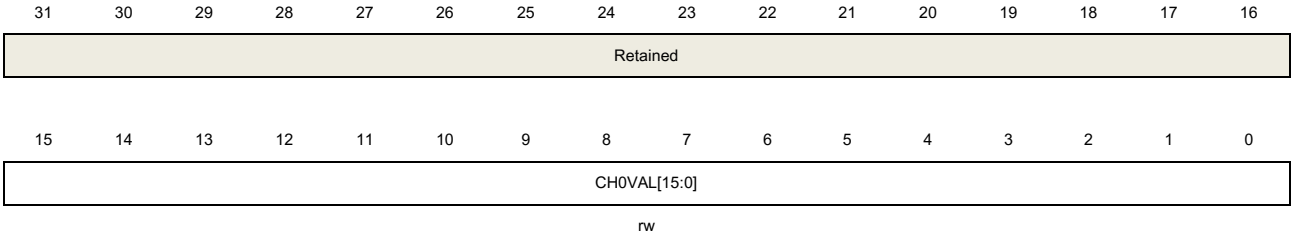
31:8	Reserved	Must be kept at reset value.
7:0	CREP[7:0]	Counter repetition value This bit-field specifies the update event generation rate. Each time the repetition counter counts down to zero, an update event will be generated. The update rate of the shadow registers is also affected by this bit-field when these shadow registers are enabled.

Channel 0 capture/compare value register (TIMERx_CH0CV)

Address offset: 0x34

Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).



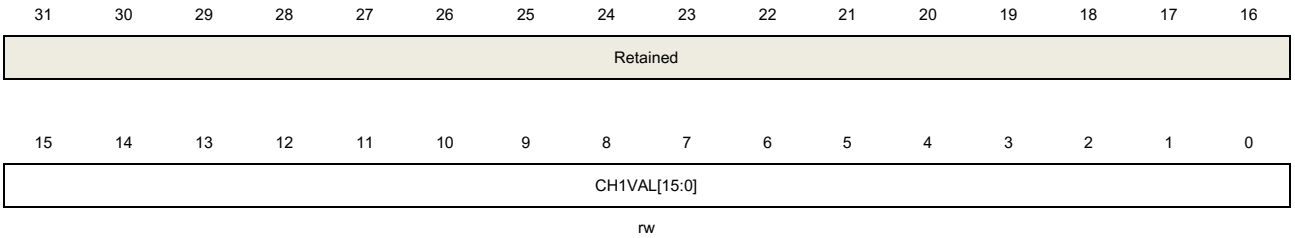
Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.
15:0	CH0VAL[15:0]	Capture or compare values for channel 0 When channel 0 is configured for input mode, these bits determine the counter value for the last capture event. And this register is read-only. When channel 0 is configured in output mode, these bits contain the values that are about to be compared to the counter. When the corresponding shadow register is enabled, the shadow register value is updated with each update event.

Channel 1 capture/compare value register (TIMERx_CH1CV)

Address offset: 0x38

Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).



Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.

15:0	CH1VAL[15:0]	Capture or compare values for channel 1 When channel 1 is configured for input mode, these bits determine the counter value for the last captured event. And this register is read-only. When channel 1 is configured for output mode, these bits contain the values that are about to be compared to the counter. When the corresponding shadow register is enabled, the shadow register value is updated with each update event.

Channel 2 capture/compare value register (TIMERx_CH2CV)

Address offset: 0x3C

Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Retained															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH2VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.
15:0	CH2VAL[15:0]	Capture or compare values for channel 2 When channel 2 is configured in input mode, these bits determine the counter value for the last captured event. And this register is read-only. When channel 2 is configured for output mode, these bits contain the values that are about to be compared with the counter. When the corresponding shadow register is enabled, the shadow register value is updated with each update event.

Channel 3 capture/compare value register (TIMERx_CH3CV)

Address offset: 0x40

Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Retained															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.

15:0	CH3VAL[15:0]	Capture or compare values for channel 3 When channel 3 is configured for input mode, these bits determine the counter value for the last captured event. And this register is read-only. When channel 3 is configured in output mode, these bits contain the values that are about to be compared to the counter. When the corresponding shadow register is enabled, the shadow register value is updated with each update event.

Complementary channel protection register (TIMERx_CCHP)

Address offset: 0x44

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															DTSEN
															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POEN	OAEN	BRKP	BRKEN	ROS	IOS	PROT[1:0]		DTCFG[7:0]							
rw	rw	rw	rw	rw	rw	rw		rw							

Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value.
16	DTSEN	Deadtime shadow is enabled. When the bit is set, dynamic change of the deadtime is supported. The deadtime takes effect when the update event arrives. 0: Deadtime shadow register is disabled. 1: Deadtime shadow register is enabled. This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.
15	POEN	Primary output enable The bit can be set to 1 by: - Write 1 to this bit - If OAEN is set to 1, this bit is set to 1 at the next update event. The bit can be cleared to 0 by: - Write 0 to this bit - Valid fault input. When one of channels is configured in output mode, setting this bit enables the channel outputs (CHx_O and MCHx_O) if the corresponding enable bits (CHxEN, MCHxEN in TIMERx_CHCTL2 register) have been set. 0: Disable channel outputs (CHxO or CHxON). 1: Enabled channel outputs (CHxO or CHxON). **Note:** This bit is only valid when CHxMS=2'b00. When the BRKENCHx in the TIMERx_CTL2 is 0, this bit does not work for the corresponding channel.

14	OAEN	Output automatic enable 0: POEN bit can only be set by software. 1: POEN can be set at the next update event, if the break input is not active. This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.
13	BRKP	BRKINx(x = 0..3) input signals polarity This bit specifies the polarity of the BRKINx(x = 0..3) input signals. 0: BRKINx(x = 0..3) input active low 1: BRKINx(x = 0..3) input active high This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.
12	BRKEN	BREAK signals and counter freeze configuration enable This bit can be set to enable the BRKINx(x = 0..3) and counters freeze configurations. 0: Break inputs and freeze counter configuration disabled 1: Break inputs and freeze counter configuration enabled This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.
11	ROS	Run mode "off-state" enable When POEN bit is set (Run mode), this bit can be set to enable the "off-state" for the channels which has been configured in output mode. Please refer to ***Table 17-5. Complementary outputs controlled by parameters (MCHxMSEL =2'b11)***. 0: "off-state" disabled. If the CHxEN or MCHxEN bit is reset, the corresponding channel is output disabled. 1: "off-state" enabled. If the CHxEN or MCHxEN bit is reset, the corresponding channel is "off-state". This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 10 or 11.
10	IOS	Idle mode "off-state" enable When POEN bit is reset (Idle mode), this bit can be set to enable the "off-state" for the channels which has been configured in output mode. Please refer to ***Table 17-5. Complementary outputs controlled by parameters (MCHxMSEL =2'b11)***. 0: "off-state" disabled. If the CHxEN/CHxNEN bits are both reset, the channels are output disabled. 1: "off-state" enabled. No matter the CHxEN/MCHxEN bits, the channels are "off-state". This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 10 or 11.
9:8	PROT[1:0]	Complementary register protect control This bit-field specifies the write protection property of registers. 00: Protect disabled. No write protection. 01: PROT mode 0. The ISOx/ISOxN bits in TIMERx_CTL1 register, the

DTSEN/BRKEN/BRKP/OAEN/DTCFG bits in TIMERx_CCHP register, the BRKxP/BRKxEN(x = 0..3) bits in TIMERx_BRKCFG register and the FCCHPxEN/DTxSEN/DTCFG bits in TIMERx_FCCHPx (x = 0..3) register, are writing protected.

10: PROT mode 1. In addition to the registers in PROT mode 0, the CHxP/MCHxP bits in TIMERx_CHCTL2 register (if related channel is configured in output mode) , the ROS/IOS bits in TIMERx_CCHP register and the ROS/IOS bits in TIMERx_FCCHPx (x = 0..3) register are writing protected.

11: PROT mode 2. In addition to the registers in PROT mode 1, the CHxCOMCTL/ CHxCOMSEN/ CHxCOMADDSEN/ MCHxCOMCTL/ MCHxCOMSEN bits in TIMERx_CHCTL0/1 and TIMERx_MCHCTL0/1 registers (if the related channel is configured in output) are writing protected.

This bit-field can be written only once after the system reset. Once the TIMERx_CCHP register has been written, this bit-field will be writing protected.

7:0 DTCFG[7:0] Dead time configuration

The relationship between DTVAL value and the duration of dead-time is as follow:

DTCFG[7:5]	The duration of dead-time
3'b0xx	DTCFG[7:0] * t_{DTS_CK}
3'b10x	(64+ DTCFG[5:0]) * t_{DTS_CK} *2
3'b110	(32+ DTCFG[4:0]) * t_{DTS_CK} *8
3'b111	(32+ DTCFG[4:0]) * t_{DTS_CK} *16

Note:

1. t_{DTS_CK} is the period of DTS_CK which is configured by CKDIV[1:0] in TIMERx_CTL0.
2. This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.

Multi mode channel control register 0 (TIMERx_MCHCTL0)

Address offset: 0x48

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MCH1MS[2]	MCH0MS[2]	MCH1COMADDSEN	MCH0COMADDSEN	MCH1COMADDUPSEL	MCH0COMADDUPSEL	Reserved	MCH1COMCTL[3]	Reserved							MCH0COMCTL[3]
rw	rw	rw	rw	rw	rw		rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCH1COMCEN	MCH1COMCTL[2:0]			MCH1COMSEN	MCH1COMFEN	MCH1MS[1:0]		MCH0COMCEN	MCH0COMCTL[2:0]			MCH0COMSEN	MCH0COMFEN	MCH0MS[1:0]	
MCH1CAPFLT[3:0]				MCH1CAPPSC[1:0]				MCH0CAPFLT[3:0]				MCH0CAPPSC[1:0]			
rw				rw		rw		rw				rw		rw	

Output compare mode:

Bits	Fields	Descriptions
31	MCH1MS[2]	Multi mode channel 1 I/O mode selection Refer to MCH1MS[1:0]description.
30	MCH0MS[2]	Multi mode channel 0 I/O mode selection Refer to MCH0MS[1:0] description.
29	MCH1COMADDSEN	Multi mode channel 1 output additional compare shadow enable Refer to MCH0COMADDSEN description.
28	MCH0COMADDSEN	Multi mode channel 0 output additional compare shadow enable When this bit is set, the shadow register of TIMERx_MCH0CV_ADD register which updates at each update event will be enabled. 0: Multi mode channel 0 output additional compare shadow disabled 1: Multi mode channel 0 output additional compare shadow enabled The PWM mode can be used without verifying the shadow register only in single pulse mode (when SPM=1). This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 and MCH0MS bit-field is 00.
27	MCH1COMADDUPSEL	Update selection of multi mode channel 1 additional compare value Refer to MCH0COMADDUPSEL.
26	MCH0COMADDUPSEL	Update selection of multi mode channel 0 additional compare value 0: multi mode channel 0 additional compare value is update from update event. 1: multi mode channel 0 additional compare value is update from compare event.
25	Reserved	Must be kept at reset value.
24	MCH1COMCTL[3]	Refer to MCH1COMCTL[2:0]
23:17	Reserved	Must be kept at reset value.
16	MCH0COMCTL[3]	Refer to MCH0COMCTL[2:0]
15	MCH1COMCEN	Multi mode channel 1 output compare clear enable. Refer to MCH0COMCEN description.
14:12	MCH1COMCTL[2:0]	Multi mode channel 1 compare output control. Refer to MCH0COMCTL description.
11	MCH1COMSEN	Multi mode channel 1 output compare shadow enable Refer to MCH0COMSEN description.
10	MCH1COMFEN	Multi mode channel 1 output compare fast enable Refer to MCH0COMFEN description.
9:8	MCH1MS[1:0]	Multi mode channel 1 I/O mode selection This bit-field specifies the direction of the channel and the input signal selection. This bit-field is writable only when the channel is not active (the MCH1EN bit in TIMERx_CHCTL2 register is reset)

		000: Multi mode channel 1 is programmed as output. 001: Multi mode channel 1 is programmed as input, MIS1 is connected to MCI1FEM1. 010: Multi mode channel 1 is programmed as input, MIS1 is connected to MCI0FEM1. 011: Multi mode channel 1 is programmed as input, MIS1 is connected to ITS. This mode is working only if an internal trigger input is selected (through TSCFG15 bits in SYSCFG_TIMERxCFG2 register). 100: Multi mode channel 1 is programmed as input, MIS1 is connected to CI1FEM1. 101~111: Reserved.
7	MCH0COMCEN	Multi mode channel 0 output compare clear enable. When this bit is set, if the ETIFP signal is detected as high level, the MO0CPRE signal will be cleared. 0: Multi mode channel 0 output compare clear disabled. 1: Multi mode channel 0 output compare clear enabled.
6:4	MCH0COMCTL [2:0]	Multi mode channel 0 compare output control **Note**: The MCH0COMCTL[3] and MCH0COMCTL[2:0] bit-field control the behavior of MO0CPRE which drives MCH0_O. The active level of O0CPRE is high, while the active level of CH0_O and MCH0_O depends on CH0P and MCH0P bits. 0000: Timing mode. The MO0CPRE signal keeps stable, independent of the comparison between the register TIMERx_MCH0CV and the counter TIMERx_CNT. 0001: Set the channel output on match. MO0CPRE signal is forced high when the counter is equals to the output compare register TIMERx_MCH0CV. 0010: Clear the channel output on match. MO0CPRE signal is forced low when the counter is equals to the output compare register TIMERx_MCH0CV. 0011: Toggle on match. MO0CPRE toggles when the counter is equals to the output compare register TIMERx_MCH0CV. 0100: Force low. MO0CPRE is forced to low level. 0101: Force high. MO0CPRE is forced to high level. 0110: PWM mode 0. When counting up, MO0CPRE is active as long as the counter is smaller than TIMERx_MCH0CV, otherwise it is inactive. When counting down, MO0CPRE is inactive as long as the counter is larger than TIMERx_MCH0CV, otherwise it is active. 0111: PWM mode 1. When counting up, MO0CPRE is inactive as long as the counter is smaller than TIMERx_MCH0CV, otherwise it is active. When counting down, MO0CPRE is active as long as the counter is larger than TIMERx_MCH0CV, otherwise it is inactive. 1000: Composite PWM mode 0, When counting up, the channel output is forced low level when the counter is matched with the value of TIMERx_MCH0CV, the channel output is forced high level when the counter matches the value of TIMERx_MCH0CV_ADD. When counting down, the channel output is forced high level when the counter is matched with the value of TIMERx_MCH0CV, the channel output is forced low level when the counter matches the value of

TIMERx_MCH0CV_ADD.

1001: Composite PWM mode 1. When counting up, the channel output is forced high level when the counter is matched with the value of TIMERx_MCH0CV, the channel output is forced low level when the counter matches the value of TIMERx_MCH0CV_ADD. When counting down, the channel output is forced low level when the counter is matched with the value of TIMERx_MCH0CV, the channel output is forced high level when the counter matches the value of TIMERx_MCH0CV_ADD.

1010: Asymmetric PWM mode 0. Only for Center aligned mode, when counting up, the channel output is forced low level when the counter is matched with the value of TIMERx_MCH0CV, when counting down, the channel output is forced high level when the counter is matched with the value of TIMERx_MCH0CV_ADD.

1011: Asymmetric PWM mode 1. Only for Center aligned mode, when counting up, the channel output is forced high level when the counter is matched with the value of TIMERx_MCH0CV, when counting down, the channel output is forced low level when the counter is matched with the value of TIMERx_MCH0CV_ADD.

1100~1111: Reserved.

Note: In the composite or asymmetric PWM mode, the PWM signal output in channel 0 is composited by TIMERx_MCH0CV and TIMERx_MCH0CV_ADD. Please refer to ***Composite PWM mode*** and ***Asymmetric PWM mode*** for more details.

If configured in PWM mode, the MO0CPRE level changes only when the output compare mode is adjusted from "Timing" mode to "PWM" mode or the comparison result changes.

This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 and MCH0MS bit-field is 00(compare mode).

3 MCH0COMSEN Multi mode channel 0 output compare shadow enable

When this bit is set, the shadow register of TIMERx_MCH0CV register which updates at each update event will be enabled.

0: Multi mode channel 0 output compare shadow disabled

1: Multi mode channel 0 output compare shadow enabled

The PWM mode can be used without verifying the shadow register only in single pulse mode (when SPM=1).

This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 and MCH0MS bit-field is 00.

2 MCH0COMFEN Multi mode Channel 0 output compare fast enable

When this bit is set, the effect of an event on the trigger in input on the capture/compare output will be accelerated if the channel is configured in PWM0 or PWM1 mode. The output channel will treat an active edge on the trigger input as a compare match, and MCH0_O is set to the compare level independently from the result of the comparison.

0: Multi mode Channel 0 output quickly compare disable. The minimum delay from an edge on the trigger input to activate MCH0_O output is 5 clock cycles.

1: Multi mode Channel 0 output quickly compare enable. The minimum delay from an edge on the trigger input to activate MCH0_O output is 3 clock cycles.

1:0	MCH0MS[1:0]	Multi mode channel 0 I/O mode selection The MCH0MS[2] and MCH0MS[1:0] specifies the work mode of the channel and the input signal selection. This bit-field is writable only when the channel is not active (MCH0EN bit in TIMERx_CHCTL2 register is reset). 000: Multi mode channel 0 is programmed as output. 001: Multi mode channel 0 is programmed as input, MIS0 is connected to MCI0FEM0. 010: Multi mode channel 0 is programmed as input, MIS0 is connected to MCI1FEM0. 011: Multi mode channel 0 is programmed as input, MIS0 is connected to ITS, this mode is working only if an internal trigger input is selected (through TSCFG15 bits in SYSCFG_TIMERxCFG2 register). 100: Multi mode channel 0 is programmed as input, MIS0 is connected to CI0FEM0. 101~111: Reserved.

Input capture mode:

Bits	Fields	Descriptions
31	MCH1MS[2]	Multi mode channel 1 I/O mode selection Refer to MCH1MS[1:0]description.
30	MCH0MS[2]	Multi mode channel 0 I/O mode selection Refer to MCH0MS[1:0] description.
29:16	Reserved	Must be kept at reset value.
15:12	MCH1CAPFLT[3:0]	Multi mode channel 1 input capture filter control. Refer to MCH0CAPFLT description.
11:10	MCH1CAPPSC[1:0]	Multi mode channel 1 input capture prescaler. Refer to MCH0CAPPSC description.
9:8	MCH1MS[1:0]	Multi mode channel 1 I/O mode selection. Same as output compare mode.
7:4	MCH0CAPFLT[3:0]	Multi mode channel 0 input capture filter control. The MCI0 input signal can be filtered by digital filter and this bit-field configure the filtering capability. Basic principle of digital filter: continuously sample the MCI0 input signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit, it is considered to be an effective level. The filtering capability configuration is as follows:

MCH0CAPFLT[3:0]	Times	f_{SAMP}
4'b0000	Filter disabled.	
4'b0001	2	f_{TIMER_CK}

4'b0010	4	
4'b0011	8	
4'b0100	6	$f_{DTS}/2$
4'b0101	8	
4'b0110	6	$f_{DTS}/4$
4'b0111	8	
4'b1000	6	$f_{DTS}/8$
4'b1001	8	
4'b1010	5	
4'b1011	6	$f_{DTS}/16$
4'b1100	8	
4'b1101	5	
4'b1110	6	$f_{DTS}/32$
4'b1111	8	

3:2	MCH0CAPPSC[1:0]	Multi mode channel 0 input capture prescaler This bit-field specifies the factor of the prescaler on channel 0 input. The prescaler is reset when MCH0EN bit in TIMERx_CHCTL2 register is cleared. 00: Prescaler disable, capture occurs on every active edge of the input signal 01: The capture input prescaler factor is 2. 10: The capture input prescaler factor is 4. 11: The capture input prescaler factor is 8.
1:0	MCH0MS[1:0]	Multi mode channel 0 I/O mode selection Same as output compare mode

Multi mode channel control register 1 (TIMERx_MCHCTL1)

Address offset: 0x4C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MCH3MS[2]	MCH2MS[2]	MCH3COMADDSEN	MCH2COMADDSEN	MCH3COMADDUPSEL	MCH2COMADDUPSEL	Reserved	MCH3COMCTL[3]	Reserved							MCH2COMCTL[3]
rw	rw	rw	rw	rw	rw		rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCH3COMCEN	MCH3COMCTL[2:0]			MCH3COMSEN	MCH3COMFEN	MCH3MS[1:0]		MCH2COMCEN	MCH2COMCTL[2:0]			MCH2COMSEN	MCH2COMFEN	MCH2MS[1:0]	
MCH3CAPFLT[3:0]				MCH3CAPPSC[1:0]				MCH2CAPFLT[3:0]				MCH2CAPPSC[1:0]			
rw				rw		rw		rw				rw		rw	

Output compare mode:

Bits	Fields	Descriptions
31	MCH3MS[2]	Multi mode channel 1 I/O mode selection Refer to MCH3MS[1:0]description.

30	MCH2MS[2]	Multi mode channel 0 I/O mode selection Refer to MCH2MS[1:0] description.
29	MCH3COMADDSEN	Multi mode channel 3 output additional compare shadow enable Refer to MCH0COMADDSEN description.
28	MCH2COMADDSEN	Multi mode channel 2 output additional compare shadow enable Refer to MCH0COMADDSEN description.
27	MCH3COMADDUPSEL	Update selection of multi mode channel 3 additional compare value Refer to MCH0COMADDUPSEL.
26	MCH2COMADDUPSEL	Update selection of multi mode channel 2 additional compare value Refer to MCH0COMADDUPSEL.
25	Reserved	Must be kept at reset value.
24	MCH3COMCTL[3]	Refer to MCH0COMCTL[2:0]
23:17	Reserved	Must be kept at reset value.
16	MCH2COMCTL[3]	Refer to MCH0COMCTL[2:0]
15	MCH3COMCEN	Multi mode channel 3 output compare clear enable Refer to MCH2COMCEN description
14:12	MCH3COMCTL[2:0]	Multi mode channel 3 compare output control Refer to MCH2COMCTL description
11	MCH3COMSEN	Multi mode channel 3 output compare shadow enable Refer to MCH2COMSEN description
10	MCH3COMFEN	Multi mode Channel 3 output compare fast enable Refer to MCH0COMFEN description.
9:8	MCH3MS[1:0]	Multi mode channel 3 I/O mode selection This bit-field specifies the direction of the channel and the input signal selection. This bit-field is writable only when the channel is not active (MCH3EN bit in TIMERx_CHCTL2 register is reset). 000: Multi mode channel 3 is programmed as output. 01: Multi mode channel 3 is programmed as input, MIS3 is connected to MCI3FEM3. 010: Multi mode channel 3 is programmed as input, MIS3 is connected to MCI2FEM3. 011: Multi mode channel 3 is programmed as input, MIS3 is connected to ITS, this mode is working only if an internal trigger input is selected (through TSCFG15 bits in SYSCFG_TIMERxCFG2 register). 100: Multi mode channel 3 is programmed as input, MIS3 is connected to CI3FEM3. 101~111: Reserved.
7	MCH2COMCEN	Multi mode channel 2 output compare clear enable. When this bit is set, if the ETIFP signal is detected as high level, the MO2CPRE signal will be cleared.

0: Multi mode channel 2 output compare clear disabled
1: Multi mode channel 2 output compare clear enabled

6:4	MCH2COMCTL[2:0]	Multi mode channel 2 compare output control Refer to MCH0COMCTL description.
3	MCH2COMSEN	Multi mode channel 2 output compare shadow enable When this bit is set, the shadow register of TIMERx_MCH2CV register, which updates at each update event will be enabled. 0: Multi mode channel 2 output compare shadow disabled 1: Multi mode channel 2 output compare shadow enabled The PWM mode can be used without verifying the shadow register only in single pulse mode (when SPM=1). This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 and CH2NMS bit-field is 00.
2	MCH2COMFEN	Multi mode Channel 2 output compare fast enable Refer to MCH0COMFEN description.
1:0	MCH2MS[1:0]	Multi mode channel 2 I/O mode selection This bit-field specifies the work mode of the channel and the input signal selection. This bit-field is writable only when the channel is not active (MCH2EN bit in TIMERx_CHCTL2 register is reset). 000: Multi mode channel 2 is programmed as output. 001: Multi mode channel 2 is programmed as input, MIS2 is connected to MCI2FEM2. 010: Multi mode channel 2 is programmed as input, MIS2 is connected to MCI3FEM2. 011: Multi mode channel 2 is programmed as input, MIS2 is connected to ITS. This mode is working only if an internal trigger input is selected (through TSCFG15 bits in SYSCFG_TIMERxCFG2 register). 100: Multi mode channel 2 is programmed as input, MIS2 is connected to CI2FEM2. 101~111: Reserved.

Input capture mode:

Bits	Fields	Descriptions
31	MCH3MS[2]	Multi mode channel 1 I/O mode selection Refer to MCH3MS[1:0]description.
30	MCH2MS[2]	Multi mode channel 0 I/O mode selection Refer to MCH2MS[1:0] description.
29:16	Reserved	Must be kept at reset value.
15:12	MCH3CAPFLT[3:0]	Multi mode channel 3 input capture filter control. Refer to MCH2CAPFLT description.
11:10	MCH3CAPPSC[1:0]	Multi mode channel 3 input capture prescaler.

Refer to MCH2CAPPSC description.

9:8 MCH3MS[1:0] Multi mode channel 3 I/O mode selection.
Same as output compare mode.

7:4 MCH2CAPFLT[3:0] Multi mode channel 2 input capture filter control.
The MCI2 input signal can be filtered by digital filter and this bit-field configure the filtering capability.
Basic principle of digital filter: continuously sample the MCI2 input signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit, it is considered to be an effective level.
The filtering capability configuration is as follows:

MCH2CAPFLT**[3:0]**	**Times**	**f_{SAMP}**
4'b0000	Filter disabled.	
4'b0001	2	
4'b0010	4	f_{TIMER_CK}
4'b0011	8	
4'b0100	6	$f_{DTS}/2$
4'b0101	8	
4'b0110	6	$f_{DTS}/4$
4'b0111	8	
4'b1000	6	$f_{DTS}/8$
4'b1001	8	
4'b1010	5	
4'b1011	6	$f_{DTS}/16$
4'b1100	8	
4'b1101	5	
4'b1110	6	$f_{DTS}/32$
4'b1111	8	

3:2 MCH2CAPPSC[1:0] Multi mode channel 2 input capture prescaler.
This bit-field specifies the factor of the prescaler on channel 2 input. The prescaler is reset when MCH2EN bit in TIMERx_CHCTL2 register is cleared.
00: Prescaler disable, capture occurs on every active edge of the input signal
01: The capture input prescaler factor is 2.
10: The capture input prescaler factor is 4.
11: The capture input prescaler factor is 8.

1:0 MCH2MS[1:0] Multi mode channel 2 I/O mode selection
Same as output compare mode.

Multi mode channel control register 2 (TIMERx_MCHCTL2)

Address offset: 0x50
Reset value: 0x0000 0000

This register can be accessed by half-word(16-bit) or word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								MCH3FP[1:0]		MCH2FP[1:0]		MCH1FP[1:0]		MCH0FP[1:0]	
								rw		rw		rw		rw	

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:6	MCH3FP[1:0]	Multi mode channel 3 capture/compare free polarity Refer to MCH0FP[1:0] description.
5:4	MCH2FP[1:0]	Multi mode channel 2 capture/compare free polarity Refer to MCH0FP[1:0] description.
3:2	MCH1FP[1:0]	Multi mode channel 1 capture/compare free polarity Refer to MCH0FP[1:0] description.
1:0	MCH0FP[1:0]	Multi mode channel 0 capture/compare free polarity When multi mode channel 0 is configured in output mode, and the MCH0MSEL[1:0] = 2b'00, these bits specify the multi mode channel 0 output signal polarity. 00: Multi mode channel 0 active high 01: Multi mode channel 0 active low 10: Reserved. 11: Reserved. When multi mode channel 0 is configured in input mode, these bits specify the multi mode channel 0 input signal's polarity. MCH0FP[1:0] will select the active trigger or capture polarity for multi mode channel 0 input signals. 00: Multi mode channel 0 input signal's rising edge is the active signal for capture or trigger operation in slave mode. And multi mode channel 0 input signal will not be inverted. 01: Multi mode channel 0 input signal's falling edge is the active signal for capture or trigger operation in slave mode. And multi mode channel 0 input signal will be inverted. 10: Reserved. 11: Noninverting/both multi mode channel 0 input signal's edges. This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 or 10.

Multi mode channel 0 capture/compare value register (TIMERx_MCH0CV)

Address offset: 0x54
Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Retained															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCH0VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.
15:0	MCH0VAL[15:0]	Capture or compare values for multimode channel 0 When multimode channel 0 is configured in input mode, these bits determine the counter value of the last captured event, and this register is read-only. When multimode channel 0 is configured for output mode, these bits contain the values that are to be compared with the counter. When the corresponding shadow register is enabled, the shadow register value is updated with each update event.

Multi mode channel 1 capture/compare value register (TIMERx_MCH1CV)

Address offset: 0x58

Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Retained															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCH1VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.
15:0	MCH1VAL[15:0]	Capture or compare values for multimode channel 1 When multimode channel 1 is configured in input mode, these bits determine the counter value of the last captured event, and this register is read-only. When multimode channel 1 is configured for output mode, these bits contain the values that are to be compared to the counter. When the corresponding shadow register is enabled, the shadow register value is updated with each update event.

Multi mode channel 2 capture/compare value register (TIMERx_MCH2CV)

Address offset: 0x5C

Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Retained															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCH2VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.
15:0	MCH2VAL[15:0]	Capture or compare values for multimode channel 2 When multimode channel 2 is configured in input mode, these bits determine the counter value of the last captured event and this register is read-only. When multimode channel 2 is configured for output mode, these bits contain the values that are to be compared to the counter. When the corresponding shadow register is enabled, the shadow register value is updated with each update event.

Multi mode channel 3 capture/compare value register (TIMERx_MCH3CV)

Address offset: 0x60

Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Retained															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCH3VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.
15:0	MCH3VAL[15:0]	Capture or compare values for multimode channel 3 When multimode channel 3 is configured in input mode, these bits determine the counter value of the last captured event, and this register is read-only. When multimode channel 3 is configured for output mode, these bits contain the values that are to be compared with the counter. When the corresponding shadow register is enabled, the shadow register value is updated with each update event.

Channel 0 additional capture/compare value register (TIMERx_CH0CV_ADD)

Address offset: 0x64

Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Retained															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH0VAL_ADD[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.
15:0	CH0VAL_ADD [15:0]	Channel 0 attaches a capture or comparison value When channel 0 is configured for input mode and multi-edge capture is enabled, these bits contain the counter comparison value of the second capture edge of the previous capture event. Moreover, this register is read-only. When channel 0 is configured in output mode, these bits contain the values that are about to be compared to the counter. When the corresponding shadow register is enabled, the shadow register value is updated with each update event. **Note**: This register is only used in composite PWM mode or asymmetric PWM mode (when CH0COMCTL[3:2] = 2'b10).

Channel 1 additional capture/compare value register (TIMERx_CH1CV_ADD)

Address offset: 0x68

Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Retained															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1VAL_ADD[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.
15:0	CH1VAL_ADD [15:0]	Channel 1 attaches capture or compare values When channel 1 is configured in input mode and multi-edge capture is enabled, these bits contain the counter comparison value of the second capture edge of the previous capture event. Moreover, this register is read-only. When channel 1 is additionally configured for output mode, these bits contain the values that are about to be compared with the counter. When the corresponding shadow register is enabled, the shadow register value is updated with each update event. **Note**: This register is only used in composite PWM mode or asymmetric PWM mode

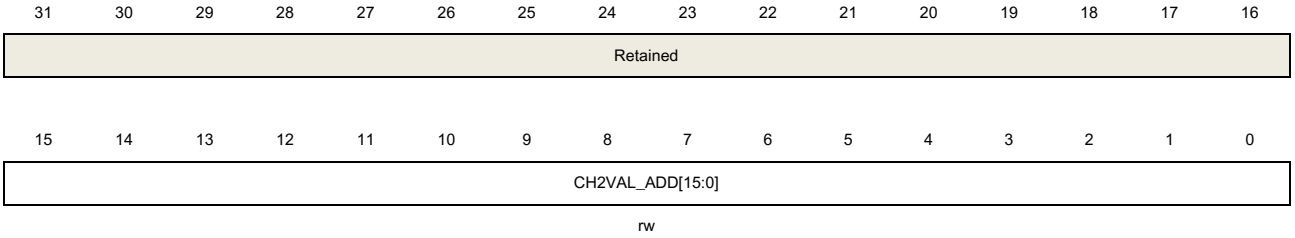
(when CH0COMCTL[3:2] = 2'b10).

Channel 2 additional capture/compare value register (TIMERx_CH2CV_ADD)

Address offset: 0x6C

Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).



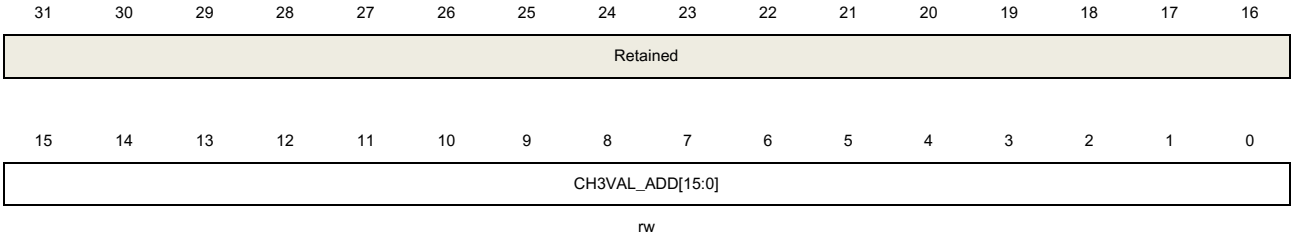
Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.
15:0	CH2VAL_ADD [15:0]	Channel 2 attaches capture or compare values When channel 2 is configured in input mode and multi-edge capture is enabled, these bits contain the counter comparison value of the second capture edge of the previous capture event. Moreover, this register is read-only. When channel 2 is additionally configured for output mode, these bits contain the values that are about to be compared to the counter. When the corresponding shadow register is enabled, the shadow register value is updated with each update event. **Note**: This register is only used in composite PWM mode or asymmetric PWM mode (when CH0COMCTL[3:2] = 2'b10).

Channel 3 additional capture/compare value register (TIMERx_CH3CV_ADD)

Address offset: 0x70

Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).



Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.
15:0	CH3VAL_ADD	Channel 3 attaches capture or compare values

[15:0] When channel 3 is configured in input mode and multi-edge capture is enabled, these bits contain the counter comparison value of the second capture edge of the previous capture event. Moreover, this register is read-only.
When channel 3 is additionally configured for output mode, these bits contain the values that are to be compared to the counter. When the corresponding shadow register is enabled, the shadow register value is updated with each update event.
Note: This register is only used in composite PWM mode or asymmetric PWM mode (when CH0COMCTL[3:2] = 2'b10).

Control register 2 (TIMERx_CTL2)

Address offset: 0x74
Reset value: 0x0FF0 00FF

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				MCH3MSEL[1:0]		MCH2MSEL[1:0]		MCH1MSEL[1:0]		MCH0MSEL[1:0]		DECDISDEN	DECJDEN	Reserved	
				rw		rw		rw		rw		rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3OMPSEL[1:0]		CH2OMPSEL[1:0]		CH1OMPSEL[1:0]		CH0OMPSEL[1:0]		BRKENCH3	BRKENCH2	BRKENCH1	BRKENCH0	DTIENCH3	DTIENCH2	DTIENCH1	DTIENCH0
rw		rw		rw		rw		rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:26	MCH3MSEL[1:0]	Multi mode channel 3 mode select 00: Independent mode, MCH3 is independent of CH3 01: Mirrored mode, just used for output, the MCH3 output is the same as CH3 output 10: Reserved 11: Complementary mode, only the CH3 is valid for input, and the outputs of MCH3 and CH3 are complementary
25:24	MCH2MSEL[1:0]	Multi mode channel 2 mode select 00: Independent mode, MCH2 is independent of CH2 01: Mirrored mode, just used for output, the MCH2 output is the same as CH2 output 10: Reserved 11: Complementary mode, only the CH2 is valid for input, and the outputs of MCH2 and CH2 are complementary
23:22	MCH1MSEL[1:0]	Multi mode channel 1 mode select 00: Independent mode, MCH1 is independent of CH1 01: Mirrored mode, just used for output, the MCH1 output is the same as CH1 output 10: Reserved 11: Complementary mode, only the CH1 is valid for input, and the outputs of MCH1

		and CH1 are complementary
21:20	MCH0MSEL[1:0]	Multi mode channel 0 mode select 00: Independent mode, MCH0 is independent of CH0 01: Mirrored mode, just used for output, the MCH0 output is the same as CH0 output 10: Reserved 11: Complementary mode, only the CH0 is valid for input, and the outputs of MCH0 and CH0 are complementary
19	DECDISDEN	Quadrature decoder signal disconnection detection enable 0: Quadrature decoder signal disconnection detection is disabled 1: Quadrature decoder signal disconnection detection is enabled
18	DECJDEN	Quadrature decoder signal jump (the two signals jump at the same time) detection enable 0: Quadrature decoder signal jump detection is disabled 1: Quadrature decoder signal jump detection is enabled
17:16	Reserved	Must be kept at reset value.
15:14	CH3OMPSEL[1:0]	Channel 3 output match pulse select When the match events occur, this bit is used to select the output of O3CPRE which drives CH3_O. 00: The O3CPRE signal is output normally with the configuration of CH3COMCTL[2:0] bits. 01: Only the counter is counting up, the O3CPRE signal is output a pulse when the match events occur, and the pulse width is one CK_TIMER clock cycle. 10: Only the counter is counting down, the O3CPRE signal is output a pulse when the match events occur, and the pulse width is one CK_TIMER clock cycle. 11: Both the counter is counting up and counting down, the O3CPRE signal is output a pulse when the match events occur, and the pulse width is one CK_TIMER clock cycle.
13:12	CH2OMPSEL[1:0]	Channel 2 output match pulse select When the match events occurs, this bit is used to select the output of O2CPRE which drives CH2_O. 00: The O2CPRE signal is output normal with the configuration of CH2COMCTL[2:0] bits. 01: Only when the counter is counting up, the O2CPRE signal is output a pulse when the match events occurs, and the pulse width is one CK_TIMER clock cycle. 10: Only when the counter is counting down, the O2CPRE signal is output a pulse when the match events occurs, and the pulse width is one CK_TIMER clock cycle. 11: Both when the counter is counting up and counting down, the O2CPRE signal is output a pulse when the match events occurs, and the pulse width is one CK_TIMER clock cycle.

11:10	CH1OMPSEL[1:0]	Channel 1 output match pulse select When the match events occurs, this bit is used to select the output of O1CPRE which drives CH1_O. 00: The O1CPRE signal is output normal with the configuration of CH1COMCTL[2:0] bits. 01: Only when the counter is counting up, the O1CPRE signal is output a pulse when the match events occurs, and the pulse width is one CK_TIMER clock cycle. 10: Only when the counter is counting down, the O1CPRE signal is output a pulse when the match events occurs, and the pulse width is one CK_TIMER clock cycle. 11: Both when the counter is counting up and counting down, the O1CPRE signal is output a pulse when the match events occurs, and the pulse width is one CK_TIMER clock cycle.
9:8	CH0OMPSEL[1:0]	Channel 0 output match pulse select When the match events occurs, this bit is used to select the output of O0CPRE which drives CH0_O. 00: The O0CPRE signal is output normal with the configuration of CH0COMCTL[2:0] bits. 01: Only when the counter is counting up, the O0CPRE signal is output a pulse when the match events occurs, and the pulse width is one CK_TIMER clock cycle. 10: Only when the counter is counting down, the O0CPRE signal is output a pulse when the match events occurs, and the pulse width is one CK_TIMER clock cycle. 11: Both when the counter is counting up and counting down, the O0CPRE signal is output a pulse when the match events occurs, and the pulse width is one CK_TIMER clock cycle.
7	BRKENCH3	Break control enable for channel 3 0: Disabled 1: Enabled
6	BRKENCH2	Break control enable for channel 2 0: Disabled 1: Enabled
5	BRKENCH1	Break control enable for channel 1 0: Disabled 1: Enabled
4	BRKENCH0	Break control enable for channel 0 0: Disabled 1: Enabled
3	DTIENCH3	Dead time inserted enable for channel 3 Enables the deadtime insertion in the outputs of MCH3_O and CH3_O. 0: Disabled 1: Enabled
2	DTIENCH2	Dead time inserted enable for channel 2

Enables the deadtime insertion in the outputs of MCH2_O and CH2_O.
0: Disabled
1: Enabled

1	DTIENCH1	Dead time inserted enable for channel 1 Enables the deadtime insertion in the outputs of MCH1_O and CH1_O. 0: Disabled 1: Enabled
0	DTIENCH0	Dead time inserted enable for channel 0 Enables the deadtime insertion in the outputs of MCH0_O and CH0_O. 0: Disabled 1: Enabled

Break configuration register (TIMERx_BRKCFG)

Address offset: 0x78
Reset value: 0x5500 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BRK3P	BRK3EN	BRK2P	BRK2EN	BRK1P	BRK1EN	BRK0P	BRK0EN	Reserved							
rw	rw	rw	rw	rw	rw	rw	rw								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BRK3F[3:0]				BRK2F[3:0]				BRK1F[3:0]				BRK0F[3:0]			
rw				rw				rw				rw			

Bits	Fields	Descriptions
31	BRK3P	BRKIN3 input signal polarity This bit specifies the polarity of the BRKIN3 input signal. 0: BRKIN3 input active low 1: BRKIN3 input active high This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.
30	BRK3EN	BRKIN3 input signal enable This bit can be set to enable the BRKIN3 input. 0: Break inputs disabled 1: Break inputs enabled This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.
29	BRK2P	BRKIN2 input signal polarity This bit specifies the polarity of the BRKIN2 input signal. 0: BRKIN2 input active low 1: BRKIN2 input active high

This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.

28 BRK2EN
BRKIN2 input signal enable
This bit can be set to enable the BRKIN2 input.
0: Break inputs disabled
1: Break inputs enabled
This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.

27 BRK1P
BRKIN1 input signal polarity
This bit specifies the polarity of the BRKIN1 input signal.
0: BRKIN1 input active low
1: BRKIN1 input active high
This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.

26 BRK1EN
BRKIN1 input signal enable
This bit can be set to enable the BRKIN1 input.
0: Break inputs disabled
1: Break inputs enabled
This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.

25 BRK0P
BRKIN0 input signal polarity
This bit specifies the polarity of the BRKIN0 input signal.
0: BRKIN0 input active low
1: BRKIN0 input active high
This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.

24 BRK0EN
BRKIN0 input signal enable
This bit can be set to enable the BRKIN0 input.
0: Break inputs disabled
1: Break inputs enabled
This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.

23:16 Reserved
Must be kept at reset value.

15:12 BRK3F[3:0]
BRKIN3 input signal filter
An event counter is used in the digital filter, in which a transition on the output occurs after N input events. This bit-field specifies the frequency used to sample BRKIN3 input signal and the length of the digital filter applied to BRKIN3.
0000: Filter disabled. BRKIN3 act asynchronously, N=1
0001: $f_{SAMP} = f_{CK_TIMER}$, N=2
0010: $f_{SAMP} = f_{CK_TIMER}$, N=4
0011: $f_{SAMP} = f_{CK_TIMER}$, N=8

0100: $f_{SAMP} = f_{DTS}/2$, N=6
0101: $f_{SAMP} = f_{DTS}/2$, N=8
0110: $f_{SAMP} = f_{DTS}/4$, N=6
0111: $f_{SAMP} = f_{DTS}/4$, N=8
1000: $f_{SAMP} = f_{DTS}/8$, N=6
1001: $f_{SAMP} = f_{DTS}/8$, N=8
1010: $f_{SAMP} = f_{DTS}/16$, N=5
1011: $f_{SAMP} = f_{DTS}/16$, N=6
1100: $f_{SAMP} = f_{DTS}/16$, N=8
1101: $f_{SAMP} = f_{DTS}/32$, N=5
1110: $f_{SAMP} = f_{DTS}/32$, N=6
1111: $f_{SAMP} = f_{DTS}/32$, N=8
This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.

11:8	BRK2F[3:0]	BRKIN2 input signal filter An event counter is used in the digital filter, in which a transition on the output occurs after N input events. This bit-field specifies the frequency used to sample BRKIN2 input signal and the length of the digital filter applied to BRKIN2. 0000: Filter disabled. BRKIN2 act asynchronously, N=1 0001: $f_{SAMP} = f_{CK_TIMER}$, N=2 0010: $f_{SAMP} = f_{CK_TIMER}$, N=4 0011: $f_{SAMP} = f_{CK_TIMER}$, N=8 0100: $f_{SAMP} = f_{DTS}/2$, N=6 0101: $f_{SAMP} = f_{DTS}/2$, N=8 0110: $f_{SAMP} = f_{DTS}/4$, N=6 0111: $f_{SAMP} = f_{DTS}/4$, N=8 1000: $f_{SAMP} = f_{DTS}/8$, N=6 1001: $f_{SAMP} = f_{DTS}/8$, N=8 1010: $f_{SAMP} = f_{DTS}/16$, N=5 1011: $f_{SAMP} = f_{DTS}/16$, N=6 1100: $f_{SAMP} = f_{DTS}/16$, N=8 1101: $f_{SAMP} = f_{DTS}/32$, N=5 1110: $f_{SAMP} = f_{DTS}/32$, N=6 1111: $f_{SAMP} = f_{DTS}/32$, N=8 This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.
7:4	BRK1F[3:0]	BRKIN1 input signal filter An event counter is used in the digital filter, in which a transition on the output occurs after N input events. This bit-field specifies the frequency used to sample BRKIN1 input signal and the length of the digital filter applied to BRKIN1. 0000: Filter disabled. BRKIN1 act asynchronously, N=1 0001: $f_{SAMP} = f_{CK_TIMER}$, N=2 0010: $f_{SAMP} = f_{CK_TIMER}$, N=4

0011: $f_{SAMP} = f_{CK_TIMER}$, N=8
0100: $f_{SAMP} = f_{DTS}/2$, N=6
0101: $f_{SAMP} = f_{DTS}/2$, N=8
0110: $f_{SAMP} = f_{DTS}/4$, N=6
0111: $f_{SAMP} = f_{DTS}/4$, N=8
1000: $f_{SAMP} = f_{DTS}/8$, N=6
1001: $f_{SAMP} = f_{DTS}/8$, N=8
1010: $f_{SAMP} = f_{DTS}/16$, N=5
1011: $f_{SAMP} = f_{DTS}/16$, N=6
1100: $f_{SAMP} = f_{DTS}/16$, N=8
1101: $f_{SAMP} = f_{DTS}/32$, N=5
1110: $f_{SAMP} = f_{DTS}/32$, N=6
1111: $f_{SAMP} = f_{DTS}/32$, N=8
This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.

3:0	BRK0F[3:0]	BRKIN0 input signal filter An event counter is used in the digital filter, in which a transition on the output occurs after N input events. This bit-field specifies the frequency used to sample BRKIN0 input signal and the length of the digital filter applied to BRKIN0. 0000: Filter disabled. BRKIN0 act asynchronously, N=1 0001: $f_{SAMP} = f_{CK_TIMER}$, N=2 0010: $f_{SAMP} = f_{CK_TIMER}$, N=4 0011: $f_{SAMP} = f_{CK_TIMER}$, N=8 0100: $f_{SAMP} = f_{DTS}/2$, N=6 0101: $f_{SAMP} = f_{DTS}/2$, N=8 0110: $f_{SAMP} = f_{DTS}/4$, N=6 0111: $f_{SAMP} = f_{DTS}/4$, N=8 1000: $f_{SAMP} = f_{DTS}/8$, N=6 1001: $f_{SAMP} = f_{DTS}/8$, N=8 1010: $f_{SAMP} = f_{DTS}/16$, N=5 1011: $f_{SAMP} = f_{DTS}/16$, N=6 1100: $f_{SAMP} = f_{DTS}/16$, N=8 1101: $f_{SAMP} = f_{DTS}/32$, N=5 1110: $f_{SAMP} = f_{DTS}/32$, N=6 1111: $f_{SAMP} = f_{DTS}/32$, N=8 This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.

Free complementary channel protection register 0 (TIMERx_FCCHP0)

Address offset: 0x7C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

This register is used to configure the outputs of CH0_O/MCH0_O.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FCCHP0EN	Reserved														
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			DT0SEN	ROS	IOS	Reserved		DTCFG[7:0]							
			rw	rw	rw			rw							

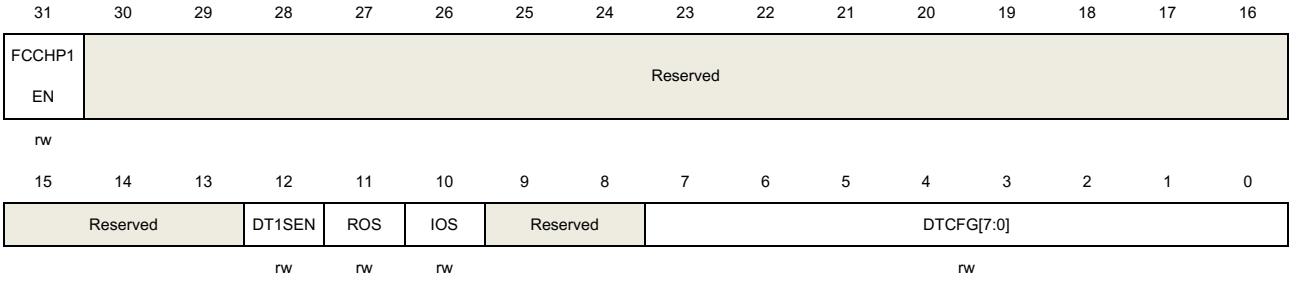
Bits	Fields	Descriptions
31	FCCHP0EN	Free complementary channel protection register 0 enable 0: the ROS、IOS and DTCFG[7:0] bits in TIMERx_CCHP register is active 1: the ROS、IOS and DTCFG[7:0] bits in TIMERx_FCCHP0 register is active This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.
30:13	Reserved	Must be kept at reset value.
12	DT0SEN	Channel 0 deadtime shadow is enabled. When the bit is set, dynamic change of channel 0 deadtime is supported. The deadtime takes effect when the update event arrives. 0: Channel 0 deadtime shadow register is disabled. 1: Channel 0 deadtime shadow register is enabled. This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.
11	ROS	Run mode off-state configure When POEN bit is set, this bit specifies the output state for the channels which has a complementary output and has been configured in output mode. 0: "off-state" disabled. If the CH0EN or MCH0EN bit is reset, the corresponding channel is output disabled. 1: "off-state" enabled. If the CH0EN or MCH0EN bit is reset, the corresponding channel is "off-state". This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 10 or 11.
10	IOS	Idle mode off-state configure When POEN bit is reset, this bit specifies the output state for the channels which has been configured in output mode. 0: "off-state" disabled. If the CH0EN/CH0NEN bits are both reset, the channels are output disabled. 1: "off-state" enabled. No matter the CH0EN/MCH0EN bits, the channels are "off-state". This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 10 or 11.

9:8	Reserved	Must be kept at reset value.
7:0	DTCFG[7:0]	Dead time configure This bit-field controls the value of the dead-time, which is inserted before the output transitions. The relationship between DTCFG value and the duration of dead-time is as follow: DTCFG [7:5] =3'b0xx: DTvalue = DTCFG [7:0]x t_{DT}, t_{DT}=t_{DTS}. DTCFG [7:5] =3'b10x: DTvalue = (64+DTCFG [5:0])xt_{DT}, t_{DT} =t_{DTS}*2. DTCFG [7:5] =3'b110: DTvalue = (32+DTCFG [4:0])xt_{DT}, t_{DT}=t_{DTS}*8. DTCFG [7:5] =3'b111: DTvalue = (32+DTCFG [4:0])xt_{DT}, t_{DT} =t_{DTS}*16. This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.

Free complementary channel protection register 1 (TIMERx_FCCHP1)

Address offset: 0x80
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).
This register is used to configure the outputs of CH1_O/MCH1_O.



Bits	Fields	Descriptions
31	FCCHP1EN	Free complementary channel protection register 1 enable 0: the ROS、IOS and DTCFG[7:0] bits in TIMERx_CCHP register is active 1: the ROS、IOS and DTCFG[7:0] bits in TIMERx_FCCHP1 register is active This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.
30:13	Reserved	Must be kept at reset value.
12	DT1SEN	Channel 1 deadtime shadow is enabled. When the bit is set, dynamic change of channel 1 deadtime is supported. The deadtime takes effect when the update event arrives. 0: Channel 1 deadtime shadow register is disabled. 1: Channel 1 deadtime shadow register is enabled. This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.
11	ROS	Run mode off-state configure

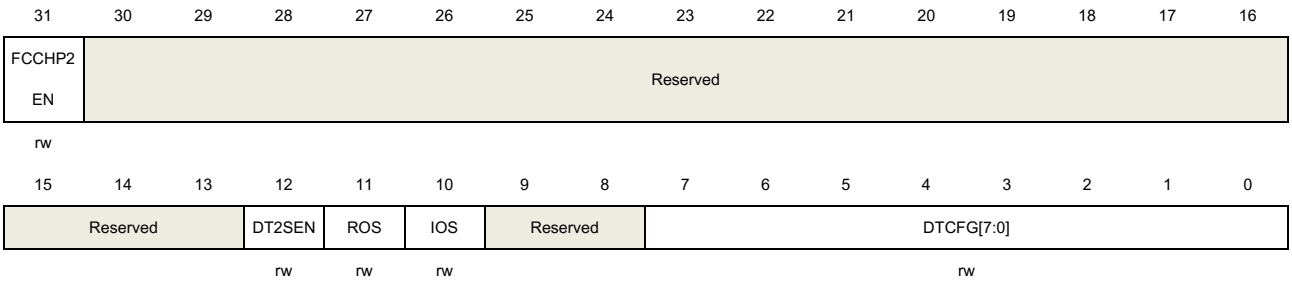
When POEN bit is set, this bit specifies the output state for the channels which has a complementary output and has been configured in output mode.
0: "off-state" disabled. If the CH1EN or MCH1EN bit is reset, the corresponding channel is output disabled.
1: "off-state" enabled. If the CH1EN or MCH1EN bit is reset, the corresponding channel is "off-state".
This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 10 or 11.

10	IOS	Idle mode off-state configure When POEN bit is reset, this bit specifies the output state for the channels which has been configured in output mode. 0: "off-state" disabled. If the CH1EN/CH1NEN bits are both reset, the channels are output disabled. 1: "off-state" enabled. No matter the CH1EN/MCH1EN bits, the channels are "off-state". This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 10 or 11.
9:8	Reserved	Must be kept at reset value.
7:0	DTCFG[7:0]	Dead time configure This bit-field controls the value of the dead-time, which is inserted before the output transitions. The relationship between DTCFG value and the duration of dead-time is as follow: DTCFG [7:5] =3'b0xx: DTvalue = DTCFG [7:0]x t_{DT}, t_{DT}=t_{DTS}. DTCFG [7:5] =3'b10x: DTvalue = (64+DTCFG [5:0])xt_{DT}, t_{DT} =t_{DTS}*2. DTCFG [7:5] =3'b110: DTvalue = (32+DTCFG [4:0])xt_{DT}, t_{DT}=t_{DTS}*8. DTCFG [7:5] =3'b111: DTvalue = (32+DTCFG [4:0])xt_{DT}, t_{DT} =t_{DTS}*16. This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.

Free complementary channel protection register 2 (TIMERx_FCCHP2)

Address offset: 0x84
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).
This register is used to configure the outputs of CH2_O/MCH2_O.



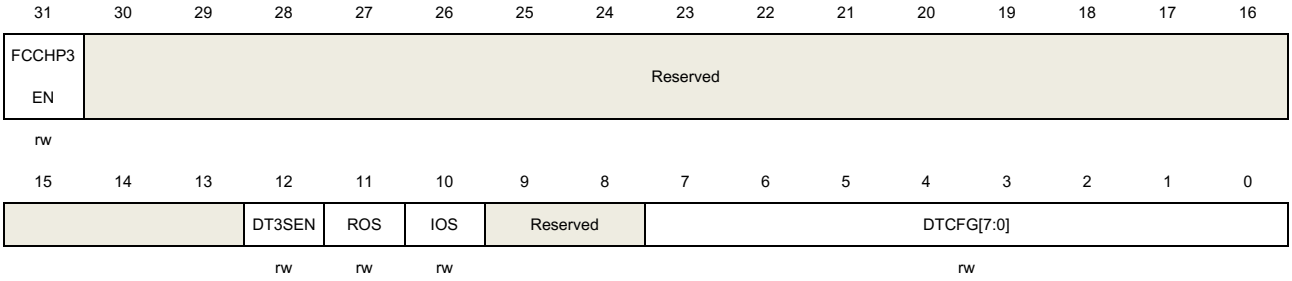
Bits	Fields	Descriptions
31	FCCHP2EN	Free complementary channel protection register 2 enable 0: the ROS、IOS and DTCFG[7:0] bits in TIMERx_CCHP register is active 1: the ROS、IOS and DTCFG[7:0] bits in TIMERx_FCCHP2 register is active This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.
30:13	Reserved	Must be kept at reset value.
12	DT2SEN	Channel 2 deadtime shadow is enabled. When the bit is set, dynamic change of channel 2 deadtime is supported. The deadtime takes effect when the update event arrives. 0: Channel 2 deadtime shadow register is disabled. 1: Channel 2 deadtime shadow register is enabled. This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.
11	ROS	Run mode off-state configure When POEN bit is set, this bit specifies the output state for the channels which has a complementary output and has been configured in output mode. 0: "off-state" disabled. If the CH2EN or MCH2EN bit is reset, the corresponding channel is output disabled. 1: "off-state" enabled. If the CH2EN or MCH2EN bit is reset, the corresponding channel is "off-state". This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 10 or 11.
10	IOS	Idle mode off-state configure When POEN bit is reset, this bit specifies the output state for the channels which has been configured in output mode. 0: "off-state" disabled. If the CH2EN/CH2NEN bits are both reset, the channels are output disabled. 1: "off-state" enabled. No matter the CH2EN/MCH2EN bits, the channels are "off-state". This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 10 or 11.
9:8	Reserved	Must be kept at reset value.
7:0	DTCFG[7:0]	Dead time configure This bit-field controls the value of the dead-time, which is inserted before the output transitions. The relationship between DTCFG value and the duration of dead-time is as follow: DTCFG [7:5] =3'b0xx: DTvalue = DTCFG [7:0]x t_{DT}, t_{DT}=t_{DTS}. DTCFG [7:5] =3'b10x: DTvalue = (64+DTCFG [5:0])xt_{DT}, t_{DT} =t_{DTS}*2. DTCFG [7:5] =3'b110: DTvalue = (32+DTCFG [4:0])xt_{DT}, t_{DT}=t_{DTS}*8.

DTCFG [7:5] =3'b111: DTvalue = (32+DTCFG [4:0])xt_{DT}, t_{DT} =t_{DTS}*16.
This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.

Free complementary channel protection register 3 (TIMERx_FCCHP3)

Address offset: 0x88
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).
This register is used to configure the outputs of CH3_O/MCH3_O.



Bits	Fields	Descriptions
31	FCCHP3EN	Free complementary channel protection register 0 enable 0: the ROS、IOS and DTCFG[7:0] bits in TIMERx_CCHP register is active 1: the ROS、IOS and DTCFG[7:0] bits in TIMERx_FCCHP3 register is active This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.
30:13	Reserved	Must be kept at reset value.
12	DT3SEN	Channel 3 deadtime shadow is enabled. When the bit is set, dynamic change of channel 3 deadtime is supported. The deadtime takes effect when the update event arrives. 0: Channel 3 deadtime shadow register is disabled. 1: Channel 3 deadtime shadow register is enabled. This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.
11	ROS	Run mode off-state configure When POEN bit is set, this bit specifies the output state for the channels which has a complementary output and has been configured in output mode. 0: "off-state" disabled. If the CH3EN or MCH3EN bit is reset, the corresponding channel is output disabled. 1: "off-state" enabled. If the CH3EN or MCH3EN bit is reset, the corresponding channel is "off-state". This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 10 or 11.

10	IOS	Idle mode off-state configure When POEN bit is reset, this bit specifies the output state for the channels which has been configured in output mode. 0: "off-state" disabled. If the CH3EN/CH3NEN bits are both reset, the channels are output disabled. 1: "off-state" enabled. No matter the CH3EN/MCH3EN bits, the channels are "off-state". This bit cannot be modified when PROT [1:0] bit-filed in TIMERx_CCHP register is 10 or 11.
9:8	Reserved	Must be kept at reset value.
7:0	DTCFG[7:0]	Dead time configure This bit-field controls the value of the dead-time, which is inserted before the output transitions. The relationship between DTCFG value and the duration of dead-time is as follow: DTCFG [7:5] =3'b0xx: DTvalue = DTCFG [7:0]x t_{DT}, t_{DT}=t_{DTS}. DTCFG [7:5] =3'b10x: DTvalue = (64+DTCFG [5:0])xt_{DT}, t_{DT} =t_{DTS}*2. DTCFG [7:5] =3'b110: DTvalue = (32+DTCFG [4:0])xt_{DT}, t_{DT}=t_{DTS}*8. DTCFG [7:5] =3'b111: DTvalue = (32+DTCFG [4:0])xt_{DT}, t_{DT} =t_{DTS}*16. This bit can be modified only when PROT [1:0] bit-filed in TIMERx_CCHP register is 00.

Multi mode channel 0 additional capture/compare value register (TIMERx_MCH0CV_ADD)

Address offset: 0x90

Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Retained															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCH0VAL_ADD[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.
15:0	MCH0VAL_ADD [15:0]	Multimode channel 0 attaches capture or comparison values When multimode channel 0 is configured for input mode and multi-edge capture is enabled, these bits contain the counter comparison value of the second capture edge of the previous capture event. Moreover, this register is read-only. When multimode channel 0 is configured for output mode, these bits contain the values that are to be compared with the counter. When the corresponding shadow

register is enabled, the shadow register value is updated with each update event.
Note: This register is only used in composite PWM mode or asymmetric PWM mode (when MCH0COMCTL[3:2] = 2'b10).

Multi mode channel 1 additional capture/compare value register (TIMERx_MCH1CV_ADD)

Address offset: 0x94
Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Retained															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCH1VAL_ADD[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.
15:0	MCH1VAL_ADD [15:0]	Multimode channel 1 attaches capture or comparison values When multimode channel 1 is configured for input mode and multi-edge capture is enabled, these bits contain the counter comparison value of the second capture edge of the previous capture event. Moreover, this register is read-only. When multimode channel 1 is configured for output mode, these bits contain the values that are to be compared to the counter. When the corresponding shadow register is enabled, the shadow register value is updated with each update event. **Note**: This register is only used in composite PWM mode or asymmetric PWM mode (when MCH1COMCTL[3:2] = 2'b10).

Multi mode channel 2 additional capture/compare value register (TIMERx_MCH2CV_ADD)

Address offset: 0x98
Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Retained															

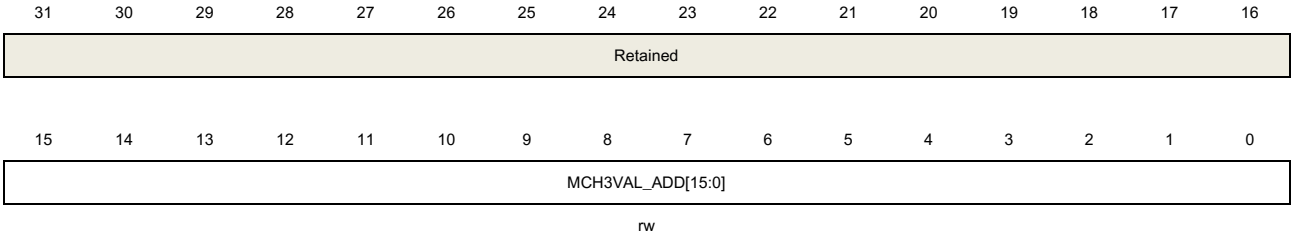
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCH2VAL_ADD[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.
15:0	MCH2VAL_ADD [15:0]	Multimode channel 2 additional capture or comparison values When multimode channel 2 is configured for input mode and multi-edge capture is enabled, these bits contain the counter comparison value of the second capture edge of the previous capture event. Moreover, this register is read-only. When multimode channel 2 is configured for output mode, these bits contain the values that are to be compared to the counter. When the corresponding shadow register is enabled, the shadow register value is updated with each update event. **Note**: This register is only used in composite PWM mode or asymmetric PWM mode (when MCH2COMCTL[3:2] = 2'b10).

Multi mode channel 3 additional capture/compare value register (TIMERx_MCH3CV_ADD)

Address offset: 0x9C
Reset value: 0x0000 0000

This register can only be accessed by word (32 bits).



Bits	Fields	Descriptions
31:16	Retained	The reset value must be maintained.
15:0	MCH3CVAL_ADD [15:0]	Multimode channel 3 additional capture or comparison values When multimode channel 3 is configured for input mode and multi-edge capture is enabled, these bits contain the counter comparison value of the second capture edge of the previous capture event. Moreover, this register is read-only. When multimode channel 3 is configured for output mode, these bits contain the values that are to be compared with the counter. When the corresponding shadow register is enabled, the shadow register value is updated with each update event. **Note**: This register is only used in composite PWM mode or asymmetric PWM mode (when MCH3COMCTL[3:2] = 2'b10).

Watchdog counter period register (TIMERx_WDGPER)

Address offset: 0xA0
Reset value: 0xFFFF FFFF

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WDGPER[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDGPER[15:0]															
rw															

Bits	Fields	Descriptions
31:0	WDGPER[31:0]	Watchdog counter period value This register contains the period of the two watchdog counter. When the counters continues to count to this value, the counter will timeout and the interrupt flag DECDISIF is set. If DECDISIE=1, the corresponding interrupt is generated. Note: This register is just used in quadrature decoder signal disconnection detection function(with DECDISDEN =1).

Counter initial control register (TIMERx_CINITCTL)

Address offset: 0xA4

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													SWSYNCG	CINITDIR	CINITVEN
													rw	rw	rw

Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2	SWSYNCG	Software synchronization event generation This bit is set by software and cleared by hardware automatically. Reads this bit always return a 0. 0: No affect. 1: Generate software synchronization event
1	CINITDIR	Counter initial count direction 0: When the synchronization event occurs, the counter count up. 1: When the synchronization event occurs, the counter count down. This bit indicates the initial direction of the counter after a synchronization event occurs and the counter initial value is loaded from the TIMERx_CINITV register. **Note**: This bit is only used when the CAM[1:0] != 00.
0	CINITVEN	Counter initial value register enable

0: Counter initial value register disable. The counter register can't load from the counter initial value register.

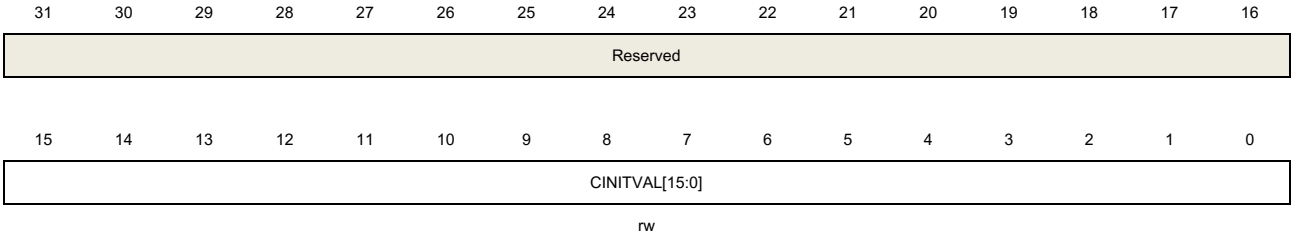
1: Counter initial value register enable. When a synchronization event (soft synchronization event generated by setting the SWSYNCG bit or by the GPIO input event) occurs, the counter register can load from the counter initial value register.

Counter initial value register (TIMERx_CINITV)

Address offset: 0xA8

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



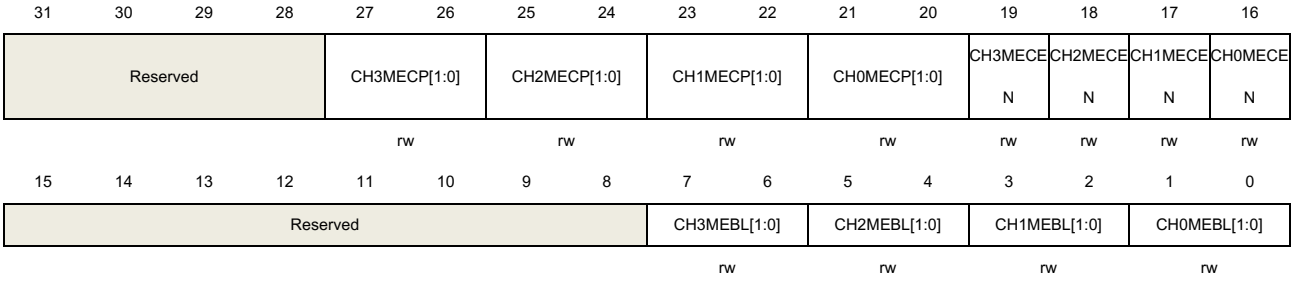
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CINITVAL[15:0]	Counter initial value This bits-field indicates the counter initial value. When the CINITVEN bit is 0, the counter register can't load the initial value which are set in CINITVAL bits-field. When the CINITVEN bit is 1, when the synchronization event occurs, the counter register can load the initial value which are set in CINITVAL bits-field.

Channel multi-edge capture configuration register (TIMERx_CHMECAPCFG)

Address offset: 0xAC

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.

27:26	CH3MECP[1:0]	Channel 3 multi-edge capture polarity. Refer to CH0MECP [1:0] description.
25:24	CH2MECP[1:0]	Channel 2 multi-edge capture polarity. Refer to CH0MECP [1:0] description.
23:22	CH1MECP[1:0]	Channel 1 multi-edge capture polarity. Refer to CH0MECP [1:0] description.
21:20	CH0MECP[1:0]	Channel 0 multi edge capture polarity. This bit defines the edge polarity of the captured signal in the two-sided edge capture mode of the channel 00: The effective edge of the first capture is the rising edge 01: The effective edge captured for the first time is the falling edge 10: Reserved 11: The effective edge captured for the first time is the edge of any polarity Note: This register is only valid if [MCH0P, CH0P] is configured as 2 'b11. This bit cannot be changed when PROT [1:0]=11 or 10 of the TIMERx_CCHP register. When CH0MS = 3 'b011, the configuration of this bit is meaningless
19	CH3MECEN	Channel 3 multi edge capture enable Refer to CH0MECEN description.
18	CH2MECEN	Channel 2 multi edge capture enable Refer to CH0MECEN description.
17	CH1MECEN	Channel 1 multi edge capture enable Refer to CH0MECEN description.
16	CH0MECEN	Channel 0 multi edge capture enable This bit is only valid if the channel is configured for input capture mode. If enabled, a channel capture/compare interrupt is triggered only after several capture behaviors have occurred. 0: enable 1: disable
15:8	Reserved	Must be kept at reset value.
7:6	CH3MEBL[1:0]	Channel 3 multi edge buffer level Refer to CH0MEBL[1:0] description.
5:4	CH2MEBL[1:0]	Channel 2 multi edge buffer level Refer to CH0MEBL[1:0] description.
3:2	CH1MEBL[1:0]	Channel 1 multi edge buffer level Refer to CH0MEBL[1:0] description.
1:0	CH0MEBL[1:0]	Channel 0 multi edge buffer level 00: A capture interrupt is occurred after one capture. 01: A capture interrupt is occurred after two capture. 10: A capture interrupt is occurred after three capture.

10: A capture interrupt is occurred after four capture.

Multi mode channel multi-edge capture configuration register (TIMERx_MCHMECAPCFG)

Address offset: 0xB0

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				MCH3MECP[1:0]		MCH2MECP[1:0]		MCH1MECP[1:0]		MCH0MECP[1:0]		MCH3MECEN	MCH2MECEN	MCH1MECEN	MCH0MECEN
				rw		rw		rw		rw		rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								MCH3MEBL[1:0]		MCH2MEBL[1:0]		MCH1MEBL[1:0]		MCH0MEBL[1:0]	
								rw		rw		rw		rw	

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:26	MCH3MECP[1:0]	Multi mode channel 3 multi-edge capture polarity. Refer to CH0MECP [1:0] description.
25:24	MCH2MECP[1:0]	Multi mode channel 2 multi-edge capture polarity. Refer to CH0MECP [1:0] description.
23:22	MCH1MECP[1:0]	Multi mode channel 1 multi-edge capture polarity. Refer to CH0MECP [1:0] description.
21:20	MCH0MECP[1:0]	Multi mode channel 0 multi edge capture polarity. This bit defines the edge polarity of the captured signal in the two-sided edge capture mode of the channel 00: The effective edge of the first capture is the rising edge 01: The effective edge captured for the first time is the falling edge 10: Reserved 11: The effective edge captured for the first time is the edge of any polarity Note: This register is only valid if [MCH0P, CH0P] is configured as 2 'b11. This bit cannot be changed when PROT [1:0]=11 or 10 of the TIMERx_CCHP register. When CH0MS = 3 'b011, the configuration of this bit is meaningless
19	MCH3MECEN	Multi mode channel 3 multi edge capture enable Refer to CH0MECEN description.
18	MCH2MECEN	Multi mode channel 2 multi edge capture enable Refer to CH0MECEN description.
17	MCH1MECEN	Multi mode channel 1 multi edge capture enable

		Refer to CH0MECEN description.
16	MCH0MECEN	Multi mode channel 0 multi edge capture enable This bit is only valid if the channel is configured for input capture mode. If enabled, a channel capture/compare interrupt is triggered only after several capture behaviors have occurred. 0: enable 1: disable
15:8	Reserved	Must be kept at reset value.
7:6	MCH3MEBL[1:0]	Multi mode channel 3 multi edge buffer level Refer to CH0MEBL[1:0] description.
5:4	MCH2MEBL[1:0]	Multi mode channel 2 multi edge buffer level Refer to CH0MEBL[1:0] description.
3:2	MCH1MEBL[1:0]	Multi mode channel 1 multi edge buffer level Refer to CH0MEBL[1:0] description.
1:0	MCH0MEBL[1:0]	Multi mode channel 0 multi edge buffer level 00: A capture interrupt is occurred after one capture. 01: A capture interrupt is occurred after two capture. 10: A capture interrupt is occurred after three capture. 10: A capture interrupt is occurred after four capture.

Channel 0 multi-edge capture register(TIMERx_CH0MECAP)

Address offset: 0xB4

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH0CAP4[15:0]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH0CAP3[15:0]															
r															

Bits	Fields	Descriptions
31:16	CH0CAP4[15:0]	Channel 0 captures 4 count values When channel 0 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the fourth capture edge of the last capture event. **Note**: And this register is read-only.
15:0	CH0CAP3[15:0]	Channel 0 captures 3 count values When channel 0 is configured in input mode and multi edge capture is enabled,

these bits contain the counter comparison value for the third capture edge of the last capture event.

Note: And this register is read-only.

Channel 1 multi-edge capture register(TIMERx_CH1MECAP)

Address offset: 0xB8

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH1CAP4[15:0]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1CAP3[15:0]															
r															

Bits	Fields	Descriptions
31:16	CH1CAP4[15:0]	Channel 1 captures 4 count values When channel 1 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the fourth capture edge of the last capture event. **Note**: And this register is read-only.
15:0	CH1CAP3[15:0]	Channel 1 captures 3 count values When channel 1 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the third capture edge of the last capture event. **Note**: And this register is read-only.

Channel 2 multi-edge capture register(TIMERx_CH2MECAP)

Address offset: 0xBC

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH2CAP4[15:0]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH2CAP3[15:0]															
r															

Bits	Fields	Descriptions
31:16	CH2CAP4[15:0]	Channel 2 captures 4 count values

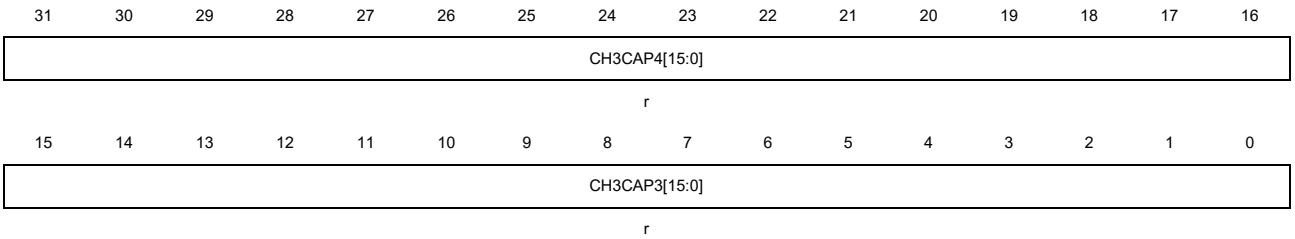
When channel 2 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the fourth capture edge of the last capture event.
Note: And this register is read-only.

15:0 CH2CAP3[15:0] Channel 2 captures 3 count values
When channel 2 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the third capture edge of the last capture event.
Note: And this register is read-only.

Channel 3 multi-edge capture register(TIMERx_CH3MECAP)

Address offset: 0XC0
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

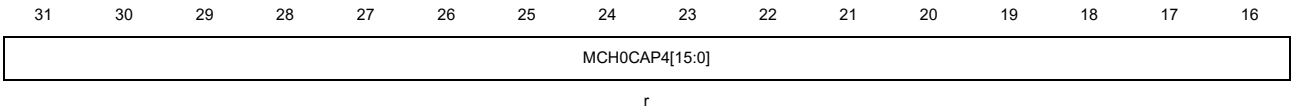


Bits	Fields	Descriptions
31:16	CH3CAP4[15:0]	Channel 3 captures 4 count values When channel 3 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the fourth capture edge of the last capture event. **Note**: And this register is read-only.
15:0	CH3CAP3[15:0]	Channel 3 captures 3 count values When channel 3 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the third capture edge of the last capture event. **Note**: And this register is read-only.

Multi mode channel 0 multi-edge capture register(TIMERx_MCH0MECAP)

Address offset: 0xC4
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCH0CAP3[15:0]															
r															

Bits	Fields	Descriptions
31:16	MCH0CAP4[15:0]	Multi mode channel 0 captures 4 count values When multi mode channel 0 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the fourth capture edge of the last capture event. **Note**: And this register is read-only.
15:0	MCH0CAP3[15:0]	Multi mode channel 0 captures 3 count values When multi mode channel 0 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the third capture edge of the last capture event. **Note**: And this register is read-only.

Multi mode channel 1 multi-edge capture register(TIMERx_MCH1MECAP)

Address offset: 0xC8

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MCH1CAP4[15:0]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCH1CAP3[15:0]															
r															

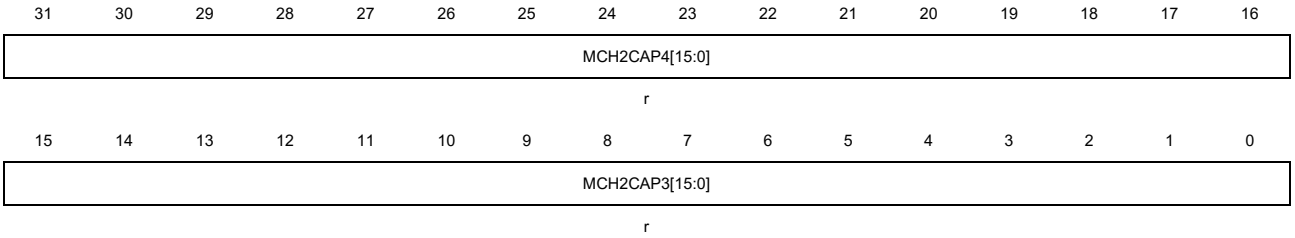
Bits	Fields	Descriptions
31:16	MCH1CAP4[15:0]	Multi mode channel 1 captures 4 count values When multi mode channel 1 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the fourth capture edge of the last capture event. **Note**: And this register is read-only.
15:0	MCH1CAP3[15:0]	Multi mode channel 1 captures 3 count values When multi mode channel 1 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the third capture edge of the last capture event. **Note**: And this register is read-only.

Multi mode channel 2 multi-edge capture register(TIMERx_MCH2MECAP)

Address offset: 0xCC

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



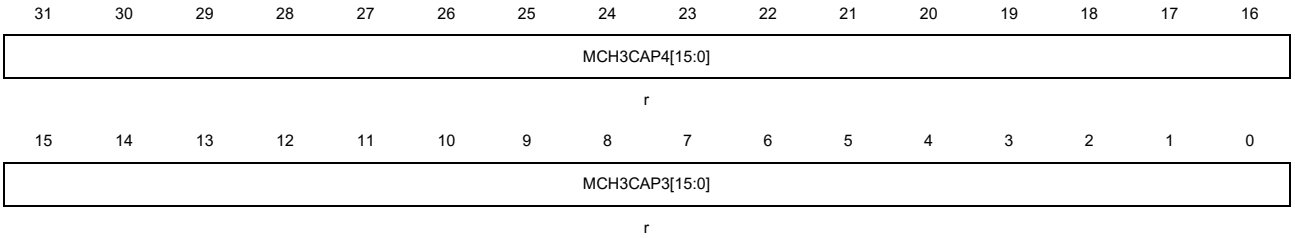
Bits	Fields	Descriptions
31:16	MCH2CAP4[15:0]	Multi mode channel 2 captures 4 count values When multi mode channel 2 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the fourth capture edge of the last capture event. **Note**: And this register is read-only.
15:0	MCH2CAP3[15:0]	Multi mode channel 2 captures 3 count values When multi mode channel 2 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the third capture edge of the last capture event. **Note**: And this register is read-only.

Multi mode channel 3 multi-edge capture register(TIMERx_MCH3MECAP)

Address offset: 0xD0

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:16	MCH3CAP4[15:0]	Multi mode channel 3 captures 4 count values When multi mode channel 3 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the fourth capture edge of the last capture event. **Note**: And this register is read-only.

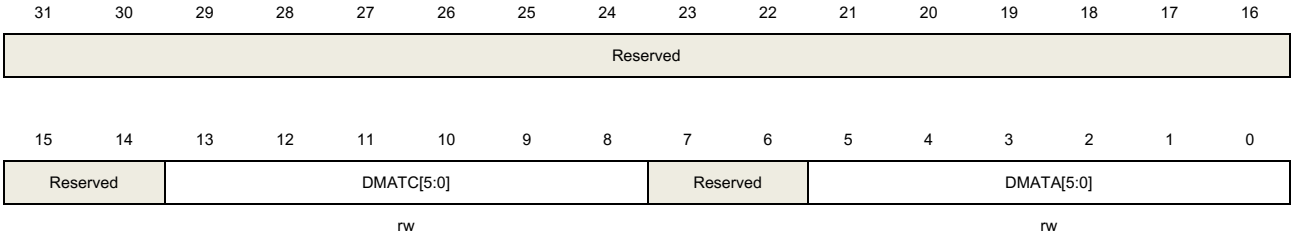
15:0	MCH3CAP3[15:0]	Multi mode channel 3 captures 3 count values When multi mode channel 3 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the third capture edge of the last capture event. **Note**: And this register is read-only.

DMA configuration register (TIMERx_DMACFG)

Address offset: 0xE0

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



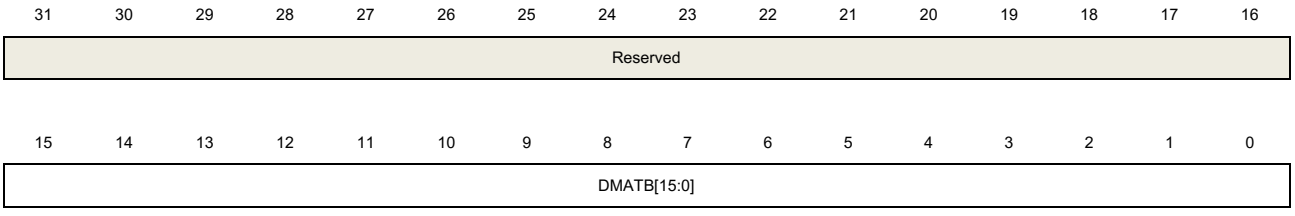
Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13:8	DMATC[5:0]	DMA transfer count This filed defines the number(n) of the register that DMA will access(R/W), n = (DMATC [5:0] +1). DMATC [5:0] is from 6'b000000 to 6'b100010.
7:6	Reserved	Must be kept at reset value.
5:0	DMATA[5:0]	DMA transfer access start address This field defines the start address of accessing the TIMERx_DMATB register by DMA. When the first access to the TIMERx_DMATB register is done, this bit-field specifies the address just accessed. And then the address of the second access to the TIMERx_DMATB register will be (start address + 0x4). In a word: start address = TIMERx_CTL0 + DMATA*4

DMA transfer buffer register (TIMERx_DMATB)

Address offset: 0xE4

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	DMATB[15:0]	DMA transfer buffer When a read or write operation is assigned to this register, the register located at the address ranges from (start address) to (start address + transfer count * 4) will be accessed. The transfer count is calculated by hardware, and ranges from 0 to DMATC.

Configuration register (TIMERx_CFG)

Address offset: 0xFC
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											RDSEL	CCUSEL	Reserved	CHVSEL	OUTSEL
											rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
4	RDSEL	Read selection This bit determines the value of TIMERx_PSC, TIMERx_CAR, TIMERx_CHyCV, TIMERx_MCHyCV, TIMERx_CHyCV_ADD, TIMERx_MCHyCV_ADD when reading the register. 0: Current configuration value 1: The shadow register value in effect
3	CCUSEL	Commutation control shadow register update selection This bit is valid only if the CCUC[2:0] bit fields are configured as 3'b100,3'b101, and 3'b110. 0: The shadow register is updated when the counter generates an overflow/underflow event 1: The shadow register is updated when CREP[7:0] value is 0 and the counter produces an overflow/underflow event.
2	Reserved	Must be kept at reset value.
1	CHVSEL	Write CHxVAL register selection bit This bit-field is set and reset by software.

		1: If the value to be written to the CHxVAL register is the same as the value of CHxVAL register, the write access is ignored. 0: No effect.
0	OUTSEL	The output value selection bit This bit-field is set and reset by software. 1: If POEN bit and IOS bit are 0, the output is disabled. 0: No effect.

17.2. General level0 timer (TIMERx, x=1, 2)

17.2.1. Overview

The general level0 timer module (TIMER1, TIMER2) is a four-channel timer that supports input capture and output compare. They can generate PWM signals to control motor or be used for power management applications. The general level0 timer has a 16-bit counter that can be used as an unsigned counter.

In addition, the general level0 timers can be programmed and be used for counting, their external events can be used to drive other timers.

Timers are completely independent with each other, but they may be synchronized to provide a larger timer with their counter value increasing in unison.

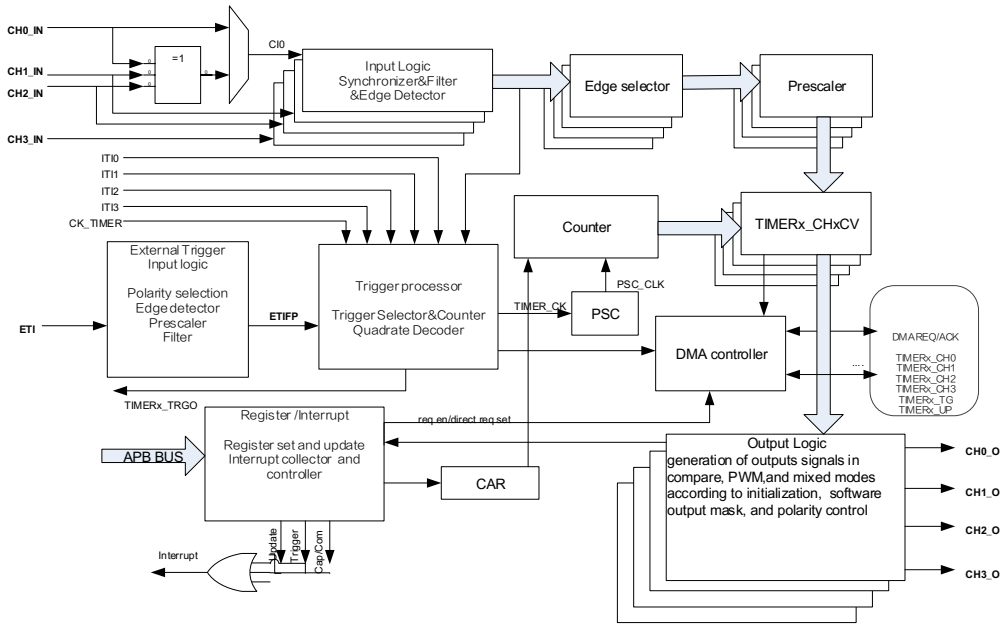
17.2.2. Characteristics

- Total channel num: 4.
- Counter width: 16 bits.
- Selectable clock source: internal clock, internal trigger, external input, external trigger.
- Multiple counter modes: up counting, down counting and center-aligned counting.
- Quadrature decoder: used for motion tracking and determination of both rotation direction and position.
- Hall sensor function: used for 3-phase motor control.
- Programmable prescaler: 16 bits. The factor can be changed ongoing.
- Each channel is user-configurable: input capture mode, output compare mode, programmable PWM mode and single pulse mode.
- Auto reload function.
- Interrupt output or DMA request: update event, trigger event and compare/capture event.
- Daisy chaining of timer module allows a single timer to start multiple timers.
- Timer synchronization allows the selected timers to start counting on the same clock cycle.
- Timer master- slave management.

17.2.3. Block diagram

Figure 17-51. General Level 0 timer block diagram provides details on the internal configuration of the general level 0 timer.

Figure 17-51. General Level 0 timer block diagram



17.2.4. Function overview

Clock source configuration

The clock source of the timer can be either the CK_TIMER or an alternate clock source controlled by TSCFGy[4:0] (y=0..11,15) in SYSCFG_TIMERxCFG (x=1, 2)

if TSCFGy[4:0] (y=0..2,6,8,9,10,11) !=5'b00000 in SYSCFG_TIMERxCFG (x=1, 2), the prescaler is clocked by other clock sources selected in the TSCFGy[4:0] register, more details will be introduced later. When the TSCFGy[4:0] (y=3,4,5,7) are setting to an available value, the internal clock TIMER_CK is the counter prescaler driving clock source.

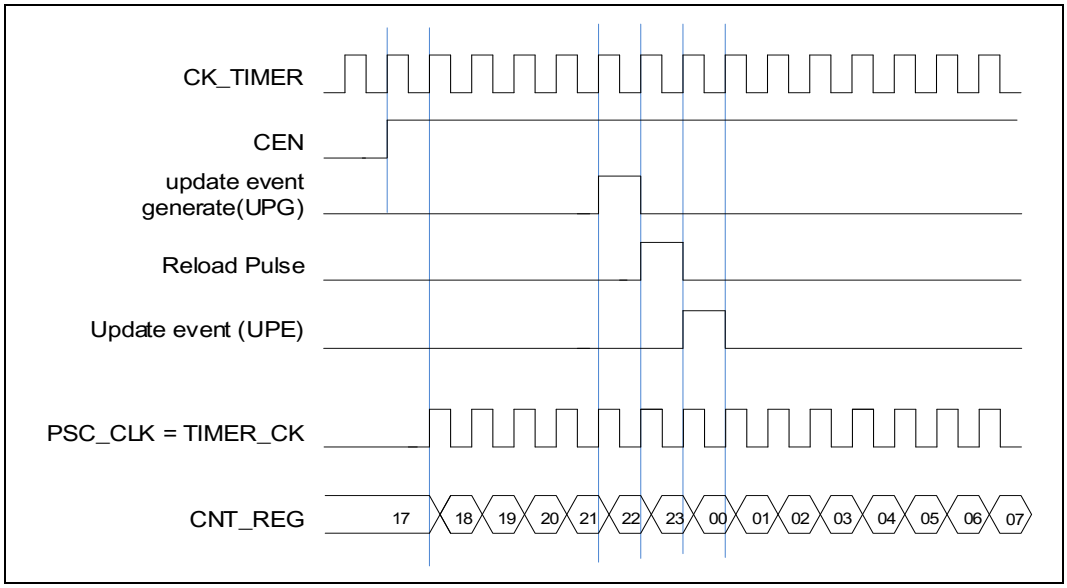
Internal clock mode

- TSCFGy[4:0] = 5'b00000 in SYSCFG_TIMERxCFG(y=0..11,15) Internal clock CK_TIMER is selected as timer clock source which is from module RCU.

The default clock source is the CK_TIMER for driving the counter prescaler when TSCFGy[4:0] (y=0..11,15).= 5'b00000 in SYSCFG_TIMERxCFG(x=1, 2) When the CEN is set, the CK_TIMER will be divided by PSC value to generate PSC_CLK.

In this mode, the TIMER_CK which drives counter's prescaler to count is equal to CK_TIMER which is from RCU module.

Figure 17-52. Timing chart of internal clock divided by 1



External clock mode

■ TSCFG6[4:0] are setting to an available value (external clock mode 0). External input pin is selected as timer clock source.

The TIMER_CK, which drives counter's prescaler to count, can be triggered by the event of rising or falling edge on the external pin TIMERx_CH0/TIMERx_CH1. This mode can be selected by setting TSCFG6[4:0] to 0x5, 0x6, 0x7.

And, the counter prescaler can also be driven by rising edge on the internal trigger input pin ITI0..9. This mode can be selected by setting TSCFG6[4:0] to 0x1~0x9.

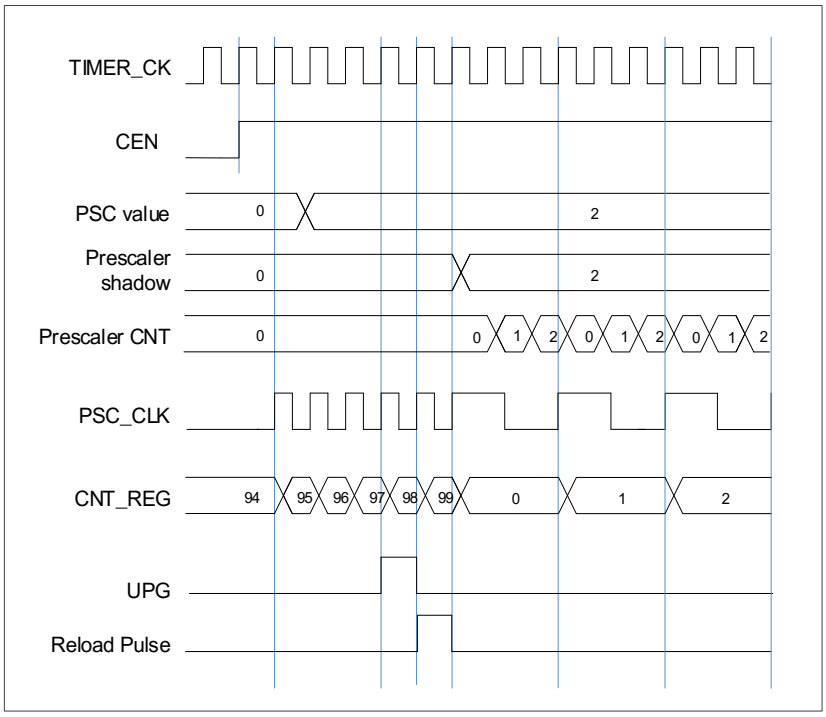
■ SMC1= 1'b1 (external clock mode 1). External input ETI is selected as timer clock source.

The TIMER_CK, which drives counter's prescaler to count, can be triggered by the event of rising or falling edge on the external pin ETI. This mode can be selected by setting the SMC1 bit in the TIMERx_SMCFG register to 1. The other way to select the ETI signal as the clock source is setting TSCFG6[4:0] to 0x18. Note that the ETI signal is derived from the ETI pin sampled by a digital filter. When the ETI signal is selected as the clock source, the trigger controller including the edge detection circuitry will generate a clock pulse on each ETI signal rising edge to clock the counter prescaler.

Clock prescaler

The counter clock (PSC_CK) is obtained by the TIMER_CK through the prescaler, and the prescale factor can be configured from 1 to 65536 through the prescaler register (TIMERx_PSC). The new written prescaler value will not take effect until the next update event.

Figure 17-53. Timing chart of PSC value change from 0 to 2



Counter up counting

In this mode, the counter counts up continuously from 0 to the counter-reload value, which is defined in the TIMERx_CAR register, in a count-up direction. Once the counter reaches the counter reload value, the counter will start counting up from 0 again. The update event is generated at each counter overflow. The counting direction bit DIR in the TIMERx_CTL0 register should be set to 0 for the up counting mode.

Whenever, if the update event software trigger is enabled by setting the UPG bit in the TIMERx_SWEVG register, the counter value will be initialized to 0 and an update event will be generated.

If the UPDIS bit in TIMERx_CTL0 register is set, the update event is disabled.

When an update event occurs, all the shadow registers (counter auto reload register, prescaler register) are updated.

Figure 17-54. Timing chart of up counting mode, PSC=0/2 and *Figure 17-55. Timing chart of up counting, change TIMERx_CAR on the go* show some examples of the counter behavior for different clock prescaler factor when TIMERx_CAR=0x63.

Figure 17-54. Timing chart of up counting mode, PSC=0/2

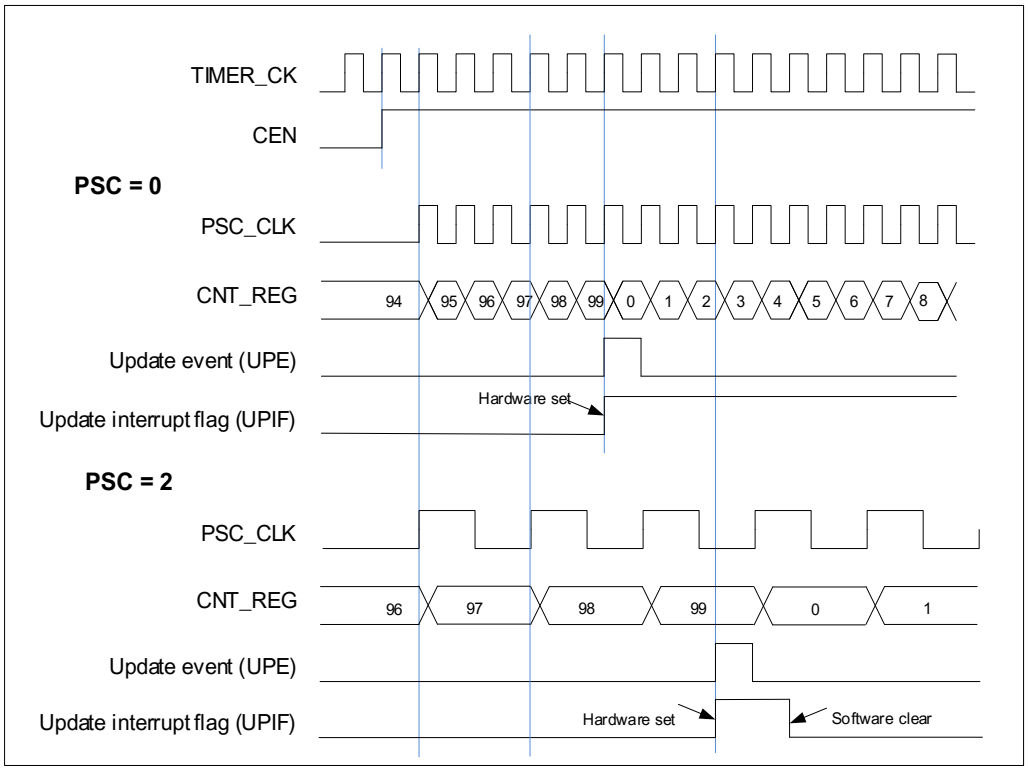
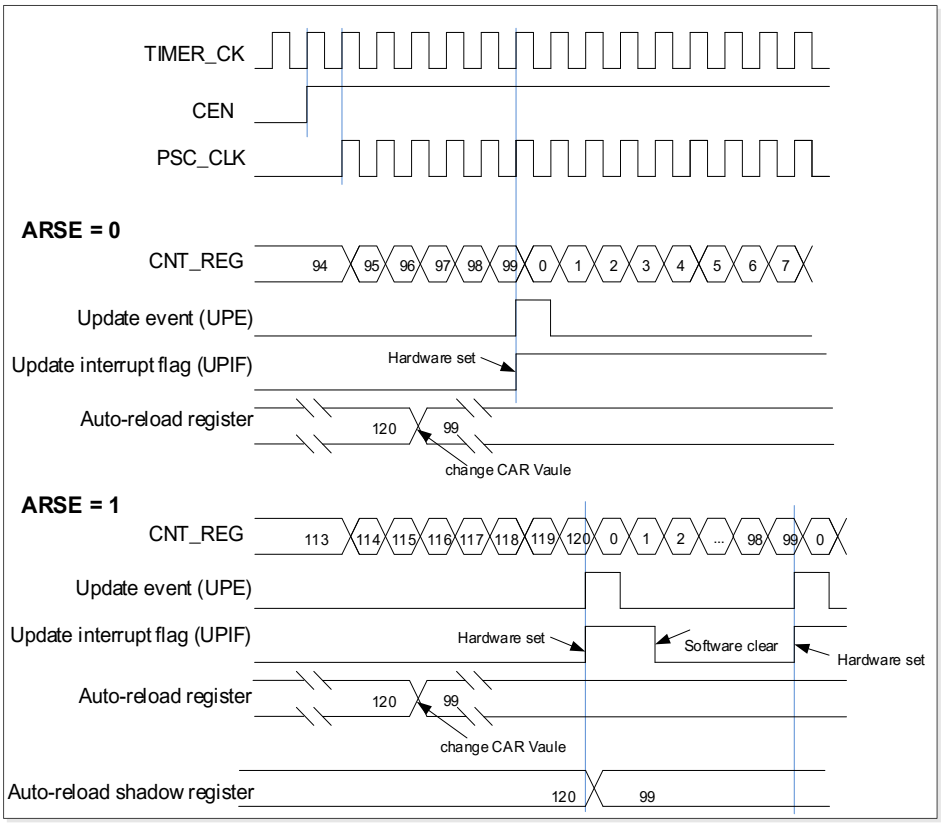


Figure 17-55. Timing chart of up counting, change TIMERx_CAR on the go



Counter down counting

In this mode, the counter counts down continuously from the counter-reload value, which is defined in the TIMERx_CAR register, to 0 in a count-down direction. Once the counter reaches to 0, the counter will start counting down from the counter-reload value. The update event is generated at each counter underflow. The counting direction bit DIR in the TIMERx_CTL0 register should be set to 1 for the down-counting mode.

When the update event is set by the UPG bit in the TIMERx_SWEVG register, the counter value will be initialized to the counter reload value and an update event will be generated.

If the UPDIS bit in TIMERx_CTL0 register is set, the update event is disabled.

When an update event occurs, all the shadow registers (counter autoreload register, prescaler register) are updated.

Figure 17-56. Timing chart of down counting mode, PSC=0/2 and *Figure 17-57. Timing chart of down counting mode, change TIMERx_CAR on the go* show some examples of the counter behavior for different clock frequencies when TIMERx_CAR = 0x63.

Figure 17-56. Timing chart of down counting mode, PSC=0/2

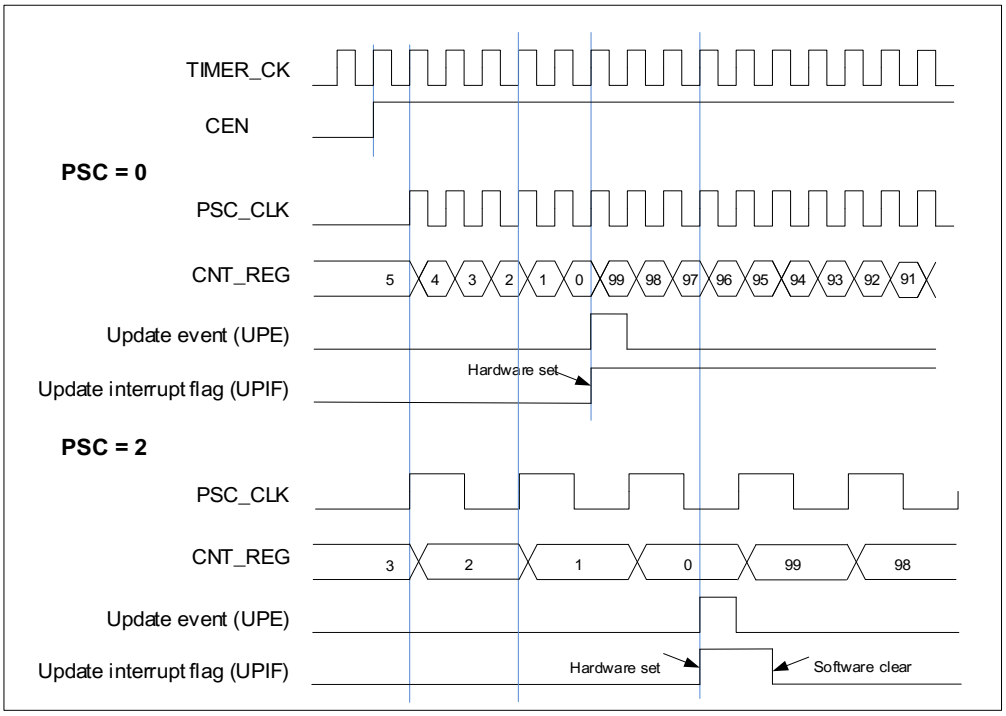
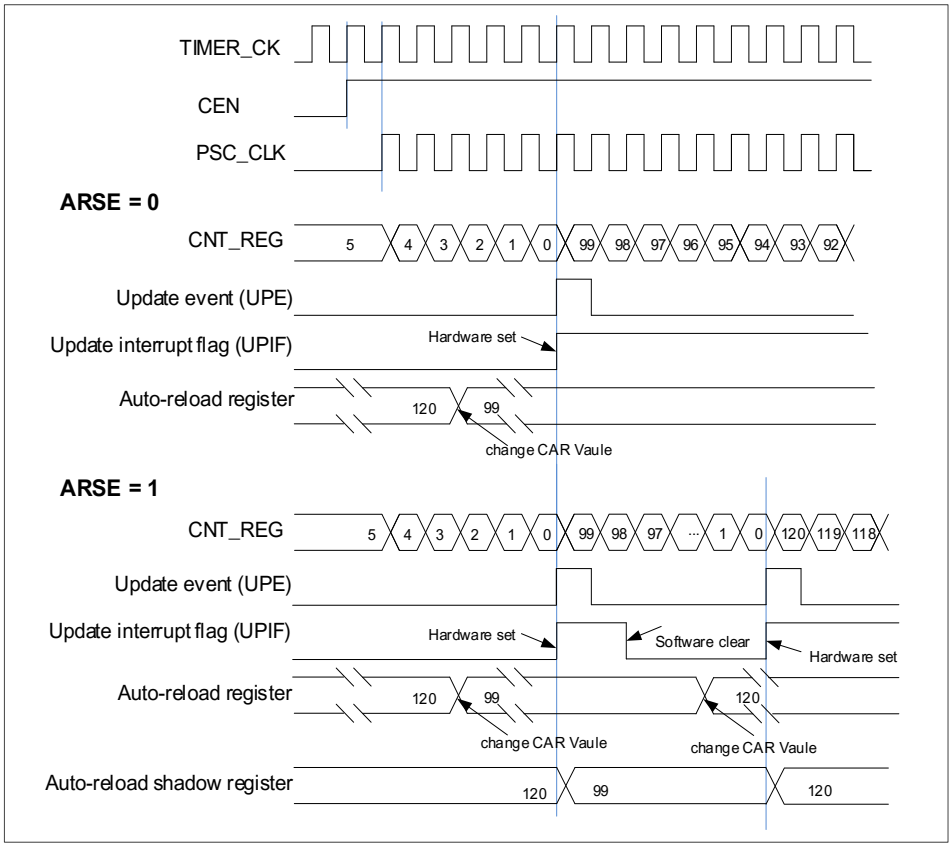


Figure 17-57. Timing chart of down counting mode, change TIMERx_CAR on the go



Counter center-aligned counting

In the center-aligned counting mode, the counter counts up from 0 to the counter-reload value and then counts down to 0 alternatively. The TIMER module generates an overflow event when the counter counts to the counter-reload value subtract 1 in the up-counting mode and generates an underflow event when the counter counts to 1 in the down-counting mode. The counting direction bit DIR in the TIMERx_CTL0 register is read-only and indicates the counting direction when in the center-aligned mode.

Setting the UPG bit in the TIMERx_SWEVG register will initialize the counter value to 0 and generate an update event irrespective of whether the counter is counting up or down in the center-aligned counting mode.

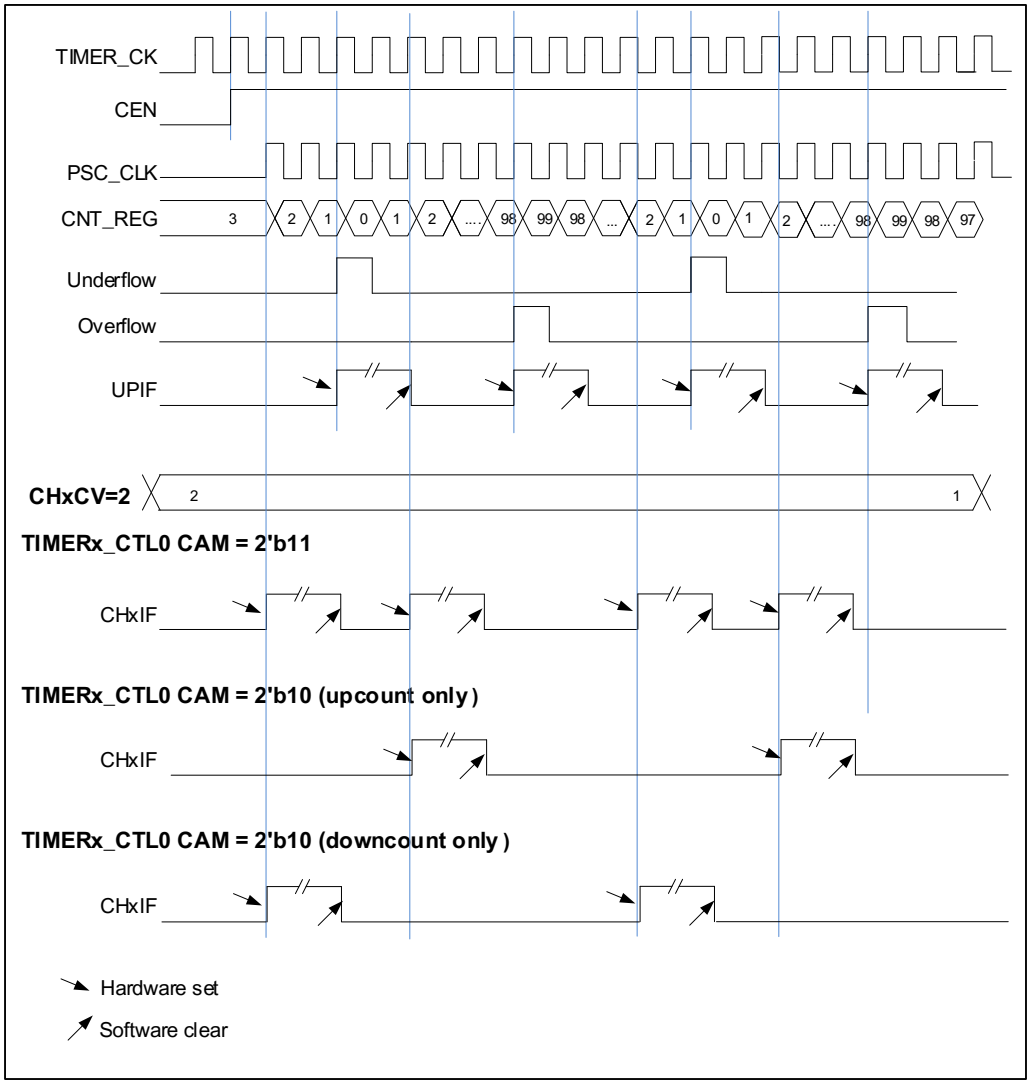
The UPIF bit in the TIMERx_INTF register will be set to 1 either when an underflow event or an overflow event occurs. While the CHxIF bit is associated with the value of CAM in TIMERx_CTL0. The details refer to ***Figure 17-58. Timing chart of center-aligned counting mode***.

If the UPDIS bit in the TIMERx_CTL0 register is set, the update event is disabled.

When an update event occurs, all the shadow registers (counter autoreload register, prescaler register) are updated.

Figure 17-58. Timing chart of center-aligned counting mode shows the example of the counter behavior when TIMERx_CAR=0x99, TIMERx_PSC=0x0.

Figure 17-58. Timing chart of center-aligned counting mode



Input capture and output compare channels

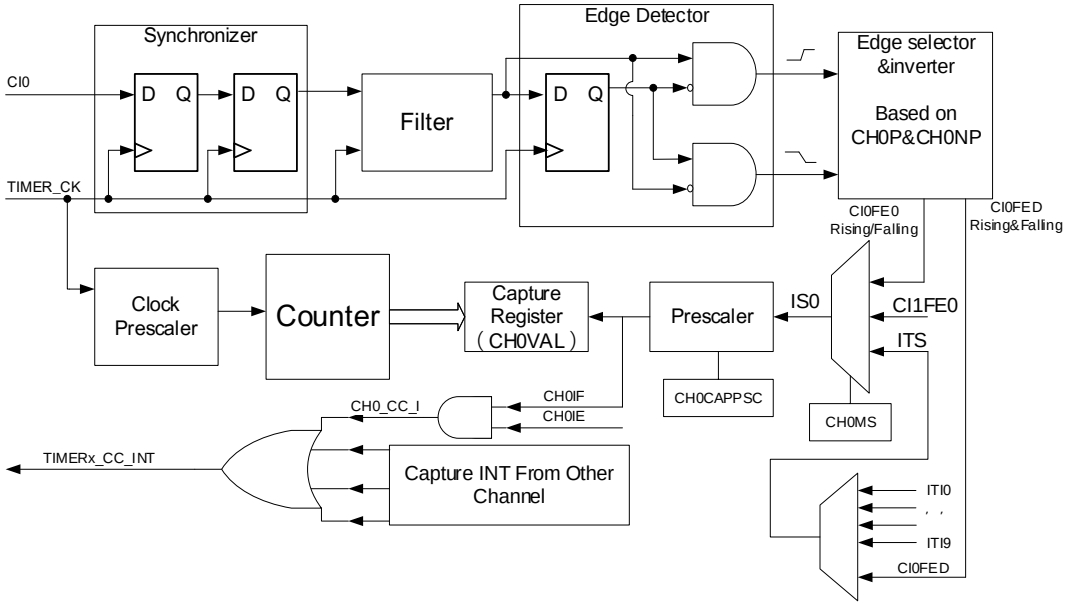
The general level0 Timer has four independent channels which can be used as capture inputs or compare match outputs. Each channel is built around a channel capture compare register including an input stage, channel controller and an output stage.

■ Channel input capture function

Channel input capture function allows the channel to perform measurements such as pulse timing, frequency, period, duty cycle and so on. The input stage consists of a digital filter, a channel polarity selection, edge detection and a channel prescaler. When a selected edge occurs on the channel input, the current value of the counter is captured into the TIMERx_CHxCV register, at the same time the CHxIF bit is set and the channel interrupt is

generated if it is enabled when CHxIE=1.

Figure 17-59. Channels input capture principle



In the diagram, CI0 is the input signal of channel 0, CI0FE0 is the valid signal of channel 0 after passing through the edge selector, CI1FE0 is the valid signal of channel 1 after passing through the edge selector, and ITS is the internal trigger source input.

The input signals of channelx (CIx) can be the TIMERx_CHx signal or the XOR signal of the TIMERx_CH0, TIMERx_CH1 and TIMERx_CH2 signals (just for CI0). First, the input signal of channel (CIx) is synchronized to TIMER_CK signal, and then sampled by a digital filter to generate a filtered input signal. Then through the edge detector, the rising or falling edge is detected by configuring CHxP bit. The input capture signal can also be selected from the input signal of other channel or the internal trigger signal by configuring CHxMS bits. The IC prescaler makes several input events generate one effective capture event. On the capture event, TIMERx_CHxCV will store the value of counter.

So, the process can be divided into several steps as below:

Step1: Filter configuration (CHxCAPFLT in TIMERx_CHCTL0).

Based on the input signal and quality of requested signal, configure compatible CHxCAPFLT.

Step2: Edge selection (CHxP/CHxNP in TIMERx_CHCTL2).

Rising edge or falling edge, choose one by configuring CHxP/CHxNP bits.

Step3: Capture source selection (CHxMS in TIMERx_CHCTL0)

As soon as selecting one input capture source by CHxMS, the channel must be set to input mode (CHxMS! =0x0) and TIMERx_CHxCV cannot be written any more.

Step4: Interrupt enable (CHxIE and CHxDEN in TIMERx_DMAINTEN)

Enable the related interrupt to get the interrupt and DMA request.

Step5: Capture enable (CHxEN in TIMERx_CHCTL2)

Result: When the wanted input signal is captured, TIMERx_CHxCV will be set by counter's value and CHxIF is asserted. If the CHxIF is 1, the CHxOF will also be asserted. The interrupt and DMA request will be asserted or not based on the configuration of CHxIE and CHxDEN in TIMERx_DMAINTEN.

Direct generation: A DMA or interrupt request is generated by setting CHxG directly.

The channel input capture function can be also used for pulse width measurement from signals on the TIMERx_CHx pins. For example, PWM signal connects to CI0 input. Select CI0 as channel 0 capture signals by setting CH0MS to 2'b01 in the channel control register (TIMERx_CHCTL0) and set capture on rising edge. Select CI0 as channel 1 capture signal by setting CH1MS to 2'b10 in the channel control register (TIMERx_CHCTL0) and set capture on falling edge. The counter is set to restart mode and is restarted on channel 0 rising edge. Then the TIMERX_CH0CV can measure the PWM period and the TIMERx_CH1CV can measure the PWM duty cycle.

■ **Channel output compare function**

Figure 17-60. Channel output compare principle (x=0,1,2,3)

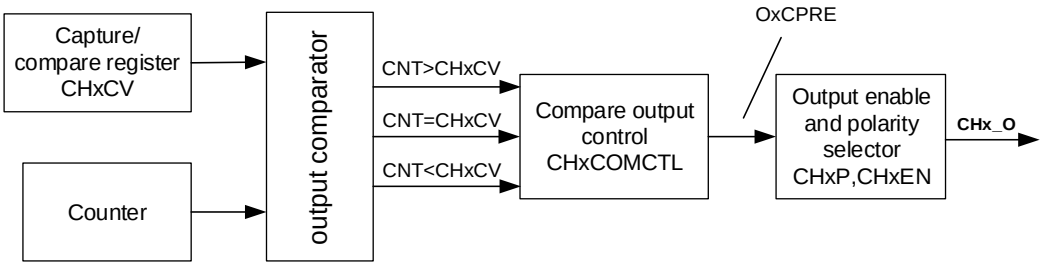


Figure 17-60. Channel output compare principle (x=0,1,2,3) shows the principle circuit of channels output compare function. The relationship between the channel output signal CHx_O and the OxCPRE signal (more details refer to *Channel output prepare signal*) is described as blew: The active level of O0CPRE is high, the output level of CH0_O depends on OxCPRE signal, CHxP bit and CH0P bit (please refer to the TIMERx_CHCTL2 register for more details).For example, configure CHxP=0 (the active level of CHx_O is high, the same as OxCPRE), CHxEN=1 (the output of CHx_O is enabled),

If the output of OxCPRE is active(high) level, the output of CHx_O is active(high) level;

If the output of OxCPRE is inactive(low) level, the output of CHx_O is active(low) level.

In channel output compare function, the TIMERx can generate timed pulses with programmable position, polarity, duration and frequency. When the counter matches the value in the TIMERx_CHxCV register of an output compare channel, the channel (n) output can be set, cleared, or toggled based on CHxCOMCTL. When the counter reaches the value in the TIMERx_CHxCV register, the CHxIF bit will be set and the channel (n) interrupt is generated if CHxIE = 1. And the DMA request will be asserted, if CHxDEN=1.

So, the process can be divided into several steps as below:

Step1: Clock configuration. Such as clock source, clock prescaler and so on.

Step2: Compare mode configuration.

- Set the shadow enable mode by CHxCOMSEN.
- Set the output mode (set/clear/toggle) by CHxCOMCTL.
- Select the active polarity by CHxP.
- Enable the output by CHxEN.

Step3: Interrupt/DMA-request enables configuration by CHxIE/CHxDEN.

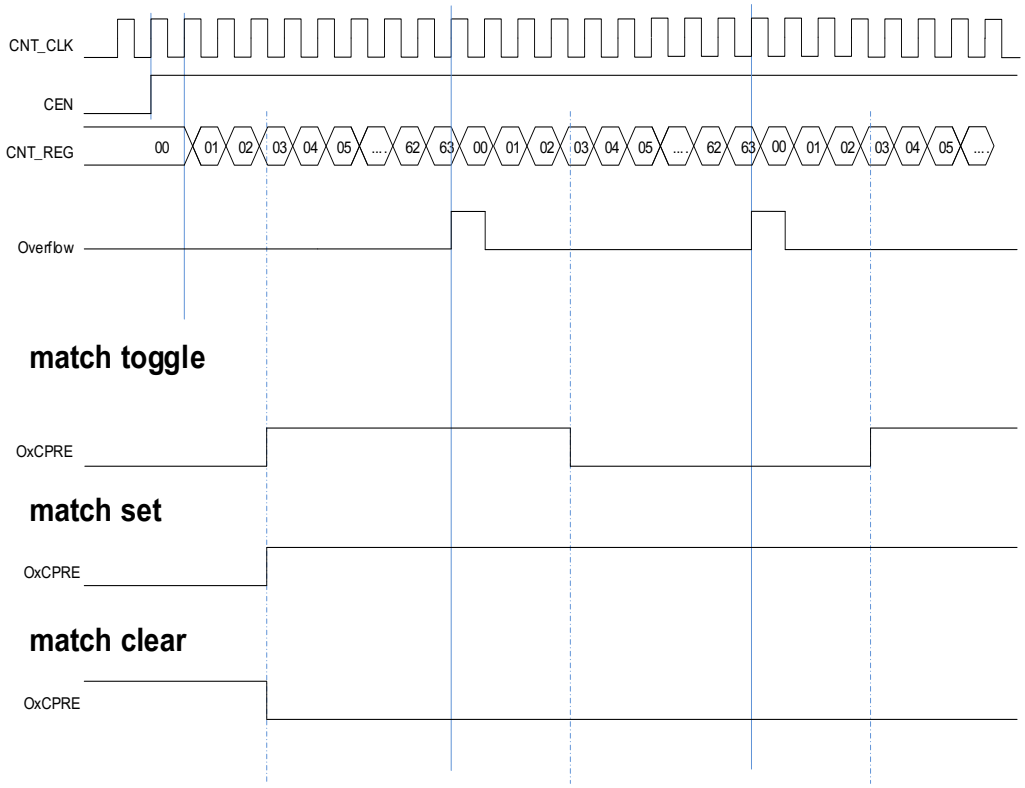
Step4: Compare output timing configuration by TIMERx_CAR and TIMERx_CHxCV.

The TIMERx_CHxCV can be changed onging to meet the expected waveform.

Step5: Start the counter by configuring CEN to 1.

Figure 17-61. Output-compare under three modes shows the three compare modes toggle/set/clear. CAR=0x63, CHxVAL=0x3

Figure 17-61. Output-compare under three modes



Channel multi-edge input capture function

In some input capture scenarios, it is necessary to capture the multi-edge of input signals, such as the measurement of input signal period and input signal duty cycle. The eight channels of advanced timer provide multi-edge capture in order to meet these application scenarios.

The prerequisite for enabling multi-edge capture is that the channel is in input mode, (CHxMS != 3 'b000, x=0,1,2,3), The CHxMECEN and CHxMECP bits of the multi-edge

capture configuration register TIMERx_CHMECAPCFG are configured to enable the corresponding channel (TIMERx_CHx , x=0,1,2,3) and edge capture polarity of the input signal. Configuring CHxMEBL determines the level of buffer captured by the current channel.The TIMERx_CHxCV and TIMERx_CHxCV_ADD registers record the counter value captured for the first time and the counter value captured for the second time.The third and fourth captured counters are recorded in TIMERx_CHxMECAP.

Output PWM function

In the PWM output mode (by setting the CHxCOMCTL bit to 3'b110 (PWM mode 0) or to 3'b 111(PWM mode 1)), the channel can generate PWM waveform according to the TIMERx_CAR registers and TIMERx_CHxCV registers.

Based on the counter mode, PWM can also be divided into EAPWM (Edge-aligned PWM) and CAPWM (Center-aligned PWM).
The EAPWM's period is determined by TIMERx_CAR and the duty cycle is determined by TIMERx_CHxCV. ***Figure 17-62. EAPWM timechart*** shows the EAPWM output and interrupts waveform.

The CAPWM period is determined by 2*TIMERx_CAR, and duty cycle is determined by 2*TIMERx_CHxCV. ***Figure 17-63. CAPWM timechart*** shows the CAPWM output and interrupts waveform.

In up counting mode, if the value of TIMERx_CHxCV is greater than the value of TIMERx_CAR, the output will be always active in PWM mode 0 (CHxCOMCTL=3'b110). And if the value of TIMERx_CHxCV is greater than the value of TIMERx_CAR, the output will be always inactive in PWM mode 1 (CHxCOMCTL=3'b111).

Figure 17-62. EAPWM timechart

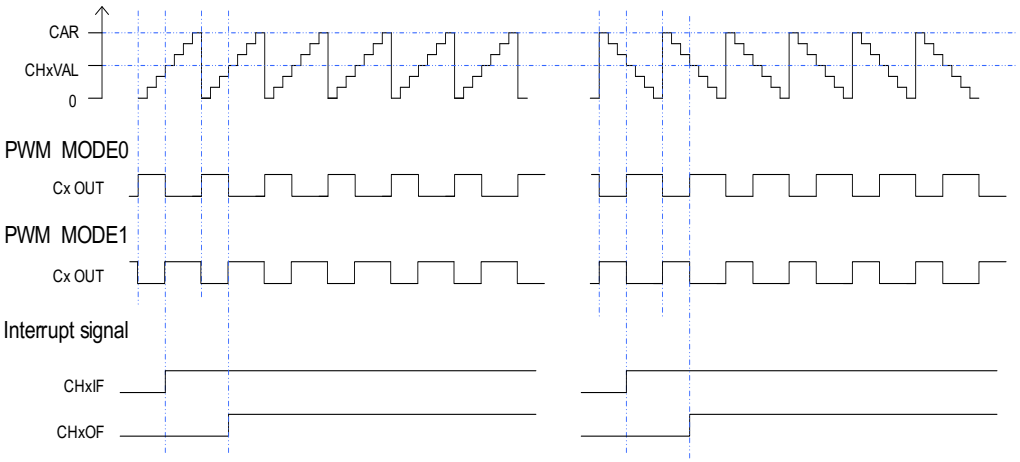
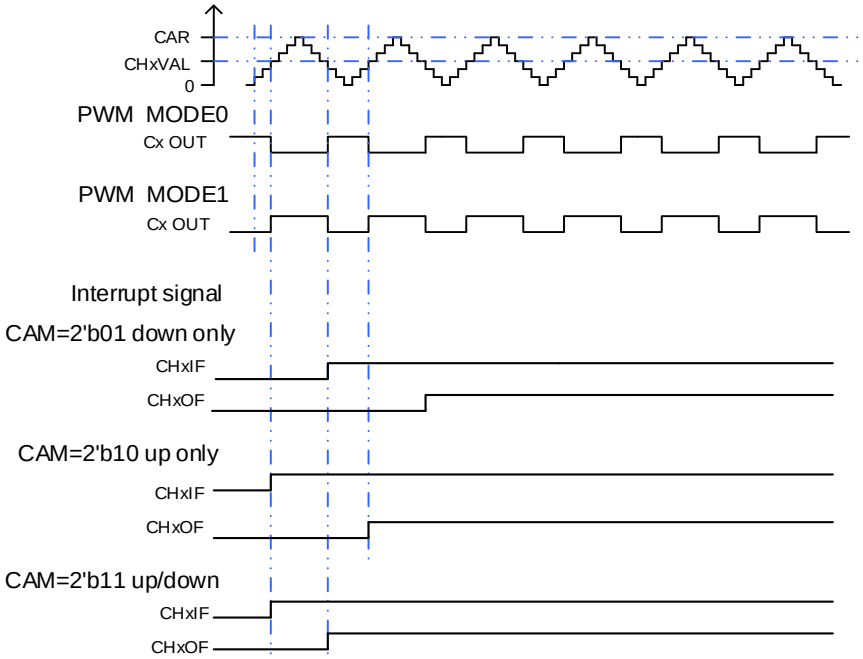


Figure 17-63. CAPWM timechart



Composite PWM mode

In the composite PWM mode (CHxMS[2:0] = 3'b000 and CHxCOMCTL = 4'b1000or 4'b1001), the PWM signal output in channel x (x=0..3) is composited by CHxVAL and CHxVAL_ADD bits.

If CHxCOMCTL = 4'b1000 (PWM mode 0) and DIR = 1'b0 (up counting mode), or CHxCOMCTL = 4'b1001 (PWM mode 1) and DIR = 1'b1 (Down counting mode), the OxCPRE is forced low when the counter matches the value of CHxVAL. The OxCPRE is forced high when the counter matches the value of CHxVAL_ADD.

If CHxCOMCTL = 4'b1001 (PWM mode 1) and DIR = 1'b0 (up counting mode), or CHxCOMCTL = 4'b1000 (PWM mode 0) and DIR = 1'b1 (down counting mode) , the OxCPRE is forced high when the counter matches the value of CHxVAL. The OxCPRE is forced low when the counter matches the value of CHxVAL_ADD.

When CHxVAL or CHxVAL_ADD = 0 / CARL, the channel x output is specially processed, and by setting the CHxPERFOREN bit in the TIMERx_CHCTL2 register, the OxCPRE output is forced to either high or low level (determined according to the selected composite PWM mode).

The PWM period is determined by (CARL + 0x0001) and the PWM pulse width is determined by the following table ***Table 17-3.The composite PWM pulse width***.

Table 17-11.The composite PWM pulse width

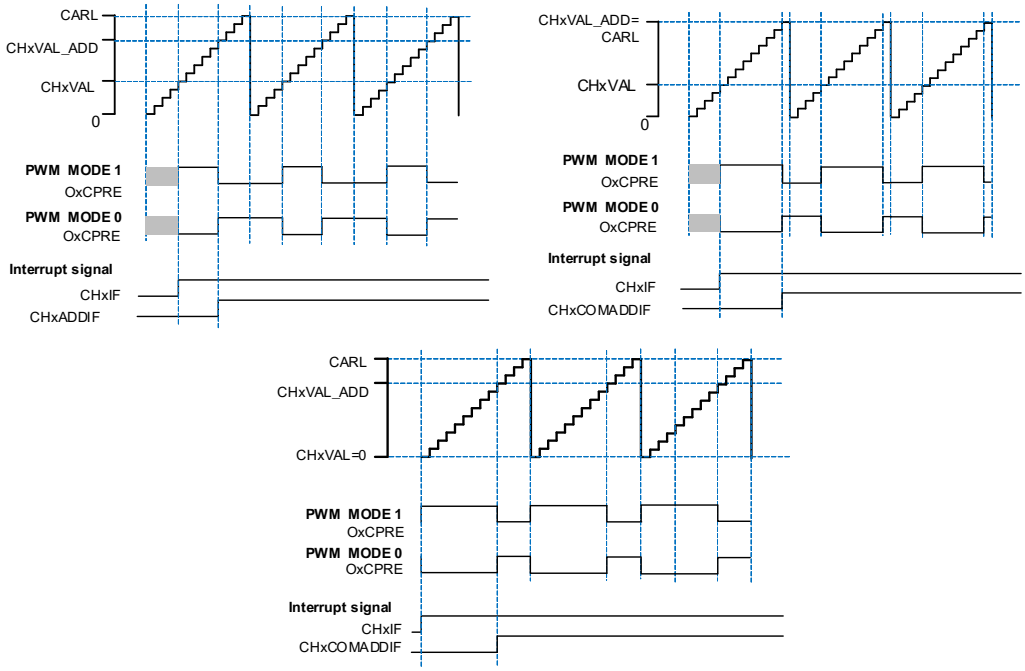
Condition	Mode	PWM pulse width
CHxVAL < CHxVAL_ADD ≤ CARL	Composite PWM mode 0	(CARL + 0x0001) + (CHxVAL - CHxVAL_ADD)
	Composite PWM mode 1	(CHxCVAL_ADD - CHxVAL)
CHxVAL_ADD < CHxVAL ≤ CARL	Composite PWM mode 0	(CHxVAL - CHxVAL_ADD)
	Composite PWM mode 1	(CARL + 0x0001) + (CHxVAL_ADD - CHxVAL)
(CHxVAL = CHxVAL_ADD ≤ CARL) or (CHxVAL > CARL > CHxVAL_ADD)	Composite PWM mode 0 (up counting) or PWM mode 1 (down counting)	100%
	Composite PWM mode 0 (down counting) or PWM mode 1 (up counting)	0%
CHxVAL_ADD > CARL > CHxVAL	composite PWM mode 0(up counting) or composite PWM mode 1(down counting)	0%
	Composite PWM mode 0(down counting) or composite PWM mode 1(up counting)	100%
(CHxVAL>CARL)and (CHxVAL_ADD > CARL)	-	The output of CHx_O is keeping

When the counter reaches the value of CHxVAL, the CHxIF bit is set and the channel x interrupt is generated if CHxIE = 1, and the DMA request will be asserted, if CHxDEN=1. When the counter reaches the value of CHxVAL_ADD, the CHxCOMADDIF bit is set (This flag just used in composite PWM mode and asymmetric PWM mode, when CHxCOMCTL[3:2]=2'b10) and the channel x additional compare interrupt is generated if CHxCOMADDIE = 1 (Only interrupt is generated, no DMA request is generated).

According to the relationship among CHxVAL, CHxVAL_ADD and CARL, it can be divided into four situations:

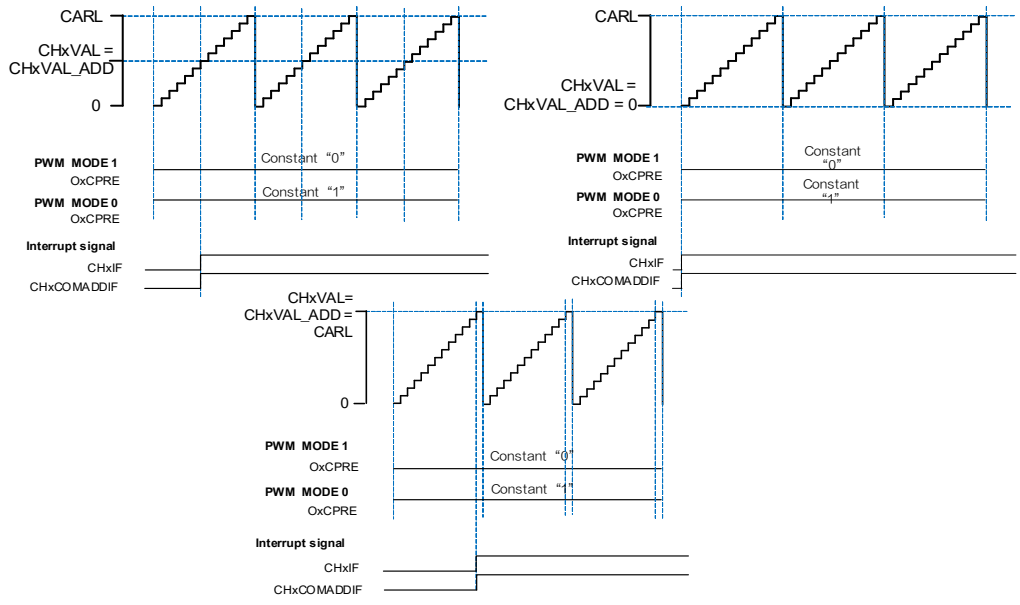
5) CHxVAL < CHxVAL_ADD, and the values of CHxVAL and CHxVAL_ADD between 0 and CARL.

Figure 17-64. Channel x output PWM with (CHxVAL < CHxVAL_ADD)



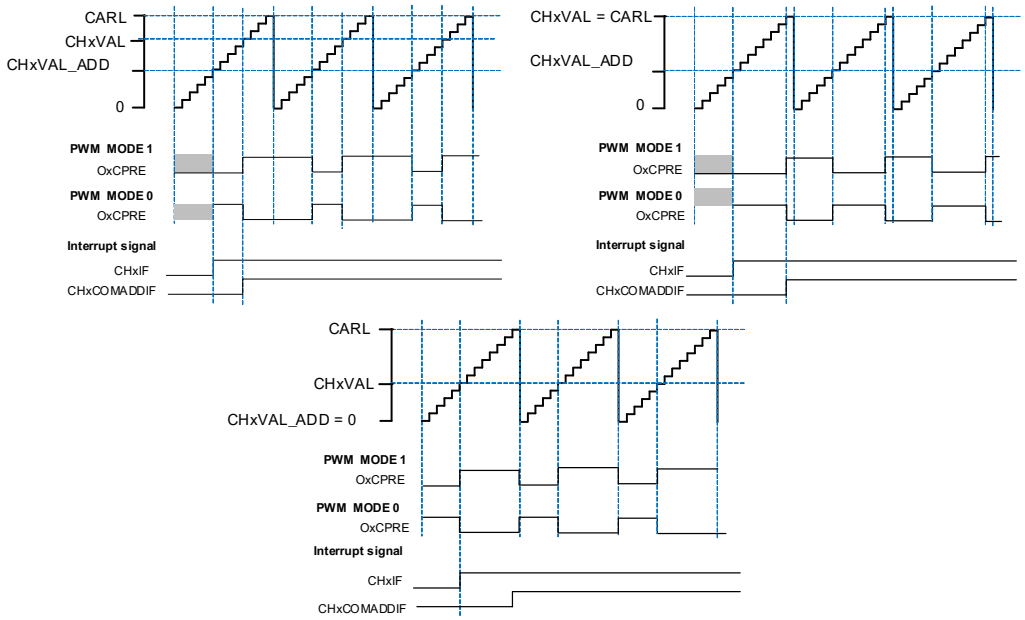
6) CHxVAL = CHxVAL_ADD, and the value of CHxVAL and CHxVAL_ADD between 0 and CARL.

Figure 17-65. Channel x output PWM with (CHxVAL = CHxVAL_ADD)



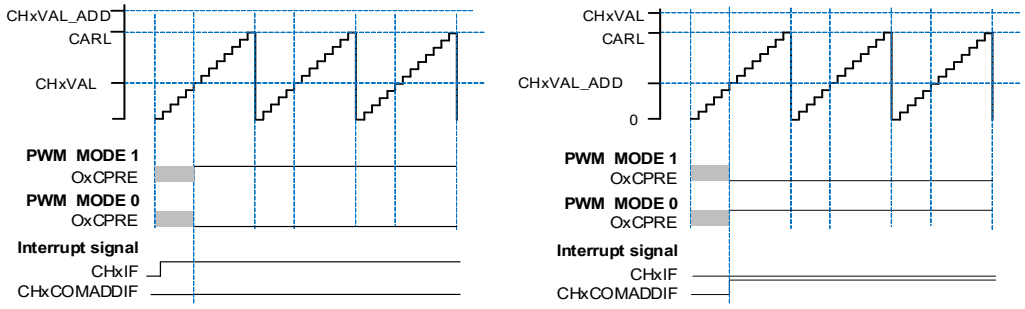
7) CHxVAL > CHxVAL_ADD, and the value of CHxVAL and CHxVAL_ADD between 0 and CARL.

Figure 17-66. Channel x output PWM with (CHxVAL > CHxVAL_ADD)



8) One of the value of CHxVAL and CHxVAL_ADD exceeds CARL.

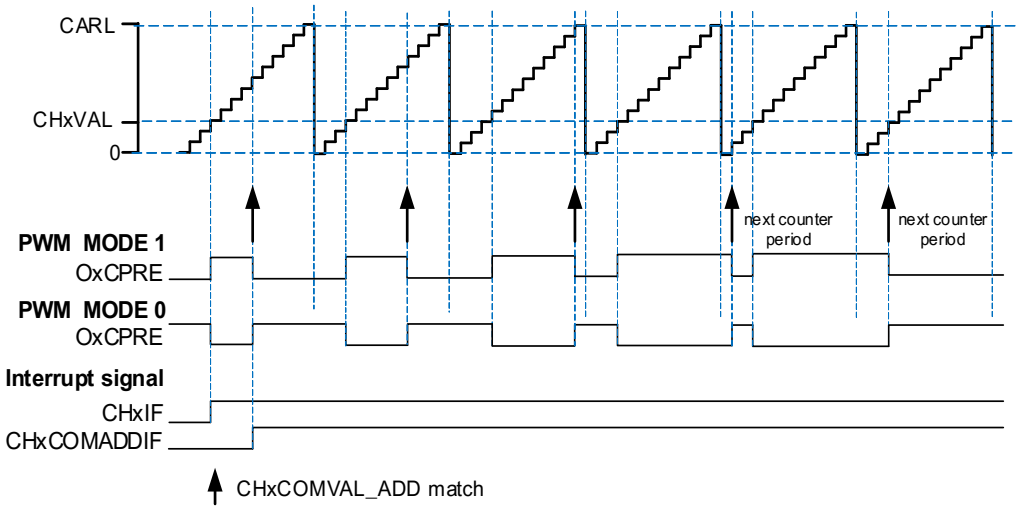
Figure 17-67. Channel x output PWM with CHxVAL or CHxVAL_ADD exceeds CARL



The composite PWM mode is intended to support the generation of PWM signals where the period is not modified while the signal is being generated, but the duty cycle will be varied. ***Figure 17-24. Channel x output PWM duty cycle changing with CHxVAL_ADD*** shows the the PWM output and interrupts waveform.

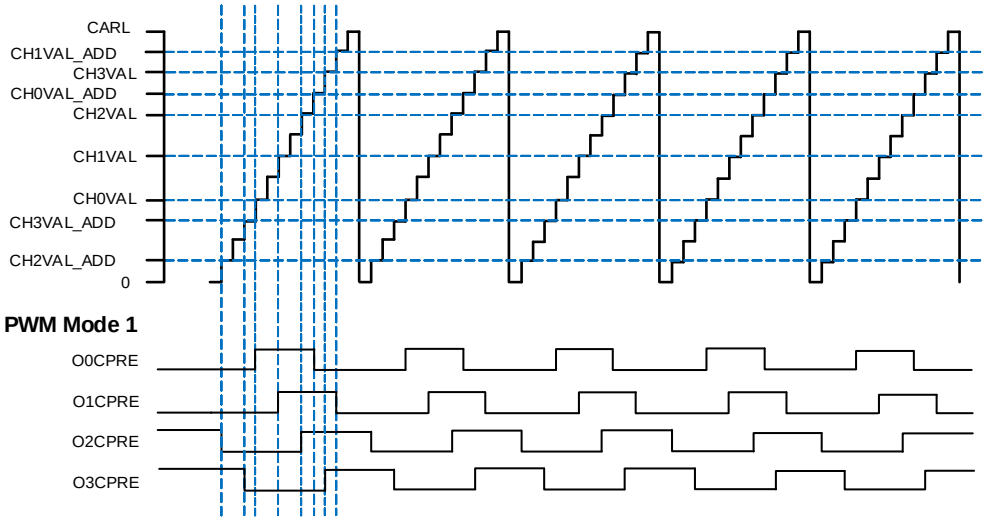
In some cases, the CHxVAL_ADD match can happen on the next counter period (the value of CHxVAL_ADD was written after the counter reaches the value of CHxVAL, and the value of CHxVAL_ADD was less than or equal to the CHxVAL).

Figure 17-68. Channel x output PWM duty cycle changing with CHxVAL_ADD



If more than one channels are configured in composite PWM mode, it is possible to fix an offset for the channel x match edge of each pair with respect to other channels. This behavior is useful in the generation of lighting PWM control signals where it is desirable that edges are not coincident with each other pair to help eliminate noise generation. The CHxVAL register value is the shift of the PWM pulse with respect to the beginning of counter period.

Figure 17-69. Four Channels outputs in composite PWM mode



Asymmetric PWM mode

Asymmetric PWM mode allows two center-aligned PWM signals to be generated with a programmable phase shift. In the asymmetric PWM mode(CHxCOMCTL[3:0]=4'b1010/4'b1011 and CHxMS[2:0] = 3'b000), the PWM signal output in channel x (x=0..3) is composited by CHxVAL and CHxVAL_ADD bits. The asymmetric PWM mode is only available in the center-aligned counting mode.

If CHxCOMCTL = 4'b1010 (asymmetric PWM mode 0) and DIR = 1'b0) (up counting mode), the OxCPRE is forced low when the counter matches the value of CHxVAL.

If CHxCOMCTL = 4'b1010 (asymmetric PWM mode 0) and DIR = 1'b1 (down counting mode),

the OxCPRE is forced high when the counter matches the value of CHxVAL_ADD.

If CHxCOMCTL = 4'b1011 (asymmetric PWM mode 1) and DIR = 1'b0 (up counting mode), the OxCPRE is forced high when the counter matches the value of CHxVAL.

If CHxCOMCTL = 4'b1011 (asymmetric PWM mode 1) and DIR = 1'b1 (down counting mode), the OxCPRE is forced low when the counter matches the value of CHxVAL_ADD.

Compared to composite PWM mode, asymmetric PWM mode is only valid with center-aligned mode, and the CHxVAL is only valid when counting up and CHxVAL_ADD is only valid when counting down.

When CHxVAL or CHxVAL_ADD = 0 / CARL, the channel x output is specially processed, and by setting the CHxPERFOREN bit in the TIMERx_CHCTL2 register, the OxCPRE output is forced to either high or low level (determined according to the selected asymmetric PWM mode).

When CHxVAL =0, the OxCPRE is forced low in asymmetric PWM mode 0. When CHxVAL =0, the OxCPRE is forced high in asymmetric PWM mode 1.

When CHxVAL_ADD =CARL, the OxCPRE is forced high in asymmetric PWM mode 0. When CHxVAL_ADD = CARL, the OxCPRE is forced low in asymmetric PWM mode 1.

When CHxVAL = CARL, the OxCPRE is forced low in asymmetric PWM mode 0. When CHxVAL = CARL, the OxCPRE is forced high in asymmetric PWM mode 1.

When CHxVAL_ADD = 0, the OxCPRE is forced high in asymmetric PWM mode 0. When CHxVAL_ADD = 0, the OxCPRE is forced low in asymmetric PWM mode 1.

The PWM period is determined by (2CARL + 0x0001) and the PWM pulse width is determined by the following *Table 17-4.The asymmetric PWM pulse width*.

Table 17-12.The asymmetric PWM pulse width

Condition	Mode	PWM pulse width
CHxVAL < CHxVAL_ADD ≤ CARL or CHxVAL_ADD < CHxVAL ≤ CARL or CHxVAL = CHxVAL_ADD	Asymmetric PWM mode 0	CHxVAL_ADD + CHxVAL
	Asymmetric PWM mode 1	(2CARL + 0x0001) - CHxVAL_ADD - CHxVAL
CHxVAL > CARL > CHxVAL_ADD	Asymmetric PWM mode 0	100%
	Asymmetric PWM mode 1	0%
CHxVAL_ADD > CARL > CHxVAL	Asymmetric PWM mode 0	0%
	Asymmetric PWM mode 1	100%
(CHxVAL>CARL)and (CHxVAL_ADD > CARL)	-	The output of CHx_O is keeping

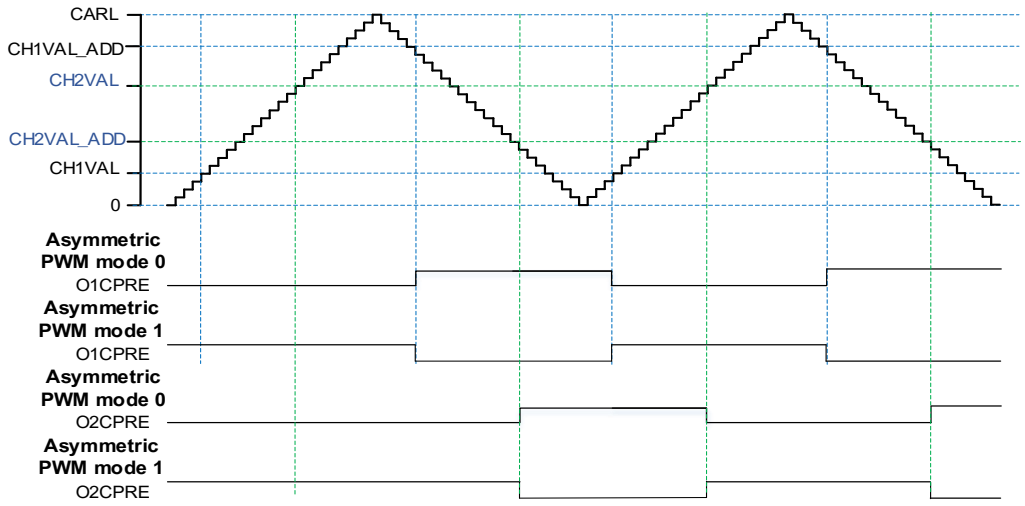
When the counter reaches the value of CHxVAL, the CHxIF bit is set and the channel x interrupt is generated if CHxIE = 1, and the DMA request will be asserted, if CHxDEN=1. When the counter reaches the value of CHxVAL_ADD, the CHxCOMADDIF bit is set (this flag just used in composite PWM mode and asymmetric PWM mode, when CHxCOMCTL[3:2]= 2'b10) and the channel x additional compare interrupt is generated if CHxCOMADDIE = 1 (Only interrupt is generated, no DMA request is generated).

According to the relationship among CHxVAL, CHxVAL_ADD and CARL, it can be divided into four situations:

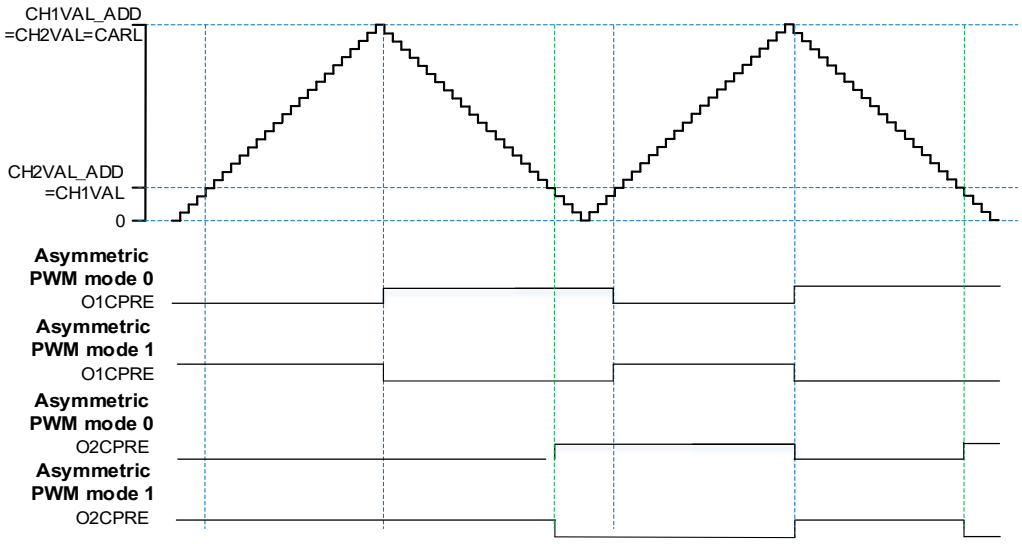
4） CHxVAL < CHxVAL_ADD, and the values of CHxVAL and CHxVAL_ADD between 0 and CARL.

Figure 17-70. Channel x output PWM with(CHxVAL < CHxVAL_ADD / CHxVAL_ADD < CHxVAL)

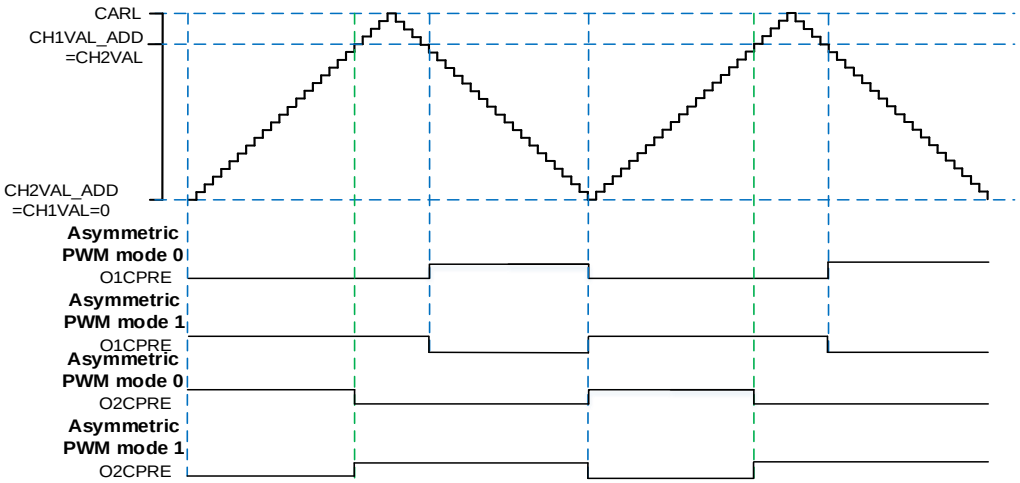
d. 0 < CH1VAL < CH1VAL_ADD < CARL / 0 < CH2VAL_ADD < CH2VAL < CARL



e. 0 < CH1VAL < CH1VAL_ADD = CARL / 0 < CH2VAL_ADD < CH2VAL = CARL, and CH1VAL = CH2VAL_ADD



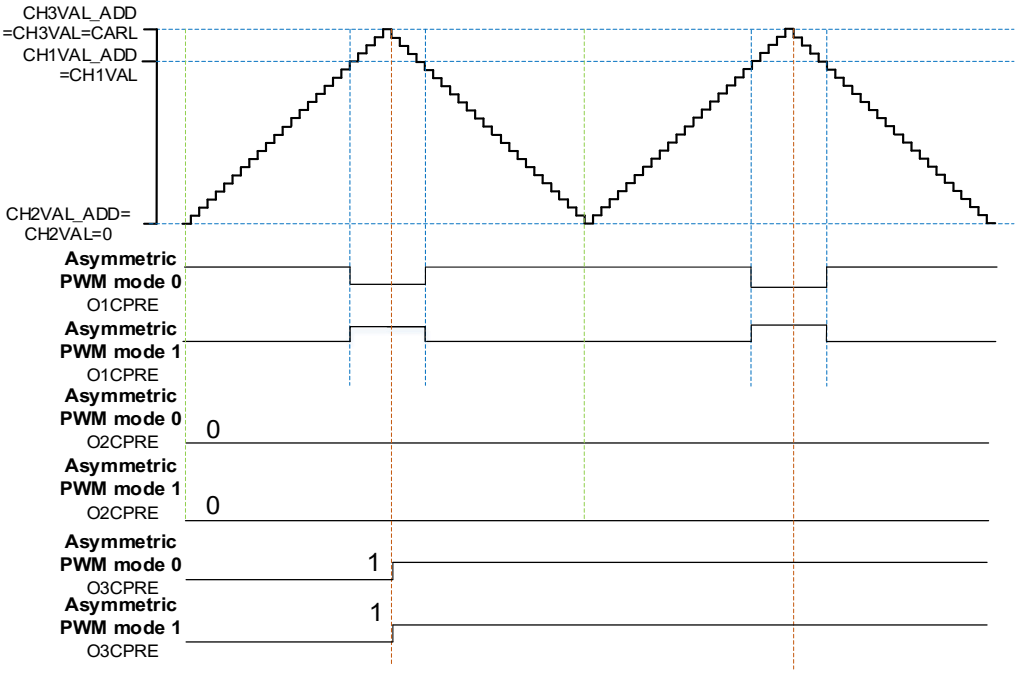
f. 0 = CH1VAL < CH1VAL_ADD < CARL / 0 = CH2VAL_ADD < CH2VAL < CARL, and CH1VAL_ADD = CH2VAL



5）CHxVAL = CHxVAL_ADD, and the value of CHxVAL and CHxVAL_ADD between 0 and CARL.

Figure 17-71. Channel x output PWM with(CHxVAL = CHxVAL_ADD)

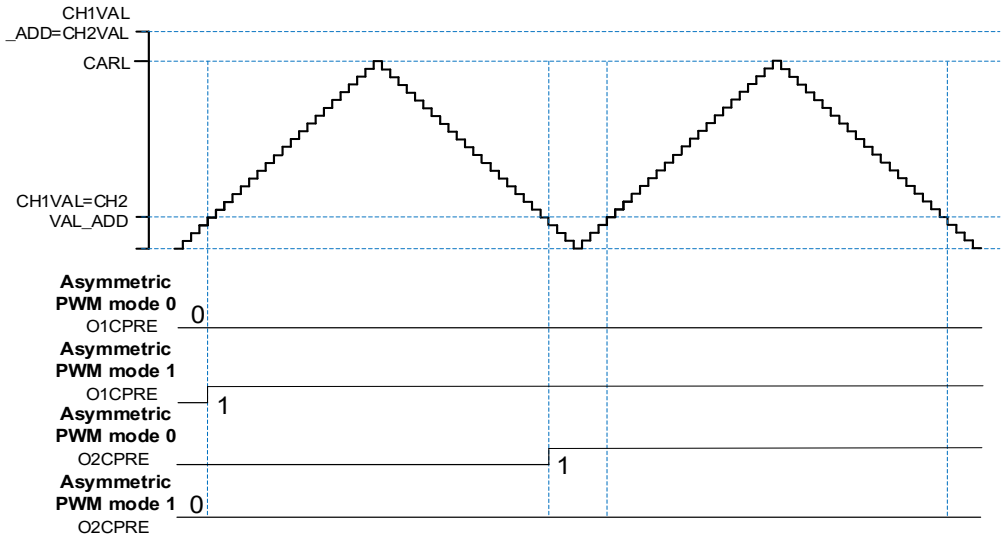
0 < CH1VAL = CH1VAL_ADD < CARL / CH2VAL = CH2VAL_ADD = 0 / CH3VAL = CH3VAL_ADD = CARL



6）One of the value of CHxVAL and CHxVAL_ADD exceeds CARL.

Figure 17-72. Channel x output PWM with CHxVAL or CHxVAL_ADD exceeds CARL

0 < CH1VAL < CARL < CH1VAL_ADD / 0 < CH2VAL_ADD < CARL < CH2VAL



Channel output prepare signal

As is shown in *Figure 17-60. Channel output compare principle (x=0,1,2,3)*, when TIMERx is configured in compare match output mode,a middle signal which is OxCPRE signal (Channel x output prepare signal) will be generated before the channel outputs signal. The OxCPRE signal type is defined by configuring the CHxCOMCTL bit. The OxCPRE signal has several types of output function. These include keeping the original level by configuring the CHxCOMCTL field to 0x00, setting to high by configuring the CHxCOMCTL field to 0x01,

setting to low by configuring the CHxCOMCTL field to 0x02 or toggling signal by configuring the CHxCOMCTL field to 0x03 when the counter value matches the content of the TIMERx_CHxCV register.

The PWM mode 0/PWM mode 1 output is another output type of OxCPRE which is setup by configuring the CHxCOMCTL field to 0x06/0x07. In these modes, the OxCPRE signal level is changed according to the counting direction and the relationship between the counter value and the TIMERx_CHxCV content. Refer to the definition of relative bit for more details.

Another special function of the OxCPRE signal is a forced output which can be achieved by configuring the CHxCOMCTL field to 0x04/0x05. The output can be forced to an inactive/active level irrespective of the comparison condition between the values of the counter and the TIMERx_CHxCV.

Configure the CHxCOMCEN bit to 1 in the TIMERx_CHCTL0 register, the OxCPRE signal can be forced to 0 when the ETIFP signal derived from the external ETI pin is set to a high level. When the next comparison result changes after the ETIFP signal becomes invalid, the OxCPRE signal will immediately return to the active level.

Break mode

Refer to ***Break mode***.

Note: The break sources of general level0 timer only have internal signals.

Quadrature decoder

Refer to ***Quadrature decoder***.

Decoder

Refer to ***Decoder***.

Hall sensor function

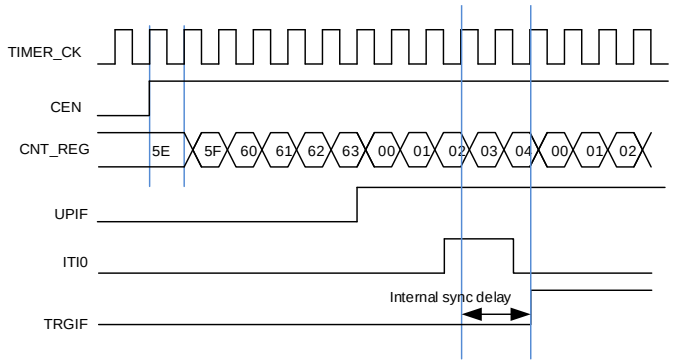
Refer to ***Hall sensor function.***

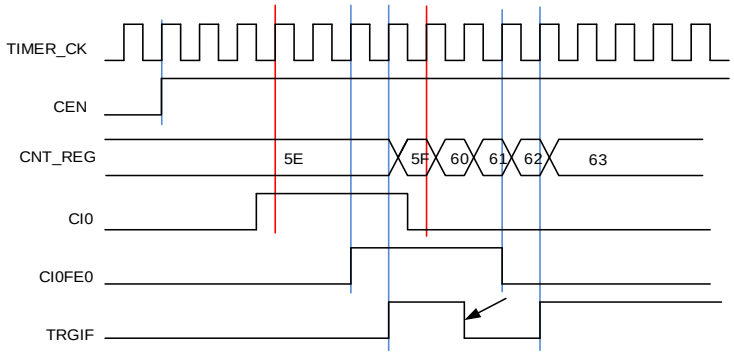
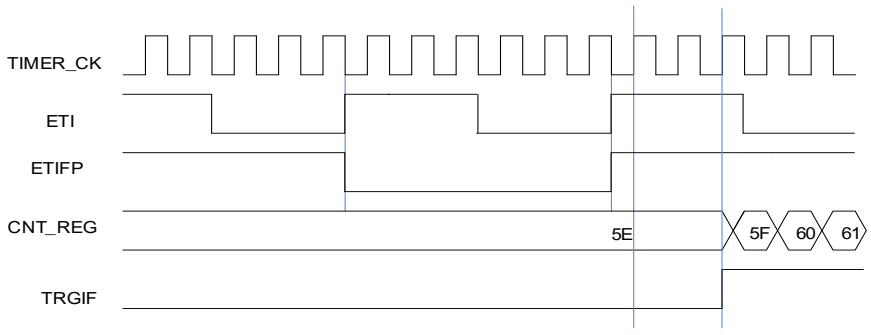
Master-slave management

The TIMERx can be synchronized with a trigger in several modes including restart mode, pause mode and event mode which is selected by the TSCFGy[4:0] in SYSCFG_TIMERxCFG(x=1,2).

Table 17-13. Examples of slave mode

	Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
LIST	TSCFGy[4:0] y=3 (restart mode) y=4 (pause mode) y=5 (event mode)	TSCFGy[4:0] 00001: Internal trigger input 0 (ITI0) 00010: Internal	If CI0FE0 or CI1FE1 is selected as the trigger source, configure the CHxP and CHxNP for	For the ITIx, no filter and prescaler can be used. For the CIx, filter can be used by configuring

	Mode Selection	Source Selection	Polarity Selection	Filter and Prescaler
		trigger input 1 (ITI1) 00011: Internal trigger input 2 (ITI2) 00100: Internal trigger input 3 (ITI3) 00101: Internal trigger input 4 (ITI4) 00110: Internal trigger input 5 (ITI5) 00111: Internal trigger input 6 (ITI6) 01000: Internal trigger input 7 (ITI7) 01001: Internal trigger input 8 (ITI8) 01010: Internal trigger input 9 (ITI9) 10101: CI0 edge flag (CI0F_ED) 10110: The filtered output of channel 0 input (CI0FE0) 10111: The filtered output of channel 1 input (CI1FE1) 11000: The filtered output of external trigger input (ETIFP)	the polarity selection and inversion. If ETIFP (the filtered output of external trigger input ETI) is selected as the trigger source, configure the ETP for polarity selection and inversion.	CHxCAPFLT, no prescaler can be used. For the ETIFP, filter can be used by configuring ETFC and prescaler can be used by configuring ETPSC.
Exam1	**Restart mode** The counter will be cleared and restart when a rising edge of trigger input comes.	TSCFG3[4:0] = 5'b00001 ITI0 is selected.	For ITI0, no polarity selector can be used.	For the ITI0, no filter and prescaler can be used.
	Figure 17-73. Restart mode 			

<table>
<tr><th></th><th>Mode Selection</th><th>Source Selection</th><th>Polarity Selection</th><th>Filter and Prescaler</th></tr>
<tr><td rowspan="2">Exam2</td><td>Pause mode
The counter will be paused when the trigger input is low, and it will start when the trigger input is high.</td><td>TSCFG4[4:0] = 5'b 10110
CI0FE0 is selected.</td><td>TI0S = 0 (Non-xor)[CH0P=0] CI0FE0 does not invert. The capture event will occur on the rising edge only.</td><td>Filter is bypassed in this example.</td></tr>
<tr><td colspan="4">Figure 17-74. Pause mode

</td></tr>
<tr><td rowspan="2">Exam3</td><td>Event mode
The counter will start to count when a rising edge of trigger input comes.</td><td>TSCFG5[4:0] = 5'b 11000
ETIFP is selected.</td><td>ETP = 0, the polarity of ETI does not change.</td><td>ETPSC = 1, ETI is divided by 2.
ETFC = 0, ETI does not filter.</td></tr>
<tr><td colspan="4">Figure 17-75. Event mode

</td></tr>
</table>

Single pulse mode

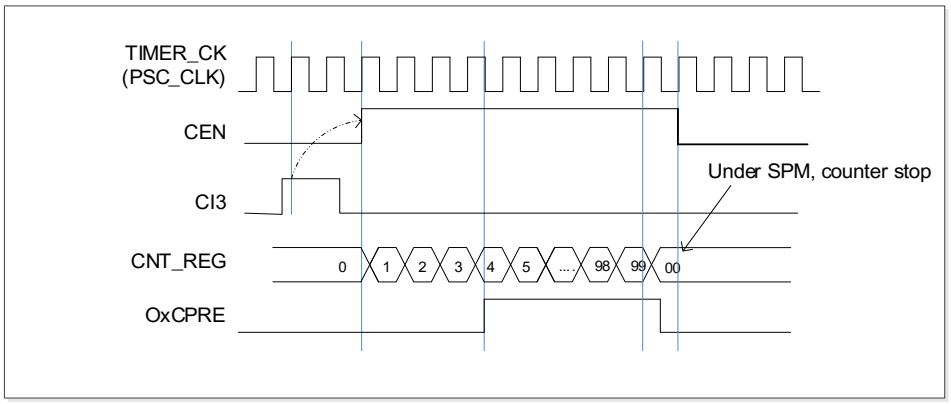
Single pulse mode is opposite to the repetitive mode, which can be enabled by setting SPM in TIMERx_CTL0. When you set SPM, the counter will be clear and stop when the next update event. In order to get pulse waveform, you can set the TIMERx to PWM mode or compare by CHxCOMCTL.

Once the timer is set to the single pulse mode, it is not necessary to configure the timer enable bit CEN in the TIMERx_CTL0 register to 1 to enable the counter. Setting the CEN bit to 1 or

a trigger signal edge can generate a pulse and then keep the CEN bit at a high state until the update event occurs or the CEN bit is written to 0 by software. If the CEN bit is cleared to 0 by software, the counter will be stopped and its value will be held.

In the single pulse mode, the active edge of trigger which sets the CEN bit to 1 will enable the counter. However, there exists several clock delays to perform the comparison result between the counter value and the TIMERx_CHxCV value. After a trigger rising occurs in the single pulse mode, the OxCPRE signal will immediately be forced to the state which the OxCPRE signal will change to, as the compare match event occurs without taking the comparison result into account.

Figure 17-76. Single pulse mode TIMERx_CHxCV = 0x04, TIMERx_CAR=0x99



Timers interconnection

Refer to ***Advanced timer (TIMERx, x=0, 7, 19, 20, 60, 61, 62, 63)Timers interconnection***.

Timer DMA mode

Timer DMA mode is the function that configures timer's register by DMA module. The relative registers are TIMERx_DMACFG and TIMERx_DMATB. Corresponding DMA request bit should be asserted to enable DMA request for internal interrupt event. TIMERx will send a request to DMA when the interrupt event occurs. DMA is configured to M2P (memory to peripheral) mode and the address of TIMERx_DMATB is configured to PADDR (peripheral base address), then DMA will access the TIMERx_DMATB. In fact, TIMERx_DMATB register is only a buffer, timer will map the TIMERx_DMATB to an internal register, appointed by the field of DMATA in TIMERx_DMACFG. If the field of DMATC in TIMERx_DMACFG is 0 (1 transfer), the timer sends only one DMA request. While if TIMERx_DMATC is not 0, such as 3 (4 transfers), then timer will send 3 more requests to DMA, and DMA will access timer's registers DMATA+0x4, DMATA+0x8 and DMATA+0xC at the next 3 accesses to TIMERx_DMATB. In a word, one-time DMA internal interrupt event asserts, (DMATC+1) times request will be sent by TIMERx.

If one more DMA request event occurs, TIMERx will repeat the process above.

Timer debug mode

When the Cortex®-M7 core is halted, and the TIMERx_HOLD configuration bit in DBG_CTL register set to 1, the TIMERx counter stops.

17.2.5. Registers definition (TIMERx, x=1,2)

TIMER1 base address: 0x4001 5C00

TIMER2 base address: 0x4000 0400

Table 17-14. TIMERx register overview

Register Name	Symbol	Address Offset	Reset Value
Control register 0	TIMERx_CTL0	0x00	0x0000 0000
Control register 1	TIMERx_CTL1	0x04	0x0000 0000
Slave mode configuration register	TIMERx_SMCFG	0x08	0x3000 0000
DMA and interrupt enable register 0	TIMERx_DMAINTEN0	0x0C	0x0000 0000
Interrupt flag register	TIMERx_INTF	0x10	0x0000 0000
Software event generation register	TIMERx_SWEVG	0x14	0x0000 0000
Channel control register 0	TIMERx_CHCTL0	0x18	0x0000 0000
Channel control register 1	TIMERx_CHCTL1	0x1C	0x0000 0000
Channel control register 2	TIMERx_CHCTL2	0x20	0x0000 0000
Counter register	TIMERx_CNT	0x24	0x0000 0000
Prescaler register	TIMERx_PSC	0x28	0x0000 0000
Counter auto reload register	TIMERx_CAR	0x2C	0x0000 FFFF
Counter repetition register	TIMERx_CREP	0x30	0x0000 0000
Channel 0 capture/compare value register	TIMERx_CH0CV	0x34	0x0000 0000
Channel 1 capture/compare value register	TIMERx_CH1CV	0x38	0x0000 0000
Channel 2 capture/compare value register	TIMERx_CH2CV	0x3C	0x0000 0000
Channel 3 capture/compare value register	TIMERx_CH3CV	0x40	0x0000 0000
Complementary channel protection register	TIMERx_CCHP	0x44	0x0000 0000
Channel 0 additional capture/compare value register	TIMERx_CH0CV_ADD	0x64	0x0000 0000
Channel 1 additional capture/compare value register	TIMERx_CH1CV_ADD	0x68	0x0000 0000
Channel 2 additional capture/compare value register	TIMERx_CH2CV_ADD	0x6C	0x0000 0000
Channel 3 additional capture/compare value register	TIMERx_CH3CV_ADD	0x70	0x0000 0000
Control register 2	TIMERx_CTL2	0x74	0x0000 0000
Watchdog counter period register	TIMERx_WDGPER	0xA0	0xFFFF FFFF
Counter initial control register	TIMERx_CINITCTL	0xA4	0x0000 0000
Counter initial value register	TIMERx_CINITV	0xA8	0x0000 0000
Channel multi-edge capture configuration register	TIMERx_CHMECAPCFG	0xAC	0x0000 0000
Channel 0 multi-edge capture register	TIMERx_CH0MECAP	0xB4	0x0000 0000
Channel 1 multi-edge capture register	TIMERx_CH1MECAP	0xB8	0x0000 0000
Channel 2 multi-edge capture register	TIMERx_CH2MECAP	0xBC	0x0000 0000
Channel 3 multi-edge capture register	TIMERx_CH3MECAP	0xC0	0x0000 0000
DMA configuration register	TIMERx_DMACFG	0xE0	0x0000 0000
DMA transfer buffer register	TIMERx_DMATB	0xE4	0x0000 0000
Configuration register	TIMERx_CFG	0xFC	0x0000 0000

Control register 0 (TIMERx_CTL0)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					CFBEN	CKDIV[1:0]		ARSE	CAM[1:0]		DIR	SPM	UPS	UPDIS	CEN
					rw	rw		rw	rw		rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:11	Reserved	Must be kept at reset value.
10	CFBEN	Counter freeze in break mode enable 0: The counter is not freeze in break mode when CFBEN is reset 1: The counter is freeze in break mode when CFBEN is set Note: This bit is valid when the BRKEN is placed in the TIMER x_CCHP.
9:8	CKDIV[1:0]	Clock division The CKDIV bits can be configured by software to specify division factor between the CK_TIMER and the dead-time and digital filter sample clock (DTS). 00: f_{DTS}= f_{CK_TIMER} 01: f_{DTS}= f_{CK_TIMER} /2 10: f_{DTS}= f_{CK_TIMER} /4 11: Reserved
7	ARSE	Auto-reload shadow enable 0: The shadow register for TIMERx_CAR register is disabled 1: The shadow register for TIMERx_CAR register is enabled
6:5	CAM[1:0]	Counter align mode selection 00: No center-aligned mode (edge-aligned mode). The direction of the counter is specified by the DIR bit. 01: Center-aligned and counting down assert mode. The counter counts in center-aligned mode and channel is configured in output mode (CHxMS = 3'b000 in TIMERx_CHCTL0 register). Only when counting down, CHxIF/MCHxIF bit and CHxCOMADDIF/MCHxADDCOMIF can be set. 10: Center-aligned and counting up assert mode. The counter counts in center-aligned mode and channel is configured in output mode (CHxMS = 3'b000 in TIMERx_CHCTL0 register). Only when counting up, CHxIF/MCHxIF bit and CHxCOMADDIF/MCHxADDCOMIF can be set. 11: Center-aligned and counting up/down assert mode. The counter counts in center-aligned mode and channel is configured in output mode (CHxMS = 3'b000 in TIMERx_CHCTL0 register). Both when counting up and counting down, CHxIF/MCHxIF bit and CHxCOMADDIF/MCHxADDCOMIF can be set. After the counter is enabled, these bits cannot be switched from 0x00 to non 0x00.
4	DIR	Direction 0: Count up

1: Count down

If the timer work in center-aligned mode or decoder mode, this bit is read only.

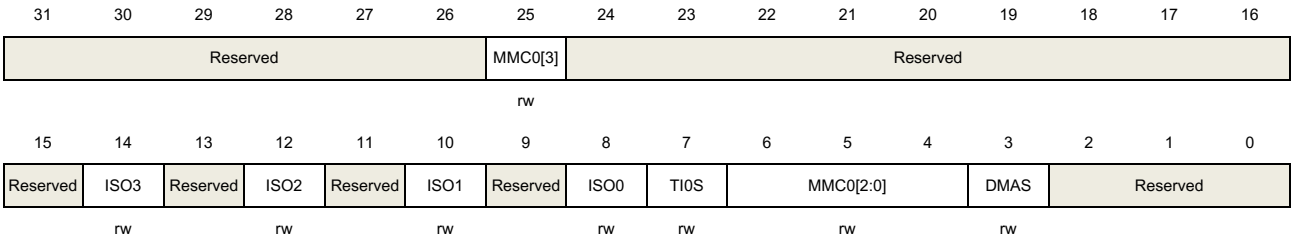
3	SPM	Single pulse mode 0: Single pulse mode is disabled. The counter continues after an update event. 1: Single pulse mode is enabled. The counter counts until the next update event occurs.
2	UPS	Update source This bit is used to select the update event sources by software. 0: These events generate an update interrupt or a DMA request: – The UPG bit is set. – The counter generates an overflow or underflow event. – The restart mode generates an update event. 1: This event generates update interrupts or DMA requests: The counter generates an overflow or underflow event
1	UPDIS	Update disable This bit is used to enable or disable the update event generation. 0: Update event enable. When an update event occurs, the corresponding shadow registers are loaded with their preloaded values. These events generate update event: – The UPG bit is set. – The counter generates an overflow or underflow event. – The restart mode generates an update event. 1: Update event disable. **Note:** When this bit is set to 1, setting UPG bit or restart mode does not generate an update event, but the counter and prescaler are initialized.
0	CEN	Counter enable 0: Counter disable 1: Counter enable The CEN bit must be set by software when timer works in external clock mode, pause mode or decoder mode.

Control register 1 (TIMERx_CTL1)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25	MMC0[3]	Refer to MMC0[2:0] description.
24:15	Reserved	Must be kept at reset value.
14	ISO3	Idle state of channel 3 output. Refer to ISO0 bit.
13	Reserved	Must be kept at reset value.
12	ISO2	Idle state of channel 2 output Refer to ISO0 bit
11	Reserved	Must be kept at reset value.
10	ISO1	Idle state of channel 1 output Refer to ISO0 bit
9	Reserved	Must be kept at reset value.
8	ISO0	Idle state of channel 0 output 0: When POEN bit is reset, CH0_O is set low. 1: When POEN bit is reset, CH0_O is set high. The CH0_O output changes after a dead time if MCH0_O is implemented. This bit can be modified only when PROT[1:0] bits in TIMERx_CCHP register is 00.
7	TI0S	Channel 0 trigger input selection 0: The TIMERx_CH0 pin input is selected as channel 0 trigger input. 1: The result of combinational XOR of TIMERx_CH0, TIMERx_CH1 and TIMERx_CH2 pins is selected as channel 0 trigger input.
6:4	MMC0[2:0]	Master mode control This bit-field combine with MMC0[3] control the selection of TRGO signal, which is sent in master mode to slave timers for synchronization function. 0000: When a counter reset event occurs, a TRGO trigger signal is output. The counter reset source: Master timer generate a reset the UPG bit in the TIMERx_SWEVG register is set 0001: Enable. When a counter start event occurs, a TRGO trigger signal is output. The counter start source : CEN control bit is set The trigger input in pause mode is high 0010: When an update event occurs, a TRGO trigger signal is output. The update source depends on UPDIS bit and UPS bit. 0011: When a capture or compare pulse event occurs in channel0, a TRGO trigger signal is output. 0100: When a compare event occurs, a TRGO trigger signal is output. The compare

		source is from O0CPRE. 0101: When a compare event occurs, a TRGO trigger signal is output. The compare source is from O1CPRE. 0110: When a compare event occurs, a TRGO trigger signal is output. The compare source is from O2CPRE. 0111: When a compare event occurs, a TRGO trigger signal is output. The compare source is from O3CPRE. 1000:reserved. 1001: When an overflow event occurs, a TRGO trigger signal is output. 1010: When a underflow event occurs, a TRGO trigger signal is output. 1011: When a software event occurs, a TRGO trigger signal is output. Other: reserved.
3	DMAS	DMA request source selection 0: When capture or compare event occurs, the DMA request of channel x is sent 1: When update event occurs, the DMA request of channel x is sent.
2:0	Reserved	Must be kept at reset value.

Slave mode configuration register (TIMERx_SMCFG)

Address offset: 0x08

Reset value: 0x3000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		ETISEL[1:0]		Reserved											
		rw													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	SMC1	ETPSC[1:0]		ETFC[3:0]				MSM	Reserved						
rw	rw	rw		rw				rw							

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29:28	ETISEL[1:0]	TIMER0 external trigger select 00: timer external trigger 0 01: timer external trigger 1 10: timer external trigger 2 11: Reserved
27:16	Reserved	Must be kept at reset value.
15	ETP	External trigger polarity This bit specifies the polarity of ETI signal. 0: ETI is active at rising edge or high level.

1: ETI is active at falling edge or low level.

14 SMC1

Part of SMC is used to enable external clock mode 1

In external clock mode 1, the counter is clocked by any active edge of the ETIFP signal.

0: External clock mode 1 disabled

1: External clock mode 1 enabled

When the slave mode is configured as restart mode, pause mode or event mode, the timer can still work in the external clock 1 mode by setting this bit. But the TRGS bits must not be 3'b111 in this case.

The external clock input will be ETIFP if external clock mode 0 and external clock mode 1 are enabled at the same time.

Note: External clock mode 0 enable is in SYSCFG_TIMERxCFG1 register's TSCFG6[4:0] bit field.

13:12 ETPSC[1:0]

The prescaler of external trigger

The frequency of external trigger signal ETIFP must not be higher than 1/4 of TIMER_CK frequency. When the frequency of external trigger signal is high, the prescaler can be enabled to reduce ETIFP frequency.

00: Prescaler disabled

01: The prescaler is 2

10: The prescaler is 4

11: The prescaler is 8

11:8 ETFC[3:0]

External trigger filter control

The external trigger can be filtered by digital filter and this bit-field configure the filtering capability.

Basic principle of digital filter: continuously sample the external trigger signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit-field, it is considered to be an effective level.

The filtering capability configuration is as follows:

EXTFC[3:0]	Times	f_{SAMP}
4'b0000	Filter disabled.	
4'b0001	2	f_{CK_TIMER}
4'b0010	4	
4'b0011	8	
4'b0100	6	$f_{DTS_CK}/2$
4'b0101	8	
4'b0110	6	$f_{DTS_CK}/4$
4'b0111	8	
4'b1000	6	$f_{DTS_CK}/8$
4'b1001	8	
4'b1010	5	$f_{DTS_CK}/16$
4'b1011	6	

4'b1100	8	
4'b1101	5	$f_{DTS_CK}/32$
4'b1110	6	
4'b1111	8	

7	MSM	Master-slave mode This bit can be used to synchronize the selected timers to begin counting at the same time. The TRGI is used as the start event, and through TRGO, timers are connected. 0: Master-slave mode disabled 1: Master-slave mode enabled
6:0	Reserved	Must be kept at reset value.

DMA and interrupt enable register 0 (TIMERx_DMAINTEN0)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH3COMADDIE	CH2COMADDIE	CH1COMADDIE	CH0COMADDIE	Reserved										DECDISIE	DECJIE
rw	rw	rw	rw											rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	TRGDEN	Reserved	CH3DEN	CH2DEN	CH1DEN	CH0DEN	UPDEN	BRKIE	TRGIE	Reserved	CH3IE	CH2IE	CH1IE	CH0IE	UPIE
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31	CH3COMADDIE	Channel 3 additional compare interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used in composite PWM mode and asymmetric PWM mode(when CH3MS[2:0] = 3'b000 and CH3COMCTL[3:2]=2'b10).
30	CH2COMADDIE	Channel 2 additional compare interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used in composite PWM mode and asymmetric PWM mode(when CH3MS[2:0] = 3'b000 and CH3COMCTL[3:2]=2'b10).
29	CH1COMADDIE	Channel 1 additional compare interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used in composite PWM mode and asymmetric PWM mode(when CH3MS[2:0] = 3'b000 and CH3COMCTL[3:2]=2'b10).

28	CH0COMADDIE	Channel 0 additional compare interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used in composite PWM mode and asymmetric PWM mode(when CH3MS[2:0] = 3'b000 and CH3COMCTL[3:2]=2'b10).
27:18	Reserved	Must be kept at reset value.
17	DECDISIE	Quadrature decoder signal disconnection interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used for quadrature decoder signal disconnection detection is enabled (when DECDISDEN =1).
16	DECJIE	Quadrature decoder signal jump (the two signals jump at the same time) interrupt enable 0: Disabled 1: Enabled **Note:** This bit just used for quadrature decoder signal jump detection is enabled (when DECJDEN =1).
15	Reserved	Must be kept at reset value.
14	TRGDEN	Trigger DMA request enable 0: Disabled 1: Enabled
13	Reserved	Must be kept at reset value.
12	CH3DEN	Channel 3 capture/compare DMA request enable 0: Disabled 1: Enabled
11	CH2DEN	Channel 2 capture/compare DMA request enable 0: Disabled 1: Enabled
10	CH1DEN	Channel 1 capture/compare DMA request enable 0: Disabled 1: Enabled
9	CH0DEN	Channel 0 capture/compare DMA request enable 0: Disabled 1: Enabled
8	UPDEN	Update DMA request enable 0: Disabled 1: Enabled
7	BRKIE	Break interrupt enable

		0: Disabled 1: Enabled
6	TRGIE	Trigger interrupt enable 0: Disabled 1: Enabled
5	Reserved	Must be kept at reset value.
4	CH3IE	Channel 3 capture/compare interrupt enable 0: Disabled 1: Enabled
3	CH2IE	Channel 2 capture/compare interrupt enable 0: Disabled 1: Enabled
2	CH1IE	Channel 1 capture/compare interrupt enable 0: Disabled 1: Enabled
1	CH0IE	Channel 0 capture/compare interrupt enable 0: Disabled 1: Enabled
0	UPIE	Update interrupt enable 0: Disabled 1: Enabled

Interrupt flag register (TIMERx_INTF)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH3COMADDIF	CH2COMADDIF	CH1COMADDIF	CH0COMADDIF	Reserved										DECDISIF	DECJIF
rc_w0	rc_w0	rc_w0	rc_w0											rc_w0	rc_w0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		SYSBIF	CH3OF	CH2OF	CH1OF	CH0OF	Reserved	BRKIF	TRGIF	Reserved	CH3IF	CH2IF	CH1IF	CH0IF	UPIF
rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	.	rc_w0	rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

Bits	Fields	Descriptions
31	CH3COMADDIF	Channel 3 additional compare interrupt flag. Refer to CH0COMADDIF description.
30	CH2COMADDIF	Channel 2 additional compare interrupt flag.

		Refer to CH0COMADDIF description.
29	CH1COMADDIF	Channel 1 additional compare interrupt flag. Refer to CH0COMADDIF description.
28	CH0COMADDIF	Channel 0 additional compare interrupt flag. This flag is set by hardware and cleared by software. If channel 0 is in output mode, this flag is set when a compare event occurs. 0: No channel 0 output compare interrupt occurred 1: Channel 0 output compare interrupt occurred **Note:** This flag just used in composite PWM mode (when CH0CPWMEN=1, CH0MS[2:0] = 3'b000 and CH0COMCTL=3'b110 or 3'b111).
27:18	Reserved	Must be kept at reset value.
17	DECDISIF	Quadrature decoder signal disconnection interrupt flag 0: No quadrature decoder signal disconnection interrupt occurred 1: Quadrature decoder signal disconnection interrupt occurred **Note:** This bit just used for quadrature decoder signal disconnection detection is enabled (when DECDISDEN =1).
16	DECJIF	Quadrature decoder signal jump (the two signals jump at the same time) interrupt flag 0: No quadrature decoder signal jump interrupt occurred 1: Quadrature decoder signal jump interrupt occurred **Note:** This bit just used for quadrature decoder signal jump detection is enabled (when DECJDEN =1).
15:14	Reserved	Must be kept at reset value.
13	SYSBIF	System source break interrupt flag This flag is set by hardware when the system sources are active, and cleared by software if the system sources are inactive. 0: No system source break interrupt occurred 1: System source break interrupt occurred **Note**: When this bit is set, this bit must be cleared by software before the channel outputs are restored.
12	CH3OF	Channel 3 over capture flag Refer to CH0OF description
11	CH2OF	Channel 2 over capture flag Refer to CH0OF description
10	CH1OF	Channel 1 over capture flag Refer to CH0OF description
9	CH0OF	Channel 0 over capture flag When channel 0 is configured in input mode, this flag is set by hardware when a capture event occurs while CH0IF flag has already been set. This flag is cleared by

		software. 0: No over capture interrupt occurred 1: Over capture interrupt occurred
8:7	Reserved	Must be kept at reset value.
7	BRKIF	Break interrupt flag When the break input is inactive, the bit is set by hardware. When the break input is inactive, the bit can be cleared by software. 0: No active level break has been detected. 1: An active level has been detected.
6	TRGIF	Trigger interrupt flag This flag is set on trigger event and cleared by software. When in pause mode, both edges on trigger input generates a trigger event, otherwise, only an active edge on trigger input can generates a trigger event. 0: No trigger event occurred 1: Trigger interrupt occurred
5	Reserved	Must be kept at reset value.
4	CH3IF	Channel 3 capture/compare interrupt flag Refer to CH0IF description
3	CH2IF	Channel 2 capture/compare interrupt flag Refer to CH0IF description
2	CH1IF	Channel 1 capture/compare interrupt flag Refer to CH0IF description
1	CH0IF	Channel 0 capture/compare interrupt flag This flag is set by hardware and cleared by software. If channel 0 is in input mode, this flag is set when a capture event occurs. If channel 0 is in output mode, this flag is set when a compare event occurs. If channel 0 is set to input mode, this bit will be reset by reading TIMERx_CH0CV. 0: No channel 0 interrupt occurred 1: Channel 0 interrupt occurred
0	UPIF	Update interrupt flag This bit is set by hardware when an update event occurs and cleared by software. 0: No update interrupt occurred 1: Update interrupt occurred

Software event generation register (TIMERx_SWEVG)

Address offset: 0x14
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH3COMADDG	CH2COMADDG	CH1COMADDG	CH0COMADDG	Reserved											
w	w	w	w												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TRGG	Reserved	CH3G	CH2G	CH1G	CH0G	UPG
									w		w	w	w	w	w

Bits	Fields	Descriptions
31	CH3COMADDG	Channel 3 additional compare event generation. Refer to CH0COMADDG description.
30	CH2COMADDG	Channel 2 additional compare event generation. Refer to CH0COMADDG description.
29	CH1COMADDG	Channel 1 additional compare event generation. Refer to CH0COMADDG description.
28	CH0COMADDG	Channel 0 additional compare event generation. This bit is set by software to generate a compare event in channel 0 additional , it is automatically cleared by hardware. When this bit is set, the CH0COMADDIF flag will be set, and the corresponding interrupt will be sent if enabled. 0: No generate a channel 0 additional compare event 1: Generate a channel 0 additional compare event **Note:** This bit just used in composite PWM mode and asymmetric PWM mode(when CH3MS[2:0] = 3'b000 and CH3COMCTL[3:2]=2'b10).
27:7	Reserved	Must be kept at reset value.
6	TRGG	Trigger event generation This bit is set by software and cleared by hardware automatically. When this bit is set, the TRGIF flag in TIMERx_INTF register will be set, related interrupt or DMA transfer can occur if enabled. 0: No generate a trigger event 1: Generate a trigger event
5	Reserved	Must be kept at reset value.
4	CH3G	Channel 3 capture or compare event generation Refer to CH0G description
3	CH2G	Channel 2 capture or compare event generation Refer to CH0G description
2	CH1G	Channel 1 capture or compare event generation Refer to CH0G description
1	CH0G	Channel 0 capture or compare event generation

This bit is set by software to generate a capture or compare event in channel 0, it is automatically cleared by hardware. When this bit is set, the CH0IF flag will be set, and the corresponding interrupt or DMA request will be sent if enabled. In addition, if channel 0 is configured in input mode, the current value of the counter is captured to TIMERx_CH0CV register, and the CH0OF flag is set if the CH0IF flag has been set.

0: No generate a channel 0 capture or compare event

1: Generate a channel 0 capture or compare event

0	UPG	Update event generation This bit can be set by software, and automatically cleared by hardware. When this bit is set, the counter is cleared if the center-aligned or up counting mode is selected, while in down counting mode it takes the auto-reload value. The prescaler counter is cleared at the same time. 0: No generate an update event 1: Generate an update event

Channel control register 0 (TIMERx_CHCTL0)

Address offset: 0x18

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		CH1COMADDSEN	CH0COMADDSEN	CH1COMADDUPSEL	CH0COMADDUPSEL	Reserved	CH1COMCTL[3]	Reserved							CH0COMCTL[3]
		Reserved	Reserved	Reserved	Reserved		Reserved								Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1COMCEN	CH1COMCTL[2:0]			CH1COMSEN	CH1COMFEN	CH1MS[1:0]		CH0COMCEN	CH0COMCTL[2:0]			CH0COMSEN	CH0COMFEN	CH0MS[1:0]	
CH1CAPFLT[3:0]				CH1CAPPSC[1:0]				CH0CAPFLT[3:0]				CH0CAPPSC[1:0]			
rw				rw		rw		rw				rw		rw	

Output compare mode:

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29	CH1COMADDSEN	Channel 1 additional compare output shadow enable Refer to CH0COMADDSEN description.
28	CH0COMADDSEN	Channel 0 additional compare output shadow enable When this bit is set, the shadow register of TIMERx_CH0COMV_ADD register which updates at each update event will be enabled. 0: Channel 0 additional compare output shadow disabled 1: Channel 0 additional compare output shadow enabled

		The PWM mode can be used without validating the shadow register only in single pulse mode (SPM bit in TIMERx_CTL0 register is set). This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 and CH0MS bit-field is 000.
27	CH1COMADDUPSEL	Refer to CH0COMADDUPSEL.
26	CH0COMADDUPSEL	Update selection of channel 0 additional compare value 0: Channel 0 additional compare value is update from update event. 1: Channel 0 additional compare value is update from compare event.
25	Reserved	Must be kept at reset value.
24	CH1COMCTL[3]	Refer to CH1COMCTL[2:0]
23:17	Reserved	Must be kept at reset value.
16	CH0COMCTL[3]	Refer to CH0COMCTL[2:0]
15	CH1COMCEN	Channel 1 output compare clear enable Refer to CH0COMCEN description
14:12	CH1COMCTL[2:0]	Channel 1 compare output control Refer to CH0COMCTL description
11	CH1COMSEN	Channel 1 output compare shadow enable Refer to CH0COMSEN description
10	CH1COMFEN	Channel 1 output compare fast enable Refer to CH0COMSEN description
9:8	CH1MS[1:0]	Channel 1 mode selection This bit-field specifies the direction of the channel and the input signal selection. This bit-field is writable only when the channel is not active (CH1EN bit in TIMERx_CHCTL2 register is reset). 00: Channel 1 is programmed as output. 01: Channel 1 is programmed as input, IS1 is connected to CI1FE1. 10: Channel 1 is programmed as input, IS1 is connected to CI0FE1. 11: Channel 1 is programmed as input, IS1 is connected to ITS. This mode is working only if an internal trigger input is selected (through SYSCFG_TIMERxCFG2 register's TSCFG15[4:0]).
7	CH0COMCEN	Channel 0 output compare clear enable When this bit is set, if the ETIFP signal is detected as high level, the O0CPRE signal will be cleared. 0: Channel 0 output compare clear disabled 1: Channel 0 output compare clear enabled
6:4	CH0COMCTL[2:0]	Channel 0 compare output control The CH0COMCTL[3] and CH0COMCTL[2:0] bit-field control the behavior of

O0CPRE which drives CH0_O. The active level of O0CPRE is high, while the active level of CH0_O depends on CH0P bit.

Note: When multi mode channel 0 is configured in output mode, and the MCH0MSEL[1:0] = 2b'11, This bit-field controls the behavior of O0CPRE which drives CH0_O and MCH0_O. The active level of MO0CPRE is high, while the active level of CH0_O and MCH0_O depends on CH0P and MCH0P bits.

0000: Timing mode. The O0CPRE signal keeps stable, independent of the comparison between the register TIMERx_CH0CV and the counter TIMERx_CNT.

0001: Set the channel output on match. O0CPRE signal is forced high when the counter matches the output compare register TIMERx_CH0CV.

0010: Clear the channel output on match. O0CPRE signal is forced low when the counter matches the output compare register TIMERx_CH0CV.

0011: Toggle on match. O0CPRE toggles when the counter matches the output compare register TIMERx_CH0CV.

0100: Force low. O0CPRE is forced low level.

0101: Force high. O0CPRE is forced high level.

0110: PWM mode 0. When counting up, O0CPRE is active as long as the counter is smaller than TIMERx_CH0CV, otherwise it is inactive. When counting down, MO0CPRE is inactive as long as the counter is larger than TIMERx_CH0CV, otherwise it is active.

0111: PWM mode 1. When counting up, O0CPRE is inactive as long as the counter is smaller than TIMERx_CH0CV, otherwise it is active. When counting down, O0CPRE is active as long as the counter is larger than TIMERx_CH0CV, otherwise it is inactive.

1000: Composite PWM mode 0, When counting up, the channel output is forced low level when the counter is matched with the value of TIMERx_CH0CV, the channel output is forced high level when the counter matches the value of TIMERx_CH0CV_ADD. When counting down, the channel output is forced high level when the counter is matched with the value of TIMERx_CH0CV, the channel output is forced low level when the counter matches the value of TIMERx_CH0CV_ADD.

1001: Composite PWM mode 1. When counting up, the channel output is forced high level when the counter is matched with the value of TIMERx_CH0CV, the channel output is forced low level when the counter matches the value of TIMERx_CH0CV_ADD. When counting down, the channel output is forced low level when the counter is matched with the value of TIMERx_CH0CV, the channel output is forced high level when the counter matches the value of TIMERx_CH0CV_ADD.

1010: Asymmetric PWM mode 0. Only for Center aligned mode, when counting up, the channel output is forced low level when the counter is matched with the value of TIMERx_CH0CV, when counting down, the channel output is forced high level when the counter is matched with the value of TIMERx_CH0CV_ADD.

1011: Asymmetric PWM mode 1. Only for Center aligned mode, when counting up, the channel output is forced high level when the counter is matched with the value of TIMERx_CH0CV, when counting down, the channel output is forced low level

		when the counter is matched with the value of TIMERx_CH0CV_ADD. 1100~1111: Reserved.
3	CH0COMSEN	Channel 0 compare output shadow enable When this bit is set, the shadow register of TIMERx_CH0CV register which updates at each update event will be enabled. 0: Channel 0 output compare shadow disabled 1: Channel 0 output compare shadow enabled The PWM mode can be used without verifying the shadow register only in single pulse mode (when SPM=1). This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 and CH0MS bit-field is 000.
2	CH0COMFEN	Channel 0 output compare fast enable When this bit is set, the effect of an event on the trigger in input on the capture /compare output will be accelerated if the channel is configured in PWM0 or PWM1 mode. The output channel will treat an active edge on the trigger input as a compare match, and CH0_O is set to the compare level independently from the result of the comparison. 0: Channel 0 output quickly compare disable. The minimum delay from an edge on the trigger input to activate CH0_O output is 5 clock cycles. 1: Channel 0 output quickly compare enable. The minimum delay from an edge on the trigger input to activate CH0_O output is 3 clock cycles.
1:0	CH0MS[1:0]	Channel 0 I/O mode selection This bit-field specifies the work mode of the channel and the input signal selection. This bit-field is writable only when the channel is not active (CH0EN bit in TIMERx_CHCTL2 register is reset). 00: Channel 0 is programmed as output. 01: Channel 0 is programmed as input, IS0 is connected to CI0FE0. 10: Channel 0 is programmed as input, IS0 is connected to CI1FE0. 11: Channel 0 is programmed as input, IS0 is connected to ITS. This mode is working only if an internal trigger input is selected (through SYSCFG_TIMERxCFG2 register's TSCFG15[4:0]).

Input capture mode:

Bits	Fields	Descriptions
31	CH1MS[2]	Channel 1 I/O mode selection Same as output compare mode.
30	CH0MS[2]	Channel 0 I/O mode selection Same as output compare mode.
29:16	Reserved	Must be kept at reset value.
15:12	CH1CAPFLT[3:0]	Channel 1 input capture filter control Refer to CH0CAPFLT description.

11:10 CH1CAPPSC[1:0] Channel 1 input capture prescaler
Refer to CH0CAPPSC description.

9:8 CH1MS[1:0] Channel 1 I/O mode selection
Same as output compare mode.

7:4 CH0CAPFLT[3:0] Channel 0 input capture filter control
The CI0 input signal can be filtered by digital filter and this bit-field configure the filtering capability.
Basic principle of digital filter: continuously sample the CI0 input signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit, it is considered to be an effective level.
The filtering capability configuration is as follows:

CH0CAPFLT [3:0]	Times	f_{SAMP}
4'b0000	Filter disabled.	
4'b0001	2	
4'b0010	4	f_{CK_TIMER}
4'b0011	8	
4'b0100	6	$f_{DTS}/2$
4'b0101	8	
4'b0110	6	$f_{DTS}/4$
4'b0111	8	
4'b1000	6	$f_{DTS}/8$
4'b1001	8	
4'b1010	5	
4'b1011	6	$f_{DTS}/16$
4'b1100	8	
4'b1101	5	
4'b1110	6	$f_{DTS}/32$
4'b1111	8	

3:2 CH0CAPPSC[1:0] Channel 0 input capture prescaler
This bit-field specifies the factor of the prescaler on channel 0 input. The prescaler is reset when CH0EN bit in TIMERx_CHCTL2 register is cleared.
00: Prescaler disabled, input capture occurs on every channel input edge.
01: The input capture occurs on every 2 channel input edges
10: The input capture occurs on every 4 channel input edges
11: The input capture occurs on every 8 channel input edges

1:0 CH0MS[1:0] Channel 0 I/O mode selection
Same as output compare mode.

Channel control register 1 (TIMERx_CHCTL1)

Address offset: 0x1C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		CH3COMADDSEN	CH2COMADDSEN	CH3COMADDUPSEL	CH2COMADDUPSEL	Reserved	CH3COMCTL[3]	Reserved							CH2COMCTL[3]
		Reserved	Reserved	Reserved	Reserved		Reserved								Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3COMCEN	CH3COMCTL[2:0]			CH3COMSEN	CH3COMFEN	CH3MS[1:0]		CH2COMCEN	CH2COMCTL[2:0]			CH2COMSEN	CH2COMFEN	CH2MS[1:0]	
CH3CAPFLT[3:0]				CH3CAPPSC[1:0]				CH2CAPFLT[3:0]				CH2CAPPSC[1:0]			
rw				rw		rw		rw				rw		rw	

Output compare mode:

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29	CH3COMADDSEN	Channel 3 additional compare output shadow enable Refer to CH2COMADDSEN description.
28	CH2COMADDSEN	Channel 2 additional compare output shadow enable When this bit is set, the shadow register of TIMERx_CH2COMV_ADD register which updates at each update event will be enabled. 0: Channel 2 additional compare shadow disabled 1: Channel 2 additional compare shadow enabled The PWM mode can be used without validating the shadow register only in single pulse mode (SPM bit in TIMERx_CTL0 register is set). This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 and CH2MS bit-field is 000.
27	CH3COMADDUPSEL	Refer to CH0COMADDUPSEL.
26	CH2COMADDUPSEL	Update selection of channel 2 additional compare value 0: Channel 2 additional compare value is update from update event. 1: Channel 2 additional compare value is update from compare event.
25	Reserved	Must be kept at reset value.
24	CH3COMCTL[3]	Refer to CH3COMCTL[2:0]
23:17	Reserved	Must be kept at reset value.
16	CH2COMCTL[3]	Refer to CH2COMCTL[2:0]
15	CH3COMCEN	Channel 3 output compare clear enable Refer to CH2COMCEN description
14:12	CH3COMCTL[2:0]	Channel 3 compare output control Refer to CH0COMCTL description

11	CH3COMSEN	Channel 3 output compare shadow enable Refer to CH2COMSEN description
10	CH3COMFEN	Channel 3 output compare fast enable Refer to CH2COMSEN description
9:8	CH3MS[1:0]	Channel 3 I/O mode selection This bit-field specifies the direction of the channel and the input signal selection. The CH3MS[2:0] bit-field is writable only when the channel is not active (When MCH3MSEL[1:0] = 2b'00, the CH3EN bit in TIMERx_CHCTL2 register is reset; when MCH3MSEL[1:0] = 2b'01 or 2b'11, the CH3EN and MCH3EN bits in TIMERx_CHCTL2 register are reset). 00: Channel 3 is programmed as output. 01: Channel 3 is programmed as input, IS3 is connected to CI3FE3. 10: Channel 3 is programmed as input, IS3 is connected to CI2FE3. 11: Channel 3 is programmed as input, IS3 is connected to ITS, this mode is working only if an internal trigger input is selected (through TSCFG15 bits in SYSCFG_TIMERxCFG2 register).
7	CH2COMCEN	Channel 2 output compare clear enable. When this bit is set, if the ETIFP signal is detected as high level, the O2CPRE signal will be cleared. 0: Channel 2 output compare clear disabled 1: Channel 2 output compare clear enabled
6:4	CH2COMCTL[2:0]	Channel 1 output compare shadow enable Refer to CH0COMSEN description
3	CH2COMSEN	Channel 2 output compare shadow enable Refer to CH0COMSEN description
2	CH2COMFEN	Channel 2 output compare fast enable When this bit is set, the effect of an event on the trigger in input on the capture/ compare output will be accelerated if the channel is configured in PWM0 or PWM1 mode. The output channel will treat an active edge on the trigger input as a compare match, and CH2_O is set to the compare level independently from the result of the comparison. 0: Channel 2 output quickly compare disable. The minimum delay from an edge on the trigger input to activate CH2_O output is 5 clock cycles. 1: Channel 2 output quickly compare enable. The minimum delay from an edge on the trigger input to activate CH2_O output is 3 clock cycles.
1:0	CH2MS[1:0]	Channel 2 I/O mode selection This bit-field specifies the work mode of the channel and the input signal selection. The CH2MS[2:0] bit-field is writable only when the channel is not active(When MCH2MSEL[1:0] = 2b'00, the CH2EN bit in TIMERx_CHCTL2 register is reset; when MCH2MSEL[1:0] = 2b'01 or 2b'11, the CH2EN and MCH2EN bits in TIMERx_CHCTL2 register are reset).

00: Channel 2 is programmed as output.
01: Channel 2 is programmed as input, IS2 is connected to CI2FE2.
10: Channel 2 is programmed as input, IS2 is connected to CI3FE2.
11: Channel 2 is programmed as input, IS2 is connected to ITS. This mode is working only if an internal trigger input is selected (through TSCFG15 bits in SYSCFG_TIMERxCFG2 register).

Input capture mode:

Bits	Fields	Descriptions
31	CH3MS[2]	Channel 3 I/O mode selection Same as output compare mode.
30	CH2MS[2]	Channel 2 I/O mode selection Same as output compare mode.
31:16	Reserved	Must be kept at reset value.
15:12	CH3CAPFLT[3:0]	Channel 3 input capture filter control Refer to CH2CAPFLT description.
11:10	CH3CAPPSC[1:0]	Channel 3 input capture prescaler Refer to CH2CAPPSC description.
9:8	CH3MS[1:0]	Channel 3 mode selection Same as output compare mode.
7:4	CH2CAPFLT[3:0]	Channel 2 input capture filter control The CI2 input signal can be filtered by digital filter and this bit-field configure the filtering capability. Basic principle of digital filter: continuously sample the CI2 input signal according to f_{SAMP} and record the number of times of the same level of the signal. After reaching the filtering capacity configured by this bit, it is considered to be an effective level. The filtering capability configuration is as follows:

CH2CAPFLT [3:0]	Times	f_{SAMP}
4'b0000	Filter disabled.	
4'b0001	2	f_{CK_TIMER}
4'b0010	4	
4'b0011	8	
4'b0100	6	$f_{DTS}/2$
4'b0101	8	
4'b0110	6	$f_{DTS}/4$
4'b0111	8	
4'b1000	6	$f_{DTS}/8$
4'b1001	8	
4'b1010	5	$f_{DTS}/16$
4'b1011	6	
4'b1100	8	

		4'b1101	5	
		4'b1110	6	$f_{DTS}/32$
		4'b1111	8	
3:2	CH2CAPPSC[1:0]	Channel 2 input capture prescaler This bit-field specifies the factor of the prescaler on channel 2 input. The prescaler is reset when CH2EN bit in TIMERx_CHCTL2 register is cleared. 00: Prescaler disabled, capture is done on each channel input edge. 01: The input capture occurs on every 2 channel input edges 10: The input capture occurs on every 4 channel input edges 11: The input capture occurs on every 8 channel input edges		
1:0	CH2MS[1:0]	Channel 2 mode selection Same as output compare mode.		

Channel control register 2 (TIMERx_CHCTL2)

Address offset: 0x20

Reset value: 0x0000 0000

This register can be accessed by half-word(16-bit) or word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH3PERFOREN	CH2PERFOREN	CH1PERFOREN	CH0PERFOREN	Reserved											
rw	rw	rw	rw												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3NP	Reserved	CH3P	CH3EN	CH2NP	Reserved	CH2P	CH2EN	CH1NP	Reserved	CH1P	CH1EN	CH0NP	Reserved	CH0P	CH0EN
rw		rw	rw	rw		rw	rw	rw		rw	rw	rw		rw	rw

Bits	Fields	Descriptions
31	CH3PERFOREN	Channel 3 in the composite PWM or asymmetric PWM mode, the period point match moment O3CPRE level setting Refer to CH0PERFOREN description.
30	CH2PERFOREN	Channel 2 in the composite PWM or asymmetric PWM mode, the period point match moment O2CPRE level setting Refer to CH0PERFOREN description.
29	CH1PERFOREN	Channel 1 in the composite PWM or asymmetric PWM mode, the period point match moment O1CPRE level setting Refer to CH0PERFOREN description.
28	CH0PERFOREN	Channel 0 in the composite PWM or asymmetric PWM mode, the period point match moment O0CPRE level setting 0: disable 1: In the composite PWM0 or asymmetric PWM0 mode, the level will be forced to high at the time of period point matching; In the composite PWM1 or asymmetric

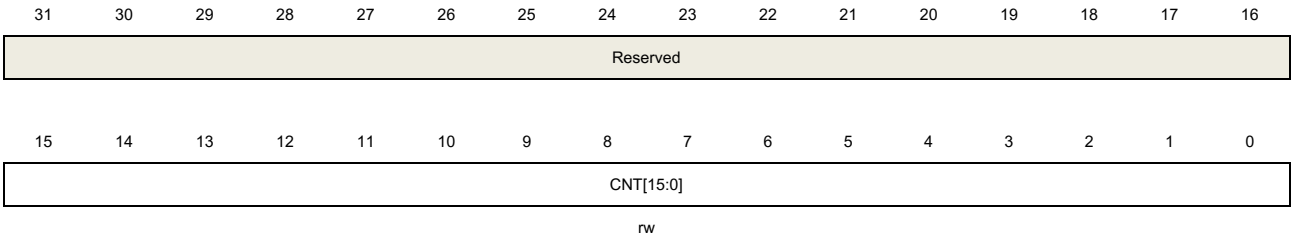
		PWM1 mode, the level at the matching time of the period point is forced to low. **Note:** In the central alignment mode, the period point is the counter underflow time; In the edge-aligned mode, the period point is the counter flow moment.
27:16	Reserved	Must be kept at reset value.
15	CH3NP	Channel 3 complementary capture/compare polarity Refer to CH0NP description.
14	Reserved	Must be kept at reset value.
13	CH3P	Channel 3 capture/compare function polarity Refer to CH0P description
12	CH3EN	Channel 3 capture/compare function enable Refer to CH0EN description
11	CH2NP	Channel 2 complementary capture/compare polarity Refer to CH0NP description.
10	Reserved	Must be kept at reset value.
9	CH2P	Channel 2 capture/compare function polarity Refer to CH0P description
8	CH2EN	Channel 2 capture/compare function enable Refer to CH0EN description
7	CH1NP	Channel 1 complementary capture/compare polarity Refer to CH0NP description.
6	Reserved	Must be kept at reset value.
5	CH1P	Channel 1 capture/compare function polarity Refer to CH0P description
4	CH1EN	Channel 1 capture/compare function enable Refer to CH0EN description
3	CH0NP	Channel 0 complementary output polarity When channel 0 complementary is configured in output mode, this bit must be kept at reset value. When CH0 is configured in input mode, in conjunction with CH0P, this bit is used to define the polarity of CH0. This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 or 10.
2	Reserved	Must be kept at reset value.
1	CH0P	Channel 0 capture/compare function polarity When channel 0 is configured in output mode, this bit specifies the output signal polarity.

		0: Channel 0 active high 1: Channel 0 active low When channel 0 is configured in input mode, this bit specifies the channel 0 input signal polarity. [CH0NP, CH0P] will select the active trigger or capture polarity for CI0FE0 or CI1FE0. 00: CIxFE0 rising edge is the active signal for capture or trigger operation in slave mode. And CIxFE0 will not be inverted. 01: CIxFE0 falling edge is the active signal for capture or trigger operation in slave mode. And CIxFE0 will be inverted. 10: Reserved. 11: Noninverting/both CIxFE0's edges. This bit cannot be modified when PROT[1:0] bit-field in TIMERx_CCHP register is 11 or 10.
0	CH0EN	Channel 0 capture/compare function enable When channel 0 is configured in output mode, setting this bit enables CH0_O signal in active state. When channel 0 is configured in input mode, setting this bit enables the capture event in channel0. 0: Channel 0 disabled 1: Channel 0 enabled

Counter register (TIMERx_CNT)

Address offset: 0x24
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

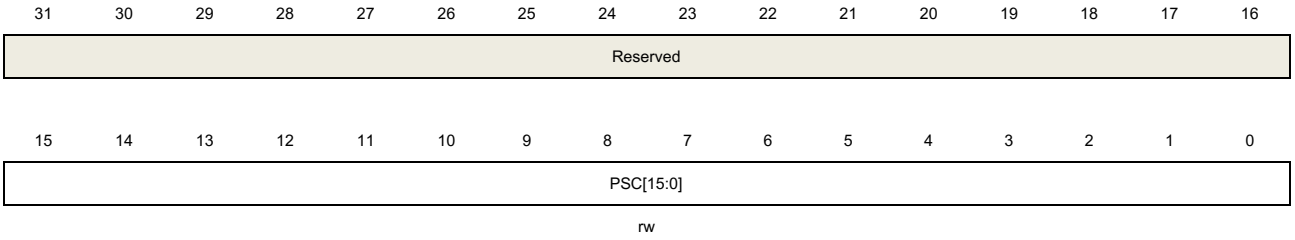


Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CNT[15:0]	This bit-field indicates the current counter value. Writing to this bit-field can change the value of the counter. Note: When a CNT is greater than the CAR, the CNT will be counted to 0xFFFF before clearing.

Prescaler register (TIMERx_PSC)

Address offset: 0x28
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

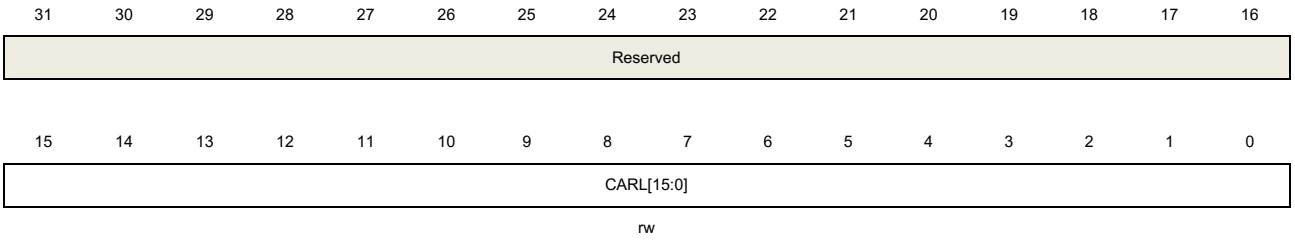


Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	PSC[15:0]	Prescaler value of the counter clock The TIMER_CK clock is divided by (PSC+1) to generate the counter clock. The value of this bit-field will be loaded to the corresponding shadow register at every update event.

Counter auto reload register (TIMERx_CAR)

Address offset: 0x2C
Reset value: 0x0000 FFFF

This register has to be accessed by word (32-bit).



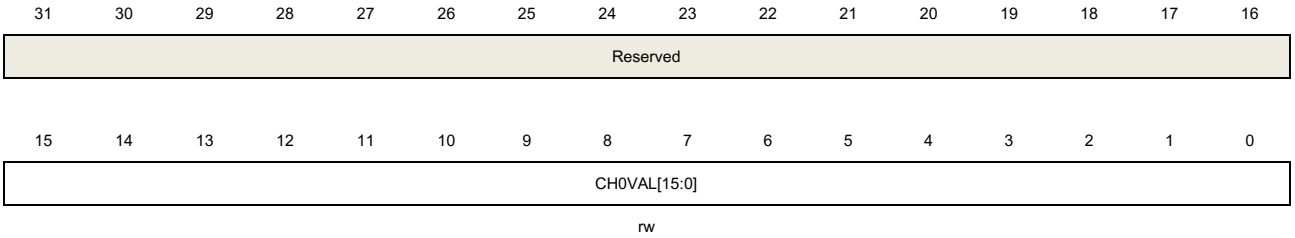
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CARL[15:0]	Counter auto reload value This bit-field specifies the auto reload value of the counter. **Note:** When the timer is configured in input capture mode, this register must be configured a non-zero value (such as 0xFFFF) which is larger than user expected value. When the timer is configured in bit output mode, this register should avoid setting the CAR to 0.

Channel 0 capture/compare value register (TIMERx_CH0CV)

Address offset: 0x34

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



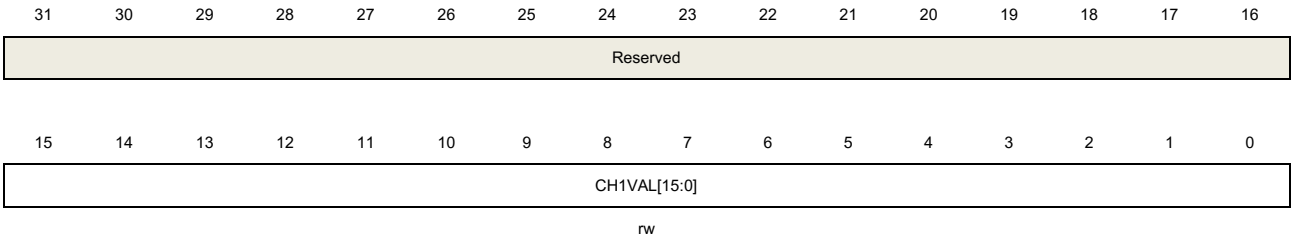
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH0VAL[15:0]	Capture/compare value of channel 0 When channel 0 is configured in input mode, this bit-field indicates the counter value at the last capture event. And this bit-field is read-only. When channel 0 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event.

Channel 1 capture/compare value register (TIMERx_CH1CV)

Address offset: 0x38

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH1VAL[15:0]	Capture/compare value of channel 1 When channel 1 is configured in input mode, this bit-field indicates the counter value at the last capture event. And this bit-field is read-only. When channel 1 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event.

Channel 2 capture/compare value register (TIMERx_CH2CV)

Address offset: 0x3C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH2VAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH2VAL[15:0]	Capture/compare value of channel 2 When channel 2 is configured in input mode, this bit-field indicates the counter value at the last capture event. And this bit-field is read-only. When channel 2 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event.

Channel 3 capture/compare value register (TIMERx_CH3CV)

Address offset: 0x40

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3VAL[15:0]															
rw															

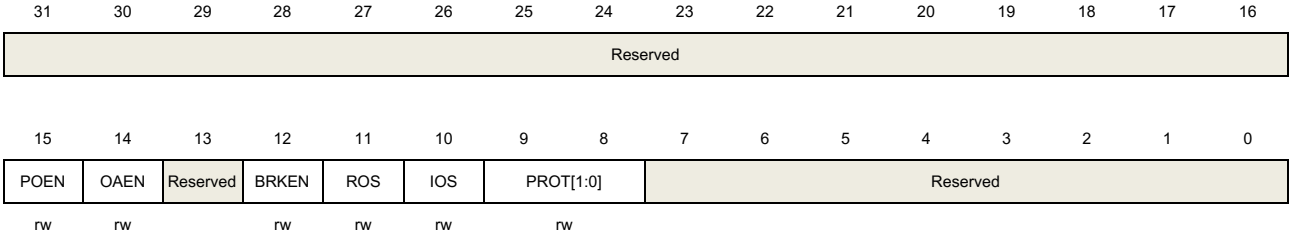
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH3VAL[15:0]	Capture/compare value of channel 3 When channel 3 is configured in input mode, this bit-field indicates the counter value at the last capture event. And this bit-field is read-only. When channel 3 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event.

Complementary channel protection register (TIMERx_CCHP)

Address offset: 0x44

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	POEN	Primary output enable The bit can be set to 1 by: - Write 1 to this bit - If OAEN is set to 1, this bit is set to 1 at the next update event. The bit can be cleared to 0 by: - Write 0 to this bit - Valid fault input. When one of channels is configured in output mode, setting this bit enables the channel outputs (CHx_O) if the corresponding enable bits (CHxEN in TIMERx_CHCTL2 register) have been set. 0: Disable channel outputs (CHxO). 1: Enabled channel outputs (CHxO). **Note:** This bit is only valid when CHxMS=2'b00
14	OAEN	Output automatic enable 0: POEN bit can only be set by software. 1: POEN can be set at the next update event, if the break input is not active. This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.
13	Reserved	Must be kept at reset value.
12	BRKEN	Counter freeze configuration enable This bit can be set to enable the counters freeze configurations. 0: Freeze counter configuration disabled 1: Freeze counter configuration enabled This bit can be modified only when PROT[1:0] bit-field in TIMERx_CCHP register is 00.
11	ROS	Run mode "off-state" enable When POEN bit is set (Run mode), this bit can be set to enable the "off-state" for

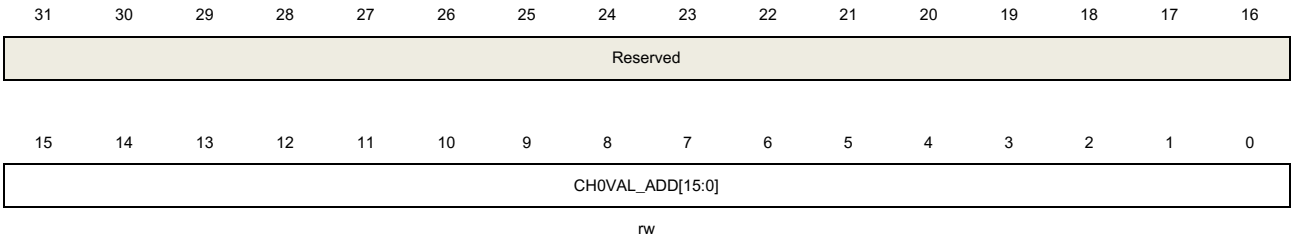
		the channels which has been configured in output mode. 0: "off-state" disabled. If the CHxEN bit is reset, the corresponding channel is output disabled. 1: "off-state" enabled. If the CHxEN bit is reset, the corresponding channel is "off-state".
10	IOS	Idle mode "off-state" enable When POEN bit is reset (Idle mode), this bit can be set to enable the "off-state" for the channels which has been configured in output mode. 0: "off-state" disabled. If the CHxEN bits are both reset, the channels are output disabled. 1: "off-state" enabled. No matter the CHxEN bits, the channels are "off-state".
9:8	PROT[1:0]	Complementary register protect control This bit-field specifies the write protection property of registers. 00: Protect disabled. No write protection. 01: PROT mode 0. The ISOx bits in TIMERx_CTL1 register, the BRKEN /OAEN bits in TIMERx_CCHP register. 10: PROT mode 1. In addition to the registers in PROT mode 0, the CHxP bits in TIMERx_CHCTL2 register (if related channel is configured in output mode) , the ROS/IOS bits in TIMERx_CCHP register. 11: PROT mode 2. In addition to the registers in PROT mode 1, the CHxCOMCTL/ CHxCOMSEN/ CHxCOMADDSEN bits in TIMERx_CHCTL0/1 /1 registers (if the related channel is configured in output) are writing protected. This bit-field can be written only once after the system reset. Once the TIMERx_CCHP register has been written, this bit-field will be writing protected.
7:0	Reserved	Must be kept at reset value.

Channel 0 additional capture/compare value register (TIMERx_CH0CV_ADD)

Address offset: 0x64

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH0VAL_ADD	Additional capture/compare value of channel 0

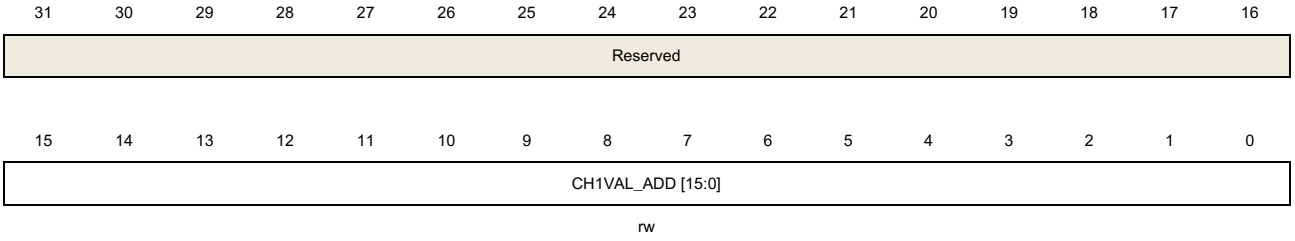
	[15:0]	When channel 0 is configured in input mode, this bit-field indicates the counter value of capture event. And this bit-field is read-only. When channel 0 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event. **Note:** This register just used in composite PWM mode(when CH0CPWMEN=1).

Channel 1 additional capture/compare value register (TIMERx_CH1CV_ADD)

Address offset: 0x68

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



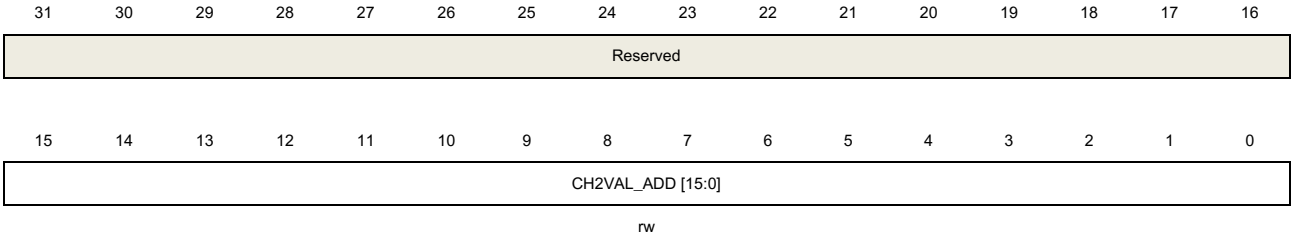
Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH1VAL_ADD [15:0]	Additional capture/compare value of channel 1 When channel 1 is configured in input mode, this bit-field indicates the counter value of capture event. And this bit-field is read-only. When channel 1 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event. **Note:** This register just used in composite PWM mode(when CH1CPWMEN=1).

Channel 2 additional capture/compare value register (TIMERx_CH2CV_ADD)

Address offset: 0x6C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions

31:16	Reserved	Must be kept at reset value.
15:0	CH2VAL_ADD [15:0]	Additional capture/compare value of channel 2 When channel 2 is configured in input mode, this bit-field indicates the counter value of capture event. And this bit-field is read-only. When channel 2 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event. **Note:** This register just used in composite PWM mode(when CH2CPWMEN=1).

Channel 3 additional capture/compare value register (TIMERx_CH3CV_ADD)

Address offset: 0x70

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3VAL_ADD [15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CH3VAL_ADD [15:0]	Additional capture/compare value of channel 3 When channel 3 is configured in input mode, this bit-field indicates the counter value of capture event. And this bit-field is read-only. When channel 3 is configured in output mode, this bit-field contains value to be compared to the counter. When the corresponding shadow register is enabled, the shadow register updates by every update event. **Note:** This register just used in composite PWM mode(when CH3CPWMEN=1).

Control register 2 (TIMERx_CTL2)

Address offset: 0x74

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												DECDISDEN	DECJDEN	Reserved	
												rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:20	Reserved	Must be kept at reset value.
19	DECDISDEN	Quadrature decoder signal disconnection detection enable 0: Quadrature decoder signal disconnection detection is disabled 1: Quadrature decoder signal disconnection detection is enabled
18	DECJDEN	Quadrature decoder signal jump (the two signals jump at the same time) detection enable 0: Quadrature decoder signal jump detection is disabled 1: Quadrature decoder signal jump detection is enabled
17:0	Reserved	Must be kept at reset value.

Watchdog counter period register (TIMERx_WDGPER)

Address offset: 0xA0

Reset value: 0xFFFF FFFF

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WDGPER[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDGPER[15:0]															
rw															

Bits	Fields	Descriptions
31:0	WDGPER[31:0]	Watchdog counter period value This register contains the period of the two watchdog counter. When the counters continues to count to this value, the counter will timeout and the interrupt flag DECDISIF is set. If DECDISIE=1, the corresponding interrupt is generated. **Note:** This register is just used in quadrature decoder signal disconnection detection function(with DECDISDEN =1).

Counter initial control register (TIMERx_CINITCTL)

Address offset: 0xA4

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													SWSYNCG	CINITDIR	CINITVEN
													rw	rw	rw

Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2	SWSYNCG	Software synchronization event generation This bit is set by software and cleared by hardware automatically. Reads this bit always return a 0. 0: No affect. 1: Generate software synchronization event
1	CINITDIR	Counter initial count direction 0: When the synchronization event occurs, the counter count up. 1: When the synchronization event occurs, the counter count down. This bit indicates the initial direction of the counter after a synchronization event occurs and the counter initial value is loaded from the TIMERx_CINITV register. **Note**: This bit is only used when the CAM[1:0] != 00.
0	CINITVEN	Counter initial value register enable 0: Counter initial value register disable. The counter register can't load from the counter initial value register. 1: Counter initial value register enable. When a synchronization event (soft synchronization event generated by setting the SWSYNCG bit or) occurs, the counter register can load from the counter initial value register.

Counter initial value register (TIMERx_CINITV)

Address offset: 0xA8

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CINITVAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	CINITVAL[15:0]	Counter initial value This bits-field indicates the counter initial value.

When the CINITVEN bit is 0, the counter register can't load the initial value which are set in CINITVAL bits-field.

When the CINITVEN bit is 1, when the synchronization event occurs, the counter register can load the initial value which are set in CINITVAL bits-field.

Channel multi-edge capture configuration register (TIMERx_CHMECAPCFG)

Address offset: 0xAC

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				CH3MECP[1:0]		CH2MECP[1:0]		CH1MECP[1:0]		CH0MECP[1:0]		CH3MECEN	CH2MECEN	CH1MECEN	CH0MECEN
				rw		rw		rw		rw		rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CH3MEBL[1:0]		CH2MEBL[1:0]		CH1MEBL[1:0]		CH0MEBL[1:0]	
								rw		rw		rw		rw	

Bits	Fields	Descriptions
31:28	Reserved	Must be kept at reset value.
27:26	CH3MECP[1:0]	Channel 3 multi-edge capture polarity. Refer to CH0MECP [1:0] description.
25:24	CH2MECP[1:0]	Channel 2 multi-edge capture polarity. Refer to CH0MECP [1:0] description.
23:22	CH1MECP[1:0]	Channel 1 multi-edge capture polarity. Refer to CH0MECP [1:0] description.
21:20	CH0MECP[1:0]	Channel 0 multi edge capture polarity. This bit defines the edge polarity of the captured signal in the two-sided edge capture mode of the channel 00: The effective edge of the first capture is the rising edge 01: The effective edge captured for the first time is the falling edge 10: Reserved 11: The effective edge captured for the first time is the edge of any polarity Note: This register is only valid if [MCH0P, CH0P] is configured as 2 'b11. This bit cannot be changed when PROT [1:0]=11 or 10 of the TIMERx_CCHP register.When CH0MS = 3 'b011, the configuration of this bit is meaningless
19	CH3MECEN	Channel 3 multi edge capture enable Refer to CH0MECEN description.
18	CH2MECEN	Channel 2 multi edge capture enable Refer to CH0MECEN description.

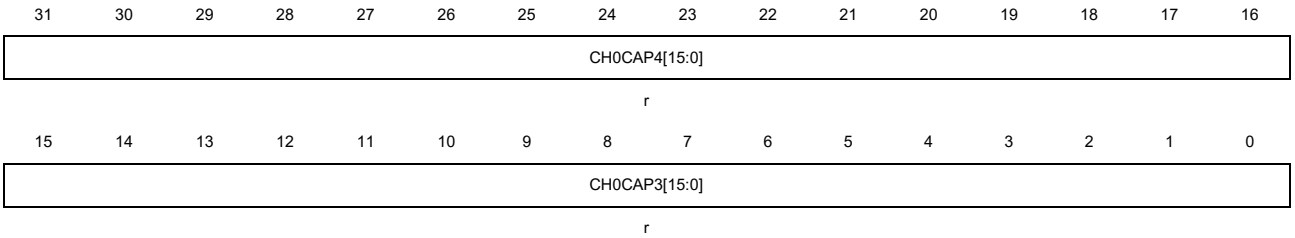
17	CH1MECEN	Channel 1 multi edge capture enable Refer to CH0MECEN description.
16	CH0MECEN	Channel 0 multi edge capture enable This bit is only valid if the channel is configured for input capture mode. If enabled, a channel capture/compare interrupt is triggered only after several capture behaviors have occurred. 0: enable 1: disable
15:8	Reserved	Must be kept at reset value.
7:6	CH3MEBL[1:0]	Channel 3 multi edge buffer level Refer to CH0MEBL[1:0] description.
5:4	CH2MEBL[1:0]	Channel 2 multi edge buffer level Refer to CH0MEBL[1:0] description.
3:2	CH1MEBL[1:0]	Channel 1 multi edge buffer level Refer to CH0MEBL[1:0] description.
1:0	CH0MEBL[1:0]	Channel 0 multi edge buffer level 00: A capture interrupt is occur after one capture. 01: A capture interrupt is occur after two capture. 10: A capture interrupt is occur after three capture. 10: A capture interrupt is occur after four capture.

Channel 0 multi-edge capture register(TIMERx_CH0MECAP)

Address offset: 0xB4

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:16	CH0CAP4[15:0]	Channel 0 captures 4 count values When channel 0 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the fourth capture edge of the last capture event. **Note**: And this register is read-only.
15:0	CH0CAP3[15:0]	Channel 0 captures 3 count values

When channel 0 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the third capture edge of the last capture event.

Note: And this register is read-only.

Channel 1 multi-edge capture register(TIMERx_CH1MECAP)

Address offset: 0xB8

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH1CAP4[15:0]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1CAP3[15:0]															
r															

Bits	Fields	Descriptions
31:16	CH1CAP4[15:0]	Channel 1 captures 4 count values When channel 1 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the fourth capture edge of the last capture event. **Note**: And this register is read-only.
15:0	CH1CAP3[15:0]	Channel 1 captures 3 count values When channel 1 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the third capture edge of the last capture event. **Note**: And this register is read-only.

Channel 2 multi-edge capture register(TIMERx_CH2MECAP)

Address offset: 0xBC

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH2CAP4[15:0]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH2CAP3[15:0]															
r															

Bits	Fields	Descriptions

31:16	CH2CAP4[15:0]	Channel 2 captures 4 count values When channel 2 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the fourth capture edge of the last capture event. **Note**: And this register is read-only.
15:0	CH2CAP3[15:0]	Channel 2 captures 3 count values When channel 2 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the third capture edge of the last capture event. **Note**: And this register is read-only.

Channel 3 multi-edge capture register(TIMERx_CH3MECAP)

Address offset: 0xC0
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH3CAP4[15:0]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3CAP3[15:0]															
r															

Bits	Fields	Descriptions
31:16	CH3CAP4[15:0]	Channel 3 captures 4 count values When channel 3 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the fourth capture edge of the last capture event. **Note**: And this register is read-only.
15:0	CH3CAP3[15:0]	Channel 3 captures 3 count values When channel 3 is configured in input mode and multi edge capture is enabled, these bits contain the counter comparison value for the third capture edge of the last capture event. **Note**: And this register is read-only.

DMA configuration register (TIMERx_DMACFG)

Address offset: 0xE0
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DMATC[5:0]						Reserved		DMATA[5:0]					
		rw								rw					

Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13:8	DMATC[5:0]	DMA transfer count This filed defines the number(n) of the register that DMA will access(R/W), n = (DMATC [5:0] +1). DMATC [5:0] is from 6'b000000 to 6'b010001.
7:6	Reserved	Must be kept at reset value.
5:0	DMATA[5:0]	DMA transfer access start address This field defines the start address of accessing the TIMERx_DMATB register by DMA. When the first access to the TIMERx_DMATB register is done, this bit-field specifies the address just accessed. And then the address of the second access to the TIMERx_DMATB register will be (start address + 0x4). In a word: start address = TIMERx_CTL0 + DMATA*4

DMA transfer buffer register (TIMERx_DMATB)

Address offset: 0xE4

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMATB[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	DMATB[15:0]	DMA transfer buffer When a read or write operation is assigned to this register, the register located at the address ranges from (start address) to (start address + transfer count * 4) will be accessed. The transfer count is calculated by hardware, and ranges from 0 to DMATC.

Input remap register (TIMERx_IRMP) (x=1, 2)

Address offset: 0x50

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														CI0_RMP[1:0]	

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1:0	CI0_RMP[1:0]	Channel 0 input remap. 00: GPIO pin. Refer to GPIO remap table 01: LXTAL 10: HXTAL 11: CKOUT0SEL for TIMER1, CKOUT1SEL for TIMER2

Configuration register (TIMERx_CFG)

Address offset: 0xFC

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											RDSEL	Reserved		CHVSEL	OUTSEL
											rw			rw	rw

Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
4	RDSEL	This bit determines the value of TIMERx_PSC, TIMERx_CAR, TIMERx_CHyCV, TIMERx_MCHyCV, TIMERx_CHyCV_ADD when reading the register. 0: Current configuration value 1: The shadow register value in effect
3:2	Reserved	Must be kept at reset value.
1	CHVSEL	Write CHxVAL register selection bit This bit-field is set and reset by software. 1: If the value to be written to the CHxVAL register is the same as the value of CHxVAL register, the write access is ignored.

		0: No effect.
0	OUTSEL	The output value selection bit This bit-field is set and reset by software. 1: If POEN bit and IOS bit are 0, the output is disabled. 0: No effect.

17.3. Basic timer (TIMERx, x=5, 6)

17.3.1. Overview

The basic timer module (TIMER5/6) has a 32-bit counter that can be used as an unsigned counter. The basic timer can be configured to generate a DMA request and a TRGO to connect to DAC.

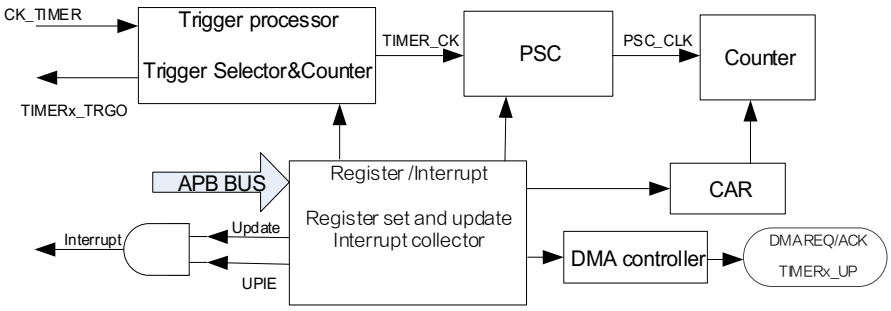
17.3.2. Characteristics

- Counter width: 32 bits.
- Source of count clock is internal clock only.
- counter modes: count up.
- Programmable prescaler: 16 bits. The factor can be changed ongoing.
- Auto reload function.
- Interrupt output or DMA request: update event.

17.3.3. Block diagram

Figure 17-77. Basic timer block diagram provides details on the internal configuration of the basic timer.

Figure 17-77. Basic timer block diagram



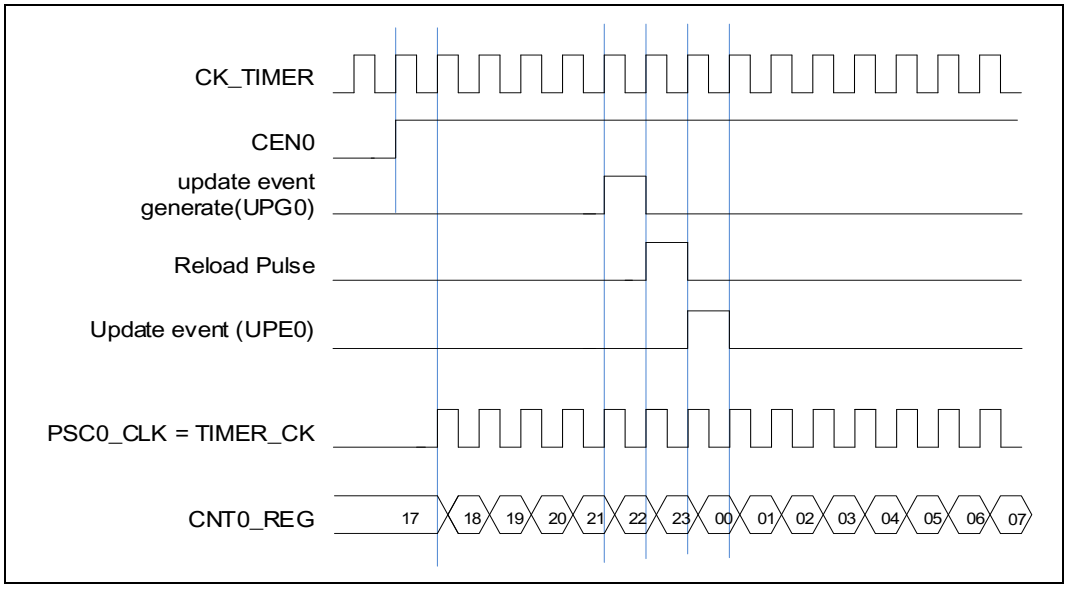
17.3.4. Function overview

Clock source configuration

The basic TIMER can only be clocked by the internal timer clock CK_TIMER, which is from the source named CK_TIMER in RCU

The TIMER_CK, driven counter's prescaler to count, is equal to CK_TIMER used to drive the counter prescaler. When the CENx(x=0,1,2,3) is set, the CK_TIMER will be divided by PSC value to generate PSC_CLK.

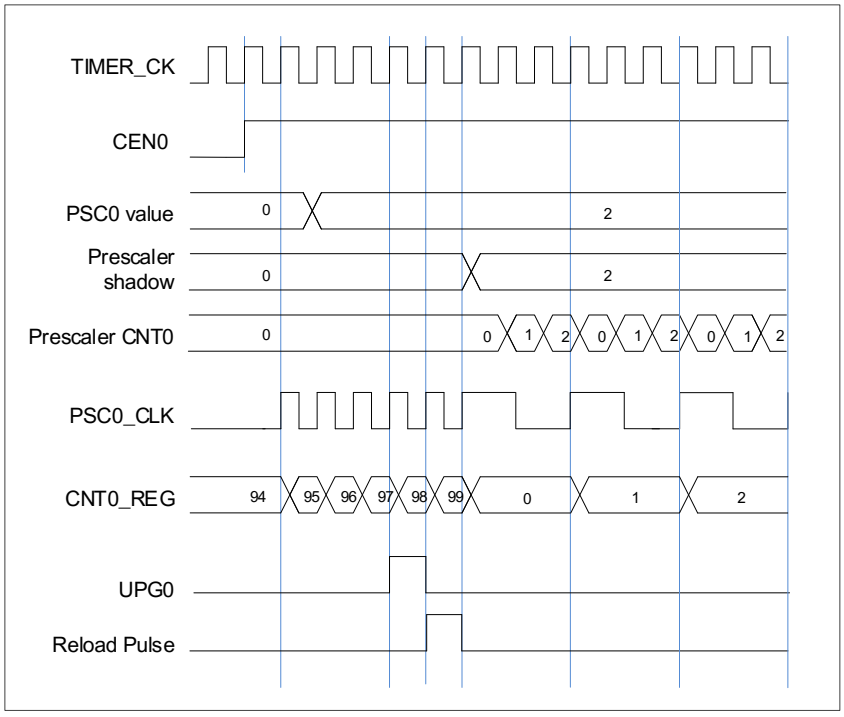
Figure 17-78. Timing chart of internal clock divided by 1 for channel 0



Clock prescaler

The counter clock (PSC_CK) is obtained by the TIMER_CK through the prescaler, and the prescale factor can be configured from 1 to 65536 through the prescaler register (TIMERx_PSCy(y=0,1,2,3)). The new written prescaler value will not take effect until the next update event.

Figure 17-79. Timing chart of PSC value change from 0 to 2 for channel 0



Counter up counting

In this mode, the counter counts up continuously from 0 to the counter reload value, which is

defined in the TIMERx_CARy(y=0,1,2,3) register, in a count-up direction. Once the counter reaches the counter reload value, the counter will start counting up from 0 again. The update event is generated at each counter overflow.

When the update event is set by the UPGx(x=0,1,2,3) bit in the TIMERx_SWEVG register, the counter value will be initialized to 0 and generates an update event.

If the UPDISx(x=0,1,2,3) bit in TIMERx_CTL0 register is set, the update event is disabled.

When an update event occurs, all the shadow registers (auto reload register, prescaler register) are updated.

The following figures show some examples of the counter behavior for different clock prescaler factor when TIMERx_CAR=0x99.

Figure 17-80. Timing chart of up counting mode, PSC=0/2 for channel 0

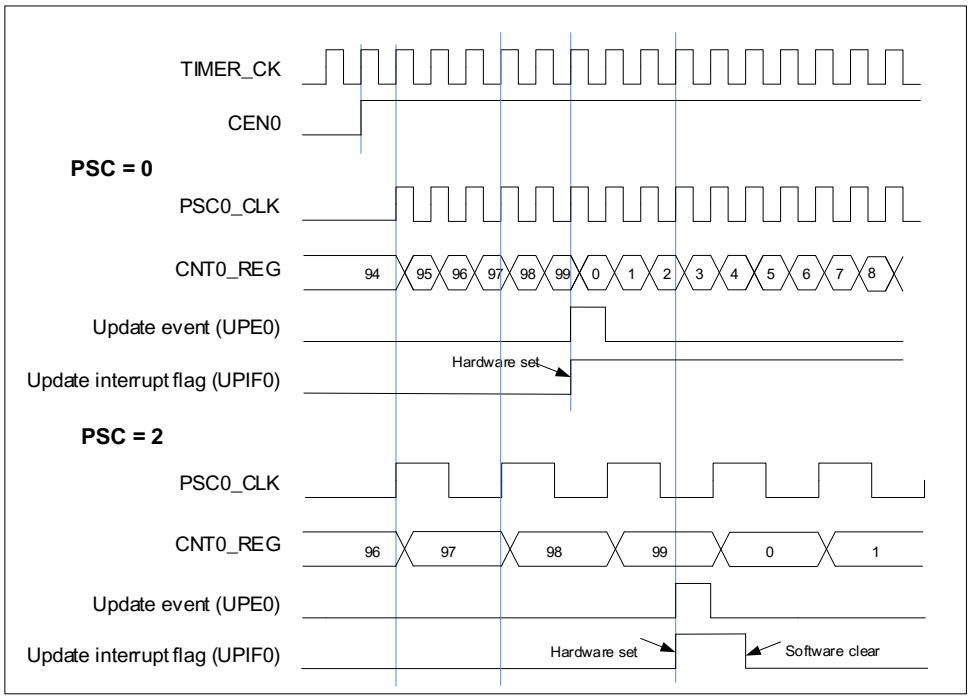
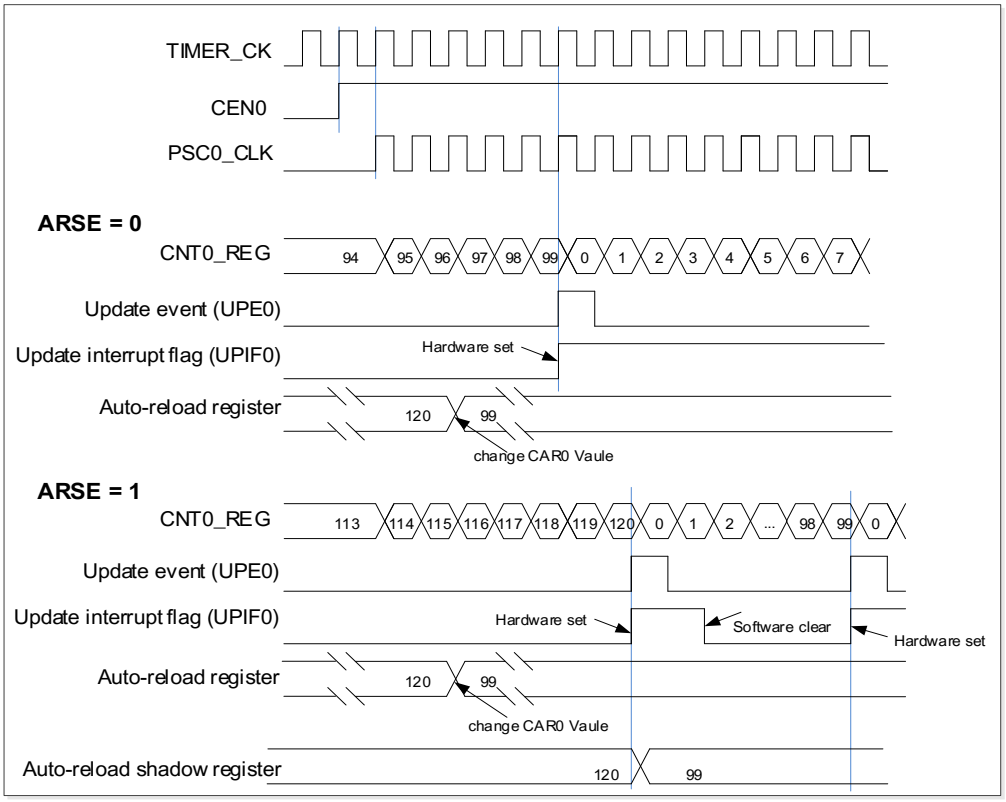


Figure 17-81. Timing chart of up counting mode, change TIMERx_CAR on the go for

channel 0



Single pulse mode

Single pulse mode is opposite to the repetitive mode, which can be enabled by setting SPMx(x=0,1,2,3) in TIMERx_CTL0. When you set SPMx, the counter will be clear and stop when the next update event.

Once the timer is set to operate in the single pulse mode, it is necessary to set the timer enable bit CENx(x=0,1,2,3) in the TIMERx_CTL0 register to 1 to enable the counter, then the CENx bit keeps at a high state until the update event occurs or the CENx bit is written to 0 by software. If the CENx bit is cleared to 0 using software, the counter will be stopped and its value held.

Cascade mode

Basic timer supports cascade two channels together to form a 64 bit count output. The cascade mode is configured in TIMERx_CTL1 by setting CASCx(x=0,1,2,3). When the count of a channel in the cascade mode is overflowed, the count of next channel is added to 1.

Timer debug mode

When the Cortex®-M7 core is halted, and the TIMERx_HOLD configuration bit in DBG_CTL0 register set to 1, the TIMERx counter stops.

17.3.5. Registers definition (TIMERx, x=5, 6)

TIMER5 base address: 0x4001 6000

TIMER6 base address: 0x4000 1400

Table 17-15. TIMERx register overview

Register Name	Symbol	Address Offset	Reset Value
Control register 0	TIMERx_CTL0	0x00	0x0000 0000
Control register 1	TIMERx_CTL1	0x04	0x0000 0000
DMA and interrupt enable register	TIMERx_DMAINTEN	0x0C	0x0000 0000
Interrupt flag register	TIMERx_INTF	0x10	0x0000 0000
Software event generation register	TIMERx_SWEVG	0x14	0x0000 0000
Channel 0 counter register	TIMERx_CNT0	0x18	0x0000 0000
Channel 1 counter register	TIMERx_CNT1	0x1C	0x0000 0000
Channel 2 counter register	TIMERx_CNT2	0x20	0x0000 0000
Channel 3 counter register	TIMERx_CNT3	0x24	0x0000 0000
Prescaler register 0	TIMERx_PSC0	0x28	0x0000 0000
Prescaler register 1	TIMERx_PSC1	0x2C	0x0000 0000
Prescaler register 2	TIMERx_PSC2	0x30	0x0000 0000
Prescaler register 3	TIMERx_PSC3	0x34	0x0000 0000
Counter auto reload register 0	TIMERx_CAR0	0x38	0xFFFF FFFF
Counter auto reload register 1	TIMERx_CAR1	0x3C	0xFFFF FFFF
Counter auto reload register 2	TIMERx_CAR2	0x40	0xFFFF FFFF
Counter auto reload register 3	TIMERx_CAR3	0x40	0xFFFF FFFF
Cascade timer high register	TIMERx_CASCH	0x48	0x0000 0000
Cascade timer low register	TIMERx_CASCL	0x4C	0x0000 0000

Control register 0 (TIMERx_CTL0)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												ARSE3	ARSE2	ARSE1	ARSE0
												rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPM3	SPM2	SPM1	SPM0	UPS3	UPS2	UPS1	UPS0	UPDIS3	UPDIS2	UPDIS1	UPDIS0	CEN3	CEN2	CEN1	CEN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:20	Reserved	Must be kept at reset value.
19	ARSE3	Auto-reload shadow enable of channel 3. Refer to ARSE0
18	ARSE2	Auto-reload shadow enable of channel 2. Refer to ARSE0
17	ARSE1	Auto-reload shadow enable of channel 1. Refer to ARSE0

16	ARSE0	Auto-reload shadow enable of channel 0. 0: The shadow register for TIMERx_CAR0 register is disabled 1: The shadow register for TIMERx_CAR0 register is enabled
15	SPM3	Single pulse mode of channel 3 Refer to SPM0
14	SPM2	Single pulse mode of channel2 Refer to SPM0
13	SPM1	Single pulse mode of channel 1 Refer to SPM0
12	SPM0	Single pulse mode of channel 0: Single pulse mode is disabled. Counter(TIMERX_CNT0) continues after an update event. 1: Single pulse mode is enabled. The CEN0 bit is cleared by hardware and the counter stops at next update event.
11	UPS3	Update source of channel 3 Refer to UPS0
10	UPS2	Update source of channel 2 Refer to UPS0
9	UPS1	Update source of channel 1 Refer to UPS0
8	UPS0	Update source of channel 0 This bit is used to select the update event sources by software. 0: Any of the following events generates an update interrupt or a DMA request: – The UPG bit is set. – The counter generates an overflow event. – The slave mode controller generates an update event. 1: Only counter overflow generates an update interrupt or a DMA request.
7	UPDIS3	Update disable of channel 3 Refer to UPDIS0
6	UPDIS2	Update disable of channel 2 Refer to UPDIS0
5	UPDIS1	Update disable of channel 1 Refer to UPDIS0
4	UPDIS0	Update disable of channel 0 This bit is used to enable or disable the update event generation. 0: Update event enable. The update event is generated and the buffered registers are loaded with their preloaded values when one of the following events occurs: – The UPG bit is set. – The counter generates an overflow event. – The slave mode controller generates an update event.

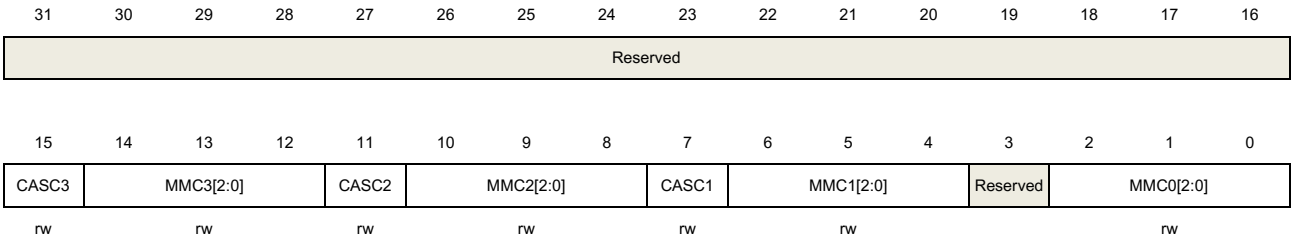
		1: Update event disable. The buffered registers keep their value, while the counter and the prescaler are reinitialized if the UG bit is set or the slave mode controller generates a hardware reset event.
3	CEN3	Counter enable of channel 3 Refer to CEN0
2	CEN2	Counter enable of channel 2 Refer to CEN0
1	CEN1	Counter enable of channel 1 Refer to CEN0
0	CEN0	Counter enable of channel 0 0: Counter disable 1: Counter enable The CEN0 bit must be set by software when timer works in external clock mode, pause mode or encoder mode. While in event mode, the hardware can set the CEN0 bit automatically.

Control register 1 (TIMERx_CTL1)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	CASC3	Cascade mode of channel 3 enable. Refer to CASC1.
14:12	MMC3[2:0]	Master mode control of channel 3. Refer to MMC0.
11	CASC2	Cascade mode of channel 2 enable. Refer to CASC1.
10:8	MMC2[2:0]	Master mode control of channel 2. Refer to MMC0.
7	CASC1	Cascade mode of channel 1 enable. 0: Cascade mode disable 1: Cascade mode enable

		When the count of channel 0 overflows, the count of channel 1 increases by 1 to form a 64bit counter and the MMC1 of channel 1 is forcibly set to 3 'b010 in cascade mode.
6:4	MMC1[2:0]	Master mode control of channel 1. Refer to MMC0.
3	Reserved	Must be kept at reset value.
2:0	MMC0[2:0]	Master mode control of channel 0. These bits control the selection of TRGO signal of channel 0, which is sent by master timer to slave timer for synchronization function. 000: When a counter reset event occurs, a TRGO trigger signal is output. The counter reset source: Master timer generate a reset the UPG bit in the TIMERx_SWEVG register is set 001: Enable. When a counter start event occurs, a TRGO trigger signal is output. The counter start source : CEN control bit is set The trigger input in pause mode is high 010: When an update event occurs, a TRGO trigger signal is output. The update source depends on UPDIS0 bit and UPS0 bit. 100~111: Reserved.

Interrupt enable register (TIMERx_DMAINTEN)

Address offset: 0x0C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				UPDEN3	UPDEN2	UPDEN1	UPDEN0	Reserved				UPIE3	UPIE2	UPIE1	UPIE0
				rw	rw	rw	rw					rw	rw	rw	rw

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11	UPDEN3	Update DMA request enable of channel 3. Refer to UPDEN0.
10	UPDEN2	Update DMA request enable of channel 2. Refer to UPDEN0.
9	UPDEN1	Update DMA request enable of channel 1. Refer to UPDEN0.

8	UPDEN0	Update DMA request enable of channel 0. 0: Disabled 1: Enabled
7:4	Reserved	Must be kept at reset value.
3	UPIE3	Update interrupt enable of channel 3. Refer to UPIE0.
2	UPIE2	Update interrupt enable of channel 2. Refer to UPIE0.
1	UPIE1	Update interrupt enable of channel 1. Refer to UPIE0.
0	UPIE0	Update interrupt enable of channel 0. 0: Disabled 1: Enabled

Interrupt flag register (TIMERx_INTF)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												UPIF3	UPIF2	UPIF1	UPIF0
												rc_w0	rc_w0	rc_w0	rc_w0

Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3	UPIF3	Update interrupt flag of channel 3. Refer to UPIE0.
2	UPIF2	Update interrupt flag of channel 2. Refer to UPIE0.
1	UPIF1	Update interrupt flag of channel 1. Refer to UPIE0.
0	UPIF0	Update interrupt flag of channel 0. This bit is set by hardware when an update event occurs and cleared by software. 0: No update interrupt occurred 1: Update interrupt occurred

Software event generation register (TIMERx_SWEVG)

Address offset: 0x14

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												UPG3	UPG2	UPG1	UPG0
												w	w	w	w

Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3	UPG3	Refer to UPG0.
2	UPG2	Refer to UPG0.
1	UPG1	Refer to UPG0.
0	UPG0	This bit can be set by software, and automatically cleared by hardware. When this bit is set, the counter is cleared. The prescaler counter is cleared at the same time. 0: No generate an update event 1: Generate an update event

Channel 0 counter register (TIMERx_CNT0)

Address offset: 0x18

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT0[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT0[15:0]															
rw															

Bits	Fields	Descriptions
31:0	CNT0[31:0]	This bit-field indicates the current counter value of channel 0 . Writing to this bit-field can change the value of the counter.

Channel 1 counter register (TIMERx_CNT1)

Address offset: 0x1C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT1[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT1[15:0]															
rw															

Bits	Fields	Descriptions
31:0	CNT1[31:0]	This bit-field indicates the current counter value of channel 1 . Writing to this bit-field can change the value of the counter.

Channel 2 counter register (TIMERx_CNT2)

Address offset: 0x20

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT2[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT2[15:0]															
rw															

Bits	Fields	Descriptions
31:0	CNT2[31:0]	This bit-field indicates the current counter value of channel 2 . Writing to this bit-field can change the value of the counter.

Channel 3 counter register (TIMERx_CNT3)

Address offset: 0x24

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT3[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT3[15:0]															
rw															

Bits	Fields	Descriptions

31:0	CNT3[31:0]	This bit-field indicates the current counter value of channel 3 . Writing to this bit-field can change the value of the counter.

Prescaler register (TIMERx_PSC0)

Address offset: 0x28
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC0[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	PSC0[15:0]	Prescaler value of the counter clock of channel 0 The TIMER_CK clock is divided by (PSC0+1) to generate the counter clock. The value of this bit-field will be loaded to the corresponding shadow register at every update event.

Prescaler register (TIMERx_PSC1)

Address offset: 0x2C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC1[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	PSC1[15:0]	Prescaler value of the counter clock of channel 1 The TIMER_CK clock is divided by (PSC1+1) to generate the counter clock. The value of this bit-field will be loaded to the corresponding shadow register at every update event.

Prescaler register (TIMERx_PSC2)

Address offset: 0x30

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC2[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	PSC2[15:0]	Prescaler value of the counter clock of channel 2 The TIMER_CK clock is divided by (PSC2+1) to generate the counter clock. The value of this bit-field will be loaded to the corresponding shadow register at every update event.

Prescaler register (TIMERx_PSC3)

Address offset: 0x34

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC3[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	PSC3[15:0]	Prescaler value of the counter clock of channel 0 The TIMER_CK clock is divided by (PSC3+1) to generate the counter clock. The value of this bit-field will be loaded to the corresponding shadow register at every update event.

Counter auto reload register (TIMERx_CAR0)

Address offset: 0x38

Reset value: 0xFFFF FFFF

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CARL0[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CARL0[15:0]															
rw															

Bits	Fields	Descriptions
31:0	CARL0[31:0]	Counter auto reload value of channel 0. This bit-field specifies the auto reload value of the counter. **Note:** When the timer is configured in input capture mode, this register must be configured a non-zero value (such as 0xFFFFFFFF) which is larger than user expected value.

Counter auto reload register (TIMERx_CAR1)

Address offset: 0x3C
Reset value: 0xFFFF FFFF

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CARL1[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CARL1[15:0]															
rw															

Bits	Fields	Descriptions
31:0	CARL1[31:0]	Counter auto reload value of channel 1. This bit-field specifies the auto reload value of the counter. **Note:** When the timer is configured in input capture mode, this register must be configured a non-zero value (such as 0xFFFFFFFF) which is larger than user expected value.

Counter auto reload register (TIMERx_CAR2)

Address offset: 0x40
Reset value: 0xFFFF FFFF

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CARL0[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

CARL2[15:0]
rw

Bits	Fields	Descriptions
31:0	CARL2[31:0]	Counter auto reload value of channel 2. This bit-field specifies the auto reload value of the counter. **Note:** When the timer is configured in input capture mode, this register must be configured a non-zero value (such as 0xFFFFFFFF) which is larger than user expected value.

Counter auto reload register (TIMERx_CAR3)

Address offset: 0x44
Reset value: 0xFFFF FFFF

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CARL0[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CARL3[15:0]															
rw															

Bits	Fields	Descriptions
31:0	CARL3[31:0]	Counter auto reload value of channel 3. This bit-field specifies the auto reload value of the counter. **Note:** When the timer is configured in input capture mode, this register must be configured a non-zero value (such as 0xFFFFFFFF) which is larger than user expected value.

Cascade timer high register (TIMERx_CASCH)

Address offset: 0x48
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CASCH[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CASCH[15:0]															
r															

Bits	Fields	Descriptions

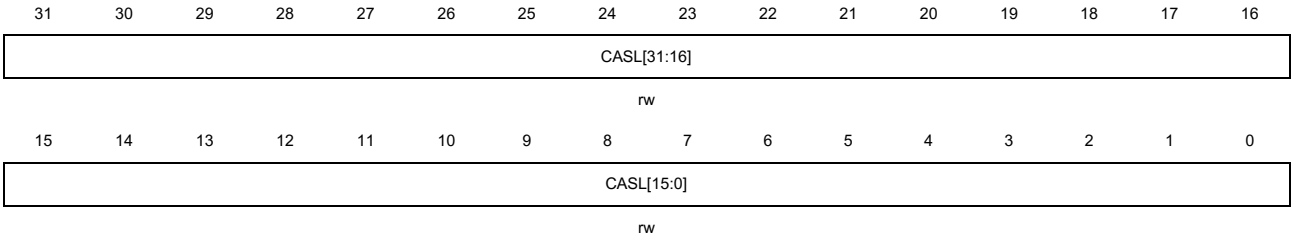
31:0	CASCH[31:0]	These bits store the high 32 bits of the 64bit counter in cascade mode.

Cascade timer low register (TIMERx_CASCL)

Address offset: 0x4C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



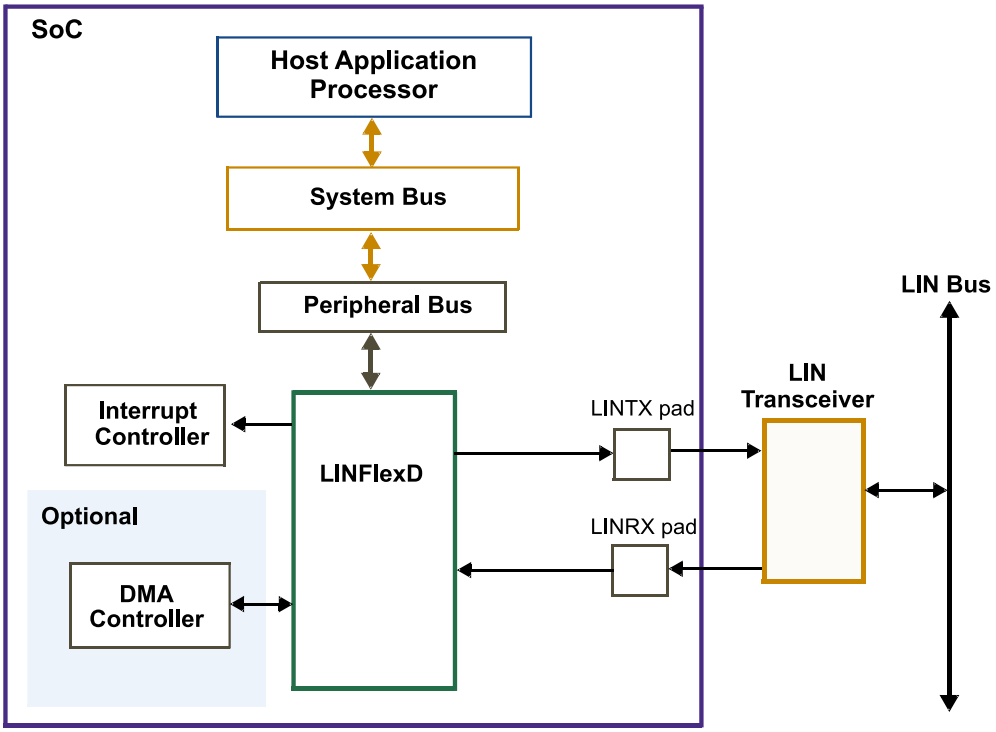
Bits	Fields	Descriptions
31:0	CASL[31:0]	These bits store the low 32 bits of the 64bit counter in cascade mode.

18. Local Interconnect Network Flexible Driver (LINFlexD)

18.1. Overview

The LINFlexD Controller is a semiconductor IP block supporting the Local Interconnect Network(LIN) protocol for both master and slave applications.

Figure 18-1 Example system using LINFlexD



The LINFlexD manages a high number of LIN messages efficiently with a minimum of CPU load. To reduce the CPU loading in master mode, the LINFlexD autonomously handles the LIN messages after software has triggered the header transmission, until the next header transmission request in transmitter mode or until checksum reception in the receiver mode. The LINFlexD supports LIN protocol versions 1.3, 2.0, 2.1, and 2.2, and provides an 8-byte buffer for transmission/reception data.

The LINFlexD also supports a UART mode for UART transfers of 8-bit,9-bit, or 12-bit data frames.

Optional DMA support is available for both transmit(Tx)and receive (Rx)data

18.2. Characteristics

LINFlexD features common to both LIN and UART modes include:

- Fractional baud rate generator

- Three operating modes for power saving and configuration registers lock
  - Initialization
  - Normal
  - Sleep
- Test mode: Loop Back
- Maskable interrupts

18.2.1. LIN mode features

- Support for LIN protocol versions 1.3,2.0,2.1, and 2.2
- Bit rates up to 20 Kbit/s (LIN protocol)
- Master/slave modes
- Classic and enhanced checksum calculation and check
- Single 8-byte buffer or FIFO for transmission/reception
- Timeout management
- Identifier filters
- DMA interface
- Supports a maximum of 16 possible identifiers
- Master mode with autonomous message handling
- Wakeup event on dominant bit detection
- True LIN field state machine
- Advanced LIN error detection
- Header, response, and frame timeout
  - Slave mode
  - Autonomous header handling
- Identifier filters for autonomous message handling in slave mode
- Separate clock for baud rate calculation

18.2.2. UART mode features

- Full-duplex communication
- Separate clock for baud rate calculation
- Baud rate is a function of baud clock, and the LINIBRR and LINFBRR registers. See "Baud Rate Generation".
- 7/8 bits data, parity
- 1/2/3 stop bits
- 12-bit +parity reception
- 4-byte buffer for reception;4-byte buffer for transmission
- 12-bit counter for timeout management
- The maximum baud rate achievable is 15 Mbit/s.
- For bit rate ≤ CK_LINFlexD / 16 Mbit/s
  - 16 times oversampling
  - 3:1 majority voting
- For CK_LINFlexD /16 Mbit/s < bit rate ≤ CK_LINFlexD / 8 Mbit/s
  - Reduced oversampling programmable by software

- 3:1 majority voting for reduced oversampling of 8 samples per bit

■ For CK_LINFlexD /8 Mbit/s < bit rate ≤ CK_LINFlexD /4 Mbit/s
- Reduced oversampling programmable by software
- 1:1 voting for all reduced oversampling of 4, 5, and 6 samples per bit

18.3. Function overview

18.3.1. LIN protocol

LIN is a serial communication protocol. A LIN cluster consists of one master task and several slave tasks. A master node contains the master task as well as a slave task. All other nodes contain a slave task only. The master node decides when and which frame is transferred on the bus. The slave task provides the data to be transported by the frame.

Frames

A frame consists of a header provided by the master task and a response provided by the slave task. The header consists of a break, followed by a sync pattern, followed by an identifier. The slave task associated with the identifier provides the response. The response consists of a data field and checksum. The slave task designated to receive the data associated with the identifier receives the response and verifies the checksum.

LIN frames are transmitted/received as shown in ***Figure 18-2 LIN frames format*** and ***Figure 18-3 Structure of LIN frame***.

Figure 18-2 LIN frames format

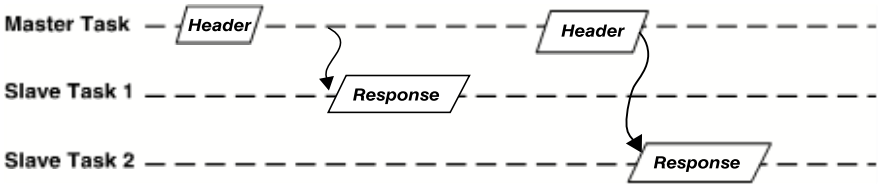
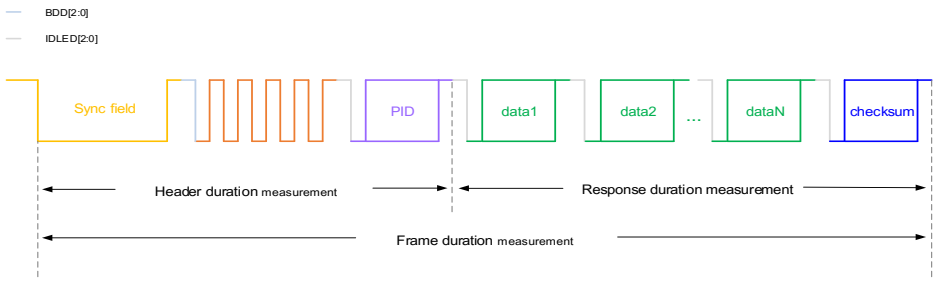


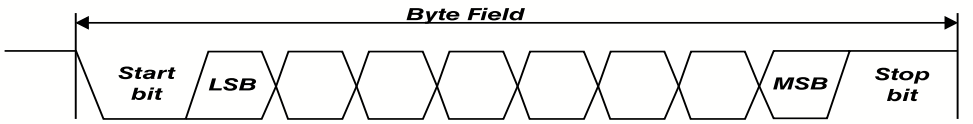
Figure 18-3 Structure of LIN frame



Data field

Each byte is transmitted as shown in ***Figure 18-4 Structure of byte field***. The LSB of the data is sent first and the MSB is sent last. The start bit is encoded as a bit with value zero (dominant)and the stop bit is encoded as bit value one (recessive).

Figure 18-4 Structure of byte field



Break

The break symbol signals the beginning of a new frame. It is the only field that does not comply with ***Figure 18-5 Structure of byte field***. The break is always generated by the master and shall be at least 13 bits of dominant value. Followed by a break delimiter as shown in ***Figure 18-6 Sync byte field***. The break delimiter must be at least one nominal bit time in length.

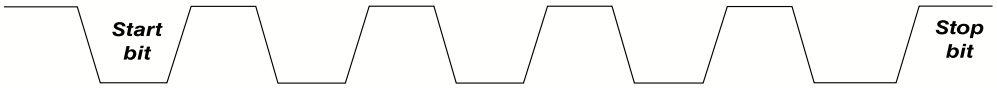
Figure 18-5 Structure of byte field



Sync byte

The sync byte is a byte field with the data value of 0x55.

Figure 18-6 Sync byte field

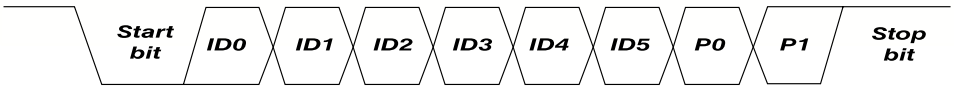


Identifier

The identifier field consists of two sub-fields, the identifier and the identifier parity. Bits 0 to 5 are the identifier. Bits 6 and 7 indicate the parity:

- P0=ID0 XOR ID1 XOR ID2 XOR ID4
- P1=NOT (ID1 XOR ID3 XOR ID4 XOR ID5)

Figure 18-7 Identifier field



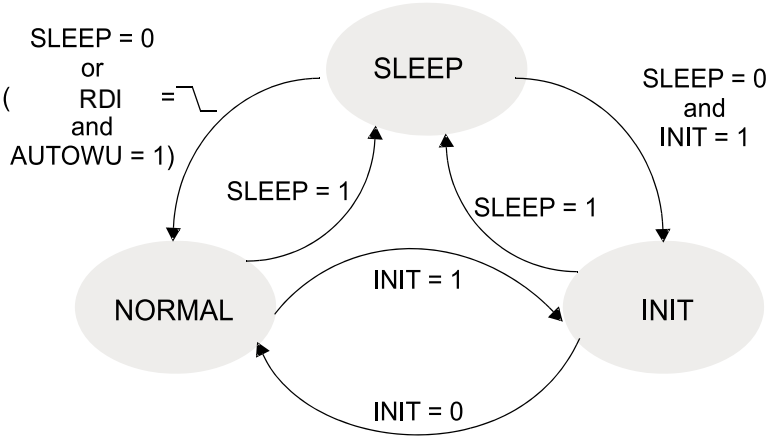
Checksum

The last field of a frame is the checksum. The checksum contains the inverted 8-bit sum with carryover of all data bytes and the identifier. Checksum calculation over the data bytes only is called classic checksum and is used for communication with LIN 1.3(and lower) slaves. Checksum calculation over the data bytes and the protected identifier byte is called enhanced checksum and is used for communication with LIN 2.0(and higher) slaves.

18.3.2. Operating modes

The LINFlexD has three operating modes: Initialization, Normal, and Sleep. After hardware reset, the LINFlexD enters Sleep mode to reduce power consumption. ***Figure 18-8 Operating modes*** shows the state diagram of operating modes.

Figure 18-8 Operating modes



Initialization mode(INIT)

To enter Initialization mode, software sets the INIT bit in LINCR1.To exit Initialization mode, software should reset the INIT bit.

When the LINFlexD is in Initialization mode, all message transfers to and from the LIN bus are stopped and the status of LIN bus output ipp_do_lintx is recessive (high). If software invokes Initialization mode when a bus transfer is in progress, the transfer is aborted.

Software should therefore check the LIN state before setting the INIT bit.

To initialize the LINFlexD Controller, software must:

1. Set up the baud rate registers.
2. Reset the UART bit.
3. Select master or slave mode.
4. Configure checksum control bit.
5. Initialize the identifier list (slave mode).

Normal mode(NM)

After initializing the LINFlexD, software can set the LINFlexD to Normal mode by clearing the INIT bit.

Sleep mode (SM)

Sleep mode in the LINFlexD reduces power consumption. Sleep mode is entered by setting the SLEEP bit in LINCR1.The LINFlexD can be awakened from Sleep mode by clearing the SLEEP bit.

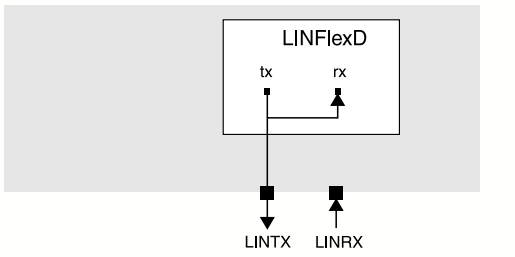
If software detects a wakeup pulse of 150 μs on the LIN Bus, it can request the LINFlexD to wake up from Sleep mode. See *18.3.8 Wakeup management*.

Loop back mode

Loop back mode is entered by setting LINCR1.LBKM. In loop back mode, the LINFlexD receives the identifier and data transmitted by itself and writes the same in the BIDR register and data buffers respectively. Loop back mode is provided for self-test functions. Bit error is checked in Loop Back mode.

While in Loop Back mode, the LINFlexD ignores LINRX signal. There is an internal feedback from the Tx output to the Rx input. The Tx pin signal can be disconnected from the chip-level LINTX pin by a user-implemented GPIO function.

Figure 18-9 LINFLEXD in loop back mode



Master Mode

Master mode is selected by setting the LINCR1.MME bit.

Header Transmission

According to the LIN protocol, any communication on the LIN bus is triggered by the master sending a header. The header is transmitted by the master task of a node, while the response

is transmitted by the slave task of a node.

To transmit the header, first program the identifier(ID), data field length (DFL), message direction (DIR), and classic checksum enable(CCS)in the BIDR register; then set the HTRQ bit in LINCR2.After header transmission starts, do not modify the BIDR register bits until the current frame is complete. The transmitted ID is also received by the master node and copied to BIDR.

Data Transmission

When the master node is the publisher of the data corresponding to the identifier sent by the master, then the slave task of the node should send the data in the response part of the frame. Therefore, software must provide the data to the LINFlexD before the header transmission is requested. The data to be transmitted is stored in the message buffers BDRL and BDRM. The number of bytes to be transmitted depends on BIDR. DFL. Software uses the BIDR.CCS bit to configure the checksum type (classic or enhanced) for each message.

When the response is sent successfully, LINSR.DTF is set. In the case of an error, LINSR.DTF is not set and the corresponding error flag is set in LINSR. (See ***18.3.3Errors*** for more details.)

The direction of the message buffer is controlled by the BIDR.DIR bit. The transmitted data is also received by the same node and copied to the buffer.

Data Reception

To receive data from a slave node, the master sends a header with the corresponding identifier. The data received from the slave is stored in the message buffer and the status of the message is stored in LINSR.

If the response is received successfully, LINSR.DRF bit is set. In the case of an error, the LINSR.DRF flag is not set and the corresponding error flag is set in LINESR. (See ***18.3.3Errors*** for more details.)

Data Discard

To discard the data after header transmission, set the LINCR2.DDRQ bit.

Slave Mode

Slave mode is selected when the LINCR1.MME bit is cleared.

Data Transmission

On header reception, the LINSR.HRF bit is set and an Rx interrupt is generated. Software must then:

1.Read the received ID in the BIDR register.

2.Fill the BDRL/BDRM registers.

3.Program the CCS and DIR bits in the BIDR register.

4.Specify the data field length in the BIDR.DFL bits.

The LINSR.HRF bit should be reset only after the LINCR2.DTRQ bit is set. For the DTRQ to be effective, the HRF bit must be set. This is to ensure that DTRQ is not set randomly, but only after a header reception. The DIR and DTRQ bits cannot be set after LINSR.RXBUSY is asserted.

Alternately, one or more identifier filters are configured for transmission by setting the DIR bit, and activated by setting the enable bits in IFER. When at least one identifier filter is active and configured for transmission, and the received ID matches the filter, a Tx interrupt is generated. Software can use the index in the IFMI register to point directly to the corresponding data array in the RAM and copy the data to BDRM/BDRL.

The use of a filter saves software the processing time required to read the ID value in the BIDR, match it, and configure the data field length and checksum type.

If the number of filters provided by the LINFlexD is not sufficient for the application, mask mode can be used for the filters.

The transmitted data is also received by the same node and copied to the buffer.

Data Reception

When the LINFlexD is the subscriber of the data of the received identifier, on header reception, the HRF flag is set and an Rx interrupt is generated. Software must read the received ID from the BIDR register and specify the data field length before the reception of the stop bit of the first data byte. When the checksum is received, an Rx interrupt is generated for software to read the received data from BDRM/BDRL and LINSR.RMB is set. Software must then release the data buffer by clearing LINSR.RMB.

When at least one identifier filter is active and configured for reception, an Rx interrupt is generated only after the checksum reception. No interrupt request is generated on reception of the ID.

If the identifier is filtered by software, software can discard the data by setting LINCR2.DDRQ while HRF is set.

Note: For software filtering, software must set the type of checksum (configure BIDR.CCS) before reception of data. Otherwise, the previous value of the CCS bit is considered while calculating checksum.

18.3.3. Errors

Header Error

A header error is an error during the header reception. The error types are:

- Break delimiter error (SDEF)

- Sync field error (SFEF)
- Identifier parity error(IDPEF)

Break Delimiter Error (SDEF)

The break delimiter must be 1 for at least one bit time: otherwise it is considered short and

the receiver discards synchronization on the current header. Consequently, the frame is discarded. An interrupt is generated if LINIER.HEIE is set.

Sync field error (SFEF)

The SFEF error condition is monitored differently depending on whether autosynchronization is enabled, as determined by LINCR1.LASE.

Case 1: Autosynchronization enabled (LINCR1.LASE=1):

When autosynchronization is enabled, the SFEF flag indicates:

- The deviation error on the sync field is outside the LIN specification, which allows up to 14%of period deviation between the slave and master oscillators
- An overflow has occurred during the sync field measurement, which leads to an overflow of the divider registers.

Note: SFEF does not indicate that an inconsistent sync field value (other than 0x55) was received by the slave.

Deviation error on the sync field

The deviation error is checked by comparing the current baud rate (relative to the slave oscillator) with the received LIN sync field (relative to the master oscillator).

This check is based on a measurement between the first falling edge and the last falling edge of the sync field. Referring to this period deviation as D:

- If SFEF is asserted>14.0625%
- If there is no error<14.84375%
- If 14.0625%<D<14.84375%, then the sync field could be either consistent or inconsistent depending on de-phasing between the signal on rx pin and CK_LINFlexD.

Overflow during sync field measurement

The overflow check is based on the measurement of each bit time between both edges of the sync field. This checks that each of these bit times is large enough (more than 12 samples) compared to the bit time of the current baud rate.

Case 2:Autosynchronization disabled (LINCR1.LASE=0)

When autosynchronization is disabled, the sync character is received as a normal character. If the received character is Ox55, the sync field is OK.

On any occurrence of an inconsistent sync field, the receiver immediately exits from

Sync_Field state, the frame is discarded, and LINESR.SFEF is set.

Identifier parity error (IDPEF)

There are two parity bits in the identifier field. The parity bits are checked against the hardware-calculated parity over the other six bits of identifier, according to the formula:

P0 = ID0 XOR ID1 XOR ID2 XOR ID4

P1 = NOT (ID1 XOR ID3 XOR ID4 XOR ID5)

The identifier parity is checked after the ID is transferred to the BIDR register (after stop of ID has been detected properly) and, upon parity mismatch, the receiver state machine exits from the identifier state immediately if LINCR2.IOPE is set.

Bit errors

A bit error is flagged in transmission mode when the value read back from the bus is different from the value transmitted. Bit error checking on each bit is ensured if transceiver delay is less than one bit time minus 6 CK_LINFlexD cycles, for example:

- One bit time at 20 Kbit/s =50 µs
- 6 CK_LINFlexD cycles at 80 MHz=75ns
- Thus, (1-bit time) -(6 CK_LINFlexD cycles) =49.925µs

Bit error is not checked during break field transmission of the frame is stopped after the corrupted bit if LINCR2.IOBE=1. If LINCR2.IOBE=0, the transmitter continues to transmit despite the bit error. An interrupt is generated if LINIER.BEIE=1.

Note: If the break delimiter is not detected by the master within one bit time after delimiter transmission because of transceiver delay or error on the bus, then a train of bit error interrupts is generated. In this case, the identifier may not be replaced in BIDR. Similarly, if the start of a falling edge of data is not detected by the transmitter node within one bit time after start bit transmission, then a train of bit error interrupts is generated. In this case, data and checksum replacements in BDRM/BFRL and LINCFR respectively are not ensured.

Framing Error

A framing error is flagged when a dominant state is sampled on the stop bit of the current received character (sync field, identifier field, data field, checksum field). The LINFlexD discards the current frame and returns to the Idle state. An interrupt is generated

LINIER.FEIE is set. The byte that caused framing error is shifted to the buffer, but LINSR.DRF is not set.

Checksum Error

A checksum error is flagged when the checksum computed by hardware does not match the received checksum.

The LINFlexD discards the received frame and returns to the Idle state. an interrupt is generated if LINIER.CEIE is set.

Overrun Error

After the message buffer is full (LINSR.RMB is set), the next valid message reception leads to an overrun and the message is lost. The LINFlexD hardware signals the overrun condition by setting the LINESR.BOF bit. Which message is lost depends on the buffer lock function control bit LINCR1.RBLM:

- If RBLM=0, the old message in the buffer is overwritten by the most recent message.
- If RBLM=1, the most recent message is discarded and the oldest message is available to the software. In slave mode, if the buffer is not released (RMB is not reset) before reception of next identifier and RBLM=1, then the ID along with the data is discarded.

Timeout Error

A timeout error occurs in the case of incomplete response (***Figure 18-10 Incomplete response (for example, missing checksum)***) or no response (***Figure 18-11 No response***) within the timeout response period.

Figure 18-10 Incomplete response (for example, missing checksum)

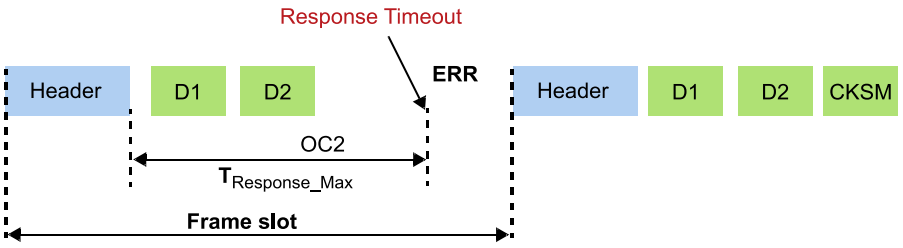
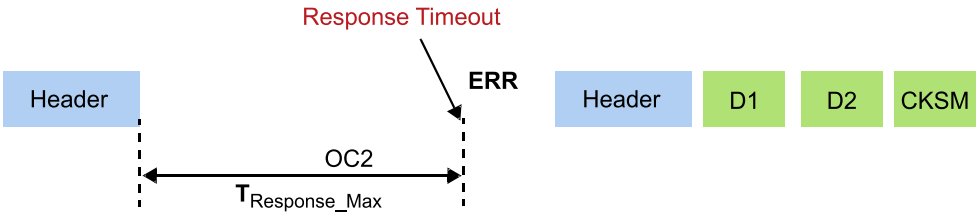


Figure 18-11 No response



Response timeout mechanism

Master mode:

- OC2 is loaded with nominal_time_out + LINTCSR.CNT at the end of the stop bit of the identifier field (where nominal_time_out = 1.4 × ((DFL + 2) ×10 bit time). This loading takes place at the end of identifier field (not at the beginning of data field). Moreover, the correct value, which depends on DFL, is loaded immediately. No further OC2 update is

required.

- In case of no response at all within this time (in other words, start of data is not received), timeout takes place when the counter reaches the OC2 value.
- In case of an incomplete response, timeout also occurs when the counter reaches the OC2 value.

Slave mode:

- Case 1: The received identifier is managed by a filter. The implementation can be similar to the master mode, because the DFL value is loaded by hardware. Thus, OC2 can be loaded with nominal_time_out + LINTCSR.CNT at the end of the stop bit of the identifier field (where nominal_time_out = 1.4 × ((DFL + 2) × 10 bit time)).
- Case 2: The received identifier is not managed by a filter. As the DFL value needs to be updated by software after the identifier field has been received, the implementation is:
  - At the end of the ID, OC2 is loaded with 36(maximum possible response space)+LINTCSR.CNT.
  - At the end of the first data byte, it is reloaded again according to DFL.
  - Before reloading, the LINFlexD checks the count_val. If count value is higher than the value to be reloaded, timeout takes place immediately and no reloading occurs.

Note: The value of nominal_time_out (1.4 × ((DFL + 2) × 10 bit time)) shown is as per the reset value of LINTOCR.RTO and LINCR1.CFD.It changes with change in value of LINTOCR.RTO and LINCR1.CFD.Here,1.4 corresponds to

(LINTOCR.RTO)/10 and 2 corresponds to (2-LINCR1.CFD)

Header timeout mechanism

- Master mode-As the header is generated by the LINFlexD, there are only two cases:

  No error occurs on the bus and timing is correct(nominal header length).

  An error occurs on the bus and the LINFlexD flags it in LINESR (typically a bit error).

Therefore, there is no meaning of header timeout in master mode, so it is disabled.

- Slave mode:
  - header_nominal =13+2+10+10=35T_{bit}
  - header_max=1.4 × header_nominal = 49T_{bit}
  - Taking into account a possible 14%clock deviation, header_max seen by the LINFlexD is 49×1.14=55.86T_{bit}
  - If the counter starts after 11 T bit(break duration), the LINTOCR.HTO value is 55.86 - 11 = 44.86T_{bit}

LINTOCR.HTO can only be programmed in slave mode and has a reset value of 44.

Counter restarts after break duration and LINOCR.OC1 is loaded with the value of LINTOCR.HTO

Note: The timeout counter can be used to detect other timeouts. In this case, the LINTCSR.MODE bit must be reset and the output compare value can be updated in LINTOCR by software.

Stuck at zero timeout error

If a dominant pulse lasts for a time of at least 100 bits, LINESR.SZF is set. If the same dominant pulse continues, the subsequent SZF settings occur 87 bit times apart (instead of 100 bit times).

Noise

During reception, each bit is sampled 16 times and the value of the bit is obtained by taking the majority value of the 8th,9th, and 10th samples. If any one of these three samples has a value different from the other two, a noise error is flagged.

When UARTCR.OSR = 8, majority of 2nd,3rd, and 4th samples are taken into account to determine noise. Noise checking is disabled for OSR values less than eight.

18.3.4. Identifier filtering

In the LIN protocol, the identifier of a message is not associated with the address of a node; it is related to the content of the message. A transmitter broadcasts its message to all the receivers. Based on the header received, the receiver decides whether to receive or transmit a response (depending on the identifier value). If the message does not target the node, the message should be discarded.

To fulfil this requirement, the LINFlexD provides configurable filters in order to eliminate software intervention. Hardware filtering saves CPU resources that would otherwise be required to perform filtering by software.

There is a maximum of 16 filters (depending on configuration parameter no_of_filters) in the LINFlexD. The filters can be programmed only during Initialization mode. To activate a filter, the corresponding FACT bit must be set in IFER. There are two modes possible for each identifier, depending on the corresponding IFMR.IFM bit.

Identifier list mode

If IFMR.IFM[n]=0, filters 2n and 2n+1 are in identifier list mode. The maximum number of filters that can be configured for transmission/reception is no_of_filters, depending on the value of IFER.FACT. The identifier received must match bit-by-bit to the ID field of IFCR2n or IFCR2n+1(if the corresponding filter is enabled in IFER.FACT).

Identifier mask mode

If the number of filters required is more than no_of_filters, then filters should be configured in mask mode. If IFMR.IFM[n]=1, filters 2n and 2n+1 are in mask mode. Filter 2n is the filter and filter 2n+1 acts as a mask for it. The IFER.FACT bit for filter 2n+1 has no effect in this case. If bit x of the mask is set, then bit x of the received identifier must match bit x of filter.

In either filter mode, if there is a match of identifier with filter number value m+1 is loaded into the IFMI register by hardware. IFMI=0 indicates a no-match condition.

On a matching ID, the LINFlexD hardware copies the DFL [2:0], CCS, and DIR values from the matching filter to the BIDR register; from then on, the BIDR register is read-only until the end of the frame. BIDR.DIR determines the transfer direction:

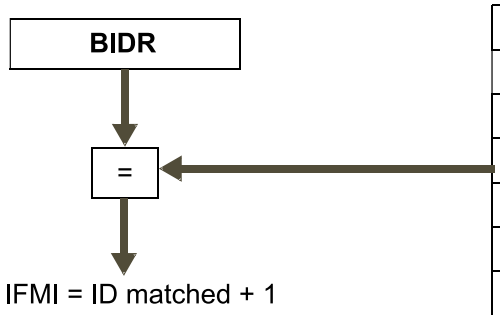
- If BIDR.DIR=1, the direction is transmit. A Tx interrupt is generated if LINIER.HRIE is set. Software can read the filter match index from the IFMI register, then transfer the relevant data from the RAM area to the BDRM/BDRL registers. After transferring the transmit data to the BDRM/BDRL registers, set LINCR2.DTRQ to start the transmission.
- If BIDR.DIR=0, the direction is receive. If LINIER.DRIE is set, an Rx interrupt is generated when the checksum has been received and there is no checksum error.

In the case of no filter match condition (IFMI=0), if LINCR1.BF=1, an Rx interrupt is generated. It is the responsibility of software to configure BIDR and start

Transmission (by loading the BDRM/BDRL registers and setting LINCR2.DTRQ) or discard reception (by setting LINCR2.DDRQ). If LINCR1.BF=0, the receiver discards the received identifier and turns to idle state in search of a new break.

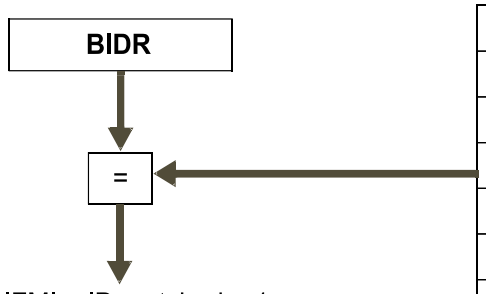
Note: If one identifier matches with two filters (one in list mode and the other in mask mode), then list mode prevails over mask mode. In mask mode, if two filters match with the identifier, the filter with the lower number prevails.

Figure 18-12 IDENTIFIER LIST MODE



IFCR0	ID5	ID4	ID3	ID2	ID1	ID0
IFCR1	ID5	ID4	ID3	ID2	ID1	ID0
IFCR2	ID5	ID4	ID3	ID2	ID1	ID0
IFCR3	ID5	ID4	ID3	ID2	ID1	ID0
IFCR4	ID5	ID4	ID3	ID2	ID1	ID0
IFCR5	ID5	ID4	ID3	ID2	ID1	ID0
IFCR6	ID5	ID4	ID3	ID2	ID1	ID0
IFCR7	ID5	ID4	ID3	ID2	ID1	ID0
...	ID5	ID4	ID3	ID2	ID1	ID0
IFCR*n*	ID5	ID4	ID3	ID2	ID1	ID0

Figure 18-13 IDENTIFIER MASK MODE



IFCR0	ID5	ID4	ID3	ID2	ID1	ID0
IFCR1	0	0	0	1	1	1
IFCR2	ID5	ID4	ID3	ID2	ID1	ID0
IFCR3	1	1	1	1	0	0
IFCR4	ID5	ID4	ID3	ID2	ID1	ID0
IFCR5	1	1	1	1	0	1
IFCR6	ID5	ID4	ID3	ID2	ID1	ID0
IFCR7	1	1	1	1	1	0
…	…	…	…	…	…	…
IFCR*n*	…	…	…	…	…	…

1 = Must match bits
0 = Don't care bits

$IFCR_{2n+1}$ = Mask for $IFCR_{2n}$

18.3.5. Start detection and break delimiter detection in receiver

There is a 10-bit-shift sample register in the receiver that shifts signal data to the next least significant bit on each incoming sample.

A start is detected as soon as it is qualified with 1110 in the sample register and verified with at least two out of three predefined verification samples being zero. Similarly, for delimiter detection, qualification samples are 0001 and at least two of the three verification samples are one. The Noise Flag is set if the start is verified with only two valid samples.

Table 18-1. Start detection and delimiter detection in receiver

9	8	7	6	5	4	3	2	1	0	Sample register
—	—	—	—	-	—	—	—	—	RT1	counter=0
—		-	—		—	—	—	RT1	RT2	counter =1
—	—	-	—	—	—	—	RT1	RT2	RT3	counter =2
			—	—	—	RT1	RT2	RT3	RT4	counter=3
—	—	—	—	—	RT1	RT2	RT3	RT4	RT5	counter =4
—	—	—	—	RT1	RT2	RT3	RT4	RT5	RT6	counter =5
	—	—	RT1	RT2	RT3	RT4	RT5	RT6	RT7	counter=6
—	—	RT1	RT2	RT3	RT4	RT5	RT6	RT7	RT8	counter =7
	RT1	RT2	RT3	RT4	RT5	RT6	RT7	RT8	RT9	counter =8
RT1	RT2	RT3	RT4	RT5	RT6	RT7	RT8	RT9	RT10	counter=9
RT2	RT3	RT4	RT5	RT6	RT7	RT8	RT9	RT10	RT11	counter =10
RT3	RT4	RT5	RT6	RT7	RT8	RT9	RT10	RT11	RT12	counter =11
RT4	RT5	RT6	RT7	RT8	RT9	RT10	RT11	RT12	RT13	counter =12

RT5	RT6	RT7	RT8	RT9	RT10	RT11	RT12	RT13	RT14	counter =13
RT6	RT7	RT8	RT9	RT10	RT11	RT12	RT13	RT14	RT15	counter =14
RT7	RT8	RT9	RT10	RT11	RT12	RT13	RT14	RT15	RT16	counter =15
1	1	1	0	X	0	X	0	X	0	start detect
1	1	1	0	X	0	X	0	X	1	
1	1	1	0	X	0	X	1	X	0	
1	1	1	0	X	1	X	0	X	0	
0	0	0	1	X	1	X	1	X	1	delimiter detect
0	0	0	1	X	1	X	1	X	0	
0	0	0	1	X	1	X	0	X	1	
0	0	0	1	X	0	X	1	X	1	

Start detection mechanism

The process for detecting a start bit is:

1. See ***Table 18-1. Start detection and delimiter detection in receiver*** for the status of the sample register (shift register) when count=6. If RT1(the first incoming sample)=0 and the previous three samples already received are allones, then it might be a start bit.
2. To make sure it is a start bit, the incoming data samples in sample register(4), (2), and (0), which correspond to RT3, RT5, and RT7 are verified using the steps mentioned below.
3. If the majority (2 or more) out of these 3 samples are equal to '0', a start bit is detected. These three samples RT3, RT5, and RT7 are called the "verification samples" which are checked at count =6.
4. Sample register bits 9, 8, 7, and 6 are called "qualification samples "which are checked at count=6.
5. At count =9, if majority value of RT8, RT9, and RT10 is not equal to '0', the Noise Flag is set.

Break delimiter detection mechanism

The process for detecting a break delimiter is:

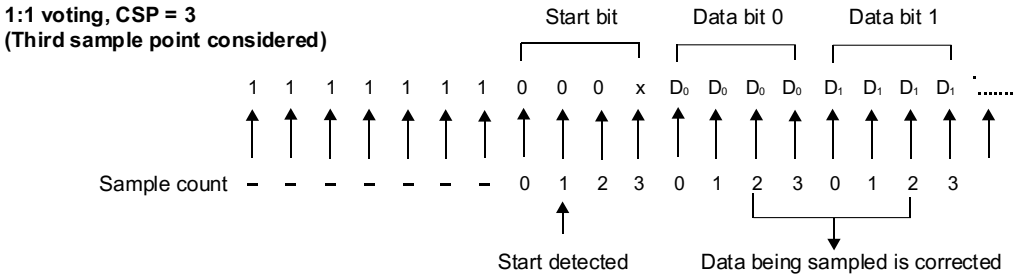
1. See ***Table 18-1. Start detection and delimiter detection in receiver*** for the status of the sample register (shift register) when count=6. If RT1 (the first incoming sample) =1 and the previous three samples already received are all zeros, then it might be a delimiter bit.
2. To make sure it is a break delimiter, the incoming data samples in the sample register(4), (2), and (0), which correspond to RT3, RT5, and RT7 are verified using the steps mentioned below.
3. If the majority (2 or more) out of these 3 samples are equal to '1', then a break delimiter

bit is detected. These three samples RT3, RT5, and RT7 are called the "verification samples" which are checked at count =6.

4. The sample register bits 9, 8, 7, and 6 are called "qualification samples" which are checked at count=6.

Therefore, a start is detected as soon as it is qualified with 1110 in the sample register and verified with at least two out of three predefined verification samples being zero. Similarly, for delimiter detection, qualification samples are 0001 and at least two out of the three verification samples are one. The Noise Flag is set if the start is verified with only two valid samples.

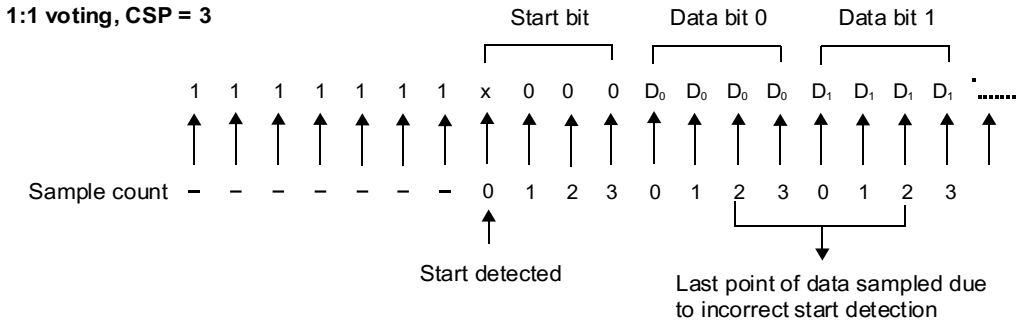
Figure 18-14. Start detection and sampling for oversampling rate = 4 (case 1)



Note:

1. D0, D1: Databits
2. X: Due to sampling during transition of the bit, value might be sampled as a '0' or '1' depending on the rise/fall time. Therefore 'X' refers to a indeterministic value.

Figure 18-15. Start detection and sampling for oversampling rate = 4 (case 2A)



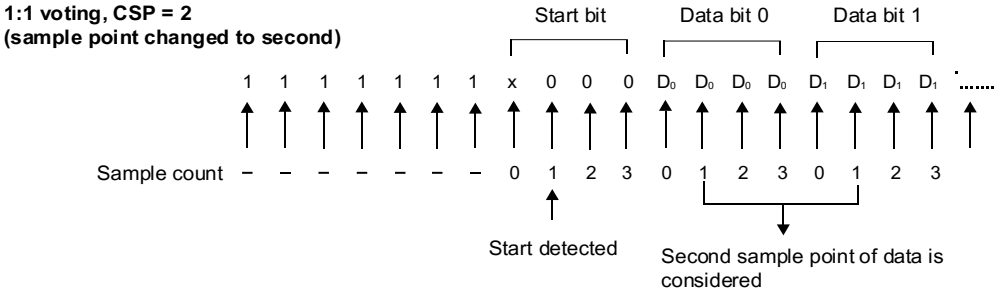
Note:

1. D0, D1: Databits
2. X: Due to sampling during transition of the bit, value might be sampled as a '0' or '1' depending on the rise/fall time. Therefore 'X' refers to a indeterministic value.

The choice of the correct sample points depends on the external Rx signal quality. During the application development process, the error information indicated by the parity error can help to find the best setting for an application-specific hardware signal.

To sample at the correct point therefore, CSP has to be changed to two; resulting in the following situation:

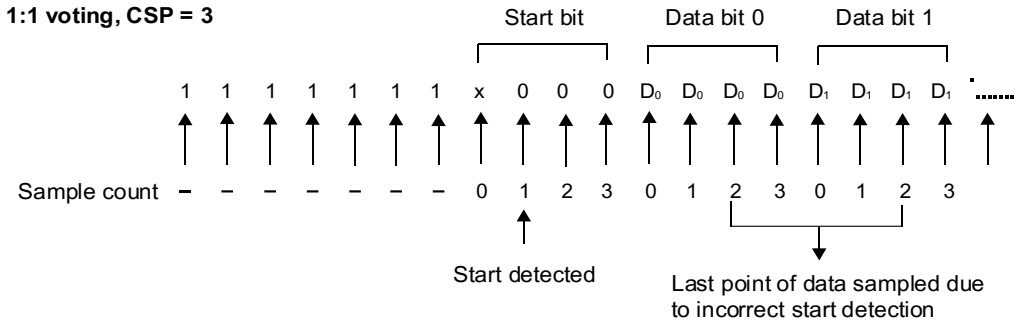
Figure 18-16. Start detection and sampling for oversampling rate = 4 (case 2B)



Note:

1. D0, D1: Databits
2. X: Due to sampling during transition of the bit, value might be sampled as a '0' or '1' depending on the rise/fall time. Therefore 'X' refers to a non deterministic value.

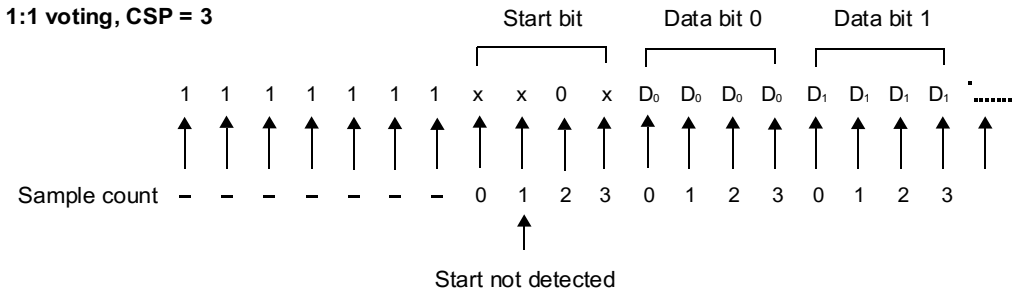
Figure 18-17. Start detection and sampling for oversampling rate = 4 (case 3)



Note:

1. D0, D1: Databits
1. X: Due to sampling during transition of the bit, value might be sampled as a '0' or '1' depending on the rise/fall time. Therefore 'X' refers to a non deterministic value.

Figure 18-18. Start detection and sampling for oversampling rate = 4 (case 4)

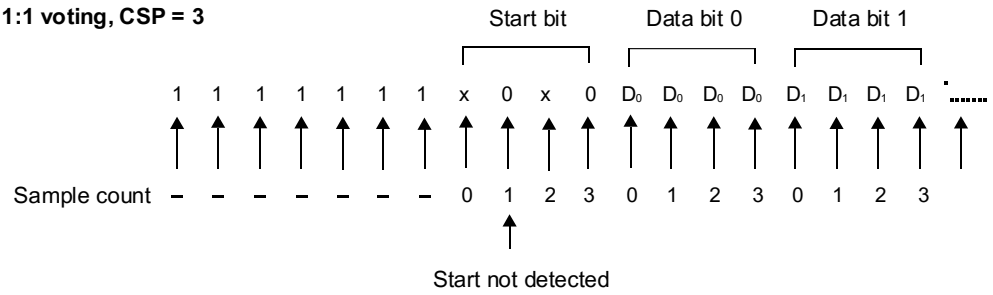


Note:

1) D0, D1: Databits

2) X: Due to sampling during transition of the bit, value might be sampled as a '0' or '1' depending on the rise/fall time. Therefore 'X' refers to a non deterministic value.

Figure 18-19. Start detection and sampling for oversampling rate = 4 (case 5)



Note:

1) D0, D1: Databits

2) X: Due to sampling during transition of the bit, value might be sampled as a '0' or '1' depending on the rise/fall time. Therefore 'X' refers to a non deterministic value.

18.3.6. Baud rate generation

The LIN baud rate is programmed in two registers: the LIN Integer Baud Rate Register (LINIBRR) and the LIN Fractional Baud Rate Register (LINFBRR). These registers can be programmed only during Initialization mode.

Baud rate is calculated with the following formula for both receiver and transmitter:

- When UARTCR.ROSE=1, Tx=Rx=CK_LINFlexD / (OSR×LDIV)
- When UARTCR.ROSE=0, Tx=Rx=CK_LINFlexD / (16×LDIV)

Where CK_LINFlexD is the frequency of the baud clock, CK_LINFlexD.LDIV is an unsigned fixed point number. The mantissa is coded into 20 bits of LINIBRR and the fraction is coded on 4 bits of LINFBRR.

When reduced oversampling is enabled,LINFBRR should not be used and programmed to zero and LDIV contains only the integer part of LINIBRR.

For example: When ROSE=0 (for LIN and UART mode): LDIV = 468.75d, CK_LINFlexD = 36MHz, LINIBRR = 468d, LINFBRR=12, Baud rate =36MHz / (16 × 468.75) = 4.8Kbit/s.

For example: When ROSE=1 (only for UART mode): LDIV = 5d, CK_LINFlexD = 80MHz, LINIBRR = 5d, OSR = 4, Baudrate = LIN_CLK / (OSR × LDIV) = 80MHz / (4 × 5) = 4Mbit/s .

18.3.7. Automatic resynchronization

To automatically adjust the baud rate based on measurement of the LIN sync field, write the nominal prescalar value (nominal baud rate) in LINIBRR and LINFBRR, then set LINCR1.LASE to enable automatic synchronization.

When auto-synchronization is enabled, after each break delimiter, the time duration between five falling edges on Rx pin (RDI) is sampled on CK_LINFlexD.

When the slave uses automatic baud rate detection, the functionality can be ensured if the master's baud rate deviation is within the ±14% range. If the master's baud rate deviation exceeds the ±14% range, it is not guaranteed that all values will respond correctly.

18.3.8. Wakeup management

Any node in a sleeping LIN cluster may request a wakeup. The wakeup request is issued by forcing the bus to the dominant state for 250μs to 5ms. Every slave node should detect the wakeup request(a dominant pulse longer than 150 μs) and be ready to listen to bus commands within 100 ms, measured from the ending edge of the dominant pulse. The master also wakes on detecting a wakeup request and when the slaves are ready, starts sending frame headers to find out the cause of the wakeup. If the master does not issue a frame header within 150 ms from the wakeup request, then the node issuing a request may try issuing a new wakeup request.

In the LINFlexD, a wakeup request can be generated by writing the wakeup character in BDRL.DATA0 and setting LINCR2.WURQ. On setting the WURQ bit,the character in BDRL.DATA0 is transmitted.

In the LINFlexD,wakeup can be detected in two ways:

- AUTOWU=1: On detecting a falling edge in sleep mode, the LINCR1.SLEEP bit is cleared by hardware, LINSR.WUF is set, and an interrupt is generated. if LINIER.WUIE=1.The LINFlex is now in normal mode and ready to receive frames.
- AUTOWU=0: On detecting a falling edge, LINSR.WUF is set and an interrupt is generated (if LINIER.WUIE=1). It is then up to the software to clear LINCR1.SLEEP.

Figure 18-20. Wakeup sequence



18.3.9. Timer

The LINFlexD includes an 8-bit counter, which has different behavior in different modes:

When the counter is in output compare mode (LINTCSR.MODE=1), the counter runs even in Sleep and Initialization modes. When the counter value matches LINOCR.OC1 or LINOCR.OC2, and LINTCSR.TOCE is set, the output compare flag (LINESR.OCF) is set. An error interrupt is generated if LINIER.OCIE is set.

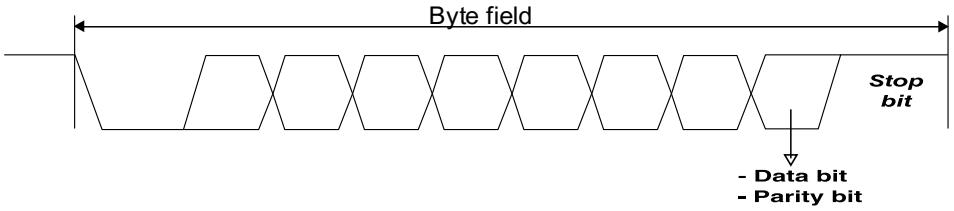
When the counter is in LIN mode (LINTCSR.MODE=0), software has no control over the TOCE bit and output compare registers are utilized for generation of LIN timeout interrupts (header, frame, and response timeouts). If the LINFlexD goes to Sleep mode or Initialization mode, the counter remains in the reset state. When UART mode is enabled, LIN mode for the counter has no meaning; the counter remains in the reset state.

18.3.10. UART mode

8-bit data frames

The eighth bit can be either a data bit or a parity bit. The parity mode (even/odd/0/1) can be selected using UARTCR.PC[1:0]. An even parity is set if the modulo-2 sum of the seven data bits is one. An odd parity is cleared in this case.

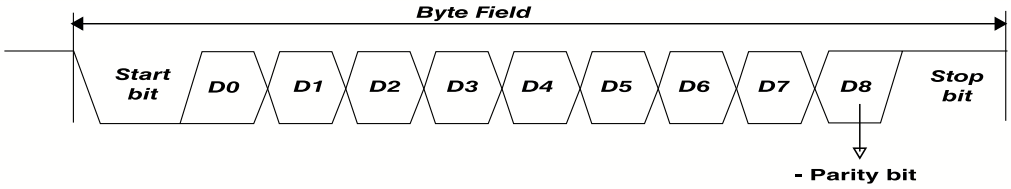
Figure 18-21 UART mode 8-bit data frame



9-bit frames

The ninth bit should be a parity bit. The parity mode (even/odd/0/1) can be selected using UARTCR.PC[1:0]. An even parity is set if the modulo-2 sum of the eight data bits is one. An odd parity is cleared in this case. Parity 0 forces the bit to a zero logical value. Parity 1 forces the bit to a high logical value.

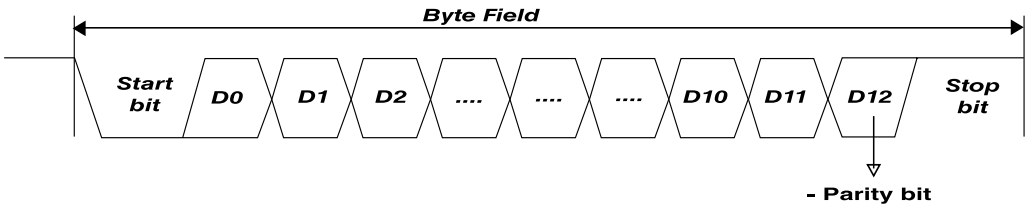
Figure 18-22 UART mode 9-bit data frame



13-bit frames

When UARTCR.WLS = 1, special word length is selected in UART mode. Special word length enables 12-data-bit + parity bit reception only in FIFO mode.

Figure 18-23 UART mode 13-bit data frame



Buffer in UART Mode

The 8-byte buffer is divided into two parts: one for receiver and one for transmitter, as shown in *Figure 18-24 Structure of 8-byte buffer*.

Figure 18-24 Structure of 8-byte buffer

Tx0	BDR0 (BDRL.DATA0)
Tx1	BDR1 (BDRL.DATA1)
Tx2	BDR2 (BDRL.DATA2)
Tx3	BDR3 (BDRL.DATA3)
Rx0	BDR4 (BDRM.DATA4)
Rx1	BDR5 (BDRM.DATA5)
Rx2	BDR6 (BDRM.DATA6)
Rx3	BDR7 (BDRM.DATA7)

Note: In UART FIFO mode, user can configure LIN_STCTL.SIZE to byte (b01) or half-word (b10) to specify the data size and avoid transmission errors.

UART transmitter

To start transmission in the UART mode, set UARTCR.UART and UARTCR.TXEN.

Transmission starts when the BDRO (least significant data byte) is programmed and continues until the number of bytes / halfwords transmitted is equal to the value in the UARTCR.TDFLbits.

The transmit buffer is four bytes (when UARTCR.WL1=0) or two halfwords (when UARTCR.WL1=1). Therefore, a maximum of four bytes (two halfwords) can be triggered for transmission. Once the programmed number of bytes or halfwords is transmitted, the UARTSR.DTF flag is set. If the UARTCR.TXEN bit is reset in the middle of a transmission, the current transmission is completed and no further transmissions are invoked.

The buffer can be configured in FIFO mode (mandatory when the DMATx is enabled) by setting control bit UARTCR.TFBM.

Note: If UARTSR.TFF=1 and a write is performed to the FIFO, the data transmitted may be

erroneous.

Table 18-2. BDRL access in UART mode

IPS operation	Register	Mode (UARTCR.TFBM)	Word length (UARTCR.WL1)	Peripheral Bus Operation Result
Write Byte0	BDRL	FIFO	Byte	OK
Write Byte1-2-3	BDRL	FIFO	Byte	Transfer error
Write Half-word0-1	BDRL	FIFO	Byte	Transfer error
Write Word	BDRL	FIFO	Byte	Transfer error
Write Byte0-1-2-3	BDRL	FIFO	Half-word	Transfer error
Write Half-word0	BDRL	FIFO	Half-word	OK
Write Half-word1	BDRL	FIFO	Half-word	Transfer error
Write Word	BDRL	FIFO	Half-word	Transfer error
Read Byte0-1-2-3	BDRL	FIFO	Byte/Half-word	Transfer error
Read Half-word0-1	BDRL	FIFO	Byte/Half-word	Transfer error
Read Word	BDRL	FIFO	Byte/Half-word	Transfer error
Write Byte0-1-2-3	BDRL	BUFFER	Byte/Half-word	OK
Write Half-word0-1	BDRL	BUFFER	Byte/Half-word	OK
Write Word	BDRL	BUFFER	Byte/Half-word	OK
Read Byte0-1-2-3	BDRL	BUFFER	Byte/Half-word	OK
Read Half-word0-1	BDRL	BUFFER	Byte/Half-word	OK
Read Word	BDRL	BUFFER	Byte/Half-word	OK

In UART FIFO mode (UARTCR.TFBM=1), any read operation causes a peripheral bus transfer error.

UART receiver

The reception of a data byte is started as soon as software exits Initialization mode (by clearing LINCR1.INIT), sets UARTCR.RXEN, and detects a start bit. There is a dedicated 4-byte (if UARTCR.WL1=0) or 2-halfword (if UARTCR.WL1=1) data buffer for received data. Once the programmed number (UARTCR.RDFL) of bytes have been received, the UARTSR.DRF flag is set, and the current reception is completed. UARTCR. RXEN needs to be set only to start the reception. The reception is automatically completed as soon as the programmed number of bytes are received.

The buffer can be configured in FIFO mode (mandatory when the DMARx is enabled) by setting the control bit UARTCR.RFBM.

Table 18-3. BDRM access in UART mode

IPS operation	Register	Mode (UARTCR.RFBM)	Word length (UARTCR.WL1)	Peripheral Bus Operation Result
Read Byte4	BDRM	FIFO	Byte	OK
Read Byte5-6-7	BDRM	FIFO	Byte	Transfer error

IPS operation	Register	Mode (UARTCR.RFBM)	Word length (UARTCR.WL1)	Peripheral Bus Operation Result
Read Half-word2-3	BDRM	FIFO	Byte	Transfer error
Read Word	BDRM	FIFO	Byte	Transfer error
Read Byte4-5-6-7	BDRM	FIFO	Half-word	Transfer error
Read Half-word2	BDRM	FIFO	Half-word	OK
Read Half-word3	BDRM	FIFO	Half-word	Transfer error
Read Word	BDRM	FIFO	Half-word	Transfer error
Write Byte4-5-6-7	BDRM	FIFO	Byte/Half-word	Transfer error
Write Half-word2-3	BDRM	FIFO	Byte/Half-word	Transfer error
Write Word	BDRM	FIFO	Byte/Half-word	Transfer error
Read Byte4-5-6-7	BDRM	BUFFER	Byte/Half-word	OK
Read Half-word2-3	BDRM	BUFFER	Byte/Half-word	OK
Read Word	BDRM	BUFFER	Byte/Half-word	OK
Write Byte4-5-6-7	BDRM	BUFFER	Byte/Half-word	Transfer error
Write Half-word2-3	BDRM	BUFFER	Byte/Half-word	Transfer error
Write Word	BDRM	BUFFER	Byte/Half-word	Transfer error

Note: Refer to the layout of the registers BDRL and BDRM to identify the mapping between byte x and data bits of registers BDRL and BDRM.

Additional notes:

- If it is not known in advance how many bytes are to be received, RDFL should not be programmed in advance. The reset value of RDFL is zero, which ensures that reception occurs byte-by-byte. The state machine goes to the Idle state after each received byte.
- If RDFL is programmed for a certain value but that number of bytes are not received, reception hangs.UARTSR.TO is set when a reception timeout occurs.
- If a STOP request arrives in the middle of a reception; it is only acknowledged after all the programmed number of data bytes have received. If the programmed number of data bytes are not received and a reception timeout occurs, a STOP request is acknowledged when the receiver FSM is reset to the Idle state.
- If a parity error occurs during reception of any byte, the corresponding UARTSR.PE[n] bit is set. No interrupt is generated in this case. If a framing error occurs in an byte (UARTSR.FEF is set), an interrupt is generated if LINIER.FEIE=1. As there is only one register bit for framing error, this interrupt helps to identify which byte has a framing error.
- If the last received frame has not been read from the buffer (in other words, RMB is not reset by software), then upon reception of the next byte, an overrun error occurs (UARTSR.BOF is set). An interrupt is generated if LINIER.BOIE=1.The new message is either discarded or overwrites the previous message in the buffer.

- When a STOP is requested when the LINFlexD is in UART FIFO mode, software must set LINCR.INIT bit to set the receiver FSM to Idle, at which point the LINFlexD. asserts ACK request.

18.3.11. Interrupts

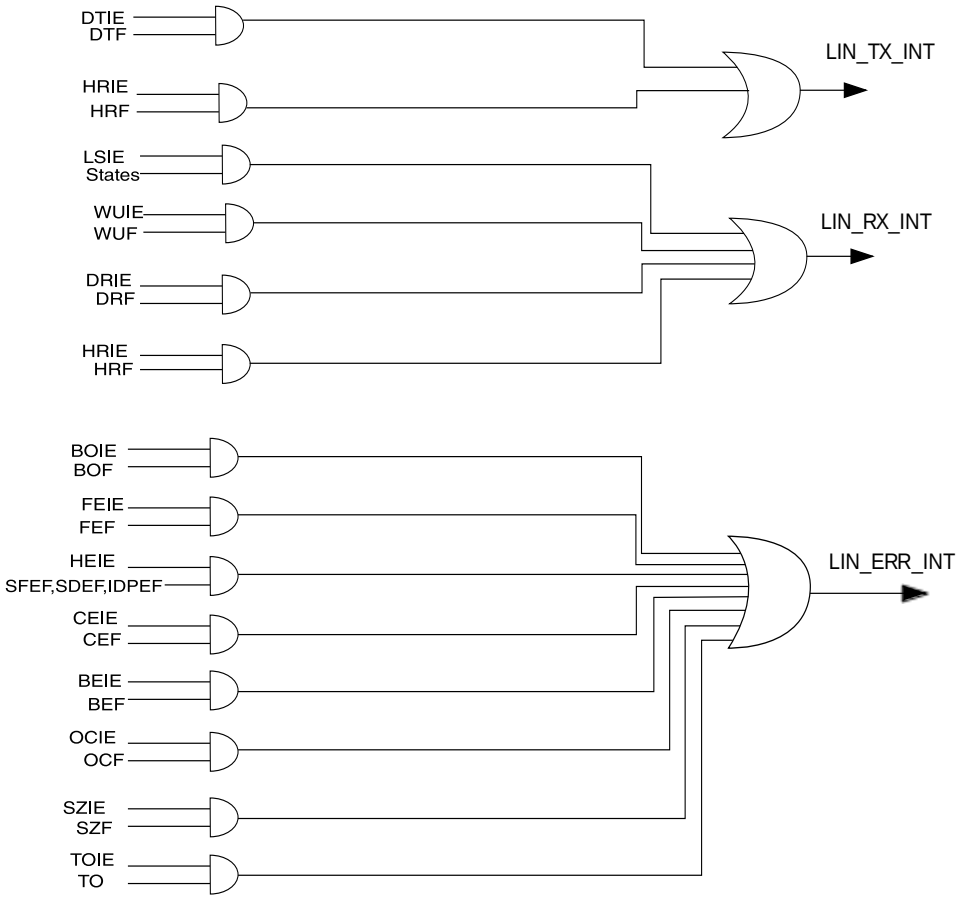
Table 18-4. Interrupts summarizes the LINFlexD interrupt events and corresponding interrupt enables.

Figure 18-25 Interrupt diagram shows the mapping interrupt flags to the LINFlexD interrupt output signals

Table 18-4. Interrupts

Interrupt event	Event flag in LINSR/LINESR/UARTSR/ LIN_STCTL	Interrupt enable bit in LINIER/ LIN_STCTL	Interrupt vector
Stuck at zero	SZF	SZIE	Error
Output compare	OCF	OCIE	Error
Bit error	BEF	BEIE	Error
Checksum error	CEF	CEIE	Error
Header error	SFEF or SDEF or IDPEF	HEIE	Error
Frame error	FEF	FEIE	Error
Buffer overrun error	BOF	BOIE	Error
UART timeout error	TO	TOIE	Error
LIN state	Sync del, Sync field, Identifier field, or Checksum field	LSIE	Rx
Wakeup	WUF	WUIE	Rx
Data reception complete	DRF	DRIE	Rx
Data transmitted	DTF	DTIE	Tx
Header received	HRF	HRIE	Rx
Header received	HRF	HRIE	Tx (if there is a filter match for Tx identifier)

Figure 18-25 Interrupt diagram



Note: Rx interrupt due to LIN state event is cleared by writing LINSR.LINS field with 0xF when LSIE is enabled.

Note: In Loop Back mode, Rx interrupt is set earlier than Tx interrupt.

Note: After enabling DMA, to use interrupts normally, it is necessary to ensure that LIN 's PCLK > CK_LINFlexD/3.

18.3.12. DMA operations

LIN Master Node: Transmit DMA Channel

As a LIN master node, the LINFlexD transmits the LIN frame header and, if the LINFlexD is the publisher of the data, it also transmits the response (data) portion of the LIN frame.

The LINFlexD uses a single transmit DMA channel (channel 0) for both the frame header and the transmit data. To enable DMA in LIN master mode, DMATXE [0] must be set.

Table 18-5. LIN master node transmit dma transfers details the data transferred from RAM to LINFlexD registers in response to accept DMA TX request:

- Master to slave: The master transmits the frame header and data.

- Slave to master: The master transmits the frame header and uses receive DMA channel 0 to receive the data.
- Slave to slave: The master transmits only the frame header.

In the cycle following the data phase of the last write for the transfer, the DMA controller must acknowledge the DMA request.

Table 18-5. LIN master node transmit dma transfers

Frame type	DMA transfer		Data values
	From RAM	To LINFlexD register	
Master to slave	LINCR2 data	LINCR2	DDRQ=1 DTRQ=0 HTRQ=1
	BIDR data	BIDR	DFL=payload size ID=identifier CCS=checksum type DIR=1
	Payload data bytes 0-3	BDRL	1-4 data bytes depending on DFL
	Payload data bytes4-7	BDRM	0-4 data bytes depending on DFL
Slave to master	LINCR2 data	LINCR2	DDRQ=0 DTRQ=0 HTRQ=0
	BIDR data	BIDR	DFL=payload size ID=identifier CCS=checksum type DIR=0
Slave to slave	LINCR2 data	LINCR2	DDRQ=0 DTRQ=0 HTRQ=0
	BIDR data	BIDR	DFL=payload size ID=identifier CCS=checksum type DIR=0

LIN master node: receive dma channel

As a LIN master node, the LINFlexD uses a single receive DMA channel (channel 0) to receive data for a slave-to-master frame.

To enable the DMA receive channel in LIN master mode, DMARXE[0] must be set.

Table 18-6. LIN master node receive dma transfer lists the data transferred to RAM from LINFlexD registers in response to accept DMA RX request:.

In the cycle after the data phase of the last read for the transfer, the DMA controller must acknowledge the DMA request.

Table 18-6. LIN master node receive dma transfer

Frame Type	DMA transfer		Data Values
	From LINFlexD Register	To RAM	
Slave to master	BIDR (optional)	BIDR data	DFL=payload size ID=identifier
	BDRL	Payload data bytes 0-3	1-4 data bytes depending on DFL
	BDRM	Payload data bytes 4-7	0-4 data bytes depending on DFL

LIN slave node: transmit dma channel

As a LIN slave node, the LINFlexD transmits the response (data) portion of the LIN frame.

The LINFlexD uses one transmit DMA channel for each ID filter that is programmed in transmit mode. To enable transmit DMA channel n in LIN slave mode, DMATXE[n] must be set.

Table 18-7. LIN slave node transmit dma transfers provides details about the data transferred from RAM to LINFlexD registers in response to accept DMA TX request:.

In the cycle following the data phase of the last write for the transfer, the DMA controller must acknowledge the DMA request.

Table 18-7. LIN slave node transmit dma transfers

Frame Type	DMA transfer		Data Values
	From RAM	To LINFlexD Register	
Slave to master	Payload data bytes 0-3	BDRL	1-4 data bytes depending on IFCRn.DFL
slave to slave	Payload data bytes 4-7	BDRM	0-4 data bytes depending on IFCRn.DFL

LIN slave node: receive dma channel

As a LIN slave node, the LINFlexD uses one receive DMA channel for each ID filter that is programmed in receive mode.

To enable receive DMA channel n in LIN slave mode, DMARXE[n] must be set.

Table 18-8. LIN slave node receive dma transfer outlines the data transferred to RAM from LINFlexD registers in response to accept DMA RX request.

In the cycle following the data phase of the last read for the transfer, the DMA controller must acknowledge the DMA request.

Table 18-8. LIN slave node receive dma transfer

Frame Type	DMA transfer		Data Values
	From LINFlexD Register	To RAM	
Master to slave or slave to slave	BIDR	BIDR data	DFL=payload size ID=identifier
	BDRL	Payload data bytes 0-3	1-4 data bytes depending on DFL
	BDRM	Payload data bytes 4-7	0-4 data bytes depending on DFL

UART mode: transmit dma channel

To use DMA when the LINFlexD is operating in UART mode, the LINFlexD UART transmit/receive buffers must be configured as FIFOs.

The LINFlexD uses a single transmit DMA channel (channel 0) to move data from RAM to the UART transmit buffer. To enable DMA for UART transmit mode, DMATXE[0] must be set.

When DMA TX request is accepted, the DMA controller transfers up to 4 bytes or 2 halfwords from RAM to the UART transmit FIFO through successive writes to the BDRL register.

In the cycle following the data phase of the last write for the transfer, the DMA controller must acknowledge the DMA request.

UART mode: receive dma channel

To use DMA when the LINFlexD is operating in UART mode, the LINFlexD UART transmit/receive buffers must be configured as FIFOs.

The LINFlexD uses a single receive DMA channel (channel 0) to move data from the UART receive buffer to RAM. To enable DMA for UART receive mode, both DMARXE[0] must be set.

When DMA RX request is accepted is asserted, the DMA controller performs successive reads from the BDRM register to transfer up to 4 bytes or 2 halfwords from the UART receive FIFO to RAM.

In the cycle following the data phase of the last read for the transfer, the DMA controller must acknowledge the DMA request.

DMA usage notes and limitations

- In LIN slave mode, the DMA capability can be used only if the ID filtering mode is activated. The number of ID filters enabled must be equal to the number of DMA channels enabled. The correspondence between channel number and ID filter is based on Identifier Filter Match Index (IFMI).
- To use the DMA capability, both transmit and receive DMA channels must be enabled.
- In UART mode the DMA capability can be used only if the UART Tx/Rx buffers are

configured as FIFOs.

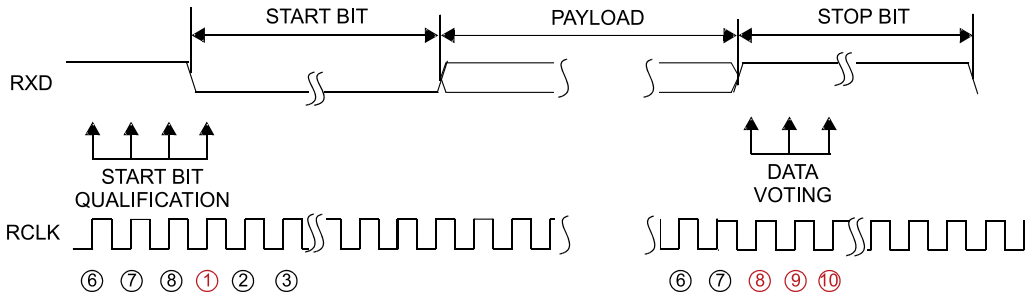
- DMA and CPU operating modes are mutually exclusive for the data/frame transfer on a UART or LIN node. Once a DMA transfer is completed, the CPU can manage subsequent accesses
- The DMA enable input signals may be tied high (enabling DMA for the respective channels) or low (disabling DMA for the respective channels). The DMA enable inputs do not need to be actively toggled to enable or disable DMA for the respective channels.

18.3.13. LINFlexD clock tolerance

Faster receiver tolerance

If the receiver has a higher baud rate than the transmitter, stop bit sampling may start during the last transmitted payload bit. To ensure the correct noise and framing error free reception, samples RS8,RS9,and RS10 must be located in the transmitted stop bit as shown in ***Figure 18-26 Faster receiver***.

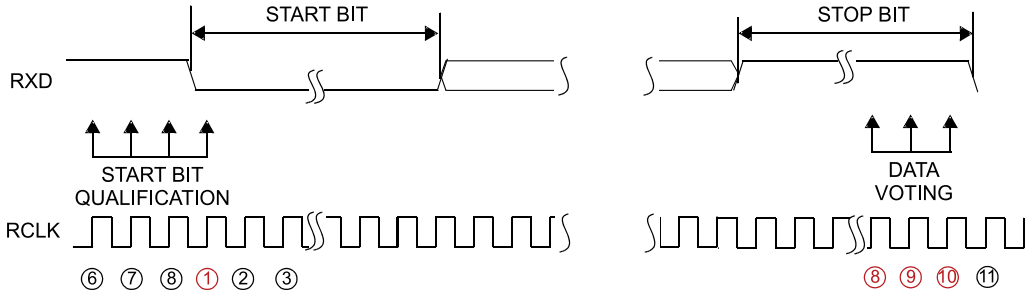
Figure 18-26 Faster receiver



Slower receiver tolerance

If the receiver has a slower baud rate than the transmitter, stop bit sampling may still be running while the next start bit is transmitted. To ensure the correct noise and framing error free reception, samples RS8, RS9, and RS10 must be located in the transmitted stop bit as shown in ***Figure 18-27 Slower receiver***.

Figure 18-27 Slower receiver



18.3.14. Low-power mode

STOP Function

Configuring LINFlexD for low-power mode involves notifying the module, ensuring active operations are completed, disabling clocks, and additional steps for UART FIFO mode. The specific steps are as follows:

1. Notify LINFlexD to enter Low-Power mode
   - Set the register LIN_STCTL.STOP bit to notify LINFlexD of the planned clock disablement.

Note: Ensure that there are no active DMA transmissions running.

2. Wait for current tasks to complete
   - Once LINFlexD completes all ongoing send, receive, or DMA operations, the LIN_STCTL.ACK bit will be set, indicating that the stop function is active.
3. Disable LINFlexD Clocks
   - After the LIN_STCTL.ACK bit is set, software can disable the clocks by RCU.
4. Automatic reset of LIN_STCTL.ACK bit
   - When the LIN_STCTL.STOP bit is cleared to 0, the LIN_STCTL.ACK bit will automatically reset.

Note: In UART FIFO mode, ensure correct operation by setting the LINCR1.INIT bit to force the receiver into idle state before clock gating

TX/RX INV Function

When the LIN_STCTL.txinv bit is set to 1, the polarity of the tx pin output level is inverted (0 becomes the recessive state, and 1 becomes the dominant state); when the LIN_STCTL.rxinv bit is set to 1, the polarity of the rx pin output level is inverted.

The level inversion function is only supported in UART MODE; configuring this bit in LIN mode may lead to errors.

18.3.15. LINS status

The description of bits 0-3 in the LIN status register (LINSR) can be referenced in the table below.

Table 18-9. LINS encodings

LINS[3:0]	LIN state	Description
0000b	Sleep mode	The LINFlexD is in Sleep mode to reduce power consumption
0001b	Init mode	The LINFlexD is in Initialization mode.

LINS[3:0]	LIN state	Description
0010b	Idle	The LINFlexD enters the Idle state when: ·LINCR1.SLEEP and LINCR1.INIT are cleared by software. ·A wakeup pulse is received rx pin and AUTOWU of LINCR1 is set. ·The previous frame transmission or reception is completed or aborted.
0011b	Break	In slave mode, a falling edge followed by a dominant state was detected. The LINFlexD is receiving a break field. In master mode, break field transmission is ongoing. **Note:** In slave mode, upon any error, the LIN state is either Idle or Break, depending on the last bit detected on rx pin. If last bit detected is dominant, the new LIN state is Break; otherwise, the new LIN state is Idle
0100b	Break Delimiter	In slave mode, a valid break field was detected (10 or 11 bit times depending on LINCR1.SSBL) and the LINFlexD is waiting for a rising edge. In master mode, break field transmission is complete, break delimiter transmission is ongoing.
0101b	Sync Field	In slave mode, a valid break delimiter was detected (recessive state for at least one bit time) and the LINFlexD is receiving a sync byte field. In master mode, sync byte field transmission is ongoing.
0110b	identifier Field	n slave mode, a valid sync field was received and the LINFlexD is receiving an identifier filed. In master mode, identifier field transmission is ongoing.
0111b	Header Reception/ Transmission	In slave mode, a valid header was received and the identifier field is available in BIDR. In master mode, header transmission is complete.
1000b	Data Reception/ Transmission	In receiver mode, reception is ongoing. In transmitter mode, response transmission is ongoing.
1001b	Checksum	Data transmission/reception is complete. Checksum transmission/reception is ongoing.

18.3.16. Header Trigger

To trigger transmission of a LIN header, the application can assert trigger source signal instead of using a peripheral bus write cycle to set the LINCR2.HTRQ bit. A rising edge on trigger signal sets LINCR2.HTRQ at the next rising edge of CK_LINFlexD. The minimum pulse width for trigger signal is one CK_LINFlexD. trigger signal must not remain asserted beyond the time at which the header transfer is complete.

Note: The Header Trigger function uses a rising edge trigger mechanism, and the trigger source signal can only be re-enabled after the current frame data transmission is completed; otherwise, it will cause the trigger signal to be lost or abort the next frame of LIN transmission.

18.4. Register definition

LINFlexD0 base address: 0x4001 0800
LINFlexD1 base address: 0x4001 0C00
LINFlexD2 base address: 0x4001 1000
LINFlexD3 base address: 0x4001 1400
LINFlexD4 base address: 0x4001 1800
LINFlexD5 base address: 0x4001 1C00

Table 18-10. LINFlexD register overview

Register Name	Symbol	Address Offset	Reset Value
LIN control register 1	LINCR1	0x00	0x0000 0082
LIN interrupt enable register	LINIER	0x04	0x0000 0000
LIN status register	LINSR	0x08	0x0000 0040
LIN error status register	LINESR	0x0C	0x0000 0004
UART mode control register	UARTCR	0x10	0x0000 0000
UART mode status register	UARTSR	0x14	0x0000 0000
LIN timeout control status register	LINTCSR	0x18	0x0000 0200
LIN output compare register	LINOCR	0x1C	0x0000 FFFF
LIN timeout control register	LINTOCR	0x20	0x0000 0E2C
LIN fractional baud rate register	LINFBRR	0x24	0x0000 0000
LIN integer baud rate register	LINIBRR	0x28	0x0000 0000
LIN checksum field register	LINCFR	0x2C	0x0000 0000
LIN control register 2	LINCR2	0x30	0x0000 0600
LIN buffer identifier register	BIDR	0x34	0x0000 0000
LIN buffer data register least significant	BDRL	0x38	0x0000 0000
LIN buffer data register most significant	BDRM	0x3C	0x0000 0000
LIN identifier filter enable register	IFER	0x40	0x0000 0000
LIN identifier filter match index	IFMI	0x44	0x0000 0000
LIN identifier filter mode register	IFMR	0x48	0x0000 0000
LIN identifier filter control register	IFCRn	0x4C	0x0000 0000
LIN global control register	GCR	0x8C	0x0000 0000
LIN UART preset timeout register	UARTPTO	0x90	0x0000 0FFF
LIN UART current timeout register	UARTCTO	0x94	0x0000 0000
LIN DMA Tx enable register	DMATXE	0x98	0x0000 0000
LIN DMA Rx enable register	DMARXE	0x9C	0x0000 0000
LIN stop control register	LIN_STCTL	0xA0	0x0000 0000

18.4.1. LIN control register 1(LINCR1)

Address offset: 0x00
Reset value: refer to Note
The LINCR1 register consists of control bits used to configure features of the LINFlexD.
Note: When accessing the LINCR1 register, each reserved bit should be written with its original reset value.



Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value.
16	NLSE	LIN state capture enable on bit error NLSE enables/disables capture of the LIN state (LINSR.LINS) whenever a bit error flag occurs (LINESR.BEF is set to 1). 0: LINSR.LINS shows the current LIN state and does not capture the LIN state on bit error. 1: LINSR.LINS is captured when a bit error flag occurs. **Note**: If NLSE = 1 and LINCR2.IOBE = 1, the internal LIN state is not captured when state transition to Idle occurs. LINSR.LINS shows the state in which the bit error occurred.
15	CCD	Checksum calculation disable 0: Checksum calculation is done by hardware. The LINCFR register is read-only. 1: Checksum calculation is disabled. The LINCFR register is read/write. If the checksum field is enabled (CFD=0), LINCFR can be programmed to send a software-calculated checksum/CRC. CCD can be read in any mode, but can be written only during Initialization mode.
14	CFD	Checksum field disable 0: Checksum field is sent after the required number of data bytes are sent. 1: No checksum field is sent in the frame. CFD can be read in any mode, but can be written only during initialization mode.
13	LASE	LIN autosynchronization enable-enables the auto synchronization feature described in ***Automatic resynchronization***. 0: Disable auto synchronization. 1: Enable auto synchronization. LASE can be read in any mode, but can be written only during initialization mode
12	AUTOWU	Automatic Wake-Up 0: The SLEEP bit is cleared by software only. 1: The SLEEP bit is cleared by hardware whenever the LINSR.WUF bit is set. AUTOWU is effective in both LIN and UART modes. AUTOWU can be read in any mode, but can be written only during Initialization mode
11:8	MBL[3:0]	Master break length-controls the length of the break field generated in LIN master mode 0000:10-bit break length

0001:11-bit break length
0010:12-bit break length
0011:13-bit break length
0100:14-bit break length
0101:15-bit break length
0110:16-bit break length
0111:17-bit break length
1000:18-bit break length
1001:19-bit break length
1010:20-bit break length
1011:21-bit break length
1100:22-bt break length
1101:23-bt break length
1110:36-bit break length
1111:50-bit break length
MBL can be read in any mode, but can be written only during Initialization mode.

7	BF	Bypass filter-controls the receive filter bypass function 0: Receiver ignores incoming frame if ID does not match any ID filter or if no ID filters are active. 1: If no ID filters are active, receiver responds to incoming frames. If ID filters are active, receiver responds if frame is received but does not match any filter BF can be read in any mode, but can be written only during Initialization mode.
6	Reserved	Must be kept at reset value. When accessing LINCR1, each reserved bit should be written with its original reset value.
5	LBKM	Loop back mode-enables or disables loop back mode as described in ***Loop back mode***. 0: Disable loop back mode. 1: Enable loop back mode. LBKM is effective in both LIN and UART modes. LBKM can be read in any mode, but can be written only during Initialization mode.
4	MME	Master mode enable 0: Slave mode. 1: Master mode. MME can be read in any mode, but can be written only during Initialization mode.
3	SSBL	Slave mode sync break length-sets the number of bit times for break field detection in slave mode 0: 11-bit break length 1: 10-bit break length SSBL can be read in any mode, but can be written only during Initialization mode.
2	RBLM	Receiver buffer locked mode 0: Receiver is buffer is not locked. The next incoming message overwrites the

		previous one. 1: Receiver buffer is locked against overrun. Once the buffer is full the next incoming message is discarded if the buffer is not released by software clearing the LINSR.RMB bit. RBLM is effective in both LIN and UART modes. RBLM can be read in any mode, but can be written only during Initialization mode.
1	SLEEP	Sleep mode request-set by software to request linflexd to enter sleep mode. The SLEEP bit is cleared by software to exit sleep mode. If LINCR1.AUTOWU and LINSR.WUF are set, SLEEP is automatically cleared by hardware to exit sleep mode. SLEEP is effective in both LIN and UART modes.
0	INIT	Initialization mode request-set by software to switch the hardware into initialization mode. When software clears the INIT bit (and if the SLEEP bit is also 0), the LINFlexD enters Normal mode. INIT is effective in both LIN and UART modes.

18.4.2. LIN interrupt enable register (LINIER)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SZIE	OCIE	BEIE	CEIE	HEIE	Reserved		FEIE	BOIE	LSIE	WUIE	Reserved	TOIE	DRIE	DTIE	HRIE
row	rw	rw	rw	rw			rw	rw	rw	rw		rw	rw	rw	rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	SZIE	Stuck at zero interrupt enable 0: Interrupt disabled. 1: Interrupt enabled. An interrupt is generated when the Stuck at Zero Flag (SZF) is set in LINESR or UARTSR.
14	OCIE	Output compare interrupt enable. 0: Interrupt disabled. 1: Interrupt enabled. An interrupt is generated when the Output Compare Flag (OCF) is set in LINESR or UARTSR.
13	BEIE	Bit error interrupt enable. 0: Interrupt disabled. 1: Interrupt enabled. An interrupt is generated when the Bit Error Flag (BEF) is set

		in LINESR.
12	CEIE	Checksum error interrupt enable. 0: Interrupt disabled. 1: Interrupt enabled. An interrupt is generated when the Checksum Error Flag (CEF) is set in LINESR.
11	HEIE	Header error interrupt enable. 0: Interrupt disabled. 1: Interrupt enabled. An interrupt is generated when the any of the following flags are set in LINESR: SFEF, SDEF, IDPEF.
10:9	Reserved	Must be kept at reset value.
8	FEIE	Frame error interrupt enable. 0: Interrupt disabled. 1: Interrupt enabled. An interrupt is generated when the Frame Error Flag(FEF) is set in LINESR or UARTSR.
7	BOE	Buffer overrun interrupt enable. 0: Interrupt disabled. 1: Interrupt enabled. An interrupt is generated when the Buffer Overrun Flag(BOF)is set in LINESR or UARTSR.
6	LSIE	LIN state interrupt enable. 0: Interrupt disabled. 1: Interrupt enabled. An interrupt generated on entering the following states: Sync Del, Sync Field, Identifier Field, Checksum
5	WUIE	Wakeup interrupt enable. 0: Interrupt disabled. 1: Interrupt enabled. An interrupt generated when the Wakeup Flag (WUF) is set in LINESR or UARTSR
4	Reserved	Must be kept at reset value.
3	TOIE	Timeout interrupt enable. 0: Interrupt disabled. 1: Interrupt enabled. An interrupt generated when the LINFlexD is in UART mode and the Timeout bit (TO) is set in UARTSR
2	DRIE	Data reception complete interrupt enable. 0: Interrupt disabled. 1: Interrupt enabled. An interrupt generated when the Data Received Flag (DRF) is set in LINESR or UARTSR.
1	DTIE	Data transmitted interrupt enable. 0: Interrupt disabled. 1: Interrupt enabled. An interrupt generated when the Data Transmitted Flag (DTF)

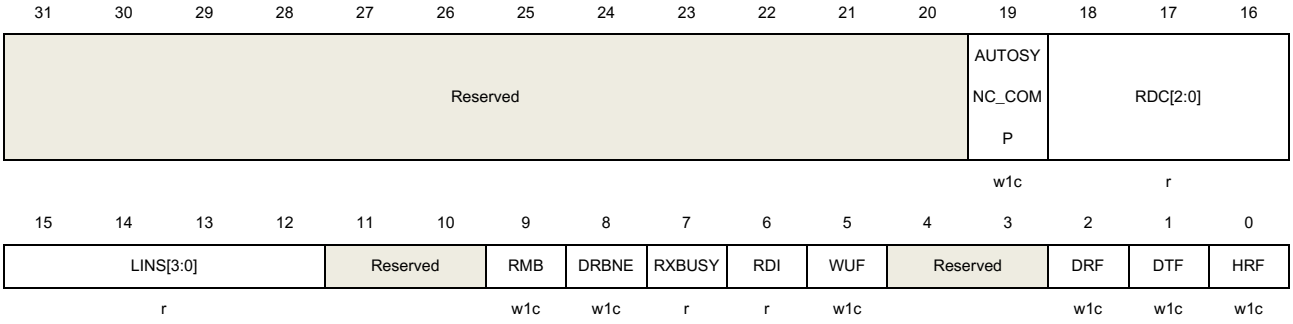
		is set in LINESR or UARTSR.
0	HRIE	Header received interrupt enable. 0: Interrupt disabled. 1: Interrupt enabled. An interrupt is generated Header Received Flag(HRF)is set in LINSR.

18.4.3. LIN status register (LINSR)

Address offset: 0x08

Reset value: 0x0000 0040

The LINSR register contains status bits indicating the state of the LINFlexD hardware.



Bits	Fields	Descriptions
31:20	Reserved	Must be kept at reset value.
19	AUTOSYNC_COMP	Autosynchronization complete AUTOSYNC_COMP is set when auto synchronization is enabled (LINCR1.LASE is set) and auto synchronization is complete. After AUTOSYNC_COMP is set, the contents of LINIBRR and LINFBRR registers can be read.
18:16	RDC[2:0]	Receive data byte count RDC contains the number of entries (bytes)in the receive data buffer in LIN mode 000: 1 byte 001: 2 bytes 010: 3 bytes 011: 4 bytes 100: 5 bytes 101: 6 bytes 110: 7 bytes 111: 8 bytes RDC is a read-only field available for debug purposes.
15:12	LINS[3:0]	LIN State. See ***LINS status***. In UART mode, the Idle, Init, Sleep, and Data Transmission/Reception states are flagged by the LIN status bits. **Note:** The value of the LINS bit field does not change by any write operation to it.

		Writing 0xF to LINS clears the Rx interrupt only when set due to the LIN state event.
11:10	Reserved	Must be kept at reset value.
9	RMB	Release message buffer 0: Buffer is free. 1: Buffer data is ready to be read by software. RMB should be cleared by software after reading the data received in the buffer RMB is cleared by hardware in Initialization mode
8	DRBNE	Data reception buffer not empty flag DRBNE is set by hardware as soon as the first byte of response is received and stored in BDRL (when there is at least one data byte in reception buffer). Software should clear DRBNE after reading all the buffers. DRBNE can be checked by software in the case of a response timeout event. DRBNE is cleared by hardware in Initialization mode.
7	RXBUSY	Receiver busy flag 0: Receiver is idle. 1: Reception is ongoing. **Note**: In slave mode, after header reception, if BIDR.DIR=0 and reception starts, the RXBUSY bit is set. In this case, software cannot set the LINCR2.DTRQ bit.
6	RDI	Receiver data input-indicates the current state of the rx pin. **Note**: After reset is released, RDI reflects the actual value of the rx pin.
5	WUF	Wakeup flag WUF is set by hardware when a falling edge is detected on the rx pin in either of the following conditions: ■ The LINFlexD is in slave mode and in Sleep mode. ■ The LINFlexD is in master mode is in either Sleep mode or Idle state. WUF is cleared by hardware in Initialization mode.
4:3	Reserved	Must be kept at reset value.
2	DRF	Data reception completed flag DRF is set by hardware and indicates that data reception has completed. DRF is cleared by hardware in Initialization mode. **Note**: DRF is not set when a framing error or checksum error occurs.
1	DTF	Data transmission completed flag DTF is set by hardware and indicates that data transmission has completed. DTF is cleared by hardware in Initialization mode. **Note**: DTF is not set in LIN mode when a bit error occurs and LINCR2.IOBE=0.
0	HRF	Header received flag HRF is set by hardware when header reception completes and one of the following conditions is true ■ All filters are inactive and LINCR1.BF=1.

- No match in any filter and LINCR1.BF=1.
- Tx filter match.

HRF is reset by hardware in Initialization mode and at end of completed or aborted frame.

18.4.4. LIN error status register (LINESR)

Address offset: 0x0C

Reset value: 0x0000 0004

The LINESR register contains status bits for various error conditions. For detailed descriptions in ***Errors***.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SZF	OCF	BEF	CEF	SFEF	SDEF	IDPEF	FEF	BOF	Reserved						NF
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1							rc_w1

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	SZF	Stuck at zero flag-set by hardware when a stuck-at-zero timeout error occurs.
14	OCF	Output compare flag In master mode, OCF is set when counter LINTCSR.CNT matches the content of LINOCR.OC2. In slave mode, OCF is set when the content of the counter LINTCSR.CNT matches the content of LINOCR.OC1 or LINOCR.OC2. If OCF is set when LINTCSR.MODE=0 and LINTCSR.IOT=1, the LINFlexD goes to Idle state. When LINTCSR.MODE=0, OCF is cleared by hardware in Initialization mode. When LINTCSR.MODE=1, OCF maintains its status regardless of the LIN state.
13	BEF	Bit error flag BEF is set by hardware when the LINFlexD detects a bit error BEF is cleared by hardware in Initialization mode.
12	CEF	Checksum error flag CEF is set by hardware when the received checksum does not match the hardware calculated checksum. CEF is cleared by hardware in Initialization mode. **Note**: CEF s never set if LINCR1.CCD or LINCR1.CFD is set.
11	SFEF	Sync field error flag SFEF is set by hardware when a sync field error occurs due to an inconsistent sync

		field. SFEF is cleared by hardware in Initialization mode.
10	SDEF	Sync Break delimiter error flag SDEF is set by hardware when a break delimiter error occurs because a break delimiter is too short (less than one bit time). SDEF is cleared by hardware in Initialization mode.
9	IPDEF	ID parity error flag IPDEF is set by hardware LINFlexD detects an error in the ID parity IPDEF is cleared by hardware in Initialization mode.
8	FEF	Framing error flag FEF is set by hardware when a framing error (invalid stop bit) occurs. FEF is cleared by hardware in Initialization mode.
7	BOF	Buffer overrun flag BOF is set by hardware when a new byte is received and LINSR.RMB bit is not cleared. BOF is cleared by hardware in Initialization mode.
6:1	Reserved	Must be kept at reset value.
0	NF	Noise flag NF is set by hardware when noise is detected in a receive character. NF is cleared by hardware in Initialization mode

18.4.5. UART mode control register (UARTCR)

Address offset: 0x10

Reset value: 0x0000 0000

Note: Always write '0' to reserved bits when writing to UARTCR.

Note: In UART mode, the LINFlexD does not support communication with special word length (UARTCR.WLS=1) in buffer mode.

Note: In UART mode, the LINFlexD supports communication with special word length (UARTCR.WLS=1) in FIFO mode only with UARTCR. WL [1:0]=11.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MIS	CSP[2:0]			OSR[3:0]				ROSE	NEF[2:0]			DTU	SBUR[1:0]		WLS
rw	rw			rw				rw	rw			rw	rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDFL_TFC[2:0]			RDFL_RFC[2:0]			RFBM	TFBM	WL1	PC1	RXEN	TXEN	PC0	PCE	WL0	UART
rw			rw			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31	MIS	Monitor idle state 0: UART CTO monitors the number of bits to be received. 1: UART CTO monitors the idle state of the reception line.

		MIS can be read in any mode, but can be written only during initialization mode.
30:28	CSP[2:0]	Configurable sample point (i) The CSP bits select the sample point during reduced oversampling. CSP can have the following range of values for the specified oversampling rate: OSR = 0x04: The CSP value is restricted to integers 0x02 or 0x03. OSR = 0x05: The CSP value is restricted to integers 0x02, 0x03, or 0x04. OSR = 0x06: The CSP value is restricted to integers 0x03, 0x04, or 0x05. OSR = 0x08: No valid CSP values are defined for this oversampling rate. CSP can be read in any mode, but can be written only during Initialization mode.
27:24	OSR[3:0]	Oversampling Rate-Selects the number of samples taken for a bit when reduced oversampling is enabled. The values allowed for OSR are 0x04, 0x05, 0x06, and 0x08. **Note**: When idle state monitoring is enabled (MIS=1), the allowed OSR values are 0x04 and 0x08. OSR can be read in any mode, but can be written only during Initialization mode.
23	ROSE	Reduced oversampling enable 0: Each bit is oversampled sixteen times. 1: OSR bits determine the oversampling rate. ROSE can be read in any mode, but can be written only during Initialization mode.
22:20	NEF[2:0]	Number of expected frames The NEF bits set the number of expected frames in UART reception mode. If the DTU bit is set, then the UART timeout counter is reset after the configured number of frames have been received. NEF can be read in any mode, but can be written only during Initialization mode.
19	DTU	Disable timeout in UART mode 0: Timeout must be handled by software. 1: Timeout in UART mode is disabled after the configured number of data frames are received. There is an internal counter that counts the number of frames received. **In Buffer mode**: The counter is reset whenever the number of bytes received (rec_byte_cnt) is equal to RDFL. At that time, the DRF bit is set, and num_of_frames (number of frames received) and rec_byte_cnt are reset. When num of frames is equal to NEF, disable timeout is generated. However, this clears num of frames. So, the timer restarts again. The value of counter is 0 when it restarts. **In FIFO mode**: When num_of_frames are equal to NEF, disable_timeout is generated, which clears num_of_frames and causes the timer to restart again. The value of counter is 0 when it restarts. **Note**: ■ disable_timeout causes the timer to restart, which enables timeout again.

■ Timer reset means it resets and starts counting again and the timeout is enabled.

DTU can be programmed in Initialization mode only when the UART bit is set.

18:17	SBUR[1:0]	Number of stop bits in UART reception mode: 00: 1 stop bit 01: 2 stop bits 10: 3 stop bits 11: Reserved When the UART is used for transmission and reception, set the same number of stop bits in GCR.STOP and UARTCR. SBUR. When the UART is used only as receiver, only the SBUR bits need to be set the specify the number of stop bits. SBUR can be read in any mode, but can be written only during Initialization mode, when the UART bit is set.
16	WLS	Special word length in UART mode 0: Disable 12-bit+parity bit in reception. 1: Enable 12-bit+parity bit in reception (UART mode) When WLS=1, the WL and PCE bits have no effect in UART reception, although parity check is enabled by default in this mode and PC0/1 bits can be used to select the parity control. The WLS bit has priority in UART reception mode. WLS can be read in any mode, but can be written only during Initialization mode, when the UART bit is set.
15:13	TDFL_TFC[2:0]	Transmitter data field length / Tx FIFO counter The TDFL_TFC field has one of two functions depending on the mode of operation. **UART Buffer Mode**: When the LINFlexD is in UART buffer mode (TFBM=0), TDFL_TFC is read/write and defines the number of bytes to be transmitted: x00: 1 byte x01: 2 bytes x10: 3 bytes x11: 4 bytes The first bit is reserved and not implemented. When the UART data length is configured as half-word (WL=10 or 11), the only valid values for TDFL_TFC are x01 and x11. **UART FIFO Mode**: When the LINFlexD is in UART FIFO mode (TFBM=1), TDFL_TFC is read-only and contains the number of entries (bytes)in the Tx FIFO: 000: Empty 001: 1 byte 010: 2 bytes 011: 3 bytes 100: 4 bytes All other values: Reserved

		TDFL_TFC can be read in any mode, but can be written only during Initialization mode, when the UART bit is set.
12:10	RDFL_RFC[2:0]	Receiver data field length/Rx FIFO counter The RDFL_RFC field has one of two functions depending on the mode of operation. **UART Buffer Mode**: When the LINFlexD is in UART buffer mode (RFBM=0), RDFL_RFC is read/write and defines the number of bytes to be received: x00: 1 byte x01: 2 bytes x10: 3 bytes x11: 4 bytes The first bit is reserved and not implemented. When the UART data length is configured as half-word (WL=10 or 11), the only valid values for RDFL_RFC are x01 and x11. RDFL_RFC should be programmed to be greater than or equal to NEF (number of expected frames). **UART FIFO Mode**: When the LINFlexD is in UART FIFO mode (RFBM=1),RDFL_RFC is read-only and contains the number of entries (bytes)in the Rx FIFO: 000: Empty 001: 1 byte 010: 2 bytes (1.5 byte if WLS=1) 011: 3 bytes 100: 4 bytes (2x1.5 bytes if WLS=1) All other values: Reserved RDFL_RFC can be read in any mode, but can be written only during Initialization mode, when the UART bit is set.
9	RFBM	Rx FIFO/Buffer mode 0: Rx buffer mode enabled 1: Rx FIFO mode enabled (mandatory in DMA Rx mode) RFBM can be read in any mode, but can be written only during Initialization mode, when the UART bit is set.
8	TFBM	Tx FIFO/Buffer mode 0: Tx buffer mode enabled 1: Tx FIFO mode enabled (mandatory in DMA Tx mode) TFBM can be read in any mode, but can be written only during Initialization mode, when the UART bit is set.
7	WL1	Word length in UART mode 1. WL1=0 and WLO=0: 7 bits data + parity. 2. WL1=0 and WLO=1: With parity disabled (PCE=0): 8-bit data format

With parity enabled (PCE=1): 8-bit data + parity bit

3. WL1=1 and WLO=0:

Data transmission is configured as 15-bit data + parity.

4. WL1=1 and WLO=1:

With parity disabled (PCE=0): 16-bit data format

With parity enabled (PCE=1): 16-bit data + parity

WL1 can be read in any mode, but can be written only during Initialization mode, when the UART bit is set.

6 PC1

Parity control

- PC1=0 and PC0=0

  The parity sent is even.

- PC1=0 and PC0=1

  The parity sent is odd.

- PC1=1 and PC0=0

  A logic 0 is always transmitted/ checked as parity bit.

- PC1=1 and PC0=1

  A logic 1 is always transmitted/ checked as parity bit

PC1 can be read in any mode, but can be written only during Initialization mode, when the UART bit is set.

5 RXEN

Receiver enable

0: Receiver disabled

1: Receiver enabled

RXEN is writable only when the UART bit is set.

4 TXEN

Transmitter enable

0: Transmitter disabled

1: Transmitter enabled

Transmission starts when TXEN is set and Data Byte 0 (BDRL.DATA0) is programmed.

TXEN is writable only when the UART bit is set.

3 PC0

Parity control

- PC1=0 and PC0=0

  The parity sent is even.

- PC1=0 and PC0=1

  The parity sent is odd.

- PC1=1 and PC0=0

  A logic 0 is always transmitted/ checked as parity bit.

- PC1=1 and PC0=1

  A logic 1 is always transmitted/ checked as parity bit

PC0 can be read in any mode, but can be written only during Initialization mode, when the UART bit is set.

2 PCE

Party control enable

0: Parity transmit/check disabled

1: Parity transmit/check enabled

		PCE can be read in any mode, but can be written only during Initialization mode, when the UART bit is set.
1	WL0	Word length in UART mode ■ WL1=0 and WL0=0 7 bits data+parity. ■ WL1=0 and WL0=1 8 bits data when PCE=0 or8 bits data+parity when PCE=1. ■ WL1=1 and WL0=0 15 bits data+parity. ■ WL1=1 and WL0=1 16 bits data when PCE=0 or16 bits data+parity when PCE=1 WLO can be read in any mode, but can be written only during Initialization mode, when the UART bit is set.
0	UART	0: LIN mode 1: UART mode UART can be read in any mode, but can be written only during Initialization mode.

18.4.6. UART mode status register(UARTSR)

Address offset: 0x14

Reset value: 0x0000 0000

The UARTSR register contains status bits for UART mode.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SZF	OCF	PE[3:0]				RMB	FEF	BOF	RDI	WUF	RFNE	TO	DRF_RFE	DTF_TFF	NF
rc_w1	rc_w1	rc_w1				rc_w1	rc_w1	rc_w1	r	rc_w1	r	rc_w1	rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	SZF	Stuck at zero flag SZF is set by hardware when 100 dominant bits are detected. An interrupt is generated if LINIER.SZIE=1. SZF reflects the same value as LINESR.SZF when the LINFlexD is not in Initialization mode and UARTCR.UART=1
14	OCF	Output compare flag: 0: No output compare event occurred. 1: The content of the timeout counter matched the content of LINOCR When OCF is set, an interrupt is generated if LINIER.OCIE=1. OCF reflects the same value as LINESR.OCF when the LINFlexD is not in Initialization mode and UARTCR.UART=1.

		Note: For baud rates above 1 Mbit/s, the OCF flag is not usable
13：10	PE[3:0]	Parity error flag-indicates parity error in corresponding receive data byte 0: No parity error detected 1: Parity error detected. Parity is checked after the complete frame is received. PE[3:0] bit usage depends on the values of UARTCR.WLS and UARTCR.WL1 When UARTCR.WLS=0 and UARTCR.WL1=0: – PE[3]=Parity error flag for fourth byte received – PE[2]=Parity error flag for third byte received. – PE[1]=Party error flag for second byte received – PE[0]=Parity error flag for first byte received. When UARTCR.WLS=1 or UARTCR.WL1=1: – PE[3]=Not used. – PE[2]=Parity error flag for third or fourth byte received PE[1]=Not used. – PE[0]=Parity error flag for first or second byte received
9	RMB	Release message buffer: 0: Buffer is free. 1: Buffer data is ready to be read by software. RMB reflects the same value as LINSR.RMB when the LINFlexD is not in Initialization mode and UARTCR.UART=1
8	FEF	Framing error flag FEF is set by hardware when the LINFlexD detects a framing error (invalid stop bit). An interrupt is generated if the FEIE bit of LINIER=1. FEF reflects the same value as LINESR.FEF when the LINFlexD is not in Initialization mode and the UART bit of UARTCR =1.
7	BOF	FIFO / Buffer overrun flag In UART buffer mode, BOF is set by hardware when there is a new byte received and the RMB bit is not cleared. If LINCR.RBLM=1, the newly received message is discarded. If LINCR.RBLM=0, the newly received message overwrites the buffer. In UART FIFO mode, BOF is set when there is a new byte received and the Rx FIFO is full. In UART FIFO mode, once Rx FIFO is full, the new received message is discarded regardless of the value of the RBLM bit. When BOF is set, an interrupt is generated if LINIER.BOIE=1. BOF reflects the same value as LINESR.BOF when the LINFlexD is not in Initialization mode and UARTCR.UART=0.
6	RDI	Receiver data input-reflects the current state of the rx pin when UARTCR.UART=1
5	WUF	Wakeup flag WUF is set by hardware when a falling edge is detected on the rx pin in Sleep mode. An interrupt is generated if LINIER.WUIE=1. WUF reflects the same value as LINSR.WUF when the LINFlexD is not in Initialization mode and UARTCR.UART=1

4	RFNE	Receive FIFO not empty RFNE is set by hardware in UART FIFO mode (RFBM=1), when there is at least one data byte present in the receive FIFO.RFNE is a read-only bit for debugging purposes. RFNE can be used by software in case of a timeout event.
3	TO	Timeout TO is set by hardware when a UART timeout occurs-in other words, the value of UARTCTO becomes equal to the present value of the timeout (UARTPTO register setting). An interrupt is generated if LINIER.TOIE=1. The GCR.SR bit should be used to reset the receiver FSM to the Idle state in the event of timeout for UART reception in both UART buffer and UART FIFO modes
2	DRF_RFE	Data reception completed flag / Rx FIFO empty flag The DRF_RFE bit function depends on whether the LINFlexD is operating in UART buffer mode or UART FIFO mode. **UART Buffer Mode:** In UART buffer mode (RFBM=0), the DRF (data received flag) function of DRF_RFE is used. DRF is set by hardware in UART buffer mode when the number of bytes programmed in UARTCR.RDFL are received. An interrupt is generated if LINIER.DRIE=1. DRF is set when the configured number of valid stop bits are received for the last frame. DRF is set regardless of parity error, overrun error, or framing error if the framing error is in the last STOP bit configured. DRF reflects the same value as LINSR.DRF when the LINFlexD is not in Initialization mode and UARTCR.UART=1. DRF can be read/cleared by software. Writing '1' clears DRF. **UART FIFO Mode:** In UART FIFO mode (RFBM=1), the RFE (Rx FIFO empty) function of DRF_RFE is used. RFE is set by hardware in UART FIFO mode when the Rx FIFO is empty. RFE is a read-only bit for debugging purposes. It is internally used by the DMA Rx interface.
1	DTF_TFF	Data transmission completed flag/ Tx FIFO full flag The DTF_TFF bit function depends on whether the LINFlexD is operating in UART buffer mode or UART FIFO mode. **UART Buffer Mode:** In UART buffer mode (RFBM=0), the DTF (data transmitted flag) function of DTF_TFF is used. DTF is set by hardware in UART buffer mode when data transmission completes. An interrupt is generated if LINIER.DTIE=1. DTF reflects the same value as LINSR.DTF when the LINFlexD is not in Initialization mode and UARTCR.UART=1. DTF can be read/cleared by software. Writing '1'clears DTF. **UART FIFO Mode:** In UART FIFO mode (RFBM=1), the TFF(Tx FIFO full)function of DTF_TFF is used. TFF is set by hardware in UART FIFO mode when Tx FIFO is full.

		TFF is a read-only bit for debugging purposes. It is internally used by the DMA Tx interface
0	NF	Noise flag-set by hardware when noise is detected in the received character. NF reflects the same value as LINESR.NF when the LINFlexD is not in Initialization mode and the UARTCR.UART=1. During reduced oversampling (UARTCR.ROSE=1),the NF bit is enabled only when UARTCR.OSR=8

18.4.7. LIN timeout control status register (LINTCSR)

Address offset: 0x18
Reset value: 0x0000 0200

The LINTCSR register contains control and status bits for the LIN timeout feature.

Note: When LINTCSR.MODE=0, any activity on the transmit or receive pins will cause an unwanted change in the value of the 8-bit field Output Compare Value 2(OC2) of the LIN Output Compare register (LINOCR). If the LINFlexD is enabled in LIN mode and the value of LINTCSR.MODE is changed from'1' to '0', the old values of LINOCR.OC1 and LINOCR.OC2 are retained. As a consequence, if the LINFlexD is reconfigured from UART mode to LIN mode, or LINTCSR.MODE is changed from '1' to '0', an incorrect timeout exception is generated when a LIN communication starts. To avoid this, set LINTCSR.MODE to '1', reconfigure the LINFlexD in LIN mode, then reset LINTCSR.MODE to '0'. Before switching LINTCSR.MODE from'1' to '0' between frames, load LINOCR with 0xFFFF.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					MODE	IOT	TOCE	CNT[7:0]							
					rw	rw	rw	r							

Bits	Fields	Descriptions
31:11	Reserved	Must be kept at reset value.
10	MODE	Timeout counter mode: 0: LIN mode 1: Output compare mode **Note**: Always set LINTCSR.MODE to '1' when UARTCR.UART=1. **Note**: Before switching LINTCSR.MODE from'1' to '0' between frames, load LINOCR with 0xFFFF. MODE can be read in any mode, but can be written only during Initialization mode.
9	IOT	Idle on timeout: 0: LIN state machine does not reset to Idle on timeout. The idle on timeout feature is applicable only when MODE=0. IOT can be read in any mode, but can be written

		only during Initialization mode. 1: LIN state machine resets to Idle on timeout event.
8	TOCE	Timeout counter enable: 0: Timeout counter disabled.UARTSR.OCF flag is not set on an output compare event. 1: Timeout counter enabled.UARTSR.OCF is set if an output compare event occurs. TOCE is always configurable by software in Initialization mode. if the timer is configured in LIN mode and the LINFlexD is not in Initialization mode, then hardware takes control of TOCE.
7:0	CNT[7:0]	Counter value - the value of the timeout counter. **Note**: For proper functionality of this counter, LINIBRR should be greater than or equal to 5.

18.4.8. LIN output compare register (LINOCR)

Address offset: 0x1C
Reset value: 0x0000 FFFF

The LINOCR register contains the value to be compared to the LINTCSR.CNT value. LINOCR is writable by software only in when the counter is in output compare mode.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2[7:0]								OC1[7:0]							
rw								rw							

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:8	OC2[7:0]	Output compare value 2
7:0	OC1[7:0]	Output compare value 1

18.4.9. LIN timeout control register (LINTOCR)

Address offset: 0x20
Reset value: 0x0000 0E2C
The LINTOCR register contains the LIN mode response and header timeout values.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				RTO[3:0]				Reserved	HTO[7:0]						

rw rw

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:8	RTO[3:0]	Response timeout value-The response timeout duration (in bit time) for 1 byte, The reset value is 0xE=14, corresponding to $T_{Response_Maximum} = 1.4 \times T_{Response_Nominal}$
7	Reserved	Must be kept at reset value.
6:0	HTO[7:0]	Header timeout value HTO specifies the header timeout duration (in bit time). HTO can be written only for slave mode. The header timeout should be programmed without considering 11 bits of break field

18.4.10. LIN fractional baud rate register(LINFBRR)

Address offset: 0x24
Reset value: 0x0000 0000

LINFBRR sets the fractional part of the LIN baud rate, as described in ***Baud rate generation***.
Note: When LINCR1.LASE=1, LINFBRR should be read only after LINSR.AUTOSYNC_COMP is set to obtain the correct value.
Note: LINFBRR cannot be used when reduced oversampling is enabled (UARTCR.ROSE=1).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												FBR[3:0]			
												rw			

Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3:0	FBR[3:0]	Fractional baud rate: 0000: Fraction(LDIV)=0 0001: Fraction(LDIV)=1/16 0010: Fraction(LDIV)=2/16 0011: Fraction(LDIV)=3/16 0100: Fraction(LDIV)=4/16 0101: Fraction(LDIV)=5/16 0110: Fraction(LDIV)=6/16 0111: Fraction(LDIV)=7/16 · 1000: Fraction(LDIV)=8/16 1001: Fraction(LDIV)=9/16 1010: Fraction(LDIV)=10/16

1011: Fraction(LDIV)=11/16
1100: Fraction(LDIV)=12/16
1101: Fraction(LDIV)=13/16
1110: Fraction(LDIV)=14/16
1111: Fraction(LDIV)=15/16
FBR can be read in any mode, but can be written only during Initialization mode

18.4.11. LIN integer baud rate register(LINIBRR)

Address offset: 0x28
Reset value: 0x0000 0000

LINIBR sets the integer part of the LIN baud rate, as described in ***Baud rate generation***.

Note: When LINCR1.LASE=1, LINIBRR should be read only after LINSR.AUTOSYNC_COMP is set to obtain the correct value.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												IBR[19:16]			
												rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IBR[15:0]															
rw															

Bits	Fields	Descriptions
31:0	IBR[19:0]	Integer baud rate The IBR bits, along with the fractional baud rate bits in LINFBRR, set the LIN baud rate: IBR=0x0: LIN clock disabled IBR=0x1: Mantissa(LDIV)=1 … IBR=0xFFFFE: Mantissa(LDIV)=1048574 IBR=0xFFFFF: Mantissa(LDIV)=1048575 BR can be read in any mode, but can be written only during Initialization mode.

18.4.12. LIN checksum field register(LINCFR)

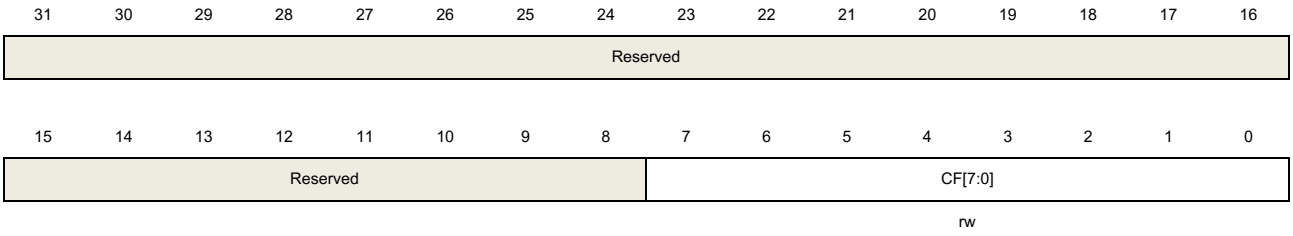
Address offset: 0x2C
Reset value: 0x0000 0000

Note: There is a delay between 4 to 6 clock cycles of CK_LINFlexD for the internal checksum's value (which is clocked with CK_LINFlexD/16*LDIV)to reflect on LINCFR.

Note: Due to synchronization structures between the two input clocks to the LINFlexD, the write to this register is delayed between 2 to 3 clock cycles of CK_LINFlexD.

The LINCR1 checksum configuration is formally described as follows:

1. CFD=1 and CCD=1
   The register accessibility is read/write
   Checksum generation is disabled
2. CFD=1 and CCD=0
   The register accessibility is read-only
   Checksum generation is disabled
3. CFD=0 and CCD=1
   The register accessibility is read/write
   Uses programmed checksum values for transmission
4. CFD=0 and CCD=0
   The register accessibility is read-only
   Uses Calculated checksum values for transmission



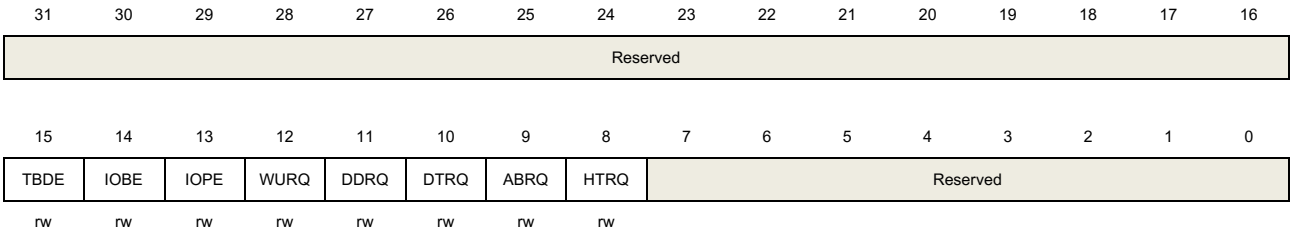
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	CF[7:0]	Checksum bits When LINCR1.CCD=0, the CF bits are read-only and are calculated by hardware. When LINCR1.CCD=1,the CF bits can be written by software.

18.4.13. LIN control register 2(LINCR2)

Address offset: 0x30
Reset value: 0x0000 6000

The LINCR2 register contains control and status bits related to buffer operations.

Note: When accessing LINCR2, each reserved bit should be written with its original reset value.



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	TBDE	Two-Bit delimiter enable:

0: Delimiter length in break field is 1 bit.
1: Delimiter length in break field is 2 bits.
TBDE can be read in any mode, but can be written only during Initialization mode.

14 IOBE
Idle on bit error:
0: Bit error does not reset LIN state machine.
1: Bit error resets LIN state machine.
IOBE can be read in any mode, but can be written only during Initialization mode.

13 IOPE
Idle on identifier parity error
0: Parity error does not reset LIN state machine.
1: Parity error resets LIN state machine.
IOPE can be read in any mode, but can be written only during Initialization mode.

12 WURQ
Wakeup generate request
Setting WURQ generates a wakeup request. The character sent during wakeup is copied from BDRL.DATA0. WURQ is reset by hardware when the wakeup character has been transmitted.
WURQ cannot be set in Sleep mode. Software must exit Sleep mode before setting WURQ.
Bit error is not checked when transmitting the wakeup request.

11 DDRQ
Data discard request
DDRQ is set by software to stop data reception if the frame does not concern the node. DDRQ is reset by hardware once the LINFlexD ignores the response and moves to idle state.
For a LIN slave, DDRQ can be set only when LINSR.HRF=1 and the identifier is software-filtered.

10 DTRQ
Data transmission request
DTRQ is set by software in slave mode to request the transmission of the LIN data field stored in the buffer data register. DTRQ can be set only when LINSR.HRF=1 (to ensure that data transmission is requested only after a header reception).
DTRQ is cleared by hardware when the request has been completed, or on abort request or error condition.
In master mode, DTRQ is set by hardware when BIDR.DIR=1 and header transmissions complete.

9 ABRQ
Abort request
ABRQ is set by software to abort the current transmission. The LINFlexD aborts the transmission at the end of the current bit.
ABRQ is cleared by hardware when the transmission has been aborted. ABRQ can abort a wakeup request and can also be used in UART mode

8 HTRQ
Header transmission request
HTRQ is set by software to request the transmission of a LIN header HTRQ is cleared by hardware when the request has been completed or on abort request.
HTRQ has no effect in UART mode.

Note: In master mode, if both HTRQ and ABRQ are set at the same time then ABRQ has no effect. Similarly, in slave mode after header reception, if DTRQ and ABRQ are simultaneously set then ABRQ has no effect.

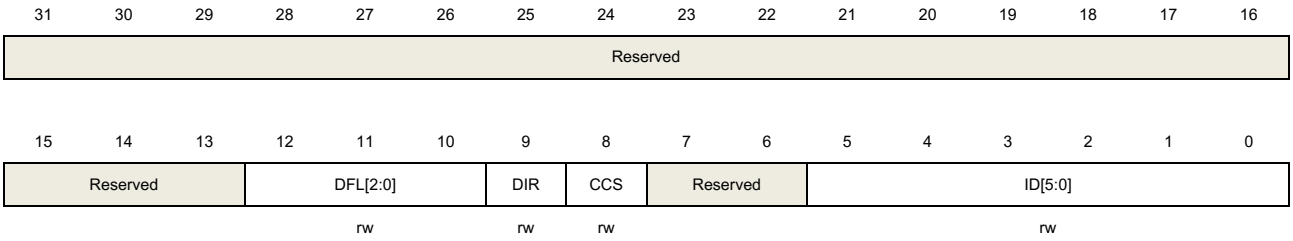
7:0	Reserved	Must be kept at reset value.

18.4.14. LIN buffer identifier register (BIDR)

Address offset: 0x34
Reset value: 0x0000 0000

The BIDR register contains bits that provide information about the identifier of the transaction and other related information.

Note: All the fields (ID, CSS, DIR, DFL) of the BIDR register must be updated when an ID filter(enabled) in Slave mode (Tx or Rx) matches the ID received.



Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12:10	DFL[2:0]	Data Field Length-Number of data bytes in the response part of the frame. DFL=Number of data bytes-1.
9	DIR	Direction-Controls the direction of the data field 0: LINFlexD receives the data and copies it to the BDRM/BDRL registers. 1: LINFlexD transmits the data from the BDRM/BDRL registers.
8	CCS	Classic checksum-controls the type of checksum applied on the current message 0: Enhanced checksum covering identifier and data fields. This is compatible with LIN specification rev.2.0 and higher. 1: Classic checksum covering data field only. This is compatible with LIN specification rev.1.3 and lower.
7:6	Reserved	Must be kept at reset value.
5:0	ID[5:0]	Identifier part of the identifier field without the identifier parity. The ID field can be written only in master mode (LINCR1.MME=1).

18.4.15. LIN buffer data register least significant (BDRL)

Address offset: 0x38
Reset value: 0x0000 0000

The BDRM register holds bytes 4-7 of the 8-byte data buffer.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA3[7:0]								DATA2[7:0]							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA1[7:0]								DATA0[7:0]							
rw								rw							

Bits	Fields	Descriptions
31:24	DATA3[7:0]	Data Byte 3-Byte 3 of the data field
23:16	DATA2[7:0]	Data Byte 2-Byte 2 of the data field
15:8	DATA1[7:0]	Data Byte 1-Byte 1 of the data field
7:0	DATA0[7:0]	Data Byte 0-Byte 0 of the data field

18.4.16. LIN buffer data register most significant (BDRM)

Address offset: 0x3C

Reset value: 0x0000 0000

The BDRM register holds bytes 4-7 of the 8-byte data buffer.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA7[7:0]								DATA6[7:0]							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA5[7:0]								DATA4[7:0]							
rw								rw							

Bits	Fields	Descriptions
31:24	DATA7[7:0]	Data Byte 7-Byte 7 of the data field
23:16	DATA6[7:0]	Data Byte 6-Byte6 of the data field
15:8	DATA5[7:0]	Data Byte 5-Byte 5 of the data field
7:0	DATA4[7:0]	Data Byte 4-Byte 4 of the data field

18.4.17. LIN identifier filter enable register (IFER)

Address offset: 0x40

Reset value: 0x0000 0000

The IFER register enables / disables the identifier filters. There is one filter active (FACT) bit for each filter.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FACT15	FACT14	FACT13	FACT12	FACT11	FACT10	FACT9	FACT8	FACT7	FACT6	FACT5	FACT4	FACT3	FACT2	FACT1	FACT0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	FACTx	Software sets FACTx to activate the filter x in identifier list mode. (x = 0…15) In identifier mask mode, FACT[2n+1] have no effect on the corresponding filters as they act as mask for identifier 2n. FACTx can be read in any mode, but can be written only during Initialization mode.

18.4.18. LIN identifier filter match index (IFMI)

Address offset: 0x44

Reset value: 0x0000 0000

The IFMI register contains the index corresponding to the received identifier. It can be used to read or write the data directly in RAM.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											IFMI[4:0]				
											r				

Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
4:0	IFMI[4:0]	Filter Match Index-Contains the index corresponding to the received identifier Upon a filter match with filter x, IFMI[N:0]=x+1. On no match, IFMI is equal to 0x00. IFMI can be used to directly write or read the data in RAM. (see ***Slave Mode*** for details)

18.4.19. LIN identifier filter mode register(IFMR)

Address offset: 0x48

Reset value: 0x0000 0000

The IFMR register configures the modes of filters. There is one IFM bit for each pair of filters.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								IFM7	IFM6	IFM5	IFM4	IFM3	IFM2	IFM1	IFM0
								rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	IFMn	Filter mode (n = 0…7) 0: Filters 2n and 2n+1 are in identifier list mode. 1: Filters 2n and 2n+1 are in mask mode. Filter 2n+1 is the mask for filter 2n. IFM[n] can be read in any mode, but can be written only during initialization mode.

18.4.20. LIN identifier filter control register (IFCRn) (n = 0…15)

Address offset: 0x04C+(n × 0x4)
Reset value: 0x0000 0000

There is an Identifier Filter Control Register(IFCRn) for each filter.

Even-numbered instances of IFCRn(IFCR0, IFCR2, IFCR4,and so on)act as fiters in both identifier list and identifier mask modes.

Odd-numbered instances of IFCRn (IFCR1, IFCR3, IFCR5,and so on)act as:

- Filters in identifier list mode.
- Amask for the preceding-numbered register in identifier mask mode. For example, if filters 0 and 1 are configured for identifier mask mode, IFCR1 acts as a mask for IFCR0.

The IFCRn registers are read-only in Normal mode and can be written only in Initialization mode.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			DFL[2:0]			DIR	CSS	Reserved		ID[5:0]					
			rw			rw	rw			rw					

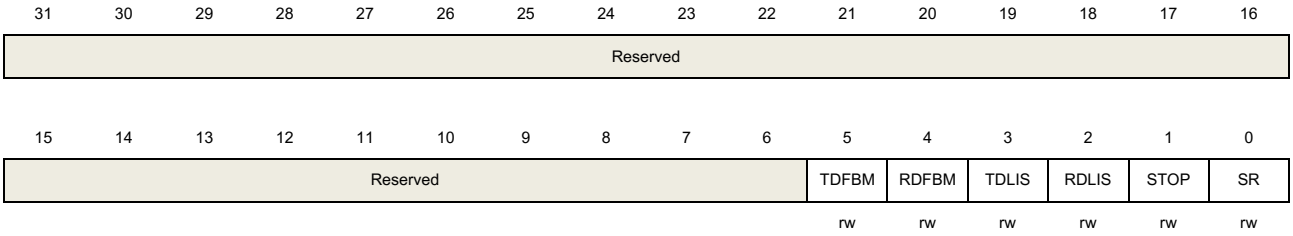
Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12:10	DFL[2:0]	Data field length-number of data bytes in the response part of the frame. DFL=Number of data bytes-1.
9	DIR	Direction-controls the direction of the data field: 0: LINFlexD receives the data and copy it to the BDRM/BDRL registers. 1: LINFlexD transmits the data from the BDRM/BDRL register.
8	CSS	Classic checksum-controls the type of checksum applied on the current message:

		0: Enhanced checksum covering identifier and data fields. This is compatible with LIN specification rev.2.0 and higher. 1: Classic checksum covering data field only. This is compatible with LIN specification rev.1.3 and lower.
7:6	Reserved	Must be kept at reset value.
5:0	ID[5:0]	Identifier-identifier part of the identifier field without the identifier parity.

18.4.21. LIN global control register (GCR)

Address offset: 0x8C
Reset value: 0x0000 0000

The GCR register contains control bits that apply to both LIN and UART modes.
The GCR register is read - only in Normal mode and can be written only in Initialization mode.



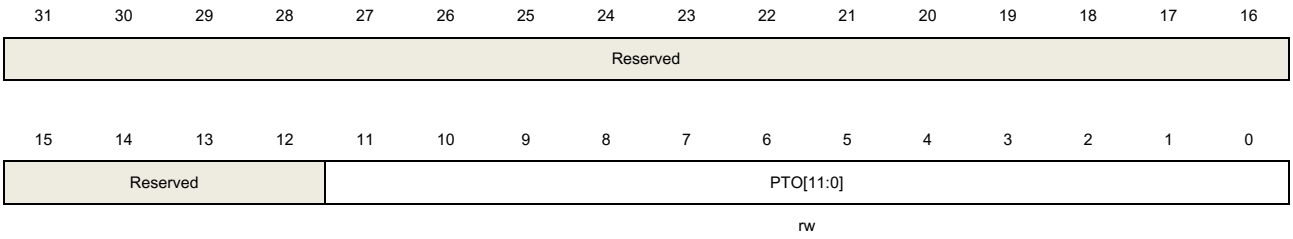
Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5	TDFBM	Transmit data first bit MSB-Controls whether the first bit of transmit data (payload only) is the most or least significant bit from the respective buffer data register. 0: The first bit of transmitted data is the least significant bit of the buffer Data register (BDRM / BDRL[0],BDRM / BDRL[8], BDRM/BDRL[16], BDRM / BDRL[24]). 1: The first bit of transmitted data is the most significant bit of the buffer Data register (BDRM / BDRL[7], BDRM/BDRL[15], BDRM/BDRL[23], BDRM / BDRL[31]).
4	RDFBM	Receive data first bit MSB-Controls whether the first bit of received data (payload only) is mapped to the most or least significant bit from the respective buffer data register. 0: The first bit of received data is mapped to least significant bit of the buffer data register (BDRM/BDRL[0], BDRM/BDRL[8], BDRM/BDRL[16], BDRM/BDRL[24]). 1: The first bit of received data is mapped to the most significant bit of the buffer data register (BDRM / BDRL[7], BDRM / BDRL[15], BDRM / BDRL[23], BDRM / BDRL[31]).
3	TDLIS	Transmit data level inversion selection-controls the data inversion of transmitted data (payload only). 0: Transmitted data is not inverted. 1: Transmitted data is inverted.

2	RDLIS	Received data level inversion selection-controls the data inversion of received data (payload only). 0: Received data is not inverted 1: Received data is inverted.
1	STOP	One or two stop bit configuration-controls the number of stop bits transmitted for all fields (delimiter, sync, ID, checksum, payload). 0: One stop bit 1: Two stop bits
0	SR	Soft reset Writing '1' to SR executes a soft reset of the LINFlexD (FSMs, FIFO pointers, counters, timers, status and error registers) without modifying the configuration registers.SR should be cleared by software to perform further operations (SR is not cleared by hardware). The SR bit can be written only be software in Initialization mode. The read value of SR is always '0'. The following content lists the register fields that are reset by writing '1' to SR. LINSR Register : All fields are reset except RXBUSY and AUTOSYNC_COMP. LINESR Register : All fields are fully reset. LINTCSR Register : Only the CNT[7:0] field is reset. UARTSR Register : All fields are reset except RFE and TFF. UARTCR Register : Only the TFC and RFC fields are reset. UARTCTO Register : All fields are fully reset.

18.4.22. LIN UART preset timeout register (UARTPTO)

Address offset: 0x90
Reset value: 0x0000 0FFF

The UARTPTO register contains the preset value of the timeout register in UART mode and is programmed according to the number of bits to be received or to monitor the idle state of the reception line. The UARTPTO register can be written by software at any time.



Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	PTO[11:0]	Preset timeout PTO defines the preset value of timeout counter. A zero value is forbidden;

otherwise, the UARTSR[TO] status bit is immediately set.

18.4.23. LIN UART current timeout register (UARTCTO)

Address offset: 0x94
Reset value: 0x0000 0000

The UARTCTO register contains the current timeout value in UART mode. UARTCTO is used with the UARTPTO register to monitor the number of bits received by UART or to monitor the idle state of the reception line.

UART timeout works in both CPU and DMA modes.

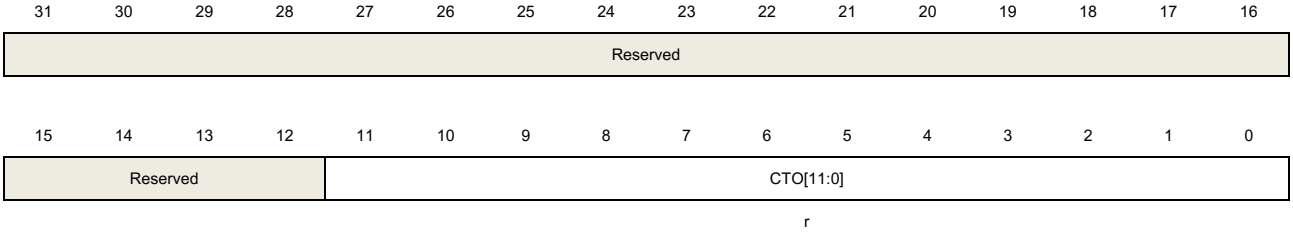
The timeout counter:

- Starts at zero and counts upward.
- Is clocked with CK_LINFlexD/(16*LDIV) synchronized to CK_LINFlexD (when UARTCR.ROSE=0).
- Is clocked with CK_LINFlexD/(UARTCR.OSR*LINIBRR.IBR) synchronized to CK_LINFlexD (when UARTCR.ROSE=1).
- Is automatically enabled when UARTCR.RXEN=1.

Note: Due to synchronization, there is a delay between 4 to 6 clock cycles of CK_LINFlexD for the internal value (which is clocked with CK_LINFlexD/16*LDIV) to reflect on UARTCTO.

The timeout counter is reset when one of the following conditions occurs:

- The number of frames received is equal to UARTCR.NEF.
- UARTCTO becomes equal to UARTPTO.
- UARTPTO is written.
- UARTSR. DRF is set and UARTCR.DTU=1.



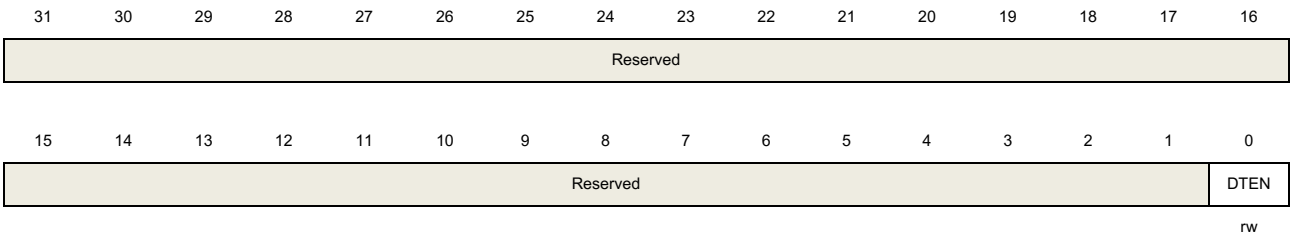
Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	CTO[11:0]	Timeout CTO defines the current value of the timeout counter. CTO is a read-only field. CTO is reset every time UARTPTO is written, or UARTCTO = UARTPTO, or by hard / soft reset. When the CTO value matches the preset value (UARTPTO.PTO), the status bit UARTSR.TO is set.

18.4.24. LIN DMA Tx enable register (DMATXE)

Address offset: 0x98
Reset value: 0x0000 0000

The DMATXE register enables the DMA Tx interface. The DMATXE register can be written by software at any time. When DMATXE=0x0000_0000,the DMA Tx interface FSM is forced (by soft reset) into the Idle state.

Note: When configuration parameter DMA_SUPPORTED=0, the LINFlexD does not support DMA operation. The DMATXE register is invalid and must be written only with value 0x0000_0000.



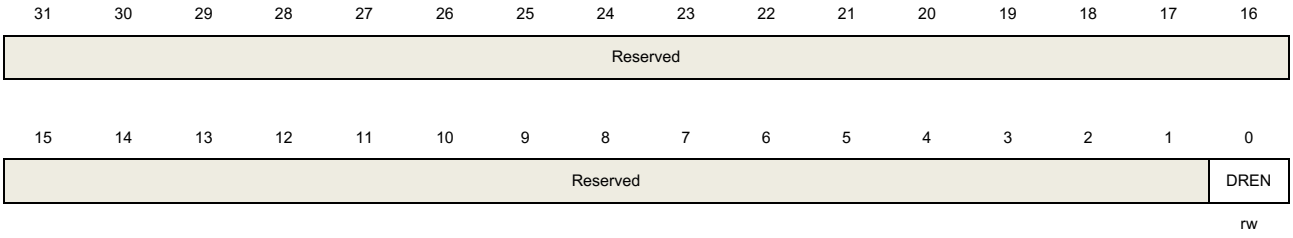
Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	DTEN	DMA Tx channel enable 0: DMA Tx channel is disabled. 1: DMA Tx channel is enabled.

18.4.25. LIN DMA Rx enable register (DMARXE)

Address offset: 0x9C
Reset value: 0x0000 0000

The DMARXE register enables the DMARx interface. The DMARXE register can be written by software at any time. When DMARXE=0x0000_0000, the DMARx interface FSM is forced (by soft reset) into the Idle state.

Note: When configuration parameter DMA_SUPPORTED=0, the LINFlexD does not support DMA operation. The DMARXE register is invalid and must be written only with value 0x0000_0000.

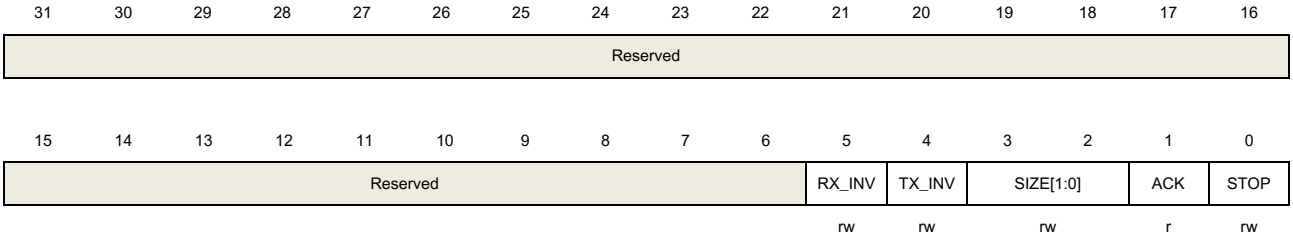


Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	DREN	DMA Rx channel enable 0: DMA Rx channel is disabled. 1: DMA Rx channel is enabled.

18.4.26. LIN stop control register (LIN_STCTL)

Address offset: 0xA0

Reset value: 0x0000 0000



Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5	RX_INV	RX pin level inversion: 0: RX pin signal value is not inverted. 1: RX pin signal value is inverted. **Note:** It is recommended to use only in UART mode, LIN mode is not supported.
4	TX_INV	TX pin level inversion: 0: TX pin signal value is not inverted. 1: TX pin signal value is inverted. **Note:** It is recommended to use only in UART mode, LIN mode is not supported.
3:2	SIZE[1:0]	The byte size transmitted on the IPS bus 00：Bits [31:0] are valid (default value) . 01：Bits [7:0] are valid, [31:8] are disabled. 10：Bits [15:0] are valid, [31:16] are disabled. By default, other configuration values are set with "wdata / rdata [31:0]" as a full word valid. When configuring via software, please ensure to set it to 01/10 in UART FIFO mode; otherwise, bus transmission errors may occur. **Note:** It is recommended to use only in UART mode, LIN mode is not supported.
1	ACK	The stop function takes effect.
0	STOP	Terminate the transmission; once pulled up, notify LINFlexD that the external clock will be disabled.

19. Inter-integrated circuit interface (I2C)

19.1. Overview

The I2C (inter-integrated circuit) module provides an I2C interface which is an industry standard two-line serial interface for MCU to communicate with external I2C interface. I2C bus uses two serial lines: a serial data line (SDA) and a serial clock line (SCL).

The I2C interface implements standard I2C protocol with standard mode, fast mode and fast mode plus as well as CRC calculation and checking, SMBus (system management bus), and PMBus (power management bus). It also supports multi-master I2C bus. The I2C interface provides DMA mode for users to reduce CPU overload.

19.2. Characteristics

- Parallel-bus to I2C-bus protocol converter and interface.
- Both master and slave functions with the same interface.
- Bi-directional data transfer between master and slave.
- Supports 7-bit and 10-bit addressing and general call addressing.
- Multiple 7-bit slave addresses (2 address, 1 with configurable mask).
- Programmable setup time and hold time.
- Multi-master capability.
- Supports standard mode (up to 100 kHz) and fast mode (up to 400 kHz) and fast mode plus (up to 1MHz, must enable Fm+ in SYSCFG_PMCFG register).
- Configurable SCL stretching in slave mode.
- Supports DMA mode.
- SMBus 3.0 and PMBus 1.3 compatible.
- Optional PEC (packet error checking) generation and check.
- Programmable analog and digital noise filters.

19.3. Function overview

Figure 19-1. I2C module block diagram below provides details on the internal configuration of the I2C interface.

Figure 19-1. I2C module block diagram

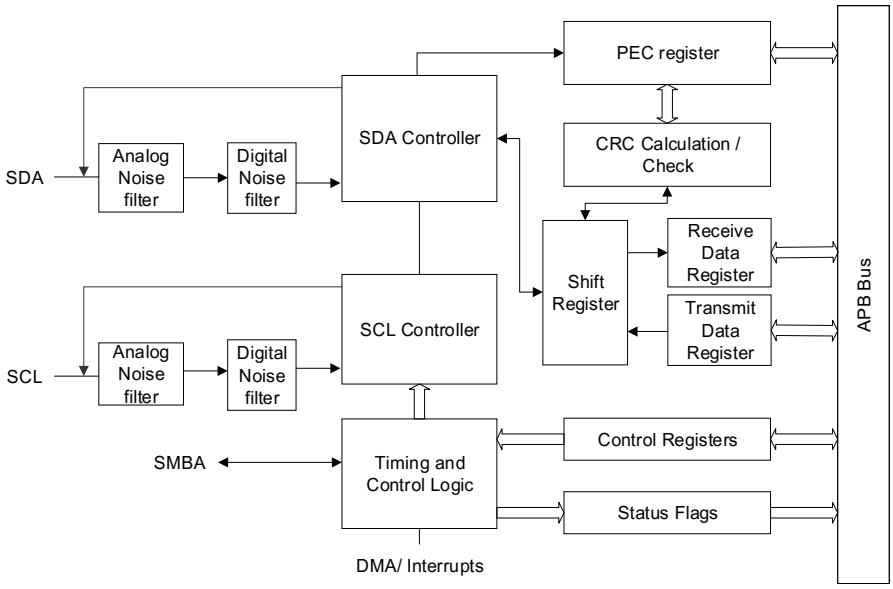


Table 19-1. Definition of I2C-bus terminology (refer to the I2C specification of Philips semiconductors)

Term	Description
Transmitter	the device which sends data to the bus
Receiver	the device which receives data from the bus
Master	the device which initiates a transfer, generates clock signals and terminates a transfer
Slave	the device addressed by a master
Multi-master	more than one master can attempt to control the bus at the same time without corrupting the message
Arbitration	procedure to ensure that, if more than one master tries to control the bus simultaneously, only one is allowed to do so and the winning master's message is not corrupted

19.3.1. Clock requirements

The I2CCLK period t_{I2CCLK} must match the conditions as follows:

- $t_{I2CCLK} < (t_{LOW} - t_{filters})/4$
- $t_{I2CCLK} < t_{HIGH}$

with:

t_{LOW}: SCL low time

t_{HIGH}: SCL high time

$t_{filters}$: When the filters are enabled, represent the delays by the analog filter and digital filter.

Analog filter delay is maximum 260ns. Digital filter delay is DNF[3:0]×t_{I2CCLK}.

The period of PCLK clock t_{PCLK} match the conditions as follows:

- $t_{PCLK}<4/3*t_{SCL}$

with:

t_{SCL}: the period of SCL

Note: When the I2C kernel is provided by PCLK, this clock must match the conditions for t_{I2CCLK}.

19.3.2. I2C communication flow

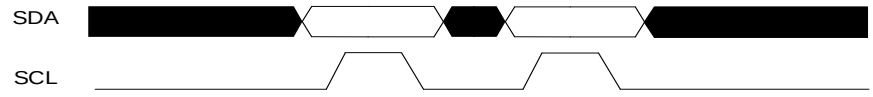
An I2C device is able to transmit or receive data whether it's a master or a slave, thus, there're 4 operation modes for an I2C device:

- Slave transmitter
- Slave receiver
- Master transmitter
- Master receiver

Data validation

The data on the SDA line must be stable during the HIGH period of the clock. The HIGH or LOW state of the data line can only change when the clock signal on the SCL line is LOW (see *Figure 19-2. Data validation*). One clock pulse is generated for each data bit transferred.

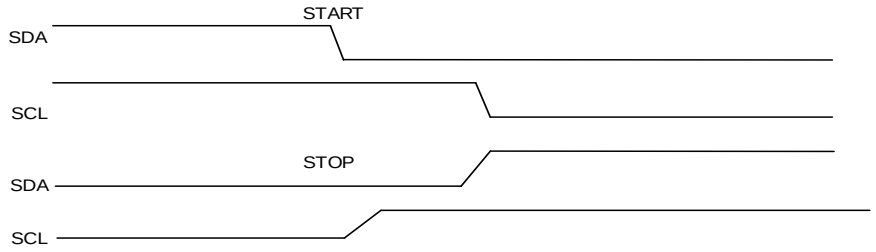
Figure 19-2. Data validation



START and STOP signal

All transactions begin with a START and are terminated by a STOP (see *Figure 19-3. START and STOP signal*). A HIGH to LOW transition on the SDA line while SCL is HIGH defines a START signal. A LOW to HIGH transition on the SDA line while SCL is HIGH defines a STOP signal.

Figure 19-3. START and STOP signal



Each I2C device is recognized by a unique address (whether it is a microcontroller, LCD driver, memory or keyboard interface) and can operate as either a transmitter or receiver, depending on the function of the device. It operates in slave mode by default. When it generates a START signal, the interface automatically switches from slave to master. If an arbitration loss or a STOP generation occurs, then the interface switches from master to slave, allowing multimaster capability.

An I2C slave will continue to detect addresses after a START signal on I2C bus and compare the detected address with its slave address which is programmable by software. Once the two addresses match, the I2C slave will send an ACK to the I2C bus and responses to the following command on I2C bus: transmitting or receiving the desired data. Additionally, if General Call is enabled by software, the I2C slave always responses to a General Call Address (0x00). The I2C block support both 7-bit and 10-bit address modes.

Data and addresses are transferred as 8-bit bytes, MSB first. The first byte(s) following the START signal contain the address (one in 7-bit mode, two in 10-bit mode). The address is always transmitted in master mode.

A 9th clock pulse follows the 8 clock cycles of byte transmission, during which the receiver must send an acknowledge bit to the transmitter. Acknowledge can be enabled or disabled by software.

An I2C master always initiates or end a transfer using START or STOP signal and it's also responsible for SCL clock generation.

In master mode, if AUTOEND=1, the STOP signal is generated automatically by hardware. If AUTOEND=0, the STOP signal generated by software, or the master can generate a RESTART signal to start a new transfer.

Figure 19-4. I2C communication flow with 10-bit address (Master Transmit)

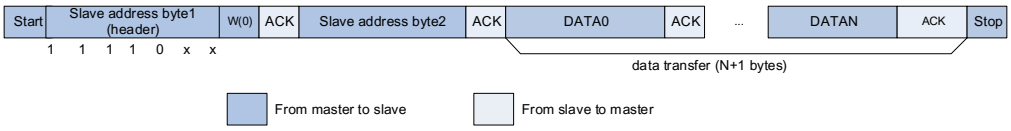


Figure 19-5. I2C communication flow with 7-bit address (Master Transmit)

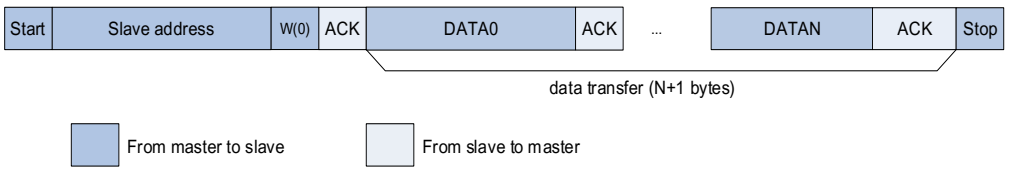
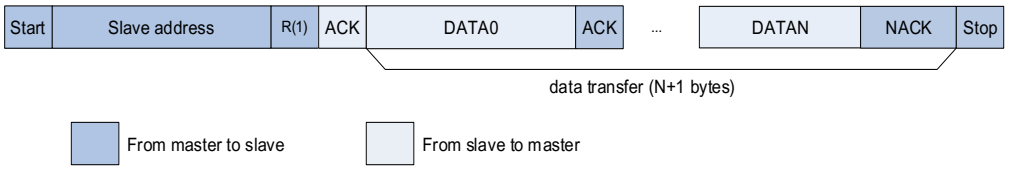


Figure 19-6. I2C communication flow with 7-bit address (Master Receive)



In 10-bit addressing mode, the HEAD10R bit can configured to decide whether the complete address sequence must be executed, or only the header to be sent. When HEAD10R=0, the complete 10 bit address read sequence must be executed with START + header of 10-bit address in write direction + slave address byte 2 + RESTART + header of 10-bit address in read direction, as is shown in ***Figure 19-7. I2C communication flow with 10-bit address (Master Receive when HEAD10R=0)***.

In 10-bit addressing mode, if the master reception follows a master transmission between the same master and slave, the address read sequence can be RESTART + header of 10-bit address in read direction, as is shown in ***Figure 19-8. I2C communication flow with 10-bit address (Master Receive when HEAD10R=1)***.

Figure 19-7. I2C communication flow with 10-bit address (Master Receive when HEAD10R=0)

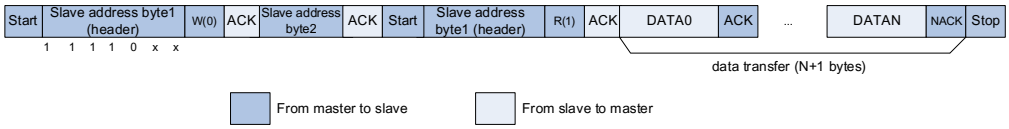
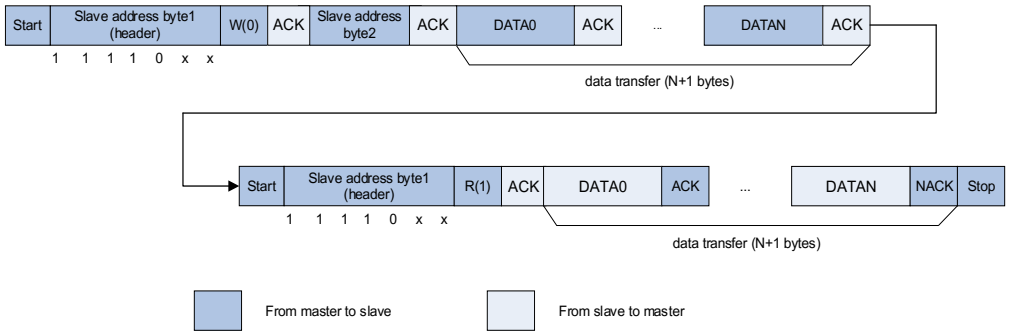


Figure 19-8. I2C communication flow with 10-bit address (Master Receive when HEAD10R=1)



Note：When HEAD10R=1 and it is a RESTART scenario, only Master mode in scenario from ***Figure 19-8. I2C communication flow with 10-bit address (Master Receive when HEAD10R=1)*** is supported.

19.3.3. Noise filter

Analog noise filter and digital noise filter are integrated in I2C peripherals, the noise filters can be configured before the I2C peripheral is enabled according to the actual requirements.

The analog noise filter is disabled by setting the ANOFF bit in I2C_CTL0 register and enabled when ANOFF is 0. It can suppress spikes with a pulse width up to 50ns in fast mode and fast mode plus.

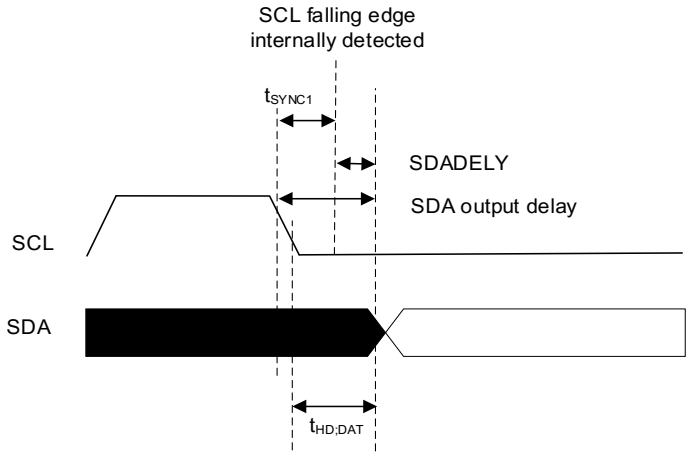
The digital noise filter can be used by configuring the DNF [3:0] bit in I2C_CTL0 register. The level of the SCL or the SDA will be changed if the level is stable for more than DNF[3:0]×t_{I2CCLK}. The length of spikes to be suppressed is configured by DNF [3:0].

19.3.4. I2C timings configuration

The PSC [3:0], SCLDELY [3:0] and SDADELY [3:0] bits in the I2C_TIMING register must be configured in order to guarantee a correct data hold and setup time used in I2C communication.

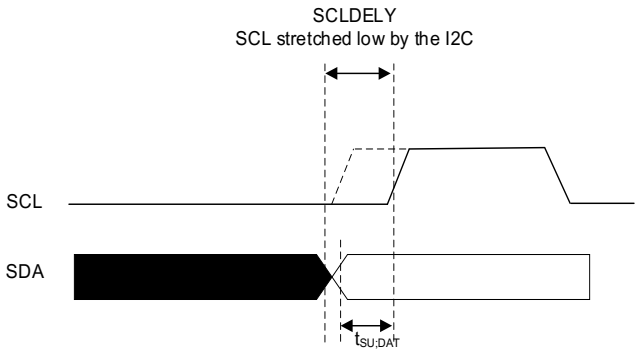
If the data is already available in I2C_TDATA register, the data will be sent on SDA after the SDADELY delay. As is shown in ***Figure 19-9. Data hold time***.

Figure 19-9. Data hold time



The SCLDELY counter starts when the data is sent on SDA output. As is shown in ***Figure 19-10. Data setup time***.

Figure 19-10. Data setup time



When the SCL falling edge is internally detected, a delay is inserted before sending SDA output. This delay is $t_{SDADELY}$ = SDADELY * t_{PSC} + t_{I2CCLK} where t_{PSC} = (PSC+1) * t_{I2CCLK}. $t_{SDADELY}$ effects $t_{HD;DAT}$. The total delay of SDA output is t_{SYNC1} + {[SDADELY * (PSC+1) + 1] * t_{I2CCLK}}. t_{SYNC1} depends on SCL falling slope, the delay of analog filter, the delay of digital filter and delay of SCL synchronization to I2CCLK clock. The delay of SCL synchronization to I2CCLK clock is 2 to 3 t_{I2CCLK}.

SDADELY must match condition as follows:

- SDADELY≥{t_f(max)+$t_{HD;DAT}$(min)-t_{AF}(min)-[(DNF+3)*t_{I2CCLK}]}/[(PSC+1)*t_{I2CCLK}]

■ SDADELY≤{$t_{HD;DAT}$(max)-t_{AF}(max)-[(DNF+4)*t_{I2CCLK}]}/[(PSC+1)*t_{I2CCLK}]

Note: t_{AF} is the delay of analog filter. The $t_{HD;DAT}$ should be less than the maximum of $t_{VD;DAT}$.

When SS = 0, after $t_{SDADELY}$ delay, the slave had to stretch the clock before the data writing to I2C_TDATA register, SCL is low during the data setup time. The setup time is $t_{SCLDELY}$=(SCLDELY+1)*t_{PSC}. $t_{SCLDELY}$ effects $t_{SU;DAT}$.

SCLDELY must match condition as follows:

■ SCLDELY≥{[t_r(max)+$t_{SU;DAT}$(min)]/[(PSC+1)*t_{I2CCLK}]}-1

In master mode, the SCL clock high and low levels must be configured by programming the PSC [3:0], SCLH [7:0] and SCLL[7:0] bits in the I2C_TIMING register.

When the SCL falling edge is internally detected, a delay is inserted before releasing the SCL output. This delay is t_{SCLL}=(SCLL+1)*t_{PSC} where t_{PSC}=(PSC+1)*t_{I2CCLK}. t_{SCLL} impacts the SCL low time t_{LOW}.

When the SCL rising edge is internally detected, a delay is inserted before forcing the SCL output to low level. This delay is t_{SCLH}=(SCLH+1)*t_{PSC} where t_{PSC}=(PSC+1)*t_{I2CCLK}. t_{SCLH} impacts the SCL high time t_{HIGH}.

Note: When the I2C is enabled, the timing configuration and SS mode must not be changed.

Table 19-2. Data setup time and data hold time

Symbol	Parameter	Standard mode		Fast mode		Fast mode plus		SMBus		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
$t_{HD;DAT}$	Data hold time	0	-	0	-	0	-	0.3	-	us
$t_{VD;DAT}$	Data valid time	-	3.45	-	0.9	-	0.45	-	-	
$t_{SU;DAT}$	Data setup time	250	-	100	-	50	-	250	-	ns
t_r	Rising time of SCL and SDA	-	1000	-	300	-	120	-	1000	
t_f	falling time of SCL and SDA	-	300	-	300	-	120	-	300	

19.3.5. I2C reset

A software reset can be performed by clearing the I2CEN bit in the I2C_CTL0 register. When a software reset is generated, the SCL and SDA are released. The communication control bits and status bits come back to the reset value. Software reset have no effect on configuration registers. The impacted register bits are START, STOP, NACKEN in I2C_CTL1 register, I2CBSY, TBE, TI, RBNE, ADDSEND, NACK, TCR, TC, STPDET, BERR, LOSTARB and OUERR in I2C_STAT register. Additionally, when the SMBus is supported, PECTRANS in I2C_CTL1 register, PECERR, TIMEOUT and SMBALT in I2C_STAT are also impacted.

In order to perform the software reset, I2CEN must be kept low during at least 3 APB clock

cycles. This is ensured by writing software sequence as follows:

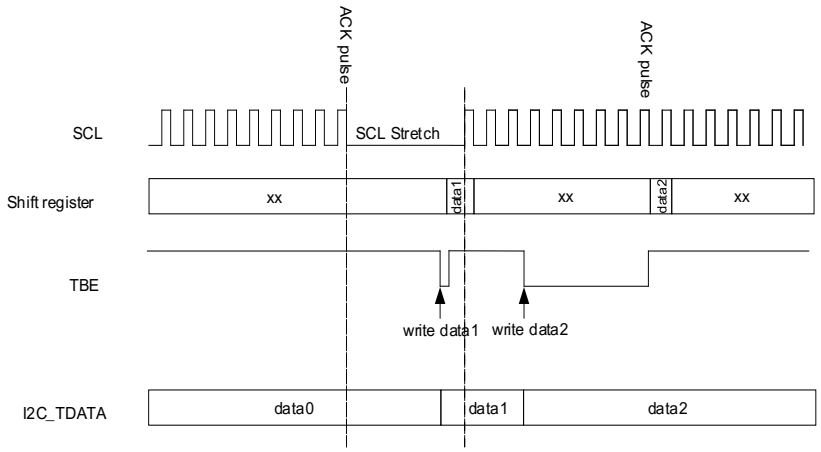
- Write I2CEN = 0
- Check I2CEN = 0
- Write I2CEN = 1

19.3.6. Data transfer

Data Transmission

When transmitting data, if TBE is 0, it indicates that the I2C_TDATA register is not empty, the data in I2C_TDATA register is moved to the shift register after the 9th SCL pulse. Then the data will be transmitted through the SDA line from the shift register. If TBE is 1, it indicates that the I2C_TDATA register is empty, the SCL line is stretched low until I2C_TDATA is not empty. The stretch begins after the 9th SCL pulse.

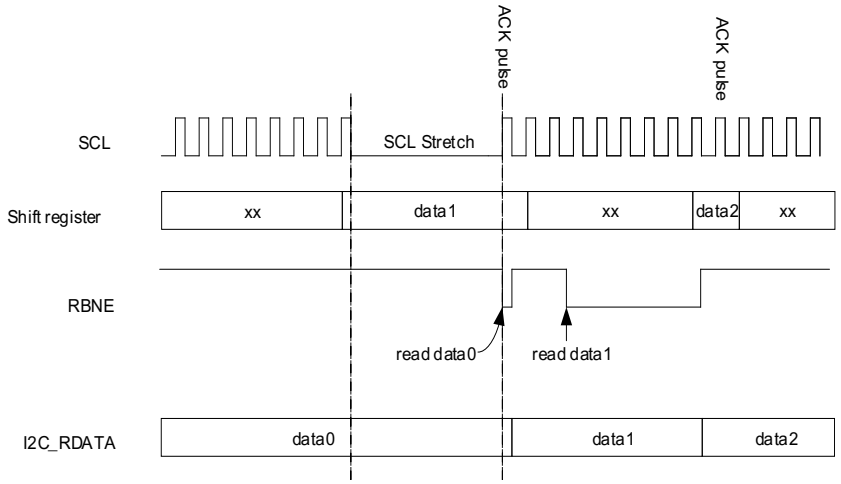
Figure 19-11. Data transmission



Data Reception

When receiving data, the data will be received in the shift register first. If RBNE is 0, the data in the shift register will move into I2C_RDATA register. If RBNE is 1, the SCL line will be stretched until the previous received data in I2C_RDATA is read. The stretch is inserted before the acknowledge pulse.

Figure 19-12. Data reception



Reload and automatic end mode

In order to manage byte transfer and to shut down the communication in modes as is shown in ***Table 19-3. Communication modes to be shut down***, the I2C embedded a byte counter in the hardware.

Table 19-3. Communication modes to be shut down

Working mode	Action
Master mode	NACK, STOP and RESTART generation
Slave receiver mode	ACK control
SMBus mode	PEC generation/checking

The number of bytes to be transferred is configured by BYTENUM [7:0] in I2C_CTL1 register. If BYTENUM is greater than 255, or in slave byte control mode, the reload mode must be enabled by setting the RELOAD bit in I2C_CTL1 register. In reload mode, when BYTENUM counts to 0, the TCR bit will be set, and an interrupt will be generated if TCIE is set. Once the TCR flag is set, SCL is stretched. The TCR bit is cleared by writing a non-zero number in BYTENUM.

Note: The reload mode must be disabled after the last reloading of BYTENUM [7:0].

The reload mode must be disabled when the automatic end mode is enabled. In automatic end mode, the master will send a STOP signal automatically when the BYTENUM [7:0] counts to 0.

When reload mode and automatic end mode are disabled, the I2C communication process needs to be terminated by software. If the number of bytes BYTENUM [7:0] has been transferred, the STOP bit should be set by software to generate a STOP signal, and then TC flag must be cleared.

19.3.7. I2C slave mode

Initialization

When works in slave mode, at least one slave address should be enabled. Slave address 1 can be programmed in I2C_SADDR0 register and slave address 2 can be programmed in I2C_SADDR1 register. ADDRESSEN in I2C_SADDR0 register and ADDRESS2EN in I2C_SADDR1 register should be set when the corresponding address is used. 7-bit address or 10-bit address can be programmed in ADDRESS [9:0] in I2C_SADDR0 register by configuring the ADDFORMAT bit in 7-bit address or 10-bit address.

The ADDM[6:0] in I2C_CTL2 register defines which bits of ADDRESS [7:1] are compared with an incoming address byte, and which bits are ignored.

The ADDMSK2[2:0] is used to mask ADDRESS2[7:1] in I2C_SADDR1 register. For details, refer to the description of ADDMSK2[2:0] in I2C_SADDR1 register.

When the I2C received address matches one of its enabled addresses, the ADDSEND will be set, and an interrupt is generated if the ADDMIE bit is set. The READDR[6:0] bits in I2C_STAT register will store the received address. And TR bit in I2C_STAT register updates after the ADDSEND is set. The bit will let the slave to know whether to act as a transmitter or receiver.

SCL line stretching

The clock stretching is used in slave mode by default (SS=0), the SCL line can be stretched low if necessary. The SCL will be stretched in following cases.

- The SCL is stretched when the ADDSEND bit is set, and released when the ADDSEND bit is cleared.
- In slave transmitting mode, after the ADDSEND bit is cleared, the SCL will be stretched before the first data byte writing to the I2C_TDATA register. Or the SCL will be stretched before the new data is written to the I2C_TDATA register after the previous data transmission is completed.
- In slave receiving mode, a new reception is completed but the data in I2C_RDATA register has not been read.
- When SBCTL=1 and RELOAD=1, after the transfer of the last byte, TCR is set. Before the TCR is cleared, the SCL will be stretched.
- The I2C stretches SCL low during $[(SDADELY+SCLDELY+1)*(PSC+1)+1]*t_{I2CCLK}$ after detecting the SCL falling edge.

The clock stretching can be disabled by setting the SS bit in I2C_CTL0 register (SS=1). The SCL will not be stretched in following cases.

- When the ADDSEND is set, the SCL will be not stretched.
- In slave transmitting mode, before the first SCL pulse, the data must be written in the I2C_TDATA register. Or else the OUERR bit in the I2C_STAT register will be set, if the

ERRIE bit is set, an interrupt will be generated. When the STPDET bit is set and the first data transmission starts, OUERR bit in the I2C_STAT register will also be set.

- In slave receiving mode, before the 9th SCL pulse (ACK pulse) occurred by the next data byte, the data must be read out from the I2C_RDATA register. Or else the OUERR bit in the I2C_STAT register will be set, if the ERRIE bit is set, and an interrupt will be generated.

Slave byte control mode

In slave receiving mode, the slave byte control mode can be enabled by setting the SBCTL bit in the I2C_CTL0 register to allow byte ACK control. When SS=1, the slave byte control mode is not allowed.

When using slave byte control mode, the reload mode must be enabled by setting the RELOAD bit in I2C_CTL1 register. In slave byte control mode, BYTENUM [7:0] in I2C_CTL1 register must be configured as 1 in the ADDSEND interrupt service routine and reloaded to 1 after each byte received. The TCR bit in I2C_STAT register will be set when a byte is received, the SCL will be stretched low by slave between the 8th and 9th clock pulses. Then the data can be read from the I2C_RDATA register, and the slave determines to send an ACK or a NACK by configuring the NACKEN bit in the I2C_CTL1 register. When the BYTENUM [7:0] is written a non-zero value, the slave will release the stretch.

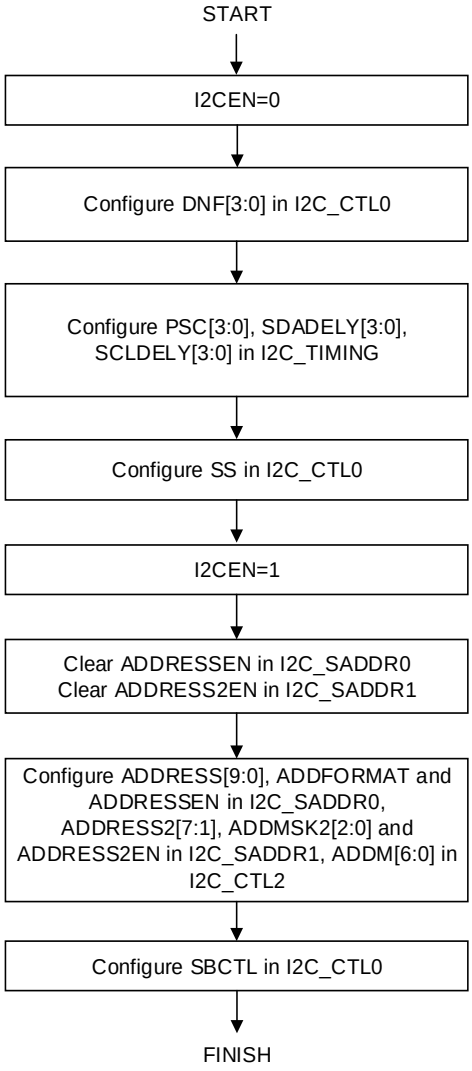
When the BYTENUM [7:0] is greater than 0x1, there is no stretch between the reception of two data bytes.

Note: The SBCTL bit can be configured in following cases:

- I2CEN=0.
- The slave has not been addressed.
- ADDSEND=1.

Only when the ADDSEND=1, or TCR=1, the RELOAD bit can be modified.

Figure 19-13. I2C initialization in slave mode



Programming model in slave transmitting mode

When the I2C_TDATA register is empty, the TI bit in I2C_STAT register will be set. If the TIE bit in I2C_CTL0 register is set, an interrupt will be generated. The NACK bit in I2C_STAT register will be set when a NACK is received. And an interrupt is generated if the NACKIE bit is set in the I2C_CTL0 register. The TI bit in I2C_STAT register will not be set when a NACK is received.

The STPDET bit in I2C_STAT register will be set when a STOP is received. If the STPDETIE in I2C_CTL0 register is set, an interrupt will be generated.

When SBCTL is 0, if ADDSEND=1, and the TBE bit in I2C_STAT register is 0, the data in I2C_TDATA register can be chosen to be transmitted or flushed. The data is flushed by setting the TBE bit.

When SBCTL=1, the slave works in slave byte control mode, the BYTENUM [7:0] must be configured in the ADDSEND interrupt service routine. And the number of TI events is equal

to the value of BYTENUM [7:0].

When SS=1, the SCL will not be stretched when ADDSEND bit in I2C_STAT register is set. In this case, the data in I2C_TDATA register cannot be flushed in ADDSEND interrupt service routine. So the first byte to be sent must be programmed in the I2C_TDATA register previously.

- This data can be the one which is written in the last TI event of the last transfer.
- Setting the TBE bit can flush the data if it is not the one to be sent, then a new byte can be written in I2C_TDATA register. The STPDET must be 0 before the first data transmission, or else the OUERR bit in I2C_STAT register will be set and an underrun error occurs.
- When interrupt or DMA is used in slave transmitter, if a TI event is needed, in order to generate a TI event both the TI bit and the TBE bit must be set.

Figure 19-14. Programming model for slave transmitting when SS=0

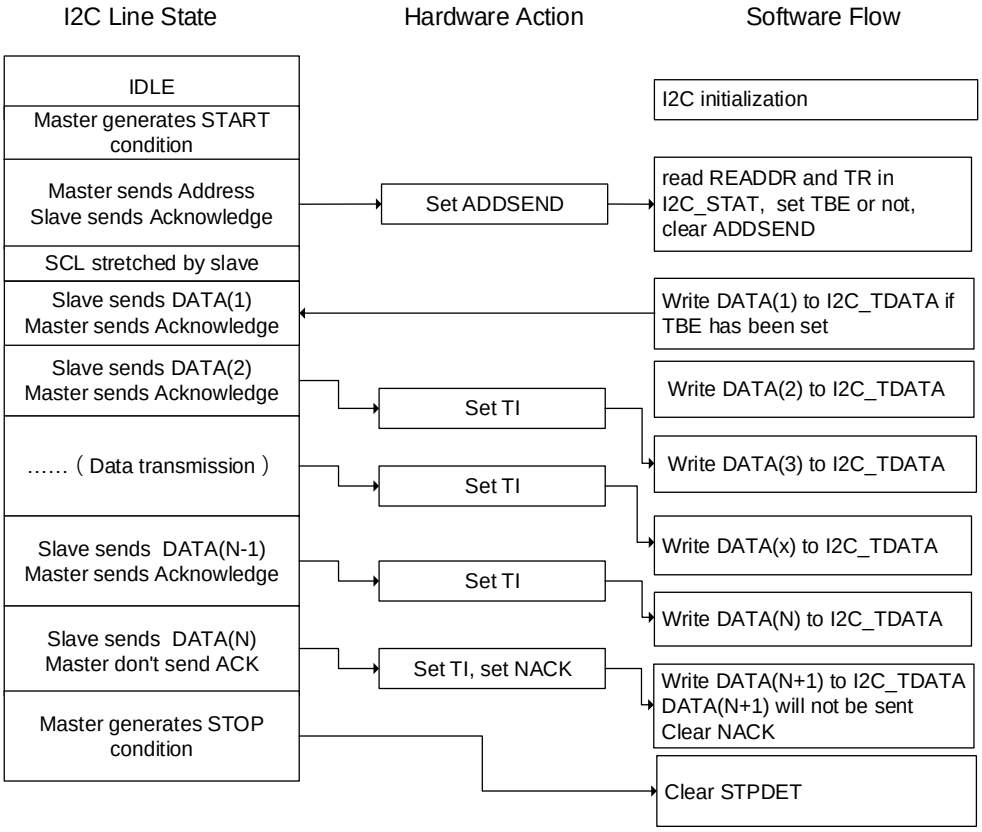
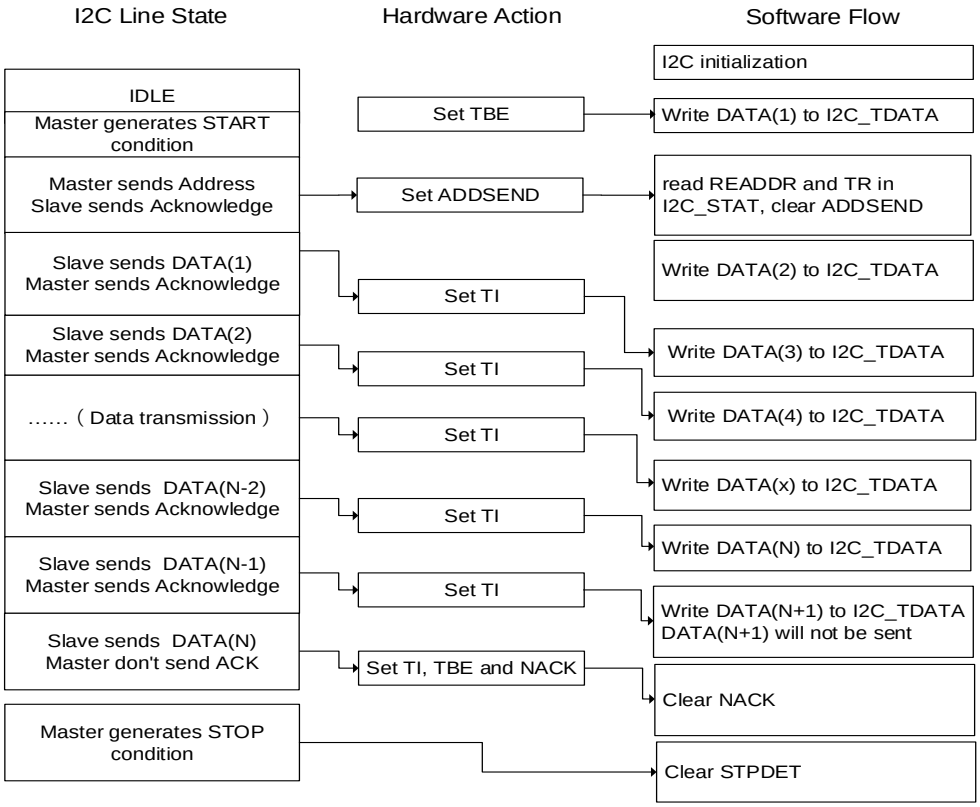


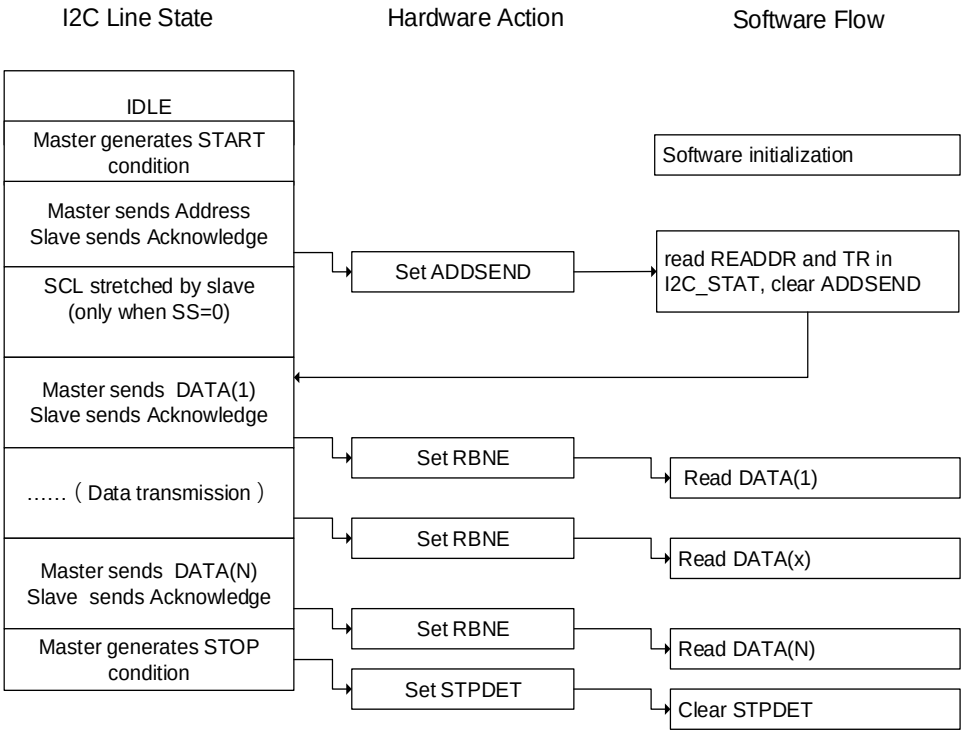
Figure 19-15. Programming model for slave transmitting when SS=1



Programming model in slave receiving mode

When the I2C_RDATA is not empty, the RBNE bit in I2C_STAT register is set, and if the RBNEIE bit in I2C_CTL0 register is set, an interrupt will be generated. When a STOP is received, STPDET will be set in I2C_STAT register. If the STPDETIE bit in I2C_CTL0 register is set, and an interrupt will be generated.

Figure 19-16. Programming model for slave receiving



19.3.8. I2C master mode

Initialization

The SCLH [7:0] and SCLL [7:0] in I2C_TIMING register should be configured when I2CEN is 0. In order to support multi-master communication and slave clock stretching, a clock synchronization mechanism is implemented.

The SCLL [7:0] and SCLH [7:0] are used for the low level counting and high level counting respectively. After a t_{SYNC1} delay, when the SCL low level is detected, the SCLL[7:0] starts counting, if the SCLL[7:0] in I2C_TIMING register is reached by SCLL[7:0] counter, the I2C will release the SCL clock. After a t_{SYNC2} delay, when the SCL high level is detected, the SCLH[7:0] starts counting, if the SCLH[7:0] in I2C_TIMING register is reached by SCLH[7:0] counter, the I2C will stretch the SCL clock.

So the master clock period is:

$$t_{SCL}=t_{SYNC1}+t_{SYNC2}+\{[(SCLH[7:0]+1)+(SCLL[7:0]+1)]*(PSC+1)*t_{I2CCLK}\}.$$

The t_{SYNC1} depends on the SCL falling slope, delay by input analog and digital noise filter and SCL synchronization with I2CCLK clock, which generally 2 to 3 I2CCLK periods. The t_{SYNC2} depends on the SCL rising slope, delay by input analog and digital noise filter and SCL synchronization with I2CCLK clock, which generally 2 to 3 I2CCLK periods. The delay by digital noise filter is DNF[3:0]*t_{I2CCLK}.

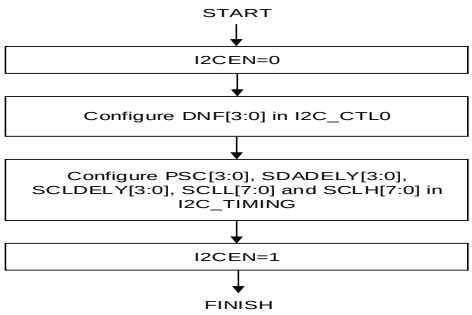
When works in master mode, the ADD10EN bit, SADDRESS [9:0] bits, TRDIR bit should be configured in I2C_CTL1 register. When the addressing mode is 10-bit in master receiving

mode, the HEAD10R bit must be configured to decide whether the complete address sequence must be executed, or only the header to be sent. The number of bytes to be transferred should be configured in BYTENUM [7:0] in I2C_CTL1 register. If the number of bytes to be transferred is equal to or greater than 255, BYTENUM [7:0] should be configured as 0xFF. Then the master sends the START signal. All the bits above should be configured before the START is set. The slave address will be sent after the START signal when the I2CBSY bit I2C_STAT register is detected as 0. When the arbitration is lost, the master changes to slave mode and the START bit will be cleared by hardware. When the slave address has been sent, the START bit will be cleared by hardware.

In 10-bit addressing mode, if the master receives a NACK after the transmission of 10-bit header, the master will resend it until ACK is received. The ADDSENDC bit must be set to stop sending the slave address.

If the START bit is set, meanwhile the ADDSEND is set by addressing as a slave, the master changes to slave mode. The ADDSENDC bit must be set to clear the START bit.

Figure 19-17. I2C initialization in master mode



Programming model in master transmitting mode

In master transmitting mode, the TI bit is set after the ACK is received of each byte transmission. If the TIE bit in I2C_CTL0 register is set, an interrupt will be generated. The bytes to be transferred is programmed in BYTENUM [7:0] in I2C_CTL1 register. If the bytes to be transferred is greater than 255, RELOAD bit in I2C_CTL1 register must be set to enable the reload mode. In reload mode, when data of BYTENUM [7:0] bytes have been transferred, the TCR bit in I2C_STAT register will be set and the SCL stretches until BYTENUM [7:0] is modified with a non-zero value.

When a NACK is received, the TI bit will not set.

- If data of BYTENUM [7:0] bytes have been transferred and RELOAD=0, the AUTOEND

bit in I2C_CTL1 can be set to generate a STOP signal automatically. When AUTOEND is 0, the TC bit in I2C_STAT register will be set and the SCL is stretched. In this case, the master can generate a STOP signal by setting the STOP bit in the I2C_CTL1 register. Or generate a RESTART signal to start a new transfer. The TC bit is cleared when the START / STOP bit is set.

- If a NACK is received, a STOP signal is automatically generated, the NACK is set in I2C_STAT register, if the NACKIE bit is set, an interrupt will be generated.

Note:

(1) When the RELOAD bit is 1, the AUTOEND has no effect.

(2) When the STPDET bit is set and the first data transmission starts, OUERR bit in the I2C_STAT register will also be set.

(3) When both NACK and TC occur simultaneously, the master no longer supports the automatic generation of STOP signals and does not support RESTART. If a STOP signal needs to be generate, the STOP bit should be manually configured.

Figure 19-18. Programming model for master transmitting (N<=255)

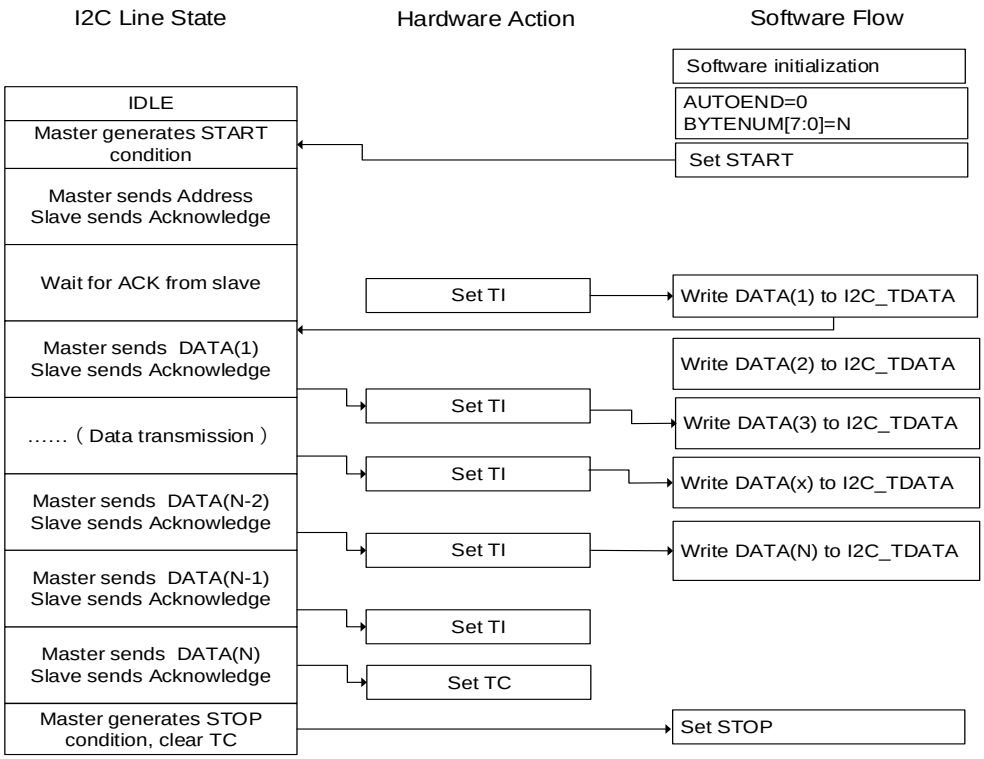
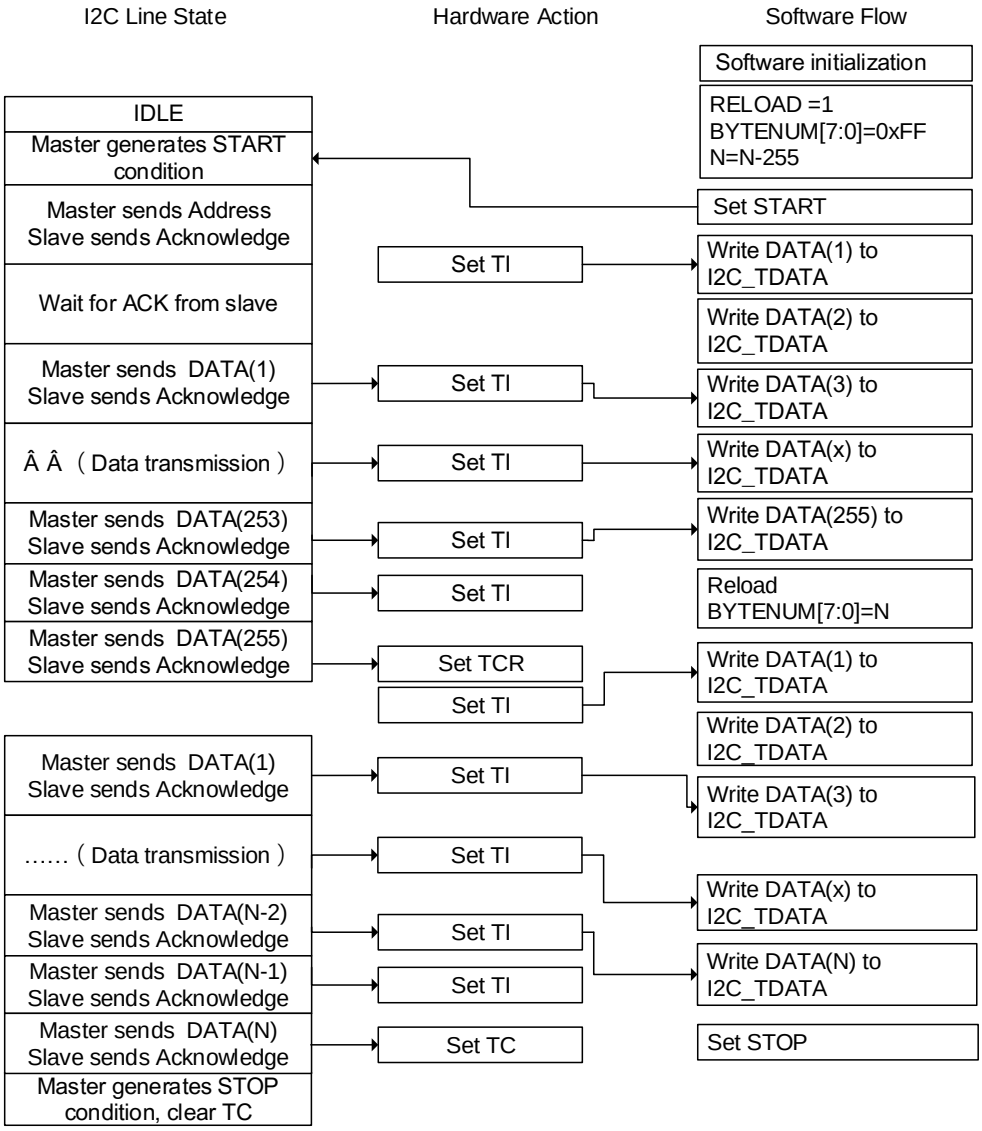


Figure 19-19. Programming model for master transmitting (N>255)



Programming model in master receiving mode

In master receiving mode, the RBNE bit in I2C_STAT register will be set when a byte is received. If the RBNEIE bit is set in I2C_CTL0 register, an interrupt will be generated. If the number of bytes to be received is greater than 255, RELOAD bit in I2C_CTL1 register must be set to enable the reload mode. In reload mode, when data of BYTENUM [7:0] bytes have been transferred, the TCR bit in I2C_STAT register will be set and the SCL stretches until BYTENUM [7:0] is modified with a non-zero value.

If data of BYTENUM [7:0] bytes have been transferred and RELOAD=0, the AUTOEND bit in I2C_CTL1 can be set to generate a STOP signal automatically. When AUTOEND is 0, the TC bit in I2C_STAT register will be set and the SCL is stretched. In this case, the master can generate a STOP signal by setting the STOP bit in the I2C_CTL1 register. Or generate a RESTART signal to start a new transfer. The TC bit is cleared when the START/STOP bit is set.

Figure 19-20. Programming model for master receiving (N<=255)

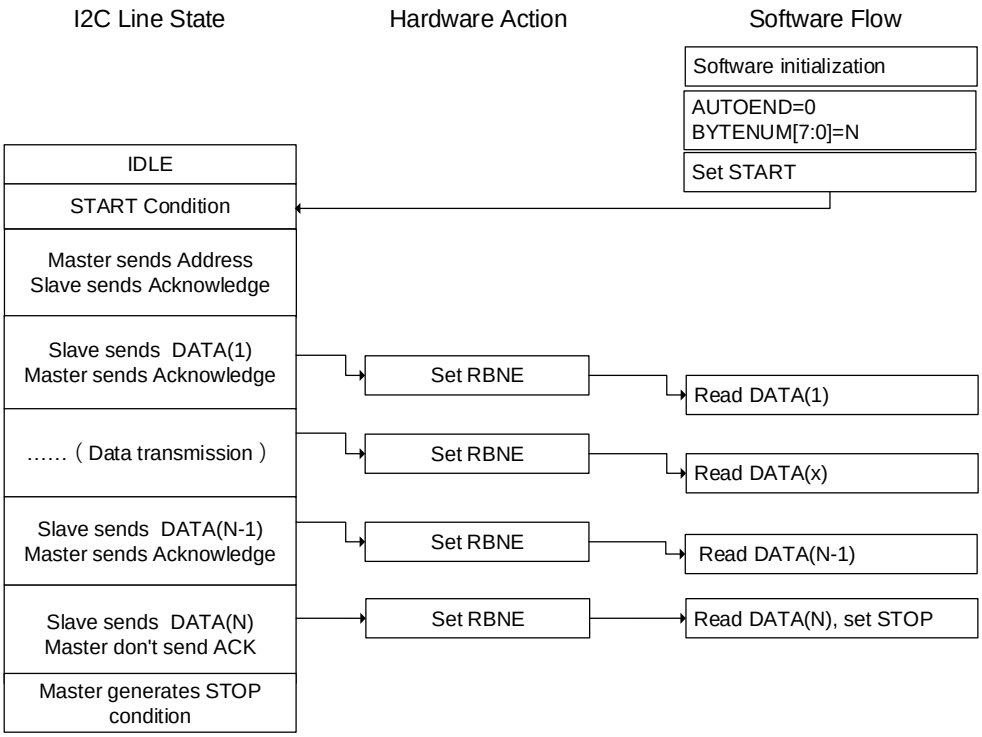
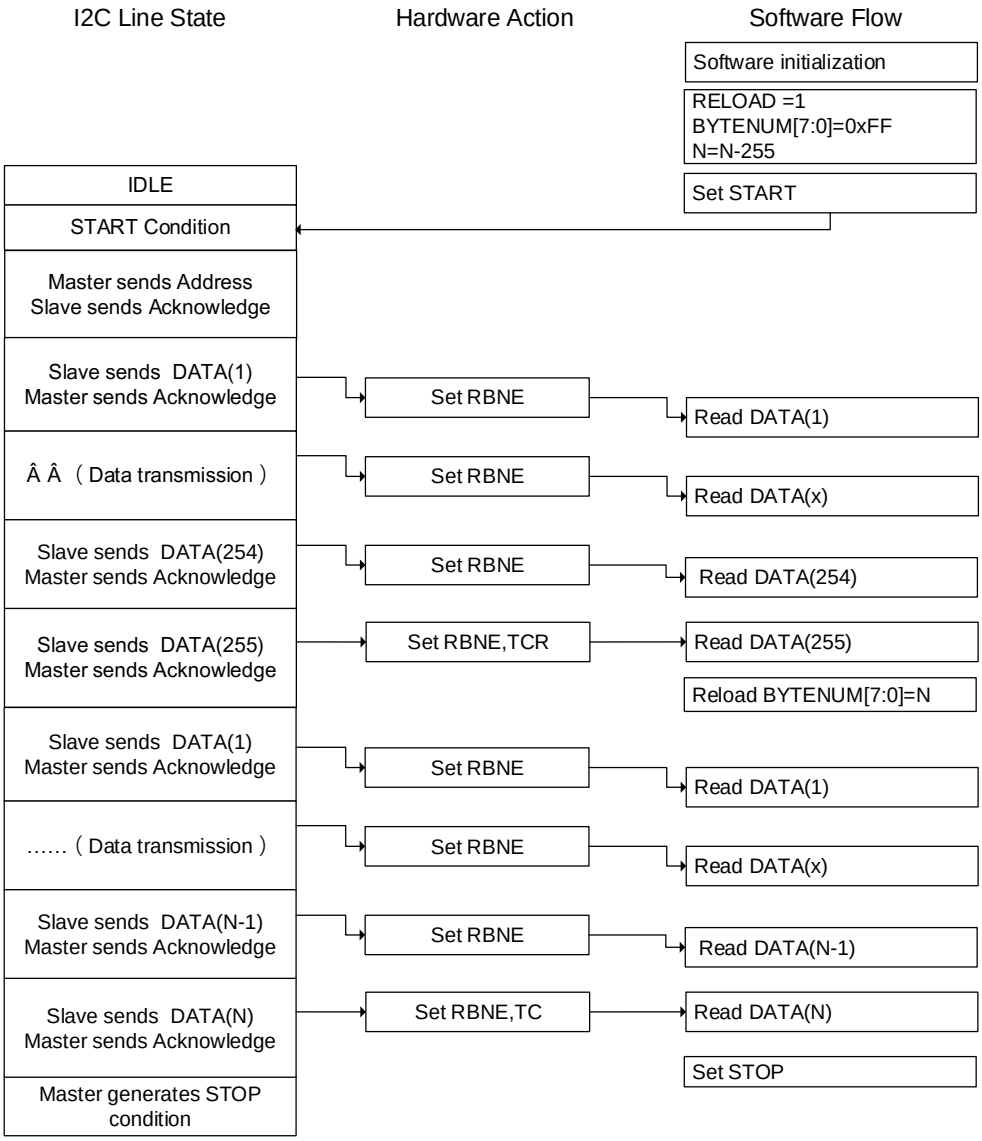


Figure 19-21. Programming model for master receiving (N>255)



19.3.9. SMBus support

The System Management Bus (abbreviated to SMBus or SMB) is a single-ended simple two-wire bus for the purpose of lightweight communication. Most commonly it is found in computer motherboards for communication with power source for ON/OFF instructions. It is derived from I2C for communication with low-bandwidth devices on a motherboard, especially power related chips such as a laptop's rechargeable battery subsystem (see Smart Battery Data).

SMBus protocol

Each message transaction on SMBus follows the format of one of the defined SMBus protocols. The SMBus protocols are a subset of the data transfer formats defined in the I2C specifications. I2C devices that can be accessed through one of the SMBus protocols are compatible with the SMBus specifications. I2C devices that do not adhere to these protocols cannot be accessed by standard methods as defined in the SMBus and Advanced

Configuration and Power Management Interface (abbreviated to ACPI) specifications.

Address resolution protocol

The SMBus uses I2C hardware and I2C hardware addressing, but adds second-level software for building special systems. Additionally, its specifications include an Address Resolution Protocol that can make dynamic address allocations. Dynamic reconfiguration of the hardware and software allow bus devices to be 'hot-plugged' and used immediately, without restarting the system. The devices are recognized automatically and assigned unique addresses. This advantage results in a plug-and-play user interface. In this protocol there is a very useful distinction between a system host and all the other devices in the system, that is the host provides address assignment function.

SMBus slave byte control

The slave byte control of SMBus receiver is the same as I2C. It allows the ACK control of each byte. The Slave Byte Control mode must be enabled by setting SBCTL bit in I2C_CTL0 register.

Host Notify protocol

When the SMBHAEN bit in the I2C_CTL0 register is set, the SMBus supports the host notify protocol. In this protocol, the device acts as a master and the host as a slave, and the host will acknowledge the SMBus host address.

Time-out feature

SMBus has a time-out feature which resets devices if a communication takes too long. This explains the minimum clock frequency of 10 kHz to prevent locking up the bus. I2C can be a 'DC' bus, meaning that a slave device stretches the master clock when performing some routine while the master is accessing it. This will notify to the master that the slave is busy but does not want to lose the communication. The slave device will allow continuation after its task is completed. There is no limit in the I2C bus protocol as to how long this delay can be, whereas for a SMBus system, it would be limited to 25~35ms. SMBus protocol just assumes that if something takes too long, then it means that there is a problem on the bus and that all devices must reset in order to clear this mode. Slave devices are not allowed to hold the clock low too long.

The timeout detection can be enabled by setting TOEN and EXTOEN bits in the I2C_TIMEOUT register. The timer must be configured to guarantee that the timeout detected before the maximum time given in the SMBus specification.

The value programmed in BUSTOA[11:0] is used to check the $t_{TIMEOUT}$ parameter. To detect the SCL low level timeout, the TOIDLE bit must be 0. And the timer can be enabled by setting the TOEN bit in the I2C_TIMEOUT register, after the TOEN bit is set, the BUSTOA [11:0] and the TOIDLE bit cannot be changed. If the low level time of SCL is greater than (BUSTOA+1)*2048*t_{I2CCLK}, the TIMEOUT flag will be set in I2C_STAT register.

The BUSTOB[11:0] is used to check the $t_{LOW:SEXT}$ of the slave and the $t_{LOW:MEXT}$ of the master. The timer can be enabled by setting the EXTOEN bit in the I2C_TIMEOUT register, after the EXTOEN bit is set, the BUSTOB [11:0] cannot be changed. If the SCL stretching time of the SMBus peripheral is greater than (BUSTOB+1)*2048*t_{I2CCLK} and within the timeout interval described in the bus idle detection section, the TIMEOUT bit in the I2C_STAT register will be set.

Packet error checking

There is a CRC-8 calculator in I2C block to perform Packet Error Checking for I2C data. A PEC (packet error code) byte is appended at the end of each transfer. The byte is calculated as CRC-8 checksum, calculated over the entire message including the address and read/write bit. The polynomial used is x^8+x^2+x+1 (the CRC-8-ATM HEC algorithm, initialized to zero).

When I2C is disabled, the PEC can be enabled by setting the PECEN bit in I2C_CTL0 register. Since the PEC transmission is managed by BYTENUM [7:0] in I2C_CTL1 register, SBCTL bit must be set when act as a slave. When PECTRANS is set and the RELOAD bit is cleared, PEC is transmitted after the BYTENUM [7:0]-1 data byte. The PECTRANS has no effect if RELOAD is set.

Table 19-4. SMBus with PEC configuration

Mode	SBCTL bit	RELOAD bit	AUTOEND bit	PECTRANS bit
Master Tx/Rx BYTENUM + PEC+ STOP	x	0	1	1
Master Tx/Rx BYTENUM + PEC + RESTART	x	0	0	1
Slave Tx/Rx with PEC	1	0	x	1

SMBus alert

The SMBus has an extra optional shared interrupt signal called SMBALERT# which can be used by slaves to tell the host to ask its slaves about events of interest. The host processes the interrupt and accesses all SMBALERT# devices through the Alert Response Address at the same time. If the SMBALERT# is pulled low by the devices, the devices will acknowledge the Alert Response Address. When SMBHAEN is 0, it is configured as a slave device, the SMBA pin will be pulled low by setting the SMBALTEN bit in the I2C_CTL0 register. Meanwhile the Alert Response Address is enabled. When SMBHAEN is 1, it is configured as a host, and the SMBALTEN is 1, as soon as a falling edge is detected on the SMBA pin, the SMBALT flag will be set in the I2C_STAT register. If the ERRIE bit is set in the I2C_CTL0 register, an interrupt will be generated. When SMBALTEN is 0, the level of ALERT line is considered high even if the SMBA pin is low. The SMBA pin can be used as a standard GPIO if SMBALTEN is 0.

Bus idle detection

If the master detects that the high level duration of the clock and data signals is greater than

$t_{HIGH,MAX}$, the bus can be considered idle.

This timing parameter includes the case of a master that has been dynamically added to the bus and may not have detected a state transition on a SMBCLK or SMBDAT lines. In this case, in order to ensure that there is no ongoing transmission, the master must wait long enough.

The BUSTOA[11:0] bits must be programmed with the timer reload value to enable the t_{IDLE} check in order to obtain the t_{IDLE} parameter. To detect SCL and SDA high level timeouts, the TOIDLE bit must be set. Then setting the TOEN bit in the I2C_TIMEOUT register to enable the timer, after the TOEN bit is set, the BUSTOA [11:0] bit and the TOIDLE bit cannot be changed. If the high level time of both SCL and SDA is greater than (BUSTOA+1)*4*t_{I2CCLK}, the TIMEOUT flag will be set in the I2C_STAT register.

SMBus slave mode

The SMBus receiver must be able to NACK each command or data it receives. For ACK control in slave mode, slave byte control mode can be enabled by setting SBCTL bit in I2C_CTL0 register.

SMBus-specific addresses should be enabled when needed. The SMBus Device Default address (0b1100 001) is enabled by setting the SMBDAEN bit in the I2C_CTL0 register. The SMBus Host address (0b0001 000) is enabled by setting the SMBHAEN bit in the I2C_CTL0 register. The Alert Response Address (0b0001 100) is enabled by setting the SMBALTEN bit in the I2C_CTL0 register.

19.3.10. SMBus mode

SMBus Master Transmitter and Slave Receiver

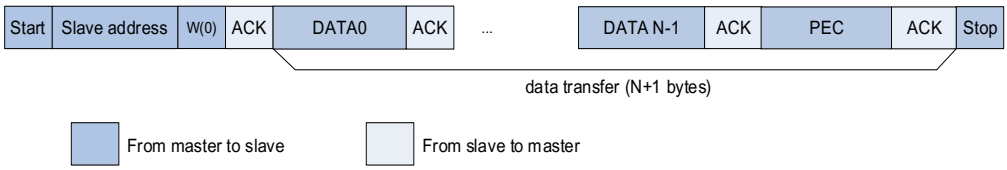
The PEC in SMBus master mode can be transmitted by setting the PECTRANS bit before setting the START bit, and the number of bytes in the BYTENUM [7:0] field must be configured. In this case, the total number of transmissions when TI interrupt occur is BYTENUM-1. So if BYTENUM=0x1 and PECTRANS bit is set at the same time, the contents of the I2C_PEC register are automatically transferred. If the automatic end mode is selected (AUTOEND=1), the SMBus master automatically sends the STOP signal after the PEC byte. If the automatic end mode is not selected (AUTOEND=0), the SMBus master can send a RESTART signal after the PEC. The I2C_PEC register content will be sent after BYTENUM-1 bytes, and the TC flag will be set after PEC is sent, then the SCL line is stretched low. The RESTART must be set in the TC interrupt routine.

When used as slave receiver, in order to allow PEC checking at the end of the number of bytes transmitted, SBCTL must be set. To configure ack control for each byte, the RELOAD must be set. In order to check the PEC byte, it is necessary to clear the RELOAD bit and set PECTRANS bit. After receiving BYTENUM-1 data, the next received byte will be compared with the data in the I2C_PEC register. If the PEC values does not match, the NACK is automatically generated. If the PEC values matches, the ACK is automatically generated, regardless of the NACKEN bit value. When PEC byte is received, it is also copied into the

I2C_RDATA register, and RBNE flag will be set. If the ERRIE bit in I2C_CTL0 register is 1, when PEC value does not match, the PECERR flag will be set and the interrupt will be generated. If ACK control is not required, then PECTRANS can be set to 1 and BYTENUM can be programmed according to the number of bytes to be received.

Note: After the RELOAD bit is set, the PECTRANS cannot be changed.

Figure 19-22. SMBus Master Transmitter and Slave Receiver communication flow



SMBus Master Receiver and Slave Transmitter

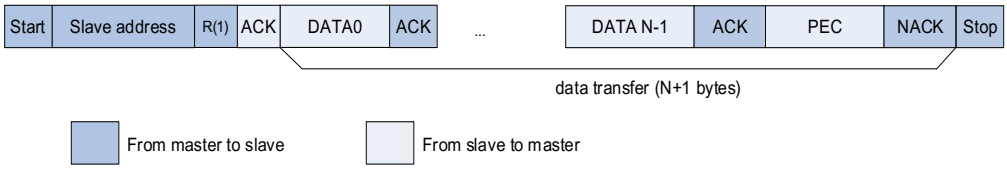
If the SMBus master is required to receive PEC at the end of bytes transfer, automatic end mode can be enabled. Before sending a START signal on the bus, PECTRANS bit must be set and slave addresses must be programmed. After receiving BYTENUM-1 data, the next received byte will be compared with the data in the I2C_PEC register automatically. A NACK is respond to the PEC byte before STOP signal.

If the SMBus master receiver is required to generate a RESTART signal after receiving PEC byte, the software mode (AUTOEND = 0) must be selected. Before sending a START signal to the bus, PECTRANS bit must be set and slave addresses must be programmed. After receiving BYTENUM-1 data, the next received byte will be compared with the contents of the I2C_PEC register automatically. The TC flag will be set after PEC is sent, then the SCL line is stretched low. The RESTART can be set in the TC interrupt routine.

When used as slave transmitter, in order to allow PEC transmission at the end of BYTENUM [7:0] bytes, SBCTL must be set. If PECTRANS bit is set, the number of bytes in BYTENUM [7:0] contains PEC byte. In this case, if the number of bytes requested by the master is greater than BYTENUM-1, the total number of TI interrupts will be BYTENUM-1, and the contents of the I2C_PEC register will be transmitted automatically.

Note: After the RELOAD bit is set, the PECTRANS cannot be changed.

Figure 19-23. SMBus Master Receiver and Slave Transmitter communication flow



19.3.11. Use DMA for data transfer

As is shown in I2C slave mode and I2C master mode, each time TI or RBNE is asserted, software should write or read a byte, this may cause CPU's high overload. The DMA controller

can be used to process TI and RBNE flag: each time TI or RBNE is asserted, DMA controller does a read or write operation automatically.

The DMA transmission request is enabled by setting the DENT bit in I2C_CTL0 register. The DMA reception request is enabled by setting the DENR bit in I2C_CTL0 register. In master mode, the slave address, transmission direction, number of bytes and START bit are programmed by software. The DMA must be initialized before setting the START bit. The number of bytes to be transferred is configured in the BYTENUM [7:0] in I2C_CTL1 register. In slave mode, the DMA must be initialized before the address match event or in the ADDSEND interrupt routine, before clearing the ADDSEND flag.

Note: When I2C is in slave mode, SBCTL=0, and uses DMA to send data, it is necessary to write TDATA register once more after the DMA transmission is completed (This data will not be sent out), otherwise, an exception will occur when the slave uses DMA to send data next time

19.3.12. I2C error and interrupts

The I2C error flags are listed in *Table 19-5. I2C error flags*.

Table 19-5. I2C error flags

I2C Error Name	Description
BERR	Bus error
LOSTARB	Arbitration lost
OUERR	Overrun/Underrun flag
PECERR	CRC value doesn't match
TIMEOUT	Bus timeout in SMBus mode
SMBALT	SMBus Alert

The I2C interrupt events and flags are listed in *Table 19-6. I2C interrupt events*.

Table 19-6. I2C interrupt events

Interrupt event	Event flag	Enable control bit
I2C_RDATA is not empty during receiving	RBNE	RBNEIE
Transmit interrupt	TI	TIE
STOP signal detected	STPDET	STPDETIE
Transfer complete reload	TCR	TCIE
Transfer complete	TC	
Address match	ADDSEND	ADDMIE
Not acknowledge received	NACK	NACKIE
Bus error	BERR	ERRIE
Arbitration Lost	LOSTARB	
Overrun/Underrun error	OUERR	
PEC error	PECERR	
Timeout error	TIMEOUT	
SMBus Alert	SMBALT	

19.3.13. I2C debug mode

When the microcontroller enters the debug mode (CM7_0 or CM7_1 core halted), the SMBus timeout either continues to work normally or stops, depending on the I2Cx_HOLD0/I2Cx_HOLD1 configuration bits in the DBG module.

19.4. Register definition

I2C0 base address: 0x4000 5400

I2C1 base address: 0x4000 5800

Table 19-7. I2C register overview

Register Name	Symbol	Address Offset	Reset Value
Control register 0	I2C_CTL0	0x00	0x0000 0000
Control register 1	I2C_CTL1	0x04	0x0000 0000
Slave address register 0	I2C_SADDR0	0x08	0x0000 0000
Slave address register 1	I2C_SADDR1	0x0C	0x0000 0000
Timing register	I2C_TIMING	0x10	0x0000 0000
Timeout register	I2C_TIMEOUT	0x14	0x0000 0000
Status register	I2C_STAT	0x18	0x0000 0001
Status clear register	I2C_STATC	0x1C	0x0000 0000
PEC register	I2C_PEC	0x20	0x0000 0000
Receive data register	I2C_RDATA	0x24	0x0000 0000
Transmit data register	I2C_TDATA	0x28	0x0000 0000
Control register 2	I2C_CTL2	0x90	0x0000 FE00

19.4.1. Control register 0 (I2C_CTL0)

Address offset: 0x00

Reset value: 0x0000 0000

This register can be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								PECEN	SMBALTEN	SMBDAEN	SMBHAEN	GCEN	Reserved	SS	SBCTL
								rw	rw	rw	rw	rw		rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DENR	DENT	Reserved	ANOFF	DNF[3:0]				ERRIE	TCIE	STPDETIE	NACKIE	ADDMIE	RBNEIE	TIE	I2CEN
rw	rw		rw	rw				rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23	PECEN	PEC Calculation Switch 0: PEC Calculation off 1: PEC Calculation on
22	SMBALTEN	SMBus Alert enable 0: SMBA pin is not pulled down (device mode) or SMBus Alert pin SMBA is disabled (host mode) 1: SMBA pin is pulled down (device mode) or SMBus Alert pin SMBA is enabled

		(host mode)
21	SMBDAEN	SMBus device default address enable 0: Device default address is disabled, the default address 0b1100001x will be not acknowledged. 1: Device default address is enabled, the default address 0b1100001x will be acknowledged.
20	SMBHAEN	SMBus host address enable 0: Host address is disabled, address 0b0001000x will be not acknowledged. 1: Host address is enabled, address 0b0001000x will be acknowledged.
19	GCEN	Whether or not to response to a General Call (0x00) 0: Slave won't response to a General Call 1: Slave will response to a General Call
18	Reserved	Must be kept at reset value.
17	SS	Whether to stretch SCL low when data is not ready in slave mode. This bit is set and cleared by software. 0: SCL Stretching is enabled 1: SCL Stretching is disabled **Note:** When in master mode, this bit must be 0. This bit can be modified when I2CEN = 0.
16	SBCTL	Slave byte control This bit is used to enable hardware byte control in slave mode. 0: Slave byte control is disabled 1: Slave byte control is enabled
15	DENR	DMA enable for reception 0: DMA is disabled for reception 1: DMA is enabled for reception
14	DENT	DMA enable for transmission 0: DMA is disabled for transmission 1: DMA is enabled for transmission
13	Reserved	Must be kept at reset value.
12	ANOFF	Analog noise filter disable 0: Analog noise filter is enabled 1: Analog noise filter is disabled **Note:** This bit can only be programmed when the I2C is disabled (I2CEN = 0).
11:8	DNF[3:0]	Digital noise filter These bits are used to configure the digital noise filter on SDA and SCL input. The digital filter will filter spikes with a length of up to DNF[3:0]*t_{I2CCLK} 0000: Digital filter is disabled 0001: Digital filter is enabled and filter spikes with a length of up to 1 t_{I2CCLK}

...

1111: Digital filter is enabled and filter spikes with a length of up to15 t_{I2CCLK}

These bits can only be modified when the I2C is disabled (I2CEN = 0).

7	ERRIE	Error interrupt enable 0: Error interrupt disabled 1: Error interrupt enabled. When BERR, LOSTARB, OUERR, PECERR, TIMEOUT or SMBALT bit is set, an interrupt will be generated.
6	TCIE	Transfer complete interrupt enable 0: Transfer complete interrupt is disabled 1: Transfer complete interrupt is enabled
5	STPDETIE	Stop detection interrupt enable 0: Stop detection (STPDET) interrupt is disabled 1: Stop detection (STPDET) interrupt is enabled
4	NACKIE	Not acknowledge received interrupt enable 0: NACK received interrupt is disabled 1: NACK received interrupt is enabled
3	ADDMIE	Address match interrupt enable in slave mode 0: Address match interrupt is disabled 1: Address match interrupt is enabled
2	RBNEIE	Receive interrupt enable 0: Receive (RBNE) interrupt is disabled 1: Receive (RBNE) interrupt is enabled
1	TIE	Transmit interrupt enable 0: Transmit (TI) interrupt is disabled 1: Transmit (TI) interrupt is enabled
0	I2CEN	I2C peripheral enable 0: I2C is disabled 1: I2C is enabled

19.4.2. Control register 1 (I2C_CTL1)

Address offset: 0x04

Reset value: 0x0000 0000

This register can be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					PECTRANS	AUTOEND	RELOAD	BYTENUM[7:0]							
					rw	rw	rw	rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

NACKEN	STOP	START	HEAD10R	ADD10EN	TRDIR	SADDRESS[9:0]
rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:27	Reserved	Must be kept at reset value.
26	PECTRANS	PEC Transfer Set by software. Cleared by hardware in the following cases: When PEC byte is transferred or ADDSEND bit is set or STOP signal is detected or I2CEN=0. 0: Don't transfer PEC value 1: Transfer PEC **Note:** This bit has no effect when RELOAD=1, or SBCTL=0 in slave mode.
25	AUTOEND	Automatic end mode in master mode 0: TC bit is set when the transfer of BYTENUM [7:0] bytes is completed. 1: a STOP signal is sent automatically when the transfer of BYTENUM [7:0] bytes is completed. **Note:** This bit works only when RELOAD=0. This bit is set and cleared by software.
24	RELOAD	Reload mode 0: After the data of BYTENUM [7:0] bytes transfer, the transfer is completed. 1: After data of BYTENUM [7:0] bytes transfer, the transfer is not completed and the new BYTENUM [7:0] will be reloaded. Every time when the BYTENUM [7:0] bytes have been transferred, the TCR bit in I2C_STAT register will be set. This bit is set and cleared by software.
23:16	BYTENUM[7:0]	Number of bytes to be transferred These bits are programmed with the number of bytes to be transferred. When SBCTL=0, these bits have no effect. **Note:** These bits should not be modified when the START bit is set.
15	NACKEN	Generate NACK in slave mode 0: an ACK is sent after receiving a new byte. 1: a NACK is sent after receiving a new byte. **Note:** The bit can be set by software, and cleared by hardware when the NACK is sent, or when a STOP signal is detected or ADDSEND is set, or when I2CEN=0. When PEC is enabled, whether to send an ACK or a NACK is not depend on the NACKEN bit. When SS=1, and the OUERR bit is set, the value of NACKEN is ignored and a NACK will be sent.
14	STOP	Generate a STOP signal on I2C bus This bit is set by software and cleared by hardware when I2CEN=0 or STOP signal is detected. 0: STOP will not be sent

		1: STOP will be sent
13	START	Generate a START signal on I2C bus This bit is set by software and cleared by hardware after the address is sent. When the arbitration is lost, or a timeout error occurred, or I2CEN=0, this bit can also be cleared by hardware. It can be cleared by software by setting the ADDSENDC bit in I2C_STATC register. 0: START will not be sent 1: START will be sent
12	HEAD10R	10-bit address header executes read direction only in master receive mode 0: The 10-bit master receive address sequence is START + header of 10-bit address (write) + slave address byte 2 + RESTART + header of 10-bit address (read). 1: The 10-bit master receive address sequence is RESTART + header of 10-bit address (read). **Note:** When the START bit is set, this bit cannot be changed.
11	ADD10EN	10-bit addressing mode enable in master mode 0: 7-bit addressing in master mode 1: 10-bit addressing in master mode **Note:** When the START bit is set, this bit cannot be modified.
10	TRDIR	Transfer direction in master mode 0: Master transmit 1: Master receive **Note:** When the START bit is set, this bit cannot be modified.
9:0	SADDRESS[9:0]	Slave address to be sent SADDRESS[9:8]: Slave address bit 9:8 If ADD10EN = 0, these bits have no effect. If ADD10EN = 1, these bits should be written with bits 9:8 of the slave address to be sent. SADDRESS[7:1]: Slave address bit 7:1 If ADD10EN = 0, these bits should be written with the 7-bit slave address to be sent. If ADD10EN = 1, these bits should be written with bits 7:1 of the slave address to be sent. SADDRESS0: Slave address bit 0 If ADD10EN = 0, this bit has no effect. If ADD10EN = 1, this bit should be written with bit 0 of the slave address to be sent **Note:** When the START bit is set, the bit filed cannot be modified.

19.4.3. Slave address register 0 (I2C_SADDR0)

Address offset: 0x08

Reset value: 0x0000 0000

This register can be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESSEN	Reserved				ADDFORMAT	ADDRESS[9:8]		ADDRESS[7:1]							ADDRESS0
rw					rw	rw		rw							rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	ADDRESSEN	I2C address enable 0: I2C address disable. 1: I2C address enable.
14:11	Reserved	Must be kept at reset value.
10	ADDFORMAT	Address mode for the I2C slave 0: 7-bit address 1: 10-bit address **Note:** When ADDRESSEN is set, this bit should not be written.
9:8	ADDRESS[9:8]	Highest two bits of a 10-bit address **Note:** When ADDRESSEN is set, this bit should not be written.
7:1	ADDRESS[7:1]	7-bit address or bits 7:1 of a 10-bit address **Note:** When ADDRESSEN is set, this bit should not be written.
0	ADDRESS0	Bit 0 of a 10-bit address **Note:** When ADDRESSEN is set, this bit should not be written.

19.4.4. Slave address register 1 (I2C_SADDR1)

Address offset: 0x0C

Reset value: 0x0000 0000

This register can be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS2EN	Reserved				ADDMSK2[2:0]			ADDRESS2[6:0]							Reserved
rw					rw			rw							

Bits	Fields	Descriptions

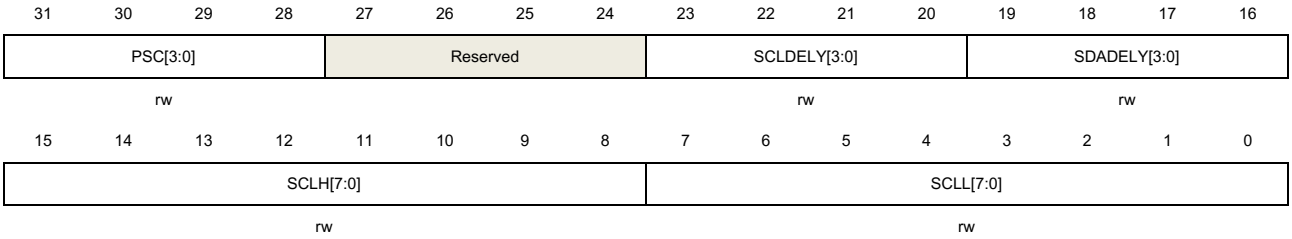
31:16	Reserved	Must be kept at reset value.
15	ADDRESS2EN	Second I2C address enable 0: Second I2C address disable. 1: Second I2C address enable.
14:11	Reserved	Must be kept at reset value.
10:8	ADDMSK2[2:0]	ADDRESS2[7:1] mask Defines which bits of ADDRESS2[7:1] are compared with an incoming address byte, and which bits are masked (don't care). 000: No mask, all the bits must be compared. N (001~110) : ADDRESS2[n:1] is masked. Only ADDRESS2[7:n+1] are compared. 111: ADDRESS2[7:1] are masked. All 7-bit received addresses are acknowledged except the reserved address (0b0000xxx and 0b1111xxx). **Note:** When ADDRESS2EN is set, these bits should not be written. If ADDMSK2 is not equal to 0, the reserved I2C addresses (0b0000xxx and 0b1111xxx) are not acknowledged even if all the bits are matched.
7:1	ADDRESS2[7:1]	Second I2C address for the slave **Note:** When ADDRESS2EN is set, these bits should not be written.
0	Reserved	Must be kept at reset value.

19.4.5. Timing register (I2C_TIMING)

Address offset: 0x10

Reset value: 0x0000 0000

This register can be accessed by word (32-bit).



Bits	Fields	Descriptions
31:28	PSC[3:0]	Timing prescaler In order to generate the clock period t_{PSC} used for data setup and data hold counters, these bits are used to configure the prescaler for I2CCLK. The t_{PSC} is also used for SCL high and low level counters. $t_{PSC}=(PSC+1)*t_{I2CCLK}$
27:24	Reserved	Must be kept at reset value.
23:20	SCLDELY[3:0]	Data setup time A delay $t_{SCLDELY}$ between SDA edge and SCL rising edge can be generated by

		configuring these bits. And during $t_{SCLDELY}$, the SCL line is stretched low in master mode and in slave mode when SS = 0. $t_{SCLDELY}=(SCLDELY+1)*t_{PSC}$
19:16	SDADELY[3:0]	Data hold time A delay $t_{SDADELY}$ between SCL falling edge and SDA edge can be generated by configuring these bits. And during $t_{SDADELY}$, the SCL line is stretched low in master mode and in slave mode when SS = 0.When SDADELY=1 and SDADELY=0, this delay does not take effect. $t_{SDADELY}=(SDADELY-1)*t_{PSC}$
15:8	SCLH[7:0]	SCL high period SCL high period can be generated by configuring these bits. $t_{SCLH}=(SCLH+1)*t_{PSC}$ **Note:** These bits can only be used in master mode.
7:0	SCLL[7:0]	SCL low period SCL low period can be generated by configuring these bits. $t_{SCLL}=(SCLL+1)*t_{PSC}$ **Note:** These bits can only be used in master mode.

19.4.6. Timeout register (I2C_TIMEOUT)

Address offset: 0x14

Reset value: 0x0000 0000

This register can be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EXTOEN	Reserved			BUSTOB[11:0]											
rw				rw											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOEN	Reserved		TOIDLE	BUSTOA[11:0]											
rw			rw	rw											

Bits	Fields	Descriptions
31	EXTOEN	Extended clock timeout detection enable When a cumulative SCL stretch time is greater than $t_{LOW:EXT}$, a timeout error will be occurred. $t_{LOW:EXT}=(BUSTOB+1)*2048*t_{I2CCLK}$. 0: Extended clock timeout detection is disabled. 1: Extended clock timeout detection is enabled.
30:28	Reserved	Must be kept at reset value.
27:16	BUSTOB	Bus timeout B Configure the cumulative clock extension timeout. In master mode, the master cumulative clock low extend time $t_{LOW:MEXT}$ is detected. In slave mode, the slave cumulative clock low extend time $t_{LOW:SEXT}$ is detected.

		$t_{LOW:EXT}$=(BUSTOB+1)*2048*t_{I2CCLK}. **Note:** These bits can be modified only when EXTOEN =0.
15	TOEN	Clock timeout detection enable If the SCL stretch time greater than $t_{TIMEOUT}$ when TOIDLE =0 or high for more than t_{IDLE} when TOIDLE =1, a timeout error is detected. 0: SCL timeout detection is disabled 1: SCL timeout detection is enabled
14:13	Reserved	Must be kept at reset value.
12	TOIDLE	Idle clock timeout detection 0: BUSTOA is used to detect SCL low timeout 1: BUSTOA is used to detect both SCL and SDA high timeout when the bus is idle **Note:** This bit can be written only when TOEN =0.
11:0	BUSTOA	Bus timeout A When TOIDLE=0, $t_{TIMEOUT}$=(BUSTOA+1)*2048*t_{I2CCLK} When TOIDLE=1, t_{IDLE}=(BUSTOA+1)*4*t_{I2CCLK} **Note:** These bits can be written only when TOEN =0.

19.4.7. Status register (I2C_STAT)

Address offset: 0x18

Reset value: 0x0000 0001

This register can be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								READDR[6:0]							TR
								r							r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I2CBSY	Reserved	SMBALT	TIMEOUT	PECERR	OUERR	LOSTARB	BERR	TCR	TC	STPDET	NACK	ADDSEND	RBNE	TI	TBE
r		r	r	r	r	r	r	r	r	r	r	r	r	rw	rw

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:17	READDR[6:0]	Received match address in slave mode When the ADDSEND bit is set, these bits store the matched address. In the case of a 10-bit address, READDR[6:0] stores the header of the 10-bit address followed by the 2 MSBs of the address.
16	TR	Whether the I2C is a transmitter or a receiver in slave mode This bit is updated when the ADDSEND bit is set. 0: Receiver 1: Transmitter

15	I2CBSY	Busy flag This bit is set by hardware when a START signal is detected and cleared by hardware after a STOP signal. When I2CEN=0, this bit is also cleared by hardware. 0: No I2C communication. 1: I2C communication active.
14	Reserved	Must be kept at reset value.
13	SMBALT	SMBus Alert When SMBHAEN=1, SMBALTEN=1, and a SMBALERT event (falling edge) is detected on SMBA pin, this bit will be set by hardware. It is cleared by software by setting the SMBALTC bit. This bit is cleared by hardware when I2CEN=0. 0: SMBALERT event is not detected on SMBA pin 1: SMBALERT event is detected on SMBA pin
12	TIMEOUT	TIMEOUT flag. When a timeout or extended clock timeout occurred, this bit will be set. It is cleared by software by setting the TIMEOUTC bit and cleared by hardware when I2CEN=0. 0: no timeout or extended clock timeout occur 1: a timeout or extended clock timeout occur
11	PECERR	PEC error This flag is set by hardware when the received PEC does not match with the content of I2C_PEC register. Then a NACK is automatically sent. It is cleared by software by setting the PECERRC bit and cleared by hardware when I2CEN=0. 0: Received PEC and content of I2C_PEC match 1: Received PEC and content of I2C_PEC don't match, I2C will send NACK regardless of NACKEN bit.
10	OUERR	Overrun/Underrun error in master / slave mode In master / slave mode, when an overrun/underrun error occurs, this bit will be set by hardware. It is cleared by software by setting the OUERRC bit and cleared by hardware when I2CEN=0. 0: No overrun or underrun occurs 1: Overrun or underrun occurs
9	LOSTARB	Arbitration Lost It is cleared by software by setting the LOSTARBC bit and cleared by hardware when I2CEN=0. 0: No arbitration lost. 1: Arbitration lost occurs and the I2C block changes back to slave mode.
8	BERR	Bus error When an unexpected START or STOP signal on I2C bus is detected, a bus error occurs and this bit will be set. It is cleared by software by setting BERRC bit and cleared by hardware when I2CEN=0. 0: No bus error

1: A bus error detected

7	TCR	Transfer complete reload This bit is set by hardware when RELOAD=1 and data of BYTENUM[7:0] bytes have been transferred. It is cleared by software when BYTENUM[7:0] is written to a non-zero value. 0: When RELOAD=1, transfer of BYTENUM[7:0] bytes is not completed 1: When RELOAD=1, transfer of BYTENUM[7:0] bytes is completed
6	TC	Transfer complete in master mode This bit is set by hardware when RELOAD=0, AUTOEND=0 and data of BYTENUM[7:0] bytes have been transferred. It is cleared by software when START bit or STOP bit is set. 0: Transfer of BYTENUM[7:0] bytes is not completed 1: Transfer of BYTENUM[7:0] bytes is completed
5	STPDET	STOP signal detected in slave mode This flag is set by hardware when a STOP signal is detected on the bus. It is cleared by software by setting STPDETC bit and cleared by hardware when I2CEN=0. 0: STOP signal is not detected. 1: STOP signal is detected.
4	NACK	Not Acknowledge flag This flag is set by hardware when a NACK is received. It is cleared by software by setting NACKC bit and cleared by hardware when I2CEN=0. 0: ACK is received. 1: NACK is received.
3	ADDSEND	Address received matches in slave mode. This bit is set by hardware when the received slave address matched with one of the enabled slave addresses. It is cleared by software by setting ADDSENDC bit and cleared by hardware when I2CEN=0. 0: Received address not matched 1: Received address matched
2	RBNE	I2C_RDATA is not empty during receiving This bit is set by hardware when the received data is shift into the I2C_RDATA register. It is cleared when I2C_RDATA is read. 0: I2C_RDATA is empty 1: I2C_RDATA is not empty, software can read
1	TI	Transmit interrupt This bit is set by hardware when the I2C_TDATA register is empty and the I2C is ready to transmit data. It is cleared when the next data to be sent is written in the I2C_TDATA register. When SS=1, this bit can be set by software, in order to generate a TI event (interrupt if TIE=1 or DMA request if DENT =1). 0: I2C_TDATA is not empty or the I2C is not ready to transmit data

1: I2C_TDATA is empty and the I2C is ready to transmit data

0	TBE	I2C_TDATA is empty during transmitting This bit is set by hardware when the I2C_TDATA register is empty. It is cleared when the next data to be sent is written in the I2C_TDATA register. This bit can be set by software in order to empty the I2C_TDATA register. 0: I2C_TDATA is not empty 1: I2C_TDATA is empty

19.4.8. Status clear register (I2C_STATC)

Address offset: 0x1C

Reset value: 0x0000 0000

This register can be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		SMBALTC	TIMEOUTC	PECERRC	OUERRC	LOSTARBC	BERRC	Reserved		STPDETC	NACKC	ADDSENDC	Reserved		
		w	w	w	w	w	w			w	w	w			

Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13	SMBALTC	SMBus alert flag clear. Software can clear the SMBALT bit of I2C_STAT by writing 1 to this bit
12	TIMEOUTC	TIMEOUT flag clear. Software can clear the TIMEOUT bit of I2C_STAT by writing 1 to this bit
11	PECERRC	PEC error flag clear. Software can clear the PECERR bit of I2C_STAT by writing 1 to this bit
10	OUERRC	Overrun/Underrun flag clear. Software can clear the OUERR bit of I2C_STAT by writing 1 to this bit
9	LOSTARBC	Arbitration Lost flag clear. Software can clear the LOSTARB bit of I2C_STAT by writing 1 to this bit
8	BERRC	Bus error flag clear. Software can clear the BERR bit of I2C_STAT by writing 1 to this bit
7:6	Reserved	Must be kept at reset value.
5	STPDETC	STPDET flag clear Software can clear the STPDET bit of I2C_STAT by writing 1 to this bit

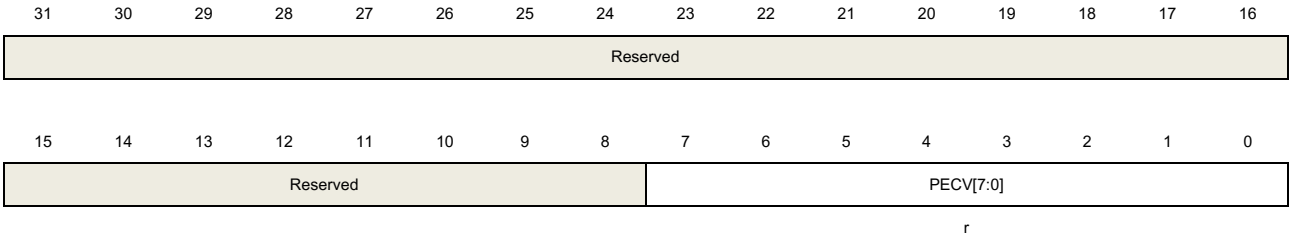
4	NACKC	Not Acknowledge flag clear Software can clear the NACK bit of I2C_STAT by writing 1 to this bit
3	ADDSENDC	ADDSEND flag clear Software can clear the ADDSEND bit of I2C_STAT by writing 1 to this bit
2:0	Reserved	Must be kept at reset value.

19.4.9. PEC register (I2C_PEC)

Address offset: 0x20

Reset value: 0x0000 0000

This register can be accessed by word (32-bit).



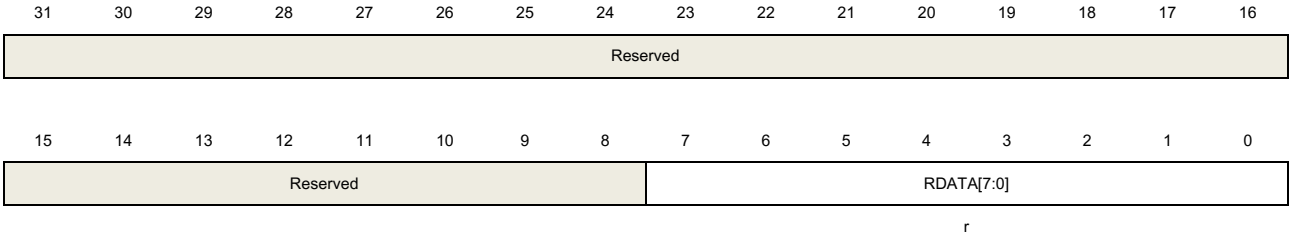
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	PECV[7:0]	Packet Error Checking Value that calculated by hardware when PEC is enabled. PECV is cleared by hardware when I2CEN = 0.

19.4.10. Receive data register (I2C_RDATA)

Address offset: 0x24

Reset value: 0x0000 0000

This register can be accessed by word (32-bit).



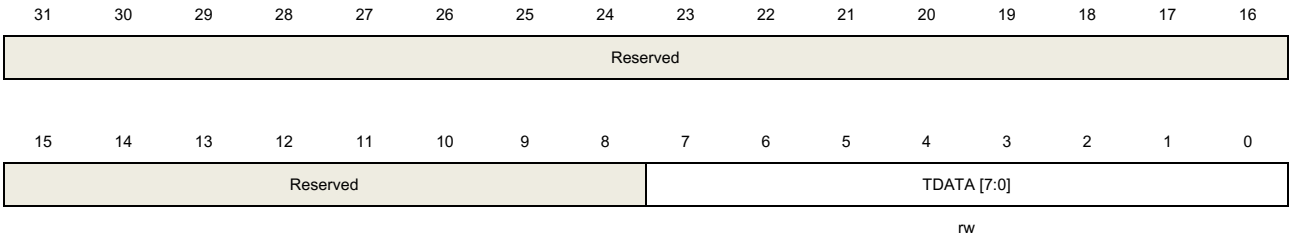
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	RDATA[7:0]	Receive data value

19.4.11. Transmit data register (I2C_TDATA)

Address offset: 0x28

Reset value: 0x0000 0000

This register can be accessed by word (32-bit).



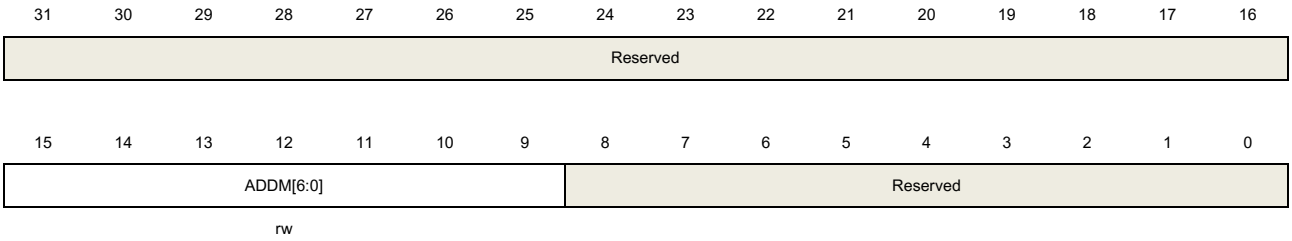
Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	TDATA[7:0]	Transmit data value

19.4.12. Control register 2 (I2C_CTL2)

Address offset: 0x90

Reset value: 0x0000 FE00

This register can be accessed by word (32-bit).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:9	ADDM[6:0]	Defines which bits of ADDRESS[7:1] are compared with an incoming address byte, and which bits are ignored. Any bit set to 1 in ADDM[6:0] enables comparisons with the corresponding bit in ADDRESS[7:1]. Bits set to 0 are ignored (can be either 0 or 1 in the incoming address).
8:0	Reserved	Must be kept at reset value.

20. Serial peripheral interface/Inter-IC sound (SPI/I2S)

20.1. Overview

The SPI/I2S module can communicate with external devices using the SPI protocol or the I2S audio protocol.

The serial peripheral interface (SPI) provides a SPI protocol of data transmission and reception function in master or slave mode. Both full-duplex and simplex communication modes are supported, with hardware CRC calculation. Quad-SPI master mode is only supported in SPI0 and SPI2.

The inter-IC sound (I2S) supports four audio standards: I2S Philips standard, MSB justified standard, LSB justified standard, and PCM standard. I2S works at either master or slave mode for transmission and reception.

20.2. Characteristics

20.2.1. SPI characteristics

- Master or slave operation with full-duplex or simplex mode.
- Separate transmission and reception 8 * 32-bit FIFO with DMA capability.
- Supports FIFO loopback mode.
- Data frame size can be 4 to 32 bits.
- Bit order can be LSB or MSB.
- Software and hardware NSS management, GD32A711 / A712 devices supports 4 NSS for SPI.
- Supports NSS polarity configuration.
- Supports master request to start SPI bus transfer.
- Hardware CRC calculation and transmission.
- CRC size is equal to SPI transmission size.
- SPI TI mode supported.
- SPI NSS pulse mode supported.
- Quad-SPI configuration available in master mode (Only SPI0 and SPI2 are supported).
- Hardware continuous frame mode supported
- Hardware single frame mode supported
- Master IDLE state timeout monitoring.
- The timing of master control signals (NSS, SCK) is configurable.
- Master receive data sampling delay is configurable.
- SCK frequency supports up to 20 MHz.

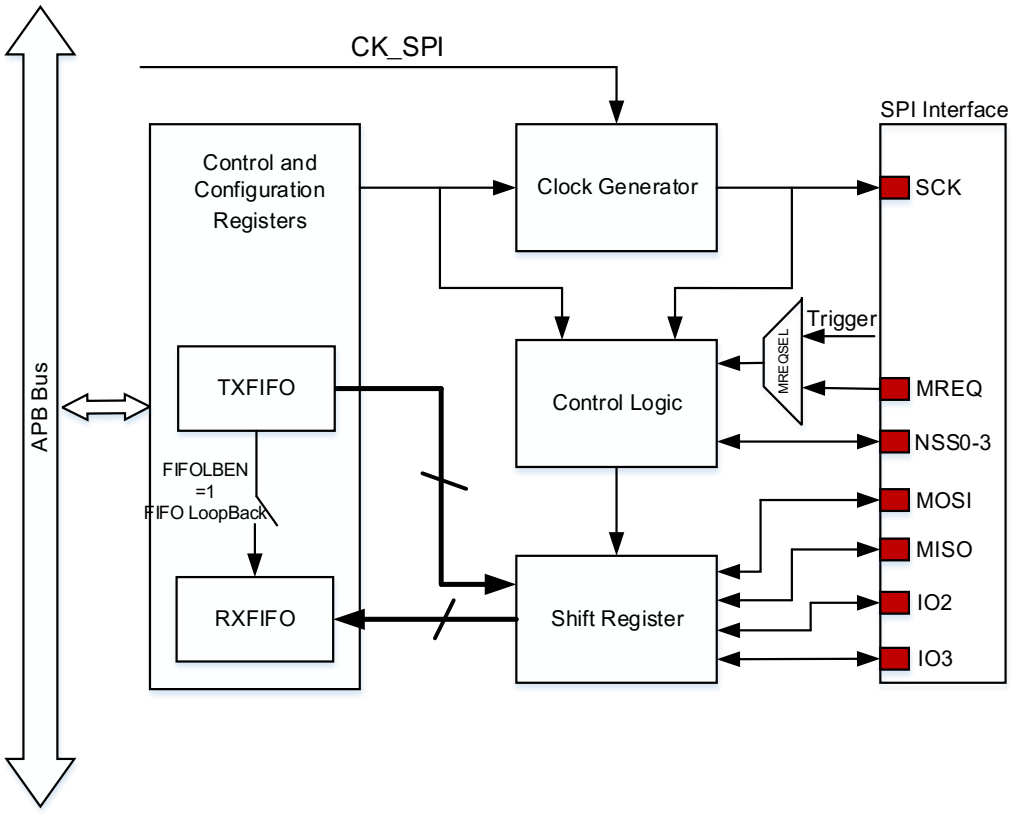
20.2.2. I2S characteristics

- Master or slave operation for transmission/reception.
- Four I2S standards supported: Philips, MSB justified, LSB justified and PCM standard.
- Data length can be 16 bits, 24 bits or 32 bits.
- Channel length can be 16 bits or 32 bits.
- Separate transmit and receive 8*32-bit FIFO with DMA capability
- Support FIFO loopback for safety usage.
- Audio sample frequency can be 8 kHz to 192 kHz using I2S clock divider.
- Programmable idle state clock polarity.
- Master clock (MCK) can be output.

20.3. SPI function overview

20.3.1. SPI block diagram

Figure 20-1. Block diagram of SPI



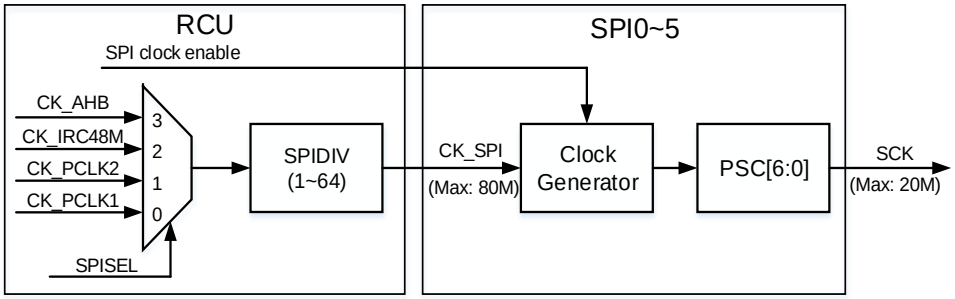
20.3.2. SPI clock sources

For GD32A711 / A712 devices, SPI0~5 supports switching among four different clock sources: CK_PCLK1, CK_PCLK2, CK_IRC48M, and CK_AHB. These clocks are divided by

configuring the SPIDIV[5:0] bits in the *5.4.12* register to generate the SPI clock (CK_SPI). The specific configuration method can be referenced in the *Clock control unit (CCTL)*.

By configuring the PSC[6:0] bit field in the SPI_CTL0 register, the SPI clock can be divided to generate SPI communication clocks (SCK) of different frequencies. The maximum supported communication rate for SPI is up to 20 MHz.

Figure 20-2. Block diagram of SPI clock sources



20.3.3. SPI signal description

Normal configuration (Not Quad-SPI Mode)

Table 20-1. SPI signal description

Pin name	Direction	Description
SCK	I/O	Master: SPI clock output Slave: SPI clock input
MISO	I/O	Master: Data reception line Slave: Data transmission line Master with bidirectional mode: Not used Slave with bidirectional mode: Data transmission and reception line.
MOSI	I/O	Master: Data transmission line Slave: Data reception line Master with bidirectional mode: Data transmission and reception line. Slave with bidirectional mode: Not used
NSS0-3	I/O	Software NSS mode: not used Master in hardware NSS mode: when NSSDRV=1, it is NSS output, suitable for single master application; when NSSDRV=0, it is NSS input, suitable for multi-master application. Slave in hardware NSS mode: NSS input, as a chip select signal for slave.
MREQ	I/O	When MREQEN = 1 and, master request function enabled. If MREQSEL = 0, MREQ pin as Master request input source

Pin name	Direction	Description
		and MREQ pin multiplexes NSS1 pin. Master mode: MREQ pin is input Slave mode: MREQ pin is output

Quad-SPI configuration

SPI is in single wire mode by default and enters into Quad-SPI mode after QMOD bit in SPI_QCTL register is set (only available in SPI0 and SPI2). Quad-SPI mode can only work in master mode.

The SPI is connected to external devices through 6 pins in Quad-SPI mode:

Table 20-2. Quad-SPI signal description

Pin name	Direction	Description
SCK	O	SPI clock output
MOSI	I/O	Transmission/Reception data 0
MISO	I/O	Transmission/Reception data 1
IO2	I/O	Transmission/Reception data 2
IO3	I/O	Transmission/Reception data 3
NSS0-3	O	NSS output

20.3.4. SPI clock timing and data format

CKPL and CKPH bits in SPI_CTL0 register decide the timing of SPI clock and data signal. The CKPL bit decides the SCK level when idle and CKPH bit decides either first or second clock edge is a valid sampling edge. These bits take no effect in TI mode.

Data order is configured by LF bit in SPI_CTL0 register, and SPI will first send the LSB if LF=1, or the MSB if LF=0. The data order is fixed to MSB first in TI mode.

By configuring the NSSPOL bit in the SPI_CTL1 register, the polarity of the NSS signal can be flexibly selected. When NSSPOL = 0, a low level indicates that NSS is active. When NSSPOL = 1, a high level indicates that NSS is active.

The SPI master supports control signal timing configuration. The NSSSCK and SCKNSS fields in the SPI_CKCFG0 register configure the minimum duration of NSS signal active to SCK signal active and SCK signal inactive to NSS signal inactive, respectively. NSSNSS is used to configure the minimum duration from NSS signal inactive to NSS signal active.

Figure 20-3. SPI timing diagram in normal mode

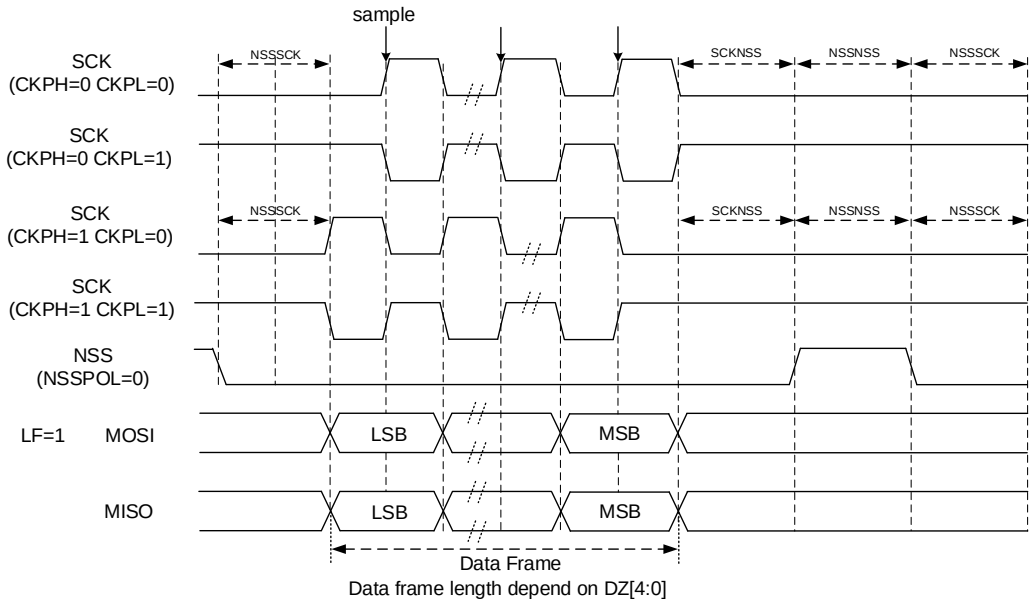
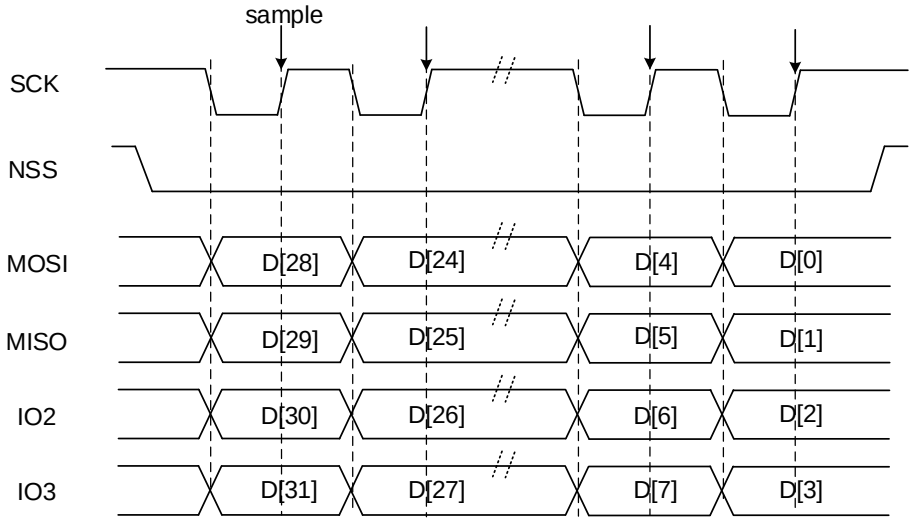


Figure 20-4. SPI timing diagram in Quad-SPI mode (CKPL=1, CKPH=1, LF=0, NSSPOL=0)

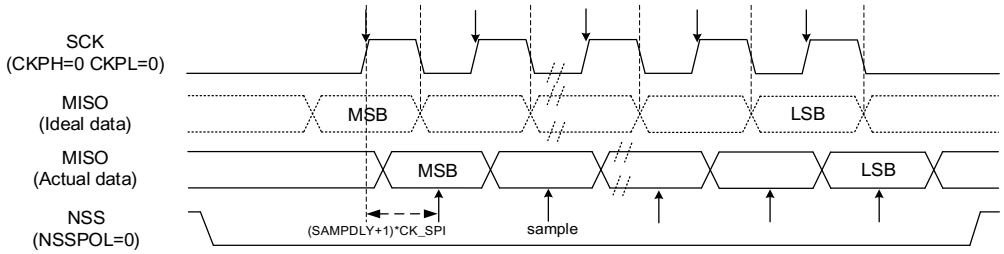


In SPI normal mode, the length of data is configured by the DZ[4:0] bits in the SPI_CTL1 register. It can be set from 4-bit up to 32-bit length and the setting applies for both transmission and reception. The data frame length is fixed to 32 bits in Quad-SPI mode.

20.3.5. SPI master receive delay compensation

During SPI master-slave communication, timing deviations may occur between the master's SCK clock signal and the slave's MISO data response due to physical link transmission delays. To optimize the reliability of data sampling, it is recommended to enable the delay compensation mechanism by setting the SAMPDLYEN bit in the SPI_CKCFG1 register. The precise delay value can be adjusted by configuring the SAMPDLY[11:0] bit field. By default, the sampling delay compensation time is set to 4 CK_SPI cycles.

Figure 20-5. SPI master receive delay diagram

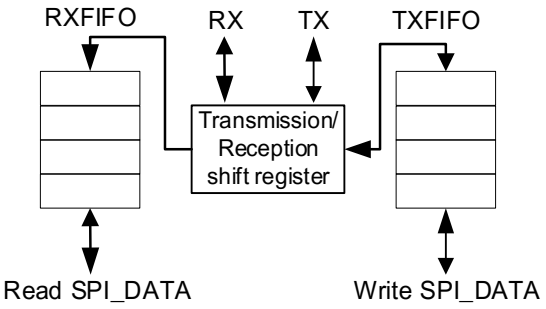


In master mode, when the receive sampling delay is disabled (SAMPDLYEN = 0), the master samples MISO at the start point of an SCK cycle. When the receive sampling delay is enabled (SAMPDLYEN = 1), the master uses the start point of SCK as the reference, waits for (SAMPDLYEN+1) / CK_SPI, and then samples MISO.

20.3.6. Separate transmission and reception FIFO

The separate 8 * 32-bit reception FIFO (RXFIFO) and transmission FIFO (TXFIFO) are used in different directions for SPI data transactions, and they can enable the SPI to work in a continuous flow.

Figure 20-6. Transmission and reception FIFO



A write access to the SPI_DATA register stores the written data in the end of TXFIFO, while a read access to the SPI_DATA returns the oldest value in RXFIFO which has not been read.

Write access of a data frame to be transmitted is managed by the TFA event. This event is triggered when the TXFIFO is deemed to have enough space for one next bus write. When TFA is cleared, the TXFIFO is considered as full. A read access to SPI_DATA is managed by the RFDR event. This is triggered when RXFIFO is considered to deemed to have enough data for one next bus read. When RFDR is cleared, the RXFIFO is considered to deemed to have no enough data for one next bus read. It is recommended that before the SPI master starts data transmission, the software reads the RXLVL bits in the SPI_STAT register. If the RXLVL value is not 0, the SPI_DATA register should be read RXLVL times to clear the receive FIFO.

Data alignment

During SPI communication, the length of each frame is determined by the value configured in DZ[4:0]. When accessing the SPI_DATA register, the data frame is always right-aligned according to the size configured by DZ[4:0]. For example, if the frame length is configured to 8 bits (DZ[4:0] = 0x7), when SPI sends data, the software writes a single word to the SPI_DATA register, occupying one TXFIFO space. The SPI hardware then continuously transmits the lower 8 bits of the data while ignoring the upper 24 bits. When SPI receives a frame of data, one RXFIFO space is occupied. When the software reads the SPI_DATA register, it must ignore the upper 24 bits of the received data.

FIFO loopback mode

In master mode, SPI TXFIFO/RXFIFO supports loopback function for safety usage. When FIFOLBEN = 1, the FIFO loopback mode is enabled, Write data to SPI_DATA will save the data to TXFIFO and then read back to RXFIFO directly, user can check the status of RXLVL[3:0] and read data from SPI_DATA to verify quality of FIFO.

20.3.7. Idle timeout

The timeout function is used to monitor the bus idle time after SPI is enabled in master mode. If the TOEN in the SPI_CTL1 register is enabled, and the idle time on the SPI bus exceeds the timeout threshold set by TOCFG[7:0], the TOF bit in the SPI_STAT register will be set. If TOIE is enabled, an interrupt will be generated to the core at this time.

20.3.8. NSS function

In GD32A711/A712 series, 4-NSS are supported for each SPI, user can use NSSSEL[1:0] to select NSS0 to NSS3.

Slave mode

When slave mode is configured (MSTMOD=0), SPI gets NSS level from NSS pin in hardware NSS mode (SWNSSEN = 0) or from SWNSS bit in software NSS mode (SWNSSEN = 1) and transmits/receives data only when NSS level is active. The NSS active level is configured through NSSPOL. In software NSS mode, NSS pin is not used.

Table 20-3. NSS function in slave mode

Mode	Register configuration	Description
Slave hardware NSS mode	MSTMOD = 0 SWNSSEN = 0	SPI slave gets NSS level from NSS pin. NSSPOL = 0: Active low NSSPOL = 1: Active high
Slave software NSS mode	MSTMOD = 0 SWNSSEN = 1	SPI slave NSS level is determined by the SWNSS bit. SWNSS = 0, NSSPOL = 0: NSS is active SWNSS = 1, NSSPOL = 1: NSS is active

Mode	Register configuration	Description
		SWNSS = 0, NSSPOL = 1: NSS is inactive SWNSS = 1, NSSPOL = 0: NSS is inactive

Master mode

In master mode (MSTMOD=1) if the application uses multi-master connection, NSS can be configured to hardware input mode (SWNSSEN=0, NSSDRV=0) or software mode (SWNSSEN=1). Then, once the NSS pin (in hardware NSS mode) or the SWNSS bit (in software NSS mode) becomes active (NSSPOL determines the active level), the SPI automatically enters slave mode and triggers a master fault flag CONFERR.

If the application wants to use NSS line to control the SPI slave, NSS should be configured to hardware output mode (SWNSSEN=0, NSSDRV=1). After enabling SPI, NSS becomes active when the transmission or reception process starts. When SPI is disabled, NSS becomes inactive.

The application may also use a general purpose IO as NSS pin to realize more flexible NSS.

Table 20-4. NSS function in master mode

Mode	Register configuration	Description
Master hardware NSS output mode	MSTMOD = 1 SWNSSEN = 0 NSSDRV=1 NSSPOL = 0: Active low NSSPOL = 1: Active high	Applicable to single-master mode. The master uses the NSS pin to control the SPI slave device. At this time, the NSS is configured as the hardware output mode. NSS becomes active after enabling SPI.
Master hardware NSS input mode	MSTMOD = 1 SWNSSEN = 0 NSSDRV=0 NSSPOL = 0: Active low NSSPOL = 1: Active high	Applicable to multi-master mode. At this time, NSS is configured as hardware input mode. Once the NSS pin level becomes active., SPI will automatically enter slave mode, and a master configuration error will occur and the CONFERR bit will be set to 1.
Master software NSS mode	MSTMOD = 1 SWNSSEN = 1 SWNSS = 0, NSSPOL = 0 (Or SWNSS = 1, NSSPOL = 1) NSSDRV: Don't care	Applicable to multi-master mode. When NSSPOL = 0, SWNSS = 0, or NSSPOL = 1, SWNSS = 1, SPI will automatically enter slave mode, and a master configuration error will occur and the CONFERR bit will be 1.
	MSTMOD = 1 SWNSSEN = 1 SWNSS = 1, NSSPOL = 0 (Or SWNSS = 0, NSSPOL = 1) NSSDRV: Don't care	The slave can use hardware or software NSS mode.

20.3.9. NSS switching

In an SPI single-master multi-slave communication topology, software can configure the master's NSSSEL to select a specific slave for SPI communication. When the master needs to switch communication to another slave, it must wait for the bus to become idle before switching the NSS pin, otherwise, incorrect data transmission may occur. The SPI bus idle timeout mechanism ensures that data transmission on the bus is complete before NSS switching. The software operation process is as follows:

1. The software initializes the master in hardware NSS output mode and the slave in hardware NSS mode. It also configures the NSS polarity (refer to ***NSS function***).
2. When the CPU is used for data transmission, after the last communication data is written into the master's transmit FIFO, the software sequentially configures the TOCFG, TOIE, and TOEN fields in the SPI_CTL1 register.
3. When the DMA is used for data transmission, the software waits for the DMA transfer completion flag to be set or the DMA completion interrupt to be triggered. After that, the fields TOCFG, TOIE, and TOEN are configured sequentially.
4. Wait for the TOF flag to be set or the timeout interrupt to be triggered (If TOIE is enabled). The software then disables TOEN and writes 0 to TOF to clear the current status.
5. Configure NSSSEL to select the next slave device for communication.

Note: In Motorola communication mode, the time interval between the bus idle state and switching NSSSEL must be greater than SCKNSS + 1 SCK clock cycles. After switching NSSSEL, the software must wait for NSSSCK + 1 SCK clock cycles before sending data. These delays need to be ensured by the software. In hardware single-frame mode, NSSP mode, or hardware continuous-frame mode, the hardware ensures the timing requirements without needing software delays. It is recommended to use hardware single-frame mode or hardware continuous-frame mode for SPI communication in a single-master multi-slave topology.

In master receive-only mode, the software first needs to clear SPIEN to end the SPI transmission and then wait for TRANS to reset to 0 before switching NSSSEL.

20.3.10. SPI operation modes

Table 20-5. SPI operation modes

Mode	Description	Register configuration	Data pin usage
MFD	Master full-duplex	MSTMOD = 1 RO = 0 BDEN = 0 BDOEN: Don't care	MOSI: Transmission MISO: Reception
MTU	Master transmission with unidirectional connection	MSTMOD = 1 RO = 0 BDEN = 0 BDOEN: Don't care	MOSI: Transmission MISO: Not used

Mode	Description	Register configuration	Data pin usage
MRU	Master reception with unidirectional connection	MSTMOD = 1 RO = 1 BDEN = 0 BDOEN: Don't care	MOSI: Not used MISO: Reception
MTB	Master transmission with bidirectional connection	MSTMOD = 1 RO = 0 BDEN = 1 BDOEN = 1	MOSI: Transmission MISO: Not used
MRB	Master reception with bidirectional connection	MSTMOD = 1 RO = 0 BDEN = 1 BDOEN = 0	MOSI: Reception MISO: Not used
SFD	Slave full-duplex	MSTMOD = 0 RO = 0 BDEN = 0 BDOEN: Don't care	MOSI: Reception MISO: Transmission
STU	Slave transmission with unidirectional connection	MSTMOD = 0 RO = 0 BDEN = 0 BDOEN: Don't care	MOSI: Not used MISO: Transmission
SRU	Slave reception with unidirectional connection	MSTMOD = 0 RO = 1 BDEN = 0 BDOEN: Don't care	MOSI: Reception MISO: Not used
STB	Slave transmission with bidirectional connection	MSTMOD = 0 RO = 0 BDEN = 1 BDOEN = 1	MOSI: Not used MISO: Transmission
SRB	Slave reception with bidirectional connection	MSTMOD = 0 RO = 0 BDEN = 1 BDOEN = 0	MOSI: Not used MISO: Reception

Figure 20-7. A typical full-duplex connection

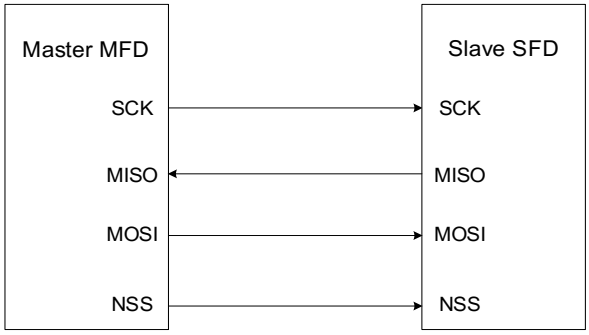


Figure 20-8. A typical simplex connection (Master: Receive, Slave: Transmit)

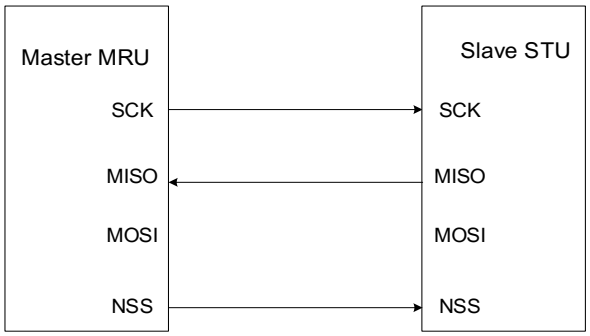


Figure 20-9. A typical simplex connection (Master: Transmit only, Slave: Receive)

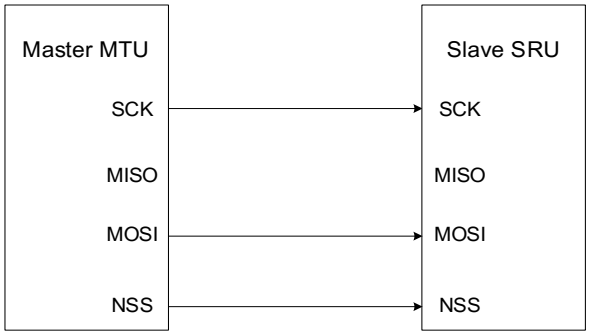
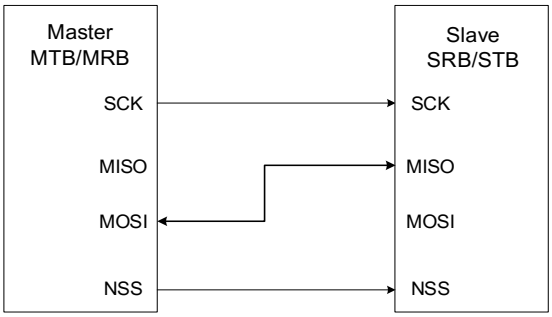


Figure 20-10. A typical bidirectional connection



Initialization sequence

Before transmitting or receiving data, application should follow the SPI initialization sequence described below:

1. If master mode is used, program the PSC[6:0] bits in SPI_CTL0 register to generate SCK with desired baud rate, otherwise, ignore this step.
2. Select SPI NSS (NSSSEL in the SPI_CTL1 register).
3. Configure the clock timing (CKPL and CKPH bits in the SPI_CTL0 register).
4. In master mode, if the user needs to configure the timing of the control signal, program the SPI_CKCFG0 register; otherwise, ignore this step.
5. In master mode, if the reception data sampling delay needs to be configured, program the SPI_CKCFG1 register; otherwise, ignore this step.
6. Program the frame format (LF bit in the SPI_CTL0 register).
7. Program data size (DZ[4:0] bits in the SPI_CTL1 register).

8. Program the NSS mode (SWNSSEN and NSSDRV bits in the SPI_CTL0 register) and NSS polarity (NSSPOL bit in the SPI_CTL0 register) according to the application's demand as described above in ***NSS function.***
9. If TI mode is used, set TMOD bit in SPI_CTL1 register, otherwise, ignore this step.
10. If NSSP mode is used, set NSSP bit in SPI_CTL1 register, otherwise, ignore this step.
11. In master mode, if using the hardware single-frame mode, set SINGMOD bit in SPI_CTL1 register, otherwise, ignore this step.
12. In master mode, if using the hardware continuous frame mode, configure the SPI_CCTL register, otherwise, ignore this step.
13. If using the master request mode, configure the SPI_MREQCFG register, otherwise, ignore this step.
14. Configure MSTMOD, RO, BDEN and BDOEN depending on the operation modes described in ***SPI operation modes*** section.
15. If Quad-SPI mode is used, set the QMOD bit in SPI_QCTL register. Ignore this step if Quad-SPI mode is not used.
16. Enable the SPI (set the SPIEN bit).

Basic transmission and reception sequence

Transmission sequence

After the initialization sequence, the SPI is enabled and stays at idle state. In master mode, the transmission starts when the application writes a data into the TXFIFO. In slave mode the transmission starts when SCK clock signal begins to toggle at SCK pin and NSS level is active, so application should ensure that data is already written into TXFIFO before the transmission starts in slave mode.

When SPI begins to send a data frame, it first loads this data frame from the data TXFIFO to the shift register and then begins to transmit the loaded data frame. After TFA flag is set, which means the TXFIFO has enough space to write data by APB bus, the application should write SPI_DATA register again if it has more data to transmit.

In master mode, software should write the next data into SPI_DATA register before the transmission of current data frame is completed if it desires to generate continuous transmission.

Reception sequence

After the last valid sample clock, the incoming data will be moved from shift register to the RXFIFO, and when RXFIFO has enough data to read by APB bus, the RFDR will be set. The application should read SPI_DATA register to get the received data and this will clear the RFDR flag automatically when RXFIFO has not enough data to read. In MRU and MRB modes, hardware continuously sends clock signal to receive the next data frame, while in full-duplex master mode (MFD), hardware only receives the next data frame when the TXFIFO not empty.

SPI operation sequence in different modes (Not Quad-SPI, TI mode, NSSP mode, Single-frame mode, Continuous-frame mode or Master request mode)

In full-duplex mode, either MFD or SFD, the RFDR and TFA flags should be monitored and then follow the sequences described above.

The transmission mode (MTU, MTB, STU or STB) is similar to the transmission sequence of full-duplex mode regardless of the RFDR and RXORERR bits.

The master reception mode (MRU or MRB) is different from the reception sequence of full-duplex mode. In MRU or MRB mode, after SPI is enabled, the SPI continuously generates SCK until the SPI is disabled. So the application should ignore the TFA flag and read out RXFIFO in time after the RFDR flag is set, otherwise a data overrun fault will occur.

The slave reception mode (SRU or SRB) is similar to the reception sequence of full-duplex mode regardless of the TFA flag.

SPI TI mode

SPI TI mode takes NSS as a special frame header flag signal and its operation sequence is similar to normal mode described above. The modes described above (MFD, MTU, MRU, MTB, MRB, SFD, STU, SRU, STB and SRB) are still supported in TI mode. While, in TI mode the CKPL bit, CKPH bit and SPI_CKCFG0 register take no effect and the SCK sample edge is falling edge.

Figure 20-11. Timing diagram of TI master mode with discontinuous transfer

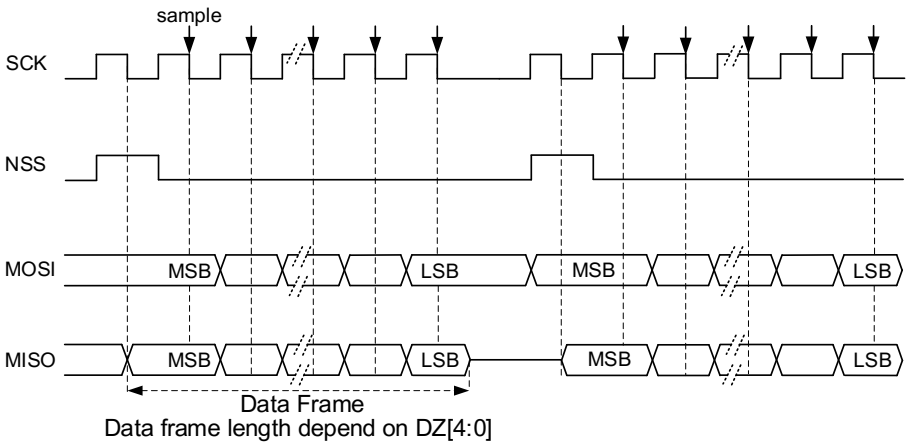
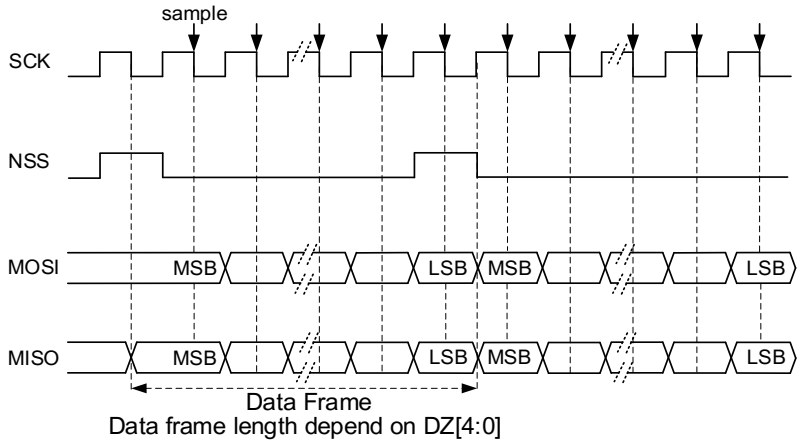
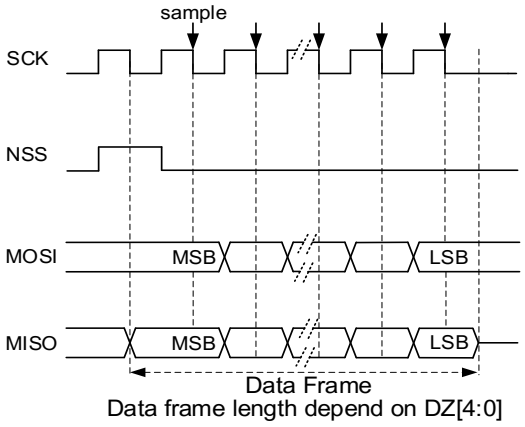


Figure 20-12. Timing diagram of TI master mode with continuous transfer



In master TI mode, SPI can perform continuous or non-continuous transfer. If the master writes SPI_DATA register fast enough, the transfer is continuous, otherwise non-continuous. In non-continuous transfer there is an extra header clock cycle before each data frame. While in continuous transfer, the extra header clock cycle only exists before the first byte and the following bytes' header clock is overlaid at the last bit of pervious bytes.

Figure 20-13. Timing diagram of TI slave mode



In slave TI mode, after the last rising edge of SCK in transfer, the slave begins to transmit the LSB bit of the last data byte, and after a half-bit time, the master begins to sample the line.

In slave mode, the slave also monitors the NSS signal and sets an error flag FERR if it detects an incorrect NSS behavior, for example: toggles at the middle bit of a byte.

NSS pulse mode operation sequence

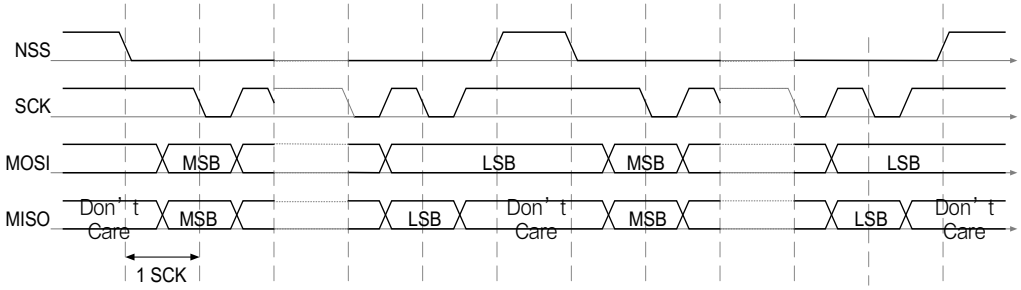
This function is controlled by NSSP bit in SPI_CTL1 register. In order to implement this function, several additional conditions must be met: configure the device to master mode, frame format should follow the normal SPI protocol, select the first clock transition as the data capture edge.

In summary, MSTMOD = 1, NSSP = 1, CKPH = 0.

When NSS pulse mode is enabled, a pulse duration of at least NSSNSS[7:0] + 1 SCK clock period is inserted between two successive data frames depending on the status of internal data TXFIFO. Multiple SCK clock cycle intervals are possible if the TXFIFO stays empty. This function is designed for single master-slave configuration for the slave to latch data.

In NSSP mode, the time interval between the NSS pulse edge and the SPI bus data validity can be flexibly configured using the SCKNSS and NSSSCK bit fields in the SPI_CKCFG0 register. The time interval from NSS active to SCK active is the NSSSCK configuration value minus 0.5 SCK cycles, while the time interval from SCK inactive to NSS inactive is the SCKNSS configuration value plus 0.5 SCK cycles. For example, with NSSSCK = 0, SCKNSS = 0, and NSSNSS = 0, the timing diagram for NSS pulse mode is as follows:

Figure 20-14. Timing diagram of NSS pulse with continuous transmit (NSSNSS = 0, NSSSCK = 0, SCKNSS = 0)



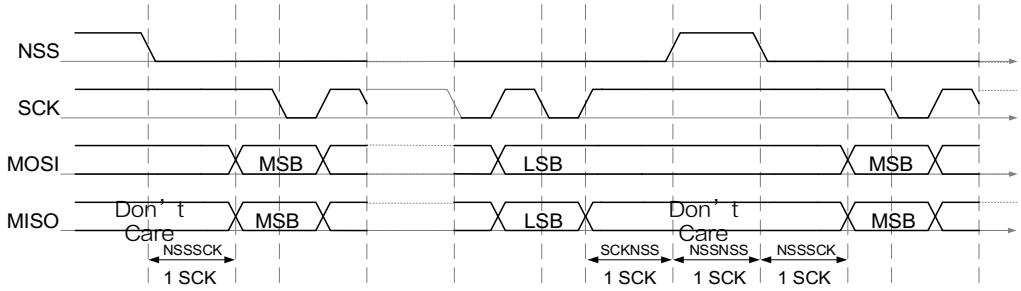
Hardware single-frame mode

Enabling the SINGMOD bit in the SPI_CTL1 register can activate the hardware single-frame mode. In single-frame mode, SPI master deactivates the NSS signal after each frame is sent. The NSS signal becomes active again when the next frame is sent.

Programming the SPI_CKCFG0 register allows for flexible control of signal line timing. Unlike NSSP mode, single-frame mode supports time intervals that are integer multiples of SCK clock cycles. The diagram below illustrates single-frame mode timing when NSSNSS = 0, NSSSCK = 0, and SCKNSS = 0.

Note: *The slave can not be configured in hardware single-frame mode.*

Figure 20-15. Timing diagram of single-frame mode with continuous transmit (NSSNSS = 0, NSSSCK = 0, SCKNSS = 0)

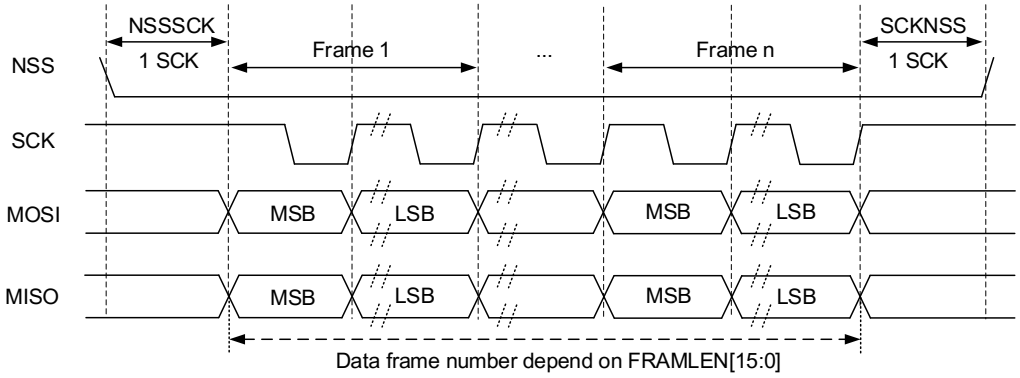


Hardware continuous-frame mode

In hardware continuous frame mode, the NSS signal remains active during the master data transmission process until the total number of transmitted frames reaches the length configured in the FRAMLEN[15:0] bit field. After a group of data has been sent, the next group of data can be sent continuously. Continuous frame mode can be enabled by setting the HCEN bit in the SPI_CCTL register. The SPI_CKCFG0 register can be configured to control signal timing in this mode. The diagram below illustrates continuous-frame mode timing when NSSNSS = 0, NSSSCK = 0, and SCKNSS = 0.

Note: *The slave can not be configured in hardware continue-frame mode.*

Figure 20-16. Timing diagram of continuous-frame mode with continuous transmit (NSSNSS = 0, NSSSCK = 0, SCKNSS = 0)



Quad-SPI mode operation sequence

The Quad-SPI mode is designed to control Quad-SPI flash.

In order to enter Quad-SPI mode, the software should first verify that the TFA bit is set and TRANS bit is cleared, then set QMOD bit in SPI_QCTL register. In Quad-SPI mode, BDEN, BDOEN, CRCEN, CRCNT, RO and LF in SPI_CTL0 register should be kept cleared and the data size is fixed at 32 bits by hardware and is not affected by the configuration of DZ[4:0], MSTMOD should be set to ensure that SPI is in master mode. SPIEN, PSC, CKPL and CKPH should be configured as desired.

Note:

1. Quad-SPI mode cannot be used together with special modes such as TI, NSSP, single-frame mode, or continuous -frame mode.
2. Quad-SPI mode does not support configuring control signal timing using the SPI_CKCFG0 register.
3. When the Quad SPI mode is enabled (QMOD = 1), the data size DZ[4:0] will be automatically set to 2b'11111 by the hardware. When switching back from Quad SPI mode to Single SPI mode (QMOD = 0), the software needs to reconfigure DZ[4:0] to the desired value.

There are two operation modes in Quad-SPI mode: quad write and quad read, decided by

QRD bit in SPI_QCTL register.

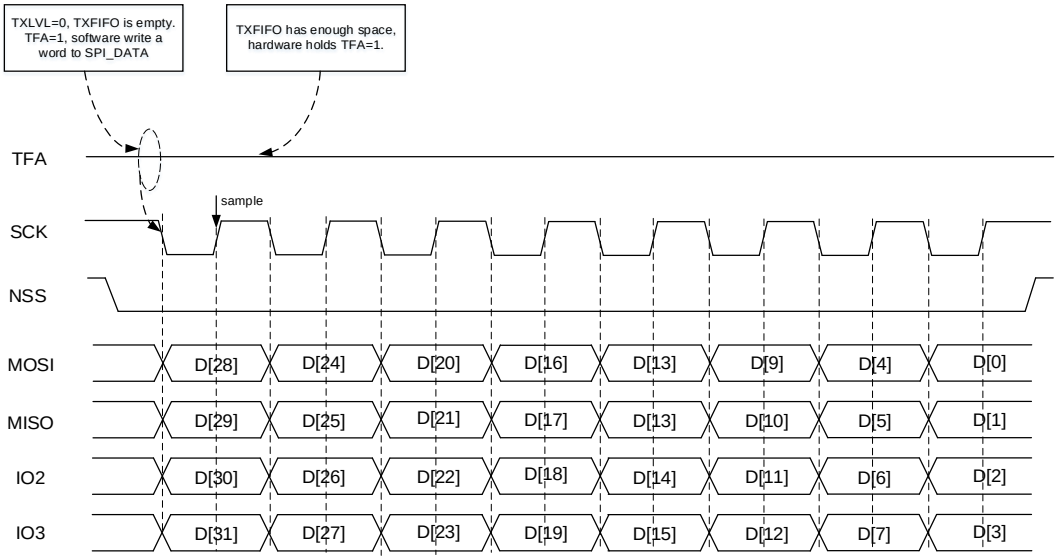
Quad write operation

SPI works in quad write mode when QMOD is set and QRD is cleared in SPI_QCTL register. In this mode, MOSI, MISO, IO2 and IO3 are all used as output pins. SPI begins to generate clock on SCK line and transmit data on MOSI, MISO, IO2 and IO3 as soon as data is written into SPI_DATA and SPIEN is set. Once SPI starts transmission, it always checks TFA status at the end of a frame and stops when condition is not met.

The operation flow for transmitting in quad mode:

1. Configure clock prescaler, clock polarity, phase, etc. in SPI_CTL0 and SPI_CTL1 based on your application requirements.
2. Set QMOD bit in SPI_QCTL register and then enable SPI by setting SPIEN in SPI_CTL0.
3. Write a word to SPI_DATA register and if TXFIFO has enough space, the TFA will remain set 1.
4. Wait until TFA is set by hardware again before writing the next data.

Figure 20-17. Timing diagram of quad write operation in Quad-SPI mode



Quad read operation

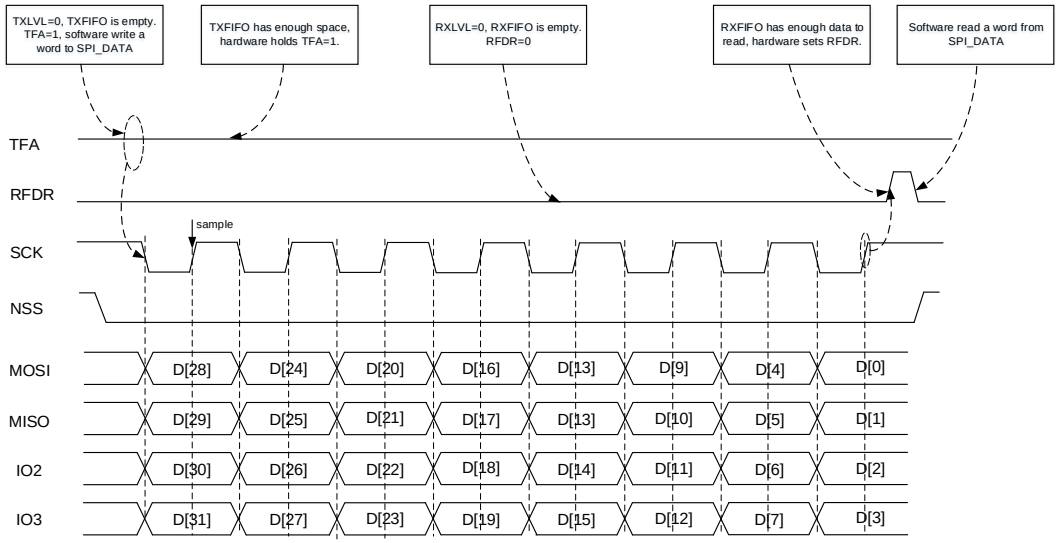
SPI works in quad read mode when QMOD and QRD are both set in SPI_QCTL register. In this mode, MOSI, MISO, IO2 and IO3 are all used as input pins. SPI begins to generate clock on SCK line as soon as a data is written into SPI_DATA and SPIEN is set. Writing data into SPI_DATA is only to generate SCK clocks, so the written data can be any value. Once SPI starts transmission, it always checks SPIEN and TFA status at the end of a frame and stops when condition is not met. So, dummy data should always be written into SPI_DATA to generate SCK.

The operation flow for receiving in quad mode is shown below:

1. Configure clock prescaler, clock polarity, phase, etc. in SPI_CTL0 and SPI_CTL1 register based on your application requirements.

2. Set QMOD and QRD bits in SPI_QCTL register and then enable SPI by setting SPIEN in SPI_CTL0 register.
3. Write an arbitrary word to SPI_DATA register.
4. Wait until the RFDR flag is set and read SPI_DATA to get the received data.
5. Write an arbitrary word to SPI_DATA to receive the next data.

Figure 20-18. Timing diagram of quad read operation in Quad-SPI mode



SPI master request

The SPI supports master request function, which users can use to control the start of SPI bus transfers. Users can enable it by setting the MREQEN bit in the SPI_MREQCFG register. The MREQ pin should only be configured as output in slave mode or input for master mode. In SPI master mode, the SPI will only start a new SPI bus transfer if the host request input is asserted. When the SPI is busy, the master request input is ignored. In SPI slave mode, the MREQ output pin will assert when data is available to be transmitted.

Note:

1. When the master request function is enabled with the MREQ pin, the NSSSEL cannot be 1.
2. For GD32A711 / A712 devices, under the condition where MREQSEL = 1 selects input trigger as master request source, the division factors for SPIDIV and PSC are constrained to predefined values corresponding to required SPI baud rates (see ***Table 20-6. SPI Master clock constraints when MREQSEL = 1***)

Table 20-6. SPI Master clock constraints when MREQSEL = 1

SCK frequency	Clock source	Clock source max frequency	SPIDIV	PSC
100K	CK_PCLK	80M	3	99
200K	CK_PCLK	80M	1	99
250K	CK_PCLK	80M	1	79
400K	CK_PCLK	80M	0	99
500K	CK_PCLK	80M	0	79

SCK frequency	Clock source	Clock source max frequency	SPIDIV	PSC
800K	CK_PCLK	80M	0	49
1M	CK_PCLK	80M	0	39
2M	CK_PCLK	80M	0	19
3M	CK_IRC48M	48M	0	7
4M	CK_PCLK	80M	0	9
5M	CK_PCLK	80M	0	7
6M	CK_IRC48M	48M	0	3
8M	CK_PCLK	80M	0	4
10M	CK_PCLK	80M	0	3
12M	CK_IRC48M	48M	0	1
20M	CK_PCLK	80M	0	1

SPI disabling sequence

Different sequences are used to disable the SPI in different operation modes:

MFD SFD

Wait until TXLVL[3:0]=0 and confirm TRANS=0. Then disable the SPI by clearing SPIEN bit. At last, read data until RXLVL[3:0]=0.

MTU MTB STU STB

Wait until TXLVL[3:0]=0 and confirm TRANS=0. Then disable the SPI by clearing SPIEN bit.

MRU MRB

The application can disable the SPI when it doesn't want to receive data, and then confirm the TRANS=0 and read data until RXLVL[3:0]=0.

Application need to disable the SPI before last data receive finished, suggest to disable the SPI once last data transaction start.

SRU SRB

The application can disable the SPI when it doesn't want to receive data, and then confirm the TRANS=0 and read data until RXLVL[3:0]=0.

TI mode

The disabling sequence of TI mode is the same as the sequences described above.

NSS pulse mode

The disabling sequence of NSSP mode is the same as the sequences described above.

Hardware single-frame mode

The disabling sequence of Single-frame mode is the same as the sequences described above.

Hardware continuous-frame mode

The disabling sequence of Continuous-frame mode is the same as the sequences described above.

Quad-SPI mode

Before leaving quad wire mode or disabling SPI, software should first check that TFA bit is set and TRANS bit is cleared, then the QMOD bit in SPI_QCTL register and SPIEN bit in SPI_CTL0 register are cleared.

20.3.11. DMA function

The DMA frees the application from data writing and reading process during transfer, to improve the system efficiency.

DMA function in SPI is enabled by setting DMATEN and DMAREN bits in SPI_CTL1 register. To use DMA function, application should first correctly configure DMA modules, then configure SPI module according to the initialization sequence, at last enable SPI.

After being enabled, if DMATEN is set, SPI will generate a DMA request each time when TFA = 1, then DMA will acknowledge to this request and write data into the SPI_DATA register automatically. If DMAREN is set, SPI will generate a DMA request each time when RFDR = 1, then DMA will acknowledge to this request and read data from the SPI_DATA register automatically.

NOTE: During SPI transmission, a non-full TXFIFO generates fast and consecutive DMA requests. If the data in the DMA channel FIFO is not yet sufficient for one complete peripheral transfer, the FIFO exception flag (FEEIF) will be set. This is the normal behavior resulting from the interaction between the SPI request mechanism and the DMA FIFO prefetching: the DMA defers its response to the current request and resumes automatically once sufficient data is available. The correctness of data transfer is not affected, and no software handling is required (it is recommended to keep this interrupt disabled).

20.3.12. CRC function

SPI supports hardware CRC transmission functionality, and the CRC calculation uses the polynomial defined in SPI_CRCPOLY register.

Application can enable the CRC function by setting CRCEN bit in SPI_CTL0 register. The CRC calculators continuously calculate CRC for each bit transmitted on lines, and the calculated CRC values can be read from SPI_TCRC register.

To transmit the calculated CRC value, application should set the CRCNT bit in SPI_CTL0 register after the last data is written to the TXFIFO. In full-duplex mode (MFD or SFD), when

the SPI hardware sends the CRC, the software can treat the received data frame as the CRC value and perform software CRC verification, or discard the received data frame. When performing software CRC calculation on the data at the receiving end, the initial value of the CRC needs to be set to 0. After receiving the CRC frame and completing the verification, the software needs to reset the CRC calculation value to 0.

A CRC-format transaction usually takes one more data frame to communicate at the end of data sequence. If DMA function is enabled, the counter for the SPI transmission DMA channel has to be set to the number of data frames to transmit excluding the CRC frame, additionally, after the configured data frames are transmitted, the hardware will automatically send one CRC data frame without requiring the software to set CRCNT. In full duplex mode (MFD or SFD), the counter of the reception DMA channel can be set to the number of data frames to receive including the CRC. User can perform software CRC verification on the received data or discard the received CRC data.

Note: When SPI is in slave mode and CRC function is enable, the CRC calculator is sensitive to input SCK clock whether SPI is enable or not. The software must enable CRC only when the clock is stable to avoid wrong CRC calculation.

When SPI is configured as slave mode and the CRC function is used, the CRC calculation will still be performed even when the NSS pin is inactive (when the NSS signal is inactive, as long as there is a clock pulse on the SCK pin, the CRC calculation will continue). This situation occurs when the master device alternately communicates with multiple slave devices. At this time, it is recommended to restart the CRC function when the NSS signal is active. When the slave device is not selected (NSS signal is inactive) and switches to being selected as a new slave device (NSS signal is active), in order to maintain the synchronization of the next CRC calculation results between the master and slave devices, the CRC values at both ends should be cleared. It is recommended to clear the CRC value according to the following steps:

1. Disable the SPI module (SPIEN=0);
2. Clear the CRCEN bit (CRCEN=0);
3. Set the CRCEN bit (CRCEN=1);
4. Enable the SPI module (SPIEN=1).

20.3.13. SPI interrupts

Status flags

- **Transmission FIFO available flag (TFA)**

This bit is set when the TXFIFO has enough space for APB bus write access, the software can write the next data to the TXFIFO by writing the SPI_DATA register.

- **Reception FIFO data ready flag (RFDR)**

This bit is set when RXFIFO is not empty and has enough data to read, the software can read the data by reading the SPI_DATA register.

- **SPI transmitting ongoing flag (TRANS)**

TRANS is a status flag to indicate whether the transfer is ongoing or not. It is set and cleared by hardware and not controlled by software. This flag doesn't generate any interrupt.

■ **SPI timeout flag (TOF)**

TOF is a status flag that indicates whether the transmission is complete by monitoring whether the duration of the SPI bus IDLE state exceeds TOCFG. It is set by hardware and cleared by software by writing 0.

Error conditions

■ **Configuration fault error (CONFERR)**

CONFERR is an error flag in master mode. In NSS hardware mode and the NSSDRV is not enabled, the CONFERR is set when the NSS level is active. In NSS software mode, the CONFERR is set when the NSSPOL and SWNSS are either set to 0 or both set to 1. When the CONFERR is set, the SPIEN bit and the MSTMOD bit are cleared by hardware, the SPI is disabled and the device is forced into slave mode.

The SPIEN and MSTMOD bit are write protection until the CONFERR is cleared. The CONFERR bit of the slave cannot be set. In multi-master configurations, the device can enter slave mode when CONFERR is set to 1, which indicates a multi-master conflict in system control.

■ **Transmission underrun error (TXURERR)**

This situation occurs when the valid SCK signal starts in slave transmission mode and the TXFIFO is empty. When the SPI slave hardware sends a CRC frame, the TXURERR flag will not be set.

■ **Reception overrun error (RXORERR)**

The RXORERR is set, if RXFIFO is full and still newly incoming data. This means the RXFIFO has not enough space to store this received data. The receive RXFIFO contents won't be covered with the newly incoming data, so the newly incoming data is lost.

■ **Format error (FERR)**

In slave TI mode, the slave also monitors the NSS signal and set an error flag if it detects an incorrect NSS behavior, for example: toggles at the middle bit of a byte.

Table 20-7. SPI interrupt requests

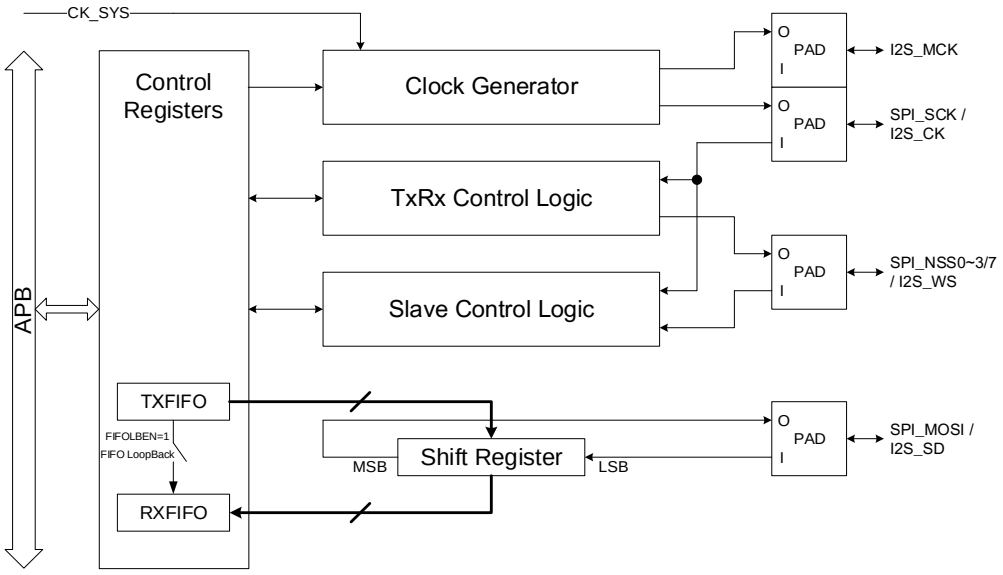
Flag	Description	Clear method	Interrupt enable bit
TFA	Transmit FIFO available	Write SPI_DATA register and TXFIFO is full.	TFAIE
RFDR	Receive FIFO data ready	Read SPI_DATA register and RXFIFO is empty.	RFDRIE
TOF	Idle state timeout	Write 0 to TOF bit	TOIE
CONFERR	Configuration fault error	Read or write SPI_STAT register,	ERRIE

Flag	Description	Clear method	Interrupt enable bit
		then write SPI_CTL0 register.	
TXURERR	Tx underrun error	Read SPI_STAT register.	
RXORERR	Rx overrun error	Read SPI_DATA register, then read SPI_STAT register.	
FERR	TI mode format error	Read SPI_STAT register	

20.4. I2S function overview

20.4.1. I2S block diagram

Figure 20-19. Block diagram of I2S



There are five sub modules to support I2S function, including control registers, clock generator, master control logic, slave control logic and shift register. All the user configuration registers are implemented in the control registers module, including the TXFIFO and RXFIFO. The clock generator is used to produce I2S communication clock in master mode. The master control logic is implemented to generate the I2S_WS signal and control the communication in master mode. The slave control logic is implemented to control the communication in slave mode according to the received I2S_CK and I2S_WS. The shift register handles the serial data transmission and reception on I2S_SD.

20.4.2. I2S signal description

There are four pins on the I2S interface, including I2S_CK, I2S_WS, I2S_SD and I2S_MCK. I2S_CK is the serial clock signal, which shares the same pin with SPI_SCK. I2S_WS is the frame control signal, which shares the same pin with SPI_NSS. I2S_SD is the serial data signal, which shares the same pin with SPI_MOSI. I2S_MCK is the master clock signal. It

produces a frequency rate equal to 256 x Fs, and Fs is the audio sampling frequency.

20.4.3. I2S audio standards

The I2S audio standard is selected by the I2SSTD bits in the SPI_I2SCTL register. Four audio standards are supported, including I2S Philips standard, MSB justified standard, LSB justified standard, and PCM standard. All standards except PCM handle audio data time-multiplexed on two channels (the left channel and the right channel). For these standards, the I2S_WS signal indicates the channel side. For PCM standard, the I2S_WS signal indicates frame synchronization information.

The data length and the channel length are configured by the DTLEN bits and CHLEN bit in the SPI_I2SCTL register. Since the channel length must be greater than or equal to the data length, four packet types are available. They are 16-bit data packed in 16-bit frame, 16-bit data packed in 32-bit frame, 24-bit data packed in 32-bit frame, and 32-bit data packed in 32-bit frame.

For all standards and packet types, the most significant bit (MSB) is always sent first. For all standards based on two channels time-multiplexed, the channel left is always sent first followed by the channel right.

I2S Philips standard

For I2S Philips standard, I2S_WS and I2S_SD are updated on the falling edge of I2S_CK. The timing diagrams for each configuration are shown below.

Figure 20-20. I2S Philips standard timing diagram (DTLEN=00, CHLEN=0, CKPL=0)

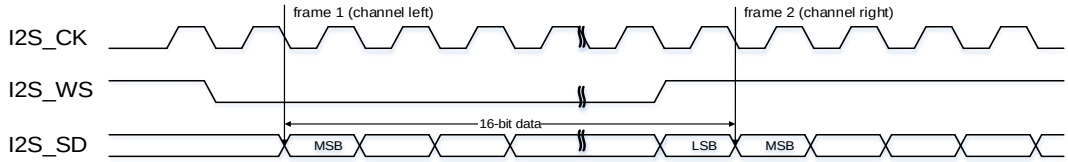
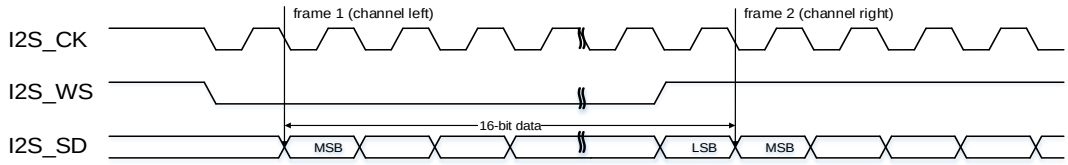


Figure 20-21. I2S Philips standard timing diagram (DTLEN=00, CHLEN=0, CKPL=1)



When the packet type is 16-bit data packed in 16-bit frame, only one write or read operation to or from the SPI_DATA register is needed to complete the transmission of a frame.

Figure 20-22. I2S Philips standard timing diagram (DTLEN=10, CHLEN=1, CKPL=0)

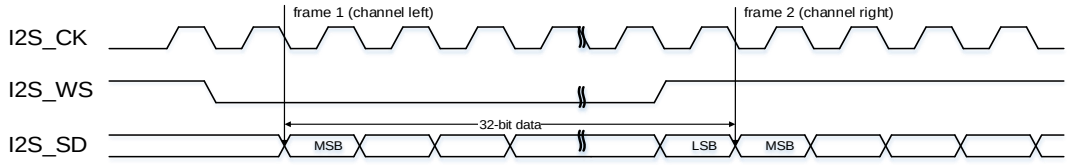


Figure 20-23. I2S Philips standard timing diagram (DTLEN=10, CHLEN=1, CKPL=1)

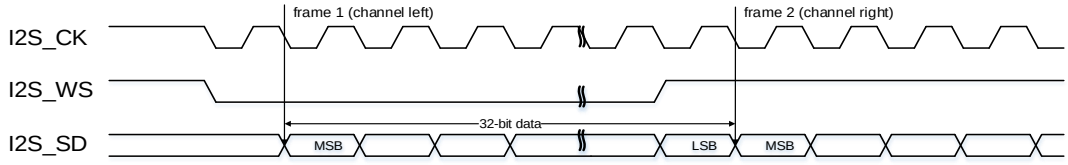


Figure 20-24. I2S Philips standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0)

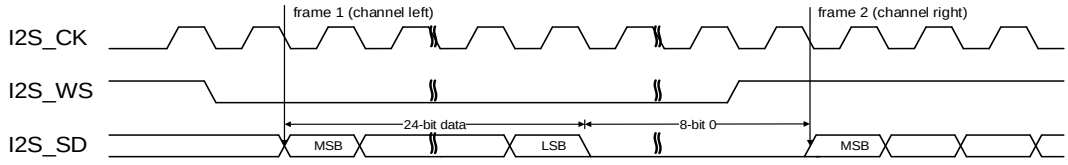


Figure 20-25. I2S Philips standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1)

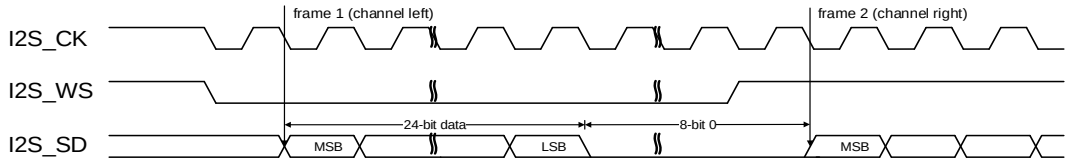


Figure 20-26. I2S Philips standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0)

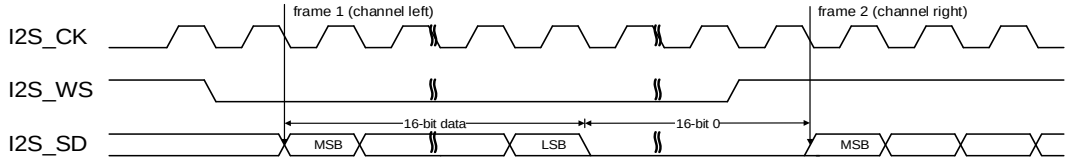
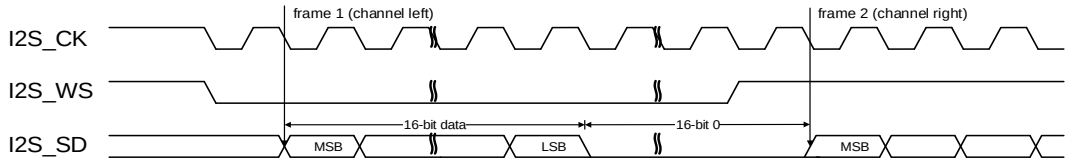


Figure 20-27. I2S Philips standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1)



MSB justified standard

For MSB justified standard, I2S_WS and I2S_SD are updated on the falling edge of I2S_CK. The SPI_DATA register is handled in the exactly same way as that for I2S Philips standard. The timing diagrams for each configuration are shown below.

Figure 20-28. MSB justified standard timing diagram (DTLEN=00, CHLEN=0, CKPL=0)

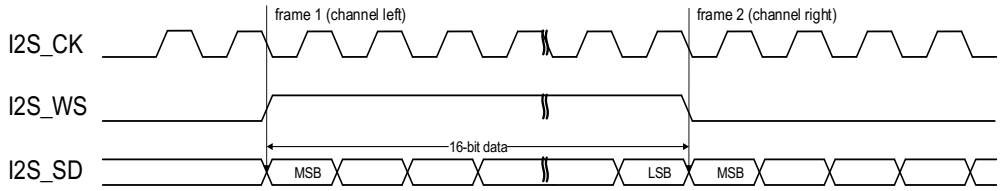


Figure 20-29. MSB justified standard timing diagram (DTLEN=00, CHLEN=0, CKPL=1)

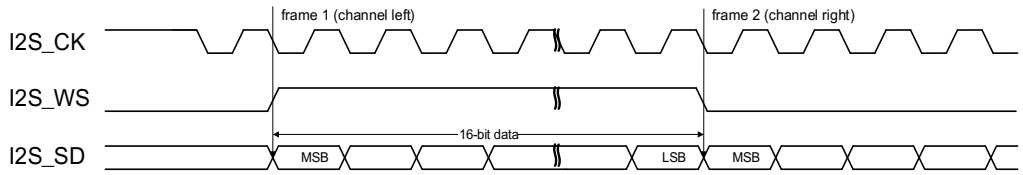


Figure 20-30. MSB justified standard timing diagram (DTLEN=10, CHLEN=1, CKPL=0)

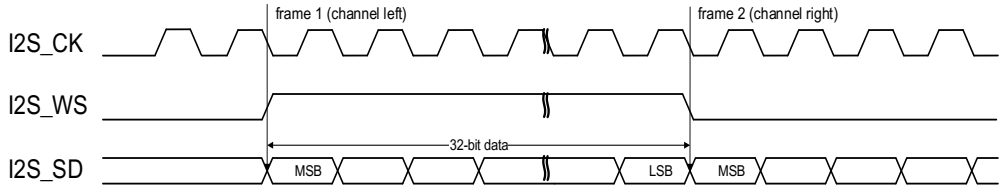


Figure 20-31. MSB justified standard timing diagram (DTLEN=10, CHLEN=1, CKPL=1)

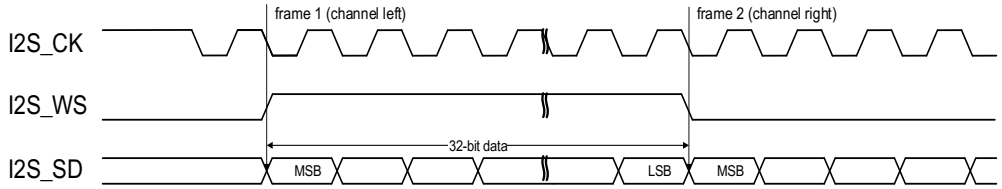


Figure 20-32. MSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0)

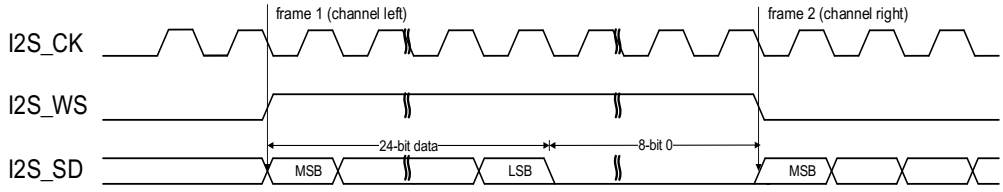


Figure 20-33. MSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1)

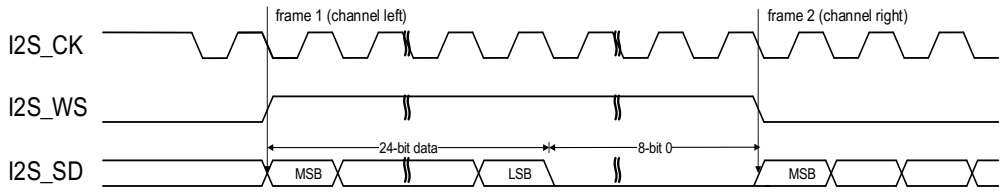


Figure 20-34. MSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0)

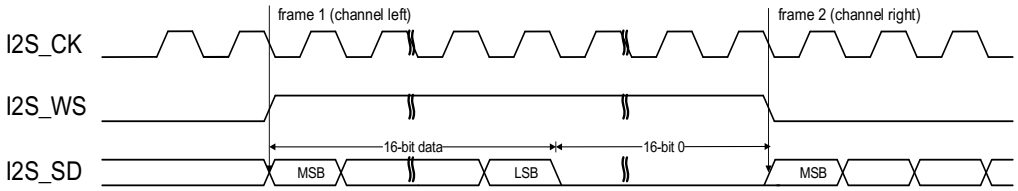
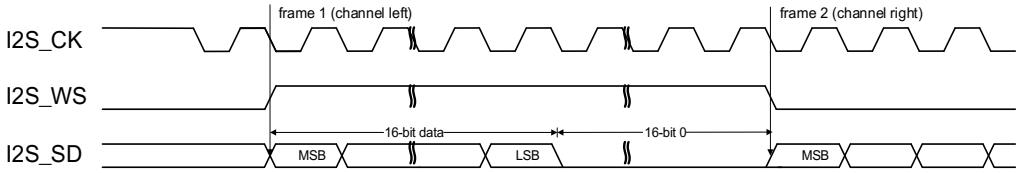


Figure 20-35. MSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1)



LSB justified standard

For LSB justified standard, I2S_WS and I2S_SD are updated on the falling edge of I2S_CK. In the case that the channel length is equal to the data length, LSB justified standard and MSB justified standard are exactly the same. In the case that the channel length is greater than the data length, the valid data is aligned to LSB for LSB justified standard while the valid data is aligned to MSB for MSB justified standard. The timing diagrams for the cases that the channel length is greater than the data length are shown below.

Figure 20-36. LSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=0)

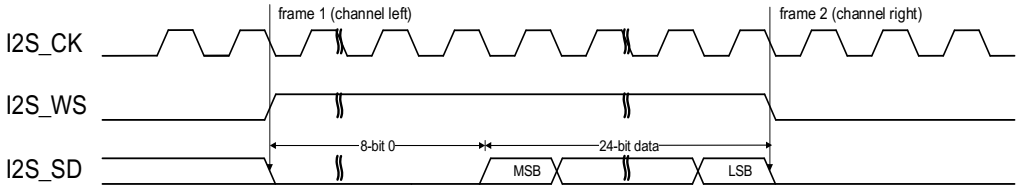


Figure 20-37. LSB justified standard timing diagram (DTLEN=01, CHLEN=1, CKPL=1)

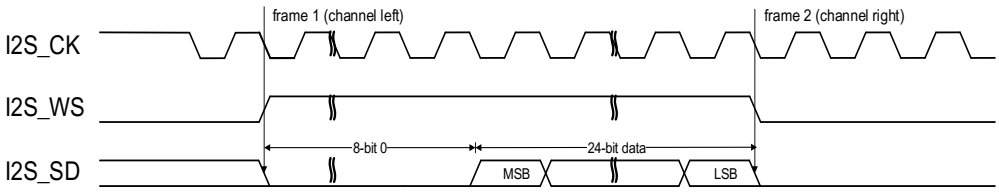


Figure 20-38. LSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=0)

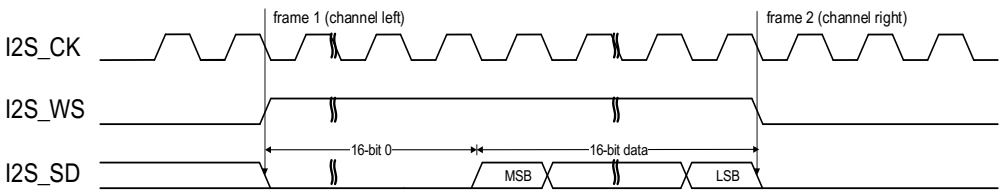
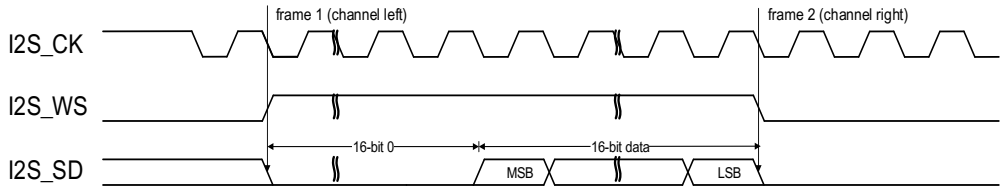


Figure 20-39. LSB justified standard timing diagram (DTLEN=00, CHLEN=1, CKPL=1)



PCM standard

For PCM standard, I2S_WS and I2S_SD are updated on the rising edge of I2S_CK, and the I2S_WS signal indicates frame synchronization information. Both the short frame synchronization mode and the long frame synchronization mode are available and configurable using the PCMSMOD bit in the SPI_I2SCTL register. The SPI_DATA register is handled in the exactly same way as that for I2S Philips standard. The timing diagrams for each configuration of the short frame synchronization mode are shown below.

Figure 20-40. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=0)

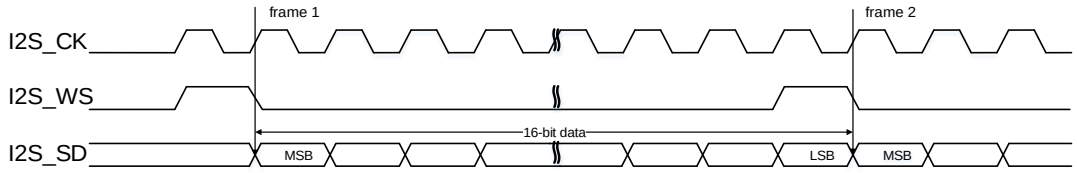


Figure 20-41. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=1)

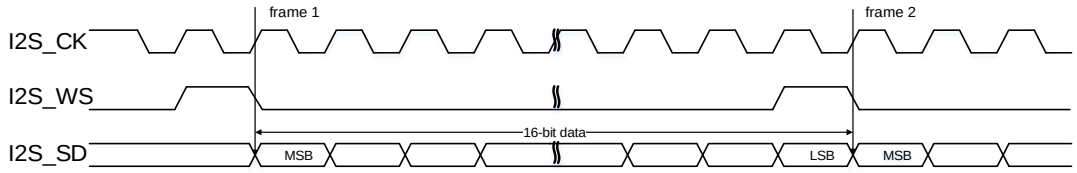


Figure 20-42. PCM standard short frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=0)

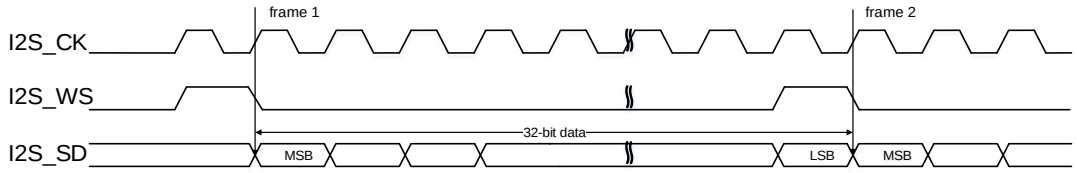


Figure 20-43. PCM standard short frame synchronization mode timing diagram

(DTLEN=10, CHLEN=1, CKPL=1)

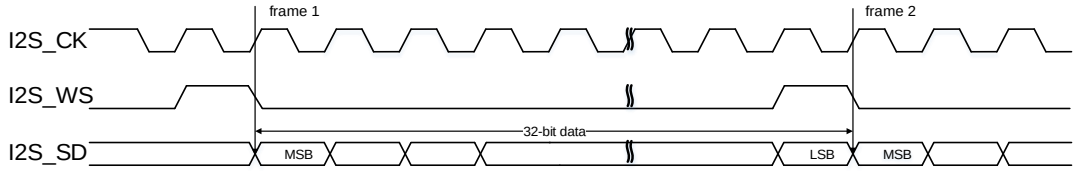


Figure 20-44. PCM standard short frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=0)

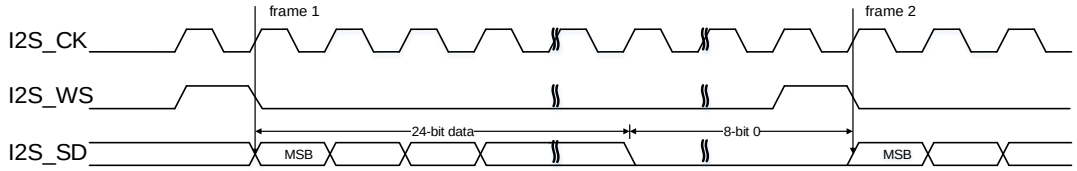


Figure 20-45. PCM standard short frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=1)

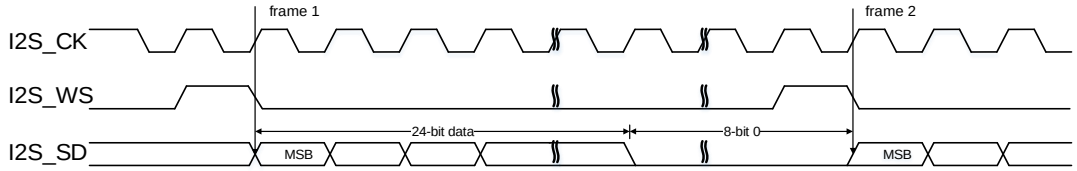


Figure 20-46. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=0)

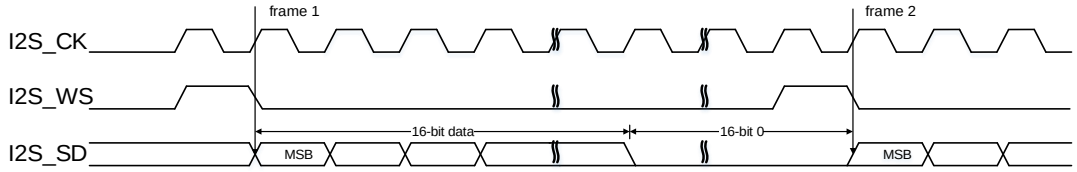
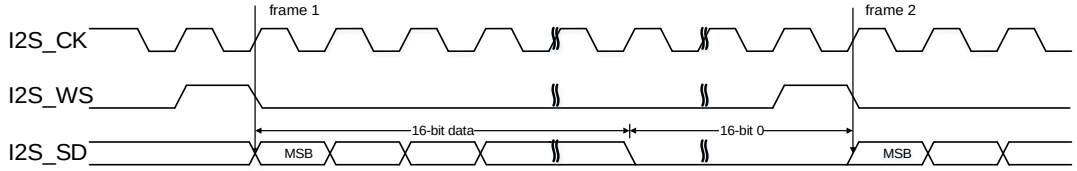


Figure 20-47. PCM standard short frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=1)



The timing diagrams for each configuration of the long frame synchronization mode are shown below.

Figure 20-48. PCM standard long frame synchronization mode timing diagram

(DTLEN=00, CHLEN=0, CKPL=0)

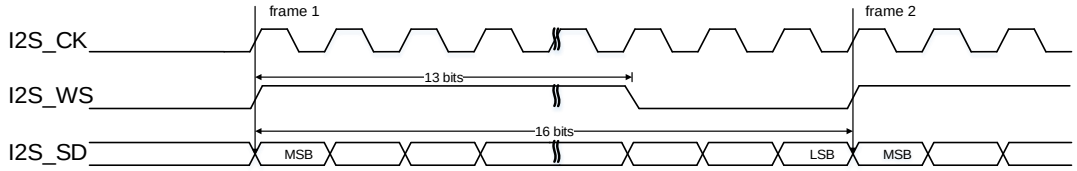


Figure 20-49. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=0, CKPL=1)

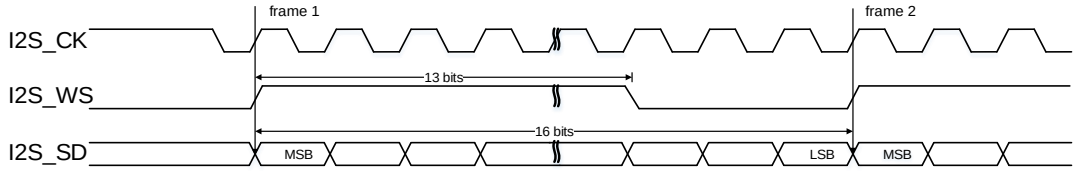


Figure 20-50. PCM standard long frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=0)

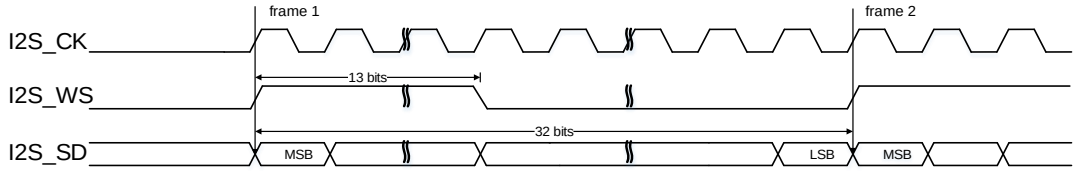


Figure 20-51. PCM standard long frame synchronization mode timing diagram (DTLEN=10, CHLEN=1, CKPL=1)

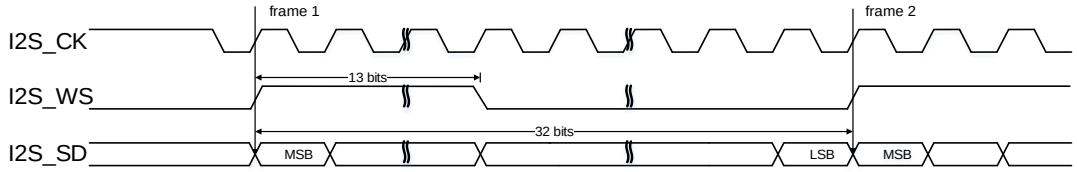


Figure 20-52. PCM standard long frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=0)

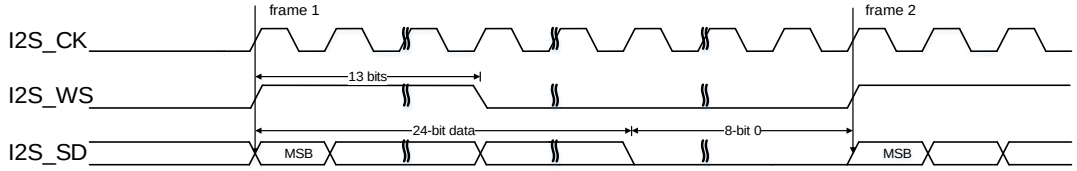


Figure 20-53. PCM standard long frame synchronization mode timing diagram (DTLEN=01, CHLEN=1, CKPL=1)

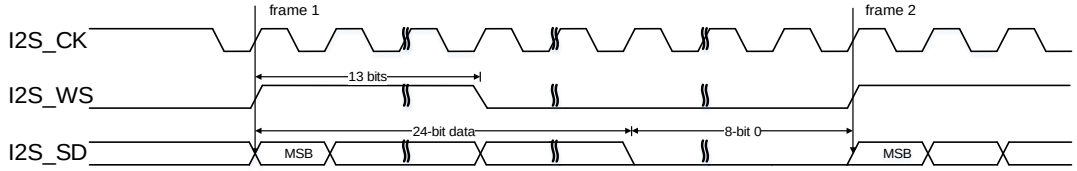


Figure 20-54. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=0)

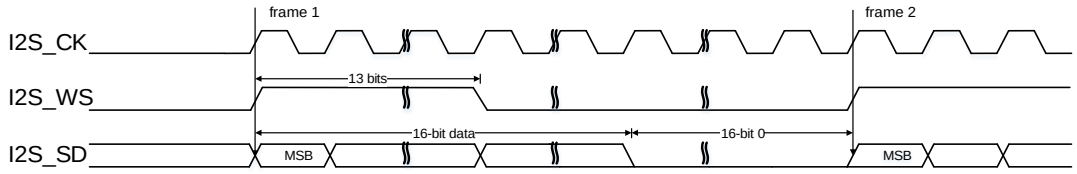
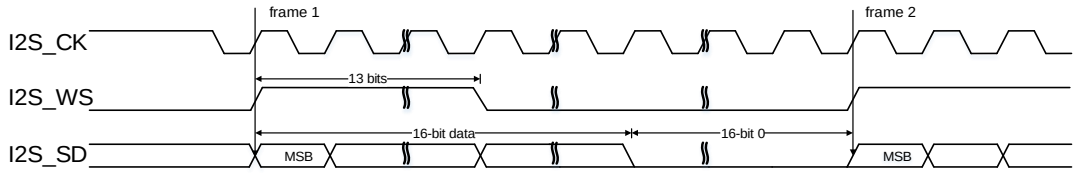
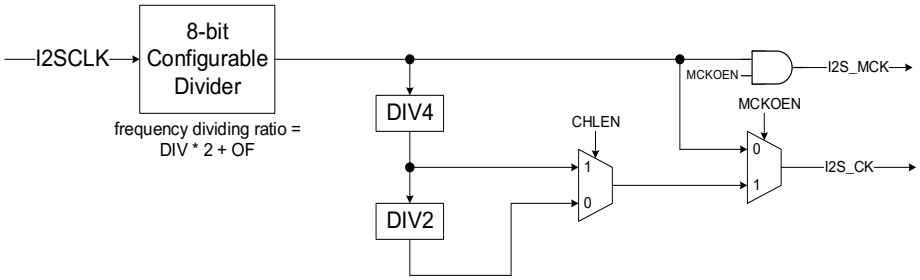


Figure 20-55. PCM standard long frame synchronization mode timing diagram (DTLEN=00, CHLEN=1, CKPL=1)



20.4.4. I2S clock

Figure 20-56. Block diagram of I2S clock generator



The block diagram of I2S clock generator is shown as *Figure 20-56. Block diagram of I2S clock generator*. The I2S interface clocks are configured by the DIV bits, the OF bit, the MCKOEN bit in the SPI_I2SPSC register and the CHLEN bit in the SPI_I2SCTL register. The source clock is the system clock(CK_SYS). The I2S bitrate can be calculated by the formulas shown in *Table 20-8. I2S bitrate calculation formulas*.

Table 20-8. I2S bitrate calculation formulas

MCKOEN	CHLEN	Formula
0	0	I2SCLK / (DIV * 2 + OF)
0	1	I2SCLK / (DIV * 2 + OF)
1	0	I2SCLK / (8 * (DIV * 2 + OF))
1	1	I2SCLK / (4 * (DIV * 2 + OF))

Note: The configuration value of the I2S serial clock must be set to less than 1/6 of the PCLK clock (excluding 1/6).

The relationship between audio sampling frequency (Fs) and I2S bitrate is defined by the following formula:

Fs = I2S bitrate / (number of bits per channel * number of channels)

So, in order to get the desired audio sampling frequency, the clock generator needs to be configured according to the formulas listed in *Table 20-9. Audio sampling frequency calculation formulas*.

Table 20-9. Audio sampling frequency calculation formulas

MCKOEN	CHLEN	Formula
0	0	I2SCLK / (32 * (DIV * 2 + OF))
0	1	I2SCLK / (64 * (DIV * 2 + OF))
1	0	I2SCLK / (256 * (DIV * 2 + OF))
1	1	I2SCLK / (256 * (DIV * 2 + OF))

Note: When the audio sampling rate exceeds 96K in master receive mode, CHLEN must be configured as 0.

20.4.5. Operation

Operation modes

The operation mode is selected by the I2SOPMOD bits in the SPI_I2SCTL register. There are four available operation modes, including master transmission mode, master reception mode, slave transmission mode, and slave reception mode. The direction of I2S interface signals for each operation mode is shown in the *Table 20-10. Direction of I2S interface signals for each operation mode*.

Table 20-10. Direction of I2S interface signals for each operation mode

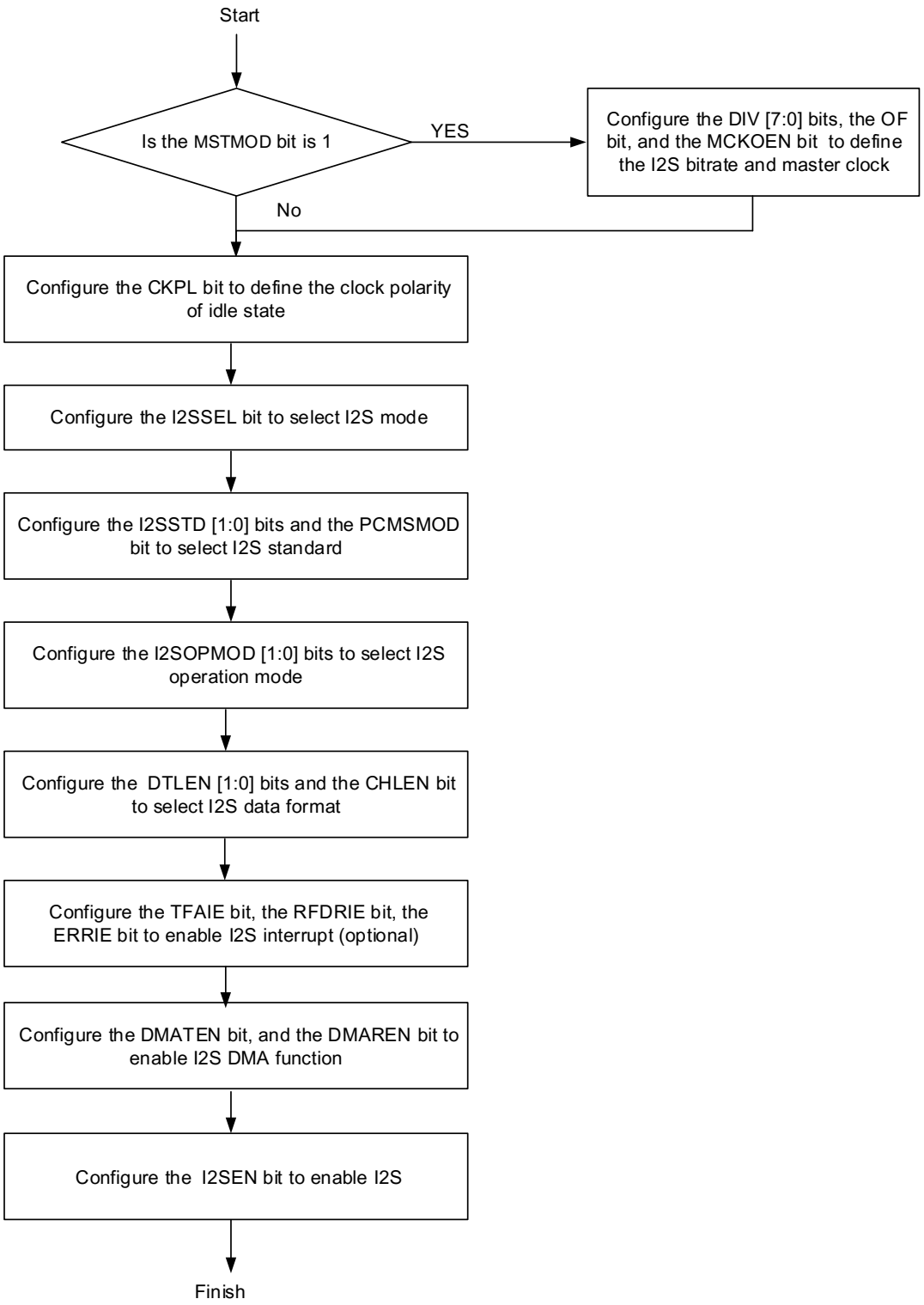
Operation mode	I2S_MCK	I2S_CK	I2S_WS	I2S_SD
Master transmission	Output or NU(1)	Output	Output	Output
Master reception	Output or NU(1)	Output	Output	Input
Slave transmission	Input or NU(1)	Input	Input	Output
Slave reception	Input or NU(1)	Input	Input	Input

(1) NU means the pin is not used by I2S and can be used by other functions.

I2S initialization sequence

I2S initialization sequence shown as below *Figure 20-57. I2S initialization sequence*.

Figure 20-57. I2S initialization sequence



I2S master transmission sequence

The TFA flag is used to control the transmission sequence. As is mentioned before, the TFA flag indicates that the TXFIFO is available, and an interrupt will be generated if the TFAIE bit

in the SPI_CTL1 register is set. At the beginning, the TXFIFO is empty (TFA is high) and no transmission sequence is processing in the shift register. When a half word or word is written to the SPI_DATA register, the data is transferred from the TXFIFO to the shift register immediately. At the moment, the transmission sequence begins.

The data is parallel loaded into the shift register, and shifted out serially to the I2S_SD pin, MSB first. The next data should be written to the SPI_DATA register, when the TFA flag is high. If TXFIFO has enough space for write, the TFA flag remains high. Software can write audio data continuously to the SPI_DATA register.

For all standards except PCM, the I2SCH flag is used to distinguish which channel side the data to transfer belongs to. At the beginning, the I2SCH flag is low, indicating the left channel data should be written to the SPI_DATA register.

In order to disable I2S, it is mandatory to clear the I2SEN bit after the TXLVL[3:0] equal to 0 and the TRANS flag is low.

Note: In I2S mode, the setting condition of TFA flag depends on whether the remaining space of TXFIFO is larger than DTLEN[1:0].

I2S master reception sequence

The RFDR flag is used to control the reception sequence. As is mentioned before, the RFDR flag indicates the RXFIFO data is ready, and an interrupt will be generated if the RFDRIE bit in the SPI_CTL1 register is set. The reception sequence begins immediately when the I2SEN bit in the SPI_I2SCTL register is set. At the beginning, the RXFIFO is empty (RFDR is low). When a reception sequence finishes, the received data in the shift register is loaded into the RXFIFO (RFDR goes high). The data should be read from the SPI_DATA register, when the RFDR flag is high. After reading all the data in RXFIFO, the RFDR flag goes low. It is mandatory to read the SPI_DATA register before the RXFIFO is full. Otherwise, reception overrun error occurs. The RXORERR flag is set and an interrupt may be generated if the ERRIE bit in the SPI_CTL1 register is set. In this case, it is necessary to disable and then enable I2S before resuming the communication.

For all standards except PCM, the I2SCH flag is used to distinguish the channel side which the received data belongs to.

The master reception disable sequence is as follows:

1. Wait for the second last RFDR.
2. Then wait one I2S clock cycle.
3. Clear the I2SEN bit.

I2S slave transmission sequence

The transmission sequence in slave mode is similar to that in master mode. The difference between them is described below.

In slave mode, the slave has to be enabled before the external master starts the

communication. The transmission sequence begins when the external master sends the clock and when the I2S_WS signal requests the transfer of data. The data has to be written to the SPI_DATA register before the master initiates the communication. Software should write the next audio data into SPI_DATA register before the current data finishes. Otherwise, transmission underrun error occurs. The TXURERR flag is set and an interrupt may be generated if the ERRIE bit in the SPI_CTL1 register is set. In this case, it is mandatory to disable and enable I2S to resume the communication. In slave mode, I2SCH is sensitive to the I2S_WS signal coming from the external master.

In order to disable I2S, it is mandatory to clear the I2SEN bit after the TXLVL[3:0] equal to 0 and the TRANS flag is low.

I2S slave reception sequence

The reception sequence in slave mode is similar to that in master mode. The differences between them are described below.

In slave mode, the slave has to be enabled before the external master starts the communication. The reception sequence begins when the external master sends the clock and when the I2S_WS signal indicates a start of the data transfer. In slave mode, I2SCH is sensitive to the I2S_WS signal coming from the external master.

In order to disable I2S, it is mandatory to clear the I2SEN bit immediately after receiving the last RFDR.

20.4.6. DMA function

DMA function is similar to SPI mode. The difference is that I2S generate DMA requests that depend on DTLEN[1:0].

The only difference is that the CRC function is not available in I2S mode.

20.4.7. I2S interrupts

Status flags

There are four status flags implemented in the SPI_STAT register, including TFA, RFDR, TRANS and I2SCH. The user can use them to fully monitor the state of the I2S bus.

- **Transmission available flag (TFA)**

This bit is set when the TXFIFO has enough space for APB bus write access, the software can write the next data to the TXFIFO by writing the SPI_DATA register.

- **Reception FIFO not empty flag (RFDR)**

This bit is set when RXFIFO is not empty and has enough data to read, the software can read the data by reading the SPI_DATA register.

■ **I2S transmitting ongoing flag (TRANS)**

TRANS is a status flag to indicate whether the transfer is ongoing or not. It is set and cleared by hardware and not controlled by software. This flag will not generate any interrupt.

■ **I2S channel side flag (I2SCH)**

This flag indicates the channel side information of the current transfer and has no meaning in PCM mode. This flag will not generate any interrupt.

Error conditions

There are three error flags:

■ **Transmission underrun error (TXURERR)**

This situation occurs when the valid SCK signal starts in slave transmission mode and the TXFIFO is empty.

■ **Reception overrun error (RXORERR)**

This situation occurs when the RXFIFO is full and a newly incoming data has been completely received. When overrun occurs, the data in RXFIFO is not updated and the newly incoming data is lost.

■ **Format error (FERR)**

In slave I2S mode, the I2S monitors the I2S_WS signal and an error flag will be set if I2S_WS toggles at an unexpected position.

I2S interrupt events and corresponding enabled bits are summed up in the *Table 20-11. I2S interrupt*.

Table 20-11. I2S interrupt

Interrupt flag	Description	Clear method	Interrupt enable bit
TFA	Transmit FIFO available	Write SPI_DATA register and TXFIFO is full.	TFAIE
RFDR	Receive FIFO data ready	Read SPI_DATA register and RXFIFO is empty.	RFDRIE
TXURERR	Transmission underrun error	Read SPI_STAT register	ERRIE
RXORERR	Reception overrun error	Read SPI_DATA register and then read SPI_STAT register.	
FERR	I2S format error	Read SPI_STAT register	

20.5. Register definition

SPI0 base address: 0x4001 3000
SPI1 base address: 0x4001 7800
SPI2 base address: 0x4000 E400
SPI3 base address: 0x4000 E800
SPI4 / I2S4 base address: 0x4001 4400
SPI5 / I2S5 base address: 0x4001 4800

Table 20-12. SPI register overview

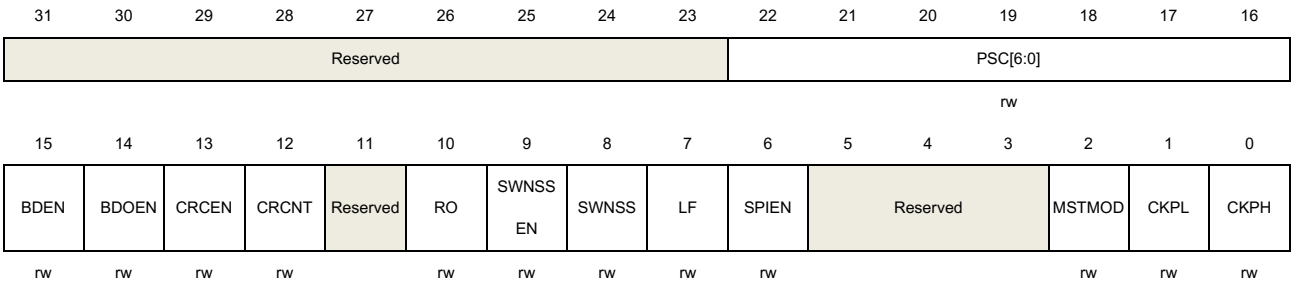
Register Name	Symbol	Address Offset	Reset Value
Control register 0	SPI_CTL0	0x00	0x0000 0000
Control register 1	SPI_CTL1	0x04	0x0100 0700
Status register	SPI_STAT	0x08	0x0000 0002
Data register	SPI_DATA	0x0C	0x0000 0000
CRC polynomial register	SPI_CRCPOLY	0x10	0x0000 0107
Transmit CRC register	SPI_TCRC	0x18	0x0000 0000
I2S control register	SPI_I2SCTL	0x1C	0x0000 0000
I2S clock prescaler register	SPI_I2SPSC	0x20	0x0000 0002
SPI Clock configuration register	SPI_CCR	0x24	0x0000 0000
Master request configuration register	SPI_MREQCFG	0x28	0x0000 0000
Clock configuration register 0	SPI_CKCFG0	0x2C	0x0000 0000
Clock configuration register 1	SPI_CKCFG1	0x30	0x0000 1003
Continuouscontrol register	SPI_CCTL	0x34	0x0000 0000
NSS polarity control register	SPI_ POLCTL	0x38	0x0000 0000
Quad-SPI mode control register	SPI_ QCTL	0x80	0x0000 0000

20.5.1. Control register 0 (SPI_CTL0)

Address offset: 0x00
Reset value: 0x0000 0000

This register can be accessed by word (32-bit).

This register has no meaning in I2S mode.



Bits	Fields	Descriptions
31:23	Reserved	Must be kept at reset value.

22:16	PSC[6:0]	Master clock prescaler selection The SPI master clock is CK_SPI / (2 * (PCS[6:0] + 1)) 0000000: CK_SPI/2 0000001: CK_SPI/4 0000010: CK_SPI/6 0000011: CK_SPI/8 … 1111110: CK_SPI/254 1111111: CK_SPI/256
15	BDEN	Bidirectional enable 0: 2 line unidirectional transmit mode 1: 1 line bidirectional transmit mode. The information transfers between the MOSI pin in master and the MISO pin in slave.
14	BDOEN	Bidirectional transmit output enable When BDEN is set, this bit determines the direction of transfer. 0: Work in receive-only mode 1: Work in transmit-only mode
13	CRCEN	CRC calculation enable 0: CRC calculation is disabled 1: CRC calculation is enabled **Note:** This bit field must be cleared when SPIEN is disabled; otherwise, the internal CRC calculation value will persist for the next SPIEN enable.
12	CRCNT	CRC next transfer 0: Next transfer is data 1: Next transfer is CRC value (TCRC) When the transfer is managed by DMA, CRC value is transferred by hardware. This bit should be cleared. In full-duplex or transmit-only mode, set this bit after the last data is written to SPI_DATA register. **Note:** CRCNT takes effect before the frame transmission is completed. SPIEN can only be disabled after TRANS is pulled low.
11	Reserved	Must be kept at reset value.
10	RO	Receive only When BDEN is cleared, this bit determines the direction of transfer. 0: Full-duplex mode 1: Receive-only mode
9	SWNSSEN	NSS software mode selection 0: NSS hardware mode. The NSS level depends on NSS pin 1: NSS software mode. The NSS level depends on SWNSS bit **Note:** This bit should not be set in SPI TI / NSSP mode.

8	SWNSS	NSS pin selection in NSS software mode 0: NSS pin is pulled low 1: NSS pin is pulled high **Note:** 1. This bit has an effect only when the SWNSSEN bit is set. 2. This bit has no meaning in SPI TI / NSSP mode.
7	LF	LSB first mode 0: Transmit MSB first 1: Transmit LSB first **Note:** This bit has no meaning in SPI TI mode.
6	SPIEN	SPI enable 0: SPI peripheral is disabled. 1: SPI peripheral is enabled. **Note:** SPI must be enabled only after the static configuration of the SPI master/slave has been fully completed; otherwise, an invalid state may affect SPI transmissions.
5:3	Reserved	Must be kept at reset value.
2	MSTMOD	Master mode enable 0: Slave mode 1: Master mode
1	CKPL	Clock polarity selection 0: SCK pin is pulled low when SPI is idle 1: SCK pin is pulled high when SPI is idle
0	CKPH	Clock phase selection 0: Capture the first data at the first clock transition 1: Capture the first data at the second clock transition

20.5.2. Control register 1 (SPI_CTL1)

Address offset: 0x04

Reset value: 0x0100 0700

This register can be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TOCFG[7:0]								TOIE	TOEN	SINGMOD	FIFOLBEN	NSSSEL[2:0]			Reserved
rw								rw	rw	rw	rw	rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NSSPOL	Reserved		DZ[4:0]					TFAIE	RFDRIE	ERRIE	TMOD	NSSP	NSSDRV	DMATEN	DMAREN
rw			rw					rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:24	TOCFG[7:0]	Timeout configuration.

		0x00: 1 * SCK 0x01: 2 * SCK … 0xFF: 256 * SCK Recommended software configuration value: TOCFG[7:0] = 10 / (PSC + 1), rounded up. **Note:** TOCFG cannot be configured as 0x00.
23	TOIE	Timeout interrupt enable 0: Disable timeout interrupt 1: Enable timeout interrupt
22	TOEN	Timeout enable 0: Disable timeout 1: Enable timeout
21	SINGMOD	Single-frame mode enable 0: Disable single-frame mode. 1: Enable single-frame mode. After the completion of each data frame transmission, the hardware will automatically release the NSS signal.
20	FIFOLBEN	FIFO loopback mode enable 0: FIFO loopback mode disabled 1: FIFO loopback mode enabled. Write data to SPI_DATA will save the data to TXFIFO and then read back to RXFIFO directly, user can check the status of RXLVL and read data from SPI_DATA to verify quality of FIFO. **Note:** 1. when FIFOLBEN is 1, SPIEN or I2SEN can't be set. 2. When FIFOLBEN is 1, MSTMOD must be configured as 1.
19:17	NSSSEL[2:0]	SPI NSS select 000: NSS0 001: NSS1 … 011: NSS3 **Note:** For GD32A711/A712 devices, only NSS0-3 are supported, and the NSSSEL[2] bit is not valid.
16	Reserved	Must be kept at reset value.
15	NSSPOL	NSS polarity 0: Low active 1: High active **Note:** 1. In TI mode: When NSSPOL=0, it is high active; when NSSPOL=1, it is low active. 2. When the SPI Master Request feature is enabled (MREQEN = 1), this bit becomes invalid.

		3. In I2S mode, this bit must be set to 0.
14:13	Reserved	Must be kept at reset value.
12:8	DZ[4:0]	Data size This field indicates the data size for transfer. 00000: Force to "00111", 8-bit 00001: Force to "00111", 8-bit 00010: Force to "00111", 8-bit 00011: 4-bit 00100: 5-bit … 11111: 32-bit **Note:** 1. When PSC = 0, DZ[4:0] must be greater than or equal to 7. 2. When the Quad SPI mode is enabled (QMOD = 1), the data size DZ[4:0] will be automatically set to 2b'11111 by the hardware. When switching back from Quad SPI mode to Single SPI mode (QMOD = 0), the software needs to reconfigure DZ[4:0] to the desired value.
7	TFAIE	Transmit TXFIFO available interrupt enable 0: TFA interrupt is disabled 1: TFA interrupt is enabled. An interrupt is generated when the TFA bit is set.
6	RFDRIE	Receive RXFIFO data ready interrupt enable 0: RFDR interrupt is disabled 1: RFDR interrupt is enabled. An interrupt is generated when the RFDR bit is set.
5	ERRIE	Errors interrupt enable 0: Error interrupt is disabled 1: Error interrupt is enabled. An interrupt is generated when the CONFERR bit or the RXORERR bit or the TXURERR bit or the FERR bit is set.
4	TMOD	SPI TI mode enable 0: SPI TI mode disabled 1: SPI TI mode enabled
3	NSSP	SPI NSS pulse mode enable 0: SPI NSS pulse mode disable 1: SPI NSS pulse mode enable
2	NSSDRV	Drive NSS output 0: NSS output is disabled 1: NSS output is enabled. If the NSS pin is configured as output, the NSS pin is active (The level depends on NSSPOL) in master mode when SPI is enabled. If the NSS pin is configured as input, the NSS pin should be inactive (The level depends on NSSPOL) in master mode, and this bit has no effect.
1	DMATEN	Transmit FIFO DMA enable

0: Transmit FIFO DMA is disabled
1: Transmit FIFO DMA is enabled. When TFA in SPI_STAT is set, SPI will generate a transmission request to the corresponding DMA channel.

0	DMAREN	Receive FIFO DMA enable 0: Receive FIFO DMA is disabled 1: Receive FIFO DMA is enabled. When RFDR in SPI_STAT is set, SPI will generate a receive request to the corresponding DMA channel.

20.5.3. Status register (SPI_STAT)

Address offset: 0x08
Reset value: 0x0000 0002

This register can be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												TOF	Reserved	TXLVL[3:2]	
												rc_w0		r	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXLVL[1:0]		Reserved	RXLVL[3:0]				FERR	TRANS	RXORER	CONFER	Reserved	TXURER	I2SCH	TFA	RFDR
r			r				rc_r	r	r	r		rc_r	r	r	r

Bits	Fields	Descriptions
31:20	Reserved	Must be kept at reset value.
19	TOF	Timeout flag This bit is set by hardware and is able to be cleared by writing 0. 0: Timeout not occurred 1: Timeout occurred
18	Reserved	Must be kept at reset value.
17:14	TXLVL[3:0]	Transmission FIFO current words number 0000: 0 word 0001: 1 word 0010: 2 words 0011: 3 words … 1000: 8 words Others: reserved
13	Reserved	Must be kept at reset value.
12:9	RXLVL[3:0]	Reception FIFO current words number 0000: 0 word 0001: 1 word

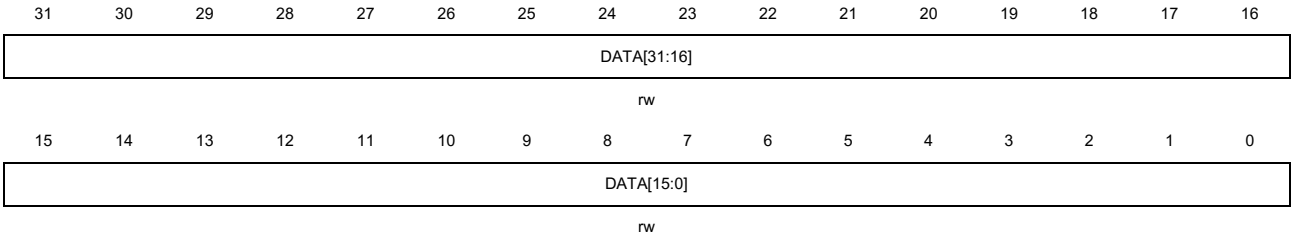
		0010: 2 words 0011: 3 words … 1000: 8 words Others: reserved
8	FERR	Format error SPI TI mode: 0: No TI mode format error 1: TI mode format error occurs This bit is set by hardware and is able to be cleared by a read access to the SPI_STAT register. I2S mode: 0: No I2S format error 1: I2S format error. In slave I2S mode, the I2S monitors the I2S_WS signal and an error flag will be set if I2S_WS toggles at an unexpected position.
7	TRANS	Transmitting ongoing bit 0: SPI is idle 1: SPI is currently transmitting and/or receiving a frame This bit is set and cleared by hardware.
6	RXORERR	Reception overrun error bit 0: No reception overrun error occurs 1: Reception overrun error occurs This bit is set by hardware and cleared by a read operation on the SPI_DATA register followed by a read access to the SPI_STAT register.
5	CONFERR	SPI configuration error 0: No configuration fault occurs 1: Configuration fault occurred. (In master mode, the NSS pin is active in NSS hardware mode or SWNSS bit is active in NSS software mode. The active level depends on the NSSPOL configuration.) This bit is set by hardware and cleared by a read or write operation on the SPI_STAT register followed by a write access to the SPI_CTL0 register.
4	Reserved	Must be kept at reset value.
3	TXURERR	Transmission underrun error bit 0: No transmission underrun error occurs 1: Transmission underrun error occurs This bit is set by hardware and cleared by a read operation on the SPI_STAT register. **Note:** When the SPI slave hardware sends a CRC frame, the TXURERR flag will not be set.
2	I2SCH	I2S channel side

		0: The next data needs to be transmitted or the data just received is channel left. 1: The next data needs to be transmitted or the data just received is channel right. This bit is set and cleared by hardware. This bit is not used in SPI mode, and has no meaning in the I2S PCM mode.
1	TFA	Transmission FIFO available 0: Transmission FIFO is unavailable, it means TXFIFO has no space for bus write access 1: Transmission FIFO is available, it means TXFIFO has enough space for bus write access
0	RFDR	Reception FIFO data ready 0: Reception FIFO data is not ready to be read. it means has no enough data for bus read 1: Reception FIFO data is ready to be read, it means has enough data for bus read access.

20.5.4. Data register (SPI_DATA)

Address offset: 0x0C
Reset value: 0x0000 0000

This register can be accessed by word (32-bit).



Bits	Fields	Descriptions
31:0	DATA[31:0]	Data transfer register The hardware has two FIFOs, including TXFIFO and RXFIFO. The SPI_DATA register serves as an interface between the TXFIFO and RXFIFO. Write data to SPI_DATA will save the data to TXFIFO and read data from SPI_DATA will get the data from RXFIFO. **Note:** Modifications to certain registers(MSTMOD\FIFOLBEN\I2SSEL\CKPL) may cause this register to generate data. It is necessary to clear the FIFO after these registers are modified to avoid reading corrupted data.

20.5.5. CRC polynomial register (SPI_CRCPOLY)

Address offset: 0x10
Reset value: 0x0000 0107

This register can be accessed word (32-bit).

This register has no meaning in I2S mode.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRCPOLY[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCPOLY[15:0]															
rw															

Bits	Fields	Descriptions
31:0	CRCPOLY[31:0]	CRC polynomial register This register contains the CRC polynomial and it is used for CRC calculation.

20.5.6. Transmit CRC register (SPI_TCRC)

Address offset: 0x18
Reset value: 0x0000 0000

This register can be accessed by word (32-bit).

This register has no meaning in I2S mode.

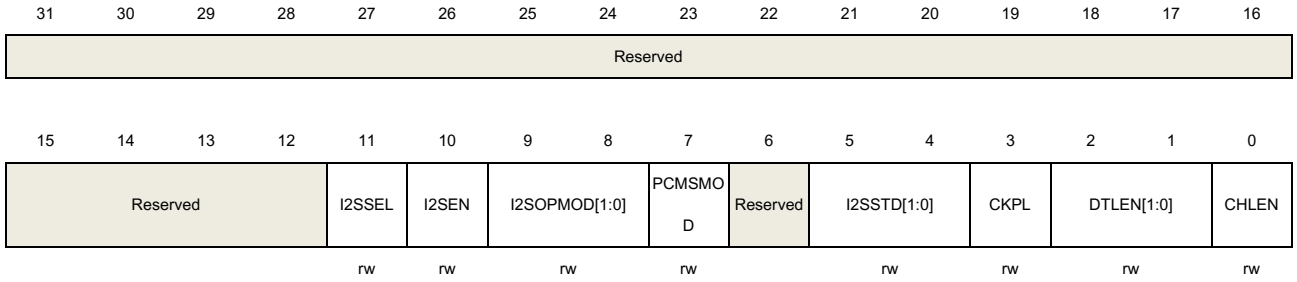
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TCRC[31:16]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCRC[15:0]															
r															

Bits	Fields	Descriptions
31:0	TCRC[31:0]	TX CRC value When the CRCEN bit of SPI_CTL0 is set, the hardware computes the CRC value of the transmitted bytes and saves them in TCRC register. CRC size is same as DZ[4:0]. The hardware computes the CRC value after each transmitted bit, when the TRANS is set, a read to this register could return an intermediate value. The different frame formats (LF bit of the SPI_CTL0) will get different CRC values. This register is reset when the CRCEN bit in SPI_CTL0 register or the SPIxRST bit in RCU reset register is set.

20.5.7. I2S control register (SPI_I2SCTL)

Address offset: 0x1C
Reset value: 0x0000 0000

This register can be accessed by word (32-bit).



Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11	I2SSEL	I2S mode selection 0: SPI mode 1: I2S mode This bit should be configured when SPI/I2S is disabled.
10	I2SEN	I2S enable 0: I2S is disabled 1: I2S is enabled This bit is not used in SPI mode.
9:8	I2SOPMOD[1:0]	I2S operation mode 00: Slave transmission mode 01: Slave reception mode 10: Master transmission mode 11: Master reception mode This bit should be configured when I2S mode is disabled. This bit is not used in SPI mode.
7	PCMSMOD	PCM frame synchronization mode 0: Short frame synchronization 1: long frame synchronization This bit has a meaning only when PCM standard is used. This bit should be configured when I2S mode is disabled. This bit is not used in SPI mode.
6	Reserved	Must be kept at reset value.
5:4	I2SSTD[1:0]	I2S standard selection 00: I2S Philips standard 01: MSB justified standard 10: LSB justified standard 11: PCM standard These bits should be configured when I2S mode is disabled. These bits are not used in SPI mode.
3	CKPL	Idle state clock polarity 0: The idle state of I2S_CK is low level

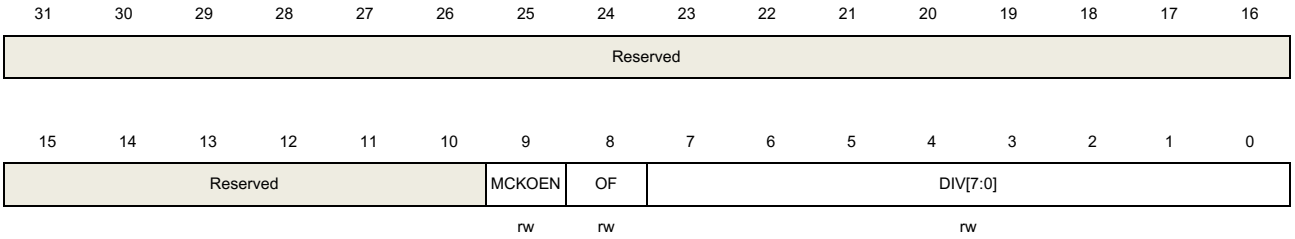
		1: The idle state of I2S_CK is high level This bit should be configured when I2S mode is disabled. This bit is not used in SPI mode.
2:1	DTLEN[1:0]	Data length 00: 16 bits 01: 24 bits 10: 32 bits 11: Reserved These bits should be configured when I2S mode is disabled. These bits are not used in SPI mode.
0	CHLEN	Channel length 0: 16 bits 1: 32 bits The channel length must be equal to or greater than the data length. This bit should be configured when I2S mode is disabled. This bit is not used in SPI mode.

20.5.8. I2S clock prescaler register (SPI_I2SPSC)

Address offset: 0x20

Reset value: 0x0000 0002

This register can be accessed by word (32-bit).



Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9	MCKOEN	I2S_MCK output enable 0: I2S_MCK output is disabled 1: I2S_MCK output is enabled This bit should be configured when I2S mode is disabled. This bit is not used in SPI mode.
8	OF	Odd factor for the prescaler 0: Real divider value is DIV * 2 1: Real divider value is DIV * 2 + 1 This bit should be configured when I2S mode is disabled. This bit is not used in SPI mode.

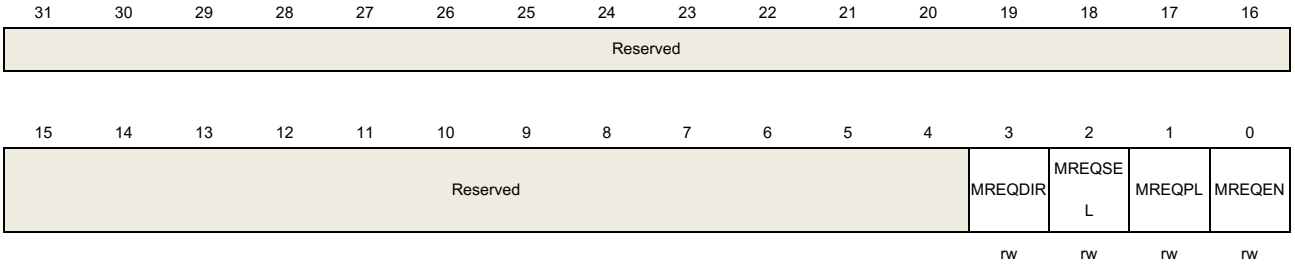
7:0	DIV[7:0]	Dividing factor for the prescaler Real divider value is DIV * 2 + OF DIV must not be 0. These bits should be configured when I2S mode is disabled. These bits are not used in SPI mode.

20.5.9. Master request configuration register (SPI_MREQCFG)

Address offset: 0x28
Reset value: 0x0000 0000

This register can be accessed by word (32-bit).

This register has no meaning in I2S mode.



Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3	MREQDIR	MREQ pin direction 0: MREQ pin is configured as input 1: MREQ pin is configured as output **Note:** The MREQ pin should only be configured as output in slave mode or input for master mode.
2	MREQSEL	Master request input source select 0: SPI_MREQ pin as master request input source 1: Input trigger as master request input source **Note:** 1. The SPI_MREQ pin is multiplexed with the SPI_NSS1 pin. When the master request function is enabled with the SPI_MREQ pin, the NSSSEL cannot be 1. 2. In slave mode, this bit field must be configured as 0. 3. For GD32A711 / A712 devices, when MREQSEL is configured to 1 in master mode, the prescaler coefficient must be configured according to ***Table 20-6*** for different communication rates. 4. For GD32A711 / A712 devices, when the input trigger is used as the master request input source, the invalid level width of the master request must be greater than 8*PCLK + 2*SCK clock cycles. The invalid level state depends on the configuration of the MREQPL bit.

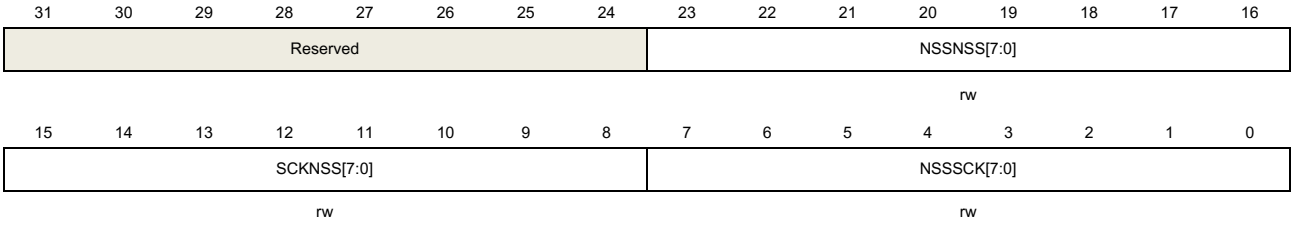
1	MREQPL	Master request polarity 0: MREQ pin or input trigger is active high 1: MREQ pin or input trigger is active low
0	MREQEN	Master request enable 0: Master request disabled 1: Master request enabled In SPI master mode, the SPI will only start a new SPI bus transfer if the host request input is asserted. When the SPI is busy, the master request input is ignored. In SPI slave mode, the MREQ output pin will assert when data is available to be transmitted. **Note:** 1. When using the master request feature, to avoid unexpected master request triggers due to the default polarity (MREQPL), the software should first enable the MREQEN bit of the SPI slave, and then enable the MREQEN bit of the SPI master. When disabling this feature, the software should first disable the MREQEN bit of the SPI master and then disable the MREQEN bit of the SPI slave. 2. The master request mode does not support hardware CRC validation. 3. The master request mode cannot be used simultaneously with NSSP, TI, Single-frame mode, and Hardware continuous mode.

20.5.10. Clock configuration register 0 (SPI_CKCFG0)

Address offset: 0x2C
Reset value: 0x0000 0000

This register can be accessed by word (32-bit).

This register has no meaning in I2S mode.



Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:16	NSSNSS[7:0]	Minimum delay from NSS inactive to NSS active 0x00: 1 * SCK 0x01: 2 * SCK … 0xFF: 256 * SCK **Note:** In SPI TI and Quad-SPI mode, this bit field is invalid.

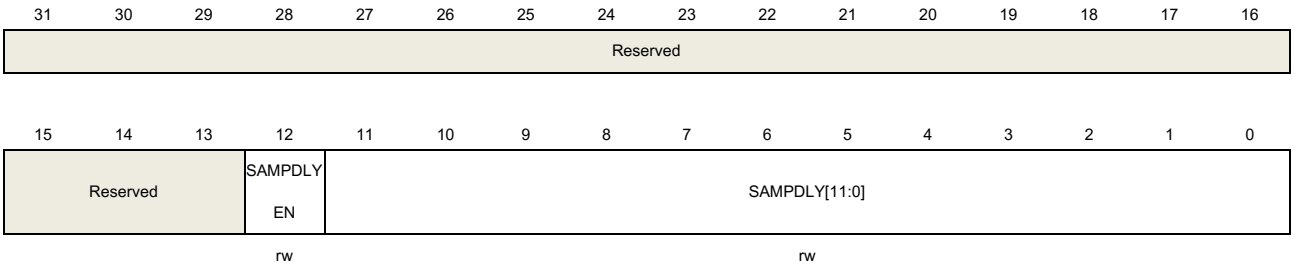
15:8	SCKNSS[7:0]	Minimum delay from SCK inactive to NSS inactive 0x00: 1 * SCK 0x01: 2 * SCK … 0xFF: 256 * SCK **Note:** In SPI TI and Quad-SPI mode, this bit field is invalid.
7:0	NSSSCK[7:0]	Minimum delay from NSS active to SCK active 0x00: 1 * SCK 0x01: 2 * SCK … 0xFF: 256 * SCK **Note:** In SPI TI and Quad-SPI mode, this bit field is invalid.

20.5.11. Clock configuration register 1 (SPI_CKCFG1)

Address offset: 0x30
Reset value: 0x0000 1003

This register can be accessed by word (32-bit).

This register has no meaning in I2S mode.



Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12	SAMPDLYEN	Master receive data sampling delay enabled 0: Disable sampling delay in master mode 1: Enable sampling delay in master mode
11:0	SAMPDLY[11:0]	Master receive data sampling delay 0x000: 1 / CK_SPI 0x001: 2 / CK_SPI … 0xFFF: 4096 / CK_SPI Recommended value: a delay of 1.5 / SCK time, i.e. SAMPDLY = (PCS[6:0] + 1) × 3 - 1, which may be fine-tuned according to various factors. Default value calculation:

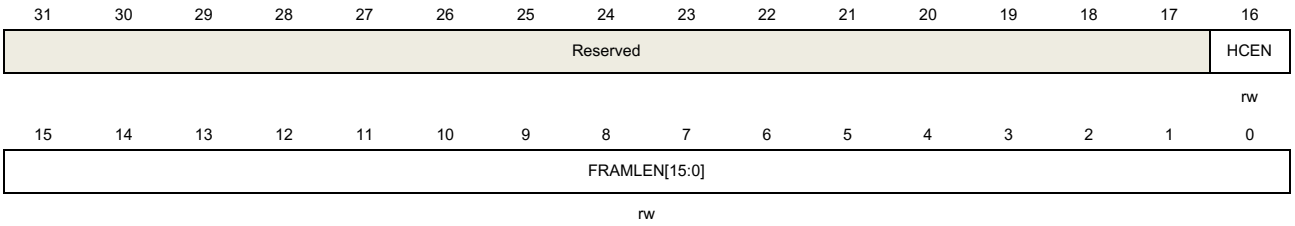
Taking SCK = 20 MHz as an example, the valid sampling window considering all factors is 38 - 62 ns;
Considering signal deviation, a mid-point value of 50 ns is adopted;
When CK_SPI = 80 MHz (12.5 ns), the corresponding delay should be set to 4 / CK_SPI to achieve 50 ns, therefore SAMPDLY = 3.

20.5.12. Continuous control register (SPI_CCTL)

Address offset: 0x34
Reset value: 0x0000 0000

This register can be accessed by word (32-bit).

This register has no meaning in I2S mode.



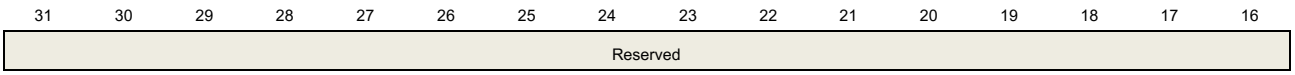
Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value.
16	HCEN	Hardware continuous frame mode enable 0: Hardware continuous mode disabled. 1: Hardware continuous mode enabled. NSS asserted when data ready and deasserted when FRAMLEN reached.
15:0	FRAMLEN[15:0]	Frame length 0x0000: 1 * Frame 0x0001: 2 * Frame … 0xFFFF: 65536 * Frame **Note:** When the hardware continuous mode is disabled, this bit field becomes invalid.

20.5.13. Quad-SPI mode control register (SPI_QCTL)

Address offset: 0x80
Reset value: 0x0000 0000

This register can be accessed by word (32-bit).

This register has no meaning in I2S mode.



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														QRD	QMOD
														rw	rw

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value
1	QRD	Quad-SPI mode select 0: SPI is in quad write mode 1: SPI is in quad read mode **Note:** This bit should only be configured when SPI is not busy (TRANS bit cleared).
0	QMOD	Quad-SPI mode enable 0: SPI is in single wire mode 1: SPI is in Quad-SPI mode **Note:** This bit should only be configured when SPI is not busy (TRANS bit cleared).

21. Quad-SPI interface (QSPI)

21.1. Overview

The QSPI is a specialized interface that communicate with flash memories. This interface supports single, dual or quad SPI FLASH. It can operate in normal mode, read polling mode and memory map mode.

21.2. Characteristics

- Three functional modes: normal mode (address extend), read polling mode and memory map mode.
- Fully programmable command format for both normal mode and memory map mode.
- Integrated FIFO for transmission/reception.
- Support SDR and DDR mode.
- Support DQS signal.
- 8, 16, or 32-bit data accesses.
- DMA channel for normal mode.

21.3. Function overview

21.3.1. QSPI block diagram

7 signals are used to interface with an external flash memory. The description is shown in *Table 21-1. QSPI signal description*.

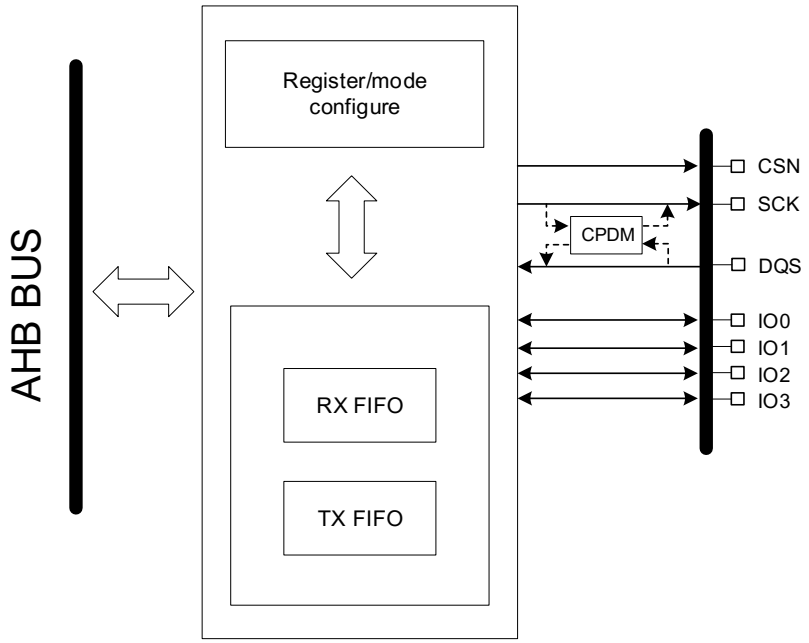
Table 21-1. QSPI signal description

Pin name	Direction	Description
CSN	O	chip select output (active low)
SCK	O	clock output
IO0/SO	I/O	single mode: data output dual mode: data input or output quad mode: data input or output
IO1/SI	I/O	single mode: data input dual mode: data input or output quad mode: data input or output
IO2	I/O	single mode: connect WP pin of flash, control "write protect" function dual mode: connect WP pin of flash, control "write protect" function quad mode: data input or output

Pin name	Direction	Description
IO3	I/O	single mode: connect HOLD pin of flash, control "hold" function dual mode: connect HOLD pin of flash, control "hold" function quad mode: data input or output
DQS	I	data strobe signal

Figure 21-1 QSPI diagram shows the block diagram of the QSPI unit.

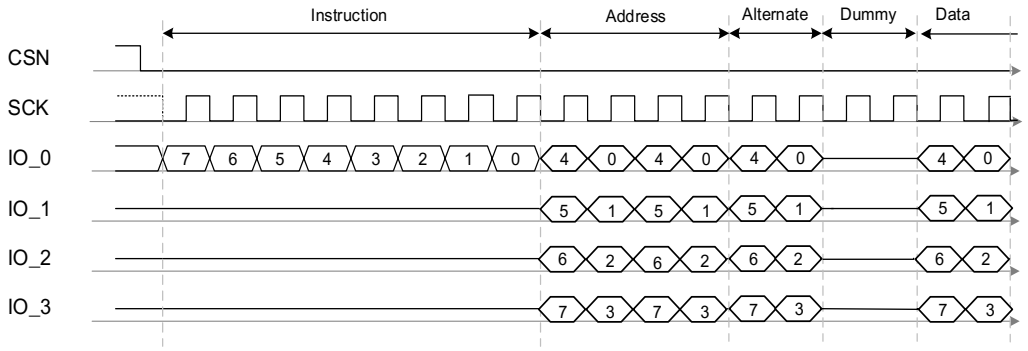
Figure 21-1 QSPI diagram



21.3.2. QSPI command format

The QSPI communicates with the flash memory using commands in various formats. There are totally 5 phases which can be included or not: instruction, address, alternate byte, dummy and data. Any phase can be skipped, but it must contain at least one of the instruction phase, address phase, alternate byte phase, and data phase. This is guaranteed by the software, and the hardware design has no protection method. In addition, the most-significant-bit always occupies the highest IO line number.

Figure 21-2 QSPI command format



The commands and the corresponding configuration are shown in ***Table 21-2. QSPI command description***.

Table 21-2. QSPI command description

Command	Send information	Configuration	Note
instruction	8-bit instruction	QSPI_TCFG register defines the instruction and signal line mode	-
address	1-4 bytes of address	QSPI_ADDR register defines the information of address. QSPI_TCFG register defines the number of address and signal line mode	-
Alternate byte	1-4 alternate-bytes	QSPI_ALTE register defines the information of alternate bytes, QSPI_TCFG register defines the number of alternate-bytes and signal line mode	-
dummy	0-31 cycles	QSPI_TCFG register defines the cycles and DATAMOD field (QSPI_TCFG register) defines the dummy signal line mode	given without any data being transferred for external flash, in order to wait flash prepare data
data	any number of bytes	In normal mode, QSPI_DTLEN register defines the number of bytes. DATAMOD field (QSPI_TCFG register) defines the data signal line mode, and the configuration of DATAMOD = 00 must only be used in normal write mode	In memory map mode, the number of bytes to be transmitted is specified as single AHB bus access operation, these could be 8, 16 or 32 read/write access, corresponding to 1, 2, or 4 bytes.

Note: Signal line mode can be no command, single line mode, dual line mode, or quad line mode.

21.3.3. QSPI signal line modes

Each of the instruction, address, alternate-byte, or data phase can be configured separately into signal line modes by setting IMOD / ADDRMOD / ALTEMOD / DATAMOD.

Table 21-3. QSPI signal line modes

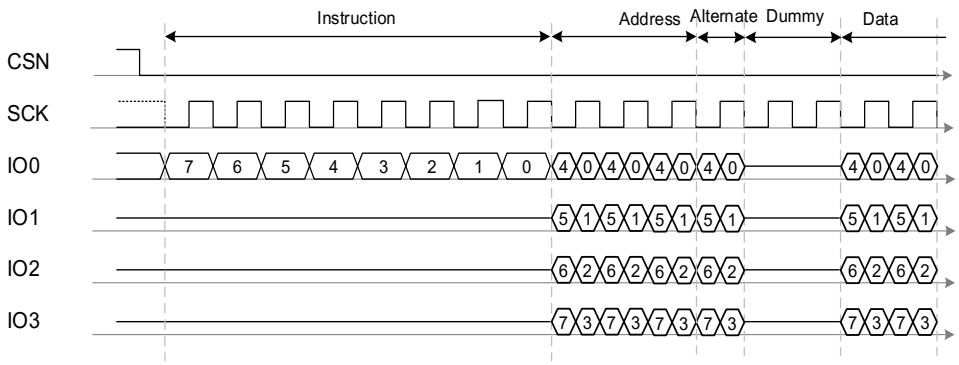
Signal line modes		Single line mode	Dual line mode	Quad line mode
Configure field	IMOD	01 or 00	10 or 00	11 or 00
	ADDRMOD			
	ALTEMOD			
	DATAMOD			
Pins	IO0 (SO)	Output	Input: data read (high impedance) output: all other phases	Input: data read (high impedance) Output: all other phases.
	IO1 (SI)	Input (high impedance)		
	IO2	Output 0 (deactivate "write protect function")		
	IO3	Output 1 (deactivate "hold" function)		
Description		In dummy phase when DATAMOD = 01, IO0 output, IO1 input (high impedance)	In dummy phase when DATAMOD = 10, IO0 / IO1 are always high-impedance.	In dummy phase when DATAMOD = 11, IO0 / IO1 / IO2 / IO3 are always high-impedance.

IO2 / IO3 are used only in quad mode, and if none of the 5 phases is configured in quad mode, IO2 / IO3 will be released and can be used for other functions even when QSPI is enabled.

21.3.4. QSPI DDR mode

By default, QSPI operates in SDR (single data rate) mode, when the DDREN bit in QSPI_TCFG register is set to 1, the QSPI operates in DDR (double data rate) mode. In DDR mode, during the address / alternate-byte / data phases, IO0, IO1, IO2 and IO3 signals all transmit data along the two clock edges (rising and falling edges) of SCK signal. During the instruction phase, SDR mode is still used for transmission, and IO0, IO1, IO2 and IO3 signals are sampled along the falling edge of SCK.

Figure 21-3 QSPI DDR mode

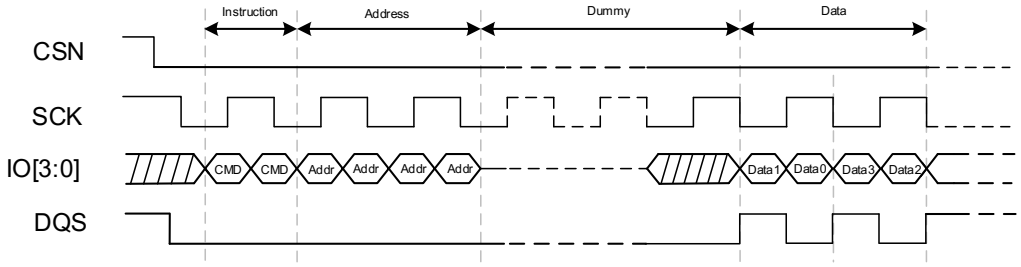


In DDR mode, according to actual usage requirements, user need to use CPDM to fine-tune the QSPI receive clock phase by setting the DLYSCEN bit in the QSPI DCFG register to 1. (In this case, set RCKSEL to 0 to select the SCK as receiving clock.) For CPDM usage, please refer to ***Clock phase delay module (CPDM)***.

21.3.5. DQS signal

The DQS signal is typically used in high speed applications to indicate when the output data of memory is ready to be fetched by the MCU. When the DQS function is enabled, the toggling frequency is the same as the SCK frequency. For SDR read operations, the data should only be latched on the rising edge of DQS signal. For DDR read operations, the data should be latched on both rising edge and falling edge of the DQS signal.

Figure 21-4 DDR read timing using DQS



21.3.6. CSN and SCK

The default value of CSN is high, and it falls before a command begins and rises as soon as it finishes.

SCK output signal is a gate signal from internal clock, where the internal clock is present all the time.

CSN falls one SCK cycle before the first valid rising SCK edge, and rises on SCK cycle after the final valid rising SCK edge.

When the FIFO stays empty in a write command phase, or full in a read command phase, SCK will be stalled and keep low until the FIFO can work again. At this moment if the CSN is high, SCK will rise back up one half of a SCK cycle after the rising edge of the CSN.

21.4. Operating modes

The QSPI can operate in normal mode, read polling mode and memory map mode. In normal mode, all operations are performed depending on QSPI registers. In read polling mode, the values of status registers in external flash memory are periodically read and checked. In memory map mode, the external flash memory is mapped to the microcontroller address space (range from 0x9000 0000 to 0x97FF FFFF) and can be accessed as an internal memory.

21.4.1. Normal mode

The write operation in normal mode is selected by configuring the FMOD[1:0] in QSPI_TCFG register to "00". The data to be transmitted is written into QSPI_DATA. The read operation in normal mode is selected by setting the FMOD[1:0] in QSPI_TCFG register to "01", and the data to be received is read from QSPI_DATA.

The DTLEN[31:0] in QSPI_DTLEN register defines the number of bytes to be transferred. If DTLEN is 0xFFFF FFFF, the number of bytes to be transferred is considered undefined, and the transmission continues until the memory size boundary is reached as specified by FMSZ[4:0] in QSPI_DCFG register. If DTLEN is 0xFFFF FFFF and FMSZ[4:0] is configured as 0x1F, the flash memory capacity is 4GB, the transmission will continue indefinitely until an abort request occurs or the QSPI is disabled.

The transfer complete flag TC will be set when the number of bytes programmed in DTLEN has been transferred. If the DTLEN is 0xFFFF FFFF, TC will be set when the transmitted/received byte number equals to the external memory size defined by FMSZ[4:0]. An interrupt is generated if TCIE and TC are both set. The TC bit is cleared by writing 1 to the TCC bit in QSPI_STATC register.

Initialize a command sequence

The command sequence starts immediately after the last information is provided by software according to communication requirement.

When neither address nor data are required, the sequence starts immediately after QSPI_TCFG has been accessed.

When address is required and no data is required, the sequence starts after QSPI_ADDR has been accessed.

When both address and data are required in normal write mode, the command sequence starts after QSPI_DATA has been accessed.

FIFO

A FIFO is implemented to transfer data. In normal write mode, the relationship between the AHB write access mode and the number of bytes add to FIFO is shown in ***Table 21-4. AHB write access mode and number of bytes add to FIFO***.

Table 21-4. AHB write access mode and number of bytes add to FIFO

AHB write access mode	Number of bytes add to FIFO
32-bit	4 bytes
16-bit	2 bytes
8-bit	1 byte

Note: When the AHB write access mode is 8-bit or 16-bit, the least significant bytes are valid in QSPI_DATA register.

FIFO threshold is defined by FTL[3:0] in QSPI_CTL register, in normal read mode, when the amount of bytes in the FIFO is equal or above the defined threshold, FIFO threshold flag FT in QSPI_STAT register will be set. FT is also set after data phase is completed if FIFO is not empty. In normal write mode, when the amount of the empty bytes in the FIFO is above the threshold, FT will be set.

An interrupt is generated if both FTIE and FT is set. If DMA is enabled, a DMA request is generated by FT, until this flag is cleared.

In normal read mode, when the FIFO becomes full, the QSPI temporarily stop SCK clock to avoid overrun. The read sequence cannot resume until there is at least 4 bytes of remaining space in the FIFO.

21.4.2. Read polling mode

The read polling mode can be selected by configuring the FMOD[1:0] to "10". In read polling mode, the QSPI periodically starts a read command with up to 4-bytes data. The received data can be bit-wise masked and compared with a defined data content. If a match happens, an interrupt will be generated if RPMFIE is set.

The read polling access sequence is started the same as that of normal read mode. The BUSY bit keeps 1 between periodic intervals.

Polling match mode configured by RPMM controls the comparison match mode. If RPMM = 0, the AND mode is selected. In this mode, status match flag RPMF will be set only when there is a match on all the unmasked bits. While if RPMM = 1, the OR mode is selected. In this mode, RPMF will be set if there is a match on any of the unmasked bit.

In read polling mode, if RPMS is set, read polling sequence will stop when a match is detected, and the BUSY flag is cleared at the end of data phase. Otherwise, the periodic sequence always continues until ABORT bit is set or the QSPI is disabled.

In read polling mode, FIFO is bypassed and the status bytes read are stored in QSPI_DATA, and the status bytes stored are not affected by the MASK control field. Contents in QSPI_DATA are renewed at the beginning of data phase if there is any.

FT is set at the end of data phase, where the external flash memory status bytes are considered read, and it is cleared when QSPI_DATA is read.

21.4.3. Memory map mode

The memory map mode can be selected by configuring the FMOD[1:0] to "11". In memory map mode, the external flash memory is considered as internal memory, no more than 128MB can be addressed even if the external memory is larger. Even if the address range defined by FMSZ is within the 128MB range, the memory-mapped mode does not allow addresses to exceed the range defined by FMSZ. Or else, an error will be generated. If the AHB master is the CPU, a hard fault interrupt will be generated. If the AHB master is the DMA, a transfer error will be generated, and the corresponding DMA channel is disabled by hardware.

In this mode, byte, half-word, and word single or burst access are supported.

The memory map mode supports prefetch function on sequential address accessing. The QSPI load the data at the following address before accessing it, if the next access is indeed at the next address, the access will be faster since the data is already prefetched. Otherwise, the read sequence is restarted, pulling CSN low before the read sequence starts.

When the FIFO is full, the output of SCK will be stopped, and the CSN is low during this period. If the TMOUTEN bit in QSPI_CTL register is set, CSN will be pulled high when the duration of the low period reaches the number of SCK clock cycles specified in QSPI_TMOUT register.

At the beginning of a transfer, BUSY goes high before CSN falls, and is cleared when a timeout occurred or abort / disable is issued.

21.5. QSPI configuration

21.5.1. Flash configuration

The configuration in QSPI_DCFG register can be used to specify the characteristics of the external flash memory, so that the QSPI interface can work consistently.

The size of the external memory is defined by FMSZ[4:0] in QSPI_DCFG register. FMSZ + 1 is the number of address bits in the flash memory. The maximum of the flash capacity can be up to 4GB in normal mode.

CSHC[2:0] defines the chip select high time, it specifies the minimum number of SCK cycles that CSN must stay high between two command sequences.

21.5.2. IP configuration

The configuration in QSPI_CTL register can be used to specify the characteristics of the QSPI IP.

PSC[7:0] in QSPI_CTL register indicates the clock prescaler division factor.

SSAMPLE is used to define which SCK edge is used to sample data. By default, the QSPI samples data one half of a SCK cycle after the external flash drives. However, it may be beneficial to sample data later because of the external signal delays. The sample edge can be shifted by half of an SCK cycle using SSAMPLE bit.

The DMA request is enabled by setting the DMAEN bit in QSPI_CTL register. The FIFO threshold level is configured in FTL[3:0] bits in QSPI_CTL register.

21.6. Send instruction only once

Sending instruction only once mode is enabled by setting the SIOO bit in QSPI_TCFG register, this function is valid for all functional modes. If SIOO bit is set, the instruction is sent only once

after QSPI_TCFG has been accessed. The subsequent command sequence will skip the instruction phase before the next configuration of QSPI_TCFG register.

21.7. Error and interrupts

TERR and AHB error will be generated if one of the conditions occurs in *Table 21-5. TERR and AHB error conditions*.

Table 21-5. TERR and AHB error conditions

Error name	Condition
TERR	1. In normal mode or read polling mode, a wrong address has been programmed in QSPI_ADDR register according to the address defined by FMSZ. 2. In normal mode, the address (ADDR) plus data length (DTLEN) is greater than external memory size.
AHB error	1. In memory map mode, out of range access is done by AHB master, or when the QSPI is disabled. 2. AHB master is accessing the memory mapped space while the memory map mode is not enabled.

The QSPI interrupt event and flags are listed in *Table 21-6. QSPI interrupt events*.

Table 21-6. QSPI interrupt events

Interrupt event	Event Flag	Interrupt enable bit	Clear method
FIFO threshold interrupt	FT	FTIE	By hardware
Transfer complete interrupt	TC	TCIE	Set TCC bit in QSPI_STATC register
Transfer error interrupt	TERR	TERRIE	Set TERRC bit in QSPI_STATC register
Timeout interrupt	TMOUT	TMOUTIE	Set TMOUTC bit in QSPI_STATC register
Status match interrupt	RPMF	RPMFIE	Set RPMFC bit in QSPI_STATC register

21.8. Clock phase delay module (CPDM)

21.8.1. Overview

The Clock Phase Delay Module (CPDM) is used to delay the phase of the input clock and then output the clock. When used, the application needs to first program the phase of the output clock, and then use the output clock in other peripherals to receive data.

Phase delay is related to voltage and temperature and may require reconfiguration of the

application and redetermination of the phase relationship between the output clock and the received data as parameters change.

21.8.2. Characteristics

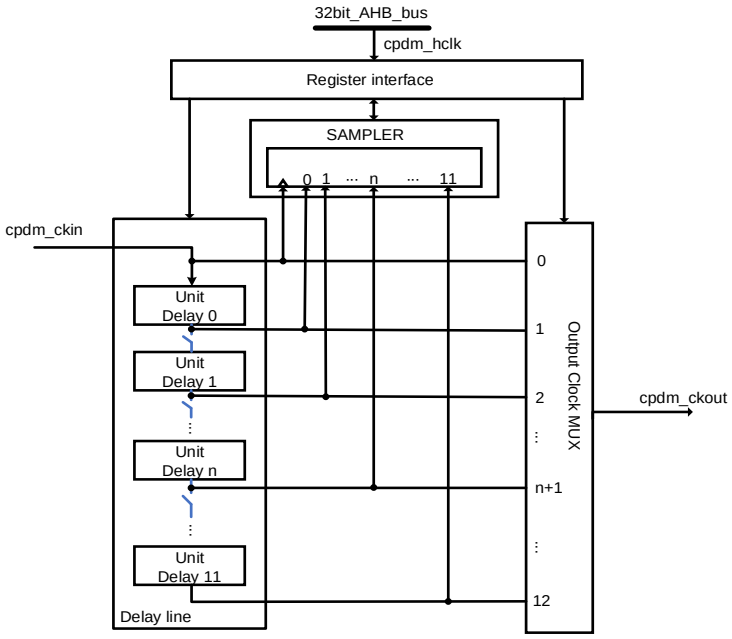
- Supports the input clock frequency ranges: 25 MHz ~ 100MHz.
- Supports up to 12 output clock phase selections.

21.8.3. Function overview

The Clock Phase Delay Module (CPDM) consists of register interface module, delay line module, delay line length sampling module and output clock selection multiplexer. The delay line module is used to generate unit delay.

Note:
CPDM is only used in QSPI, please refer to *Figure 21-5. Schematic diagram of CPDM*.

Figure 21-5. Schematic diagram of CPDM



Note:

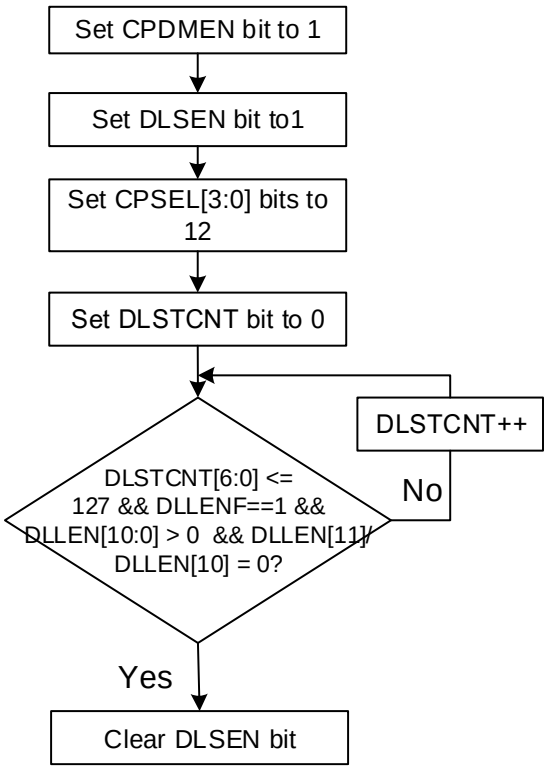
cpdm_hclk: register interface clock of CPDM.
cpdm_ckin: the input clock of CPDM.
cpdm_ckout: the output clock of CPDM.

21.8.4. Operation process

Step one

The delay line length needs to be configured before configuring the output clock phase. The configured delay line length should cover a complete input clock cycle. *Figure 21-6. CPDM delay line length configuration flowchart* shows the configuration process.

Figure 21-6. CPDM delay line length configuration flowchart



Note:

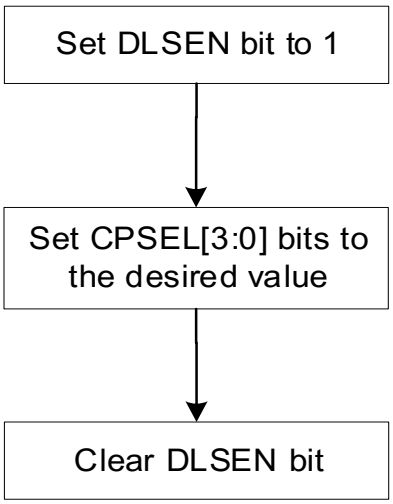
1. If DLLEN[10:0] > 0 and DLLEN[11] / DLLEN[10] = 0, configure the delay line length to an input clock cycle.

2. For N = 10 to 0, if DLLEN[N] = 1, the number of unit delays covering an input clock cycle is N.

Step two

The user can configure any output clock phase between unit delay N when the delay line length has been configured for one full input clock cycle. The specific method is as follows *Figure 21-7. CPDM output clock phase configuration flowchart*.

Figure 21-7. CPDM output clock phase configuration flowchart



21.9. QSPI register definition

QSPI base address: 0x4900 1400

Table 21-7. QSPI register overview

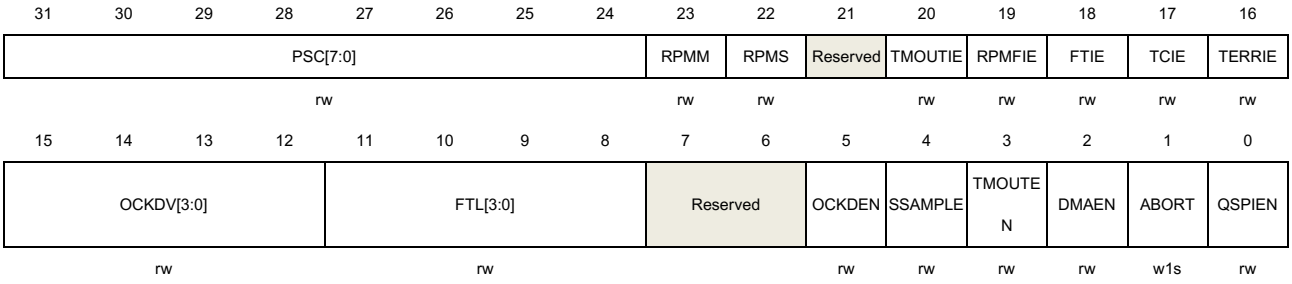
Register Name	Symbol	Address Offset	Reset Value
Control register	QSPI_CTL	0x00	0x0000 0010
Device configuration register	QSPI_DCFG	0x04	0x001F 0000
Status register	QSPI_STAT	0x08	0x0000 0004
Status clear register	QSPI_STATC	0x0C	0x0000 0000
Data length register	QSPI_DTLEN	0x10	0x0000 0000
Transfer configuration register	QSPI_TCFG	0x14	0x0000 0000
Address register	QSPI_ADDR	0x18	0x0000 0000
Alternate bytes register	QSPI_ALTE	0x1C	0x0000 0000
DATA register	QSPI_DATA	0x20	0x0000 0000
Status mask register	QSPI_STATMK	0x24	0x0000 0000
Status match register	QSPI_STATMATCH	0x28	0x0000 0000
Interval register	QSPI_INTERVAL	0x2C	0x0000 0000
Timeout register	QSPI_TMOUT	0x30	0x0000 0000
Flush register	QSPI_FLUSH	0x34	0x0000 0000
Timer register	QSPI_TMR	0x38	0x0000 0055

21.9.1. Control register (QSPI_CTL)

Address offset: 0x00

Reset value: 0x0000 0010

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:24	PSC[7:0]	The clock prescaler of AHB clock for generating SCK. The relationship between the frequency of SCK and the AHB is $f_{SCK}=f_{AHB}/(PSC+1)$. 0: $f_{SCK}=f_{AHB}$ 1: $f_{SCK}=f_{AHB}/2$ 2: $f_{SCK}=f_{AHB}/3$ ... 255: $f_{SCK}=f_{AHB}/256$

		When the clock division factor is odd, the duty cycle of SCK is not 50%. And the low level is one cycle longer than the high level. These bits can be modified only when BUSY = 0.
23	RPMM	Read polling match mode 0: AND match mode. If all the unmasked bits received from the flash memory match with the corresponding bits in QSPI_STATMATCH register, RPMF will be set. 1: OR match mode. If any one of the unmasked bits received from the Flash memory matches its corresponding bit in the QSPI_STATMATCH register, RPMF will be set. This bit can be modified only when BUSY = 0.
22	RPMS	Read polling mode stop This bit determines if read polling is stopped after a match. 0: Read polling mode stops when ABORT bit is set or the QSPI is disabled. 1: Read polling mode stops when a match happens. This bit can be modified only when BUSY = 0.
21	Reserved	Must be kept at reset value.
20	TMOUTIE	Timeout interrupt enable 0: disable timeout interrupt 1: enable timeout interrupt
19	RPMFIE	Read polling mode match interrupt enable 0: disable read polling mode match interrupt 1: enable read polling mode match interrupt
18	FTIE	FIFO threshold interrupt enable 0: disable FIFO threshold interrupt 1: enable FIFO threshold interrupt
17	TCIE	Transfer complete interrupt enable 0: disable transfer complete interrupt 1: enable transfer complete interrupt
16	TERRIE	Transfer error interrupt enable 0: disable transfer error interrupt 1: enable transfer error interrupt
15:12	OCKDV[3:0]	output clock delay value. These bits only useful when OCKDEN is enabled.
11:8	FTL[3:0]	FIFO threshold level These bits are useful in normal mode, the threshold number of bytes in the FIFO that will cause the FIFO threshold flag to be set. In normal write mode (FMOD = 00): 0: FT is set if there are 1 or more free bytes available to be written to in the FIFO 1: FT is set if there are 2 or more free bytes available to be written to in the FIFO ...

		15: FT is set if there are 16 free bytes available to be written to in the FIFO In normal read mode (FMOD = 01): 0: FT is set if there are 1 or more valid bytes that can be read from the FIFO 1: FT is set if there are 2 or more valid bytes that can be read from the FIFO ... 15: FT is set if there are 16 valid bytes that can be read from the FIFO If DMAEN = 1, then the DMA controller for the corresponding channel must be disabled before changing the FTL value.
7:6	Reserved	Must be kept at reset value.
5	OCKDEN	Output clock delay is enabled when reading data from flash. 0: SCK delay disabled 1: SCK delay enabled
4	SSAMPLE	Sample delay QSPI samples data 1/2 SCK clock cycle after flash memory drives data by default. Taking the delay of external signals into account, this bit can be configured to allow the data to be sampled later. 0: No delay 1: 1/2 cycle delay These bits can be modified only when BUSY = 0.
3	TMOUTEN	Timeout counter enable In memory map mode (FMOD = 11). When setting this bit, the chip select output (CSN) will be high if there is no access after a certain amount of time which specified by TMOUTCYC[15:0]. 0: Disable the timeout counter, the chip select (CSN) will remain low after an access in memory map mode. 1: Enable the timeout counter, the chip select (CSN) will be high if there is no access to flash memory after TMOUTCYC[15:0] cycles in memory map mode. This bit can be modified only when BUSY = 0.
2	DMAEN	DMA enable DMA can be used to transfer data in normal mode. And when FT is set, DMA transfers initiates. 0: disable DMA 1: enable DMA
1	ABORT	Abort request This bit is used to stop the current command. It is automatically cleared once the abort is completed. In read polling mode or memory map mode, when setting this bit, the RPMS bit or the DMAEN bit will be cleared. 0: No abort request 1: Abort request

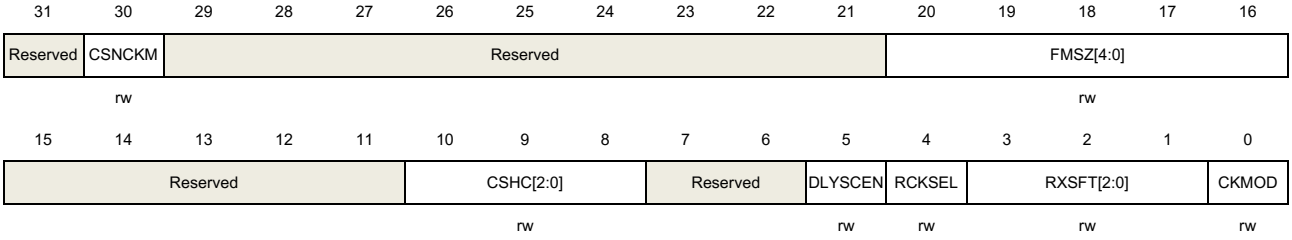
0	QSPIEN	the QSPI enable 0: disable QSPI 1: enable QSPI

21.9.2. Device configuration register (QSPI_DCFG)

Address offset: 0x04

Reset value: 0x001F 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30	CSNCKM	Select whether the CSN falls and rises one or two SCK clock cycles before the first SCK effective rising edge and after the last SCK effective rising edge. 0: 1 cycle 1: 2 cycles
29:21	Reserved	Must be kept at reset value.
20:16	FMSZ[4:0]	Flash memory size These bits define the size of external memory. And the number of bytes is $2^{[FMSZ+1]}$. FMSZ+1 is the number of address bits in the flash memory. In normal mode, the flash memory capacity can be up to 4GB, while it is limited to 128MB in memory map mode. These bits can be modified only when BUSY = 0.
15:11	Reserved	Must be kept at reset value.
10:8	CSHC[2:0]	Chip select high cycle The chip select (CSN) must stay high for at least CSHC+1 SCK cycles between two command sequences. 000: CSN must stay high for at least 1 SCK cycle between two flash memory command sequences. 001: CSN must stay high for at least 2 SCK cycles between two flash memory command sequences. ... 111: CSN must stay high for at least 8 SCK cycles between two flash memory command sequences.

		These bits can be modified only when BUSY = 0.
7:6	Reserved	Must be kept at reset value.
5	DLYSCEN	Delay scan enable 0: Disable 1: Enable Note: This bit only takes effect when RCKSEL=0. The SCK of QSPI is only output when needed, and is not continuous. Enable this bit, enable continuous output of SCK to CPDM, and fine-tune the clock through CPDM to find the accurate sampling point. Please refer to CPDM for more details. After adjusting the clock with CPDM, DLYSCEN should be set to 0.
4	RCKSEL	Receive clock select 0: SCK 1: DQS Select the receiving clock source is the SCK generated inside the QSPI or the DQS provided by an external device. When the DQS clock is selected, CPDM can also be used for clock adjustment, and there is no need to set DLYSCEN bit.
3:1	RXSFT[2:0]	Shift receive step if the received data delay more than 0.5 cycle. 000: No shift 001: 1 cycle 010: 2 cycles 011: 3 cycles 100: 4 cycles 101: 5 cycles 110: 6 cycles 111: 7 cycles This bit field can be adjusted together with the SSAMPLE bit to modify the receive sampling point.
0	CKMOD	This bit indicates the SCK level when QSPI is free. 0: SCK must stay low when CSN is high. 1: SCK must stay high when CSN is high. This bit can be modified only when BUSY = 0.

21.9.3. Status register (QSPI_STAT)

Address offset: 0x08

Reset value: 0x0000 0004

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	FL[4:0]	Reserved	BUSY	TMOUT	RPMF	FT	TC	TERR
	rw		r	r	r	r	r	r

Bits	Fields	Descriptions
31:13	Reserved	Must be kept at reset value.
12:8	FL[4:0]	FIFO level These bits are used to configure the number of valid bytes which are being stored in the FIFO in normal mode. In memory map mode and in read polling mode, FL is 0.
7:6	Reserved	Must be kept at reset value.
5	BUSY	Busy flag This bit is set when a command is transferring. In normal mode, this bit will be cleared once the command phase is completed and if in read operation of normal mode, FIFO also needs be empty.
4	TMOUT	Timeout flag When the TMOUTEN is set and there is no access to flash memory after TMOUTCYC[15:0] cycles. This bit will be set.
3	RPMF	Read polling match flag This bit is set in read polling mode when the unmasked received data matches the expected value in QSPI_STATMATCH register.
2	FT	FIFO threshold flag In normal mode, when the FIFO threshold has been reached or if the FIFO is not empty after the last read operation from the flash memory, this bit will be set. And it is cleared by hardware as soon as the threshold condition is not true. In read polling mode, this bit will be set. When the status register is read by the external flash, and it is cleared when the QSPI_DATA register is read.
1	TC	Transfer complete flag This bit will be set when the number of data programmed in QSPI_DTLEN register has been transmitted in normal mode or when abort operation is completed.
0	TERR	Transfer error flag In normal mode, when an invalid address has been accessed. This bit will be set

21.9.4. Status clear register (QSPI_STATC)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											TMOUTC	RPMFC	Reserved	TCC	TERRC
											w	w		w	w

Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
4	TMOUTC	Clear timeout flag Writing 1 to this bit clears the TMOUT bit in the QSPI_STAT register.
3	RPMFC	Clear read polling mode match flag Writing 1 to this bit clears the RPMF bit in the QSPI_STAT register.
2	Reserved	Must be kept at reset value.
1	TCC	Clear transfer complete flag Writing 1 to this bit clears the TC bit in the QSPI_STAT register.
0	TERRC	Clear transfer error flag Writing 1 to this bit clears the TERR bit in the QSPI_STAT register.

21.9.5. Data length register (QSPI_DTLEN)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DTLEN[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DTLEN[15:0]															
rw															

Bits	Fields	Descriptions
31:0	DTLEN[31:0]	Data length These bits specify the number of data to be retrieved (value+1) in normal and read polling modes. When in read polling mode, the value of these bits should no greater than 3 (indicating 4 bytes). When these bits are configured as 0xFFFF FFFF in normal mode, it means undefined length, the QSPI will continue transferring data until the end of memory according to FMSZ[4:0] in QSPI_DCFG register. 0x0000 0000: the data length to be transferred is 1 byte 0x0000 0001: the data length to be transferred is 2 bytes 0x0000 0002: the data length to be transferred is 3 bytes 0x0000 0003: the data length to be transferred is 4 bytes ...

0xFFFF FFFD: the data length to be transferred is 4,294,967,294 (4G-2) bytes
0xFFFF FFFE: the data length to be transferred is 4,294,967,295 (4G-1) bytes
0xFFFF FFFF: undefined length, all bytes defined by FMSZ[4:0] in QSPI_DCFG register until the end of flash memory will be transferred. If FMSZ[4:0] is 0x1F, it will continue reading indefinitely.
When in memory map mode, these bits have no effect.
These bits can be modified only when BUSY = 0.

21.9.6. Transfer configuration register (QSPI_TCFG)

Address offset: 0x14
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DDREN	DDRHEN	Reserved	SIOO	FMOD[1:0]		DATAMOD[1:0]		Reserved	DUMYC[4:0]					ALTESZ[1:0]	
rw	rw		rw	rw		rw			rw					rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALTEMOD[1:0]		ADDRSZ[1:0]		ADDRMOD[1:0]		IMOD[1:0]		INSTRUCTION[7:0]							
rw		rw		rw		rw		rw							

Bits	Fields	Descriptions
31	DDREN	Double data rate mode enable 0: disable 1: enable
30	DDRHEN	DDR hold enable This bit delays the data output by 1/4 of the QUADSPI output clock cycle in DDR mode: 0: disable 1： enable
29	Reserved	Must be kept at reset value.
28	SIOO	Send instruction only once mode When IMOD is 00, this bit has no effect. 0: Send instruction on every command sequence 1: Send instruction only for the first command sequence These bits can be modified only when BUSY = 0.
27:26	FMOD[1:0]	Functional mode 00: write operation in normal mode 01: read operation in normal mode 10: read polling mode 11: Memory map mode These bits can be modified when the DMA controller for the corresponding channel

		is disabled if DMAEN is 1. These bits can be modified only when BUSY = 0.
25:24	DATAMOD[1:0]	Data mode These bits define the data phase's mode of operation. It also specifies the dummy phase mode of operation. 00: No data 01: Data on a single line 10: Data on two lines 11: Data on four lines These bits can be modified only when BUSY = 0.
23	Reserved	Must be kept at reset value.
22:18	DUMYC[4:0]	Number of dummy cycles These bits define the duration of the dummy phase. These bits can be modified only when BUSY = 0.
17:16	ALTESZ[1:0]	Alternate byte size 00: 8-bit alternate byte 01: 16-bit alternate bytes 10: 24-bit alternate bytes 11: 32-bit alternate bytes These bits can be modified only when BUSY = 0.
15:14	ALTEMOD[1:0]	Alternate byte mode 00: No alternate bytes 01: Alternate bytes on a single line 10: Alternate bytes on two lines 11: Alternate bytes on four lines These bits can be modified only when BUSY = 0.
13:12	ADDRSZ[1:0]	Address size 00: 8-bit address 01: 16-bit address 10: 24-bit address 11: 32-bit address These bits can be modified only when BUSY = 0.
11:10	ADDRMOD[1:0]	Address mode 00: No address 01: Address on a single line 10: Address on two lines 11: Address on four lines These bits can be modified only when BUSY = 0.
9:8	IMOD[1:0]	Instruction mode 00: No instruction

		01: Instruction on a single line 10: Instruction on two lines 11: Instruction on four lines These bits can be modified only when BUSY = 0.
7:0	INSTRUCTION[7:0]	Instruction Command information to be send to the flash memory. These bits can be modified only when BUSY = 0.

21.9.7. Address register (QSPI_ADDR)

Address offset: 0x18

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR[15:0]															
rw															

Bits	Fields	Descriptions
31:0	ADDR[31:0]	Address to be send to the external Flash memory When BUSY=1 or in memory mapped mode, writing values to these bits will be ignored.

21.9.8. Alternate bytes register (QSPI_ALTE)

Address offset: 0x1C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ALTE[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALTE[15:0]															
rw															

Bits	Fields	Descriptions
31:0	ALTE[31:0]	Alternate Bytes Optional data to be send to the flash memory follow by address. These bits can be modified only when BUSY = 0

21.9.9. Data register (QSPI_DATA)

Address offset: 0x20

Reset value: 0x0000 0000

This register can be accessed by byte(8-bit), half-word(16-bit) and word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA[15:0]															
rw															

Bits	Fields	Descriptions
31:0	DATA[31:0]	Data that will be interacting with flash memory. In write operation of normal mode, the data written to this register will be stored in the FIFO before sending to the flash memory. If the FIFO is full, a write operation will be stopped and wait until the FIFO has enough space. In read operation of normal mode, the data received from the flash memory can be read in this register. If the bytes in the FIFO are less than the requested bytes by the read command, and BUSY=1, the read operation will be stopped and wait until enough data is stored in the FIFO or until the transfer is completed. In read polling mode, this register contains the last data read from the flash memory.

21.9.10. Status mask register (QSPI_STATMK)

Address offset: 0x24

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MASK[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASK[15:0]															
rw															

Bits	Fields	Descriptions
31:0	MASK[31:0]	Status mask in read polling mode Mask of the status bytes received from the flash memory in read polling mode. For bit n of MASK[31:0]: 0: Bit n of the data received cannot match 1: Bit n of the data received must match These bits can be modified only when BUSY = 0.

21.9.11. Status match register (QSPI_STATMATCH)

Address offset: 0x28

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MATCH[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MATCH[15:0]															
rw															

Bits	Fields	Descriptions
31:0	MATCH[31:0]	Status match in read polling mode Expected value to be compared with the value in the QSPI_STATMK register to get a match. These bits can be modified only when BUSY = 0.

21.9.12. Interval register (QSPI_INTERVAL)

Address offset: 0x2C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTERVAL[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	INTERVAL[15:0]	Interval cycle Number of SCK cycles between two read commands in read polling mode. These bits can be modified only when BUSY = 0.

21.9.13. Timeout register (QSPI_TMOUT)

Address offset: 0x30

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMOUTCYC[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	TMOUTCYC[15:0]	Timeout cycle When the FIFO is full in memory map mode, these bits indicate how many SCK cycles the QSPI waits for next access. In this duration, CSN keeps low. These bits can be modified only when BUSY = 0.

21.9.14. FIFO flush register (QSPI_FLUSH)

Address offset: 0x34
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															FLUSH
															w

Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	FLUSH	Used to flush all QSPI internal FIFO.

21.9.15. TMR register (QSPI_TMR)

Address offset: 0x38
Reset value: 0x0000 0055

This register has to be accessed by word (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TMR_ST3	TMR_EN3	TMR_ST2	TMR_EN2	TMR_ST1	TMR_EN1	TMR_ST0	TMR_EN0
								rc_w1	rw	rc_w1	rw	rc_w1	rw	rc_w1	rw

Bits	Fields	Descriptions

31:8	Reserved	Must be kept at reset value
7	TMR_ST3	QSPI_FLUSH register TMR error status 0: Normal 1: Error
6	TMR_EN3	QSPI_FLUSH register TMR error output enable 0: Disable 1: Enable
5	TMR_ST2	QSPI_TCFG register TMR error status 0: Normal 1: Error
4	TMR_EN2	QSPI_TCFG register TMR error output enable 0: Disable 1: Enable
3	TMR_ST1	QSPI_DTLEN register TMR error status 0: Normal 1: Error
2	TMR_EN1	QSPI_DTLEN register TMR error output enable 0: Disable 1: Enable
1	TMR_ST0	QSPI_CTL register TMR error status 0: Normal 1: Error
0	TMR_EN0	QSPI_CTL register TMR error output enable 0: Disable 1: Enable

21.10. CPDM register definition

CPDM base address: 0x4900 1800

21.10.1. Control register (CPDM_CTL)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	DLSEN	CPDMEN
	rw	rw

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	DLSEN	CPDM delay line sample module enable bit 0: disable CPDM delay line sample module 1: enable CPDM delay line sample module
0	CPDMEN	CPDM enable bit 0: disable CPDM 1: enable CPDM

21.10.2. Configuration register (CPDM_CFG)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DLLENF	Reserved			DLLEN[11:0]											
r				r											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	DLSTCNT[6:0]							Reserved				CPSEL[3:0]			
	rw											rw			

Bits	Fields	Descriptions
31	DLLENF	Valid mark of delay line length This bit is set when the delay line length value contained in DLLEN[11:0] corresponds to one input clock cycle after the DLSTCNT[6:0] bit field is modified. 0: The length value in DLLEN[11:0] is invalid. 1: The length value in DLLEN[11:0] is valid.
30:28	Reserved	Must be kept at reset value.
27:16	DLLEN[11:0]	Delay line length 12-unit delay values sampled on the rising edge of the input clock Only valid if DLLENF = 1.
15	Reserved	Must be kept at reset value.
14:8	DLSTCNT[6:0]	Defines a delay step count for a unit delay UNIT These bits can be written only when DLSEN = 1. 0000000: UNIT delay = Initial delay 0000001: UNIT delay = Initial delay + 1 * Delay step …

		1111111: UNIT delay = Initial delay + 127 * Delay step
7:4	Reserved	Must be kept at reset value.
3:0	CPSEL[3:0]	Output clock phase selection These bits can be written only when DLSEN = 1. 0000: Output clock phase = input clock 0001: Output clock phase = input clock + 1 * UNIT delay .. 1100: Output clock phase = input clock + 12 * UNIT delay 1101 ~ 1111: Reserved

22. Controller area network (CAN)

22.1. Overview

A CAN module is a CAN cluster implemented with two to three DTM_CANs, an integrated intelligent CAN RAM controller, and a CAN_SYSREG for the DTM_CANs integration configuration to the CAN cluster.

The DTM_CAN includes a Bosch's M_CAN, along with an additional DMU.

The M_CAN performs communication according to ISO11898-1:2015. Additional transceiver hardware is required for connection to the physical layer.

Table 22-1. CAN module feature configuration

Module (Cluster)	Instance type and number	CAN FD support
CAN0	DTM_CAN0	YES
	DTM_CAN1	YES
CAN1	DTM_CAN2	YES
	DTM_CAN3	YES
	DTM_CAN4	YES
CAN2	DTM_CAN5	YES
	DTM_CAN6	YES
	DTM_CAN7	YES

Table 22-2. CAN cluster memory map

ADDR Space	Start ADDR (offset value for the base address)	End ADDR	Size
MRAM	0x0000	0x17FF	6KB
CAN_SYSREG	0x1800	0x1FFF	2KB
DTM_CAN0 / 2 / 5	0x2000	0x27FF	2KB
DTM_CAN1 / 3 / 6	0x2800	0x2FFF	2KB
DTM_CAN4 / 7	0x3000	0x37FF	2KB

A CAN cluster consists of up to 3 DTM_CANs. The distribution of the DTM_CAN section is shown in *Table 22-8. CAN Cluster register overview*

Register Name	Symbol	Address Offset	Reset Value
MRAM initialization register	CAN_MRAM	0x00	0x0000 0000
Disable modification on read register	CAN_DMR	0x20	0x0000 0000
Clock stop request register	CAN_CSR	0x24	0x0000 0000
Clock stop acknowledge register	CAN_CSA	0x28	0x0000 0008

Register Name	Symbol	Address Offset	Reset Value
Control register	CAN_CTL	0x70	0x0000 0000
Memory error control register	CAN_MERRCTL	0x80	0x8000 0000
Correctable error record register	CAN_CERRR0 ~ CAN_CERRR3	0x84 + 0x04 * x (x = 0 to 3)	0x0000 0000
Noncorrectable error record register	CAN_NCERRR	0x94	0x0000 0000
Memory error status register	CAN_MERRSTAT	0x9C	0x0000 0000

Table 22-9. DTM_CANx (x = 0...7) register distribution (2KB)

.

The (Message RAM) MRAM memory map refers to *Table 22-3. Shared memory map*.

22.2. Characteristics

- Up to 8 DTM_CAN nodes are implemented in three CAN clusters.
- Implementation Bosch M_CAN Module IP, together with add-on DMU.
- Bit rates up to 1 Mbit/s in classical CAN mode, 8 Mbit/s for CAN FD mode.
- Multiple CANs share the same Message RAM with flexible arbiter and ECC.
- DMA interface, automatic message handling without CPU.
- Separate interrupts per DTM_CAN node: M_CAN interrupt line 0, M_CAN interrupt line 1, DMU interrupt.
- A noncorrectable error interrupt and a correctable error interrupt for each CAN cluster.

22.3. Function overview

22.3.1. Block diagram

Figure 22-1. CAN cluster diagram below provides details on the internal configuration of the CAN cluster.

Figure 22-1. CAN cluster diagram

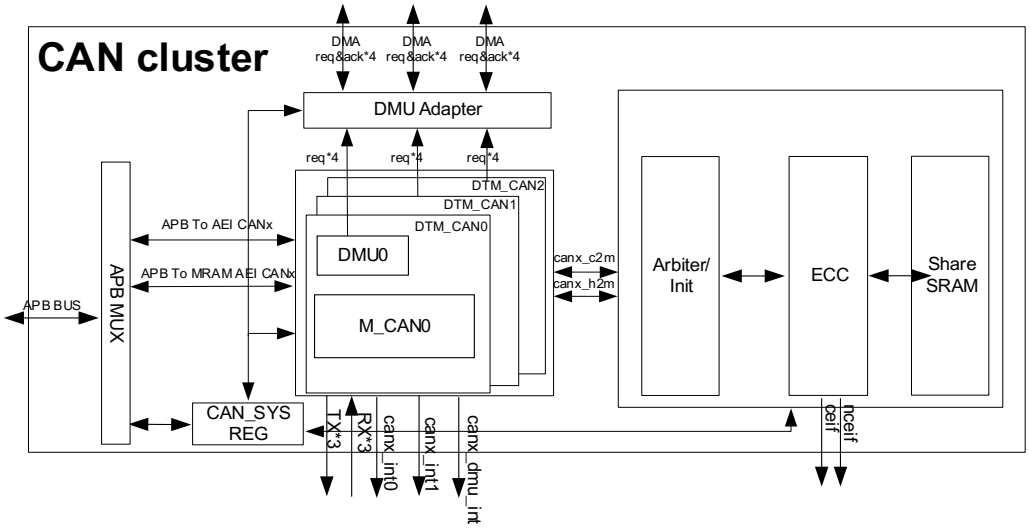
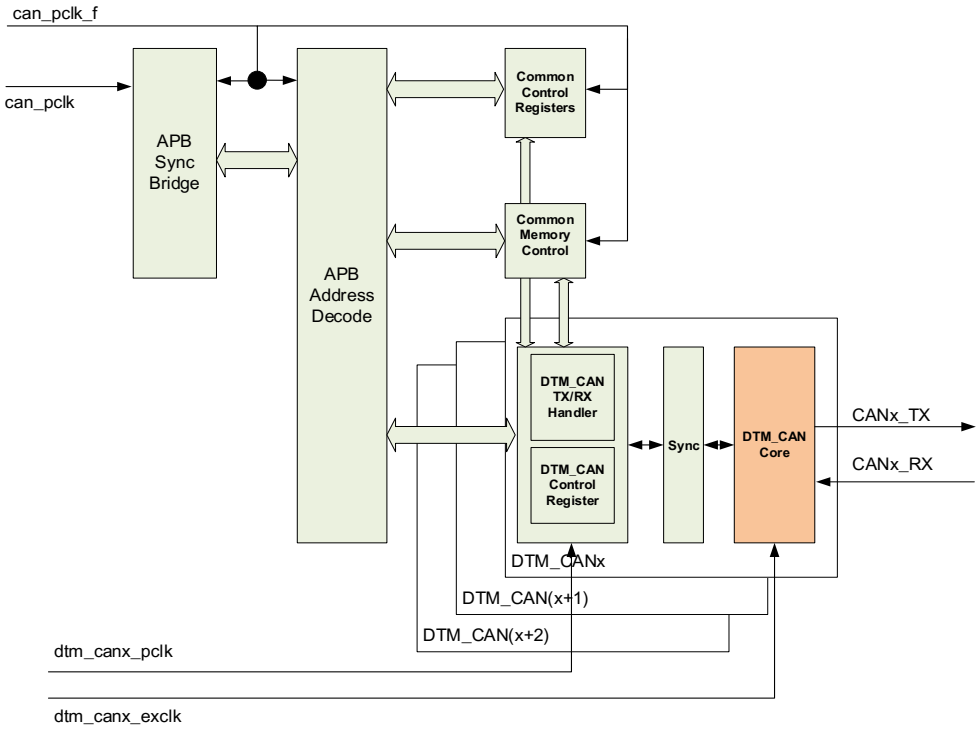


Figure 22-2. CAN cluster clock structure diagram below provides details on the clock structure configuration of the CAN cluster.

Figure 22-2. CAN cluster clock structure diagram



The frequency of can_hclk (bus interface clock) must not be lower than that of can_cclk (CAN protocol clock), but it may share the same frequency or may be significantly higher.

In the diagram, dtm_canx_exclk corresponds to can_cclk signal of each M_CAN in DTM_CAN, dtm_canx_pclk corresponds to can_hclk signal of each M_CAN in DTM_CAN.

The dtm_canx_exclk clock source contains IRC48M, HXTAL, HCLK, PCLK2. dtm_canx_exclk and dtm_canx_pclk are enabled by RCU_APB2EN_ADD.DTM_CANxEN.

can_pclk clock source is the PCLK2, can_pclk_f clock source is the HCLK, they are enabled by:

RCU_APB2EN_ADD.DTM_CANxEN (x=0,1) for CAN cluster0.

RCU_APB2EN_ADD.DTM_CANxEN (x=2,3,4) for CAN cluster1.

RCU_APB2EN_ADD.DTM_CANxEN (x=5,6,7) for CAN cluster2.

Note:

All dtm_canx_pclk in a can cluster must be enable at the same time (for GD32A71xx devices).

22.3.2. Modular CAN (M_CAN) cores

Reference to ***Modular controller area network (M_CAN)***.

22.3.3. DMU Add-On

Reference to ***DMA interface unit (DMU)***.

22.3.4. Power saving mode

M_CAN can be put into a power-down mode through CAN_CSR.CSRx or by using the CC control register CAN_M_CAN_CCCR.CSR. As long as the clock stop request signal is active (CAN_CSR.CSRx), bit CAN_M_CAN_CCCR.CSR will be read as 1.

Once all pending transmission requests are completed, M_CAN waits to detect an idle state on the bus. Then, M_CAN sets CAN_M_CAN_CCCR.INIT to 1 to prevent any further CAN transmissions. At this point, M_CAN acknowledges that it is ready for power down by setting output signal m_can_clkstop_ack to 1 (connect to CAN_CSA.CSAx) and setting CAN_M_CAN_CCCR.CSA to 1. Before the module clocks m_can_hclk and m_can_cclk are switched off, further register accesses can be made except for CAN_M_CAN_CCCR.INIT which is held at one. The module clock inputs m_can_hclk and m_can_cclk can now be turned off.

To exit power down mode, the application must turn on the module clocks before resetting the signal CAN_CSR.CSRx or the CC control register flag CAN_M_CAN_CCCR.CSR. M_CAN will acknowledge this operation by resetting the output signal CAN_CSA.CSAx and resetting CAN_M_CAN_CCCR.CSA. After this, the application can restart CAN communication by resetting the bit CAN_M_CAN_CCCR.INIT.

22.3.5. Share memory map

Table 22-3. Shared memory map shows the shared memory map versus CAN cluster transmit, receive, and filters elements.

Table 22-3. Shared memory map

Start Address Offset (byte)	End Address Offset (byte)	CAN block		Comment
CAN cluster				
0x0000	0x17FF	M_CANs in a CAN cluster	Standard Filters	This available physical memory space could be configured for Filters / FIFO / Buffer. This memory space can be flexible, assigned to any of the implemented M_CAN modules.
			Extended Filters	
			Rx FIFO 0	
			Rx FIFO 1	
			Rx Buffer	
			Tx Event FIFO	
			Tx Buffers	

Note: There is no write access protection from a CAN module to the message memory space. The start address for every memory section is to be configured by the user in the relevant M_CAN modules registers. However there is no hardware check in the RAM interface for the start address consistency.

22.3.6. Frame capacity

Table 22-4. Minimum frame capacity for each CAN / DTM_CAN shows the minimum frame capacity for each CANx or DTM_CANx under different configurations.

Table 22-4. Minimum frame capacity for each CAN / DTM_CAN

CANx	The data byte number of each frame (ranging from 0 to 64 bytes)																
	0	1	2	3	4	5	6	7	8	12	16	20	24	32	48	64	Byte
CAN0 (DTM_CAN0/1)	320 (160)	320 (160)	320 (160)	320 (160)	320 (160)	320 (160)	320 (160)	320 (160)	320 (160)	256 (128)	213 (106)	182 (91)	160 (80)	128 (64)	91 (45)	71 (35)	Frame
CAN1 (DTM_CAN2/3/4)	320 (106)	320 (106)	320 (106)	320 (106)	320 (106)	320 (106)	320 (106)	320 (106)	320 (106)	230 (76)	192 (64)	164 (54)	144 (48)	115 (38)	82 (27)	64 (21)	Frame
CAN2 (DTM_CAN5/6/7)	320 (106)	320 (106)	320 (106)	320 (106)	320 (106)	320 (106)	320 (106)	320 (106)	320 (106)	230 (76)	192 (64)	164 (54)	144 (48)	115 (38)	82 (27)	64 (21)	Frame

Note:

1. Each CANx shares a total SRAM space of 6KB, which is 1536 words. Considering the maximum number of filters for each DTM_CANx to be 128 words, the space after subtracting 128 filters for each DTM_CANx is used as the data-frame space. CAN0, CAN1, and CAN2 have 1280, 1152, and 1152 words remaining respectively.

2. In the table, 320(160). Here, 320 represents the frame capacity of each CANx, and 160 represents the average frame capacity for each DTM_CANx.
3. Each frame of data has a frame header of 2 words, followed by the DATA. The amount of DATA bytes can range from 0 to 64 bytes. However, the minimum space occupied by the DATA in each frame of data should be configured as 8 bytes. Therefore, when the DATA is between 0 and 8 bytes, the frame capacity remains the same.

Table 22-5. Maximum frame capacity for each CAN / DTM CAN shows the maximum frame capacity for each CANx or DTM_CANx under different configurations.

Table 22-5. Maximum frame capacity for each CAN / DTM_CAN

CANx	The data byte number of each frame (ranging from 0 to 64 bytes)																
	0	1	2	3	4	5	6	7	8	12	16	20	24	32	48	64	Byte
CAN0 (DTM_CAN0/1)	384 (192)	384 (192)	384 (192)	384 (192)	384 (192)	384 (192)	384 (192)	384 (192)	384 (192)	307 (153)	256 (128)	219 (109)	192 (96)	153 (76)	109 (54)	85 (42)	Frame
CAN1 (DTM_CAN2/3/4)	384 (128)	384 (128)	384 (128)	384 (128)	384 (128)	384 (128)	384 (128)	384 (128)	384 (128)	307 (102)	256 (85)	219 (73)	192 (64)	153 (51)	109 (36)	85 (28)	Frame
CAN2 (DTM_CAN5/6/7)	384 (128)	384 (128)	384 (128)	384 (128)	384 (128)	384 (128)	384 (128)	384 (128)	384 (128)	307 (102)	256 (85)	219 (73)	192 (64)	153 (51)	109 (36)	85 (28)	Frame

Note:

1. Each CANx shares a total of 6KB SRAM space, which is equivalent to 1536 words. Without considering the filters of each DTM_CANx, each CAN has 1536 words of space available as the data-frame space.
2. In the table, 384(192), 384 represents the frame capacity of each CANx, and 192 represents the average frame capacity for each DTM_ CANx.
3. Each data frame has a frame header of 2 words, followed by the DATA. The amount of DATA bytes can range from 0 to 64 bytes. However, the minimum space occupied by the DATA in each frame must be configured as 8 bytes. Therefore, when the DATA size is between 0 and 8 bytes, the frame capacity remains the same.

22.3.7. CAN RAM Arbiter

The module acts as a dynamic arbiter, allocating access to shared CAN memory among various DTM_CAN nodes and the CPU within a single CAN cluster.

The dynamic arbitration mechanism ensures that when all master nodes and the CPU request access simultaneously, the CPU obtains 50% of the bandwidth, while the remaining 50% is shared by DTM_CAN nodes. If no other master nodes request access, the bandwidth is dynamically adjusted to the currently requesting master node. It should be noted that since the arbiter limits the CPU's bandwidth to 50%, the CPU may be forced to wait when accessing shared memory.

When a single master node accesses the SRAM, each read/write command requires two

cycles: one for the address and another for the data. The arbiter uses a pseudo address prefetch mechanism, allowing the overlap of the address cycle of one master node with the data cycle of another when multiple master nodes access the SRAM, thereby saving multiple clock cycles. When the peripheral module is enabled, it becomes the request signal for each master node. Each master node must wait at least one clock cycle before being granted access. The arbiter ensures that the CPU's wait time does not exceed one clock cycle, guaranteeing that the CPU always obtains 50% of the bandwidth. A timeslot is defined as one host clock cycle.

22.3.8. Interrupts

Each CAN cluster outputs interrupts from up to 3 DTM_CAN nodes. In addition to the DTM_CAN interrupts, there is also a noncorrectable error interrupt and a correctable error interrupt for each CAN cluster. The sources of interrupts are as follows:

- CANx (x=0...2) noncorrectable error.
- CANx (x=0...2) correctable error.
- DTM_CANx (x=0...7) M_CAN interrupt line 0.
- DTM_CAN x (x=0...7) M_CAN interrupt line 1.
- DTM_CAN x (x=0...7) DMU interrupt.

Table 22-6. Interrupt events

Interrupt event		Flag		Enable control			
		Bit	Register	Enable bit	Control bit	Enable register	Control register
DTM_CANx (x=0..7) M_CAN interrupt line 0 & 1	Protocol error in data phase	PED	CAN_M_CAN_IR	PEDE	PEDL	CAN_M_CAN_IE	CAN_M_CAN_ILS
	Protocol error in arbitration phase	PEA		PEAE	PEAL		
	Message RAM Watchdog	WDI		WDIE	WDIL		
	Bus_Off status changed	BO		BOE	BOL		
	Error_Warning status changed	EW		EWE	EWL		
	Error_Passive status changed	EP		EPE	EPL		
	Overflow of CAN error logging counter	ELO		ELOE	ELOL		
	Message stored to dedicated Rx Buffer	DRX		DRXE	DRXL		
	Timeout occurred	TOO		TOOE	TOOL		
	Message RAM access failure	MRAF		MRAFE	MRAFL		
	Timestamp wraparound	TSW		TSWE	TSWL		
	Tx Event FIFO event lost	TEFL		TEFLE	TEFLL		
	Tx Event FIFO full	TEFF		TEFFE	TEFFL		
	Tx Event FIFO watermark reached	TEFW		TEFWE	TEFWL		
	Tx Event FIFO new entry	TEFN		TEFNE	TEFNL		

Interrupt event		Flag		Enable control			
		Bit	Register	Enable bit	Control bit	Enable register	Control register
	Tx FIFO empty	TFE		TFEE	TFEL		
	High priority message	HPM		HPME	HPML		
	Rx FIFO 0/1 message lost	RFxL (x=0...1)		RFxLE (x=0...1)	RFxLL (x=0...1)		
	Rx FIFO 0/1 full	RFxF (x=0...1)		RFxFE (x=0...1)	RFxFL (x=0...1)		
	Rx FIFO 0/1 watermark reached	RFxW (x=0...1)		RFxWE (x=0...1)	RFxWL (x=0...1)		
	Rx FIFO 0/1 new message	RFxN (x=0...1)		RFxNE (x=0...1)	RFxNL (x=0...1)		
	Transmission cancellation finished	TCF & CFx (x=0...31)	CAN_M_CAN_IR & CAN_M_CAN_TXBCF	TCFE & CFIEx (x=0...31)	TCFL	CAN_M_CAN_IE & CAN_M_CAN_TXBCIE	
	Transmission completed	TC & TOx (x=0...31)	CAN_M_CAN_IR & CAN_M_CAN_TXBTO	TCE & TIEx (x=0...31)	TCL	CAN_M_CAN_IE & CAN_M_CAN_TXBTIE	
DTM_CANx (x=0...7) DMU interrupt	Illegal Access while CCE mode, TX event element related events, RX0/1 element related events, TX element related events, bit error events	All bits	CAN_DMU_DMUIR	All bites		CAN_DMU_DMUIE	
CANx (x=0...2) noncorrectable error		NCDEF | NCE_AEDCF | NCE_EAEF	CAN_MERRSTAT	NCEIE		CAN_MERRCTL	
CANx (x=0...2) correctable error		CEF	CAN_MERRSTAT	CEIE		CAN_MERRCTL	

Table 22-7. CAN and DTM_CAN interrupt mapping

CAN Number	DTM_CAN Number	Interrupt Name
CAN0	DTM_CAN0	dtm_can0_int0
		dtm_can0_int1
		dtm_can0_dmu_int
	DTM_CAN1	dtm_can1_int0
		dtm_can1_int1

CAN Number	DTM_CAN Number	Interrupt Name
CAN1		dtm_can1_dmu_int
	Common	can0_ce_int
		can0_nce_int
	DTM_CAN2	dtm_can2_int0
		dtm_can2_int1
		dtm_can2_dmu_int
	DTM_CAN3	dtm_can3_int0
		dtm_can3_int1
		dtm_can3_dmu_int
	DTM_CAN4	dtm_can4_int0
		dtm_can4_int1
		dtm_can4_dmu_int
	Common	can1_ce_int
		can1_nce_int
CAN2	DTM_CAN5	dtm_can5_int0
		dtm_can5_int1
		dtm_can5_dmu_int
	DTM_CAN6	dtm_can6_int0
		dtm_can6_int1
		dtm_can6_dmu_int
	DTM_CAN7	dtm_can7_int0
		dtm_can7_int1
		dtm_can7_dmu_int
	Common	can2_ce_int
		can2_nce_int

22.4. Example for a typical configuration flow of CAN

After a power-on reset or system reset, the application can configure and start the CAN module according to the following typical workflow:

■ Configure the various clock sources for the CAN module and enable each clock:

Configure the CAN clock, configure the DTM_CAN's bus clock and protocol clock.

■ Configure the communication interface:

Configure the GPIO and AFIO modules, mapping the corresponding functional pins to the AF function.

■ Enter Initialization Mode:

1) Enable Initialization Mode via CAN_M_CAN_CCCR.INIT and CAN_M_CAN_CCCR.CCE.
2) Configure RAM space allocation via CAN_M_CAN_SIDFC, CAN_M_CAN_XIDFC, CAN_M_CAN_RXF0C, CAN_M_CAN_RXF1C, CAN_M_CAN_RXBC, CAN_M_CAN_TXEFC, and CAN_M_CAN_TXBC.
3) Configure the reception filters as the allocated RAM space (if receiving is needed). If using DMA functionality, read the RX0/1 FIFO elements via the DMU RX0/1 element, and read the Tx Event FIFO elements via the DMU TX event element.
4) Configure the Tx buffer according to the allocated RAM space (if transmitting is needed). If using DMA functionality, write to the TX FIFO / queue elements via the DMU TX element.
5) Configure the communication rate parameters using CAN_M_CAN_DBTP and CAN_M_CAN_NBTP. If necessary, use CAN_M_CAN_CCCR.EFBI and CAN_M_CAN_CCCR.PXHD to enable bus integration state support.
6) Configure the buffer and FIFO parameters related to transmission using CAN_M_CAN_TXBC, CAN_M_CAN_TXESC, CAN_M_CAN_TXEFC. If needed, suspend transmission by setting CAN_M_CAN_CCCR.TXP or enable transmission cancellation via CAN_M_CAN_TXBCR.
7) Configure the buffer and FIFO parameters related to reception using CAN_M_CAN_RXF0C, CAN_M_CAN_RXBC, CAN_M_CAN_RXF1C, CAN_M_CAN_RXESC.
8) Enable CAN FD mode if required:

Select the CAN FD protocol via the CAN_M_CAN_CCCR.NISO bit.

Enable CAN FD mode using the CAN_M_CAN_CCCR.FDOE bit.

If necessary, perform transmission delay compensation configuration for CAN FD using CAN_M_CAN_DBTP.TDC and CAN_M_CAN_PSR.TDCV.

9) Configure interrupts using CAN_M_CAN_IE, CAN_M_CAN_ILS, CAN_M_CAN_ILE, CAN_M_CAN_TXBTIE, and CAN_M_CAN_TXBCIE.
10) If ECC functionality is required, configure ECC-related registers (MCAN_MERRCTL, MCAN_CERRRx, MCAN_NCERRR, MCAN_MERRSTAT).

■ Exit Initialization Mode. Afterward, the CAN node will resynchronize with the CAN bus, awaiting reception or handling transmission requests via writing transmission requests to CAN_M_CAN_TXBAR.

22.5. CAN cluster register definition

CAN0 base address: 0x4001 8000

CAN1 base address: 0x4001 C000

CAN2 base address: 0x4002 8000

CAN_SYSREG address offset refers to ***Table 22-2. CAN cluster memory map***.

DTM_CANx (x=0...7) address offset refers to ***Table 22-2. CAN cluster memory map***.

The distribution of the DTM_CAN section is shown in ***Table 22-8. CAN Cluster register overview***

Register Name	Symbol	Address Offset	Reset Value
MRAM initialization register	CAN_MRAM	0x00	0x0000 0000
Disable modification on read register	CAN_DMR	0x20	0x0000 0000
Clock stop request register	CAN_CSR	0x24	0x0000 0000
Clock stop acknowledge register	CAN_CSA	0x28	0x0000 0008
Control register	CAN_CTL	0x70	0x0000 0000
Memory error control register	CAN_MERRCTL	0x80	0x8000 0000
Correctable error record register	CAN_CERRR0 ~ CAN_CERRR3	0x84 + 0x04 * x (x = 0 to 3)	0x0000 0000
Noncorrectable error record register	CAN_NCERRR	0x94	0x0000 0000
Memory error status register	CAN_MERRSTAT	0x9C	0x0000 0000

Table 22-9. DTM_CANx (x = 0...7) register distribution (2KB)

Table 22-8. CAN Cluster register overview

Register Name	Symbol	Address Offset	Reset Value
MRAM initialization register	CAN_MRAM	0x00	0x0000 0000
Disable modification on read register	CAN_DMR	0x20	0x0000 0000
Clock stop request register	CAN_CSR	0x24	0x0000 0000
Clock stop acknowledge register	CAN_CSA	0x28	0x0000 0008
Control register	CAN_CTL	0x70	0x0000 0000
Memory error control register	CAN_MERRCTL	0x80	0x8000 0000
Correctable error record register	CAN_CERRR0 ~ CAN_CERRR3	0x84 + 0x04 * x (x = 0 to 3)	0x0000 0000
Noncorrectable error record register	CAN_NCERRR	0x94	0x0000 0000
Memory error status register	CAN_MERRSTAT	0x9C	0x0000 0000

Table 22-9. DTM_CANx (x = 0...7) register distribution (2KB)

Address offset	Name
0x0000 - 0x00FC	M_CAN address range
0x0100 - 0x01FC	Reserved
0x0200 - 0x07FC	DMU extension

22.5.1. CAN system register (CAN_SYSREG)

MRAM initialization register (CAN_MRAM)

Address offset: 0x00
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															IRF
															r

Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	IRF	CAN share message SRAM initialization finish 0: Initialization not finish 1: Initialization finish

Disable modification on read register (CAN_DMR)

Address offset: 0x20
Reset value: 0x0000 0000

Input signal DMORDx controls the "disable modification on read" function. With DMORDx = '1', resetting of bit field CAN_M_CAN_ECR.CEL and bits CAN_M_CAN_PSR.PXE, CAN_M_CAN_PSR.RFDF, CAN_M_CAN_PSR.RBRS, CAN_M_CAN_PSR.RESI as well as setting of bit fields CAN_M_CAN_PSR.LEC and CAN_M_CAN_PSR.DLEC by a read access is disabled.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													DMORD2	DMORD1	DMORD0
													rw	rw	rw

Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2	DMORD2	"disable modification on read" function enable for M_CAN of the third DTM_CAN in a CAN cluster 0: Disable "disable modification on read" function

		1：Enable "disable modification on read" function This bit is invalid for CAN0.
1	DMORD1	"disable modification on read" function enable for M_CAN of the second DTM_CAN in a CAN cluster 0：Disable "disable modification on read" function 1：Enable "disable modification on read" function
0	DMORD0	"disable modification on read" function enable for M_CAN of the first DTM_CAN in a CAN cluster 0：Disable "disable modification on read" function 1：Enable "disable modification on read" function

Clock stop request register (CAN_CSR)

Address offset: 0x24

Reset value: 0x0000 0000

Note: CAN_M_CAN_CCCR.CSR and CAN_M_CAN_CCCR.CSA can be reset only when the corresponding CSRx bit in CAN_CSR register is 0.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													CSR2	CSR1	CSR0
													rw	rw	rw

Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2	CSR2	Clock stop request signal for M_CAN of the third DTM_CAN in a CAN cluster 0：Clock stop request is inactive 1：Clock stop request is active This bit is invalid for CAN0.
1	CSR1	Clock stop request signal for M_CAN of the second DTM_CAN in a CAN cluster 0：Clock stop request is inactive 1：Clock stop request is active
0	CSR0	Clock stop request signal for M_CAN of the first DTM_CAN in a CAN cluster 0：Clock stop request is inactive 1：Clock stop request is active

Clock stop acknowledge register (CAN_CSA)

Address offset: 0x28

Reset value: 0x0000 000X (For CAN0 is 0x0000 000C, for CAN1 and CAN2 is 0x0000 0008)

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													CSA2	CSA1	CSA0
													r	r	r

Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2	CSA2	Clock stop acknowledge for M_CAN of the third DTM_CAN in a CAN cluster 0: Clock stop acknowledge signal is inactive 1: Clock stop acknowledge signal is active This bit is invalid for CAN0.
1	CSA1	Clock stop acknowledge for M_CAN of the second DTM_CAN in a CAN cluster 0: Clock stop acknowledge signal is inactive 1: Clock stop acknowledge signal is active
0	CSA0	Clock stop acknowledge for M_CAN of the first DTM_CAN in a CAN cluster 0: Clock stop acknowledge signal is inactive 1: Clock stop acknowledge signal is active

Control register (CAN_CTL)

Address offset: 0x70

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		ECRWEN	Reserved												
		rw													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	DMAEN2	DMAEN1	DMAEN0	Reserved											
	rw	rw	rw												

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29	ECRWEN	CAN_MERRCTL register write access enable 0: CAN_MERRCTL register can not be written 1: CAN_MERRCTL register can be written
28:15	Reserved	Must be kept at reset value.

14	DMAEN2	DMA request enable for M_CAN of the third DTM_CAN in a CAN cluster 0: DMA request is disabled 1: DMA request is enabled This bit is invalid for CAN0.
13	DMAEN1	DMA request enable for M_CAN of the second DTM_CAN in a CAN cluster 0: DMA request is disabled 1: DMA request is enabled
12	DMAEN0	DMA request enable for M_CAN of the first DTM_CAN in a CAN cluster 0: DMA request is disabled 1: DMA request is enabled
11:0	Reserved	Must be kept at reset value.

Memory error control register (CAN_MERRCTL)

Address offset: 0x80

Reset value: 0x8000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RWDIS	Reserved													NCEIE	CEIE
rw														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							ECCEN	Reserved							
							rw								

Bits	Fields	Descriptions
31	RWDIS	CAN_MERRCTL register write access disable When ECRWEN bit in CAN_CTL register is set, this bit is automatically set by hardware. When ECRWEN bit in CAN_CTL register is reset, CAN_MERRCTL register is not writable. 0: CAN_MERRCTL register write access is enabled. 1: CAN_MERRCTL register write access is disabled.
30:18	Reserved	Must be kept at reset value.
17	NCEIE	Noncorrectable error interrupt enable 0: Disable noncorrectable error interrupt 1: Enable noncorrectable error interrupt
16	CEIE	Correctable error interrupt enable 0: Disable correctable error interrupt 1: Enable correctable error interrupt
15:9	Reserved	Must be kept at reset value.
8	ECCEN	Enable ECC

		Enable error report and enable error status update. Set this bit to 0 will clear CAN_CERRRx (x=0...3) registers and CAN_NCERRR register. 0: Disable ECC 1: Enable ECC
7:0	Reserved	Must be kept at reset value.

Correctable error record register x (CAN_CERRRx)(x=0...3)

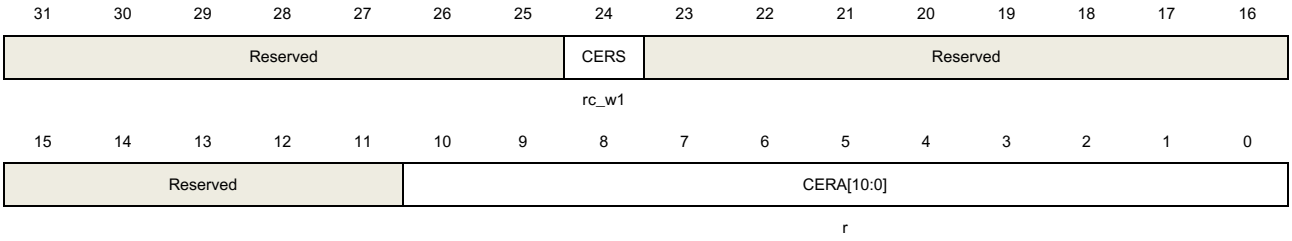
Address offset: 0x84 + 4 * x
Reset value: 0x0000 0000

These registers are used for memory address records when correctable error occurs.

These registers will be cleared by writing 0 to bit ECCEN in CAN_MERRCTL register.

If these registers are all recorded and not cleared when a new correctable error occurs, the new error address will not be recorded, and an overflow flag will be set.

These register have to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:25	Reserved	Must be kept at reset value.
24	CERS	Correctable error record status Clear this bit by writing 1 to it. Only when this bit is 0, CERA[10:0] bits can be rewrite, and the bits value are ignored. 0: No correctable error is recorded in this register 1: A correctable error is already recorded in this register
23:11	Reserved	Must be kept at reset value.
10:0	CERA[10:0]	Correctable error record address

Noncorrectable error record register (CAN_NCERRR)

Address offset: 0x94
Reset value: 0x0000 0000

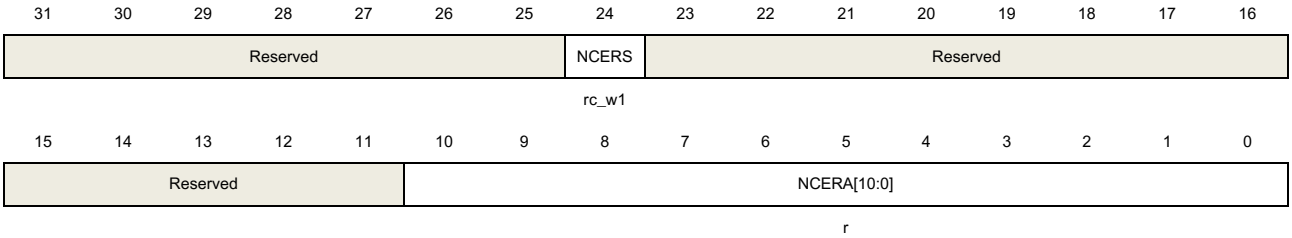
This register is used for memory address records when noncorrectable error occurs.

This register will be cleared by writing 0 to bit ECCEN in CAN_MERRCTL register.

If this register is not cleared when a new noncorrectable error occurs, the new error address

record will overwrite the old one.

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:25	Reserved	Must be kept at reset value.
24	NCERS	Noncorrectable error record status Clear this bit by writing 1 to it. 0: No noncorrectable error is recorded in this register 1: A noncorrectable error is already recorded in this register
23:11	Reserved	Must be kept at reset value.
10:0	NCERA[10:0]	Noncorrectable error record address

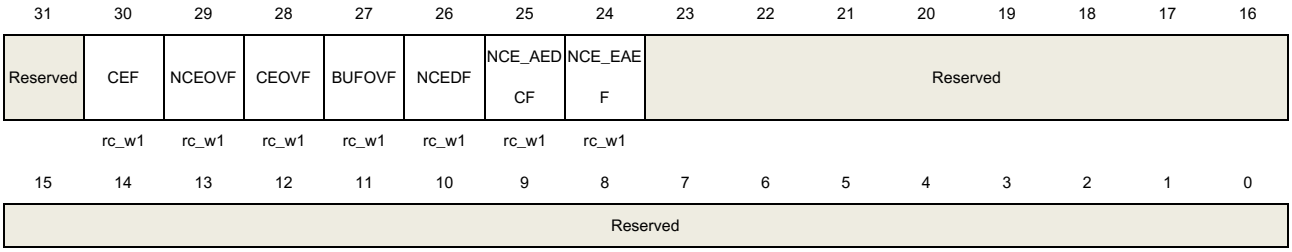
Memory error status register (CAN_MERRSTAT)

Address offset: 0x9C

Reset value: 0x0000 0000

This register is set by hardware automatically.

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30	CEF	Correctable error flag 0: No correctable error occurred 1: Correctable error occurred
29	NCEOVF	Noncorrectable error overflow flag 0: No noncorrectable error overflow occurred 1: Noncorrectable error overflow occurred

28	CEOVF	Correctable error overflow flag 0: No correctable error overflow occurred 1: Correctable error overflow occurred
27	BUFOVF	Correctable error record buffer overflow flag 0: Correctable error record buffer has not overflow 1: Correctable error record buffer overflow occurred
26	NCDEF	Noncorrectable double bit error flag 0: No noncorrectable double bit error occurred 1: Noncorrectable double bit error occurred
25	NCE_AEDCF	Noncorrectable address EDC error flag 0: No noncorrectable address EDC error occurred 1: Noncorrectable address EDC error occurred
24	NCE_EAEF	Noncorrectable EDC after ECC error flag 0: No noncorrectable EDC after ECC error occurred 1: Noncorrectable EDC after ECC error occurred
23:0	Reserved	Must be kept at reset value.

22.6. Modular controller area network (M_CAN)

22.6.1. Overview

Table 22-10. IP user manual versions

IP name	Rvision
BOSCH M_CAN	3.3.1

The M_CAN conforms with CAN protocol version 2.0 part A/B and CAN Flexible data-rate (ISO CAN FD) protocol with up to 64 data bytes. The M_CAN performs communication according to ISO 11898-1:2015. Additional transceiver hardware is required for connection to the physical layer.

The message storage is intended to be a single- or dual-ported Message RAM outside of the module. It is connected to the M_CAN via the Generic Master Interface. Depending on the chosen device, multiple M_CAN controllers can share the same Message RAM.

All functions concerning the handling of messages are implemented by the Rx Handler and the Tx Handler. The Rx Handler manages message acceptance filtering, the transfer of received messages from the CAN Core to the Message RAM as well as providing receive message status information. The Tx Handler is responsible for the transfer of transmit messages from the Message RAM to the CAN Core as well as providing transmit status information.

Acceptance filtering is implemented by a combination of up to 128 filter elements where each one can be configured as a range, as a bit mask, or as a dedicated ID filter.

The M_CAN's clock domain concept allows the separation between the high precision CAN clock and the Host clock, which may be generated by an FM-PLL.

22.6.2. Characteristics

- Conforms with ISO 11898-1: 2015.
- CAN FD with up to 64 data bytes supported.
- CAN Error Logging.
- AUTOSAR support.
- SAE J1939 support.
- Improved acceptance filtering.
- Two configurable Receive FIFOs .
- Separate signalling on reception of High Priority Messages.
- Up to 64 dedicated Receive Buffers.
- Up to 32 dedicated Transmit Buffers.
- Configurable Transmit FIFO.
- Configurable Transmit Queue.
- Configurable Transmit Event FIFO.
- Direct Message RAM access for Host CPU.

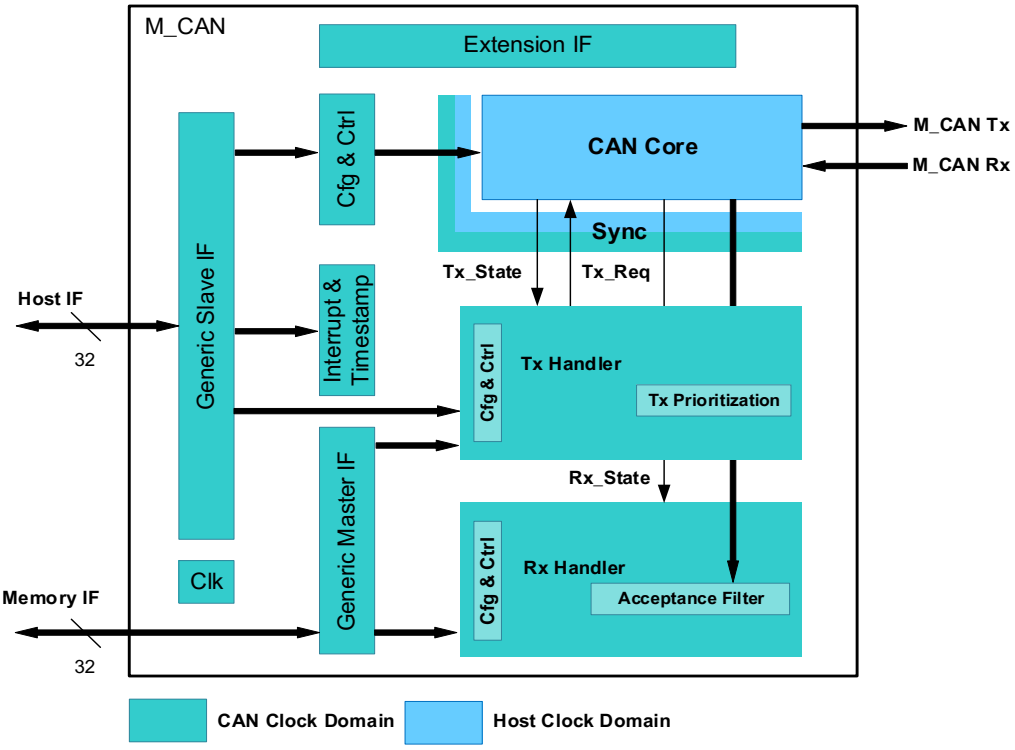
- Multiple M_CANs may share the same Message RAM.
- Programmable loop-back test mode.
- Maskable module interrupts.
- Two clock domains (CAN clock and Host clock).
- Power-down support.
- DMA Support.

22.6.3. Function overview

Block diagram

Figure 22-3. M_CAN block diagram below provides details on the internal configuration of the M_CAN interface.

Figure 22-3. M_CAN block diagram



CAN Core

CAN Protocol Controller and Rx / Tx Shift Register. Handles all ISO 11898-1:2015 protocol functions. Supports 11-bit and 29-bit identifiers.

Sync

Synchronizes signals from the Host clock domain to the CAN clock domain and vice versa.

Clk

Synchronizes reset signal to the Host clock domain and to the CAN clock domain.

Cfg & Ctrl

CAN Core related configuration and control bits.

Interrupt & Timestamp

Interrupt control and 16-bit CAN bit time counter for receive and transmit timestamp generation.

Tx handler

Controls the message transfer from the external Message RAM to the CAN Core. A maximum of 32 Tx Buffers can be configured for transmission. Tx buffers can be used as dedicated Tx Buffers, as Tx FIFO, part of a Tx Queue, or as a combination of them. A Tx Event FIFO stores Tx timestamps together with the corresponding Message ID. Transmit cancellation is also supported.

Rx handler

Controls the transfer of received messages from the CAN Core to the external Message RAM. The Rx Handler supports two Receive FIFOs, each of configurable size, and up to 64 dedicated Rx Buffers for storage of all messages that have passed acceptance filtering. A dedicated Rx Buffer, in contrast to a Receive FIFO, is used to store only messages with a specific identifier. An Rx timestamp is stored together with each message. Up to 128 filters can be defined for 11-bit IDs and up to 64 filters for 29-bit IDs.

Generic slave interface

Connects the M_CAN to a customer specific Host CPU. The Generic Slave Interface is capable to connect to an 32-bit bus to support a wide range of interconnection structures.

Generic master interface

Connects the M_CAN access to an external 32-bit Message RAM. The maximum Message RAM size is 16K * 32 bit. A single M_CAN can use at most 4.25K * 32 bit.

Note: For each CAN cluster, 1.5K*40bit(32bit Message DATA, 8bit ECC Code) is released for message RAM.

Extension interface

All flags from the Interrupt Register CAN_M_CAN_IR as well as selected internal status and control signals are routed to this interface. The interface is intended for connection of the M_CAN to a module-external interrupt unit or to other module-external components. The connection of these signals is optional.

Dual clock sources

To improve the EMC behavior, a spread spectrum clock can be used for the Host clock domain. Due to the high precision clocking requirements of the CAN core, a separate clock without any modulation has to be provided as CAN clock.

Within the M_CAN module there is a synchronization mechanism implemented to ensure save data transfer between the two clock domains.

Note: In order to achieve a stable function of the M_CAN, the Host clock must always be faster than or equal to the CAN clock. Also the modulation depth of the spread spectrum clock has to be regarded.

Dual interrupt lines

The module provides two interrupt lines. Interrupts can be routed either to M_CAN interrupt 0 or M_CAN interrupt 1. By default all interrupts are routed to interrupt line M_CAN interrupt 0. By programming EINT0 and EINT1 bits of the Interrupt Line Enable (CAN_M_CAN_ILE) register, the interrupt lines can be enabled or disabled separately.

Operating modes

Software initialization

Software initialization is started by setting bit INIT in CAN_M_CAN_CCCR register, either by software or by a hardware reset, when an uncorrected bit error was detected in the Message RAM, or by going Bus_Off. While the bit INIT in CAN_M_CAN_CCCR register is set, message transfer from and to the CAN bus is stopped, the status of the CAN bus transmit output is recessive (HIGH). The counters of the Error Management Logic EML are unchanged. Setting bit INIT in CAN_M_CAN_CCCR register does not change any configuration register. Resetting bit INIT in CAN_M_CAN_CCCR register finishes the software initialization. Afterwards the Bit Stream Processor BSP synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (= Bus_Idle) before it can take part in bus activities and start the message transfer.

Access to the M_CAN configuration registers is only enabled when both bits CAN_M_CAN_CCCR.INIT and CAN_M_CAN_CCCR.CCE are set (protected write).

The bit CCE in CAN_M_CAN_CCCR register can only be set/reset while CAN_M_CAN_CCCR.INIT = '1'. The bit CCE in CAN_M_CAN_CCCR register is automatically reset when The bit INIT in CAN_M_CAN_CCCR register is reset.

The following registers are reset when bit CCE in CAN_M_CAN_CCCR register is set

- CAN_M_CAN_HPMS - High Priority Message Status.
- CAN_M_CAN_RXF0S - Rx FIFO 0 Status.
- CAN_M_CAN_RXF1S - Rx FIFO 1 Status.
- CAN_M_CAN_TXFQS - Tx FIFO/Queue Status.
- CAN_M_CAN_TXBRP - Tx Buffer Request Pending.
- CAN_M_CAN_TXBTO - Tx Buffer Transmission Occurred.
- CAN_M_CAN_TXBCF - Tx Buffer Cancellation Finished.
- CAN_M_CAN_TXEFS - Tx Event FIFO Status.

The Timeout Counter value (CAN_M_CAN_TOCV.TOC) is preset to the value configured by

TOP in CAN_M_CAN_TOCC register when CCE in CAN_M_CAN_CCCR register is set.

In addition the state machines of the Tx Handler and Rx Handler are held in idle state while CAN_M_CAN_CCCR.CCE = '1'.

The following registers are only writeable while CAN_M_CAN_CCCR.CCE = '0'

- CAN_M_CAN_TXBAR - Tx Buffer Add Request.
- CAN_M_CAN_TXBCR - Tx Buffer Cancellation Request.

The TEST and MON in CAN_M_CAN_CCCR register can only be set by the Host while CAN_M_CAN_CCCR.INIT = '1' and CAN_M_CAN_CCCR.CCE = '1'. Both bits may be reset at any time. The DAR in CAN_M_CAN_CCCR register can only be set/reset while CAN_M_CAN_CCCR.INIT = '1' and CAN_M_CAN_CCCR.CCE = '1'.

Note: In case the Message RAM is equipped with parity or ECC functionality, it is recommended to initialize the Message RAM after hardware reset by writing e.g. 0x00000000 to each Message RAM word to create valid parity/ECC checksums.

Normal operation

Once the M_CAN is initialized and bit INIT in CAN_M_CAN_CCCR register is reset to zero, the M_CAN synchronizes itself to the CAN bus and is ready for communication.

After passing the acceptance filtering, received messages including Message ID and DLC are stored into a dedicated Rx Buffer or into Rx FIFO 0 or Rx FIFO 1.

For messages to be transmitted dedicated Tx Buffers and/or a Tx FIFO or a Tx Queue can be initialized or updated. Automated transmission on reception of remote frames is not implemented.

CAN FD operation

There are two variants in the CAN FD frame transmission, first the CAN FD frame without bit rate switching. The second variant is the CAN FD frame where control field, data field, and CRC field are transmitted with a higher bit rate than the beginning and the end of the frame.

The previously reserved bit in CAN frames with 11-bit identifiers and the first previously reserved bit in CAN frames with 29-bit identifiers will now be decoded as FDF bit. FDF = recessive signifies a CAN FD frame, FDF = dominant signifies a Classic CAN frame. In a CAN FD frame, the two bits following FDF, res and BRS, decide whether the bit rate inside of this CAN FD frame is switched. A CAN FD bit rate switch is signified by res = dominant and BRS = recessive. The coding of res = recessive is reserved for future expansion of the protocol. In case the M_CAN receives a frame with FDF = recessive and res = recessive, it will signal a Protocol Exception Event by setting bit PXE in CAN_M_CAN_PSR register. When Protocol Exception Handling is enabled (CAN_M_CAN_CCCR.PXHD = '0'), this causes the operation state to change from Receiver (CAN_M_CAN_PSR.ACT = "10") to Integrating (CAN_M_CAN_PSR.ACT = "00") at the next sample point. In case Protocol Exception Handling is disabled (CAN_M_CAN_CCCR.PXHD = '1'), the M_CAN will treat a recessive res bit as an form error and will respond with an error frame.

CAN FD operation is enabled by programming FDOE in CAN_M_CAN_CCCR register. In case CAN_M_CAN_CCCR.FDOE = '1', transmission and reception of CAN FD frames is enabled. Transmission and reception of Classic CAN frames is always possible. Whether a CAN FD frame or a Classic CAN frame is transmitted can be configured via bit FDF in the respective Tx Buffer element. With CAN_M_CAN_CCCR.FDOE = '0', received frames are interpreted as Classic CAN frames, which leads to the transmission of an error frame when receiving a CAN FD frame. When CAN FD operation is disabled, no CAN FD frames are transmitted even if bit FDF of a Tx Buffer element is set. CAN_M_CAN_CCCR.FDOE and CAN_M_CAN_CCCR.BRSE can only be changed while CAN_M_CAN_CCCR.INIT and CAN_M_CAN_CCCR.CCE are both set.

With CAN_M_CAN_CCCR.FDOE = '0', the setting of bits FDF and BRS is ignored and frames are transmitted in Classic CAN format. With CAN_M_CAN_CCCR.FDOE = '1' and CAN_M_CAN_CCCR.BRSE = '0', only bit FDF of a Tx Buffer element is evaluated. With CAN_M_CAN_CCCR.FDOE = '1' and CAN_M_CAN_CCCR.BRSE = '1', transmission of CAN FD frames with bit rate switching is enabled. All Tx Buffer elements with bits FDF and BRS set are transmitted in CAN FD format with bit rate switching.

A mode change during CAN operation is only recommended under the following conditions:

- The failure rate in the CAN FD data phase is significantly higher than in the CAN FD arbitration phase. In this case disable the CAN FD bit rate switching option for transmissions.
- During system startup all nodes are transmitting classic CAN messages until it is verified that they are able to communicate in CAN FD format. If this is true, all nodes switch to CAN FD operation.
- Wake-up messages in CAN partial networking have to be transmitted in classic CAN format.
- End-of-line programming in case not all nodes are CAN FD capable. Non CAN FD nodes are held in silent mode until programming has completed. Then all nodes switch back to classic CAN communication.

In the CAN FD format, the coding of the DLC differs from the standard CAN format. The DLC codes 0 to 8 have the same coding as in standard CAN, the codes 9 to 15, which in standard CAN all code a data field of 8 bytes, are coded according to ***Table 22-11. Coding of DLC in CAN FD***.

Table 22-11. Coding of DLC in CAN FD

DLC	9	10	11	12	13	14	15
Number of Data Bytes	12	16	20	24	32	48	64

In CAN FD frames, the bit timing will be switched inside the frame, after the BRS (Bit Rate Switch) bit, if this bit is recessive. Before the BRS bit, in the CAN FD arbitration phase, the nominal CAN bit timing is used as defined by the Nominal Bit Timing & Prescaler Register CAN_M_CAN_NBTP. In the following CAN FD data phase, the data phase bit timing is used as defined by the Data Bit Timing & Prescaler Register CAN_M_CAN_DBTP. The bit timing is switched back from the data phase timing at the CRC delimiter or when an error is detected,

whichever occurs first.

The maximum configurable bit rate in the CAN FD data phase depends on the CAN clock frequency. Example: with a CAN clock frequency of 20MHz and the shortest configurable bit time of 4 tq, the bit rate in the data phase is 5 Mbit/s.

In both data frame formats, CAN FD and CAN FD with bit rate switching, the value of the bit ESI (Error Status Indicator) is determined by the transmitter's error state at the start of the transmission. If the transmitter is error passive, ESI is transmitted recessive, else it is transmitted dominant.

Transmitter delay compensation

During the data phase of a CAN FD transmission only one node is transmitting, all others are receivers. The length of the bus line has no impact. When transmitting via CAN M_CAN's transmit output pin the M_CAN receives the transmitted data from its local CAN transceiver via CAN receive input pin. The received data is delayed by the transmitter delay. In case this delay is greater than TSEG1 (time segment before sample point), a bit error is detected. In order to enable a data phase bit time that is even shorter than the transmitter delay, the delay compensation is introduced. Without transmitter delay compensation, the bit rate in the data phase of a CAN FD frame is limited by the transmitter delay.

Description

The M_CAN's protocol unit has implemented a delay compensation mechanism to compensate the transmitter delay, thereby enabling transmission with higher bit rates during the CAN FD data phase independent of the delay of a specific CAN transceiver.

To check for bit errors during the data phase of transmitting nodes, the delayed transmit data is compared against the received data at the Secondary Sample Point SSP. If a bit error is detected, the transmitter will react on this bit error at the next following regular sample point. During arbitration phase the delay compensation is always disabled.

The transmitter delay compensation enables configurations where the data bit time is shorter than the transmitter delay, it is described in detail in ISO 11898-1:2015. It is enabled by setting bit TDC in CAN_M_CAN_DBTP register.

The received bit is compared against the transmitted bit at the SSP. The SSP position is defined as the sum of the measured delay from the M_CAN's transmit output through the transceiver to the receive input plus the transmitter delay compensation offset as configured by TDCO in CAN_M_CAN_TDCR register. The transmitter delay compensation offset is used to adjust the position of the SSP inside the received bit (e.g. half of the bit time in the data phase). The position of the secondary sample point is rounded down to the next integer number of mtq.

The TDCV in CAN_M_CAN_PSR register shows the actual transmitter delay compensation value. The TDCV in CAN_M_CAN_PSR register is cleared when INIT in CAN_M_CAN_CCCR register is set and is updated at each transmission of an FD frame while TDC in CAN_M_CAN_DBTP register is set.

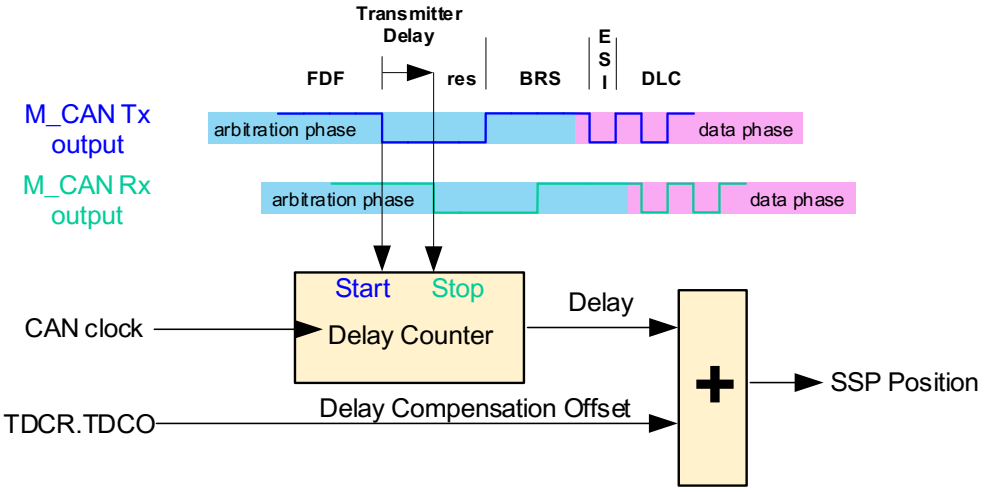
The following boundary conditions have to be considered for the transmitter delay compensation implemented in the M_CAN:

- The sum of the measured delay from CAN Tx output to CAN Rx input and the configured transmitter delay compensation offset TDCO in CAN_M_CAN_TDCR register has to be less than 6 bit times in the data phase.
- The sum of the measured delay from CAN Tx output to CAN Rx input and the configured transmitter delay compensation offset TDCO in CAN_M_CAN_TDCR register has to be less or equal 127 mtq. In case this sum exceeds 127 mtq, the maximum value of 127 mtq is used for transmitter delay compensation.
- The data phase ends at the sample point of the CRC delimiter, that stops checking of receive bits at the SSPs.

Transmitter delay compensation measurement

If transmitter delay compensation is enabled by programming TDC in CAN_M_CAN_DBTP register = '1', the measurement is started within each transmitted CAN FD frame at the falling edge of bit FDF to bit res. The measurement is stopped when this edge is seen at the receive input CAN Rx input of the transmitter. The resolution of this measurement is one mtq.

Figure 22-4. Transmitter delay measurement



To avoid that a dominant glitch inside the received FDF bit ends the delay compensation measurement before the falling edge of the received res bit, resulting in a to early SSP position, the use of a transmitter delay compensation filter window can be enabled by programming TDCF in CAN_M_CAN_TDCR register. This defines a minimum value for the SSP position. Dominant edges on CAN Rx input, that would result in an earlier SSP position are ignored for transmitter delay measurement. The measurement is stopped when the SSP position is at least TDCF in CAN_M_CAN_TDCR register AND CAN Rx input is low.

Note: The M_CAN always performs Transmitter Delay measurement, it does not support the configuration of a static (fixed) Transmitter Delay.

Restricted operation mode

In Restricted Operation Mode the node is able to receive data and remote frames and to give acknowledge to valid frames, but it does not send data frames, remote frames, active error frames, or overload frames. In case of an error condition or overload condition, it does not send dominant bits, instead it waits for the occurrence of bus idle condition to resynchronize itself to the CAN communication. The error counters (REC and TEC in CAN_M_CAN_ECR register) are frozen while Error Logging (CEL in CAN_M_CAN_ECR register) is active. The Host can set the M_CAN into Restricted Operation mode by setting bit ASM in CAN_M_CAN_CCCR register. The bit can only be set by the Host when both CCE and INIT in CAN_M_CAN_CCCR register are set to '1'. The bit can be reset by the Host at any time.

Restricted Operation Mode is automatically entered when the Tx Handler was not able to read data from the Message RAM in time. To leave Restricted Operation Mode, the Host CPU has to reset ASM in CAN_M_CAN_CCCR register.

The Restricted Operation Mode can be used in applications that adapt themselves to different CAN bit rates. In this case the application tests different bit rates and leaves the Restricted Operation Mode after it has received a valid frame.

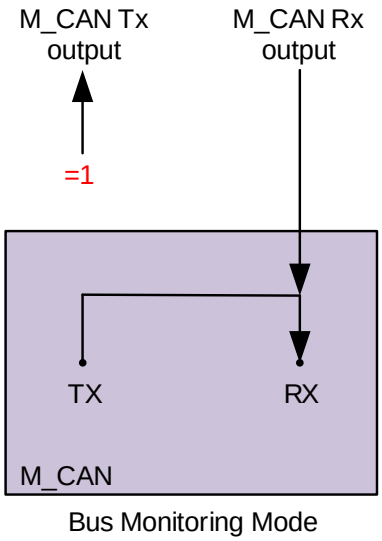
Note: The Restricted Operation Mode must not be combined with the Loop Back Mode (internal or external).

Bus monitoring mode

The M_CAN is set in Bus Monitoring Mode by programming MON in CAN_M_CAN_CCCR register to one. In Bus Monitoring Mode (see ISO 11898-1:2015, 10.14 Bus monitoring), the M_CAN is able to receive valid data frames and valid remote frames, but cannot start a transmission. In this mode, it sends only recessive bits on the CAN bus. If the M_CAN is required to send a dominant bit (ACK bit, overload flag, active error flag), the bit is rerouted internally so that the M_CAN monitors this dominant bit, although the CAN bus may remain in recessive state. In Bus Monitoring Mode register CAN_M_CAN_TXBRP is held in reset state.

The Bus Monitoring Mode can be used to analyze the traffic on a CAN bus without affecting it by the transmission of dominant bits. ***Figure 22-5. Pin control in bus monitoring mode*** shows the connection of M_CAN Tx and M_CAN Rx signals to the M_CAN in Bus Monitoring Mode.

Figure 22-5. Pin control in bus monitoring mode



Bus Monitoring Mode

Disabled automatic retransmission

According to the CAN Specification (see ISO 11898-1:2015, 8.3.4 Recovery Management), the M_CAN provides means for automatic retransmission of frames that have lost arbitration or that have been disturbed by errors during transmission. By default automatic retransmission is enabled.

Frame transmission in DAR mode

In DAR mode all transmissions are automatically cancelled after they started on the CAN bus. A Tx Buffer's Tx Request Pending bit (TRPx in CAN_M_CAN_TXBRP register) is reset after successful transmission, when a transmission has not yet been started at the point of cancellation, has been aborted due to lost arbitration, or when an error occurred during frame transmission.

- Successful transmission:
  Corresponding Tx Buffer Transmission Occurred bit TOx in CAN_M_CAN_TXBTO register set.
  Corresponding Tx Buffer Cancellation Finished bit CFx in CAN_M_CAN_TXBCF register not set.
- Successful transmission in spite of cancellation:
  Corresponding Tx Buffer Transmission Occurred bit TOx in CAN_M_CAN_TXBTO register set.
  Corresponding Tx Buffer Cancellation Finished bit CFx in CAN_M_CAN_TXBCF register set.
- Arbitration lost or frame transmission disturbed:
  Corresponding Tx Buffer Transmission Occurred bit TOx in CAN_M_CAN_TXBTO register not set.
  Corresponding Tx Buffer Cancellation Finished bit CFx in CAN_M_CAN_TXBCF register set.

In case of a successful frame transmission, and if storage of Tx events is enabled, a Tx Event

FIFO element is written with Event Type ET = "10" (transmission in spite of cancellation).

Power down (sleep mode)

The M_CAN can be set into power down mode controlled by input signal clock stop request or via CC Control Register CAN_M_CAN_CCCR.CSR. As long as the clock stop request signal is active (CAN_CSR.CSRx), bit CSR in CAN_M_CAN_CCCR register is read as one.

When all pending transmission requests have completed, the M_CAN waits until bus idle state is detected. Then the M_CAN sets then INIT bit in CAN_M_CAN_CCCR register to one to prevent any further CAN transfers. Now the M_CAN acknowledges that it is ready for power down by setting output signal clock stop acknowledge to one (connect to CAN_CSA.CSAx) and CSA bit in CAN_M_CAN_CCCR register to one. Before the module clocks Host clock and CAN clock are switched off, further register accesses can be made except for INIT bit in CAN_M_CAN_CCCR register which is held at one.

Note: In case of a heavily disturbed CAN bus, it may happen that idle state is never reached and INIT bit in CAN_M_CAN_CCCR register is therefore not set by the M_CAN. This situation can be detected by polling ACT in CAN_M_CAN_PSR register. In case the M_CAN does not enter idle state, the software can write CAN_M_CAN_CCCR.INIT = '1' which stops CAN communication of the M_CAN immediately, regardless whether there is a transmission/reception ongoing or not.

To leave power down mode, the application has to turn on the module clocks before resetting signal clock stop request resp. CC Control Register flag CAN_M_CAN_CCCR.CSR. The M_CAN will acknowledge this by resetting clock stop acknowledge output signal and resetting CSA bit in CAN_M_CAN_CCCR. Afterwards, the application can restart CAN communication by resetting bit INIT CAN_M_CAN_CCCR register.

Test modes

To enable write access to register CAN_M_CAN_TEST, bit TEST in CAN_M_CAN_CCCR register has to be set to one. This allows the configuration of the test modes and test functions.

Four output functions are available for the CAN transmit pin by programming TX in CAN_M_CAN_TEST register. Additionally to its default function - the serial data output - it can drive the CAN Sample Point signal to monitor the M_CAN's bit timing and it can drive constant dominant or recessive values. The actual value at M_CAN Rx pin can be read from RX in CAN_M_CAN_TEST register. Both functions can be used to check the CAN bus' physical layer.

Due to the synchronization mechanism between CAN clock and Host clock domain, there may be a delay of several Host clock periods between writing to TX in CAN_M_CAN_TEST register until the new configuration is visible at output Tx pin. This applies also when reading input Rx pin via RX in CAN_M_CAN_TEST register.

Note: Test modes should be used for production tests or self test only. The software control for output Tx pin interferes with all CAN protocol functions. It is not recommended to use test modes for application.

External loop back mode

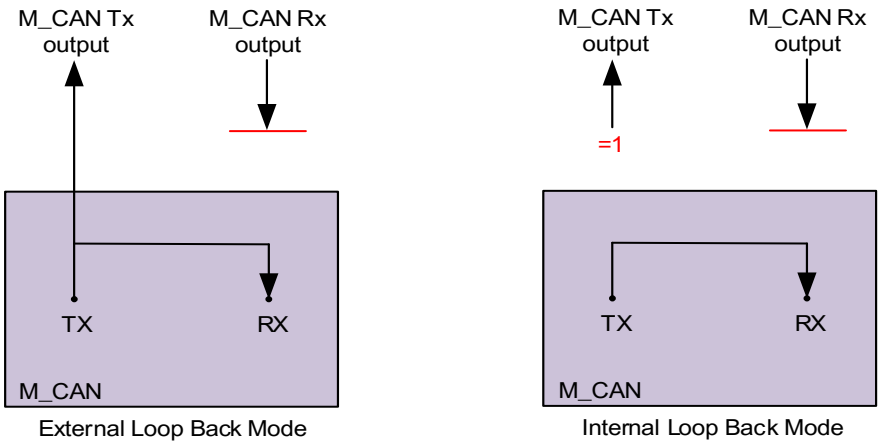
The M_CAN can be set in External Loop Back Mode by programming bit LBCK in CAN_M_CAN_TEST register to one. In Loop Back Mode, the M_CAN treats its own transmitted messages as received messages and stores them (if they pass acceptance filtering) into an Rx Buffer or an Rx FIFO. ***Figure 22-6. Pin control in loop back modes*** shows the connection of Tx and Rx signals to the M_CAN in External Loop Back Mode.

This mode is provided for hardware self-test. To be independent from external stimulation, the M_CAN ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/remote frame) in Loop Back Mode. In this mode the M_CAN performs an internal feedback from its Tx output to its Rx input. The actual value of the CAN Rx input pin is disregarded by the M_CAN. The transmitted messages can be monitored at the CAN Tx output pin.

Internal Loop Back Mode

Internal Loop Back Mode is entered by programming bits CAN_M_CAN_TEST.LBCK and CAN_M_CAN_CCCR.MON to one. This mode can be used for a "Hot Selftest", meaning the M_CAN can be tested without affecting a running CAN system connected to the CAN Tx output and Rx input pins. In this mode M_CAN Rx pin is disconnected from the M_CAN and M_CAN Tx pin is held recessive. ***Figure 22-6. Pin control in loop back modes*** shows the connection of M_CAN Tx pin and Rx to the M_CAN in case of Internal Loop Back Mode.

Figure 22-6. Pin control in loop back modes



Timestamp generation

Internal timestamp generation

For internal timestamp generation the M_CAN supplies a 16-bit wrap-around counter. A prescaler CAN_M_CAN_TSCC.TCP can be configured to clock the counter in multiples of CAN bit times (1…16). The counter is readable via CAN_M_CAN_TSCV.TSC. A write access to register CAN_M_CAN_TSCV resets the counter to zero. When the timestamp counter wraps around interrupt flag CAN_M_CAN_IR.TSW is set.

On start of frame reception / transmission (SOF) the counter value is captured and stored into the timestamp section of an Rx Buffer / Rx FIFO (RXTS[15:0]) or Tx Event FIFO (TXTS[15:0]) element.

Note that the timestamp is not supported for CAN FD frames.

Timeout counter

To signal timeout conditions for Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO the M_CAN supplies a 16-bit Timeout Counter. It operates as down-counter and uses the same prescaler controlled by TCP in CAN_M_CAN_TSCC register as the Timestamp Counter. The Timeout Counter is configured via register CAN_M_CAN_TOCC. The actual counter value can be read from TOC in CAN_M_CAN_TOCV register. The Timeout Counter can only be started while CAN_M_CAN_CCCR.INIT = '0'. It is stopped when CAN_M_CAN_CCCR.INIT = '1', e.g. when the M_CAN enters Bus_Off state.

The operation mode is selected by TOS in CAN_M_CAN_TOCC register. When operating in Continuous Mode, the counter starts when bit INIT in CAN_M_CAN_CCCR register is reset. A write to CAN_M_CAN_TOCV presets the counter to the value configured by TOP in CAN_M_CAN_TOCC register and continues down-counting.

When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by TOP in CAN_M_CAN_TOCC register. Down-counting is started when the first FIFO element is stored. Writing to CAN_M_CAN_TOCV has no effect.

When the counter reaches zero, interrupt flag TOO in CAN_M_CAN_IR register is set. In Continuous Mode, the counter is immediately restarted at TOP in CAN_M_CAN_TOCC register.

Note: The clock signal for the Timeout Counter is derived from the CAN Core's sample point signal. Therefore the point in time where the Timeout Counter is decremented may vary due to the synchronization / re-synchronization mechanism of the CAN Core. If the bit rate switch feature in CAN FD is used, the timeout counter is clocked differently in arbitration and data field.

Rx handling

The Rx Handler controls the acceptance filtering, the transfer of received messages to the Rx Buffers or to one of the two Rx FIFOs, as well as the Rx FIFO's Put and Get Indices.

Acceptance filtering

The M_CAN offers the possibility to configure two sets of acceptance filters, one for standard identifiers and one for extended identifiers. These filters can be assigned to an Rx Buffer or to Rx FIFO 0,1. For acceptance filtering each list of filters is executed from element #0 until the first matching element. Acceptance filtering stops at the first matching element. The following filter elements are not evaluated for this message.

The main features are:

- Each filter element can be configured as range filter (from - to), filter for one or two dedicated IDs or classic bit mask filter.
- Each filter element is configurable for acceptance or rejection filtering.
- Each filter element can be enabled / disabled individually.
- Filters are checked sequentially, execution stops with the first matching filter element.

Related configuration registers are:

- Global Filter Configuration CAN_M_CAN_GFC.
- Standard ID Filter Configuration CAN_M_CAN_SIDFC.
- Extended ID Filter Configuration CAN_M_CAN_XIDFC.
- Extended ID AND Mask CAN_M_CAN_XIDAM.

Depending on the configuration of the filter element (SFEC/EFEC) a match triggers one of the following actions:

- Store received frame in FIFO 0 or FIFO 1.
- Store received frame in Rx Buffer.
- Store received frame in Rx Buffer and generate pulse at filter event pin.
- Reject received frame.
- Set High Priority Message interrupt flag HPM in CAN_M_CAN_IR register.
- Set High Priority Message interrupt flag HPM in CAN_M_CAN_IR register and store received frame in FIFO 0 or FIFO 1.

Acceptance filtering is started after the complete identifier has been received. After acceptance filtering has completed, and if a matching Rx Buffer or Rx FIFO has been found, the Message Handler starts writing the received message data in portions of 32 bit to the matching Rx Buffer or Rx FIFO. If the CAN protocol controller has detected an error condition (e.g. CRC error), this message is discarded with the following impact on the affected Rx Buffer or Rx FIFO:

Rx Buffer: New Data flag of matching Rx Buffer is not set, but Rx Buffer (partly) overwritten with received data. For error type see LEC in CAN_M_CAN_PSR register respectively DLEC in CAN_M_CAN_PSR register.

Rx FIFO: Put index of matching Rx FIFO is not updated, but related Rx FIFO element (partly) overwritten with received data. For error type see LEC in CAN_M_CAN_PSR register respectively DLEC in CAN_M_CAN_PSR register. In case the matching Rx FIFO is operated in overwrite mode, the boundary conditions described in Section "Rx FIFO Overwrite Mode" have to be considered

Note: When an accepted message is written to one of the two Rx FIFOs, or into an Rx Buffer, the unmodified received identifier is stored independent of the filter(s) used. The result of the acceptance filter process is strongly depending on the sequence of configured filter elements.

Range filter

The filter matches for all received frames with Message IDs in the range defined by SFID1/SFID2 resp. EFID1/EFID2.

There are two possibilities when range filtering is used together with extended frames:

EFT = "00": The Message ID of received frames is ANDed with the Extended ID AND Mask (CAN_M_CAN_XIDAM) before the range filter is applied.

EFT = "11": The Extended ID AND Mask (CAN_M_CAN_XIDAM) is not used for range filtering.

Filter for specific IDs

A filter element can be configured to filter for one or two specific Message IDs. To filter for one specific Message ID, the filter element has to be configured with SFID1 = SFID2 resp. EFID1 = EFID2.

Classic bit mask filter

Classic bit mask filtering is intended to filter groups of Message IDs by masking single bits of a received Message ID. With classic bit mask filtering SFID1/EFID1 is used as Message ID filter, while SFID2/EFID2 is used as filter mask.

A zero bit at the filter mask will mask out the corresponding bit position of the configured ID filter, e.g. the value of the received Message ID at that bit position is not relevant for acceptance filtering. Only those bits of the received Message ID where the corresponding mask bits are one are relevant for acceptance filtering.

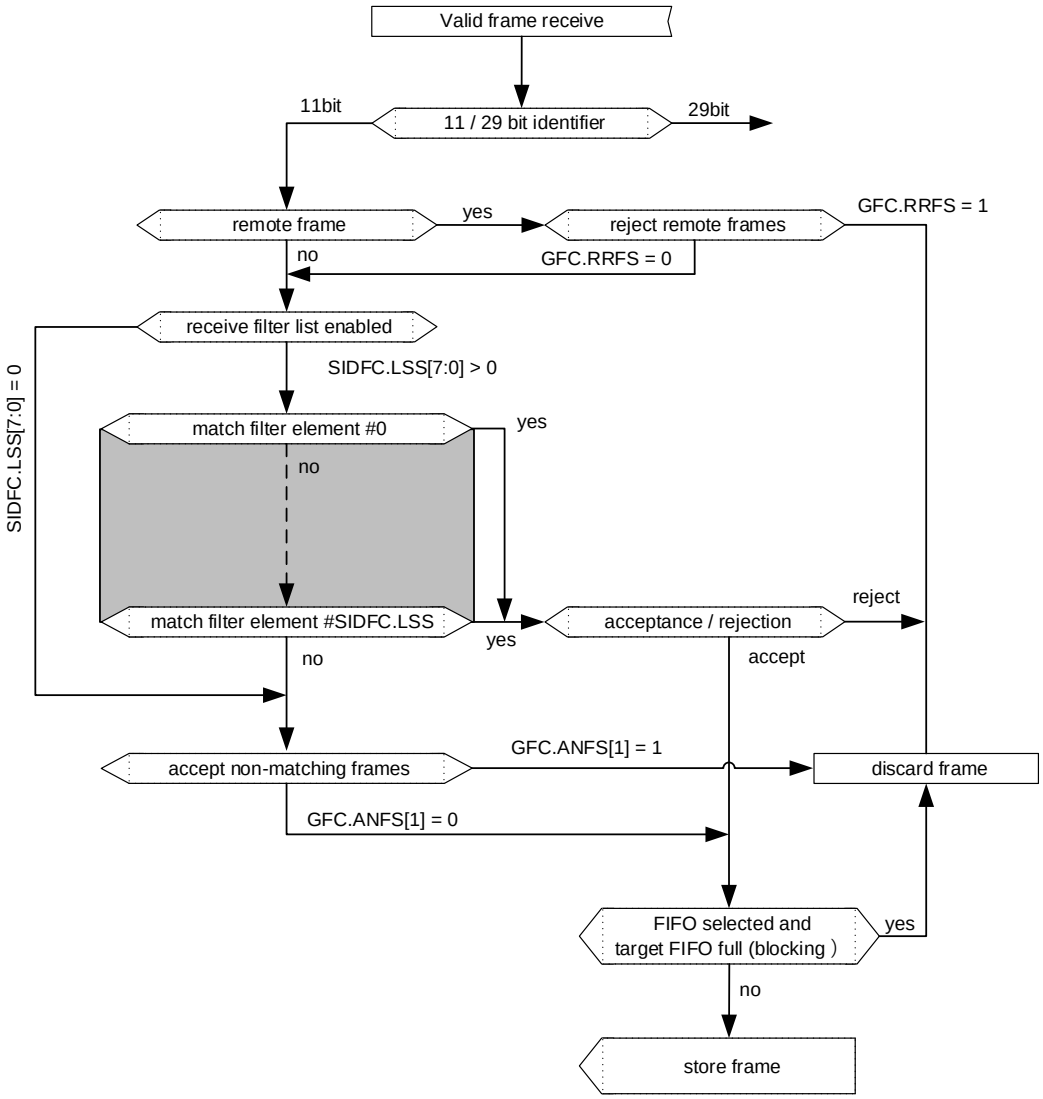
In case all mask bits are one, a match occurs only when the received Message ID and the Message ID filter are identical. If all mask bits are zero, all Message IDs match.

Standard message ID filtering

Figure 22-7. Standard message ID filter path below shows the flow for standard Message ID (11-bit Identifier) filtering. The Standard Message ID Filter element is described in Section "Standard message ID filter element".

Controlled by the Global Filter Configuration CAN_M_CAN_GFC and the Standard ID Filter Configuration CAN_M_CAN_SIDFC Message ID, Remote Transmission Request bit (RTR), and the Identifier Extension bit (IDE) of received frames are compared against the list of configured filter elements.

Figure 22-7. Standard message ID filter path



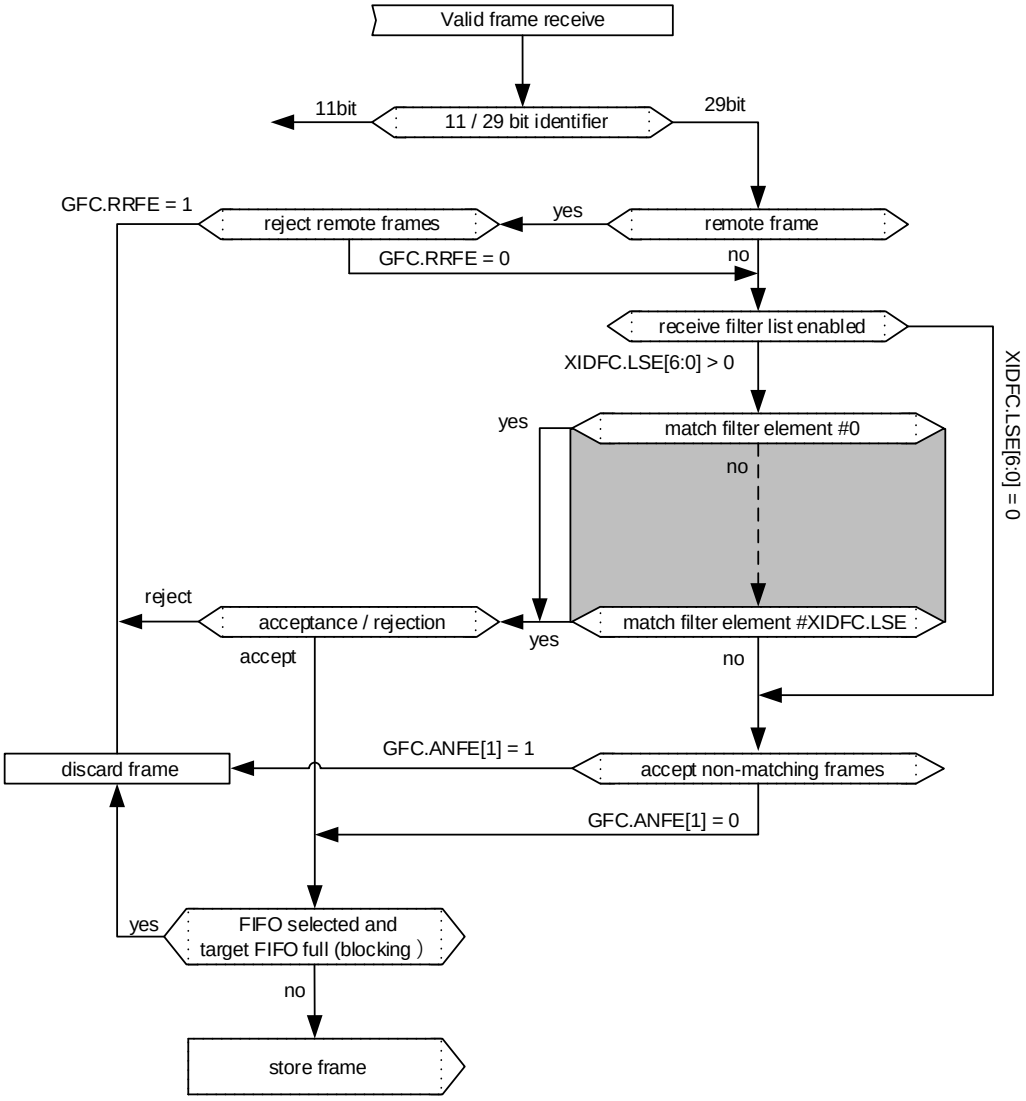
Extended message ID filtering

Figure 22-8. Extended message ID filter path below shows the flow for extended Message ID (29-bit Identifier) filtering. The Extended Message ID Filter element is described in Section "Extended message ID filter element".

Controlled by the Global Filter Configuration CAN_M_CAN_GFC and the Extended ID Filter Configuration CAN_M_CAN_XIDFC Message ID, Remote Transmission Request bit (RTR), and the Identifier Extension bit (IDE) of received frames are compared against the list of configured filter elements.

The Extended ID AND Mask CAN_M_CAN_XIDAM is ANDed with the received identifier before the filter list is executed.

Figure 22-8. Extended message ID filter path



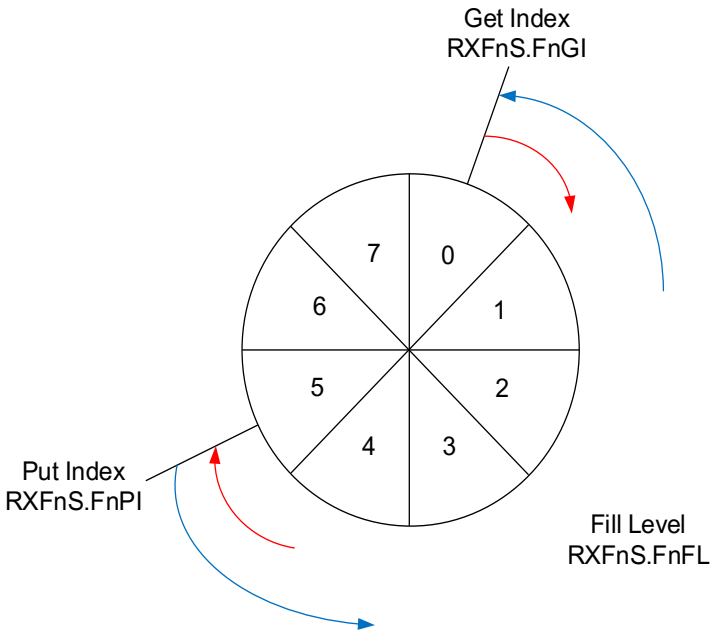
Rx FIFOs

Rx FIFO 0 and Rx FIFO 1 can be configured to hold up to 64 elements each. Configuration of the two Rx FIFOs is done via registers CAN_M_CAN_RXF0C and CAN_M_CAN_RXF1C.

Received messages that passed acceptance filtering are transferred to the Rx FIFO as configured by the matching filter element. For a description of the filter mechanisms available for Rx FIFO 0 and Rx FIFO 1 see section ***Acceptance filtering***. The Rx FIFO element is described in section "Rx buffer and FIFO element".

To avoid an Rx FIFO overflow, the Rx FIFO watermark can be used. When the Rx FIFO fill level reaches the Rx FIFO watermark configured by FnWM in RXFnC register, interrupt flag RFnW in CAN_M_CAN_IR register is set. When the Rx FIFO Put Index reaches the Rx FIFO Get Index an Rx FIFO Full condition is signalled by FnF in RXFnS register. In addition interrupt flag RFnF in CAN_M_CAN_IR register is set.

Figure 22-9. Rx FIFO status



When reading from an Rx FIFO, Rx FIFO Get Index RXFnS.FnGI * FIFO Element Size has to be added to the corresponding Rx FIFO start address (RXFnC.FnSA).

Table 22-12. Rx Buffer / FIFO element size

CAN_M_CAN_RXESC .RBDS[2:0] CAN_M_CAN_RXESC .FnDS[2:0]	Data Field [bytes]	FIFO Element Size [RAM words]
000	8	4
001	12	5
010	16	6
011	20	7
100	24	8
101	32	10
110	48	14
111	64	18

Rx FIFO blocking mode

The Rx FIFO blocking mode is configured by RXFnC.FnOM = '0'. This is the default operation mode for the Rx FIFOs.

When an Rx FIFO full condition is reached (RXFnS.FnPI = RXFnS.FnGI), no further messages are written to the corresponding Rx FIFO until at least one message has been read out and the Rx FIFO Get Index has been incremented. An Rx FIFO full condition is signalled by RXFnS.FnF = '1'. In addition interrupt flag RFnF in CAN_M_CAN_IR register is

set.

In case a message is received while the corresponding Rx FIFO is full, this message is discarded and the message lost condition is signalled by RXFnS.RFnL = '1'. In addition interrupt flag RFnL in CAN_M_CAN_IR register is set.

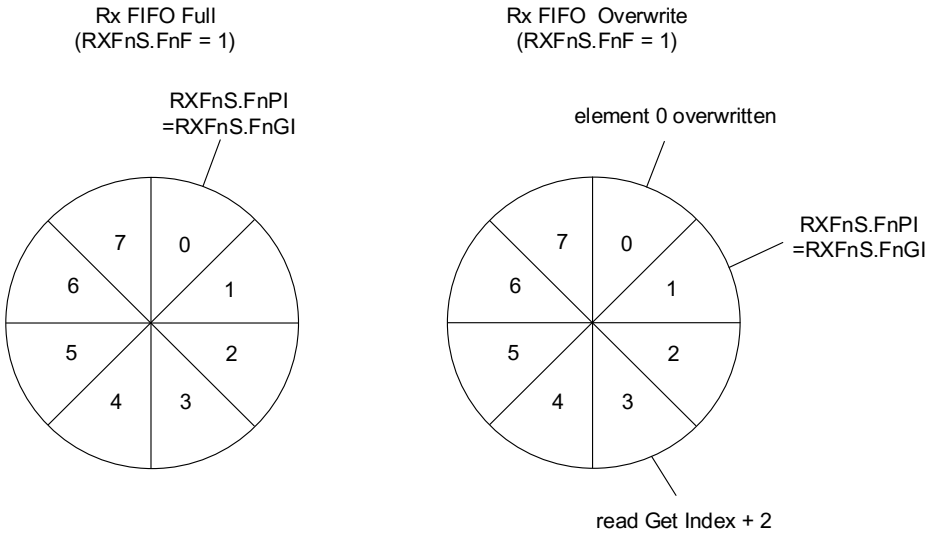
Rx FIFO overwrite mode

The Rx FIFO overwrite mode is configured by RXFnC.FnOM = '1'.

When an Rx FIFO full condition (RXFnS.FnPI = RXFnS.FnGI) is signalled by RXFnS.FnF = '1', the next message accepted for the FIFO will overwrite the oldest FIFO message. Put and get index are both incremented by one.

When an Rx FIFO is operated in overwrite mode and an Rx FIFO full condition is signalled, reading of the Rx FIFO elements should start at least at get index + 1. The reason for that is, that it might happen, that a received message is written to the Message RAM (put index) while the CPU is reading from the Message RAM (get index). In this case inconsistent data may be read from the respective Rx FIFO element. Adding an offset to the get index when reading from the Rx FIFO avoids this problem. The offset depends on how fast the CPU accesses the Rx FIFO. *Figure 22-10. Rx FIFO overflow handling* shows an offset of two with respect to the get index when reading the Rx FIFO. In this case the two messages stored in element 1 and 2 are lost.

Figure 22-10. Rx FIFO overflow handling



After reading from the Rx FIFO, the number of the last element read has to be written to the Rx FIFO Acknowledge Index (RXFnA.FnA). This increments the get index to that element number. In case the put index has not been incremented to this Rx FIFO element, the Rx FIFO full condition is reset (RXFnS.FnF = '0').

Dedicated Rx Buffers

The M_CAN supports up to 64 dedicated Rx Buffers. The start address of the dedicated Rx

Buffer section is configured via CAN_M_CAN_RXBC.RBSA.

For each Rx Buffer a Standard or Extended Message ID Filter Element with SFEC / EFEC = "111" and SFID2 / EFID2[10:9] = "00" has to be configured (see section ***Standard message ID filter element*** and section ***Extended message ID filter element***).

After a received message has been accepted by a filter element, the message is stored into the Rx Buffer in the Message RAM referenced by the filter element. The format is the same as for an Rx FIFO element. In addition the flag DRX in CAN_M_CAN_IR register (Message stored in Dedicated Rx Buffer) in the interrupt register is set.

Table 22-13. Example filter configuration for Rx buffers

Filter Element	SFID1[10:0] EFID1[28:0]	SFID2[10:9] EFID2[10:9]	SFID2[5:0] EFID2[5:0]
0	ID message 1	00	00 0000
1	ID message 2	00	00 0001
2	ID message 3	00	00 0010

After the last word of a matching received message has been written to the Message RAM, the respective New Data flag in register CAN_M_CAN_NDAT1, CAN_M_CAN_NDAT2 is set. As long as the New Data flag is set, the respective Rx Buffer is locked against updates from received matching frames. The New Data flags have to be reset by the Host by writing a '1' to the respective bit position.

While an Rx Buffer's New Data flag is set, a Message ID Filter Element referencing this specific Rx Buffer will not match, causing the acceptance filtering to continue. Following Message ID Filter Elements may cause the received message to be stored into another Rx Buffer, or into an Rx FIFO, or the message may be rejected, depending on filter configuration.

Rx buffer handling

- Reset interrupt flag DRX in CAN_M_CAN_IR register.
- Read New Data registers.
- Read messages from Message RAM.
- Reset New Data flags of processed messages.

Tx handling

The Tx Handler handles transmission requests for the dedicated Tx Buffers, the Tx FIFO, and the Tx Queue. It controls the transfer of transmit messages to the CAN Core, the Put and Get Indices, and the Tx Event FIFO. Up to 32 Tx Buffers can be set up for message transmission. The CAN mode for transmission (Classic CAN or CAN FD) can be configured separately for each Tx Buffer element. The Tx Buffer element is described in section "Tx buffer element". ***Table 22-14. Possible Configurations for Frame Transmission*** describes the possible configurations for frame transmission.

Table 22-14. Possible Configurations for Frame Transmission

CAN_M_CAN_CCCR	Tx Buffer Element	Frame Transmission

BRSE	FDOE	FDF	BRS	
ignored	0	ignored	ignored	Classic CAN
0	1	0	ignored	Classic CAN
0	1	1	ignored	FD without bit rate switching
1	1	0	ignored	Classic CAN
1	1	1	0	FD without bit rate switching
1	1	1	1	FD with bit rate switching

Note: AUTOSAR requires at least three Tx Queue Buffers and support of transmit cancellation.

The Tx Handler starts a Tx scan of the Message RAM to check for the highest priority pending Tx request (Tx Buffer with lowest Message ID) when the Tx Buffer Request Pending register CAN_M_CAN_TXBRP is updated. CAN_M_CAN_TXBRP is updated when a transmission finishes, or after the Host has requested a transmission by writing to CAN_M_CAN_TXBAR, or when the Host has cancelled a pending transmission by writing to CAN_M_CAN_TXBCR.

When there is a new scan-result, the temporary buffers holding the preloaded first four RAM words of the two messages with the highest Tx priority are updated. The time required for a Tx scan and the preloading depends on the Host clock frequency, on the number of configured Tx Buffers, and on the number of M_CANs connected the Message RAM.

As the Tx scan of the Message RAM takes some time, the following scenarios are possible:

- Temporary buffer preloaded before ongoing transmission/reception finished: Transmission of preloaded Tx message starts at first opportunity.
- Preloading of temporary buffer not completed before ongoing transmission/reception finished: Tx message cannot be started at first opportunity. In this case another node may start a transmission without having to arbitrate against the transmit message of the M_CAN. If this other message has lower priority, that may be regarded as an outer priority inversion.

Transmit pause

The transmit pause feature is intended for use in CAN systems where the CAN message identifiers are (permanently) specified to specific values and cannot easily be changed. These message identifiers may have a higher CAN arbitration priority than other defined messages, while in a specific application their relative arbitration priority should be inverse. This may lead to a case where one ECU sends a burst of CAN messages that cause another ECU's CAN messages to be delayed because that other messages have a lower CAN arbitration priority.

If e.g. CAN ECU-1 has the transmit pause feature enabled and is requested by its application software to transmit four messages, it will, after the first successful message transmission, wait for two CAN bit times of bus idle before it is allowed to start the next requested message. If there are other ECUs with pending messages, those messages are started in the idle time, they would not need to arbitrate with the next message of ECU-1. After having received a message, ECU-1 is allowed to start its next transmission as soon as the received message

releases the CAN bus.

The transmit pause feature is controlled by bit TXP in CAN_M_CAN_CCCR register. If the bit is set, the M_CAN will, each time it has successfully transmitted a message, pause for two CAN bit times before starting the next transmission. This enables other CAN nodes in the network to transmit messages even if their messages have lower prior identifiers. Default is transmit pause disabled (CAN_M_CAN_CCCR.TXP = '0').

This feature looses up burst transmissions coming from a single node and it protects against "babbling idiot" scenarios where the application program erroneously requests too many transmissions.

Dedicated Tx buffers

Dedicated Tx Buffers are intended for message transmission under complete control of the Host CPU. Each Dedicated Tx Buffer is configured with a specific Message ID. In case that multiple Tx Buffers are configured with the same Message ID, the Tx Buffer with the lowest buffer number is transmitted first. These Tx buffers shall be requested in ascending order with lowest buffer number first. Alternatively, all Tx buffers configured with the same Message ID can be requested simultaneously by a single write access to CAN_M_CAN_TXBAR.

If the data section has been updated, a transmission is requested by an Add Request via CAN_M_CAN_TXBAR.ARn. The requested messages arbitrate internally with messages from an optional Tx FIFO or Tx Queue and externally with messages on the CAN bus, and are sent out according to their Message ID.

A Dedicated Tx Buffer allocates Element Size 32-bit words in the Message RAM (see ***Table 22-15. Tx Buffer / FIFO / Queue element size***). Therefore the start address of a dedicated Tx Buffer in the Message RAM is calculated by adding transmit buffer index (0…31) * Element Size to the Tx Buffer Start Address (CAN_M_CAN_TXBC.TBSA).

Table 22-15. Tx Buffer / FIFO / Queue element size

CAN_M_CAN_TXESC.TBDS[2:0]	Data Field [bytes]	Element Size [RAM words]
000	8	4
001	12	5
010	16	6
011	20	7
100	24	8
101	32	10
110	48	14
111	64	18

Tx FIFO

Tx FIFO operation is configured by programming TFQM in CAN_M_CAN_TXBC register to '0'. Messages stored in the Tx FIFO are transmitted starting with the message referenced by

the Get Index CAN_M_CAN_TXFQS.TFGI. After each transmission the Get Index is incremented cyclically until the Tx FIFO is empty. The Tx FIFO enables transmission of messages with the same Message ID from different Tx Buffers in the order these messages have been written to the Tx FIFO. The M_CAN calculates the Tx FIFO Free Level (CAN_M_CAN_TXFQS.TFFL) as difference between Get and Put Index. It indicates the number of available (free) Tx FIFO elements.

New transmit messages have to be written to the Tx FIFO starting with the Tx Buffer referenced by the Put Index (CAN_M_CAN_TXFQS.TFQPI). An Add Request increments the Put Index to the next free Tx FIFO element. When the Put Index reaches the Get Index, Tx FIFO Full (CAN_M_CAN_TXFQS.TFQF = '1') is signalled. In this case no further messages should be written to the Tx FIFO until the next message has been transmitted and the Get Index has been incremented.

When a single message is added to the Tx FIFO, the transmission is requested by writing a '1' to the CAN_M_CAN_TXBAR bit related to the Tx Buffer referenced by the Tx FIFO's Put Index.

When multiple (n) messages are added to the Tx FIFO, they are written to n consecutive Tx Buffers starting with the Put Index. The transmissions are then requested via CAN_M_CAN_TXBAR. The Put Index is then cyclically incremented by n. The number of requested Tx buffers should not exceed the number of free Tx Buffers as indicated by the Tx FIFO Free Level.

When a transmission request for the Tx Buffer referenced by the Get Index is cancelled, the Get Index is incremented to the next Tx Buffer with pending transmission request and the Tx FIFO Free Level is recalculated. When transmission cancellation is applied to any other Tx Buffer, the Get Index and the FIFO Free Level remain unchanged.

A Tx FIFO element allocates Element Size 32-bit words in the Message RAM (see ***Table 22-15. Tx Buffer / FIFO / Queue element size***). Therefore the start address of the next available (free) Tx FIFO Buffer is calculated by adding Tx FIFO/Queue Put Index CAN_M_CAN_TXFQS.TFQPI (0…31) * Element Size to the Tx Buffer Start Address (CAN_M_CAN_TXBC.TBSA).

Tx Queue

Tx Queue operation is configured by programming TFQM in CAN_M_CAN_TXBC register to '1'. Messages stored in the Tx Queue are transmitted starting with the message with the lowest Message ID (highest priority). In case that multiple Queue Buffers are configured with the same Message ID, the transmission order depends on numbers of the buffers where the messages were stored for transmission. As these buffer numbers depend on the then current states of the Put index, a prediction of the transmission order is not possible.

New messages have to be written to the Tx Buffer referenced by the Put Index (CAN_M_CAN_TXFQS.TFQPI). The Put Index always points to that free buffer of the Tx Queue with the lowest buffer number. In case that the Tx Queue is full (CAN_M_CAN_TXFQS.TFQF = '1'), the Put Index is not valid and no further message should

be written to the Tx Queue until at least one of the requested messages has been sent out or a pending transmission request has been cancelled.

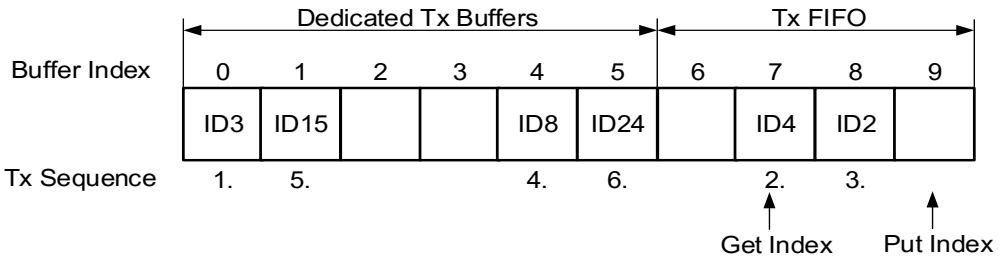
The application may use register CAN_M_CAN_TXBRP instead of the Put Index and may place messages to any Tx Buffer without pending transmission request.

A Tx Queue Buffer allocates Element Size 32-bit words in the Message RAM (see ***Table 22-15. Tx Buffer / FIFO / Queue element size***). Therefore the start address of the next available (free) Tx Queue Buffer is calculated by adding Tx FIFO/Queue Put Index CAN_M_CAN_TXFQS.TFQPI (0…31) • Element Size to the Tx Buffer Start Address (CAN_M_CAN_TXBC.TBSA).

Mixed dedicated Tx buffers / Tx FIFO

In this case the Tx Buffers section in the Message RAM is subdivided into a set of Dedicated Tx Buffers and a Tx FIFO. The number of Dedicated Tx Buffers is configured by NDTB in CAN_M_CAN_TXBC register. The number of Tx Buffers assigned to the Tx FIFO is configured by TFQS in CAN_M_CAN_TXBC register. In case TFQS in CAN_M_CAN_TXBC register is programmed to zero, only Dedicated Tx Buffers are used.

Figure 22-11. Example of mixed configuration dedicated Tx buffers / Tx FIFO



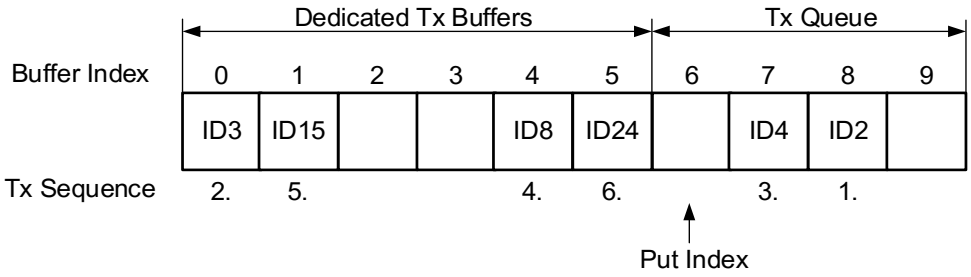
Tx prioritization:

- Scan Dedicated Tx Buffers and oldest pending Tx FIFO Buffer (referenced by TFGI in TXFS register)
- Buffer with lowest Message ID gets highest priority and is transmitted next

Mixed dedicated Tx buffers / Tx queue

In this case the Tx Buffers section in the Message RAM is subdivided into a set of Dedicated Tx Buffers and a Tx Queue. The number of Dedicated Tx Buffers is configured by NDTB in CAN_M_CAN_TXBC register. The number of Tx Queue Buffers is configured by TFQS in CAN_M_CAN_TXBC register. In case TFQS in CAN_M_CAN_TXBC register is programmed to zero, only Dedicated Tx Buffers are used.

Figure 22-12. Example of mixed configuration dedicated Tx Buffers / Tx Queue



Tx prioritization:

- Scan all Tx Buffers with activated transmission request.
- Tx Buffer with lowest Message ID gets highest priority and is transmitted next.

Transmit cancellation

The M_CAN supports transmit cancellation. This feature is especially intended for gateway applications and AUTOSAR based applications. To cancel a requested transmission from a dedicated Tx Buffer or a Tx Queue Buffer the Host has to write a '1' to the corresponding bit position (=number of Tx Buffer) of register CAN_M_CAN_TXBCR. Transmit cancellation is not intended for Tx FIFO operation.

Successful cancellation is signalled by setting the corresponding bit of register CAN_M_CAN_TXBCF to '1'.

In case a transmit cancellation is requested while a transmission from a Tx Buffer is already ongoing, the corresponding CAN_M_CAN_TXBRP bit remains set as long as the transmission is in progress. If the transmission was successful, the corresponding CAN_M_CAN_TXBTO and CAN_M_CAN_TXBCF bits are set. If the transmission was not successful, it is not repeated and only the corresponding CAN_M_CAN_TXBCF bit is set.

Note: In case a pending transmission is cancelled immediately before this transmission could have been started, there follows a short time window where no transmission is started even if another message is also pending in this node. This may enable another node to transmit a message which may have a lower priority than the second message in this node.

Tx event handling

To support Tx event handling the M_CAN has implemented a Tx Event FIFO. After the M_CAN has transmitted a message on the CAN bus, Message ID and timestamp are stored in a Tx Event FIFO element. To link a Tx event to a Tx Event FIFO element, the Message Marker from the transmitted Tx Buffer is copied into the Tx Event FIFO element.

Whether an 8-bit Message Marker or a 16-bit Wide Message Marker is used can be configured by WMM in CAN_M_CAN_CCCR register.

Note: With CAN_M_CAN_CCCR.WMM = '1', internal timestamping is disabled (see section *Tx event FIFO element*).

The Tx Event FIFO can be configured to a maximum of 32 elements. The Tx event FIFO element is described in section ***Tx event FIFO element***.

The purpose of the Tx Event FIFO is to decouple handling transmit status information from transmit message handling i.e. a Tx Buffer holds only the message to be transmitted, while the transmit status is stored separately in the Tx Event FIFO. This has the advantage, especially when operating a dynamically managed transmit queue, that a Tx Buffer can be used for a new message immediately after successful transmission. There is no need to save transmit status information from a Tx Buffer before overwriting that Tx Buffer.

When a Tx Event FIFO full condition is signalled by TEFF in CAN_M_CAN_IR register, no further elements are written to the Tx Event FIFO until at least one element has been read out and the Tx Event FIFO Get Index has been incremented. In case a Tx event occurs while the Tx Event FIFO is full, this event is discarded and interrupt flag TEFL in CAN_M_CAN_IR register is set.

To avoid a Tx Event FIFO overflow, the Tx Event FIFO watermark can be used. When the Tx Event FIFO fill level reaches the Tx Event FIFO watermark configured by EFWM in CAN_M_CAN_TXEFC register, interrupt flag TEFW in CAN_M_CAN_IR register is set.

When reading from the Tx Event FIFO, two times the Tx Event FIFO Get Index (CAN_M_CAN_TXEFS.EFGI) has to be added to the Tx Event FIFO start address (CAN_M_CAN_TXEFC.EFSA).

FIFO acknowledge handling

The Get Indices of Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO are controlled by writing to the corresponding FIFO Acknowledge Index (see section ***Rx FIFO 0 acknowledge register (CAN_M_CAN_RXF0A)***", section ***Rx FIFO 1 acknowledge register (CAN_M_CAN_RXF1A)***, and section ***Tx event FIFO acknowledge register (CAN_M_CAN_TXEFA)***). Writing to the FIFO Acknowledge Index will set the FIFO Get Index to the FIFO Acknowledge Index plus one and thereby updates the FIFO Fill Level. There are two use cases:

When only a single element has been read from the FIFO (the one being pointed to by the Get Index), this Get Index value is written to the FIFO Acknowledge Index.

When a sequence of elements has been read from the FIFO, it is sufficient to write the FIFO Acknowledge Index only once at the end of that read sequence (value: Index of the last element read), to update the FIFO's Get Index.

Due to the fact that the CPU has free access to the M_CAN's Message RAM, special care has to be taken when reading FIFO elements in an arbitrary order (Get Index not considered). This might be useful when reading a High Priority Message from one of the two Rx FIFOs. In this case the FIFO's Acknowledge Index should not be written because this would set the Get Index to a wrong position and also alters the FIFO's Fill Level. In this case some of the older FIFO elements would be lost.

Note: The application has to ensure that a valid value is written to the FIFO Acknowledge Index. The M_CAN does not check for erroneous values.

22.6.4. Programmer's model

Hardware reset description

The Bus_Off state is reset and the output M_CAN Tx output is set to recessive (HIGH). The value 0x0001 (CAN_M_CAN_CCCR.INIT = '1') in the CC Control Register enables software initialization. The M_CAN does not influence the CAN bus until the CPU resets INIT in CAN_M_CAN_CCCR register to '0'.

Message RAM

For storage of Rx/Tx messages and for storage of the filter configuration a single- or dual-ported Message RAM has to be connected to the M_CAN module.

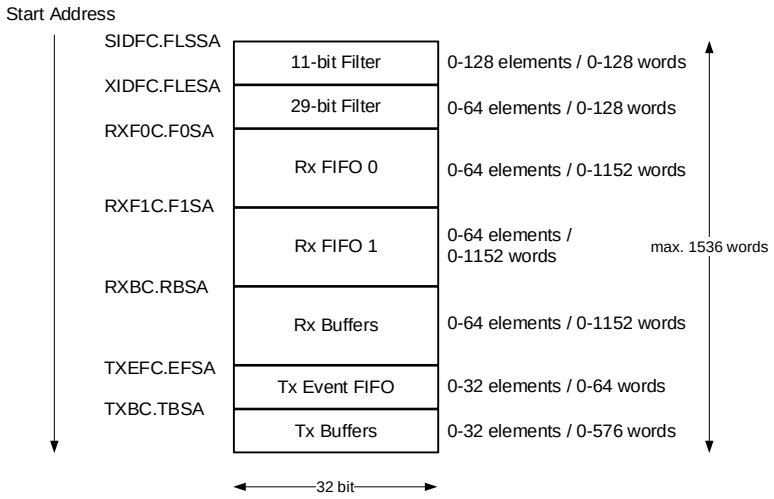
Note: In case the Message RAM is equipped with parity or ECC functionality, it is recommended to initialize the Message RAM after hardware reset by writing e.g. 0x00000000 to each Message RAM word to create valid parity/ECC checksums.

Message RAM configuration

The Message RAM has a width of 32 bits. In case parity checking or ECC is used a respective number of bits has to be added to each word. The M_CAN module can be configured to allocate up to 4352 words in the Message RAM. It is not necessary to configure each of the sections listed in ***Figure 22-13. Message RAM configuration***, nor is there any restriction with respect to the sequence of the sections.

When operated in CAN FD mode the required Message RAM size strongly depends on the element size configured for Rx FIFO0, Rx FIFO1, Rx Buffers, and Tx Buffers via CAN_M_CAN_RXESC.F0DS, CAN_M_CAN_RXESC.F1DS, CAN_M_CAN_RXESC.RBDS, and CAN_M_CAN_TXESC.TBDS.

Figure 22-13. Message RAM configuration



When the M_CAN addresses the Message RAM it addresses 32-bit words, not single bytes. The configurable start addresses are 32-bit word addresses i.e. only bits 15 to 2 are evaluated,

the two least significant bits are ignored.

Note: The M_CAN does not check for erroneous configuration of the Message RAM. Especially the configuration of the start addresses of the different sections and the number of elements of each section has to be done carefully to avoid falsification or loss of data.

Rx buffer and FIFO element

Up to 64 Rx buffers and two Rx FIFOs can be configured in the message RAM. Each Rx FIFO section can be configured to store up to 64 received messages. The structure of a Rx buffer / FIFO element is shown in the following. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field via register CAN_M_CAN_RXESC.

R1A: R1A.RXTS[15:0] holds the 16-bit timestamp generated by the M_CAN's internal timestamping logic.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R0	ESI	XTD	RTR	ID[28:0]																												
R1A	ANMF	FIDX[6:0]							Reserved	FDF	BRS	DLC[3:0]				RXTS[15:0]																
R2	DB3[7:0]								DB2[7:0]								DB1[7:0]								DB0[7:0]							
R3	DB7[7:0]								DB6[7:0]								DB5[7:0]								DB4[7:0]							
…	…								…								…								…							
Rn	DBm[7:0]								DBm-1[7:0]								DBm-2[7:0]								DBm-3[7:0]							

Bits	Fields	Descriptions
R0 31	ESI	Error state indicator 0: Transmitting node is error active 1: Transmitting node is error passive
R0 30	XTD	Extended identifier Signals to the Host whether the received frame has a standard or extended identifier. 0: 11-bit standard identifier 1: 29-bit extended identifier
R0 29	RTR	Remote transmission request Signals to the Host whether the received frame is a data frame or a remote frame. 0: Received frame is a data frame 1: Received frame is a remote frame **Note:** There are no remote frames in CAN FD format. In CAN FD frames (FDF = 1'), the dominant RRS (Remote Request Substitution) bit replaces bit RTR (Remote Transmission Request).
R0 28:0	ID[28:0]	Identifier Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].
R1A 31	ANMF	Accepted non-matching frame Acceptance of non-matching frames may be enabled via

		CAN_M_CAN_GFC.ANFS and CAN_M_CAN_GFC.ANFE. 0: Received frame matching filter index FIDX 1: Received frame did not match any Rx filter element
R1A 30:24	FIDX[6:0]	Filter index 0-127: Index of matching Rx acceptance filter element (invalid if ANMF = '1'). Range is 0 to CAN_M_CAN_SIDFC.LSS - 1 resp. CAN_M_CAN_XIDFC.LSE - 1.
R1A 23:22	Reserved	Must be kept at reset value.
R1A 21	FDF	FD format 0: Classical CAN frame format 1: CAN FD frame format (new DLC-coding and CRC)
R1A 20	BRS	Bit rate switch 0: Frame received without bit rate switching 1: Frame received with bit rate switching
R1A 19:16	DLC[3:0]	Data length code 0-8: CAN + CAN FD: received frame has 0-8 data bytes 9-15: CAN: received frame has 8 data bytes 9-15: CAN FD: received frame has 12/16/20/24/32/48/64 data bytes
R1A 15:0	RXTS[15:0]	Rx timestamp Timestamp Counter value captured on start of frame reception. Resolution depending on configuration of the Timestamp Counter Prescaler (CAN_M_CAN_TSCC.TCP). **Note:** For CAN FD frames, these bits are not effective.
R2 31:24	DB3[7:0]	Data byte 3
R2 23:16	DB2[7:0]	Data byte 2
R2 15:8	DB1[7:0]	Data byte 1
R2 7:0	DB0[7:0]	Data byte 0
R3 31:24	DB7[7:0]	Data byte 7
R3 23:16	DB6[7:0]	Data byte 6
R3 15:8	DB5[7:0]	Data byte 5
R3 7:0	DB4[7:0]	Data byte 4
…	…	…
Rn 31:24	DBm[7:0]	Data byte m
Rn 23:16	DBm-1[7:0]	Data byte m-1
Rn 15:8	DBm-2[7:0]	Data byte m-2

Rn 7:0	DBm-3[7:0]	Data byte m-3

Note: Depending on the configuration of the element size (CAN_M_CAN_RXESC), between two and sixteen 32-bit words (Rn = 2...17) are used for storage of a CAN message's data field.

Tx buffer element

The Tx Buffers section can be configured to hold dedicated Tx Buffers as well as a Tx FIFO / Tx Queue. In case that the Tx Buffers section is shared by dedicated Tx buffers and a Tx FIFO / Tx Queue, the dedicated Tx Buffers start at the beginning of the Tx Buffers section followed by the buffers assigned to the Tx FIFO or Tx Queue. The Tx Handler distinguishes between dedicated Tx Buffers and Tx FIFO / Tx Queue by evaluating the Tx Buffer configuration CAN_M_CAN_TXBC.TFQS and CAN_M_CAN_TXBC.NDTB. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field via register CAN_M_CAN_TXESC.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T0	ESI	XTD	RTR	ID[28:0]																											
T1	MM[7:0]							EFC	Reserved	FDF	BRS	DLC[3:0]				MM[15:8]								Reserved							
T2	DB3[7:0]							DB2[7:0]								DB1[7:0]								DB0[7:0]							
T3	DB7[7:0]							DB6[7:0]								DB5[7:0]								DB4[7:0]							
…	…							…								…								…							
Tn	DBm[7:0]							DBm-1[7:0]								DBm-2[7:0]								DBm-3[7:0]							

Bits	Fields	Descriptions
T0 31	ESI	Error state indicator 0: ESI bit in CAN FD format depends only on error passive flag 1: ESI bit in CAN FD format transmitted recessive **Note:** The ESI bit of the transmit buffer is or'ed with the error passive flag to decide the value of the ESI bit in the transmitted FD frame. As required by the CAN FD protocol specification, an error active node may optionally transmit the ESI bit recessive, but an error passive node will always transmit the ESI bit recessive
T0 30	XTD	Extended identifier 0: 11-bit standard identifier 1: 29-bit extended identifier
T0 29	RTR	Remote transmission request 0: Transmit data frame 1: Transmit remote frame **Note:** When RTR = '1', the M_CAN transmits a remote frame according to ISO 11898-1:2015, even if CAN_M_CAN_CCCR.FDOE enables the transmission in CAN FD format.

T0 28:0	ID[28:0]	Identifier Standard or extended identifier depending on bit XTD. A standard identifier has to be written to ID[28:18].
T1 31:24	MM[7:0]	Message marker Written by CPU during Tx Buffer configuration. Copied into Tx Event FIFO element for identification of Tx message status.
T1 23	EFC	Event FIFO control 0: Don't store Tx events 1: Store Tx events
T1 22	Reserved	Must be kept at reset value.
T1 21	FDF	FD format 0: Frame transmitted in Classic CAN format 1: Frame transmitted in CAN FD format
T1 20	BRS	Bit rate switch 0: CAN FD frames transmitted without bit rate switching 1: CAN FD frames transmitted with bit rate switching **Note:** Bits ESI, FDF, and BRS are only evaluated when CAN FD operation is enabled CAN_M_CAN_CCCR.FDOE = '1'. Bit BRS is only evaluated when in addition CAN_M_CAN_CCCR.BRSE = '1'.
T1 19:16	DLC[3:0]	Data length code 0-8: CAN + CAN FD: transmit frame has 0-8 data bytes 9-15: CAN: transmit frame has 8 data bytes 9-15: CAN FD: transmit frame has 12/16/20/24/32/48/64 data bytes
T1 15:8	MM[15:8]	Message marker High byte of Wide Message Marker, written by CPU during Tx Buffer configuration. Copied into Tx Event FIFO element for identification of Tx message status. Available only when CAN_M_CAN_CCCR.WMM = '1'.
T1 7:0	Reserved	Must be kept at reset value.
T2 31:24	DB3[7:0]	Data byte 3
T2 23:16	DB2[7:0]	Data byte 2
T2 15:8	DB1[7:0]	Data byte 1
T2 7:0	DB0[7:0]	Data byte 0
T3 31:24	DB7[7:0]	Data byte 7
T3 23:16	DB6[7:0]	Data byte 6
T3 15:8	DB5[7:0]	Data byte 5
T3 7:0	DB4[7:0]	Data byte 4

…	…	…
Tn 31:24	DBm[7:0]	Data byte m
Tn 23:16	DBm-1[7:0]	Data byte m-1
Tn 15:8	DBm-2[7:0]	Data byte m-2
Tn 7:0	DBm-3[7:0]	Data byte m-3

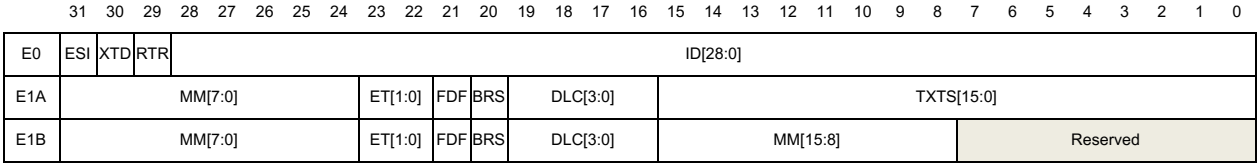
Note: Depending on the configuration of the element size (CAN_M_CAN_TXESC), between two and sixteen 32-bit words (Tn = 2...17) are used for storage of a CAN message's data field.

Tx event FIFO element

Each element stores information about transmitted messages. By reading the Tx Event FIFO the Host CPU gets this information in the order the messages were transmitted. Status information about the Tx Event FIFO can be obtained from register CAN_M_CAN_TXEFS.

E1A: When CAN_M_CAN_CCCR.WMM = '0', E1A.TXTS[15:0] holds the 16-bit timestamp generated by the M_CAN's internal timestamping logic.

E1B: When 16-bit Message Markers are enabled (CAN_M_CAN_CCCR.WMM = '1'), E1B.MM[15:8] holds the upper 8 bit of the Wide Message Marker.

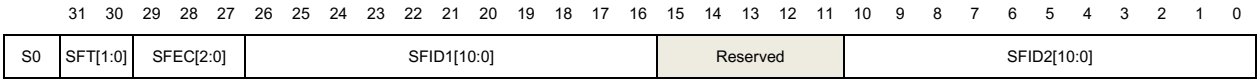


Bits	Fields	Descriptions
E0 31	ESI	Error state indicator 0: Transmitting node is error active 1: Transmitting node is error passive
E0 30	XTD	Extended identifier 0: 11-bit standard identifier 1: 29-bit extended identifier
E0 29	RTR	Remote transmission request 0: Data frame transmitted 1: Remote frame transmitted
E0 28:0	ID[28:0]	Identifier Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].
E1A/B 31:24	MM[7:0]	Message marker Copied from Tx Buffer into Tx Event FIFO element for identification of Tx message status.
E1A/B 23:22	ET[1:0]	Event type

		00: Reserved 01: Tx event 10: Transmission in spite of cancellation (always set for transmissions in DAR mode) 11: Reserved
E1A/B 21	FDF	FD format 0: Classical CAN frame format 1: CAN FD frame format (new DLC-coding and CRC)
E1A/B 20	BRS	Bit rate switch 0: Frame transmitted without bit rate switching 1: Frame transmitted with bit rate switching
E1A/B 19:16	DLC[3:0]	Data length code 0-8: CAN + CAN FD: transmit frame has 0-8 data bytes transmitted 9-15: CAN: transmit frame has 8 data bytes transmitted 9-15: CAN FD: transmit frame has 12/16/20/24/32/48/64 data bytes transmitted
E1A 15:0	TXTS[15:0]	Tx timestamp Timestamp Counter value captured on start of frame transmission. Resolution depending on configuration of the Timestamp Counter Prescaler (CAN_M_CAN_TSCC.TCP). **Note:** For CAN FD frames, these bits are not effective.
E1B 15:8	MM[15:8]	Message marker High byte of Wide Message Marker, written by CPU during Tx Buffer configuration. Copied into Tx Event FIFO element for identification of Tx message status.
E1B 7:0	Reserved	Must be kept at reset value.

Standard message ID filter element

Up to 128 filter elements can be configured for 11-bit IDs. When accessing a Standard Message ID Filter element, its address is the Filter List Standard Start Address (CAN_M_CAN_SIDFC.FLSSA) plus the index of the filter element (0…127).

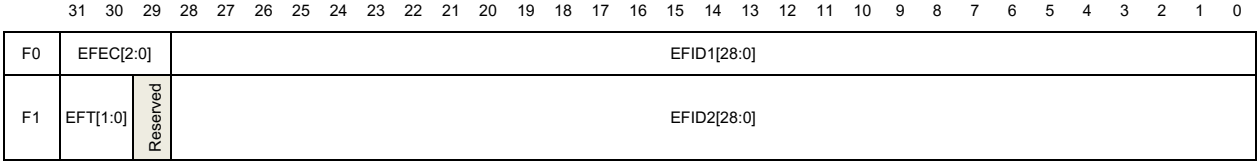


Bits	Fields	Descriptions
31:30	SFT[1:0]	Standard filter type 00: Range filter from SFID1 to SFID2 (SFID2 ≥ SFID1) 01: Dual ID filter for SFID1 or SFID2 10: Classic filter: SFID1 = filter, SFID2 = mask 11: Filter element disabled **Note:** With SFT = "11" the filter element is disabled and the acceptance filtering continues (same behaviour as with SFEC = "000")
29:27	SFEC[2:0]	Standard filter element configuration

		All enabled filter elements are used for acceptance filtering of 11-bit ID frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If SFEC = "100", "101", or "110" a match sets interrupt flag HPM in CAN_M_CAN_IR register and, if enabled, an interrupt is generated. In this case register CAN_M_CAN_HPMS is updated with the status of the priority match. 000: Disable filter element 001: Store in Rx FIFO 0 if filter matches 010: Store in Rx FIFO 1 if filter matches 011: Reject ID if filter matches, not intended to be used with Sync messages 100: Set priority if filter matches, not intended to be used with Sync messages, no storage 101: Set priority and store in FIFO 0 if filter matches 110: Set priority and store in FIFO 1 if filter matches 111: Store into Rx Buffer, configuration of SFT[1:0] ignored
26:16	SFID1[10:0]	Standard filter ID 1 First ID of standard ID filter element. When filtering for Rx Buffers, Sync messages, this field defines the ID of the message to be stored. The received identifiers must match exactly, no masking mechanism is used.
15:11	Reserved	Must be kept at reset value.
10:0	SFID2[10:0]	Standard filter ID 2 This bit field has a different meaning depending on the configuration of SFEC: SFEC = "001"..."110": Second ID of standard ID filter element SFEC = "111": Filter for Rx Buffers SFID2[10:9] = "00": Ddecides the received message is stored into an Rx Buffer SFID2[8:6] is used to control the filter event pins at the Extension Interface. A one at the respective bit position enables generation of a pulse at the related filter event pin with the duration of one Host clock period in case the filter matches. SFID2[5:0] defines the offset to the Rx Buffer Start Address (CAN_M_CAN_RXBC.RBSA) for storage of a matching message.

Extended message ID filter element

Up to 64 filter elements can be configured for 29-bit IDs. When accessing an Extended Message ID Filter element, its address is the Filter List Extended Start Address CAN_M_CAN_XIDFC.FLESA plus two times the index of the filter element (0…63).



Bits	Fields	Descriptions
F0 31:29	EFEC[2:0]	Extended filter element configuration

		All enabled filter elements are used for acceptance filtering of 29-bit ID frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If EFEC = "100", "101", or "110" a match sets interrupt flag HPM in CAN_M_CAN_IR register and, if enabled, an interrupt is generated. In this case register CAN_M_CAN_HPMS is updated with the status of the priority match. 000: Disable filter element 001: Store in Rx FIFO 0 if filter matches 010: Store in Rx FIFO 1 if filter matches 011: Reject ID if filter matches, not intended to be used with Sync messages 100: Set priority if filter matches, not intended to be used with Sync messages, no storage 101: Set priority and store in FIFO 0 if filter matches 110: Set priority and store in FIFO 1 if filter matches 111: Store into Rx Buffer, configuration of EFT[1:0] ignored
F0 28:0	EFID1[28:0]	Extended filter ID 1 First ID of extended ID filter element. When filtering for Rx Buffers, Sync messages, this field defines the ID of an extended message to be stored. The received identifiers must match exactly, only CAN_M_CAN_XIDAM masking mechanism (see section ***Extended message ID filter element***) is used.
F1 31:30	EFT[1:0]	Extended filter type 00: Range filter from EFID1 to EFID2 (EFID2 ≥ EFID1) 01: Dual ID filter for EFID1 or EFID2 10: Classic filter: EFID1 = filter, EFID2 = mask 11: Range filter from EFID1 to EFID2 (EFID2 ≥ EFID1), CAN_M_CAN_XIDAM mask not applied
F1 29	Reserved	Must be kept at reset value.
F1 28:0	EFID2[28:0]	Extended filter ID 2 This bit field has a different meaning depending on the configuration of EFEC: EFEC = "001"..."110": Second ID of extended ID filter element EFEC = "111": Filter for Rx Buffers EFID2[10:9] = "00": Decides the received message is stored into an Rx Buffer EFID2[8:6] is used to control the filter event pins at the Extension Interface. A one at the respective bit position enables generation of a pulse at the related filter event pin with the duration of one Host clock period in case the filter matches. EFID2[5:0] defines the offset to the Rx Buffer Start Address (CAN_M_CAN_RXBC.RBSA) for storage of a matching message.

22.6.5. Register definition

Register base address reference section ***CAN cluster register definition***.

Table 22-16. CAN register overview

Register Name	Symbol	Address Offset	Reset Value
Core release register	CAN_M_CAN_CREL	0x00	0x3313 0327
Endian register	CAN_M_CAN_ENDN	0x04	0x8765 4321
Data bit timing and prescaler register	CAN_M_CAN_DBTP	0x0C	0x0000 0A33
Test register	CAN_M_CAN_TEST	0x10	0x0000 0000
RAM watchdog	CAN_M_CAN_RWD	0x14	0x0000 0000
CC control register	CAN_M_CAN_CCCR	0x18	0x0000 0001
Nominal bit timing & prescaler register	CAN_M_CAN_NBTP	0x1C	0x0600 0A03
Timestamp counter configuration register	CAN_M_CAN_TSCC	0x20	0x0000 0000
Timestamp counter value register	CAN_M_CAN_TSCV	0x24	0x0000 0000
Timeout counter configuration register	CAN_M_CAN_TOCC	0x28	0xFFFF 0000
Timeout counter value register	CAN_M_CAN_TOCV	0x2C	0x0000 FFFF
Error counter register	CAN_M_CAN_ECR	0x40	0x0000 0000
Protocol status register	CAN_M_CAN_PSR	0x44	0x0000 0707
Transmitter delay compensation register	CAN_M_CAN_TDCR	0x48	0x0000 0000
Interrupt register	CAN_M_CAN_IR	0x50	0x0000 0000
Interrupt enable	CAN_M_CAN_IE	0x54	0x0000 0000
Interrupt line select register	CAN_M_CAN_ILS	0x58	0x0000 0000
Interrupt line enable register	CAN_M_CAN_ILE	0x5C	0x0000 0000
Global filter configuration	CAN_M_CAN_GFC	0x80	0x0000 0000
Standard ID filter configuration register	CAN_M_CAN_SIDFC	0x84	0x0000 0000
Extended ID filter configuration register	CAN_M_CAN_XIDFC	0x88	0x0000 0000
Extended ID and mask register	CAN_M_CAN_XIDAM	0x90	0x1FFF FFFF
High priority message status	CAN_M_CAN_HPMS	0x94	0x0000 0000
New data 1 register	CAN_M_CAN_NDAT1	0x98	0x0000 0000
New data 2 register	CAN_M_CAN_NDAT2	0x9C	0x0000 0000
Rx FIFO 0 Configuration	CAN_M_CAN_RXF0C	0xA0	0x0000 0000
Rx FIFO 0 status register	CAN_M_CAN_RXF0S	0xA4	0x0000 0000
Rx FIFO 0 acknowledge register	CAN_M_CAN_RXF0A	0xA8	0x0000 0000
Rx buffer configuration	CAN_M_CAN_RXBC	0xAC	0x0000 0000
Rx FIFO 1	CAN_M_CAN_RXF1C	0xB0	0x0000 0000

Register Name	Symbol	Address Offset	Reset Value
configuration			
Rx FIFO 1 status register	CAN_M_CAN_RXF1S	0xB4	0x0000 0000
Rx FIFO 1 acknowledge register	CAN_M_CAN_RXF1A	0xB8	0x0000 0000
Rx buffer / FIFO element size configuration	CAN_M_CAN_RXESC	0xBC	0x0000 0000
Tx buffer configuration	CAN_M_CAN_TXBC	0xC0	0x0000 0000
Tx FIFO / Queue status	CAN_M_CAN_TXFQS	0xC4	0x0000 0000
Tx Buffer Element Size Configuration	CAN_M_CAN_TXESC	0xC8	0x0000 0000
Tx buffer request pending	CAN_M_CAN_TXBRP	0xCC	0x0000 0000
Tx buffer add request	CAN_M_CAN_TXBAR	0xD0	0x0000 0000
Tx buffer cancellation request	CAN_M_CAN_TXBCR	0xD4	0x0000 0000
Tx buffer transmission occurred	CAN_M_CAN_TXBTO	0xD8	0x0000 0000
Tx buffer cancellation finished	CAN_M_CAN_TXBCF	0xDC	0x0000 0000
Tx buffer transmission interrupt enable	CAN_M_CAN_TXBTIE	0xE0	0x0000 0000
Tx buffer cancellation finished interrupt enable	CAN_M_CAN_TXBCIE	0xE4	0x0000 0000
Tx event FIFO configuration	CAN_M_CAN_TXEFC	0xF0	0x0000 0000
Tx event FIFO status	CAN_M_CAN_TXEFS	0xF4	0x0000 0000
Tx event FIFO acknowledge register	CAN_M_CAN_TXEFA	0xF8	0x0000 0000

Core release register (CAN_M_CAN_CREL)

Address offset: 0x00

Reset value: 0x3313 0327

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).



Bits	Fields	Descriptions
31:28	REL[3:0]	Core release (One digit, BCD-coded)
27:24	STEP[3:0]	Step of core release (One digit, BCD-coded)

23:20	SUBSTEP[3:0]	Sub-step of core release (One digit, BCD-coded)
19:16	YEAR[3:0]	Time stamp year (One digit, BCD-coded. This field is set by generic parameter on synthesis.)
15:8	MON[7:0]	Time stamp month (Two digit, BCD-coded. This field is set by generic parameter on synthesis.)
7:0	DAY[7:0]	Time stamp day (Two digit, BCD-coded. This field is set by generic parameter on synthesis.) **Note:** BCD 16#YMMDD# = 201Y-MM-DD

Endian register (CAN_M_CAN_ENDN)

Address offset: 0x04
Reset value: 0x8765 4321

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETV[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETV[15:0]															
r															

Bits	Fields	Descriptions
31:0	ETV[31:0]	Endianness test value The endianness test value is 0x87654321.

Data bit timing and prescaler register (CAN_M_CAN_DBTP)

Address offset: 0x0C
Reset value: 0x0000 0A33

The CAN bit time may be programmed in the range of 4 to 49 time quanta. The CAN time quantum may be programmed in the range of 1 to 32 M_CAN clock periods.

t_q=(DBRP+1)*M_CAN clock period.

(DTSEG1+1) is the sum of Prop_Seg and Phase_Seg1. (DTSEG2+1) is Phase_Seg2.

Therefore the length of the bit time is (programmed values) [DTSEG1+DTSEG2+3]*t_q or (functional values) [Sync_Seg+Prop_Seg+Phase_Seg1+Phase_Seg2]*t_q.

[DTSEG1+DTSEG2+3]*t_q=[SYNC+(DTSEG1+1)+(DTSEG2+1)]*t_q
= [Sync_Seg+(Prop_Seg+Phase_Seg1)+Phase_Seg2]*t_q

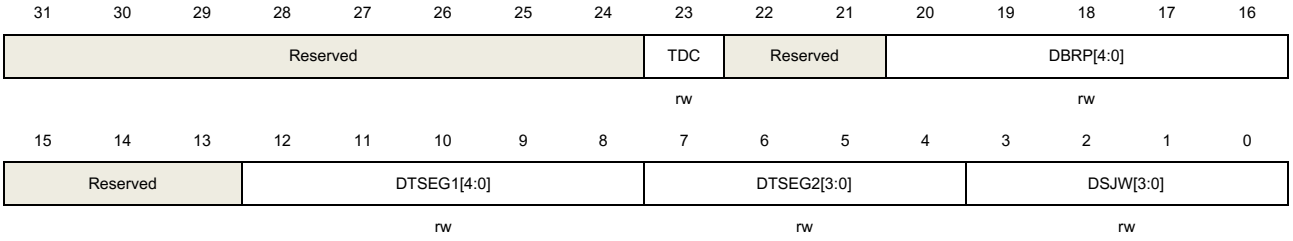
The Information Processing Time (IPT) is zero, meaning that the data for the next bit is available at the first clock edge after the sample point.

Note: With a CAN clock of 8 MHz, the reset value of 0x00000A33 configures the M_CAN for a data phase bit rate of 500 kBit/s.

The bit rate configured for the CAN FD data phase via CAN_M_CAN_DBTP must be higher or equal to the bit rate configured for the arbitration phase via CAN_M_CAN_NBTP.

For communication reliability, it is recommended that each bit time be configured with at least 10 time quanta (tq).

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.



Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23	TDC	Transceiver delay compensation 0: Transceiver delay compensation disabled 1: Transceiver delay compensation enabled
22:21	Reserved	Must be kept at reset value.
20:16	DBRP[4:0]	Data bit rate prescaler The value by which the oscillator frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Baud Rate Prescaler are 0 to 31. 0x00: divides by 1 0x01: divides by 2 0x02: divides by 3 … 0x1F: divides by 32 When TDC = 1, the range is limited to 0, 1. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.
15:13	Reserved	Must be kept at reset value.
12:8	DTSEG1[4:0]	Data time segment before sample point Valid values are 0 to 31. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. 0x00: Data time segment before sample point is 1 0x01: Data time segment before sample point is 2 0x02: Data time segment before sample point is 3 …

		0x1F: Data time segment before sample point is 32
7:4	DTSEG2[4:0]	Data time segment after sample point Valid values are 1 to 15. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. 0x01: Data time segment after sample point is 2 0x02: Data time segment after sample point is 3 … 0x0F: Data time segment after sample point is 16
3:0	DSJW[3:0]	Data (re) synchronization jump width Valid values are 0 to 15. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. 0x00: Data (re) synchronization jump width is 1 0x01: Data (re) synchronization jump width is 2 0x02: Data (re) synchronization jump width is 3 … 0x0F: Data (re) synchronization jump width is 16

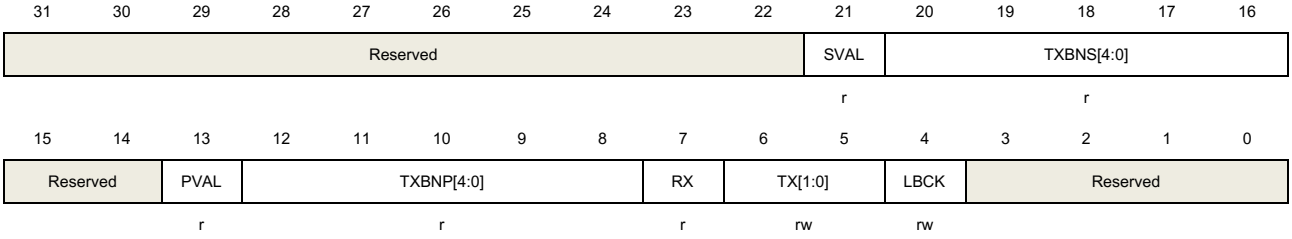
Test register (CAN_M_CAN_TEST)

Address offset: 0x10

Reset value: 0x0000 0000

Write access to the Test Register has to be enabled by setting TEST bit in CAN_M_CAN_CCCR register to 1. All Test Register functions are set to their reset values when TEST bit in CAN_M_CAN_CCCR register is reset. Loop Back mode and software control of M_CAN Tx pin are hardware test modes. Programming of TX != 00 may disturb the message transfer on the CAN bus.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.



Bits	Fields	Descriptions
31:22	Reserved	Must be kept at reset value.
21	SVAL	Started Valid 0: Value of TXBNS not valid 1: Value of TXBNS valid
20:16	TXBNS[4:0]	Tx Buffer number started

		Tx Buffer number of message whose transmission was started last. Valid when SVAL is set. 0x00: Tx Buffer number started is 0 0x01: Tx Buffer number started is 1 0x02: Tx Buffer number started is 2 … 0x1F: Tx Buffer number started is 31
15:14	Reserved	Must be kept at reset value.
13	PVAL	Prepared Valid 0: Value of TXBNP not valid 1: Value of TXBNP valid
12:8	TXBNP[4:0]	Tx Buffer number prepared Tx Buffer number of message that is ready for transmission. Valid when PVAL is set. 0x00: Tx Buffer number prepared is 0 0x01: Tx Buffer number prepared is 1 0x02: Tx Buffer number prepared is 2 … 0x1F: Tx Buffer number prepared is 31
7	RX	Receive pin Monitors the actual value of pin CAN Rx pin 0: The CAN bus is dominant (CAN Rx input = 0) 1: The CAN bus is recessive (CAN Rx input = 1)
6:5	TX[1:0]	Control of transmit pin 00: Reset value, CAN Tx output is controlled by the CAN Core, updated at the end of the CAN bit time 01: Sample Point can be monitored at pin CAN Tx output 10: Dominant (0) level at pin CAN Tx output 11: Recessive (1) at pin CAN Tx output
4	LBCK	Loop back mode 0: Reset value, Loop Back Mode is disabled 1: Loop Back Mode is enabled (see section ***Test modes***)
3:0	Reserved	Must be kept at reset value.

RAM watchdog (CAN_M_CAN_RWD)

Address offset: 0x14
Reset value: 0x0000 0000

The RAM Watchdog monitors the READY output of the Message RAM A Message RAM access via the M_CAN's Generic Master Interface starts the Message RAM Watchdog Counter with the value configured by WDC. The counter is reloaded with WDC when the

Message RAM signals successful completion by activating its READY output. In case there is no response from the Message RAM until the counter has counted down to zero, the counter stops and interrupt flag WDI in CAN_M_CAN_IR register is set. The RAM Watchdog Counter is clocked by the Host clock.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDV[7:0]								WDC[7:0]							
r								rw							

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:8	WDV[7:0]	Watchdog value Actual Message RAM Watchdog Counter Value.
7:0	WDC[7:0]	Watchdog configuration Start value of the Message RAM Watchdog Counter. With the reset value of 0x00 the counter is disabled.

CC control register (CAN_M_CAN_CCCR)

Address offset: 0x18

Reset value: 0x0000 0001

For details about setting and resetting of single bits see section ***Software initialization***.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NISO	TXP	EFBI	PXHD	WMM	Reserved	BRSE	FDOE	TEST	DAR	MON	CSR	CSA	ASM	CCE	INIT
rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	r	rw	rw	rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	NISO	Non ISO operation If this bit is set, the M_CAN uses the CAN FD frame format as specified by the Bosch CAN FD Specification V1.0. 0: CAN FD frame format according to ISO11898-1 1: CAN FD frame format according to Bosch CAN FD Specification V1.0

		Note: This bit can only be written when the bit CCE and INIT of CAN_M_CAN_CCCR register are set to 1.
14	TXP	Transmit pause If this bit is set, the M_CAN pauses for two CAN bit times before starting the next transmission after itself has successfully transmitted a frame (see section ***Tx handling***). 0: Transmit pause disabled 1: Transmit pause enabled **Note:** This bit can only be written when the bit CCE and bit INIT of CAN_M_CAN_CCCR register are set to 1.
13	EFBI	Edge filtering during bus integration 0: Edge filtering disabled 1: Two consecutive dominant tq required to detect an edge for hard synchronization **Note:** This bit can only be written when the bit CCE and bit INIT of CAN_M_CAN_CCCR register are set to 1.
12	PXHD	Protocol exception handling disable 0: Protocol exception handling enabled 1: Protocol exception handling disabled **Note:** When protocol exception handling is disabled, the M_CAN transmits an error frame when it detects a protocol exception condition. This bit can only be written when the bit CCE and bit INIT of CAN_M_CAN_CCCR register are set to 1.
11	WMM	Wide message marker Enables the use of 16-bit Wide Message Markers. When 16-bit Wide Message Markers are used (WMM = 1), 16-bit internal timestamping is disabled for the Tx Event FIFO. 0: 8-bit Message Marker used 1: 16-bit Message Marker used, replacing 16-bit timestamps in Tx Event FIFO **Note:** This bit can only be written when the bit CCE and bit INIT of CAN_M_CAN_CCCR register are set to 1.
10	Reserved	Must be kept at reset value.
9	BRSE	Bit rate switch enable 0: Bit rate switching for transmissions disabled 1: Bit rate switching for transmissions enabled **Note:** When CAN FD operation is disabled FDOE = 0, BRSE is not evaluated. This bit can only be written when the bit CCE and bit INIT of CAN_M_CAN_CCCR register are set to 1.
8	FDOE	FD operation enable 0: FD operation disabled 1: FD operation enabled **Note:** This bit can only be written when the bit CCE and bit INIT of

CAN_M_CAN_CCCR register are set to 1.

7	TEST	Test mode enable 0: Normal operation, register TEST holds reset values 1: Test Mode, write access to register TEST enabled **Note:** This bit can only be written when the bit CCE and bit INIT of CAN_M_CAN_CCCR register are set to 1.
6	DAR	Disable automatic retransmission 0: Automatic retransmission of messages not transmitted successfully enabled 1: Automatic retransmission disabled **Note:** This bit can only be written when the bit CCE and bit INIT of CAN_M_CAN_CCCR register are set to 1.
5	MON	Bus monitoring mode 0: Bus Monitoring Mode is disabled 1: Bus Monitoring Mode is enabled **Note:** Bit MON can only be set by the Host when both CCE and INIT are set to 1. The bit can be reset by the Host at any time.
4	CSR	Clock stop request 0: No clock stop is requested 1: Clock stop requested. When clock stop is requested, first INIT and then CSA is set after all pending transfer requests have been completed and the CAN bus reached idle.
3	CSA	Clock stop acknowledge 0: No clock stop acknowledged 1: M_CAN may be set in power down by stopping Host clock and CAN clock
2	ASM	Restricted operation mode 0: Normal CAN operation 1: Restricted Operation Mode active **Note:** Bit ASM is only set by the Host when both CCE and INIT are set to 1. The bit is reset by the Host at any time. For a description of the Restricted Operation Mode see section ***Restricted operation mode***.
1	CCE	Configuration change enable 0: The CPU has no write access to the protected configuration registers 1: The CPU has write access to the protected configuration registers (while the bit INIT of CAN_M_CAN_CCCR is set to 1)
0	INIT	Initialization 0: Normal Operation 1: Initialization is started **Note:** Due to the synchronization mechanism between the two clock domains, there may be a delay until the value written to INIT can be read back. Therefore the programmer has to ensure that the previous value written to INIT has been accepted

by reading INIT before setting INIT to a new value.

Nominal bit timing & prescaler register (CAN_M_CAN_NBTP)

Address offset: 0x1C
Reset value: 0x0600 0A03

The CAN bit time may be programed in the range of 4 to 385 time quanta. The CAN time quantum may be programmed in the range of 1 to 512 M_CAN clock periods. $t_q=(NBRP+1)*mt_q$.

(NTSEG1 + 1) is the sum of Prop_Seg and Phase_Seg1. (NTSEG2 + 1) is Phase_Seg2.

Therefore the length of the bit time is (programmed values) $[NTSEG1+NTSEG2+3]*t_q$ or (functional values) $[Sync_Seg+Prop_Seg+Phase_Seg1+Phase_Seg2]*t_q$.

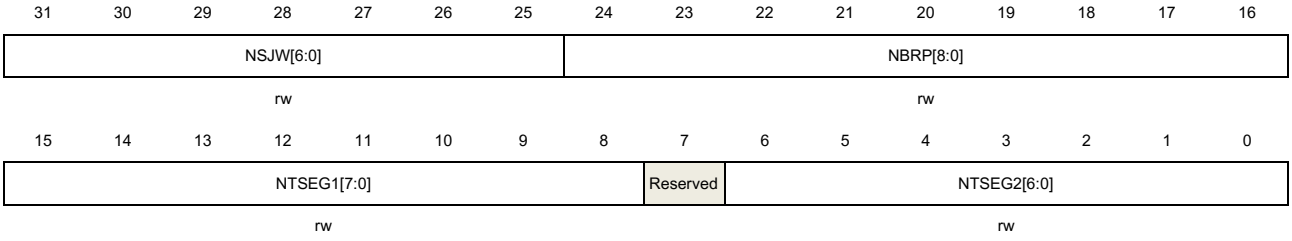
$$[NTSEG1+NTSEG2+3]*t_q=[SYNC+(NTSEG1+1)+(NTSEG2+1)]*t_q$$
$$= [Sync_Seg+(Prop_Seg+Phase_Seg1)+Phase_Seg2]*t_q$$

The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sample point.

Note: With a CAN clock of 8 MHz, the reset value of 0x06000A03 configures the M_CAN for a bit rate of 500 kBit/s.

For communication reliability, it is recommended that each bit time be configured with at least 10 time quanta (tq).

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.



Bits	Fields	Descriptions
31:25	NSJW[6:0]	Nominal (Re) synchronization jump width The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. 0x00: Nominal (Re) synchronization jump width is 1 0x01: Nominal (Re) synchronization jump width is 2 0x02: Nominal (Re) synchronization jump width is 3 … 0x7F: Nominal (Re) synchronization jump width is 128
24:16	NBRP[8:0]	Nominal baud rate prescaler The value by which the oscillator frequency is divided for generating

		the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Baud Rate Prescaler are 0 to 511. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. 0x000: Nominal baud rate prescaler is 1 0x001: Nominal baud rate prescaler is 2 0x002: Nominal baud rate prescaler is 3 … 0x1FF: Nominal baud rate prescaler is 512
15:8	NTSEG1[7:0]	Nominal Time segment before sample point Valid values are 1 to 255. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. 0x01: Nominal Time segment before sample point is 2 0x02: Nominal Time segment before sample point is 3 0x03: Nominal Time segment before sample point is 4 … 0xFF: Nominal Time segment before sample point is 256
7	Reserved	Must be kept at reset value.
6:0	NTSEG2[6:0]	Nominal time segment after sample point Valid values are 1 to 127. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. 0x01: Nominal time segment after sample point is 2 0x02: Nominal time segment after sample point is 3 0x03: Nominal time segment after sample point is 4 … 0x7F: Nominal time segment after sample point is 128

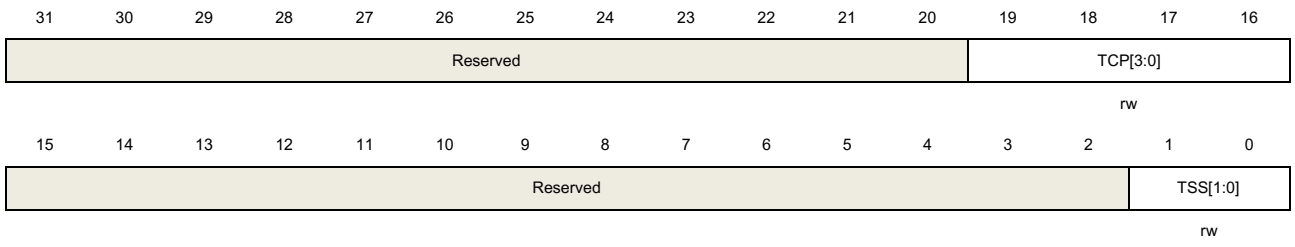
Timestamp counter configuration register (CAN_M_CAN_TSCC)

Address offset: 0x20

Reset value: 0x0000 0000

Configuration of the internal 16-bit Timestamp Counter. Handling of internal timestamps is described in section "Timestamp generation".

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.



Bits	Fields	Descriptions
31:20	Reserved	Must be kept at reset value.
19:16	TCP[3:0]	Timestamp counter prescaler Configures the timestamp and timeout counters time unit in multiples of CAN bit times [1…16]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. 0x00: Timestamp counter prescaler is 1 0x01: Timestamp counter prescaler is 2 0x02: Timestamp counter prescaler is 3 … 0x0F: Timestamp counter prescaler is 16 **Note:** with CAN FD this bit field is not effective.
15:2	Reserved	Must be kept at reset value.
1:0	TSS[1:0]	Timestamp select 00: Timestamp counter value always 0x0000 01: Timestamp counter value incremented according to TCP (Timestamp Counter Prescaler) 10: Reserved 11: Same as 00

Timestamp counter value register (CAN_M_CAN_TSCV)

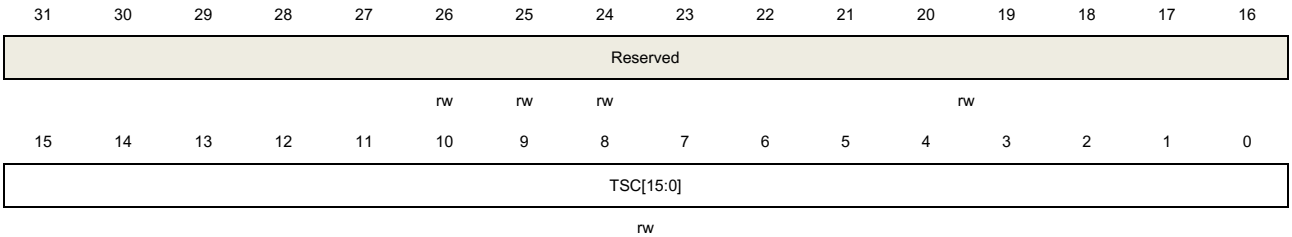
Address offset: 0x24

Reset value: 0x0000 0000

Note: A "wrap around" is a change of the Timestamp Counter value from non-zero to zero not caused by write access to CAN_M_CAN_TSCV.

Writing CAN_M_CAN_TSCV register will reset the Timestamp Counter.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	TSC[15:0]	Timestamp Counter The internal Timestamp Counter value is captured on start of frame (both Rx and Tx). When the bits TSS in CAN_M_CAN_TSCC register = 01, the Timestamp Counter is incremented in multiples of CAN bit times [1…16] depending on the

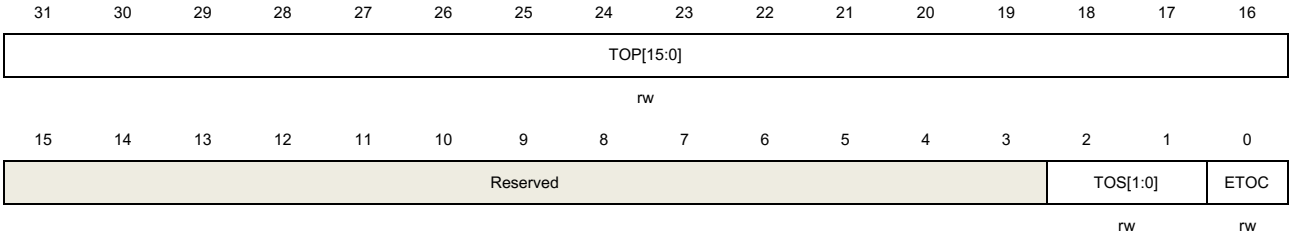
configuration of the bits TCP in CAN_M_CAN_TSCC register. A wrap around sets interrupt flag TSW in CAN_M_CAN_IR register. Write access resets the counter to zero.

Timeout counter configuration register (CAN_M_CAN_TOCC)

Address offset: 0x28
Reset value: 0xFFFF 0000

For a description of the Timeout Counter see section ***Timeout counter***.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.



Bits	Fields	Descriptions
31:16	TOP[15:0]	Timeout period Start value of the Timeout Counter (down-counter). Configures the Timeout Period.
15:3	Reserved	Must be kept at reset value.
2:1	TOS[1:0]	Timeout select When operating in continuous mode, a write to CAN_M_CAN_TOCV presets the counter to the value configured by the TOP and continues down-counting. When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by the TOP. Down-counting is started when the first FIFO element is stored. 00: Continuous operation 01: Timeout controlled by Tx Event FIFO 10: Timeout controlled by Rx FIFO 0 11: Timeout controlled by Rx FIFO 1
0	ETOC	Enable timeout counter 0: Timeout Counter disabled 1: Timeout Counter enabled **Note:** For use of timeout function with CAN FD refer to section ***Timeout counter***.

Timeout counter value register (CAN_M_CAN_TOCV)

Address offset: 0x2C
Reset value: 0x0000 FFFF

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). Clear on

write.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOC[15:0]															
rw															

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	TOC[15:0]	Timeout counter The Timeout Counter is decremented in multiples of CAN bit times [1…16] depending on the configuration of the bits TCP in CAN_M_CAN_TSCC register. When decremented to zero, interrupt flag TOO in CAN_M_CAN_IR register is set and the Timeout Counter is stopped. Start and reset/restart conditions are configured via the bits TOS in CAN_M_CAN_TOCC register.

Note: When the bits TOS in CAN_M_CAN_TOCC register = 0x00, writing register CAN_M_CAN_TOCV will preset the Timeout Counter.

Error counter register (CAN_M_CAN_ECR)

Address offset: 0x40

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								CEL[7:0]							
								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RP	REC[6:0]							TEC[7:0]							
r	r							r							

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:16	CEL[7:0]	CAN error logging The counter is incremented each time when a CAN protocol error causes the 8-bit Transmit Error Counter TEC or the 7-bit Receive Error Counter REC to be incremented. The counter is also incremented when the Bus_Off limit is reached. It is not incremented when only RP is set without changing REC. The increment of CEL follows after the increment of REC or TEC. The counter is reset by read access to CEL. The counter stops at 0xFF; the next increment of TEC or REC sets interrupt flag CAN_M_CAN_IR.ELO. **Note:** Reading CAN_M_CAN_ECR register will reset CEL to zero.

15	RP	Receive error passive 0: The Receive Error Counter is below the error passive level of 128 1: The Receive Error Counter has reached the error passive level of 128
14:8	REC	Receive error counter Actual state of the Receive Error Counter, values between 0 and 127.
7:0	TEC	Transmit error counter Actual state of the Transmit Error Counter, values between 0 and 255.

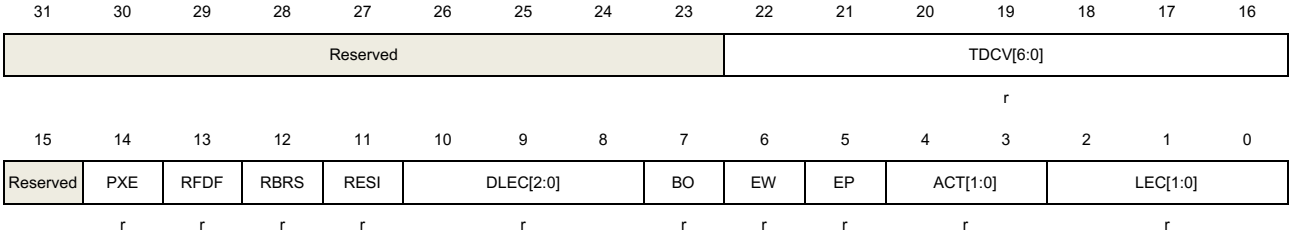
Note: When the bit ASM in CAN_M_CAN_CCCR register is set, the CAN protocol controller does not increment TEC and REC when a CAN protocol error is detected, but CEL is still incremented.

Protocol status register (CAN_M_CAN_PSR)

Address offset: 0x44

Reset value: 0x0000 0707

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).



Bits	Fields	Descriptions
31:23	Reserved	Must be kept at reset value.
22:16	TDCV[6:0]	Transmitter delay compensation value Position of the secondary sample point, defined by the sum of the measured delay from CAN Tx to CAN Rx and the bits TDCO in CAN_M_CAN_TDCR register. The SSP position is, in the data phase, the number of mtq between the start of the transmitted bit and the secondary sample point. 0x00: Transmitter delay compensation value is 0 0x01: Transmitter delay compensation value is 1 0x02: Transmitter delay compensation value is 2 … 0x7F: Transmitter delay compensation value is 127
15	Reserved	Must be kept at reset value.
14	PXE	Protocol exception event 0: No protocol exception event occurred since last read access 1: Protocol exception event occurred **Note:** Reading CAN_M_CAN_PSR register will reset PXE.

13	RFDF	Received a CAN FD message This bit is set independent of acceptance filtering. 0: Since this bit was reset by the CPU, no CAN FD message has been received 1: Message in CAN FD format with FDF flag set has been received **Note:** Reading CAN_M_CAN_PSR register will reset RFDF.
12	RBRS	BRS flag of last received CAN FD message This bit is set together with RFDF, independent of acceptance filtering. 0: Last received CAN FD message did not have its BRS flag set 1: Last received CAN FD message had its BRS flag set **Note:** Reading CAN_M_CAN_PSR register will reset RBRS.
11	RESI	ESI flag of last received CAN FD message This bit is set together with RFDF, independent of acceptance filtering. 0: Last received CAN FD message did not have its ESI flag set. 1: Last received CAN FD message had its ESI flag set **Note:** Reading CAN_M_CAN_PSR register will reset RESI.
10:8	DLEC[2:0]	Data phase last error code Type of last error that occurred in the data phase of a CAN FD format frame with its BRS flag set. Coding is the same as for LEC. This field is cleared to zero when a CAN FD format frame with its BRS flag set has been transferred (reception or transmission) without error. **Note:** Reading CAN_M_CAN_PSR register will set DLEC to"111".
7	BO	Bus_Off status 0: The M_CAN is not Bus_Off 1: The M_CAN is in Bus_Off state
6	EW	Warning status 0: Both error counters are below the Error_Warning limit of 96 1: At least one of error counter has reached the Error_Warning limit of 96
5	EP	Error passive 0: The M_CAN is in the Error_Active state. It normally takes part in bus communication and sends an active error flag when an error has been detected 1: The M_CAN is in the Error_Passive state
4:3	ACT[1:0]	Activity Monitors the module's CAN communication state. 00: Synchronizing-node is synchronizing on CAN communication 01: Idle-node is neither receiver nor transmitter 10: Receiver-node is operating as receiver 11: Transmitter-node is operating as transmitter **Note:** ACT is set to "00" by a Protocol Exception Event.
2:0	LEC[2:0]	Last error code The LEC indicates the type of the last error to occur on the CAN bus. This field is

cleared to 0 when a message has been transferred (reception or transmission) without error.

000: No Error: No error occurred since LEC has been reset by successful reception or transmission.

001: Stuff Error: More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed.

010: Form Error: A fixed format part of a received frame has the wrong format.

011: AckError: The message transmitted by the M_CAN was not acknowledged by another node.

100: Bit1Error: During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (bit of logical value 1), but the monitored bus value was dominant.

101: Bit0Error: During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device wanted to send a dominant level (data or identifier bit logical value 0), but the monitored bus value was recessive. During Bus_Off recovery this status is set each time a sequence of 11 recessive bits has been monitored. This enables the CPU to monitor the proceeding of the Bus_Off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed).

110: CRCError: The CRC check sum of a received message was incorrect. The CRC of an incoming message does not match with the CRC calculated from the received data.

111: NoChange: Any read access to the Protocol Status Register re-initializes the LEC to 7. When the LEC shows the value 7, no CAN bus event was detected since the last CPU read access to the Protocol Status Register.

Note: When a frame in CAN FD format has reached the data phase with BRS flag set, the next CAN event (error or valid frame) is shown in DLEC instead of LEC. An error in a fixed stuff bit of a CAN FD CRC sequence is shown as a Form Error, not Stuff Error.

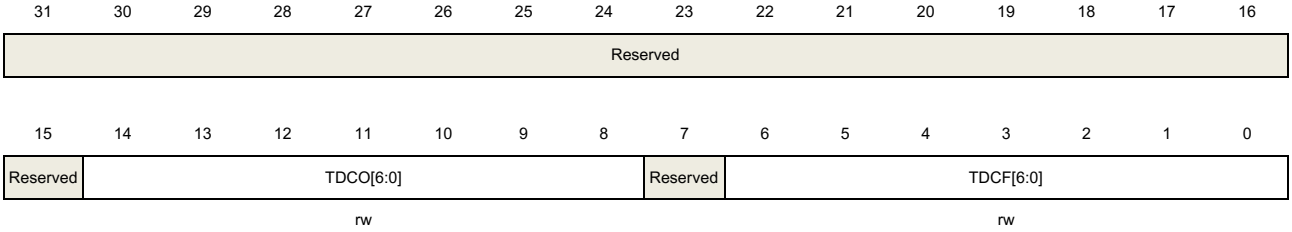
The Bus_Off recovery sequence (see ISO11898-1:2015) cannot be shortened by setting or resetting the bit INIT in CAN_M_CAN_CCCR register. If the device goes Bus_Off, it will set the bit INIT in CAN_M_CAN_CCCR register of its own accord, stopping all bus activities. Once the bit INIT in CAN_M_CAN_CCCR register has been cleared by the CPU, then the device waits for 129 occurrences of Bus Idle (129 * 11 consecutive recessive bits) before resuming normal operation. At the end of the Bus_Off recovery sequence, the Error Management Counters will be reset. During the waiting time after the resetting of the bit INIT in CAN_M_CAN_CCCR register, each time a sequence of 11 recessive bits has been monitored, a Bit0Error code is written to LEC, enabling the CPU to readily check up whether the CAN bus is stuck at dominant or continuously disturbed and to monitor the Bus_Off recovery sequence. the bits REC in CAN_M_CAN_ECR register is used to count these sequences.

Reading CAN_M_CAN_PSR register will set LEC to "111".

Transmitter delay compensation register (CAN_M_CAN_TDCR)

Address offset: 0x48
Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.



Bits	Fields	Descriptions
31:15	Reserved	Must be kept at reset value.
14:8	TDCO[6:0]	Transmitter delay compensation SSP offset Offset value defining the distance between the measured delay from CAN Tx to CAN Rx and the secondary sample point. 0x00: Transmitter delay compensation SSP offset is 0 0x01: Transmitter delay compensation SSP offset is 1 0x02: Transmitter delay compensation SSP offset is 2 … 0x7F: Transmitter delay compensation SSP offset is 127
7	Reserved	Must be kept at reset value.
6:0	TDCF[6:0]	Transmitter delay compensation filter window length Defines the minimum value for the SSP position, dominant edges on CAN Rx that results in an earlier SSP position are ignored for transmitter delay measurement. The feature is enabled when TDCF is configured to a value greater than TDCO. 0x00: Transmitter delay compensation filter window length is 0 0x01: Transmitter delay compensation filter window length is 1 0x02: Transmitter delay compensation filter window length is 2 … 0x7F: Transmitter delay compensation filter window length is 127.

Interrupt register (CAN_M_CAN_IR)

Address offset: 0x50
Reset value: 0x0000 0000

The flags are set when one of the listed conditions is detected (edge-sensitive). The flags remain set until the Host clears them. A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect. A hard reset will clear the register. The configuration of CAN_M_CAN_IE register controls whether an interrupt is generated. The configuration of

CAN_M_CAN_ILS register controls on which interrupt line an interrupt is signalled.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			PED	PEA	WDI	BO	EW	EP	ELO	Reserved		DRX	TOO	MRAF	TSW
			rw	rw	rw	rw	rw	rw	rw			rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEFL	TEFF	TEFW	TEFN	TFE	TCF	TC	HPM	RF1L	RF1F	RF1W	RF1N	RF0L	RF0F	RF0W	RF0N
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:29	Reserved	Must be kept at reset value.
28	PED	Protocol error in data phase (data bit time is used) 0: No protocol error in data phase 1: Protocol error in data phase detected (bits DLEC in CAN_M_CAN_PSR register != 0,7)
27	PEA	Protocol error in arbitration phase (nominal bit time is used) 0: No protocol error in arbitration phase 1: Protocol error in arbitration phase detected (bits LEC in CAN_M_CAN_PSR register != 0,7)
26	WDI	Watchdog interrupt 0: No message RAM watchdog event occurred 1: Message RAM Watchdog event due to missing READY
25	BO	Bus_Off status 0: Bus_Off status unchanged 1: Bus_Off status changed
24	EW	Warning status 0: Error_Warning status unchanged 1: Error_Warning status changed
23	EP	Error passive 0: Error_Passive status unchanged 1: Error_Passive status changed
22	ELO	Error logging overflow 0: CAN error logging counter did not overflow 1: Overflow of CAN error logging counter occurred
21:20	Reserved	Must be kept at reset value.
19	DRX	Message stored to dedicated Rx Buffer The flag is set whenever a received message has been stored into a dedicated Rx Buffer. 0: No Rx Buffer updated

		1: At least one received message stored into a Rx Buffer
18	TOO	Timeout occurred 0: No timeout 1: Timeout reached
17	MRAF	Message RAM access failure The flag is set, when the Rx Handler has not completed acceptance filtering or storage of an accepted message until the arbitration field of the following message has been received. In this case acceptance filtering or message storage is aborted and the Rx Handler starts processing of the following message. On the other hand, the flag is set, when the Rx Handler was not able to write a message to the Message RAM. In this case message storage is aborted. In both cases the FIFO put index is not updated resp. the New Data flag for a dedicated Rx Buffer is not set, a partly stored message is overwritten when the next message is stored to this location. The flag is also set when the Tx Handler was not able to read a message from the Message RAM in time. In this case message transmission is aborted. In case of a Tx Handler access failure the M_CAN is switched into Restricted Operation Mode (see section ***Restricted operation mode***). To leave Restricted Operation Mode, the Host CPU has to reset CAN_M_CAN_CCCR.ASM. 0: No Message RAM access failure occurred 1: Message RAM access failure occurred
16	TSW	Timestamp wraparound 0: No timestamp counter wrap-around 1: Timestamp counter wrapped around
15	TEFL	Tx Event FIFO event lost 0: No Tx Event FIFO element lost 1: Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero
14	TEFF	Tx Event FIFO full 0: Tx Event FIFO not ful 1: Tx Event FIFO full
13	TEFW	Tx Event FIFO watermark reached 0: Tx Event FIFO fill level below watermark 1: Tx Event FIFO fill level reached watermark
12	TEFN	Tx Event FIFO new entry 0: Tx Event FIFO unchanged 1: Tx Handler wrote Tx Event FIFO element
11	TFE	Tx FIFO empty 0: Tx FIFO non-empty 1: Tx FIFO empty

10	TCF	Transmission cancellation finished 0: No transmission cancellation finished 1: Transmission cancellation finished
9	TC	Transmission completed 0: No transmission completed 1: Transmission completed
8	HPM	High priority message 0: No high priority message received 1: High priority message received
7	RF1L	Rx FIFO 1 message lost 0: No Rx FIFO 1 message lost 1: Rx FIFO 1 message lost, also set after write attempt to Rx FIFO 1 of size zero
6	RF1F	Rx FIFO 1 full 0: Rx FIFO 1 not full 1: Rx FIFO 1 full
5	RF1W	Rx FIFO 1 watermark reached 0: Rx FIFO 1 fill level below watermark 1: Rx FIFO 1 fill level reached watermark
4	RF1N	Rx FIFO 1 new message 0: No new message written to Rx FIFO 1 1: New message written to Rx FIFO 1
3	RF0L	Rx FIFO 0 message lost 0: No Rx FIFO 0 message lost 1: Rx FIFO 0 message lost, also set after write attempt to Rx FIFO 0 of size zero
2	RF0F	Rx FIFO 0 full 0: Rx FIFO 0 not full 1: Rx FIFO 0 full
1	RF0W	Rx FIFO 0 watermark reached 0: Rx FIFO 0 fill level below watermark 1: Rx FIFO 0 fill level reached watermark
0	RF0N	Rx FIFO 0 new message 0: No new message written to Rx FIFO 0 1: New message written to Rx FIFO 0

Interrupt enable (CAN_M_CAN_IE)

Address offset: 0x54

Reset value: 0x0000 0000

The settings in the Interrupt Enable register determine which status changes in the Interrupt

register are signaled on an interrupt line.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			PEDE	PEAE	WDIE	BOE	EWE	EPE	ELOE	Reserved		DRXE	TOOE	MRAFE	TSWE
			rw	rw	rw	rw	rw	rw	rw	rw			rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEFLE	TEFFE	TEFWE	TEFNE	TFEE	TCFE	TCE	HPME	RF1LE	RF1FE	RF1WE	RF1NE	RF0LE	RF0FE	RF0WE	RF0NE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:29	Reserved	Must be kept at reset value.
28	PEDE	Protocol error in data phase enable 0: Interrupt disabled 1: Interrupt enabled
27	PEAE	Protocol error in arbitration phase enable 0: Interrupt disabled 1: Interrupt enabled
26	WDIE	Watchdog interrupt enable 0: Interrupt disabled 1: Interrupt enabled
25	BOE	Bus_Off status interrupt enable 0: Interrupt disabled 1: Interrupt enabled
24	EWE	Warning status interrupt enable 0: Interrupt disabled 1: Interrupt enabled
23	EPE	Error passive interrupt enable 0: Interrupt disabled 1: Interrupt enabled
22	ELOE	Error logging overflow interrupt enable 0: Interrupt disabled 1: Interrupt enabled
21:20	Reserved	Must be kept at reset value.
19	DRXE	Message stored to dedicated Rx Buffer interrupt enable 0: Interrupt disabled 1: Interrupt enabled
18	TOOE	Timeout occurred interrupt enable 0: Interrupt disabled

		1: Interrupt enabled
17	MRAFE	Message RAM access failure interrupt enable 0: Interrupt disabled 1: Interrupt enabled
16	TSWE	Timestamp wraparound interrupt enable 0: Interrupt disabled 1: Interrupt enabled
15	TEFLE	Tx Event FIFO event lost interrupt enable 0: Interrupt disabled 1: Interrupt enabled
14	TEFFE	Tx Event FIFO full interrupt enable 0: Interrupt disabled 1: Interrupt enabled
13	TEFWE	Tx Event FIFO watermark reached interrupt enable 0: Interrupt disabled 1: Interrupt enabled
12	TEFNE	Tx Event FIFO new entry interrupt enable 0: Interrupt disabled 1: Interrupt enabled
11	TFEE	Tx FIFO empty interrupt enable 0: Interrupt disabled 1: Interrupt enabled
10	TCFE	Transmission cancellation finished interrupt enable 0: Interrupt disabled 1: Interrupt enabled
9	TCE	Transmission completed interrupt enable 0: Interrupt disabled 1: Interrupt enabled
8	HPME	High priority message interrupt enable 0: Interrupt disabled 1: Interrupt enabled
7	RF1LE	Rx FIFO 1 message lost interrupt enable 0: Interrupt disabled 1: Interrupt enabled
6	RF1FE	Rx FIFO 1 full interrupt enable 0: Interrupt disabled 1: Interrupt enabled
5	RF1WE	Rx FIFO 1 watermark reached interrupt enable

		0: Interrupt disabled 1: Interrupt enabled
4	RF1NE	Rx FIFO 1 new message interrupt enable 0: Interrupt disabled 1: Interrupt enabled
3	RF0LE	Rx FIFO 0 message lost interrupt enable 0: Interrupt disabled 1: Interrupt enabled
2	RF0FE	Rx FIFO 0 full interrupt enable 0: Interrupt disabled 1: Interrupt enabled
1	RF0WE	Rx FIFO 0 Watermark Reached Interrupt Enable 0: Interrupt disabled 1: Interrupt enabled
0	RF0NE	Rx FIFO 0 New Message Interrupt Enable 0: Interrupt disabled 1: Interrupt enabled

Interrupt line select register (CAN_M_CAN_ILS)

Address offset: 0x58

Reset value: 0x0000 0000

The Interrupt Line Select register assigns an interrupt generated by a specific interrupt flag from the Interrupt Register to one of the two module interrupt lines. For interrupt generation the respective interrupt line has to be enabled via the bit EINT0 and EINT1 in CAN_M_CAN_ILE register.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			PEDL	PEAL	WDIL	BOL	EWL	EPL	ELOL	Reserved		DRXL	TOOL	MRAFL	TSWL
			rw	rw	rw	rw	rw	rw	rw			rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEFLL	TEFFL	TEFWL	TEFNL	TFEL	TCFL	TCL	HPML	RF1LL	RF1FL	RF1WL	RF1NL	RF0LL	RF0FL	RF0WL	RF0NL
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:29	Reserved	Must be kept at reset value.
28	PEDL	Protocol error in data phase line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
27	PEAL	Protocol error in arbitration phase line

		0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
26	WDIL	Watchdog interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
25	BOL	Bus_Off status interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
24	EWL	Warning status interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
23	EPL	Error passive interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
22	ELOL	Error logging overflow interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
21:20	Reserved	Must be kept at reset value.
19	DRXL	Message stored to dedicated Rx Buffer interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
18	TOOL	Timeout occurred interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
17	MRAFL	Message RAM access failure interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
16	TSWL	Timestamp wraparound interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
15	TEFLL	Tx Event FIFO event lost interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
14	TEFFL	Tx Event FIFO full interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
13	TEFWL	Tx Event FIFO watermark reached interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0

		1: Interrupt assigned to interrupt line M_CAN INT1
12	TEFNL	Tx Event FIFO new entry interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
11	TFEL	Tx FIFO empty interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
10	TCFL	Transmission cancellation finished interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
9	TCL	Transmission completed interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
8	HPML	High priority message interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
7	RF1LL	Rx FIFO 1 message lost interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
6	RF1FL	Rx FIFO 1 full interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
5	RF1WL	Rx FIFO 1 watermark reached interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
4	RF1NL	Rx FIFO 1 new message interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
3	RF0LL	Rx FIFO 0 message lost interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
2	RF0FL	Rx FIFO 0 full interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
1	RF0WL	Rx FIFO 0 watermark reached interrupt line 0: Interrupt assigned to interrupt line M_CAN INT0 1: Interrupt assigned to interrupt line M_CAN INT1
0	RF0NL	Rx FIFO 0 new message interrupt line

0: Interrupt assigned to interrupt line M_CAN INT0

1: Interrupt assigned to interrupt line M_CAN INT1

Interrupt line enable register (CAN_M_CAN_ILE)

Address offset: 0x5C

Reset value: 0x0000 0000

Each of the two interrupt lines to the CPU can be enabled / disabled separately by programming bits EINT0 and EINT1.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														EINT1	EINT0
														rw	rw

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	EINT1	Enable Interrupt Line 1 0: Interrupt line 1 disabled 1: Interrupt line 1 enabled
0	EINT0	Enable Interrupt Line 0 0: Interrupt line 0 disabled 1: Interrupt line 0 enabled

Global filter configuration (CAN_M_CAN_GFC)

Address offset: 0x80

Reset value: 0x0000 0000

Global settings for Message ID filtering. The Global Filter Configuration controls the filter path for standard and extended messages as described in ***Figure 22-7. Standard message ID filter path*** and ***Figure 22-8. Extended message ID filter path***.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										ANFS[1:0]		ANFE[1:0]		RRFS	RRFE
										rw		rw		rw	rw

Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5:4	ANFS[1:0]	Accept non-matching frames standard Defines how received messages with 11-bit IDs that do not match any element of the filter list are treated. 00: Accept in Rx FIFO 0 01: Accept in Rx FIFO 1 10: Reject 11: Reject
3:2	ANFE[1:0]	Accept non-matching frames extended Defines how received messages with 29-bit IDs that do not match any element of the filter list are treated. 00: Accept in Rx FIFO 0 01: Accept in Rx FIFO 1 10: Reject 11: Reject
1	RRFS	Reject remote frames standard 0: Filter remote frames with 11-bit standard IDs 1: Reject all remote frames with 11-bit standard IDs
0	RRFE	Reject remote frames extended 0: Filter remote frames with 29-bit extended IDs 1: Reject all remote frames with 29-bit extended IDs

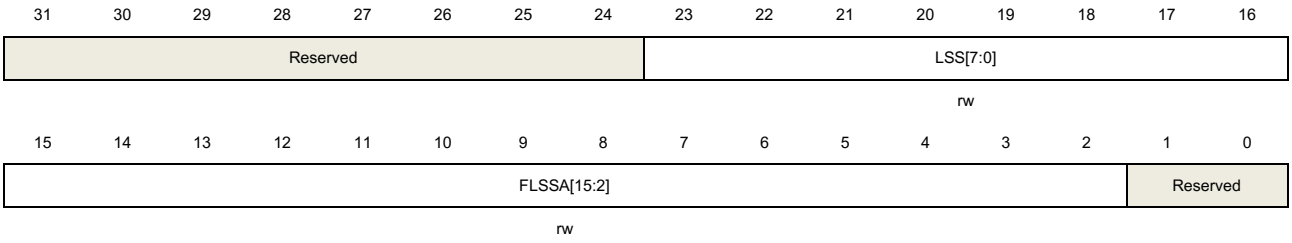
Standard ID filter configuration register (CAN_M_CAN_SIDFC)

Address offset: 0x84

Reset value: 0x0000 0000

Settings for 11-bit standard Message ID filtering. The Standard ID Filter Configuration controls the filter path for standard messages as described in ***Figure 22-7. Standard message ID filter path***.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.



Bits	Fields	Descriptions
31:27	Reserved	Must be kept at reset value.

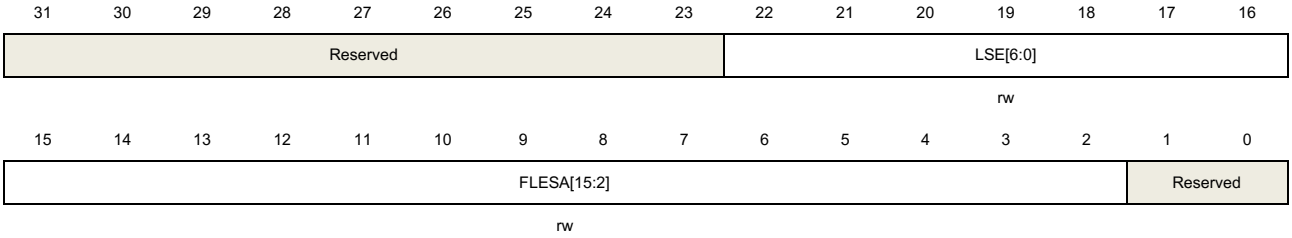
26	LSS[7:0]	List size standard 0x00: No standard Message ID filter; 0x01 - 0x80: Number of standard Message ID filter elements; 0x81 - 0xFF: Values greater than 128 are interpreted as 128.
25	FLSSA[15:2]	Filter list standard start address Start address of standard Message ID filter list (32-bit word address, refer to ***Figure 22-13. Message RAM configuration***).
1:0	Reserved	Must be kept at reset value.

Extended ID filter configuration register (CAN_M_CAN_XIDFC)

Address offset: 0x88
Reset value: 0x0000 0000

Settings for 29-bit extended Message ID filtering. The Extended ID Filter Configuration controls the filter path for standard messages as described in ***Figure 22-8. Extended message ID filter path***.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.



Bits	Fields	Descriptions
31:23	Reserved	Must be kept at reset value.
26	LSE[6:0]	List size extended 0x00: No extended Message ID filter; 0x01 - 0x40: Number of extended Message ID filter elements; 0x41 - 0x3F: Values greater than 64 are interpreted as 64.
25	FLESA[15:2]	Filter list extended start address Start address of extended Message ID filter list (32-bit word address, refer to ***Figure 22-13. Message RAM configuration***).
1:0	Reserved	Must be kept at reset value.

Extended ID and mask register (CAN_M_CAN_XIDAM)

Address offset: 0x90
Reset value: 0x1FFF FFFF

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). This register

can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			EIDM[28:16]												
			rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EIDM[15:0]															
rw															

Bits	Fields	Descriptions
31:29	Reserved	Must be kept at reset value.
28:0	EIDM[28:0]	Extended ID Mask For acceptance filtering of extended frames the Extended ID AND Mask is ANDed with the Message ID of a received frame. Intended for masking of 29-bit IDs in SAE J1939. With the reset value of all bits set to one the mask is not active.

High priority message status (CAN_M_CAN_HPMS)

Address offset: 0x94

Reset value: 0x0000 0000

This register is updated every time a Message ID filter element configured to generate a priority event matches. This can be used to monitor the status of incoming high priority messages and to enable fast access to these messages.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLST	FIDX[6:0]							MSI[1:0]		BIDX[5:0]					
r	r							r		r					

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15	FLST	Filter list Indicates the filter list of the matching filter element. 0: Standard Filter List 1: Extended Filter List
14:8	FIDX[6:0]	Filter index Index of matching filter element. Range is 0 to CAN_M_CAN_SIDFC.LSS - 1 resp. CAN_M_CAN_XIDFC.LSE - 1.
7:6	MSI[1:0]	Message storage indicator 00: No FIFO selected

01: FIFO message lost
10: Message stored in FIFO 0
11: Message stored in FIFO 1

5:0 BIDX[5:0] Buffer index
Index of Rx FIFO element to which the message was stored. Only valid when MSI[1] = 1.

New data 1 register (CAN_M_CAN_NDAT1)

Address offset: 0x98
Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND31	ND30	ND29	ND28	ND27	ND26	ND25	ND24	ND23	ND22	ND21	ND20	ND19	ND18	ND17	ND16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ND15	ND14	ND13	ND12	ND11	ND10	ND9	ND8	ND7	ND6	ND5	ND4	ND3	ND2	ND1	ND0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:0	ND[31:0]	New data The register holds the New Data flags of Rx Buffers 0 to 31. The flags are set when the respective Rx Buffer has been updated from a received frame. The flags remain set until the Host clears them. A flag is cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect. A hard reset will clear the register. 0: Rx Buffer not updated 1: Rx Buffer updated from new message

New data 2 register (CAN_M_CAN_NDAT2)

Address offset: 0x9C
Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND63	ND62	ND61	ND60	ND59	ND58	ND57	ND56	ND55	ND54	ND53	ND52	ND51	ND50	ND49	ND48
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ND47	ND46	ND45	ND44	ND43	ND42	ND41	ND40	ND39	ND38	ND37	ND36	ND35	ND34	ND33	ND32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:0	ND[63:32]	New data The register holds the New Data flags of Rx Buffers 32 to 63. The flags are set

when the respective Rx Buffer has been updated from a received frame. The flags remain set until the Host clears them. A flag is cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect. A hard reset will clear the register.

0: Rx Buffer not updated

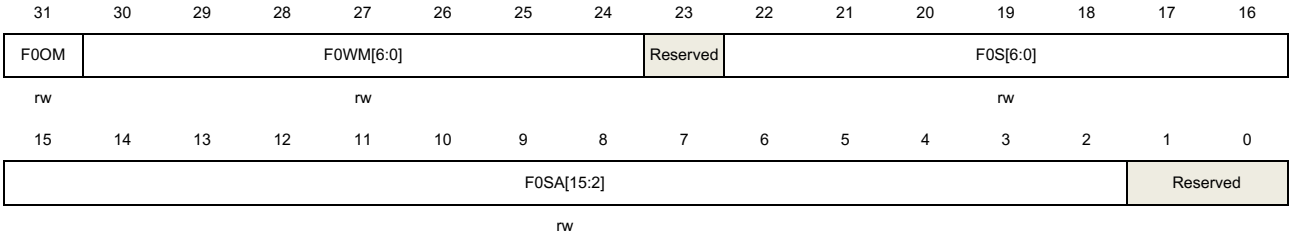
1: Rx Buffer updated from new message

Rx FIFO 0 Configuration (CAN_M_CAN_RXF0C)

Address offset: 0xA0

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.



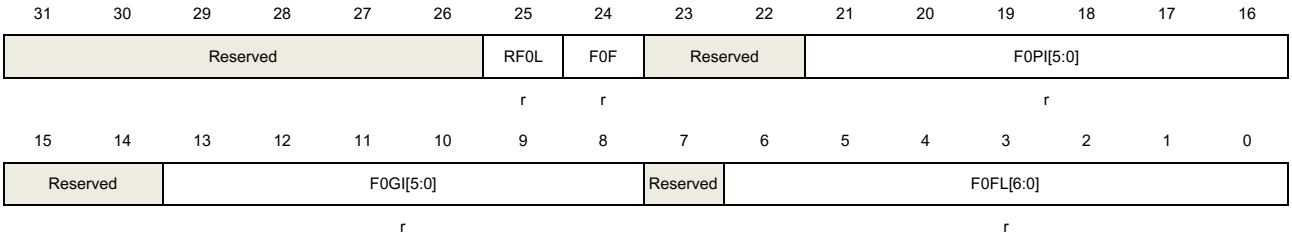
Bits	Fields	Descriptions
31	F0OM	FIFO 0 operation mode FIFO 0 can be operated in blocking or in overwrite mode (see section ***Rx FIFOs***). 0: FIFO 0 blocking mode 1: FIFO 0 overwrite mode
30:24	F0WM[6:0]	Rx FIFO 0 watermark 0x00: Watermark interrupt disabled 0x01 - 0x40: Level for Rx FIFO 0 watermark interrupt (the bit RF0W in CAN_M_CAN_IR register) 0x41 - 0x7F: Watermark interrupt disabled
23	Reserved	Must be kept at reset value.
22:16	F0S[6:0]	Rx FIFO 0 size 0x00: No Rx FIFO 0 0x01 - 0x40: Number of Rx FIFO 0 elements 0x41 - 0x7F: Values greater than 64 are interpreted as 64 The Rx FIFO 0 elements are indexed from 0 to F0S - 1.
15:2	F0SA[15:2]	Rx FIFO 0 start address Start address of Rx FIFO 0 in Message RAM (32-bit word address, see ***Figure 22-13. Message RAM configuration***).
1:0	Reserved	Must be kept at reset value.

Rx FIFO 0 status register (CAN_M_CAN_RXF0S)

Address offset: 0xA4

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).



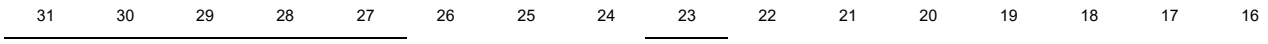
Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25	RF0L	Rx FIFO 0 message lost This bit is a copy of interrupt flag RF0L in CAN_M_CAN_IR register. When the bit RF0L in CAN_M_CAN_IR register is reset, this bit is also reset. 0: No Rx FIFO 0 message lost 1: Rx FIFO 0 message lost, also set after write attempt to Rx FIFO 0 of size zero **Note:** Overwriting the oldest message when the bit F0OM in CAN_M_CAN_RXF0C register = '1' will not set this flag.
24	F0F	Rx FIFO 0 full 0: Rx FIFO 0 not full 1: Rx FIFO 0 full
23:22	Reserved	Must be kept at reset value.
21:16	F0PI[5:0]	Rx FIFO 0 put index Rx FIFO 0 write index pointer, range 0 to 63.
15:14	Reserved	Must be kept at reset value.
13:8	F0GI[5:0]	Rx FIFO 0 get index Rx FIFO 0 read index pointer, range 0 to 63.
7	Reserved	Must be kept at reset value.
6:0	F0FL[6:0]	Rx FIFO 0 fill level Number of elements stored in Rx FIFO 0, range 0 to 64.

Rx FIFO 0 acknowledge register (CAN_M_CAN_RXF0A)

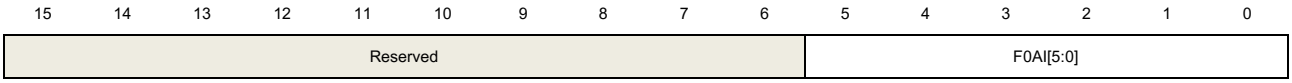
Address offset: 0xA8

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).





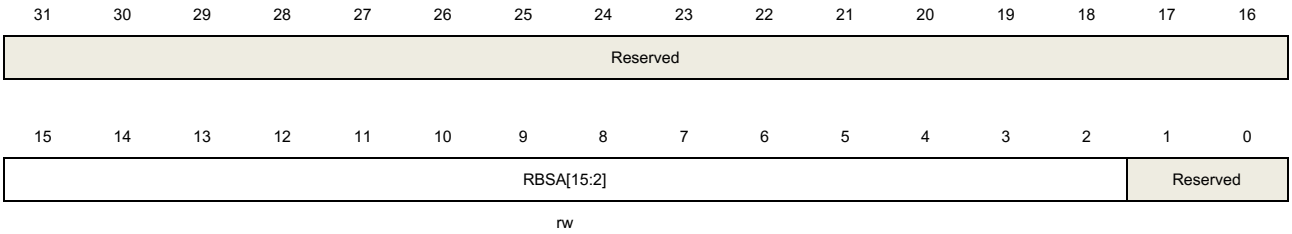


Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5:0	F0AI[5:0]	Rx FIFO 0 acknowledge index After the Host has read a message or a sequence of messages from Rx FIFO 0 it has to write the buffer index of the last element read from Rx FIFO 0 to F0AI. This sets the Rx FIFO 0 Get Index (F0GI in CAN_M_CAN_RXF0S register) to F0AI + 1 and update the FIFO 0 Fill Level (F0FL in CAN_M_CAN_RXF0S register).

Rx buffer configuration (CAN_M_CAN_RXBC)

Address offset: 0xAC
Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.

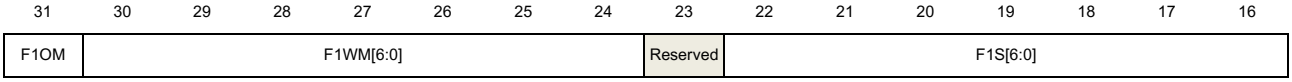


Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:2	RBSA	Rx Buffer start address Configures the start address of the Rx Buffers section in the Message RAM (32-bit word address).
1:0	Reserved	Must be kept at reset value.

Rx FIFO 1 configuration (CAN_M_CAN_RXF1C)

Address offset: 0xB0
Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.



rw				rw								rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F1SA[15:2]														Reserved	
rw															

Bits	Fields	Descriptions
31	F1OM	FIFO 1 operation mode FIFO 1 can be operated in blocking or in overwrite mode (see section ***Rx FIFOs***). 0: FIFO 1 blocking mode 1: FIFO 1 overwrite mode
30:24	F1WM[6:0]	Rx FIFO 1 watermark 0x00: Watermark interrupt disabled 0x01 - 0x40: Level for Rx FIFO 1 watermark interrupt (the bit RF1W in CAN_M_CAN_IR register) 0x41 - 0x7F: Watermark interrupt disabled
23	Reserved	Must be kept at reset value.
22:16	F1S[6:0]	Rx FIFO 1 size 0x00: No Rx FIFO 1 0x01 - 0x40: Number of Rx FIFO 1 elements 0x41 - 0x7F: Values greater than 64 are interpreted as 64 The Rx FIFO 0 elements are indexed from 0 to F1S - 1.
15:2	F1SA[15:2]	Rx FIFO 1 start address Start address of Rx FIFO 1 in Message RAM (32-bit word address, ***Figure 22-13. Message RAM configuration***).
1:0	Reserved	Must be kept at reset value.

Rx FIFO 1 status register (CAN_M_CAN_RXF1S)

Address offset: 0xB4

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						RF1L	F1F	Reserved		F1PI[5:0]					
						r	r			r					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		F1GI[5:0]						Reserved	F1FL[6:0]						
		r							r						

Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25	RF1L	Rx FIFO 1 message lost

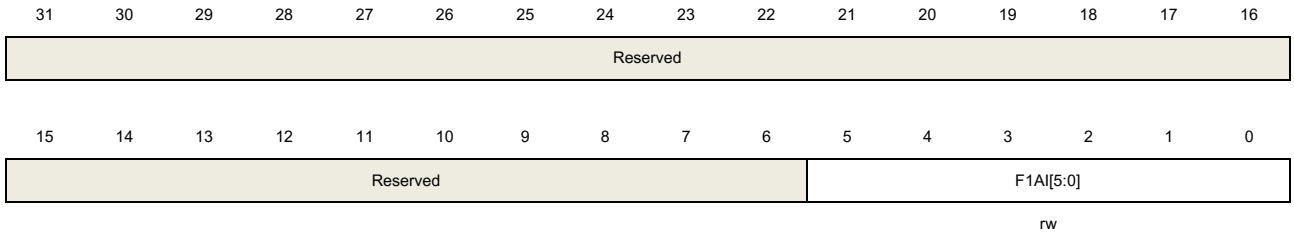
		This bit is a copy of interrupt flag RF1L in CAN_M_CAN_IR register. When the bit RF1L in CAN_M_CAN_IR register is reset, this bit is also reset. 0: No Rx FIFO 1 message lost 1: Rx FIFO 1 message lost, also set after write attempt to Rx FIFO 1 of size zero **Note:** Overwriting the oldest message when the bit F1OM in CAN_M_CAN_RXF1C register = '1' will not set this flag.
24	F1F	Rx FIFO 1 full 0: Rx FIFO 1 not full 1: Rx FIFO 1 full
23:22	Reserved	Must be kept at reset value.
21:16	F1PI[5:0]	Rx FIFO 1 put index Rx FIFO 1 write index pointer, range 0 to 63.
15:14	Reserved	Must be kept at reset value.
13:8	F1GI[5:0]	Rx FIFO 1 get index Rx FIFO 1 read index pointer, range 0 to 63.
7	Reserved	Must be kept at reset value.
6:0	F1FL[6:0]	Rx FIFO 1 fill level Number of elements stored in Rx FIFO 1, range 0 to 64.

Rx FIFO 1 acknowledge register (CAN_M_CAN_RXF1A)

Address offset: 0xB8

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).



Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5:0	F1AI[5:0]	Rx FIFO 1 acknowledge index After the Host has read a message or a sequence of messages from Rx FIFO 1 it has to write the buffer index of the last element read from Rx FIFO 1 to F1AI. This sets the Rx FIFO 1 Get Index (F1GI in CAN_M_CAN_RXF1S register) to F1AI + 1 and update the FIFO 1 Fill Level (F1FL in CAN_M_CAN_RXF1S register).

Rx buffer / FIFO element size configuration (CAN_M_CAN_RXESC)

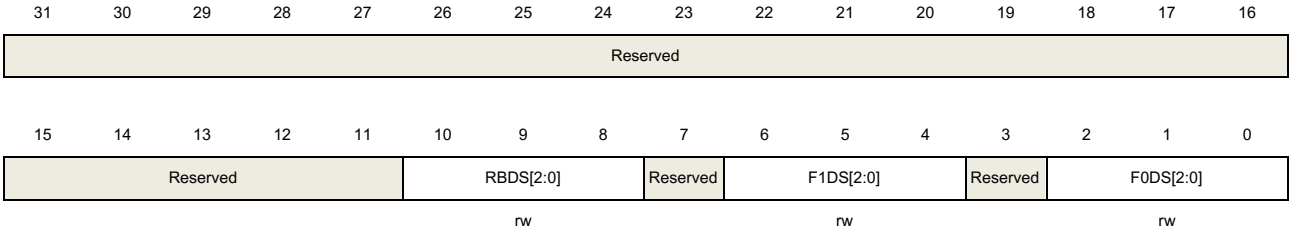
Address offset: 0xBC
Reset value: 0x0000 0000

Configures the number of data bytes belonging to an Rx Buffer / Rx FIFO element. Data field sizes > 8 bytes are intended for CAN FD operation only.

This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.

Note: In case the data field size of an accepted CAN frame exceeds the data field size configured for the matching Rx Buffer or Rx FIFO, only the number of bytes as configured by CAN_M_CAN_RXESC are stored to the Rx Buffer resp. Rx FIFO element. The rest of the frame's data field is ignored.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).



Bits	Fields	Descriptions
31:11	Reserved	Must be kept at reset value.
10:8	RBDS[2:0]	Rx buffer data field size 000: 8 byte data field 001: 12 byte data field 010: 16 byte data field 011: 20 byte data field 100: 24 byte data field 101: 32 byte data field 110: 48 byte data field 111: 64 byte data field
7	Reserved	Must be kept at reset value.
6:4	F1DS[2:0]	Rx FIFO 1 data field size 000: 8 byte data field 001: 12 byte data field 010: 16 byte data field 011: 20 byte data field 100: 24 byte data field 101: 32 byte data field 110: 48 byte data field 111: 64 byte data field

3	Reserved	Must be kept at reset value.
2:0	F0DS[2:0]	Rx FIFO 0 data field size 000: 8 byte data field 001: 12 byte data field 010: 16 byte data field 011: 20 byte data field 100: 24 byte data field 101: 32 byte data field 110: 48 byte data field 111: 64 byte data field

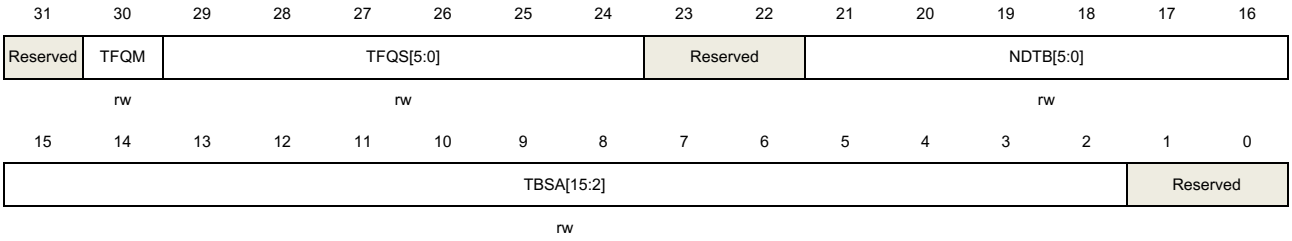
Tx buffer configuration (CAN_M_CAN_TXBC)

Address offset: 0xC0
Reset value: 0x0000 0000

This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.

Note: Be aware that the sum of TFQS and NDTB may be not greater than 32. There is no check for erroneous configurations. The Tx Buffers section in the Message RAM starts with the dedicated Tx Buffers.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).



Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30	TFQM	Tx FIFO / Queue mode 0: Tx FIFO operation 1: Tx Queue operation
29:24	TFQS[5:0]	Tx FIFO / Queue size 0x00: No Tx FIFO / Queue 0x01 - 0x20: Number of Tx Buffers used for Tx FIFO / Queue 0x21 - 0x3F: Values greater than 32 are interpreted as 32
23:22	Reserved	Must be kept at reset value.
21:16	NDTB[5:0]	Number of dedicated transmit buffers 0x00: No Dedicated Tx Buffers

		0x01 - 0x20: Number of Dedicated Tx Buffers 0x21 - 0x3F: Values greater than 32 are interpreted as 32
15:2	TBSA[15:2]	Tx Buffers start address Start address of Tx Buffers section in Message RAM (32-bit word address, refer to ***Figure 22-13. Message RAM configuration***).
1:0	Reserved	Must be kept at reset value.

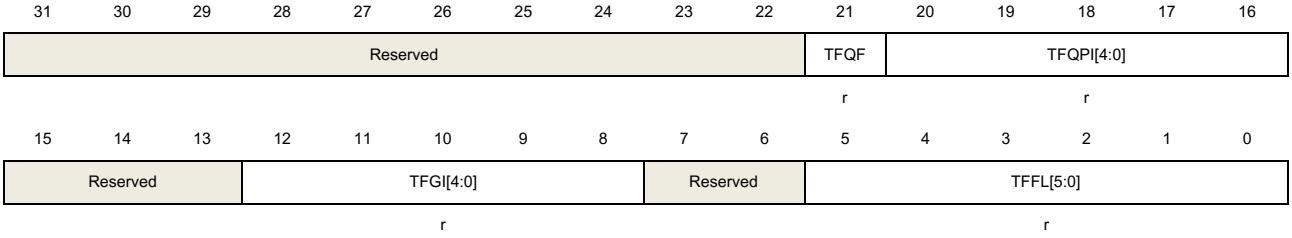
Tx FIFO / Queue status (CAN_M_CAN_TXFQS)

Address offset: 0xC4
Reset value: 0x0000 0000

The Tx FIFO / Queue status is related to the pending Tx requests listed in register CAN_M_CAN_TXBRP. Therefore the effect of Add / Cancellation requests may be delayed due to a running Tx scan (CAN_M_CAN_TXBRP not yet updated).

Note: In case of mixed configurations where dedicated Tx Buffers are combined with a Tx FIFO or a Tx Queue, the Put and Get Indices indicate the number of the Tx Buffer starting with the first dedicated Tx Buffers. Example: For a configuration of 12 dedicated Tx Buffers and a Tx FIFO of 20 Buffers a Put Index of 15 points to the fourth buffer of the Tx FIFO.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).



Bits	Fields	Descriptions
31:22	Reserved	Must be kept at reset value.
21	TFQF	Tx FIFO / Queue full 0: Tx FIFO / Queue not ful 1: Tx FIFO / Queue full
20:16	TFQPI[4:0]	Tx FIFO/Queue put index Tx FIFO/Queue write index pointer, range 0 to 31.
15:13	Reserved	Must be kept at reset value.
12:8	TFGI[4:0]	Tx FIFO get index Tx FIFO read index pointer, range 0 to 31. Read as zero when Tx Queue operation is configured (the bit TFQM in CAN_M_CAN_TXBC register = '1').
7:6	Reserved	Must be kept at reset value.
5:0	TFFL[5:0]	Tx FIFO free level

Number of consecutive free Tx FIFO elements starting from TFGI, range 0 to 32. Read as zero when Tx Queue operation is configured (the bit TFQM in CAN_M_CAN_TXBC register = '1').

Tx Buffer Element Size Configuration (CAN_M_CAN_TXESC)

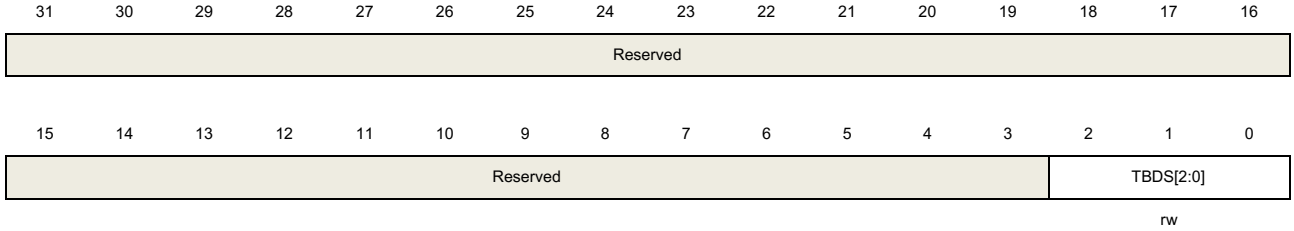
Address offset: 0xC8
Reset value: 0x0000 0000

Configures the number of data bytes belonging to a Tx Buffer element. Data field sizes > 8 bytes are intended for CAN FD operation only.

This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.

Note: In case the data length code DLC of a Tx Buffer element is configured to a value higher than the Tx Buffer data field size (TBDS), the bytes not defined by the Tx Buffer are transmitted as "0xCC" (padding bytes).

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).



Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2:0	TBDS[2:0]	Tx Buffer data field size 000: 8 byte data field 001: 12 byte data field 010: 16 byte data field 011: 20 byte data field 100: 24 byte data field 101: 32 byte data field 110: 48 byte data field 111: 64 byte data field

Tx buffer request pending (CAN_M_CAN_TXBRP)

Address offset: 0xCC
Reset value: 0x0000 0000

Note: CAN_M_CAN_TXBRP bits which are set while a Tx scan is in progress are not considered during this particular Tx scan. In case a cancellation is requested for such a Tx Buffer, this Add Request is cancelled immediately, the corresponding CAN_M_CAN_TXBRP

bit is reset.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRP31	TRP30	TRP29	TRP28	TRP27	TRP26	TRP25	TRP24	TRP23	TRP22	TRP21	TRP20	TRP19	TRP18	TRP17	TRP16
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRP15	TRP14	TRP13	TRP12	TRP11	TRP10	TRP9	TRP8	TRP7	TRP6	TRP5	TRP4	TRP3	TRP2	TRP1	TRP0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits	Fields	Descriptions
31	TRP[31:0]	Transmission request pending Each Tx Buffer has its own Transmission Request Pending bit. The bits are set via register CAN_M_CAN_TXBAR. The bits are reset after a requested transmission has completed or has been cancelled via register CAN_M_CAN_TXBCR. CAN_M_CAN_TXBRP bits are set only for those Tx Buffers configured via CAN_M_CAN_TXBC. After a CAN_M_CAN_TXBRP bit has been set, a Tx scan (see section ***Tx handling***) is started to check for the pending Tx request with the highest priority (Tx Buffer with lowest Message ID). A cancellation request resets the corresponding transmission request pending bit of register CAN_M_CAN_TXBRP. In case a transmission has already been started when a cancellation is requested, this is done at the end of the transmission, regardless whether the transmission was successful or not. The cancellation request bits are reset directly after the corresponding CAN_M_CAN_TXBRP bit has been reset. After a cancellation has been requested, a finished cancellation is signalled via CAN_M_CAN_TXBCF ■ after successful transmission together with the corresponding CAN_M_CAN_TXBTO bit. ■ when the transmission has not yet been started at the point of cancellation. ■ when the transmission has been aborted due to lost arbitration. ■ when an error occurred during frame transmission. In DAR mode all transmissions are automatically cancelled if they are not successful. The corresponding CAN_M_CAN_TXBCF bit is set for all unsuccessful transmissions. 0: No transmission request pending 1: Transmission request pending

Tx buffer add request (CAN_M_CAN_TXBAR)

Address offset: 0xD0
Reset value: 0x0000 0000

Note: If an add request is applied for a Tx Buffer with pending transmission request (corresponding CAN_M_CAN_TXBRP bit already set), this add request is ignored.

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AR31	AR30	AR29	AR28	AR27	AR26	AR25	AR24	AR23	AR22	AR21	AR20	AR19	AR18	AR17	AR16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AR15	AR14	AR13	AR12	AR11	AR10	AR9	AR8	AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:0	AR[31:0]	Add request Each Tx Buffer has its own Add Request bit. Writing a '1' will set the corresponding Add Request bit; writing a '0' has no impact. This enables the Host to set transmission requests for multiple Tx Buffers with one write to CAN_M_CAN_TXBAR. CAN_M_CAN_TXBAR bits are set only for those Tx Buffers configured via CAN_M_CAN_TXBC. When no Tx scan is running, the bits are reset immediately, else the bits remain set until the Tx scan process has completed. 0: No transmission request added 1: Transmission requested added

Tx buffer cancellation request (CAN_M_CAN_TXBCR)

Address offset: 0xD4

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CR31	CR30	CR29	CR28	CR27	CR26	CR25	CR24	CR23	CR22	CR21	CR20	CR19	CR18	CR17	CR16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CR15	CR14	CR13	CR12	CR11	CR10	CR9	CR8	CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:0	CR[31:0]	Cancellation request Each Tx Buffer has its own Cancellation Request bit. Writing a '1' will set the corresponding Cancellation Request bit; writing a '0' has no impact. This enables the Host to set cancellation requests for multiple Tx Buffers with one write to CAN_M_CAN_TXBCR. CAN_M_CAN_TXBCR bits are set only for those Tx Buffers configured via CAN_M_CAN_TXBC. The bits remain set until the corresponding bit of CAN_M_CAN_TXBRP is reset. 0: No cancellation pending 1: Cancellation pending

Tx buffer transmission occurred (CAN_M_CAN_TXBTO)

Address offset: 0xD8

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TO31	TO30	TO29	TO28	TO27	TO26	TO25	TO24	TO23	TO22	TO21	TO20	TO19	TO18	TO17	TO16
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TO15	TO14	TO13	TO12	TO11	TO10	TO9	TO8	TO7	TO6	TO5	TO4	TO3	TO2	TO1	TO0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits	Fields	Descriptions
31:0	TO[31:0]	Transmission occurred Each Tx Buffer has its own Transmission Occurred bit. The bits are set when the corresponding CAN_M_CAN_TXBRP bit is cleared after a successful transmission. The bits are reset when a new transmission is requested by writing a '1' to the corresponding bit of register CAN_M_CAN_TXBAR. 0: No transmission occurred 1: Transmission occurred

Tx buffer cancellation finished (CAN_M_CAN_TXBCF)

Address offset: 0xDC

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits	Fields	Descriptions
31:0	CF[31:0]	Cancellation finished Each Tx Buffer has its own Cancellation Finished bit. The bits are set when the corresponding CAN_M_CAN_TXBRP bit is cleared after a cancellation was requested via CAN_M_CAN_TXBCR. In case the corresponding CAN_M_CAN_TXBRP bit was not set at the point of cancellation, CF is set immediately. The bits are reset when a new transmission is requested by writing a '1' to the corresponding bit of register CAN_M_CAN_TXBAR. 0: No transmit buffer cancellation 1: Transmit buffer cancellation finished

Tx buffer transmission interrupt enable (CAN_M_CAN_TXBTIE)

Address offset: 0xE0

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIE31	TIE30	TIE29	TIE28	TIE27	TIE26	TIE25	TIE24	TIE23	TIE22	TIE21	TIE20	TIE19	TIE18	TIE17	TIE16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIE15	TIE14	TIE13	TIE12	TIE11	TIE10	TIE9	TIE8	TIE7	TIE6	TIE5	TIE4	TIE3	TIE2	TIE1	TIE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:0	TIE[31:0]	Transmission interrupt enable Each Tx Buffer has its own Transmission Interrupt Enable bit. 0: Transmission interrupt disabled 1: Transmission interrupt enable

Tx buffer cancellation finished interrupt enable (CAN_M_CAN_TXBCIE)

Address offset: 0xE4

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CFIE31	CFIE30	CFIE29	CFIE28	CFIE27	CFIE26	CFIE25	CFIE24	CFIE23	CFIE22	CFIE21	CFIE20	CFIE19	CFIE18	CFIE17	CFIE16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CFIE15	CFIE14	CFIE13	CFIE12	CFIE11	CFIE10	CFIE9	CFIE8	CFIE7	CFIE6	CFIE5	CFIE4	CFIE3	CFIE2	CFIE1	CFIE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:0	CFIE[31:0]	Cancellation finished interrupt enable Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0: Cancellation finished interrupt disabled 1: Cancellation finished interrupt enabled

Tx event FIFO configuration (CAN_M_CAN_TXEFC)

Address offset: 0xF0

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit). This register can be modified only when CCE and INIT bit in CAN_M_CAN_CCCR register are set.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved	EFWM[5:0]	Reserved	EFS[5:0]
	rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EFSA[15:2]														Reserved	
rw															

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29:24	EFWM[5:0]	Event FIFO watermark 0x00: Watermark interrupt disabled 0x01 - 0x20: Level for Tx Event FIFO watermark interrupt (the bit TEFW in CAN_M_CAN_IR register) 0x21 - 0x3F: Watermark interrupt disabled
23:22	Reserved	Must be kept at reset value.
21:16	EFS[5:0]	Event FIFO size 0x00: Tx Event FIFO disabled 0x01 - 0x20: Number of Tx Event FIFO elements 0x21 - 0x3F: Values greater than 32 are interpreted as 32 The Tx Event FIFO elements are indexed from 0 to EFS - 1
15:2	EFSA[15:2]	Event FIFO start address. Start address of Tx Event FIFO in Message RAM (32-bit word address, ***Figure 22-13. Message RAM configuration***)
1:0	Reserved	Must be kept at reset value.

Tx event FIFO status (CAN_M_CAN_TXEFS)

Address offset: 0xF4

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						TEFL	EFF	Reserved			EFPI[4:0]				
						r	r				r				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			EFGI[4:0]					Reserved		EFFL[5:0]					
			r							r					

Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25	TEFL	Tx Event FIFO element lost This bit is a copy of interrupt flag TEFL in CAN_M_CAN_IR register. When the bit TEFL in CAN_M_CAN_IR register is reset, this bit is also reset.

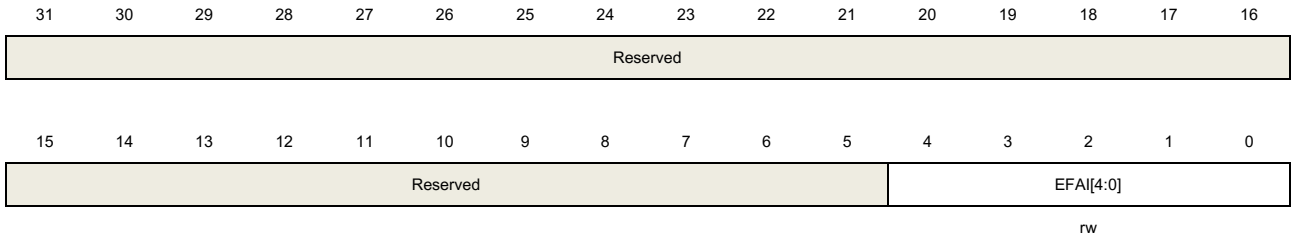
		0: No Tx Event FIFO element lost 1: Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero.
24	EFF	Event FIFO full 0: Tx Event FIFO not full 1: Tx Event FIFO full
23:21	Reserved	Must be kept at reset value.
20:16	EFPI[4:0]	Event FIFO put index Tx Event FIFO write index pointer, range 0 to 31
15:13	Reserved	Must be kept at reset value.
12:8	EFGI[4:0]	Event FIFO get index Tx Event FIFO read index pointer, range 0 to 31.
7:6	Reserved	Must be kept at reset value.
5:0	EFFL[5:0]	Event FIFO fill level Number of elements stored in Tx Event FIFO, range 0 to 32.

Tx event FIFO acknowledge register (CAN_M_CAN_TXEFA)

Address offset: 0xF8

Reset value: 0x0000 0000

This register can be accessed by byte (8-bit), half-word (16-bit) or word (32-bit).



Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
4:0	EFAI[4:0]	Event FIFO acknowledge index After the Host has read an element or a sequence of elements from the Tx Event FIFO it has to write the index of the last element read from Tx Event FIFO to EFAI. This will set the Tx Event FIFO Get Index (the EFGI in CAN_M_CAN_TXEFS register) to EFAI + 1 and update the Event FIFO Fill Level (the EFFI in CAN_M_CAN_TXEFS register).

22.7. DMA interface unit (DMU)

22.7.1. Overview

Table 22-17. IP user manual versions

IP name	Rvision
BOSCH DMU	1.0.1

The DMU is an extension-module for the M_CAN, which provides an additional interface that allows simple enqueuing and dequeuing to TX and RX FIFOs of the M_CAN by linear write or read accesses, in order to be able to offload these tasks from CPU to standard DMA controllers.

M_CAN message handling (without DMU)

The M_CAN has an associated Message RAM (MRAM) which contains the CAN messages, organized e.g. in FIFOs. The controlling of that data is done by the M_CAN in conjunction with several registers which provide status and means for handling via CPU. To interchange messages between CRAM (Central memory of the SoC, bound to DMA and CPU) and MRAM without the DMU add-on, the CPU has to process the following tasks:

- React on interrupt of M_CAN and check status registers (Alternatively poll M_CAN interrupt / status registers).
- Transfer CAN message between MRAM and CRAM.
- Acknowledge transfer at M_CAN registers.

This complex interaction cannot be handled by standard DMA controllers.

Message handling with DMU

The basic idea of the DMU is to define Virtual Buffers for CAN message transfer. The DMU redirects accesses to these Virtual Buffers dynamically to the MRAM. Redirections are controlled by the FIFO pointers of the M_CAN. The DMU supports message transfers from the CRAM to the TX FIFO/Queue and from the RX FIFOs / TX Event FIFO to the CRAM.

Table 22-18. Supported M_CAN message buffers

MRAM	Direction	Condition	DMU Request signal
TX FIFO/Queue	CRAM to MRAM	not full	dmu_txr
RX FIFO 0	MRAM to CRAM	not empty	dmu_rx0r
RX FIFO 1	MRAM to CRAM	not empty	dmu_rx1r
TX Event FIFO	MRAM to CRAM	not empty	dmu_txer

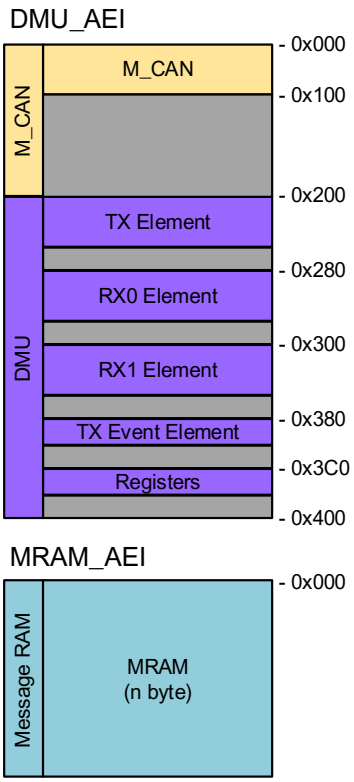
The M_CAN FIFOs have to be configured as described in chapter "M_CAN", e.g. base addresses and data field sizes. When the M_CAN is setup and running, the CAN messages have to be interchanged via dedicated DMU Virtual Buffers as shown in ***Figure 22-14. Address map***. These sections are bound to the Tx FIFO respective Rx FIFO head elements,

i.e. the elements, pointed by M_CAN PUT_INDEX respective GET_INDEX. The DMU just redirects the accesses to the MRAM.

The data structure of the FIFO elements is described in chapter "M_CAN". Each DMU Virtual Buffer is dimensioned to the maximal M_CAN element size, i.e. 72 byte for Tx and Rx FIFO elements respective 8 byte for Tx Event FIFO element. Transfer requests to the external DMA controller will be indicated by separate DMU Request signals, one for each Virtual Buffer, see ***Table 22-18. Supported M_CAN message buffers***. On active request, the DMA controller may interchange data, by linear access to the dedicated Virtual Buffer. With this linear transfer, everything internal is implicitly handled, e.g. that a Tx message will be sent by the M_CAN, or a Rx FIFO element will be acknowledged and disengaged at the M_CAN.

It is possible to use DMU only for some M_CAN FIFOs and to handle the others by software, but a mix of software and DMU for the same FIFO leads into malfunction.

Figure 22-14. Address map

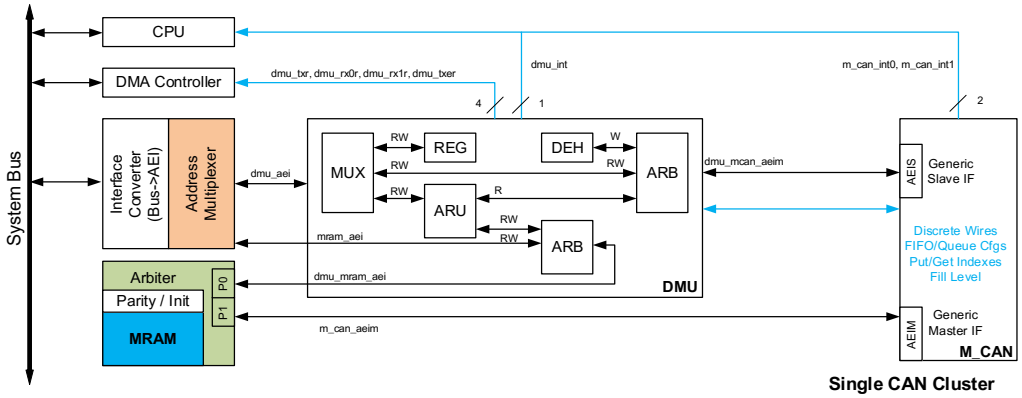


22.7.2. Function overview

Block diagram

The following figure shows the internal structure of the DMU and exemplary the integration into a SoC together with M_CAN.

Figure 22-15. DMU integration



DMU TX element (0x200 - 0x244)

This chapter describes the TX Virtual Buffer of the DMU. It allows to enqueue messages to the Tx FIFO/Queue of the M_CAN in a simplified way. Both queue types of the M_CAN are supported: The so called "Tx Queue" and the so called "Tx FIFO". The configuration of the M_CAN Tx FIFO/Queue has to be done exactly in the same way as without using DMU, since the DMU provides just another access path to the M_CAN's Tx FIFO/Queue elements and does not supersede them.

Enqueue a message

An enqueueing of a Tx message via the TX Element section of the DMU should only be started if this is requested by DMU Request signal dmu_txr.

The CAN message format of the DMU TX Element is the format described in chapter "M_CAN". To enqueue a Tx message, a strict write access sequence has to be processed. It has to start by writing the T0 word of the CAN message to the first address of the TX Virtual Buffer (0x200), which is called the start address. This triggers the DMU to expect a CAN Tx message transfer. The CAN message has to be transferred by linear addressing to ascending and consecutive addresses, word by word. The transfer completes, by writing the last element word Tn (with n belonging to {1, 2, 3, 4, 5, 6, 7, 9, 13, 17}) which is called is the Trigger Address. It is the last required write access to the TX Element Section for a particular CAN message, which triggers the scheduling of that TX message element inside the M_CAN.

The DMU derives the Trigger Address from the RTR bit in T0, the FDF bit in T1, and the Data Length Code (DLC) in T1. This means the DMU derives the Trigger Address dynamically for each message, depending on the individual payload to be transferred. In case of a CAN message with DLC = 4 the Trigger Address is T2. After completion, normally the next expected access to the DMU Element sections is a start address. The DMU TX Element section is also compatible to DMA with fixed transfer length (independent of payload), see flag CAN_DMU_DMUIR.TXEWATA.

Status flags in the Interrupt Register CAN_DMU_DMUIR provide detailed information, when one of the access rules is violated. It is highly recommended, to carefully check all

CAN_DMU_DMUIR.TXExxx status flags during runtime for unintended activations.

The following flags should never get active during normal operation:

- CAN_DMU_DMUIR.TXENSA: TX Element Not Start Address.
- CAN_DMU_DMUIR.TXEIE: TX Element Illegal Enqueueing.
- CAN_DMU_DMUIR.TXEIAS: TX Element Illegal Access Sequence.
- CAN_DMU_DMUIR.TXEIDLC: TX Element Illegal DLC.
- CAN_DMU_DMUIR.TXEIR: TX Element Illegal Read.

DMU RX element (0x280 - 0x2C4 and 0x300 - 0x344)

This chapter describes the DMU RX0 Element and DMU RX1 Element handling. The two Elements behave identically, therefore the expression DMU RX Element is used, which describes both.

The DMU RX Element allows dequeueing of messages from Rx FIFO 0,1 of the M_CAN in a simplified way. The DMU RX Element behaves similar to the previously described TX Element (see section ***DMU TX element (0x200 - 0x244)***). The main differences are the transfer direction (MRAM to CRAM) and that the transfer size is constant (according M_CAN:CAN_M_CAN_RXESC.FnDS) and does not dynamically change by DLC of a given message.

Dequeueing a message

Dequeueing of an Rx message with the DMU RX Element should only be started, if this is requested by DMU Request signals dmu_rx0r respective dmu_rx1r.

The message format of the DMU RX Element is the format, described in chapter "M_CAN" Rx Buffer and FIFO Element. To dequeue an Rx message, a strict read access sequence has to be processed. It has to start by reading the R0 word of the CAN message from the first address of the DMU RX Element (0x280 resp. 0x300), which is called the start address. This triggers the DMU to expect a CAN Rx message transfer. The CAN message has to be transferred by linear addressing to ascending and consecutive addresses, word by word. The transfer completes, by reading the Trigger Address Rn, with n belonging {3, 4, 5, 6, 7, 9, 13, 17}.

The Trigger Address is derived from the configuration of the dedicated M_CAN Rx FIFO n Element Size. E.g. if M_CAN: CAN_M_CAN_RXESC.F0DS = 0 and M_CAN: CAN_M_CAN_RXESC.F1DS = 7, the Trigger Address of DMU RX 0 Element is 0x28C (Rx FIFO element word R3) and Trigger Address of DMU RX 1 Element is 0x344 (Rx FIFO element word R17). After completion, the next expected access to the DMU Element sections is a start address.

Status flags in the Interrupt Register CAN_DMU_DMUIR provide detailed information, when one of the access rules is violated. It is highly recommended, to carefully check all CAN_DMU_DMUIR.RXnExxx status flags during runtime for unexpected activations.

The following flags should never get active during normal operation:

- CAN_DMU_DMUIR.RX0,1EIO: RX Element Illegal Overwrite.
- CAN_DMU_DMUIR.RX0,1EIW: RX Element Illegal Write.
- CAN_DMU_DMUIR.RX0,1EIAS: RX Element Illegal Access Sequence.
- CAN_DMU_DMUIR.RX0,1EID: RX Element Illegal Dequeueing.
- CAN_DMU_DMUIR.RX0,1ENSA: RX Element Not Start Address.

TX Event Element (0x380 - 0x384)

This chapter describes the TX Event Element of the DMU, which allows to dequeue Tx events from the Tx Event FIFO of the M_CAN in a simplified way.

The Tx Event FIFO of the M_CAN provides information about transmitted CAN messages (see chapter ***Modular controller area network (M_CAN)***). Like with the Rx FIFOs, data has to be transferred from MRAM to CRAM. Data handling is the same as with the DMU RX Element (see section ***DMU RX element (0x280 - 0x2C4 and 0x300 - 0x344)***). The only difference is, that the Trigger Address is 0x384 when CAN_DMU_DMUC.TTS = '0'.

Status flags of the Interrupt Register CAN_DMU_DMUIR provide detailed information when one of the access rules is violated. It is highly recommended, to carefully check all CAN_DMU_DMUIR.TXEExxx status flags during runtime for unintended activations.

The following flags should never get active during normal operation:

- CAN_DMU_DMUIR.TXEEIW: TX Event Element Illegal Write.
- CAN_DMU_DMUIR.TXEEIAS: TX Event Element Illegal Access Sequence.
- CAN_DMU_DMUIR.TXEEID: TX Event Element Illegal Dequeueing.
- CAN_DMU_DMUIR.TXEENSA: TX Event Element Not Start Address.

Exception recovery at DMU virtual buffer

If the DMU detects an illegal access to the DMU Virtual Buffers, it starts an exception recovery in two steps.

The first step is, to immediately abort an ongoing service, i.e. a partly transferred Tx messages will be discarded, i.e. not scheduled at M_CAN, to avoid corrupted messages on the CAN bus, respective a partly transferred Rx messages regarded as not transferred.

The second step is to wait for the next access to a start address of one of the DMU Virtual Buffers. Accesses to Virtual Buffer addresses except one of the four Start Addresses will be ignored and do not activate further interrupt flags, so the active status flag in the CAN_DMU_DMUIR register shows the first occurred access error.

M_CAN in CCE mode

After hardware reset the DMU is disabled. If the DMU is disabled, it waits for a transition from 1 to 0 of bit M_CAN:CAN_M_CAN_CCCR.CCE, which finishes M_CAN's CCE mode. A new activation of CCE mode will again disable the DMU.

If the DMU is disabled it behaves the following way:

- Access to the DMU Element sections (TX, RX0, RX1, TX Event) is not allowed. If accessed anyhow, flag CAN_DMU_DMUIR.IAC is set.
  Note: There is no write protection for the DMU TX Element, all other DMU Elements (RX0, RX1, TX Event) are always write protected.
- The DMU Element Handler set to reset state, e.g. a pending transaction will be aborted immediately. Therefore it is highly recommended, to shut down DMA services for the DMU before entering CCE mode.
- All DMU Service Request signals are set to zero.

Note:

- Access to DMU configuration registers is always allowed, independent of M_CAN:CAN_M_CAN_CCCR.CCE.
- When leaving M_CAN's CCE mode, DMU could request for DMA services. Even if CCE mode will be re-entered immediately, the DMU requests could already have started DMA services.

DMU service request signals

The DMU provides signals, to trigger an external DMA Controller for a service, i.e. to transfer a M_CAN FIFO element from CRAM to MRAM via the DMU TX Element, or vice versa from MRAM to CRAM via the DMU RX Elements respective DMU TX Event Element. See also section ***Message handling with DMU***.

Timing of service request signals

The DMU Service Request signals get set, when the following conditions are met:

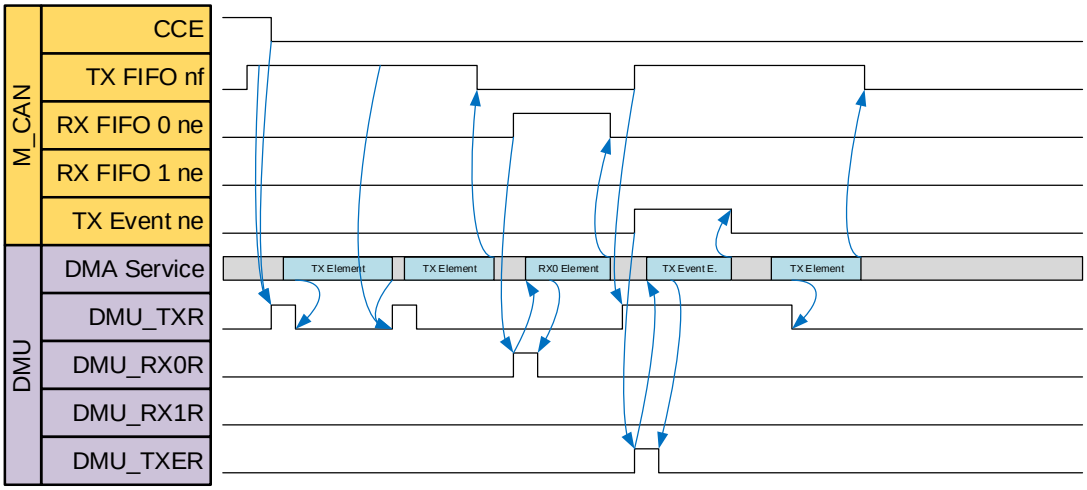
- M_CAN:CAN_M_CAN_CCCR.CCE = '0'.
- Service of Virtual Buffer is not ongoing.
- For DMU TX Virtual Buffer: M_CAN Tx FIFO/Tx Queue is not full.
- For other DMU Virtual Buffers: M_CAN Rx FIFO/Tx Event FIFO is not empty.

The external DMA Controller should start a service only if requested by DMU Service Request signals. When the DMA Controller starts the service by accessing the Start Address of a DMU Element, the Service Request signal of the addressed Element will be de-asserted.

The DMA Controller will complete the service by accessing the Trigger Address. Now the DMU starts internal follow-up interactions with M_CAN, e.g. acknowledge of the element access. On completion of these tasks, the Service Request signal of the dedicated Element will be re-enabled and is getting active, as soon as the service is needed again e.g. due to M_CAN FIFO status.

The ***Figure 22-16. DMU timing*** shows the M_CAN:CAN_M_CAN_CCCR.CCE bit, the FIFO/Queue states (not full (nf), not empty (ne)) of the M_CAN, the accesses of the external DMA Controller to the DMU (DMA Service), and the DMU Service Request signals, which trigger the external DMA services. The sequence shown is just an example to explain the sequence of events, marked by the blue arrows.

Figure 22-16. DMU timing



Unused service request signals

If an application does not use the full DMU support, e.g. because some M_CAN FIFOs are accessed directly by CPU, the unused DMU Service Request signals have to be ignored by external DMA. Disabling by DMU is not supported.

22.7.3. Register definition

The DMU is mapped into the M_CAN address space starting at address 0x200. The address 0x000 - 0x1FC are assigned to M_CAN. Register base address reference section ***CAN cluster register definition***.

Table 22-19. DMU register overview

Register Name	Symbol	Address Offset	Reset Value
DMU core release register	CAN_DMU_DMUCR	0x03C0	0x1012 0505
DMU internals register	CAN_DMU_DMUI	0x03C4	0x0007 0000
DMU queueing counter register	CAN_DMU_DMUQC	0x03C8	0x0000 0000
DMU interrupt register	CAN_DMU_DMUIR	0x03CC	0x0000 0000
DMU Interrupt Enable Register	CAN_DMU_DMUIE	0x03D0	0x0000 0000
DMU Configuration Register	CAN_DMU_DMUC	0x03D4	0x0000 0000

Table 22-20. Address map

ADDRESS	NAME	RESET	Acc
0x200 - 0x244	DMU TX Element (18)	MRAM(1)	w(2)
0x248 - 0x27C	Reserved	/	/
0x280 - 0x2C4	DMU RX0 Element (18)	MRAM(1)	r(2)
0x2C8 - 0x2FC	Reserved	/	/
0x300 - 0x344	DMU RX1 Element (18)	MRAM(1)	r(2)
0x348 - 0x37C	Reserved	/	/
0x380 - 0x384	DMU TX Event Element (2)	MRAM(1)	r(2)
0x388 - 0x3BC	Reserved	/	/
0x3C0 - 0x3D4	DMU registers	see next	see next
0x3D8 - 0x3FC	Reserved	/	/
0x400 - 0x444	Reserved	/	/
0x448 - 0x47C	Reserved	/	/
0x480 - 0x4C4	Reserved	/	/
0x4C8 - 0x4FC	Reserved	/	/
0x500 - 0x544	Reserved	/	/
0x548 - 0x57C	Reserved	/	/
0x580 - 0x588	Reserved	/	/
0x58C - 0x7FC	Reserved	/	/

Note:

1. Indirect access to MRAM by DMU. MRAM not initialized by HW reset.
2. Access controlled by DMU Element handler.

Accesses to reserved addresses are directly signaled to the Host CPU by status signals of the DMU interface.

DMU core release register (CAN_DMU_DMUCR)

Address offset: 0x03C0

Reset value: 0x1012 0505

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REL[3:0]				STEP[3:0]				SUBSTEP[3:0]				YEAR[3:0]			
r				r				r				r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MON[7:0]								DAY[7:0]							
r								r							

Bits	Fields	Descriptions
31:28	REL[3:0]	Core release (One digit, BCD-coded)
27:24	STEP[3:0]	Step of core release (One digit, BCD-coded)
23:20	SUBSTEP[3:0]	Sub-step of core release (One digit, BCD-coded)
19:16	YEAR[3:0]	Time stamp year (One digit, BCD-coded. This field is set by generic parameter on synthesis.)
15:8	MON[7:0]	Time stamp month (Two digit, BCD-coded. This field is set by generic parameter on synthesis.)
7:0	DAY[7:0]	Time stamp day (Two digit, BCD-coded. This field is set by generic parameter on synthesis.) **Note:** BCD 16#YMMDD# = 201Y-MM-DD

DMU internals register (CAN_DMU_DMUI)

Address offset: 0x03C4

Reset value: 0x0007 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXE	RX1	RX0	TX	Reserved	EHS[2:0]			Reserved							
r	r	r	r		r										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENA	Reserved		TFQPIP[4:0]					Reserved				TXER	RX1R	RX0R	TXR
r			r									r	r	r	r

Bits	Fields	Descriptions
31:28	TXE, RX1, RX0, TX	Actual DMU element service These four flags show which DMU Virtual Buffer is currently served, see ***Figure 22-14. Address map***. Only one Virtual Buffer can be served at one time. The respective bit will be set by accessing the service's Start Address. The bit will be

		reset when service finishes after entering Element Handler State wait4sa (queueing successful) or recovery (queueing failure). 0: DMU Virtual Buffer is currently not served 1: DMU Virtual Buffer is currently served
27	Reserved	Must be kept at reset value.
26:24	EHS[2:0]	Element handler state Current state of the DMU Element Handler FSM, which controls the interaction with all DMU Elements, i.e. TX, RX0, RX1, TXE. 0: wait4cce - wait for bit M_CAN:CAN_M_CAN_CCCR.CCE getting zero 1: wait4sa - wait for Start Address 2: wait4ta - wait for Trigger Address 3: transfer - wait for transfer of Element word 4: ack2mcan - acknowledge to M_CAN 5: recovery - exception recovery 6,7: unused
23:16	Reserved	Must be kept at reset value.
15	ENA	DMU is enabled This flag shows the state of the DMU, i.e. enabled or disabled, depending on CCE mode of M_CAN, see ***M_CAN in CCE mode***. 0: DMU is disabled 1: DMU is enabled and can process DMA data
14:13	Reserved	Must be kept at reset value.
12:8	TFQPIP[4:0]	TX FIFO/Queue put index previous Put index, used by DMU Element Handler for previous enqueueing of a Tx Element.
7:4	Reserved	Must be kept at reset value.
3	TXER	TX event service request line of DMU Logical value of the output signal dmu_txer, which requests a service from the external DMA to dequeue Tx event information with DMU TX Event Element. 0: No TX Event DMA service requested 1: TX Event DMA Service requested
2	RX1R	RX1 service request line of DMU Logical value of the output Signal dmu_rx1r, which requests a service from the external DMA to dequeue a CAN message with DMU RX1 Element. 0: No RX1 DMA service requested 1: RX1 DMA Service requested
1	RX0R	RX0 service request line of DMU Logical value of the output signal dmu_rx0r, which requests a service from the external DMA to dequeue a CAN message with DMU RX0 Element. 0: No RX0 DMA service requested

		1: RX0 DMA Service requested
0	TXR	TX service request line of DMU Logical value of the output signal dmu_txr, which requests a service from the external DMA to enqueue a CAN message with DMU TX Element. 0: No TX DMA service requested 1: TX DMA Service requested

DMU queueing counter register (CAN_DMU_DMUQC)

Address offset: 0x03C8
Reset value: 0x0000 0000

Every successful queueing will be counted, for each DMU Element separately. The counters wraps around at the end-value, i.e. modulo 256. All counters can be reset by a write access, write data is don't care.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXEEDC[7:0]								RX1EDC[7:0]							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX0EDC[7:0]								TXEEC[7:0]							
rw								rw							

Bits	Fields	Descriptions
31:24	TXEEDC[7:0]	TX event element dequeueing counter Increments on each successful dequeueing of a CAN TX Event information via DMU TX Event Element.
23:16	RX1EDC[7:0]	RX1 element dequeueing counter Increments on each successful dequeueing of a CAN RX message via DMU RX1 Element.
15:8	RX0EDC[7:0]	RX0 element dequeueing counter Increments on each successful dequeueing of a CAN RX message via DMU RX0 Element.
7:0	TXEEC[7:0]	TX element enqueueing counter Increments on each successful enqueueing of a CAN TX message via DMU TX Element.

DMU interrupt register (CAN_DMU_DMUIR)

Address offset: 0x03CC
Reset value: 0x0000 0000

This register provides information about occurrence of several events inside the DMU. A flag

is set when the related condition is detected (edge-sensitive). The flags remain set until the Host (CPU) clears them. A flag is cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect. A hard reset will clear the register. The configuration of CAN_DMU_DMUIE controls whether the DMU signals a level based interrupt on the interrupt line dmu_int.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	IAC	Reserved	TXEED	TXEEIW	TXEEIAS	TXEEID	TXEENSA	Reserved			RX1ED	RX1EIW	RX1EIAS	RX1EID	RX1ENSA
	rw		rw	rw	rw	rw	rw				rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			RX0ED	RX0EIW	RX0EIAS	RX0EID	RX0ENSA	Reserved	TXEE	TXEIR	TXEWATA	TXEIDLC	TXEIAS	TXEIE	TXENSA
			rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30	IAC	Illegal access while in configuration mode As long as the associated M_CAN is in CCE mode (M_CAN:CAN_M_CAN_CCCR.CCE = 1), the DMU is disabled and the Element Handler FSM is hold in state wait4cce, enqueueing and dequeueing is not possible, see ***M_CAN in CCE mode***. Any access to the DMU Virtual Buffers (TX, RX0, RX1, TX Event) sets the flag. 0: No Illegal Access while CCE mode 1: Illegal Access while CCE mode
29	Reserved	Must be kept at reset value.
28	TXEED	TX event element dequeued A successful dequeueing of a TX Event Element by the DMU sets this flag. 0: No TX Event Element dequeued 1: TX Event Element successfully dequeued
27	TXEEIW	TX event element illegal write All write accesses to the DMU TX Event Element are illegal. 0: No write access detected 1: Illegal write access to DMU TX Event Element detected, exception recovery started, see section ***Exception recovery at DMU virtual buffer***.
26	TXEEIAS	TX event element illegal access sequence When the Start Address of the TX Event Element was accessed, the DMU expects the transfer of the TX Event Element. These read accesses to DMU Section have to be strictly linear to ascending and consecutive addresses until reaching the Trigger Address. This flag gets also active, when accessing another Virtual Buffer, before TX Event Element Service has been completed. 0: No illegal addressing sequence detected 1: Accesses are not strictly linear to ascending and consecutive addresses,

		exception recovery started, see section ***Exception recovery at DMU virtual buffer***.
25	TXEEID	TX event element illegal dequeueing A dequeueing of a TX Event Element with the DMU should only be started (reading from the Start Address of the TX Event Virtual Buffer) if this is requested by DMU Request signal dmu_txer. 0: No illegal dequeueing 1: Start of dequeueing without request detected, exception recovery started, see section ***Exception recovery at DMU virtual buffer***.
24	TXEENSA	TX event element not start address An interaction with the TX Event Element Section of the DMU always has to start with a read access to the start address. This detector is only active when DMU is idle and thus awaits a start of a new service by an access to one of the Virtual Buffer's start addresses. 0: No illegal read access 1: Read from TX Event Element begins without using start address, exception recovery started, see section ***Exception recovery at DMU virtual buffer***.
23:21	Reserved	Must be kept at reset value.
20	RX1ED	RX1 element dequeued A successful dequeueing of a Rx message with the DMU RX1 Element sets this flag. 0: No Rx message dequeued 1: Rx message successfully dequeued
19	RX1EIW	RX1 element illegal write All write access to the DMU RX1 Element are illegal. 0: No write access detected 1: Illegal write access to DMU RX1 Element detected, exception recovery started, see section ***Exception recovery at DMU virtual buffer***.
18	RX1EIAS	RX1 element illegal access sequence When the start address of the RX1 Element was accessed, the DMU expects the transfer of the CAN message. These read accesses to DMU RX1 Element have to be strictly linear to ascending and consecutive addresses until reaching the Trigger Address. This flag gets also active, when accessing another Virtual Buffer, before RX1 Element service has been completed. 0: No illegal addressing sequence detected 1: Accesses are not strictly linear to ascending and consecutive addresses, exception recovery started, see section ***Exception recovery at DMU virtual buffer***.
17	RX1EID	RX1 element illegal dequeueing

		A dequeueing of a Rx message with the Rx section of the DMU should only be started (reading from the start address of the RX1 Virtual Buffer) if this is requested by DMU Request signal dmu_rx1r. 0: No illegal dequeueing 1: Start of dequeueing without request detected, exception recovery started, see section ***Exception recovery at DMU virtual buffer***.
16	RX1ENSA	RX1 element not start address An interaction with the RX1 section of the DMU always has to start with a read access to the start address. This detector is only active when DMU is idle and thus awaits a start of a new service by an access to one of the Virtual Buffer's start addresses. 0: No illegal read access 1: Read from RX1 Element begins without using start address, exception recovery started, see section ***Exception recovery at DMU virtual buffer***.
15:13	Reserved	Must be kept at reset value.
12	RX0ED	This flag has the same function as the RX1 element flag RX1ED.
11	RX0EIW	This flag has the same function as the RX1 element flag RX1EIW.
10	RX0EIAS	This flag has the same function as the RX1 element flag RX1EIAS.
9	RX0EID	This flag has the same function as the RX1 element flag RX1EID.
8	RX0ENSA	This flag has the same function as the RX1 element flag RX1ENSA.
7	Reserved	Must be kept at reset value.
6	TXEE	TX element enqueued A successful enqueueing of a Tx message with the DMU TX Element section sets this flag. 0: No Tx message enqueued 1: Tx message successfully enqueued
5	TXEIR	TX element illegal read All read accesses to the DMU TX Element section are illegal and return undefined read data. 0: No read access 1: Illegal read access to DMU TX Element section detected, exception recovery started, see section ***Exception recovery at DMU virtual buffer***.
4	TXEWATA	TX element write after trigger address The DMU expects that only the active words of a TX Element are transferred, see section ***Enqueue a message***. In some applications, the message transfer with dynamic length, controlled by DLC is not applicable. In such a case, the DMA controller may transfer (write) the maximum size Tx Buffer element, independent of the actual length of the contained message. The DMU tolerates this by discarding these accesses and setting this flag. This exception handling will be finished with

		accessing an address other than between Trigger Address and last address of TX Element section or on occurrence of an access error, which normally aborts a pending enqueueing. 0: No write after Trigger Address 1: Write after Trigger address detected
3	TXEIDLC	TX element illegal DLC In the case that the DLC written to TX Element indicates a message that is larger than the configuration of M_CAN's Tx Buffer element size (M_CAN:CAN_M_CAN_TXESC.TBDS), the DMU aborts the enqueueing. Example: Tx Buffer element size is set to 8 byte data fields (M_CAN:CAN_M_CAN_TXESC.TBDS = 0). The DLC in T1 written to DMU TX Element indicates a CAN message with 64 byte data field. 0: No illegal DLC detected 1: DLC exceeds Tx Buffer element size of M_CAN, exception recovery started, see section ***Exception recovery at DMU virtual buffer***.
2	TXEIAS	TX element illegal access sequence By accessing the start address of the TX Element, the DMU awaits the transfer of the CAN message. These write accesses to DMU TX Element have to be strictly linear to ascending and consecutive addresses until reaching the last word of the CAN message. This flag gets also active, when accessing another Virtual Buffer, before TX Element service has been completed. 0: No illegal addressing sequence detected 1: Accesses are not strictly linear to ascending and consecutive addresses, exception recovery started, see section ***Exception recovery at DMU virtual buffer***.
1	TXEIE	TX element illegal enqueueing An enqueueing of a Tx message with the TX Section of the DMU should only be started (writing to the start address of the TX Section) if this is requested by DMU Request signal dmu_txr. 0: No illegal enqueueing 1: Start of enqueueing without request detected, exception recovery started, see section ***Exception recovery at DMU virtual buffer***.
0	TXENSA	TX element not start address An interaction with the TX Section of the DMU always has to start with a write access to the start address. This detector is only active when DMU is idle and thus awaits a start of a new service by an access to one of the Virtual Buffer's start addresses. 0: No illegal write access 1: Write to TX Element begins without using start address, exception recovery started, see section ***Exception recovery at DMU virtual buffer***.

DMU Interrupt Enable Register (CAN_DMU_DMUIE)

Address offset: 0x03D0
Reset value: 0x0000 0000

The register controls whether the DMU signals an interrupt on interrupt line dmu_int. Thus, for each flag in the Interrupt register CAN_DMU_DMUIR this register provides a dedicated enable bit.

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	IACE	Reserved	TXEEDE	TXEEIWE	TXEEIASE	TXEEIDE	TXEENSAE	Reserved			RX1EDE	RX1EIWE	RX1EIASE	RX1EIDE	RX1ENSAE
	rw		rw	rw	rw	rw	rw				rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			RX0EDE	RX0EIWE	RX0EIASE	RX0EIDE	RX0ENSAE	Reserved	TXEEE	TXEIRE	TXEWATAE	TXEIDLCE	TXEIASE	TXEIEE	TXENSAE
			rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30	IACE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
29	Reserved	Must be kept at reset value.
28	TXEEDE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
27	TXEEIWE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
26	TXEEIASE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
25	TXEEIDE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
24	TXEENSAE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
23:21	Reserved	Must be kept at reset value.
20	RX1EDE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
19	RX1EIWE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
18	RX1EIASE	This bit has the same behaviour as the TX element not start address enable flag

		TXENSAE.
17	RX1EIDE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
16	RX1ENSAE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
15:13	Reserved	Must be kept at reset value.
12	RX0EDE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
11	RX0EIWE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
10	RX0EIASE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
9	RX0EIDE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
8	RX0ENSAE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
7	Reserved	Must be kept at reset value.
6	TXEEE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
5	TXEIRE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
4	TXEWATAE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
3	TXEIDLCE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
2	TXEIASE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
1	TXEIEE	This bit has the same behaviour as the TX element not start address enable flag TXENSAE.
0	TXENSAE	TX element not start address enable 0: Flag CAN_DMU_DMUIR.TXENSA does not activate the interrupt line dmu_int 1: If CAN_DMU_DMUIR.TXENSA = '1' the interrupt line dmu_int will be activated

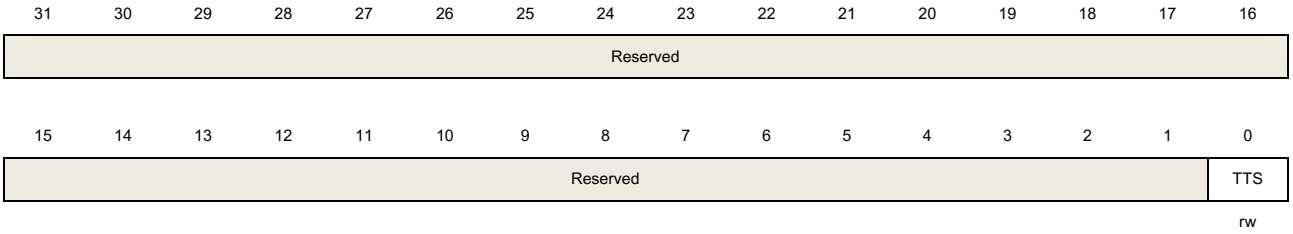
DMU Configuration Register (CAN_DMU_DMUC)

Address offset: 0x03D4

Reset value: 0x0000 0000

This is the configuration register of the DMU.

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	TTS	Transfer timestamp This flag is kept to '0'. 0: No timestamp will be transferred via DMU Virtual Buffer 1: Reserved

23. Comparator (CMP)

23.1. Overview

The general purpose CMP can work either standalone (all terminal are available on I / Os) or together with the timers.

It can provide a trigger source when an analog signal is in a certain condition, achieve some current control by working together with a PWM output of a timer and the DAC.

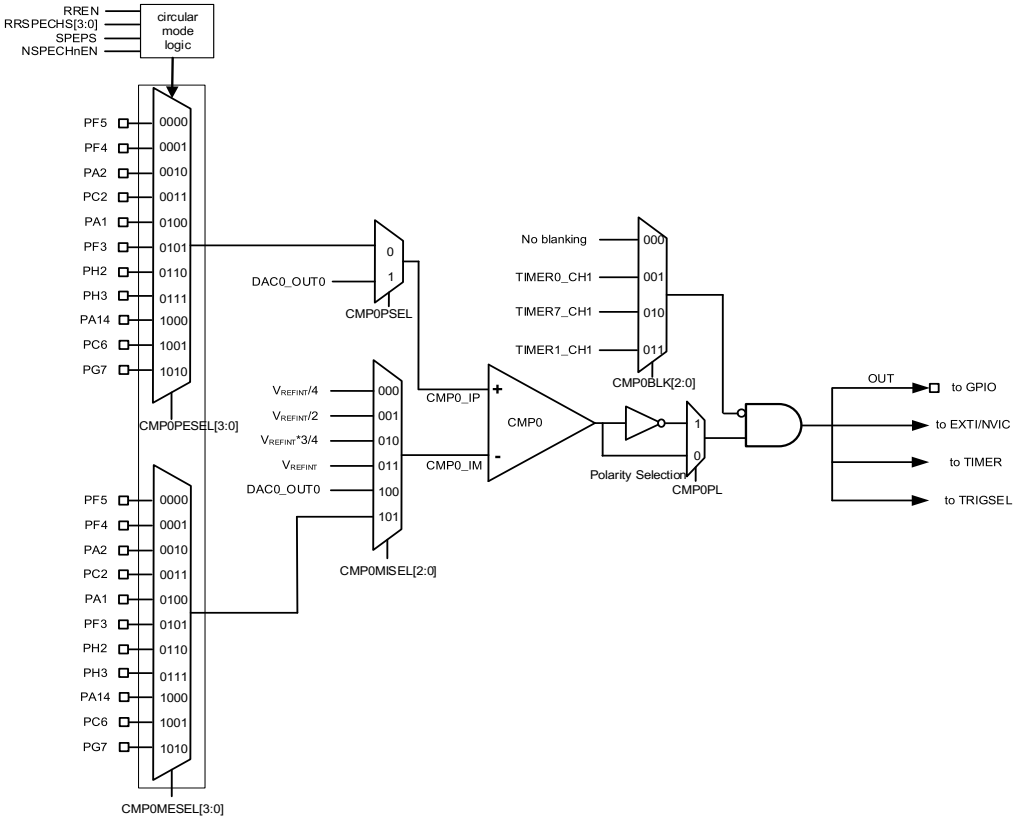
23.2. Characteristics

- Rail-to-rail comparators.
- Configurable hysteresis.
- Configurable speed and consumption.
- Configurable analog input source.
  - DAC output.
  - Multiplexed I / O pins.
  - The whole or sub-multiple values of internal reference voltage.
- Outputs with blanking source.
- Outputs to I / O.
- Outputs to timers.
- Outputs to EXTI and NVIC.
- Outputs to TRIGSEL.
- Support circular mode.

23.3. Function overview

The block diagram of CMP0 is shown below(CMP1 is similar to CMP0):

Figure 23-1. CMP block diagram



Note: V_{REFINT} is 1.2V.

23.3.1. CMP clock

The clock of the CMP which is connected to APB bus, is synchronous with PCLK.

23.3.2. CMP I / O configuration

These I / Os must be configured in analog mode in the GPIOs registers before they are selected as CMP inputs.

The CMP output can be redirected internally and externally simultaneously.

Refer to pin definitions in datasheet, and the CMP output can be connected to the corresponding I/O port via the alternate function of the GPIO.

CMP output internally connect to the TIMER and the connections between them are as follows:

- CMP output to the TIMER input channel.

Table 23-1. CMP inputs and outputs summary details the inputs and outputs of the CMP.

Table 23-1. CMP inputs and outputs summary

	CMP0	CMP1
CMP non inverting inputs connected to I / Os	PF5 PF4 PA2 PC2 PA1 PF3 PH2 PH3 PA14 PC6 PG7	PC6 PG7 PJ11 PA14 PK1 PK2 PG2 PG5
CMP inverting inputs connected to I / Os	PF5 PF4 PA2 PC2 PA1 PF3 PH2 PH3 PA14 PC6 PG7	PC6 PG7 PJ11 PA14 PK1 PK2 PG2 PG5
CMP inverting inputs connected to internal signals	V_{REFINT} / 4 V_{REFINT} / 2 V_{REFINT} * 3 / 4 V_{REFINT} DAC0_OUT0	V_{REFINT} / 4 V_{REFINT} / 2 V_{REFINT} * 3 / 4 V_{REFINT} DAC1_OUT0
CMP outputs connected to I / Os	PA13 PK6 PL4 PL5	PD6 PM4
CMP outputs connected to I / Os(circular mode)	PF3 PF7 PK7 PM5	PA15 PD7 PE6 PL10
CMP outputs connected to EXTI	•	
CMP outputs connected to TRIGSEL	•	

	CMP0	CMP1
CMP outputs connected to internal signals	TIMER0_CH0 TIMER7_CH0	TIMER20_CH0 TIMER60_CH0

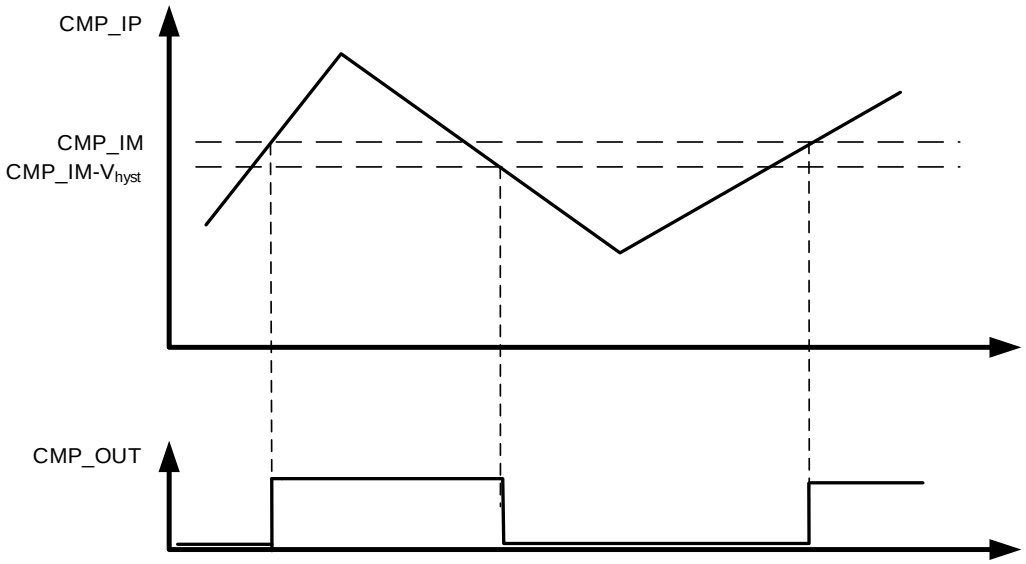
23.3.3. CMP operating mode

For a given application, there is a trade-off between the CMP power consumption versus propagation delay, which is adjusted by configuring bits CMPxM [1:0] in CMPx_CS register. The CMP works fastest with highest power consumption when CMPxM[1:0] = 2'b00, while works slowest with lowest power consumption when CMPxM[1:0] = 2'b11.

23.3.4. CMP hysteresis

In order to avoid spurious output transitions that caused by the noise signal, a programmable hysteresis is designed to force the hysteresis value by configuring CMPx_CS register. This function could be shut down if it is unnecessary.

Figure 23-2. CMP hysteresis



23.3.5. CMP register write protection

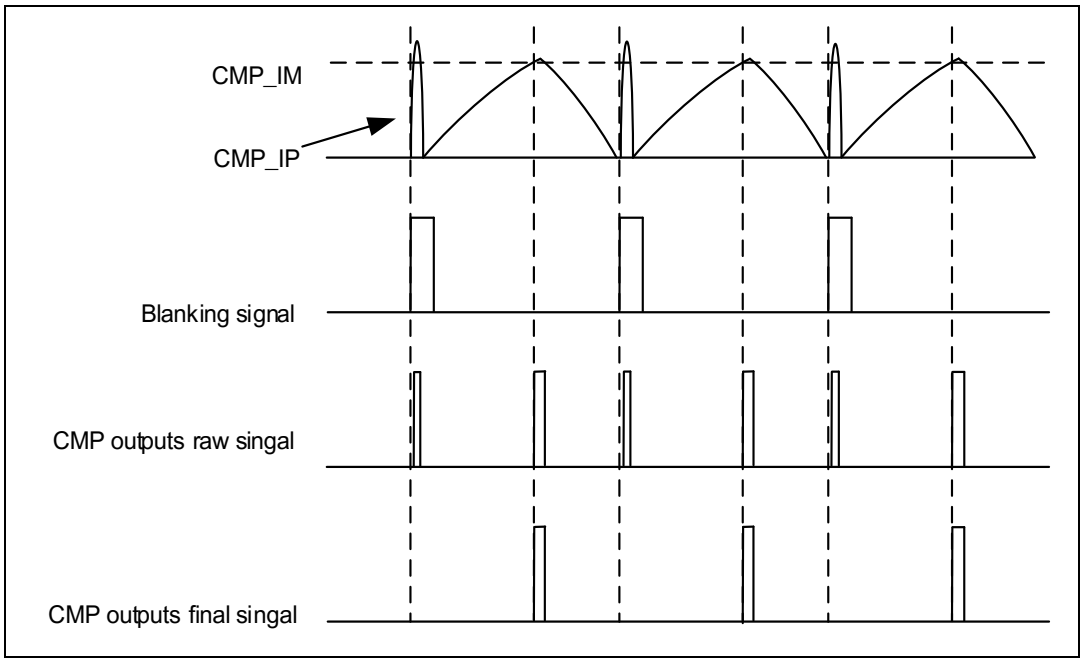
The CMP control and status register (CMPx_CS) can be protected from writing by setting CMPxLK bit to 1. The CMPx_CS register, including the CMPxLK bit will be read-only, and can only be reset by the MCU reset.

23.3.6. CMP output blanking

CMP output blanking function can be used to avoid interference of short pulses in the input signal to CMP output signal. If the CMPxBLK[2:0] bits in the CMPx_CS register are setting to an available value, the CMP output final signal is obtained by ANDing the complementary signal of the selected blanking signal with the raw output of the comparator. The blanking function can be used for false overcurrent detection in motor control applications.

Figure 23-3. The CMP outputs signal blanking shows the comparator output blank function.

Figure 23-3. The CMP outputs signal blanking



23.3.7. CMP voltage scaler function

The voltage scaler function can provide selectable 1 / 4, 1 / 2, 3 / 4 reference voltage for CMP input. It is controlled by CMPxSEN and CMPxBEN bits in CMPx control / status register. The CMPxSEN and CMPxBEN bits are used to enable the V_{REFINT} voltage output and the divider circuit, respectively, to generate the selected voltage.

23.3.8. CMP interrupt

The CMP output is connected to the EXTI and the EXTI line is exclusive to CMP. With this function, CMP can generate either interrupt or event which could be used to exit from low-power mode.

23.3.9. CMP circular mode

The circular mode is mainly about a specified channel for polling comparison, and it is realized through Round-Robin algorithm (support DAC loop test). The circular mode is enable by

RREN bit in the CMPx_RRCTL register and CMPxEN bit in the CMPx_CS register. And the CMP circular mode supports standby mode.

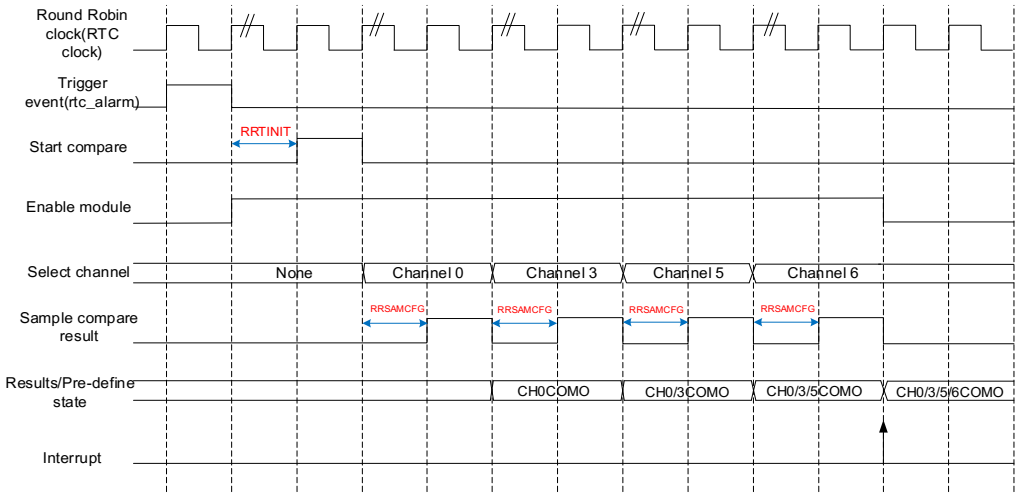
The specified port for the non-inverting or inverting port is determined by SPEPS bit in the CMPx_RRCHCTL register. The specified channel is determined by RRSPECHS[3:0] bits in the CMPx_RRCHCTL register. The non-specified channels are selected by NSPECHnEN(n=0…7, and 8, 9, 10 only for CMP0) bit in the CMPx_RRCHCTL register.

The CMP will be active when the trigger event arrives. The analog module stabilization time should be longer than the chip datasheet design, and the time set is controlled by RRTINIT[5:0] bits in the CMPx_RRCTL register. After the stabilization process is completed, the Round Robin comparison sequence starts. After the number of clocks defined by RRSAMCFG[1:0] bits in the CMPx_RRCTL register, the comparison result of the selected active channel is sampled.

The pre-define state is set by CHnCOMO(n=0…7, and 8, 9, 10 only for CMP0) bit in the CMPx_RRCOMO register, and the latest comparison result of each channel is stored in this bit. After comparison of all active channels, if the comparison result is different from the pre-define state, the corresponding CHnCOMF(n=0…7, and 8, 9, 10 only for CMP0) bit in the CMPx_RRCOMF register is set. If Interrupt enable for Round-Robin, when any flag in the CMPx_RRCOMF register is produced, an interrupt occurs. These flags do not generate dma event.

As is shown in ***Figure 23-4. Round Robin diagram***. Channel0, channel3, channel5 and channel6 are selected. Set the analog module initialize time through RRTINIT[5:0] bits, and set time through RRSAMCFG[1:0] bits before sampling. If the comparison results of any channel is different from the pre-define state, an interrupt will be generated when Interrupt enable.

Figure 23-4. Round Robin diagram



Note: The trigger event starts a comparison sequence, and the next trigger event must wait until the current sequence is completed.

So the process can be divided to several steps as below:

Step1: Delay time and sample clock configuration. Configure RRTINIT[5:0] bits and RRSAMCFG[1:0] bits in the CMPx_RRCTL register.

Step2: Specified port selection. Select the specified port of CMP by configuring SPEPS bit in the CMPx_RRCHCTL register. User can select a fixed channel as a specified port by configuring RRSPECHS[3:0] bits in the CMPx_RRCHCTL register, or choose the DAC output by CMPxPSEL bit / CMPxMISEL[2:0] bits in the CMPx_CS.

Step3: Non-specified port selection. The channels are selected by NSPECHnEN bit in the CMPx_RRCHCTL register.

Step4: Pre-define the state configuration. Pre-define the state of CHnCOMO bit in the CMPx_RRCOMO register.

Step5: Clear latest flags. Clear the CHnCOMF bit in the CMPx_RRCOMF register.

Step6: Interrupt enable. User can enable or disable Interrupt by RRIE bit in the CMPx_RRCTL register.

Step7: Round-Robin mode enable. Round-Robin enable by RREN bit in the CMPx_RRCTL register.

Step8: Cmp enable. CMP enable by CMP0EN bit in the CMPx_CS register.

Step9: Trigger event. The CLRCNT bit is set in the CMPx_RRTCTL register, and RRTVAL[31:0] bits in the CMPx_RRTTRGV register are configured after reading the CLRCNT bit zero, and then updated by UPVAL bit in the CMPx_RRTCTL register. When UPVAL bit is zero, enable RRTRGEN bit and RRCEN bit in the CMPx_RRTCTL register.

When the reference terminal value is greater than the output of a configured non-reference terminal channel, the corresponding non-reference terminal channel flag bit CHnCOMO (n = 0..10) in the polling comparison output register (CMPx_RRCOMO) is set to 1. At this time, if the comparison result of the non-reference channel is inconsistent with the pre-set result of CHnCOMO (n = 0..10) (if the cyclic polling interrupt is enabled in step 6), a comparator interrupt will be triggered, and the interrupt signal will be output via the CMP-specific EXTI line.

23.4. Register definition

CMP0 base address: 0x4001 7C00

CMP1 base address: 0x4001 6C00

Table 23-49. CMP register overview

Register name	Symbol	Address offset	Reset value
CMP0 Control / status register	CMP0_CS	0x00	0x0000 0000
CMP1 Control / status register	CMP1_CS	0x00	0x0000 0000
Round Robin control register	CMPx_RRCTL	0x10	0x0000 0000
Round Robin channel control Register	CMPx_RRCHCTL	0x14	0x0000 0000
Round Robin compare output register	CMPx_RRCOMO	0x18	0x0000 0000
Round Robin compare result flag register	CMPx_RRCOMF	0x1C	0x0000 0000
Round Robin timer control register	CMPx_RRTCTL	0x20	0x0000 0000
Round Robin timer trigger value register	CMPx_RRTTRGV	0x24	0xFFFF FFFF
Round Robin timer counter register	CMPx_RRTCNT	0x28	0x0000 0000

23.4.1. CMP0 Control / status register (CMP0_CS)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMP0LK	CMP0O	Reserved		CMP0PESEL[3]	CMP0MESEL[3]	CMP0OSEL[1:0]		CMP0SEN	CMP0BEN	Reserved	CMP0BLK[2:0]			CMP0HST[1:0]	
rwo	r			rw	rw	rw		rw	rw		rw			rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP0PL	Reserved	CMP0PSEL	CMP0PESEL[2:0]			CMP0MISEL[2:0]			CMP0MESEL[2:0]			CMP0M[1:0]		Reserved	CMP0EN
rw		rw	rw			rw			rw			rw			rw

Bits	Fields	Descriptions
31	CMP0LK	CMP0 lock This bit can set all control bits of CMP0 as read-only. It can only be set once by software and cleared by a system reset. 0: CMP0_CS[31:0] bits are read-write 1: CMP0_CS[31:0] bits are read-only
30	CMP0O	CMP0 output state This bit is a copy of CMP output state, which is read only. 0: Non-inverting input below inverting input and the output is low

		1: Non-inverting input above inverting input and the output is high
29:28	Reserved	Must be kept at reset value.
27	CMP0PESEL[3]	CMP0_IP external input selection Refer to CMP0PESEL[2:0] description.
26	CMP0MESEL[3]	CMP0_IM external input selection Refer to CMP0MESEL [2:0] description.
25:24	CMP0OSEL[1:0]	CMP0 output selection These bits are used to select the destination of the CMP0 output. 00: no selection 01: TIMER0 CH0 input capture 10: TIMER7 CH0 input capture 11: Reserved **Note:** It is recommended to enable CMP first, and then configure the timer channel, when using TIMER to capture the output signal of the comparator.
23	CMP0SEN	Voltage scaler enable bit This bit is set and cleared by software. This bit enables the outputs of the VREFINT divider, which is treated as the inverting input of the CMP. 0: Disable bandgap scaler in case that CMP1SEN bit of CMP1_CS is also reset. 1: Enable bandgap scaler
22	CMP0BEN	Scaler bridge enable bit 0: Disable scaler resistor bridge in case that CMP1BEN bit of CMP1_CS is also reset 1: Enable scaler resistor bridge
21	Reserved	Must be kept at reset value.
20:18	CMP0BLK[2:0]	CMP0 output blanking source This bit is used to select which timer output controls the CMP0 output blanking. 000: No blanking 001: Select TIMER0_CH1 as blanking source 010: Select TIMER7_CH1 as blanking source 011: Select TIMER1_CH1 as blanking source 100~111: Reserved
17:16	CMP0HST[1:0]	CMP0 hysteresis These bits are used to control the hysteresis level. 00: No hysteresis 01: Low hysteresis 10: Medium hysteresis 11: High hysteresis
15	CMP0PL	Polarity of CMP0 output This bit is used to select the polarity of CMP0 output.

		0: Output is not inverted 1: Output is inverted
14	Reserved	Must be kept at reset value.
13	CMP0PSEL	CMP0_IP input selection These bits are used to select the source connected to the CMP0_IP input of the CMP. 0: Result of CMP0_IP external input selection 1: DAC0_OUT0
12:10	CMP0PESEL[2:0]	CMP0_IP external input selection These bits and CMP0PESEL[3] are used to select the external source connected to the CMP0_IP input of the CMP0. 0000: PF5 0001: PF4 0010: PA2 0011: PC2 0100: PA1 0101: PF3 0110: PH2 0111: PH3 1000: PA14 1001: PC6 1010: PG7
9:7	CMP0MISEL[2:0]	CMP0_IM internal input selection These bits and CMP0MISEL[3] are used to select the internal source connected to the CMP0_IM input of the CMP. 000: V_{REFINT} / 4 001: V_{REFINT} / 2 010: V_{REFINT} * 3 / 4 011: V_{REFINT} 100: DAC0_OUT0 101: Result of CMP0_IM external input selection 110~111: Reserved
6:4	CMP0MESEL[2:0]	CMP0_IM external input selection These bits and CMP0MESEL[3] are used to select the external source connected to the CMP0_IM input of the CMP0. 0000: PF5 0001: PF4 0010: PA2 0011: PC2 0100: PA1 0101: PF3 0110: PH2

		0111: PH3 1000: PA14 1001: PC6 1010: PG7
3:2	CMP0M[1:0]	CMP0 mode These bits are used to control the operating mode of the CMP adjust the speed / consumption. 00: High speed / full power 01 / 10: Medium speed / medium power 11: Low speed / low power
1	Reserved	Must be kept at reset value.
0	CMP0EN	CMP0 enable 0: CMP0 disabled 1: CMP0 enabled

23.4.2. CMP1 Control / status register (CMP1_CS)

Address offset: 0x00
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMP1LK	CMP1O	Reserved				CMP1OSEL[1:0]		CMP1SEN	CMP1BEN	Reserved	CMP1BLK[2:0]			CMP1HST[1:0]	
rwo	r					rw		rw	rw		rw			rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP1PL	Reserved	CMP1PSEL	CMP1PESEL[2:0]			CMP1MISEL[2:0]			CMP1MESEL[2:0]			CMP1M[1:0]		Reserved	CMP1EN
rw		rw	rw			rw			rw			rw			rw

Bits	Fields	Descriptions
31	CMP1LK	CMP1 lock This bit can set all control bits of CMP1 as read-only. It can only be set once by software and cleared by a system reset. 0: CMP1_CS[31:0] bits are read-write 1: CMP1_CS[31:0] bits are read-only
30	CMP1O	CMP1 output state This bit is a copy of CMP output state, which is read only. 0: Non-inverting input below inverting input and the output is low 1: Non-inverting input above inverting input and the output is high
29:26	Reserved	Must be kept at reset value.
25:24	CMP1OSEL[1:0]	CMP1 output selection

		These bits are used to select the destination of the CMP1 output. 00: no selection 01: TIMER20 CH0 input capture 10: TIMER60 CH0 input capture 11: Reserved **Note:** It is recommended to enable CMP first, and then configure the timer channel, when using TIMER to capture the output signal of the comparator.
23	CMP1SEN	Voltage scaler enable bit This bit is set and cleared by software. This bit enables the outputs of the VREFINT divider, which is treated as the inverting input of the CMP. 0: Disable bandgap scaler in case that CMP0SEN bit of CMP0_CS is also reset. 1: Enable bandgap scaler
22	CMP1BEN	Scaler bridge enable bit 0: Disable scaler resistor bridge in case that CMP0BEN bit of CMP0_CS is also reset 1: Enable scaler resistor bridge
21	Reserved	Must be kept at reset value.
20:18	CMP1BLK[2:0]	CMP1 output blanking source This bit is used to select which timer output controls the CMP1 output blanking. 000: No blanking 001: Select TIMER0_CH1 as blanking source 010: Select TIMER7_CH1 as blanking source 011: Select TIMER1_CH1 as blanking source 100~111: Reserved
17:16	CMP1HST[1:0]	CMP1 hysteresis These bits are used to control the hysteresis level. 00: No hysteresis 01: Low hysteresis 10: Medium hysteresis 11: High hysteresis
15	CMP1PL	Polarity of CMP1 output This bit is used to select the polarity of CMP1 output. 0: Output is not inverted 1: Output is inverted
14	Reserved	Must be kept at reset value.
13	CMP1PSEL	CMP1_IP input selection These bits are used to select the source connected to the CMP1_IP input of the CMP. 0: Result of CMP1_IP external input selection 1: DAC1_OUT0

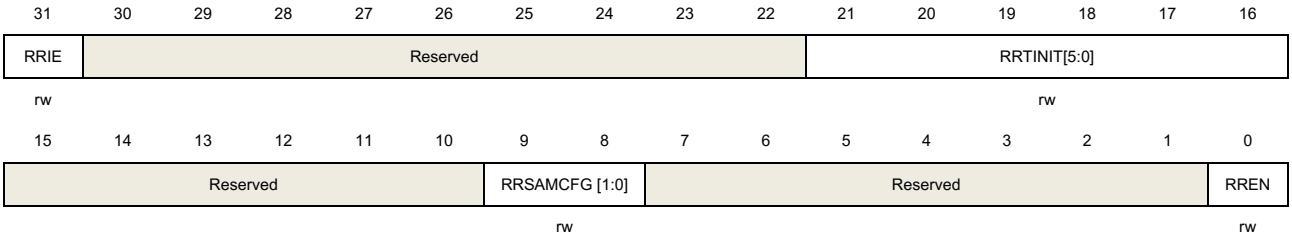
12:10	CMP1PESEL[2:0]	CMP1_IP external input selection These bits are used to select the external source connected to the CMP1_IP input of the CMP1. 000: PC6 001: PG7 010: PJ11 011: PA14 100: PK1 101: PK2 110: PG2 111: PG5
9:7	CMP1MISEL[2:0]	CMP1_IM internal input selection These bits are used to select the internal source connected to the CMP1_IM input of the CMP. 000: V_{REFINT} / 4 001: V_{REFINT} / 2 010: V_{REFINT} * 3 / 4 011: V_{REFINT} 100: DAC1_OUT0 101: Result of CMP1_IM external input selection 110~111: Reserved
6:4	CMP1MESEL[2:0]	CMP1_IM external input selection These bits are used to select the external source connected to the CMP1_IM input of the CMP1. 000: PC6 001: PG7 010: PJ11 011: PA14 100: PK1 101: PK2 110: PG2 111: PG5
3:2	CMP1M[1:0]	CMP1 mode These bits are used to control the operating mode of the CMP adjust the speed / consumption. 00: High speed / full power 01 / 10: Medium speed / medium power 11: Low speed / low power
1	Reserved	Must be kept at reset value.
0	CMP1EN	CMP1 enable 0: CMP1 disabled 1: CMP1 enabled

23.4.3. Round Robin control register(CMPx_RRCTL)

Address offset: 0x10

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31	RRIE	Interrupt enable for Round-Robin 0: Round-Robin interrupt disable 1: Round-Robin interrupt enable **Note**: 1. Interrupt generate when the Specified channel comparison result change. 2. When this bit is enabled, Round-Robin interrupt will occupy the CMP EXTI interrupt line.
30:22	Reserved	Must be kept at reset value.
21:16	RRTINIT[5:0]	Round-Robin time initialization Determine the value of round-robin clock(RTC clock) cycles for CMP and DAC initialization. The delay formula is: (RRTINIT+1) * (round-robin clock) 000000: 1 cycle 000001: 2 cycle … 111111: 64 cycles Note: The configuration cycle time of RRTINIT[5:0] needs to be greater than t_{START_SCALER}. Refer to the Datasheet for specific values. When the RTC clock frequency is 32KHz, 1 cycle meets the requirement.
15:10	Reserved	Must be kept at reset value.
9:8	RRSAMCFG[1:0]	Round-Robin sample clocks configuration Before sampling, wait for RRSAMCFG round-robin clock cycles in each active channel. 00: 0 clocks 01: 1 clocks 10: 2 clocks 11: 3 clocks
7:1	Reserved	Must be kept at reset value.

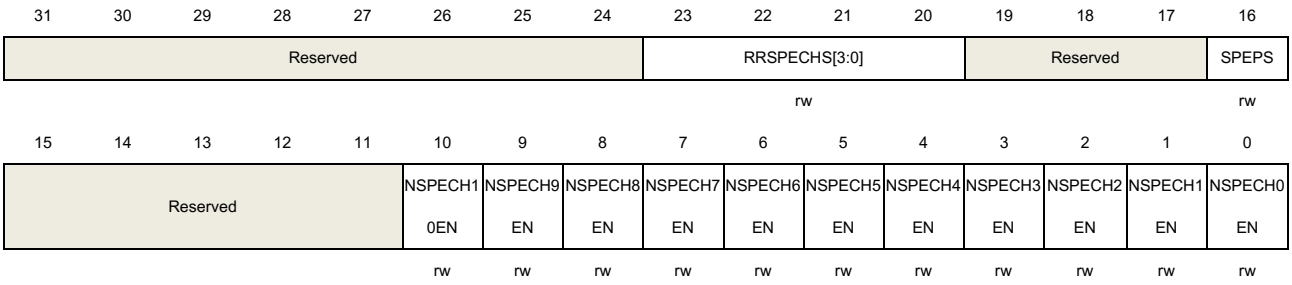
0	RREN	Round-Robin enable 0: Round-Robin disable 1: Round-Robin enable

23.4.4. Round Robin channel control Register (CMPx_RRCHCTL)

Address offset: 0x14

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:20	RRSPECHS[3:0]	Round-Robin specified channel selection(Channel 8\9\10 only for CMP0)) Select a channel in the mux port for trigger mode. The pin of the channel refer to CMPxPESEL or CMPxMESEL in CMPx_CS register. 0000: Channel 0 … 1010: Channel 10
19:17	Reserved	Must be kept at reset value.
16	SPEPS	Specified port selection Select a mux port (inverting or non-inverting port) for trigger mode. The inputs to the non-specified port are swept during each round. 0: Select the inverting port. The inputs to the non-inverting port are scan during round. 1: Select the non-inverting port. The inputs to the inverting port are scan during round.
15:11	Reserved	Must be kept at reset value.
10	NSPECH10EN	Non-specified mux port channel 10 enable (only for CMP0) Refer to NSPECH0EN bit
9	NSPECH9EN	Non-specified mux port channel 9 enable (only for CMP0) Refer to NSPECH0EN bit

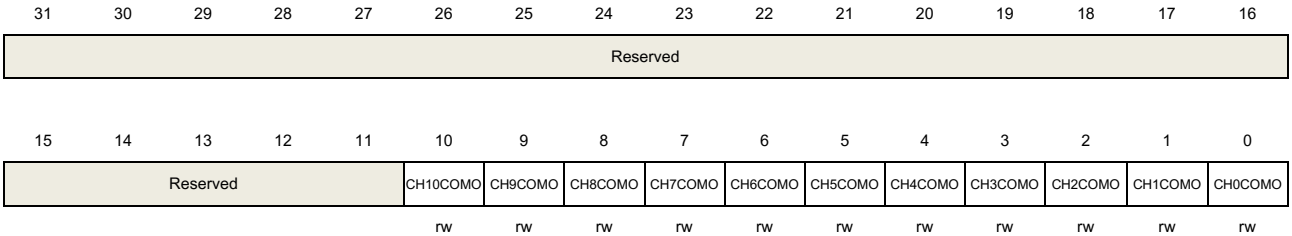
8	NSPECH8EN	Non-specified mux port channel 8 enable (only for CMP0) Refer to NSPECH0EN bit
7	NSPECH7EN	Non-specified mux port channel 7 enable Refer to NSPECH0EN bit
6	NSPECH6EN	Non-specified mux port channel 6 enable Refer to NSPECH0EN bit
5	NSPECH5EN	Non-specified mux port channel 5 enable Refer to NSPECH0EN bit
4	NSPECH4EN	Non-specified mux port channel 4 enable Refer to NSPECH0EN bit
3	NSPECH3EN	Non-specified mux port channel 3 enable Refer to NSPECH0EN bit
2	NSPECH2EN	Non-specified mux port channel 2 enable Refer to NSPECH0EN bit
1	NSPECH1EN	Non-specified mux port channel 1 enable Refer to NSPECH0EN bit
0	NSPECH0EN	Non-specified mux port channel 0 enable Enable channel 0 for non-specified mux ports to round compare. 0: Disable channel 0 in circular mode 1: Enable channel 0 in circular mode

23.4.5. Round Robin compare output register(CMPx_RRCOMO)

Address offset: 0x18

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:11	Reserved	Must be kept at reset value.
10	CH10COMO	Channel 10 compare output state (only for CMP0) Refer to CH0COMO bit
9	CH9COMO	Channel 9 compare output state (only for CMP0)

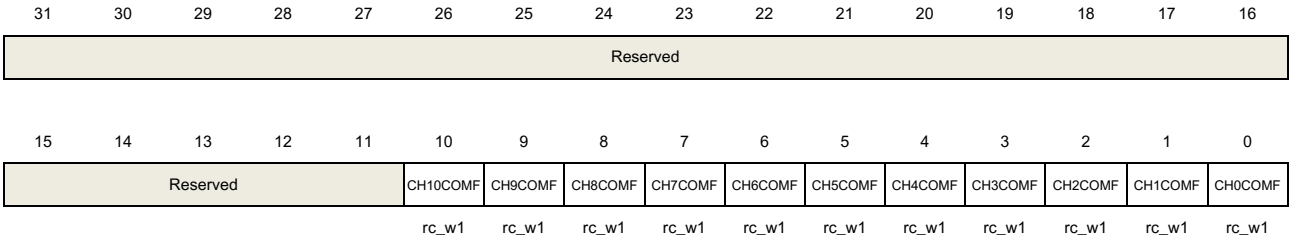
		Refer to CH0COMO bit
8	CH8COMO	Channel 8 compare output state (only for CMP0) Refer to CH0COMO bit
7	CH7COMO	Channel 7 compare output state Refer to CH0COMO bit
6	CH6COMO	Channel 6 compare output state Refer to CH0COMO bit
5	CH5COMO	Channel 5 compare output state Refer to CH0COMO bit
4	CH4COMO	Channel 4 compare output state Refer to CH0COMO bit
3	CH3COMO	Channel 3 compare output state Refer to CH0COMO bit
2	CH2COMO	Channel 2 compare output state Refer to CH0COMO bit
1	CH1COMO	Channel 1 compare output state Refer to CH0COMO bit
0	CH0COMO	Channel 0 compare output state User can pre-define the state of CH0COMO by writing values, or reading the Channel 0 latest comparison result.

23.4.6. Round Robin compare result flag register (CMPx_RRCOMF)

Address offset: 0x1C

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:11	Reserved	Must be kept at reset value.
10	CH10COMF	Channel 10 compare result flag (only for CMP0) Refer to CH0COMF bit
9	CH9COMF	Channel 9 compare result flag (only for CMP0)

		Refer to CH0COMF bit
8	CH8COMF	Channel 8 compare result flag (only for CMP0) Refer to CH0COMF bit
7	CH7COMF	Channel 7 compare result flag Refer to CH0COMF bit
6	CH6COMF	Channel 6 compare result flag Refer to CH0COMF bit
5	CH5COMF	Channel 5 compare result flag Refer to CH0COMF bit
4	CH4COMF	Channel 4 compare result flag Refer to CH0COMF bit
3	CH3COMF	Channel 3 compare result flag Refer to CH0COMF bit
2	CH2COMF	Channel 2 compare result flag Refer to CH0COMF bit
1	CH1COMF	Channel 1 compare result flag Refer to CH0COMF bit
0	CH0COMF	Channel 0 compare result flag When the latest comparison result of channel 0 is different from the pre-define value, this bit will be set. 0: The result is not different 1: The result is different

23.4.7. Round Robin timer control register (CMPx_RRTCTL)

Address offset: 0x20
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												UPVAL	CLRCNT	RRTRGEN	RRCEN
												rc_w1	rc_w1	rw	rw

Bits	Fields	Descriptions
31:4	Reserved	Must be kept at reset value.
3	UPVAL	Write 1 to update the RRTVAL[31:0] value, automatically reset by hardware

2	CLRCNT	Write 1 to clear the RRCNT[31:0] value, automatically reset by hardware
1	RRTRGEN	Round Robin trigger enable 0: Disable 1: Enable
0	RRCEN	Round Robin counter enable 0: Disable 1: Enable

23.4.8. Round Robin timer trigger value register (CMPx_RRTTRGV)

Address offset: 0x24
Reset value: 0xFFFF FFFF

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RRTVAL[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RRTVAL[15:0]															
rw															

Bits	Fields	Descriptions
31:0	RRTVAL[31:0]	Round Robin timer trigger value When these bit fields match the TIMER counter value, RRCNT[31:0] will be reset, and round robin trigger generates if the RRTRGEN bit is set. Note: For the roundrobin function to operate properly, the value of the RRTVAL bit field (Round-Robin timer trigger value) must be greater than (RRTINIT[5:0] + 1) * the number of polling channels + the value of RRSAMCFG[1:0].

23.4.9. Round Robin timer counter register (CMPx_RRTCNT)

Address offset: 0x28
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RRCNT[31:15]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RRCNT[15:0]															
r															

Bits	Fields	Descriptions
31:0	RRCNT[31:0]	Round Robin timer count This bit-field indicates the current counter value. Due to clock synchronization, the RRCNT[31:0] value may represent the value of the counter at the previous instant.

24. Self-test control module (STCM)

24.1. Overview

The self-test control module, STCM, provides the built-in self-test (BIST) control, which controls the memory built-in self-test (MBIST) being executed in serial or parallel according to the power and timing consumption, test coverage constrains. GD32A711 / A712 devices only support MBIST.

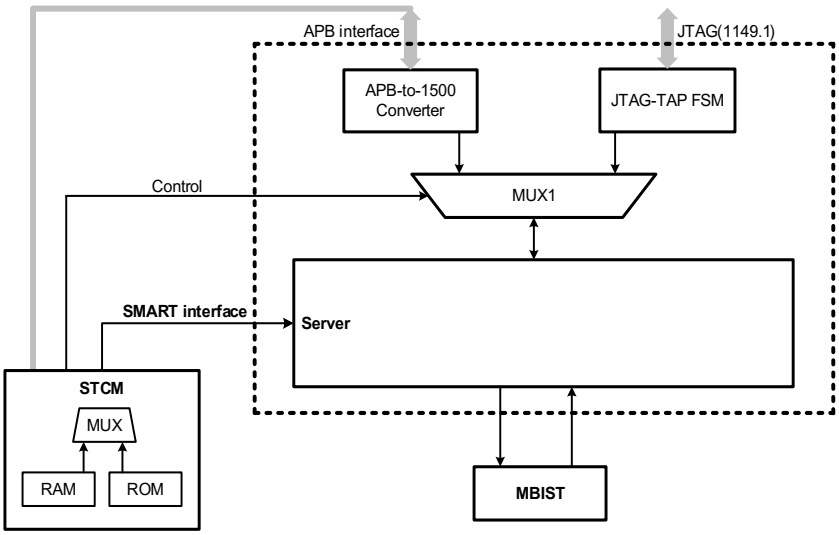
24.2. Characteristics

- Control MBIST serial or parallel execution through SMART interface.
- Support serial or parallel execution between MBIST.
- Support specific MBIST partition bypass in a BIST phase.
- Support 14 kinds of MBIST execution algorithm.
- Support recoverable / unrecoverable error state mapping during BIST execution.
- Provide BIST execution information: MBIST execution time, execution result, error information.
- Support watchdog timer mechanism for monitoring MBIST execution time-out.
- Support fault injection.

24.3. Function overview

24.3.1. System overview

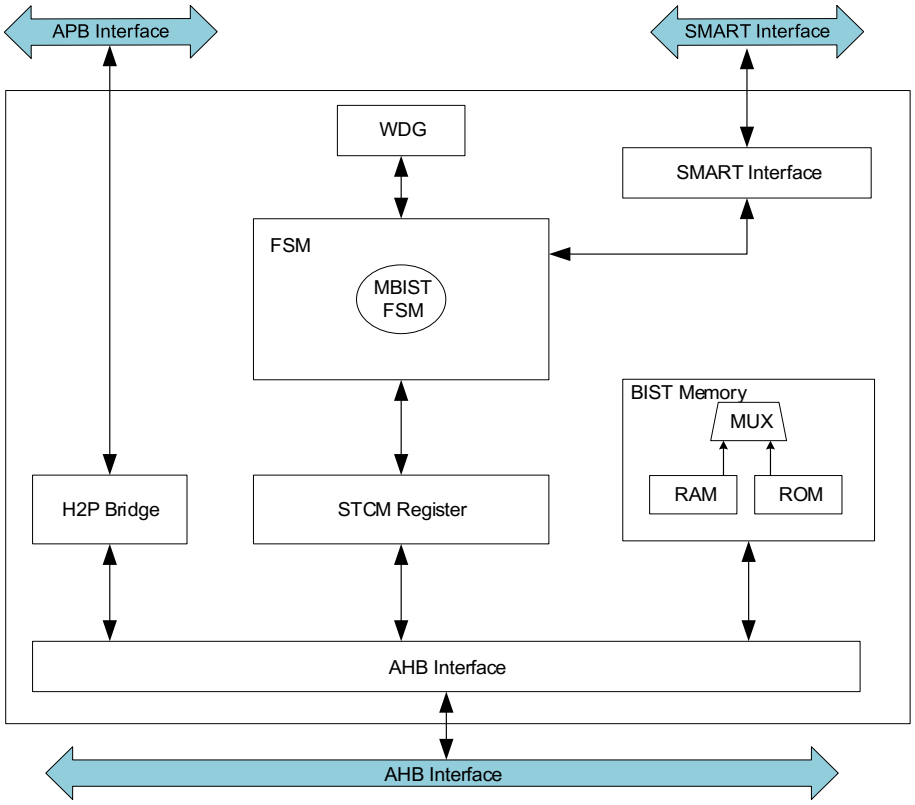
Figure 24-1. STCM system diagram of GD32A711 / A712 devices



The execution of MBIST is implemented through an internal SMART interface.

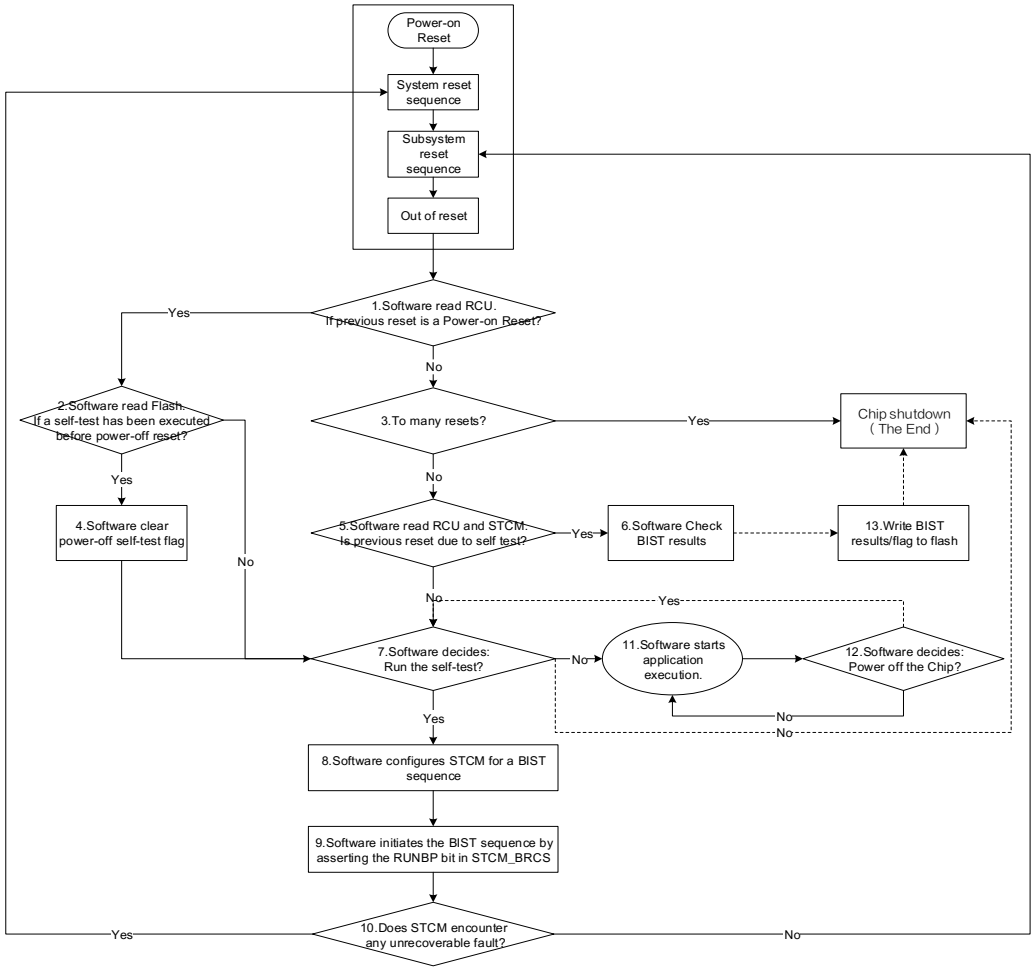
24.3.2. Block diagram

Figure 24-2. STCM block diagram of GD32A711 / A712 devices



24.3.3. Self-test process

Figure 24-3. Self-test process



Power on self-test:

The following steps show the power on self-test sequence.

1. After power-on reset sequence execution is finished, software reads PORF in RCU_SYSRSTSTAT to check whether the last reset is power-on reset. If yes, go to Step 2. Otherwise, go to Step 3.
2. Software reads the flash to check whether self-test is performed before power-off. If yes, go to Step 4. Otherwise, go to Step 7.
3. If reset too many times, shut down the chip. Otherwise, go to Step 5.
4. Software clears the self-test flag in flash which is set before power-off, go to Step 7.
5. Software reads STCM_REC_FAIL in RCU_SYSRSTSTAT and STCM to check whether last reset is caused by self-test. If yes, go to Step 6. Otherwise, go to Step 7.
6. Software checks BIST execution result and then go to Step 7.

7. Software determines whether to do a self-test. If yes, go to Step 8. Otherwise, go to Step 11.
8. Software configures BIST phase.
9. Software starts a self-test by configuring RUNBP in STCM_BRCS register.
10. If an unrecoverable error occurs during self-test process, perform a system reset sequence, otherwise perform a subsystem reset sequence.

Power off self-test:

11. Software start application execution.
12. Software determines whether to perform a power-off reset sequence. If yes, go to Step 7. Otherwise, go to Step 11.
13. Software determines whether to do a self-test. If yes, the execution sequence is 8 / 9 / 10 / 1 / 3 / 5 / 6 / 13. Otherwise, shutdown the chip.
14. Software writes BIST result into flash, then go to Step 7.

24.3.4. BIST partition mapping

The table lists below shows the MBIST mapping *Table 24-1. GD32A711 / A712 MBIST mapping* of the chip:

Table 24-1. GD32A711 / A712 MBIST mapping

BIST index	BIST name	Memory index	Memory instance name
MBIST partition0	CM7 D-CACHE data	0	CM7 data cache block 0
	CM7 D-CACHE data	1	CM7 data cache block 1
	CM7 D-CACHE data	2	CM7 data cache block 2
	CM7 D-CACHE data	3	CM7 data cache block 3
	CM7 D-CACHE data	4	CM7 data cache block 4
	CM7 D-CACHE data	5	CM7 data cache block 5
	CM7 D-CACHE data	6	CM7 data cache block 6
	CM7 D-CACHE data	7	CM7 data cache block 7
	CM7 D-CACHE tag	8	CM7 data cache tag block 0
	CM7 D-CACHE tag	9	CM7 data cache tag block 1
	CM7 D-CACHE tag	10	CM7 data cache tag block 2
	CM7 D-CACHE tag	11	CM7 data cache tag block 3
	CM7 I-CACHE data	12	CM7 instruction cache data block 0

BIST index	BIST name	Memory index	Memory instance name
	CM7 I-CACHE data	13	CM7 instruction cache data block 1
	CM7 I-CACHE tag	14	CM7 instruction cache tag block 0
	CM7 I-CACHE tag	15	CM7 instruction cache tag block 1
MBIST partition1	FMC	0	FMC RAM0
	FMC	1	FMC RAM1
	FMC	2	FMC RAM2
	FMC	3	FMC RAM3
	FMC	4	FMC RAM4
MBIST partition2	CAN0	0	CAN0 SRAM
	CAN1	1	CAN1 SRAM
	CAN2	2	CAN2 SRAM
	CAN3	3	CAN3 SRAM
	CAN4	4	CAN4 SRAM
	CAN5	5	CAN5 SRAM
	CAN6	6	CAN6 SRAM
	CAN7	7	CAN7 SRAM
MBIST partition3	CM7_D0TCM	0	CM7 D0TCM SRAM block 0
	CM7_D1TCM	1	CM7 D1TCM SRAM block 0
	CM7_D0TCM	2	CM7 D0TCM SRAM block 1
	CM7_D1TCM	3	CM7 D1TCM SRAM block 1
	CM7_ITCM	4	CM7 ITCM SRAM block 0
	CM7_ITCM	5	CM7 ITCM SRAM block 1
MBIST partition4	HSM_PKCAU	0	HSM PKCAU SRAM block 0
	HSM_PKCAU	1	HSM PKCAU SRAM block 1
	HSM_PKCAU	2	HSM PKCAU SRAM block 2
	HSM_SM2	3	HSM SM2 SRAM block 0
	HSM SRAM	4	HSM SRAM block 0
	HSM SRAM	5	HSM SRAM block 1
	HSM SRAM	6	HSM SRAM block 2
	HSM SRAM	7	HSM SRAM block 3

BIST index	BIST name	Memory index	Memory instance name
	HSM SRAM	8	HSM SRAM block 4
	HSM SRAM	9	HSM SRAM block 5
MBIST partition5	SRAM(retention in standby mode)	0	SRAM (retention in standby mode) block 0
	SRAM(retention in standby mode)	1	SRAM (retention in standby mode) block 1
MBIST partition6	HSM ROM	0	HSM ROM

24.3.5. BIST sequence / BIST phase / MBIST execution

MBIST execution: STCM starts a MBIST execution through the SMART interface and gets the execution result. Only one MBIST algorithm can be executed at a time.

BIST phase: From software set the RUNBP in STCM_BRCS register to hardware reset it, and go into the corresponding reset sequence.

BIST sequence: Multiple BIST phases constitute a BIST sequence.

24.3.6. Register access

STCM registers belong to AHB slave domain:

Only support word(32-bit) read access for all register.

Support byte (8-bit), half word (16-bit) or word (32-bit) write access operation, and write access is valid only when WAP in STCM_CFG0 is set to 0, but WAP in STCM_CFG0, UES / RES in STCM_BESTAT1 are excepted.

STCM will generate bus errors in the following cases:

1. WPK register related:
   a. Improper bus access: non-word (32-bit).
   b. Not correct protect key:
      - Wrong protect key.
      - Wrong sequence of writing protect key (Writing protect key 2 first, and then protect key 2 or Writing protect key 2/1 twice in a row).
      -Write access between writing protect key.
2. Write/read access to STCM register domain reserved address.
3. Non byte (8-bit), half word (16-bit) or word (32-bit) access

4. Write access under STCM register is not unlocked.
5. Write access during WAP in STCM_CFG0 is 1.
6. Write access to read only register.
7. Non word (32-bit) write access to word (32-bit) write only register.

Note: Write access to register's reserved bit will not cause any transfer error. Write access is invalid during RUNBP in STCM_BRCS is 1 and read access will always return 0.

24.3.7. Clock and reset

STCM module clock input:

CK_ATE: 50Mhz or 48Mhz, from RCU-CCTL.

STCM module has two reset sources: system reset, power on reset

Configuration and result registers in self-test belong to power on reset, others belong to system reset.

24.3.8. AHB interface

AHB interface mainly completes the following tasks:

1. Based on the AHB access address, determine whether current selected slave is STCM register or the RAM in BIST Memory.
2. Based on the configuration, when syscfg.muxctl is set to 1, transfer AHB access to APB access and then give it to Server
3. When STCM_ITSEL in SYSCFG_PMCFG is set to 0, isolate APB interface as invalid

Table 24-2. STCM address information

Item	Address
STCM	0x4002 5000 - 0x4002 5BFF
STCM Register	0x4002 5000 - 0x4002 50FF
BIST Memory (AHB access)	0x4002 5200 - 0x4002 59FF

24.3.9. Write protection mechanisms

The STCM_WPK register provides the write protection mechanism for write access to other AHB register in STCM domain.

The security key is automatically updated in 3-loop mode, is shown in *Table 24-3. Security key loop* :

Table 24-3. Security key loop

Loop	Key 1	Key 2
1	0x8E6B2508	0xAC4B130A

Loop	Key 1	Key 2
2	0x9F7C3619	0xBD5C241B
3	0xA08D472A	0xCE6D352C

After power-on reset, the security key is reset to the values corresponding to Loop 1. Each time STCM register is unlocked thereafter, the security key is updated to the next set of values. After Loop3 is unlocked, the security key is returned to Loop 1.

After power-on reset, software needs to write the following sequence of keys to STCM_WPK register to unlock the write protection of STCM register:

- Write key 1
- Write key 2

Write access to STCM_WPK registers only supports word (32-bit) full key write. STCM_WPK is unreadable, so read access to this register will get all 0.

There is a special watchdog's mechanism for STCM_WPK register:
After each reset, watchdog's counter value is reset to the initial value 0xFFFFFFFF, write access to STCM register is in locked state, and then this watchdog counter start counting down. The counting clock of watchdog is HCLK.

After the correct security key sequence is written into STCM_WPK register, the watchdog counter is reset to its initial value and write access to STCM is unlocked (if the write access was previously locked). And writing access time window will be extended, if the security key is written to STCM_WPK register before watchdog timeout.

When the window value reaches 0, a timeout of watchdog occurs and write access to the STCM register field is locked.

The following write access to the STCM_WPK register will cause a bus error and simultaneously lock STCM module write access:

1. Unreasonable bus access: non word (32-bit) write.
2. Incorrect key:
   a. Incorrect key.
   b. Incorrect timing of key writing (key 2 followed by key 1, key 1/2 for two consecutive times, etc.).
3. Write access to STCM between STCM_WPK key sequences.

24.3.10. Watchdog mechanism

STCM provides a watchdog window mechanism for MBIST execution. The watchdog window value is configurable by software, and the watchdog uses downward counting.

When STCM is in MBIST phase, MBIST watchdog starts counting, and when timeout occurs, STCM returns to idle state and exits current BIST phase.

Users can obtain the configured MBSIT running time from STCM_MBWDGT.

24.3.11. BIST chain configuration

The execution configuration of MBIST is implemented through a linked list. Each MBIST partition has its own pointer.

Firstly, user configures the FBPTR in STCM_CFG0 register, which points to the MBIST for the first execution.

For each MBIST partition, there is a corresponding STCM_MBxPCFG register, which stores the next BIST pointer NXTPTR and the execution mode (serial / parallel) of current and next BIST.

For each MBIST partition, there is a bit MBEN in corresponding STCM_MBxPCFG register which can be used to configure whether this part in linked list is bypassed.

24.3.12. Recoverable / Unrecoverable error

STCM supports mapping the errors happened in self-test to recoverable or unrecoverable categories. The error mapping can be configured in registers STCM_BEM / STCM_MBPEM.

When unrecoverable error occurs during execution, the RUNBP in STCM_BRCS register will be set to 0, and this error will be giving to RCU, the system reset sequence is then entered.

When recoverable error occurs during execution, the RUNBP in STCM_BRCS register will be set to 0 as well, and this error will be given to FMU, the subsystem reset sequence is then entered.

24.3.13. FSM state machines

STCM has three state machines that work together: ENGFSM, LBFSM and MBFSM.

The ENGFSM aims to dispatch the MBIST task assigned in STCM register.

The LBFSM aims to report to ENGFSM and send an order to AIT interface (Applicable for GD32A72x / A74x devices only).

The MBFSM aims to report to ENGFSM and send an order to SMART interface.

24.3.14. Reset events and configurations while self-test

Due to self-test is controlled by software, when self-test is running, it is expected to guarantee that the whole process of self-test should not be interrupted by subsystem reset source. So getting all subsystem reset sources appropriately configured is way to achieve this. Items below specifies the configurations of subsystem reset sources for self-test:

- FMU reset: The FMU is a part of logic which is under self-test and thus it should be disabled during the self-test
- Self-test done reset: The self-test done indication resets the device post self-test. While self-test is running, the self-test done indication will be in disabled state

- WWDGT0 / WWDGT1 / FWWDG0 / FWWDG1 reset: WDG should be disabled in self-test configuration sequence
- JTAG reset: JTAG communication should be inactive while self-test
- HSM WDG reset: HSM WDG reset should be disabled in self-test configuration sequence.
- Software subsystem reset: The software is inactive during self-test and hence software subsystem reset will not arrive while the chip is under self-test.
- Debug subsystem reset: Debug should not be configured for debug subsystem reset generation while self-test is running.

Different from the subsystem reset sources above, the system reset sources should be available in self-test. If a system reset event occurs, a system reset sequence should be performed to reset STCM and self-test logic.

When all of self-test are finished, a subsystem reset sequence will be executed by RCU-RCTL.

24.3.15. Fault injection

STCM supports unrecoverable / recoverable fault injection through software, and recoverable fault injection through FMU hardware. The unrecoverable error is achieved by setting the UES in STCM_BESTAT1 register. The recoverable error is achieved by setting the RES in STCM_BESTAT1 register.

24.4. Register definition

STCM base address: 0x4002 5000

Table 24-4.STCM register overview

Register name	Symbol	Address offset	Reset value
TMR control and status register	STCM_TMRCS	0x00	0x0000 0001
BIST running control and status register	STCM_BRCS	0x04	0x0000 0000
Write protection key register	STCM_WPK	0x08	0x0000 0000
Configuration register 0	STCM_CFG0	0x0C	0x0000 0000
MBIST watchdog timer register	STCM_MBWDGT	0x18	0x0000 FFFF
BIST error state register 0	STCM_BESTAT0	0x1C	0x0000 0000
BIST error state register 1	STCM_BESTAT1	0x20	0x0000 0000
BIST error mapping register	STCM_BEM	0x24	0x0000 0000
BIST interval register	STCM_BINTVL	0x28	0x0000 0000
MBIST algorithm selection register	STCM_MBALGSEL	0x80	0x0000 0000
MBIST phase state register	STCM_MBPSTAT	0x84	0x0000 0000
MBIST phase end state register	STCM_MBPE_STAT	0x88	0x0000 0000
MBIST phase error mapping register	STCM_MBPEM	0x8C	0x0000 0000
MBIST phase launch register	STCM_MBLAUNCH	0x90	0x0000 0000
MBISTx phase configuration register	STCM_MBxPCFG	0xA0 + 0x04*x (x = 0...6)	0x0000 0000

24.4.1. TMR control and status register (STCM_TMRCS)

Address offset: 0x00
Reset value: 0x0000 0001

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).
This register can only be read by word (32-bit).

Note: This register can only be written when WAP is set to 0 in STCM_CFG0 register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														TMR_RUNBP	TMR_RUNBP_EN
														rc_w1	rw

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1	TMR_RUNBP	TMR (Triple-Modular Redundancy) state of run BIST phase

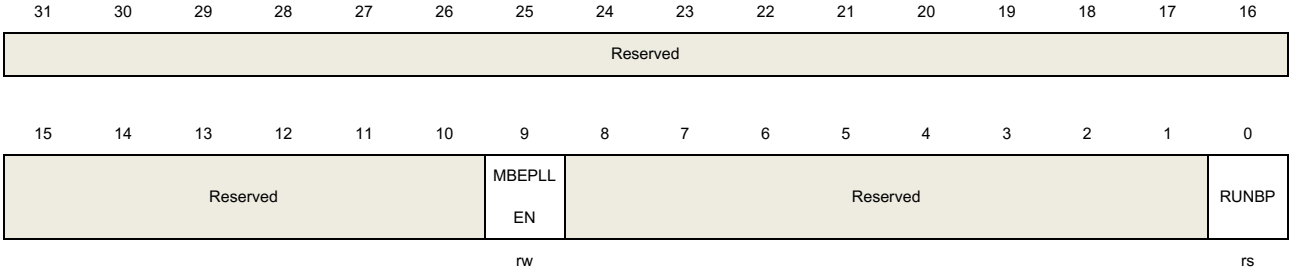
		0: Run BIST phase normal state 1: Run BIST phase error state
0	TMR_RUNBP_EN	TMR (Triple-Modular Redundancy) run BIST phase enable 0: Disable run BIST phase TMR fault to FMU reaction. 1: Enable run BIST phase TMR fault to FMU reaction.

24.4.2. BIST running control and status register (STCM_BRCS)

Address offset: 0x04
Reset value: 0x0000 0000

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).
This register can only be read by word (32-bit).

Note: This register can only be written when WAP is set to 0 in STCM_CFG0 register.



Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9	MBEPLLEN	MBIST executed with PLL enabled 0: Without using on-chip PLL. 1: With PLL enabled.
8:1	Reserved	Must be kept at reset value.
0	RUNBP	Run BIST phase 0: Idle. 1: Online BIST sequence procedure is running. This bit will be cleared automatically by hardware after current BIST phase execution is complete, and write 0 by software has no effect on this bit. During the period that RUNBP is 1,writing access to STCM's registers is invalid, and reading access to STCM's registers will return all Fs.

24.4.3. Write protection key register (STCM_WPK)

Offset: 0x08
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

Note: Writing this register will not be affected by WAP configuration in STCM_CFG0 register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WPK[31:16]															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WPK[15:0]															
w															

Bits	Fields	Descriptions
31:0	WPK[31:0]	STCM write protection key STCM write protection key for online test 8e6b2508 write protection key 0 to unlock the write access the STCM ac4b130a write protection key 1 to unlock the write access the STCM

24.4.4. Configuration register 0 (STCM_CFG0)

Address offset: 0x0C
Reset value: 0x0000 0000

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit), but for non-byte WAP clearing to 0, if the written value of other bits is different from actual value, MCU might go into hardfault. This register can only be read by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	FBPTR[5:0]						Reserved						MBWDG_DIV[2:0]		
	rw													rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							WAP	Reserved							
							rw								

Bits	Fields	Descriptions
31	Reserved	Must be kept at reset value.
30:25	FBPTR[5:0]	First BIST pointer 000000: Pointer to MBIST0 000001: Pointer to MBIST1 000010: Pointer to MBIST2 000011: Pointer to MBIST3 000100: Pointer to MBIST4 000101: Pointer to MBIST5 000110: Pointer to MBIST6 000111 ~ 111110: Reserved 111111: pointer to NIL. No BIST execution An error will be set into the INVPTR bit in STCM_BESTAT0 register if BIST pointer points to reserved area.

24:19	Reserved	Must be kept at reset value.
18:16	MBWDG_DIV[2:0]	MBIST watchdog clock divider 000: HCLK / 1 001: HCLK / 4 010: HCLK / 16 011: HCLK / 64 Others: HCLK / 1
15:9	Reserved	Must be kept at reset value.
8	WAP	Write access protection 0: All STCM registers can be written 1: All STCM registers except WAP and STCM_WPK cannot be written
7:0	Reserved	Must be kept at reset value.

24.4.5. MBIST watchdog timer register (STCM_MBWDGT)

Address offset: 0x18
Reset value: 0x0000 FFFF

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).
This register can only be read by word (32-bit).

Note: This register can only be written when WAP is set to 0 in STCM_CFG0 register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MBWDGT[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBWDGT[15:0]															
rw															

Bits	Fields	Descriptions
31:0	MBWDGT[31:0]	MBIST watch dog timer These bits store the timeout duration for completing all MBISTs in the current BIST phase. This register can prevent latch-up conditions during MBIST execution and avoid endless conditions when STCM schedules the MBIST execution queue, which could result in the MBIST failing to stop.

24.4.6. BIST error state register 0 (STCM_BESTAT0)

Address offset: 0x1C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

MBWDGEG	MBWDG0	MBWDG1	MBWDG2	MBWDG3	MBWDG4	MBWDG5	MBWDG6	Reserved
r	r	r	r	r	r	r	r	
15	14	13	12	11	10	9	8	7 6 5 4 3 2 1 0

Reserved	INVPTR	Reserved	PLLUNLK
	r		r

Bits	Fields	Descriptions
31	MBWDGEG	MBIST watchdog global error state 0: No error 1: Error detected **Note**: MBWDGx (x = 0...6) is valid only when MBWDGEG is 1.Since MBIST's WDG is HCLK counting and MBIST's execution clock is DIV_WRCK, the situation that MBWDGEG is 1 and MBWDGx is all 0 would occur, which can be avoided by appropriately increasing the count threshold.
30:24	MBWDGx	MBIST patition x watchdog error state (x = 0...6) 0: No error 1: Error detected
23:3	Reserved	Must be kept at reset value.
2	INVPTR	Invalid BIST pointer error state 0: No error 1: Error detected When the pointer of MBIST is out of bounds, this bit will be set. When the pointer points to NIL, this bit will be set. When BIST pointer chain configuration is detected to form a loop, this bit will be set.
1	Reserved	Must be kept at reset value.
0	PLLUNLK	PLL unlock error state 0: No error 1: Error detected When MBEPLLEN / LBEPLLEN in STCM_BRCS register is enabled and PLL unlock is detected during self-test, this bit will be set to 1.

24.4.7. BIST error state register 1 (STCM_BESTAT1)

Address offset: 0x20
Reset value: 0x0000 0000

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).
This register can only be read by word (32-bit).

Note: This register can only be written when WAP is set to 0 in STCM_CFG0 register, and

during word writing, MCU will go into hardfault if writing value is not same as the actual value for read only bits.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UES	RES	Reserved													
rw	rw														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													SP2	SP1	SP0
													r	r	r

Bits	Fields	Descriptions
31	UES	Unrecoverable error state. 0: No errors that trigger the unrecoverable error condition 1: There are errors that trigger the unrecoverable error condition This field can be set or cleared by software.
30	RES	Recoverable error state. 0: No errors that trigger the recoverable error condition 1: There are errors that trigger the recoverable error condition This field can be set or cleared by software,FMU dedicated channel.
29:3	Reserved	Must be kept at reset value.
2	SP2	SMART protocol error state 2 0: No error 1: Error detected
1	SP1	SMART protocol error state 1 0: No error 1: Error detected
0	SP0	SMART protocol error state 0 0: No error 1: Error detected

24.4.8. BIST error mapping register (STCM_BEM)

Address offset: 0x24
Reset value: 0x0000 0000

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).
This register can only be read by word (32-bit).

Note: This register can only be written when WAP is set to 0 in STCM_CFG0 register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											INVPTR	ENG	Reserved	MBWDG	PLLUNLK
											rw	rw		rw	rw

Bits	Fields	Descriptions
31:5	Reserved	Must be kept at reset value.
4	INVPTR	Invalid pointer unrecoverable error mapping 0: Recoverable error mapping 1: Unrecoverable error mapping
3	ENG	Engine unrecoverable error mapping 0: Recoverable error mapping 1: Unrecoverable error mapping Engine error mainly reports some SMART bus or AIT bus timing errors or FSM(Finite State Machine) related errors in STCM.
2	Reserved	Must be kept at reset value.
1	MBWDG	MBIST watchdog unrecoverable error mapping 0: Recoverable error mapping 1: Unrecoverable error mapping
0	PLLUNLK	PLL unlock unrecoverable error mapping 0: Recoverable error mapping 1: Unrecoverable error mapping

24.4.9. BIST interval register (STCM_BINTVL)

Address offset: 0x28
Reset value: 0x0000 0000

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).
This register can only be read by word (32-bit).

Note: This register can only be written when WAP is set to 0 in STCM_CFG0 register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INTVL[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTVL [15:0]															
rw															

Bits	Fields	Descriptions
31:0	INTVL[31:0]	BIST interval, number of HCLK cycles between BISTs execution.

24.4.10. MBIST algorithm selection register (STCM_MBALGSEL)

Address offset: 0x80
Reset value: 0x0000 0000

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).
This register can only be read by word (32-bit).

Note: This register can only be written when WAP is set to 0 in STCM_CFG0 register, and when multiple algorithms are selected, the algorithms will be executed in LSB order in current phase

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		MBALG13	MBALG12	MBALG11	MBALG10	MBALG9	MBALG8	MBALG7	MBALG6	MBALG5	MBALG4	MBALG3	MBALG2	MBALG1	MBALG0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13	MBALG13	0: Algorithm VLP21_LSDM_83N_CH_SP is not selected 1: Algorithm VLP21_LSDM_83N_CH_SP is selected
12	MBALG12	0: Algorithm VLSP_55N6 is not selected 1: Algorithm VLSP_55N6 is selected
11	MBALG11	0: Algorithm VLP9_14N_SO_SP is not selected 1: Algorithm VLP9_14N_SO_SP is selected
10	MBALG10	0: Algorithm VLP5_14N_CH SP is not selected 1: Algorithm VLP5_14N_CH SP is selected
9	MBALG9	0: Algorithm VLP3_22N_SO_SP is not selected 1: Algorithm VLP3_22N_SO_SP is selected
8	MBALG8	0: Algorithm VLP3_22N_CH_SP is not selected 1: Algorithm VLP3_22N_CH_SP is selected
7	MBALG7	0: Algorithm VLP2_26N_SO_SP is not selected 1: Algorithm VLP2_26N_SO_SP is selected
6	MBALG6	0: Algorithm VLP2_26N_CH_SP is not selected 1: Algorithm VLP2_26N_CH_SP is selected

5	MBALG5	0: Algorithm VLP1_26N_SO_SP is not selected 1: Algorithm VLP1_26N_SO_SP is selected
4	MBALG4	0: Algorithm VLP1_26N_CH_SP is not selected 1: AlgorithmVLP1_26N_CH_SP is selected
3	MBALG3	0: Algorithm VLP19_26N_SO_SP is not selected 1: Algorithm VLP19_26N_SO_SP is selected
2	MBALG2	0: Algorithm VLP18_14N_SO_SP is not selected 1: Algorithm VLP18_14N_SO_SP is selected
1	MBALG1	0: Algorithm VLP10_29N_SO_SP is not selected 1: Algorithm VLP10_29N_SO_SP is selected
0	MBALG0	0: Algorithm VLSP_55N6 is not selected 1: Algorithm VLSP_55N6 is selected Default.

The following are descriptions of the above 14 algorithms:

1. SO stands for the solid algorithm, which tests using all 0s/all 1s.
2. CH stands for the checkboard algorithm, which tests using adjacent addresses.
   Note: For the same type of fault, the checkboard algorithm generally has better coverage than the solid algorithm, though it takes slightly longer. If testing time is not a concern, it is recommended to use algorithms with the CH suffix.
3. #N represents the algorithm complexity. The larger the #, the higher the complexity and the longer the testing time.
4. The explanation for VLP# is as follows in the table:

Table 24-6.Algorithm description:

Test algorithms	Traditional faults	Address decoder faults	StaStatic unlinked faultstic unlinked	Static linked faults	Bit write enable faults	Bit-line leakage faults	Memory enable faults	Inter-port faults	Dynamic unlinked faults	Delay coupling faults	Data path faults	Memory biasing faults	Memory read-write contention faults	Static weakfaults	Address decoder activation delay faults	Address decoder deactivation delay	Write dominance faults	Write-thru faults	Bank address decoder faults	Read-write mask faults	Write enable faults	Linked static and dynamic faults	Bit-line pre-charge faults	Read mask faults	FinFET-specific faults	Retention faults	Process variation faults	Random telegraph faults
March	+	+	+	+	+	+	+																					
March	+	+							+	+																		

March	+	+							+	+																
March	+	+							+																	
March VLP5[3]	+	+													+											
March VLP9[3]	+	+													+											

24.4.11. MBIST phase state register (STCM_MBPSTAT)

Address offset: 0x84

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

Note: If MBIST partitionx fails to execute an algorithm(not including fail caused by timeout), MBPx will be reset and MBIST algorithms after this MBIST partition will not be executed, other MBIST partitions executing in parallel with this MBIST partition will continue to execute.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		MBP6[1:0]		MBP5[1:0]		MBP4[1:0]		MBP3[1:0]		MBP2[1:0]		MBP1[1:0]		MBP0[1:0]	
		r		r		r		r		r		r		r	

Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value
13:0	MBPx	MBIST partition x state (x = 0...6) 00: MBIST partition x execution failed 11: MBIST partition x execution successful Others: reserved

24.4.12. MBIST phase end state register (STCM_MBPE_STAT)

Address offset: 0x88

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

Note: MBPxEF will be set to 1 If MBIST partition x fails to execute an algorithm(not including fail caused by timeout), and the corresponding state and flag will be meaningless if execution timeout happened of a MBIST partition

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		MBP6EF[1:0]		MBP5EF[1:0]		MBP4EF[1:0]		MBP3EF[1:0]		MBP2EF[1:0]		MBP1EF[1:0]		MBP0EF[1:0]	
		r		r		r		r		r		r		r	

Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value
13:0	MBPxEF	MBIST partition x end flag (x = 0...6) 00: MBIST partition x execution not yet completed 11: MBIST partition x execution finished Others: reserved

24.4.13. MBIST phase error mapping register (STCM_MBPEM)

Address offset: 0x8C

Reset value: 0x0000 0000

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).

This register can only be read by word (32-bit).

Note: This register can only be written when WAP is set to 0 in STCM_CFG0 register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									MBP6	MBP5	MBP4	MBP3	MBP2	MBP1	MBP0
									rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value
6:0	MBPx	MBIST partition x unrecoverable error mapping (x = 0…6) 0: MBIST partition x recoverable error mapping 1: MBIST partition x unrecoverable error mapping

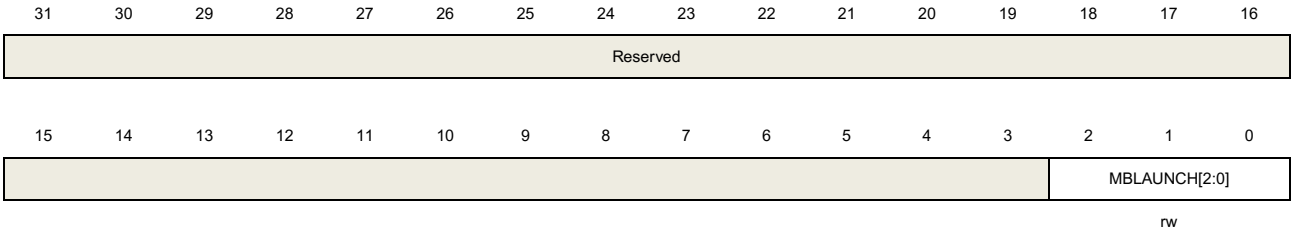
24.4.14. MBIST phase launch register (STCM_MBLAUNCH)

Address offset: 0x90

Reset value: 0x0000 0000

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).
This register can only be read by word (32-bit).

Note: This register can only be written when WAP is set to 0 in STCM_CFG0 register, and attempted accesses with the undefined reserved values in the register can result in undefined behavior.



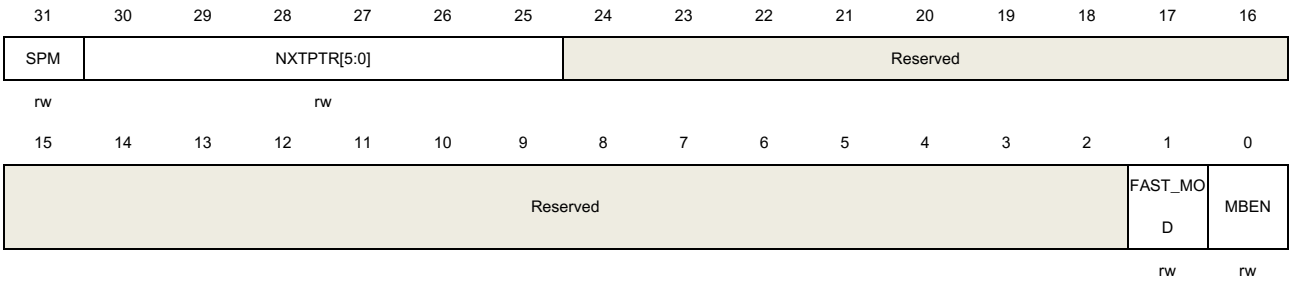
Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value
2:0	MBLAUNCH[2:0]	MBIST launch configuration 000: NOP,reset value 001: RUN_ONLY, run BIST only, STCM runs the selected BISTs without programming them first. 100: PROG_ONLY, program BIST only,STCM only programs BISTs and does not start them. This enables different BISTs to be programmed with different algorithms before all the selected BISTs are started using the RUN_ONLY command. 101: PROG_RUN, program BISTs and start them. Others: Reserved.

24.4.15. MBISTx phase configuration register (STCM_MBxPCFG)

Address offset: 0xA0 + 0x04*x (x = 0...6)
Reset value: 0x0000 0000

This register can be written by byte (8-bit), half word (16-bit) or word (32-bit).
This register can only be read by word (32-bit).

Note: This register can only be written when WAP is set to 0 in STCM_CFG0 register.



Bits	Fields	Descriptions
31	SPM	Serial/parallel mode

		0: Serial mode 1: Parallel mode
30:25	NXTPTR[5:0]	Next MBIST pointer When SPM in this register is set to 1, next MBIST partition will be executed in parallel with current, otherwise serial execution. And when PTR point to 111111 NIL, SPM in this register must be set to 0 to indicate current BIST phase is in the last partition. 000000: Pointer to MBIST0 000001: Pointer to MBIST1 000010: Pointer to MBIST2 000011: Pointer to MBIST3 000100: Pointer to MBIST4 000101: Pointer to MBIST5 000110: Pointer to MBIST6 000111 ~ 111110: Reserved 111111: pointer to NIL. No BIST execution An error will be set into the INVPTR bit in STCM_BESTAT0 register if BIST pointer points to reserved area. If current PTR is invalid and in the parallel execution configuration chain, all the rest valid PTR will not be executed as well, and INVPTR in STCM_BESTAT0 register will be set. If a partition fails to be executed in this part (one or more partitions that are selected to execute in parallel) of the linked list, the update of the linked list will not be affected, that is, the tasks of subsequent partitions will continue to be executed
24:2	Reserved	Must be kept at reset value.
1	FAST_MOD	Fast mode 0: MBIST partition selected by this phase will be executed in normal mode 1: MBIST partition selected by this phase will be executed in fast mode(In fast mode, certain memory addresses will undergo MBIST.)
0	MBEN	MBIST enable 0: MBIST disable for execution 1: MBIST enable for execution

25. Memory test controller (MTC)

25.1. Overview

The memory test controller (MTC) is used to generate one or multiple error bits to test memories / bus widgets in each channel. It is an AHB slave module that provides a memory test bus to connect to all safety related memory. MTC can generate one or more error bits to memory and buses with ECC, EDC, or ECC-after-EDC protection mechanisms.

25.2. Characteristics

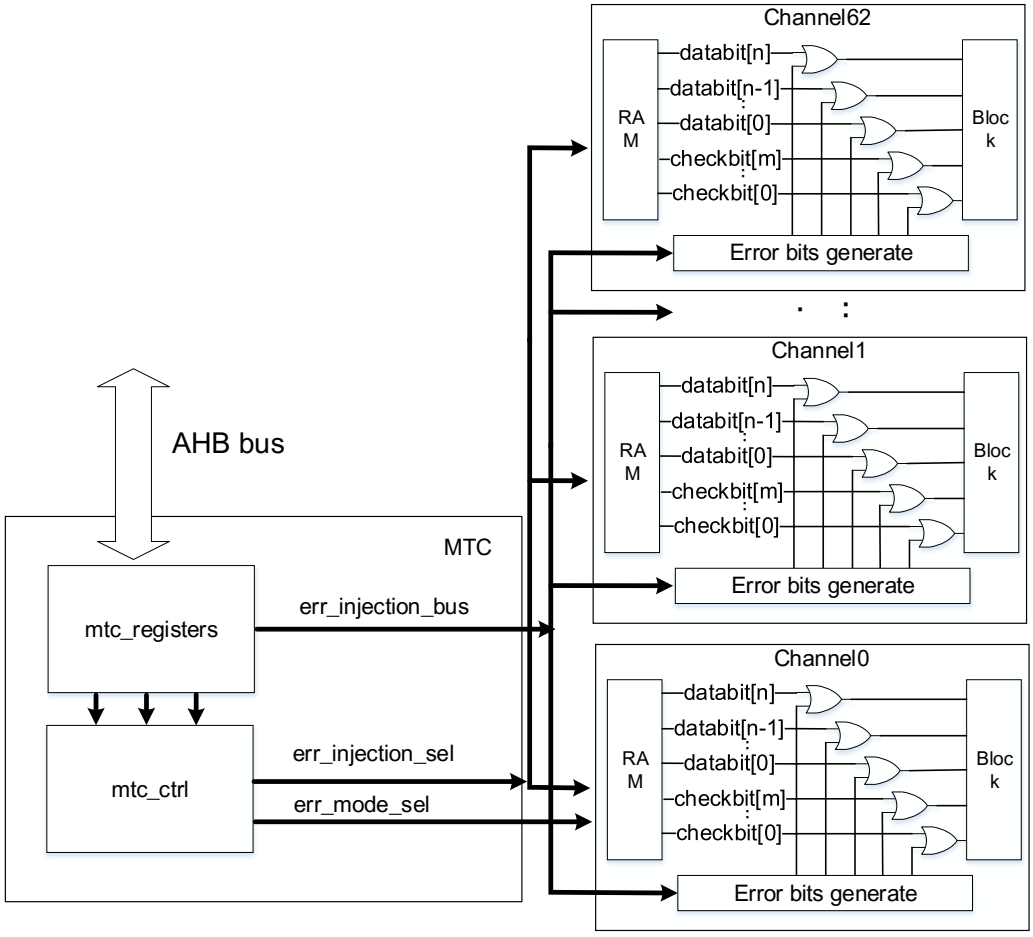
- Inject single or multi bits error to on-chip memory for test.
- Support three kinds of error injection modes: ECC, EDC, EDC after ECC
- Support parallel ECC injection with ECC address or EDC after ECC.
- Control registers MTC_CTL0 to enable error injection function.

25.3. Function overview

25.3.1. Block diagram

MTC supports to inject errors for Flash、CPU、CAN and BUS in global SRAM/local SRAM, to test if the ECC/EDC/EDC after ECC effective.And MTC support error injection in different modes simultaneously. MTC also support inject one or multi bits error to memory or bus gaskets in one injection mode. Every data/address bit in memory can be inversed. The function of this module is to provide incentives that memory or bus required. Every channel is corresponds to a type of error requirement. After the error injection is completed, modules will have conrresponding flag bits. See the specific module section for the information of the flags.

Figure 25-1. MTC block diagram



25.3.2. Error injection enable

The MTC module provides a two stage start protection mechanism to against accidental enabling the error injection function.

1st, write the register to select which channel or block will be inject errors, and no memory module will react to the default value;

2nd, write 1 to the enhanced safety registers to start error injection.

25.3.3. Single or multi bit error injection

Each bit of MTC error inject bus can inject an error to the selected memory array output data. Writing one or multi control register bits of error inject bus will inject one or multi bits error to the selected memory with no influence to the stored data. According to the Figure 3-2, the corresponding memory output data bits is inversed because the another input value of the XOR gate is 1. So that, there is one or multi bits different from the original data. The ECC and EDC logic in memory module shall find these errors and set relevant flags.all kinds of channels shares a set of bit fields. And some of injection modes also share a set of bit field.If

the number of bits flipped at the same time exceeds the limit required by the corresponding module, the expected error result cannot be obtained.

Single-bit injection type supported:ECC data,Address, EDC data, EDC after ECC.

Multi-bits injection type supported:1bit ECC data + 1bit ECC data,1bit Address + 1bit Address,1bit Address + 1bit ECC data,1bit Address + 1bit check bit,1bit EDC after ECC + 1bit EDC after ECC， 1bit EDC after ECC + 1bit ECC data,1bit EDC after ECC + 1bit check bit,2bit EDC after ECC+1bit ECC data.

Note: Address and EDC after ECC share the same bit field in MTC_EID register. MTC_EMS register is to distinguish the types.ECC data and EDC data belongs to different bit field disregard to the status of MTC_EMS register.

25.3.4. ECC data injection

ECC (Error correction code)data injection is to generate single-bit or multi-bits for checkbits or data,which is corretable. ECC data support 1 or 2 bits injection at one time,while check bit only support 1 bit.

Table 25-1. MTC ECC data channel mapping

Channel number	Memory of block	Check bits	ECC data bits
0	SRAM0	[136:128]	[63:0].
2	CM7_0 ITCM	[136:128]	[63:0].
3	CM7_0 D0TCM	[71:64]	[31:0].
4	CM7_0 D1TCM	[71:64]	[31:0].
8	MCAN_0	[71:64]	[31:0].
9	MCAN_1	[71:64]	[31:0].
10	MCAN_2	[71:64]	[31:0].
16	CPU0_ICACHE_512X72	CPU0_ICACHE_DATA0[143:136], CPU0_ICACHE_DATA1[135:128]	[127:0]-CPU0_ICACHE_DATA0[127:64], CPU0_ICACHE_DATA1[63:0]
17	CPU0_ICACHE_128X28	CPU0_ICACHE_TAG0[55:49], CPU0_ICACHE_TAG1[48:42]	[41:0]-CPU0_ICACHE_TAG0[41:21], CPU0_ICACHE_TAG1[20:0]
18	CPU0_DCACHE0_256X39	CPU0_DCACHE0_DATA0[155:149], CPU0_DCACHE0_DATA1[148:142], CPU0_DCACHE0_DATA2[141:135], CPU0_DCACHE0_DATA3[134:128]	[127:0]-CPU0_DCACHE_DATA0[127:96], CPU0_DCACHE_DATA1[95:64], CPU0_DCACHE_DATA2[63:32], CPU0_DCACHE_DATA3[31:0]
19	CPU0_DCACHE1_256X3	CPU0_DCACHE1_DATA0[155:14	[127:0]-

Channel number	Memory of block	Check bits	ECC data bits
	9	9], CPU0_DCACHE1_DATA1[148:142], CPU0_DCACHE1_DATA2[141:135], CPU0_DCACHE1_DATA3[134:128]	CPU0_DCACHE_DATA0[127:96], CPU0_DCACHE_DATA1[95:64], CPU0_DCACHE_DATA2[63:32], CPU0_DCAsCHE_DATA3[31:0]
20	CPU0_DCACHE_64X32	CPU0_DCACHE_TAG0[127:121], CPU0_DCACHE_TAG1[120:114], CPU0_DCACHE_TAG2[113:107], CPU0_DCACHE_TAG3[106:100]	[99:0]- CPU0_DCACHE_TAG0[99:75], CPU0_DCACHE_TAG1[74:50], CPU0_DCACHE_TAG2[49:25], CPU0_DCACHE_TAG3[24:0]
27	Buffer0	[136:128]	[63:0].
29	Buffer2	[136:128]	[63:0].
30	Buffer3	[136:128]	[63:0].
31	Fast program buffer	[136:128]	[63:0].
32	Pflash0	[139:128]	[63:0].
33	Pflash1	[139:128]	[63:0].
34	Dflash0	[135:128]	[63:0].
35	Dflash1	[135:128]	[63:0].

Note: 1. ECC data generation can only invert one bit at a time. ECC data multi-bit generation can only invert two bits at a time. ECC data support the generation: 1bit ECC data, 1bit ECC checkbit, 1bit ECC data + 1bit ECC data, 1bit address + 1bit ECC data, 1bit EDC after ECC + 1bit ECC data, 2bits EDC after ECC + 1bit ECC data.

2. Eflash macro must be written before MTC generation. If eflash macro is erased but not written, MTC generation will work abnormally.

25.3.5. ECC address injection

ECC address injection is to generate single-bit or multi -bit error bits on checkbits or address, which is uncorrectable.

Table 25-2. MTC ECC address channel mapping

Channel number	Memory of block	Check bits	ECC address bits
0	SRAM0	[136:128]	[78:64]
2	CM7_0 ITCM	[136:128]	[75:64]
3	CM7_0 D0TCM	[71:64]	[44:32]
4	CM7_0 D1TCM	[71:64]	[44:32]

Channel number	Memory of block	Check bits	ECC address bits
8	MCAN_0	[71:64]	[42:32]
9	MCAN_1	[71:64]	[42:32]
10	MCAN_2	[71:64]	[42:32]
27	Buffer0	[136:128]	[69:64]
29	Buffer2	[136:128]	[69:64]
30	Buffer3	[136:128]	[69:64]
31	Fast program buffer	[136:128]	[69:64]
32	Pflash0	[139:128]	[81:64]
33	Pflash1	[139:128]	[81:64]
37	TCM CHK widget	[102:96]	[31:0]
47	QSPI CHK widget	[102:96]	[31:0]
48	SRAM0 CHK widget	[102:96]	[31:0]
50	FMC-0 CHK widget	[102:96]	[31:0]
52	FMC-2 CHK widget	[102:96]	[31:0]
55	AHB1 CHK widget	[70:64]	[31:0]
56	AHB2 CHK widget	[70:64]	[31:0]
57	APB1 CHK widget	[70:64]	[31:0]
58	APB2 CHK widget	[70:64]	[31:0]

Note: 1. ECC address injection can only invert one bit at a time. ECC address multi-bit generation can only invert two bits at a time. ECC address support the generation: 1bit address, 1bit checkbit, 1bit address + 1bit address; 1bit address + 1bit check bit.

2. Eflash macro must be written before MTC generation. If Eflash macro is erased but not written, MTC generation will work abnormally.

25.3.6. EDC data injection

EDC (Error detection code) is to generate error bits for the read/write data in BUS.

Table 25-3. MTC EDC data channel mapping

Channel number	Memory of block	Check bits	EDC data bits
37	TCM CHK widget	[110:103]	EDC wdata [95:32],
38	DMA0 BM1 GEN widget memory	[71:64]	EDC rdata [63:0]
39	DMA1 BM1 GEN widget memory	[71:64]	EDC rdata [63:0]
40	DMA0 BM1 GEN widget peripheral	[71:64]	EDC rdata [63:0]
41	DMA1 BM1 GEN widget peripheral	[71:64]	EDC rdata [63:0]
42	DMA0 BM2 GEN	[38:32]	EDC rdata [31:0]

Channel number	Memory of block	Check bits	EDC data bits
	widget memory		
43	DMA1 BM2 GEN widget memory	[38:32]	EDC rdata [31:0]
44	DMA0 BM2 GEN widget peripheral	[38:32]	EDC rdata [31:0]
45	DMA1 BM2 GEN widget peripheral	[38:32]	EDC rdata [31:0]
46	HSM BM1 GEN widget	[71:64]	EDC rdata [63:0]
47	QSPI CHK widget	[110:103]	EDC wdata [95:32],
48	SRAM0 CHK widget	[110:103]	EDC wdata [95:32],
50	FMC-0 CHK widget	[110:103]	EDC wdata [95:32],
52	FMC-2 CHK widget	[110:103]	EDC wdata [95:32],
54	HSM BM2 GEN widget	[38:32]	EDC rdata [31:0]
55	AHB1 CHK widget	[77:71]	EDC wdata [63:32]
56	AHB2 CHK widget	[77:71]	EDC wdata [63:32]
57	APB1 CHK widget	[77:71]	EDC wdata [63:32]
58	APB2 CHK widget	[77:71]	EDC wdata [63:32]
59	CM7 AXIM GEN	[71:64]	EDC rdata [63:0]
60	CM7 AHBP GEN	[38:32]	EDC rdata [31:0]

Note: 1. EDC data generation can only invert one bit at a time. EDC data multi-bit generation can only invert two bits at a time. EDC data support the generation: 1bit EDC data, 1bit checkbit, 1bit EDC data + 1bit EDC data, 1bit EDC data + 1bit check bit, 1bit EDC check bit +1bit EDC check bit.

25.3.7. EDC after ECC injection

An "EDC after ECC" check is used to verify the correct functionality of an ECC function. For this purpose, a further EDC element is inserted in the read path after the memory that observes the ECC function. Using the corrected data readback, the same steps in the above ECC process are implemented, but without the correction part. Check if the new EDC code and readback EDC code are the same. If not same, it indicates a fault in correction function. EDC after ECC error bit generation is allowed to operate simultaneously with ECC error bit generation.

An "EDC after ECC" check is used to verify the correct functionality of an ECC function. For this purpose a further EDC element is inserted in the read path after the memory that observes the ECC function.Using the corrected data readback, a set of same steps in the above ECC process only without correction part implemented, and a new EDC code generated. Check if the new EDC code and readback ECC code are the same. If not same, it indicates a fault in correction function. EDC after ECC injection and ECC injection simultaneously is allowed.

Table 25-4. MTC EDC after ECC channel mapping

Channel number	Memory of block	EDC after ECC data bits
0	SRAM0	[127:64]
2	CM7_0 ITCM	[127:64]
3	CM7_0 D0TCM	[63:32]
4	CM7_0 D1TCM	[63:32]
8	MCAN_0	[63:32]
9	MCAN_1	[63:32]
10	MCAN_2	[63:32]
27	Buffer0	[127:64]
29	Buffer2	[127:64]
30	Buffer3	[127:64]
31	Fast program buffer	[127:64]
32	Pflash0	[127:64]
33	Pflash1	[127:64]

Note: 1. EDC after ECC single-bit generation can only invert one bit at a time. EDC after ECC multi-bit generation can only invert two bits at a time. EDC after ECC support the generation: 1bit EDC after ECC, 1bit EDC after ECC + 1bit EDC after ECC; 1bit EDC after ECC + 1bit ECC data; 2bits EDC after ECC+1bit ECC data.

2. Eflash macro must be written before MTC generation. If eflash macro is erased but not written, MTC generation will work abnormally.

25.3.8. Multi-channel parallel test

MTC supports multi-channel parallel testing to meet the need to generate error bits in multiple memories. However, multi-channel parallel testing has the following requirements:

1. The length and format of check bits and data bits among channels must be the same.
2. The configurations for error bits generation among channels must be the same, that is, the channels perform bit inversion on the same bits.
3. If error bits are generated in EDC mode, the modes of EDC which are specified by EDCMOD[1:0] bits in MTC_CTL1 register among channels must be the same.

25.3.9. Memory test flow

Please strictly follow the software configuration process for setup. When error bits do not need to be generated, promptly close the single-channel enable and global enable. During the error bit generation process, apart from disabling enablement, improper operations on MTC, such as accessing reserved areas or modifying MTC_EBGCFGx after enabling it, may interrupt the error bit generation process.

The following steps show the memory test configuration sequence:

1. According to the format of data bits and check bits of the channels in MTC channel mapping, configure the bits to be inverted in the MTC_EBGCFGx register. (Note: Each time the MTC_EBGCFGx register is written, the MTC_CHCTL0 and MTC_CHCTL1 registers will be cleared to zero.)
2. According to the channel number in MTC channel mapping, configure the channel numbers to be tested in the MTC_CHCTL0 or MTC_CHCTL1 register.
3. If error bits to be generated are in ECC mode, go to step 4. If in ECC / EDC address or EDC after ECC mode, the corresponding test mode needs to be selected in the MTC_CTL1 register.
4. Write 0xA55A in GMTEN[15:0] bits of MTC_CTL0 register to enable global memory test.

For example, SRAM0 is going to generate an ECC error in data[0]. And MTC_EBGCFG0 should be configured as 32'h00000001, and MTC_CHCTL0 should be 32'h00000001, Then MTC_CTL0 is 32'h0000A55A which is enabled.

When the configuration is completed, MTC will generate error bits on the specific module.

Note:

1.Please strictly follow the software process for configuration. When error injection is not required, promptly disable single-channel enablement and global enablement. During the error injection process, improper operations on MTC, such as accessing reserved areas or modifying MTC_EBGCFGx after enabling, may interrupt the error injection process.

2.The error injection process only affects the data read and does not impact the actual memory data.

3.When the program is running on the memory corresponding to a specific channel, do not perform error injection on the MTC of that channel, as it may cause unpredictable read errors and affect program execution.

25.4. Register definition

MTC base address: 0x4002 4400

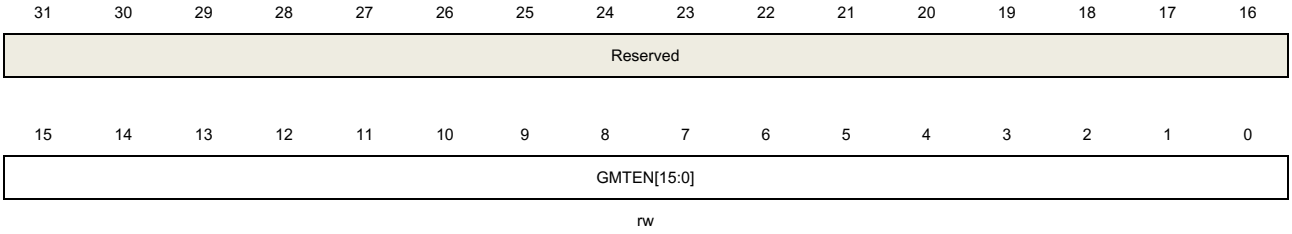
Table 25-5. MTC register overview

Register name	Symbol	Address offset	Reset value
Control register 0	MTC_CTL0	0x00	0x0000 0000
Channel control register 0	MTC_CHCTL0	0x04	0x0000 0000
Channel control register 1	MTC_CHCTL1	0x08	0x0000 0000
Control register 1	MTC_CTL1	0x0C	0x0000 0000
Error bits generate configuration register x	MTC_EBGCFGx	0x10 + 0x4 * x(x = 0…4)	0x0000 0000
MTC Triple Modular Redundancy enable register	MTC_TMREN	0x24	0x0000 0000
MTC Triple Modular Redundancy Status register	MTC_TMRSTAT	0x28	0x0000 0000
MTC Triple Modular Redundancy Flag Clear register	MTC_TMRFC	0x2C	0x0000 0000

25.4.1. Control register 0 (MTC_CTL0)

Address offset: 0x00
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value.
15:0	GMTEN[15:0]	Error injection enable Error injection is enable only after MTC_CHCTLx and MTC_EBGCFGx registers have been configured. When the bits are set to 0xA55A by the software, memory test is enabled. Any value except 0xA55A will disable memory test.

25.4.2. Channel control register 0 (MTC_CHCTL0)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CHEN31	CHEN30	CHEN29	CHEN28	CHEN27	CHEN26	CHEN25	CHEN24	CHEN23	CHEN22	CHEN21	CHEN20	CHEN19	CHEN18	CHEN17	CHEN16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHEN15	CHEN14	CHEN13	CHEN12	CHEN11	CHEN10	CHEN9	CHEN8	CHEN7	CHEN6	CHEN5	CHEN4	CHEN3	CHEN2	CHEN1	CHEN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:0	CHENx	Error Injection Channel enable bit x (x=0…31) 0: Disable error injection channel x 1: Enable error injection channel x These bits enable the corresponding memory test channel. After configuring this register, you must also enable global memory test function in the MTC_CTL0 register to enable memory test. The memory test remains valid until the memory test channel is disabled. Any write to the MTC_EBGCFGx register clears all the CHENx bits and disables the error bit generation channel.

25.4.3. Channel control register 1 (MTC_CHCTL1)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			CHEN60	CHEN59	CHEN58	CHEN57	CHEN56	CHEN55	CHEN54	CHEN53	CHEN52	CHEN51	CHEN50	CHEN49	CHEN48
			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHEN47	CHEN46	CHEN45	CHEN44	CHEN43	CHEN42	CHEN41	CHEN40	CHEN39	CHEN38	CHEN37	CHEN36	CHEN35	CHEN34	CHEN33	CHEN32
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:29	Reserved	Must be kept at reset value.
28:0	CHENx	Error injection Channel enable bit x(x=32…60) 0: Disable error injection channel x 1: Enable error injection channel x These bits enable the corresponding memory test channel. After configuring this register, you must also enable global memory test function in the MTC_CTL0

register to enable memory test. The memory test remains valid until the memory test channel is disabled.

Any write to the MTC_EBGCFGx register clears all the CHENx bits and disables the error bit generation channel.

25.4.4. Control register 1 (MTC_CTL1)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														EDCMOD[1:0]	
														rw	

Bits	Fields	Descriptions
31:2	Reserved	Must be kept at reset value.
1:0	EDCMOD[1:0]	Error injection mode selection These bits are only used for channels that have ECC / EDC address or EDC after ECC test requirements, and are invalid when channels have only ECC/EDC data test requirements. These bits select the EDC error bits generation mode in MTC channel mapping. 01: Error bits generation of ECC / EDC address. 10: Error bits generation of EDC after ECC. Others: No error bits are generated in ECC / EDC address or EDC after ECC mode.

25.4.5. Error bits generate configuration register x (MTC_EBGCFGx) (x = 0…4)

Address offset: 0x10 + 0x4 * x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBG31	EBG30	EBG29	EBG28	EBG27	EBG26	EBG25	EBG24	EBG23	EBG22	EBG21	EBG20	EBG19	EBG18	EBG17	EBG16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EBG15	EBG14	EBG13	EBG12	EBG11	EBG10	EBG9	EBG8	EBG7	EBG6	EBG5	EBG4	EBG3	EBG2	EBG1	EBG0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:0	EBGy	Error bits generate configuration bit

Refer to the sequence of check bits and data bits in MTC channel mapping, where MTC_EBGCFG0 corresponds to the lower 32 bits of data bits, and so on.

If it is set, corresponding bit y+32*x will generate error bit when memory test.

0: Do not invert the corresponding memory output data bit.

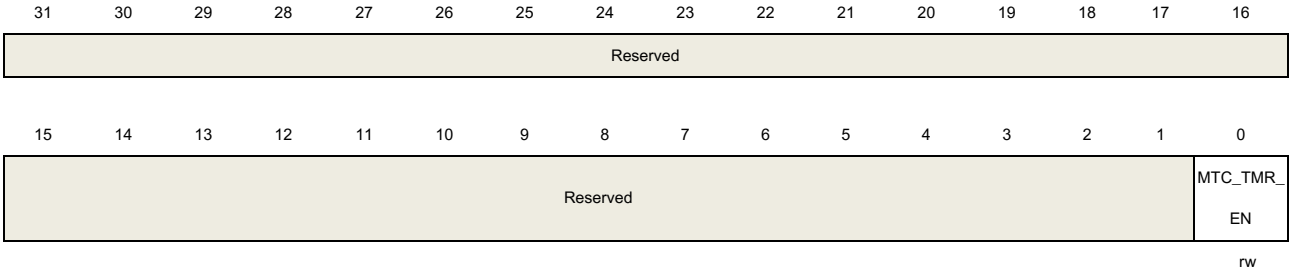
1: Invert the corresponding memory output data bit.

25.4.6. MTC Triple Modular Redundancy enable register (MTC_TMREN)

Address offset: 0x24

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



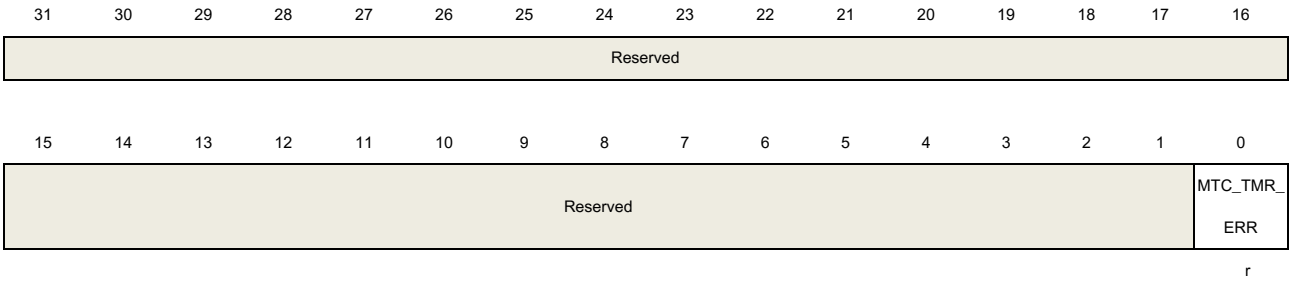
Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	MTC_TMR_EN	MTC TMR enable bit 0: Disable MTC triple modular redundancy fault to FMU reaction. 1: Enable MTC triple modular redundancy fault to FMU reaction.

25.4.7. MTC Triple Modular Redundancy Status register (MTC_TMRSTAT)

Address offset: 0x28

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	MTC_TMR_ERR	MTC TMR error flag bit. 0: MTC TMR error is not generated.

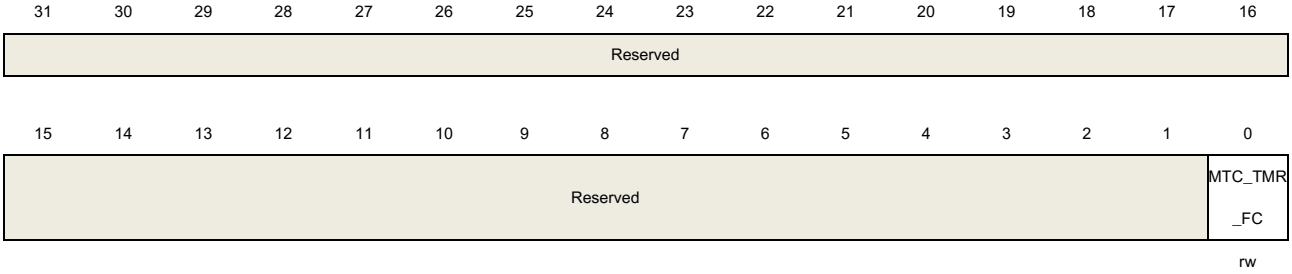
1: MTC TMR error is generated.

25.4.8. MTC Triple Modular Redundancy Flag Clear register (MTC_TMRFC)

Address offset: 0x2C

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	MTC_TMR_FC	Clear MTC TMR error flag. 0: No effect. 1: Clear MTC TMR error flag.

26. Interrupt requests management (IRM)

26.1. Overview

Cortex®-M7 integrates the Nested Vectored Interrupt Controller (NVIC) for efficient exception and interrupts processing. NVIC facilitates low-latency exception and interrupt handling and controls power management. It's tightly coupled to the processer core. For more details about the NVIC, please refer to the technical reference manual of Cortex®-M7.

The IRM module has four monitors, and each monitor can monitor the response time after an interrupt request to ensure that the response time of critical interrupts is within a set threshold to increase the reliability of the device. Also interrupt requests can be either routed to a subset or to all the cores by routing controller of IRM.

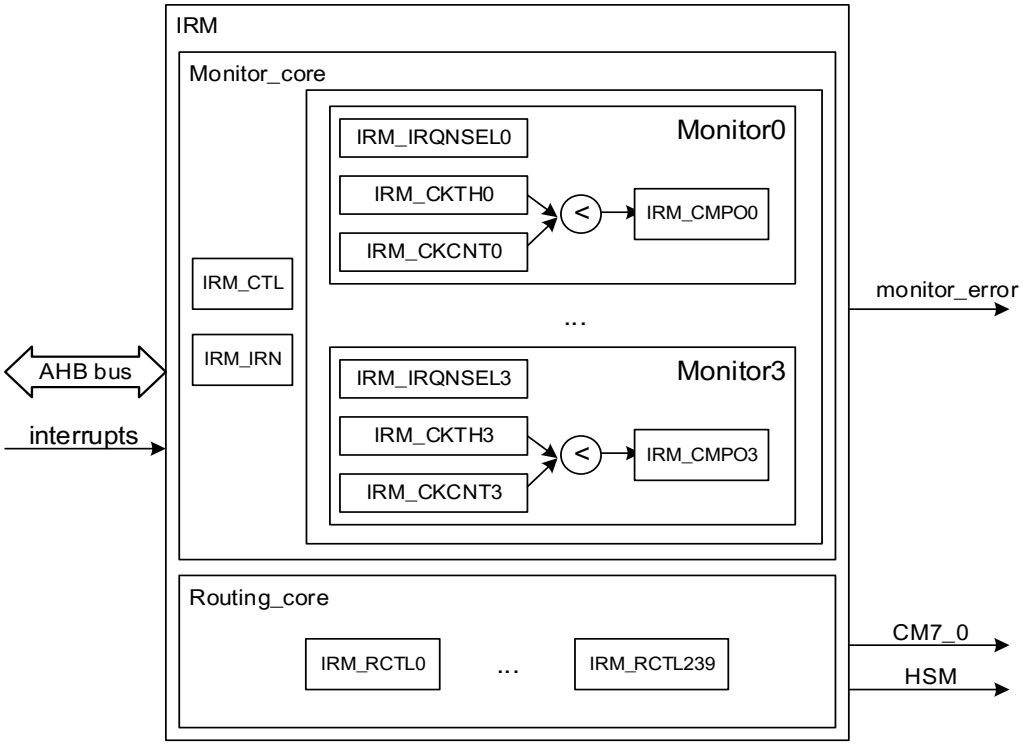
26.2. Characteristics

- Cortex®-M7 system exception.
- Up to 240 maskable peripheral interrupts.
- 4 bits interrupt priority configuration-16 priority levels.
- Efficient interrupt processing.

- Support exception pre-emption and tail-chaining.
- Up to 4 programmable interrupt monitors and 240 interrupts can be routed.
- Programmable interrupt source per monitor and routing controller.
- Programmable 24-bit timer for monitoring the maximum response latency of the per interrupt monitor.
- Support timer expiration comparison function for each interrupt monitor
- One interrupt response for all monitors.
- Routing controller be used to route interrupt requests to a subset or to all the cores.

26.3. Block Diagram

Figure 26-1.block diagram



The *Figure 26-1.block diagram* is the block diagram of IRM. IRM includes one monitor control register IRM_CTL, one interrupt response number register IRM_IRN and four monitors with their corresponding registers IRM_IRQNSELx, IRM_CKTHx, IRM_CKCNTx and IRM_CMPOx, x=0,1,2,3. By configuring the corresponding register, configure the interrupt source, the maximum interrupt response latency and the expired comparison of the timer per monitor. CKCNT[23:0] is used to count the latency of the response on interrupt request per monitor, and the value can be read by IRM_CKCNTx (x=0,1,2,3) register. IRM interacts with data via the AHB bus. IRM monitor detected error is sent to the FMU.

Also, IRM has one interrupt requests routing controller. By configuring each core enable bit of IRM_RCTLx register(x=0..239), each interrupt request can be either routed to a subset or to all the cores. IRM_RCTLx register has register lock protection, When LOCK bit of IRM_RCTLx register is set, the IRM_RCTLx will be locked and writing to IRM_RCTLx will be ignored until the next reset writes 0 to LOCK bit. Routing controller also check the IRM_RCTLx register data size and the bus master id. This register can only be accessed by half a word (16 bits) ,any other access will terminate with a system bus error. Only privileged mode accesses, and privileged accesses from non-core (and non-debug) bus masters are treated as read all zero or writing ignore.

26.4. Function overview

The Arm® Cortex®-M7 processor and the Nested Vectored Interrupt Controller (NVIC) prioritize and handle all exceptions in Handler Mode. The processor state is automatically stored to the stack on an exception and automatically restored from the stack at the end of the Interrupt Service Routine (ISR).

The vector is fetched in parallel to the state saving, enabling efficient interrupt entry. The processor supports tail-chaining, which enables back-to-back interrupts to be performed without the overhead of state saving and restoration. The following tables list all exception types.

Table 26-1. NVIC exception types in Cortex®-M7

Exception Type	Vector Number	Priority (a)	Vector Address	Description
-	0	-	0x0000_0000	Reserved
Reset	1	-3	0x0000_0004	Reset
NMI	2	-2	0x0000_0008	Non maskable interrupt.
HardFault	3	-1	0x0000_000C	All class of fault
MemManage	4	Programmable	0x0000_0010	Memory management
BusFault	5	Programmable	0x0000_0014	Prefetch fault, memory access fault
UsageFault	6	Programmable	0x0000_0018	Undefined instruction or illegal state
-	7-10	-	0x0000_001C - 0x0000_002B	Reserved
SVCall	11	Programmable	0x0000_002C	System service call via SWI instruction
Debug Monitor	12	Programmable	0x0000_0030	Debug Monitor
-	13	-	0x0000_0034	Reserved
PendSV	14	Programmable	0x0000_0038	Pendable request for system service
SysTick	15	Programmable	0x0000_003C	System tick timer

Table 26-2. GD32A711/A712 interrupt vector table

Interrupt Number	Vector Number	Peripheral Interrupt Description	Vector Address
IRQ 0	16	WWDGT0 interrupt	0x0000_0040
IRQ 1	17	LVD/HVD through EXTI Line detection interrupt	0x0000_0044
IRQ 2	18	RTC alarm through EXTI Line detection interrupt	0x0000_0048
IRQ 3	19	RTC global Interrupt	0x0000_004C
IRQ 4	20	FMC global interrupt	0x0000_0050
IRQ 5	21	RCU clock ready interrupt	0x0000_0054
IRQ 6	22	EXTI Line0 interrupt	0x0000_0058

Interrupt Number	Vector Number	Peripheral Interrupt Description	Vector Address
IRQ 7	23	EXTI Line1 interrupt	0x0000_005C
IRQ 8	24	EXTI Line2 interrupt	0x0000_0060
IRQ 9	25	EXTI Line3 interrupt	0x0000_0064
IRQ 10	26	EXTI Line4 interrupt	0x0000_0068
IRQ 11	27	DMA0 Channel0 global interrupt	0x0000_006C
IRQ 12	28	DMA0 Channel1 global interrupt	0x0000_0070
IRQ 13	29	DMA0 Channel2 global interrupt	0x0000_0074
IRQ 14	30	DMA0 Channel3 global interrupt	0x0000_0078
IRQ 15	31	DMA0 Channel4 global interrupt	0x0000_007C
IRQ 16	32	DMA0 Channel5 global interrupt	0x0000_0080
IRQ 17	33	DMA0 Channel6 global interrupt	0x0000_0084
IRQ 18	34	ADC0 interrupt	0x0000_0088
IRQ 19	35	DTM_CAN0 Interrupt0	0x0000_008C
IRQ 20	36	DTM_CAN0 Interrupt1	0x0000_0090
IRQ 21	37	Reserved	0x0000_0094
IRQ 22	38	Reserved	0x0000_0098
IRQ 23	39	Reserved	0x0000_009C
IRQ 24	40	Reserved	0x0000_00A0
IRQ 25	41	DTM_CAN0 DMU interrupt	0x0000_00A4
IRQ 26	42	TIMER0 Break, update, trigger and commutation, dec interrupt	0x0000_00A8
IRQ 27	43	TIMER2 update, trigger and commutation, dec interrupt	0x0000_00AC
IRQ 28	44	TIMER1 update, trigger and commutation, dec interrupt	0x0000_00B0
IRQ 29	45	TIMER19 Break, update, trigger and commutation, dec interrupt	0x0000_00B4
IRQ 30	46	JDC interrupt to host	0x0000_00B8
IRQ 31	47	I2C0 event interrupt	0x0000_00BC
IRQ 32	48	I2C0 error interrupt	0x0000_00C0
IRQ 33	49	I2C1 event interrupt	0x0000_00C4
IRQ 34	50	I2C1 error interrupt	0x0000_00C8
IRQ 35	51	SPI0 global interrupt	0x0000_00CC
IRQ 36	52	SPI1 global interrupt	0x0000_00D0
IRQ 37	53	Reserved	0x0000_00D4
IRQ 38	54	Reserved	0x0000_00D8
IRQ 39	55	Reserved	0x0000_00DC
IRQ 40	56	EXTI line[15:10] interrupts	0x0000_00E0
IRQ 41	57	EXTI line[9:5] interrupts	0x0000_00E4
IRQ 42	58	BKP tamper interrupt	0x0000_00E8
IRQ 43	59	TIMER20 Break, update, trigger and	0x0000_00EC

Interrupt Number	Vector Number	Peripheral Interrupt Description	Vector Address
		commutation, dec interrupt	
IRQ 44	60	EXTI line[101:42], EXTI line[18] interrupts	0x0000_00F0
IRQ 45	61	TIMER7 Break, update, trigger and commutation, dec interrupt	0x0000_00F4
IRQ 46	62	PLL LOL system reset interrupt	0x0000_00F8
IRQ 47	63	DMAMUX overrun interrupt	0x0000_00FC
IRQ 48	64	CPU FPU interrupt or Write abort interrupt	0x0000_0100
IRQ 49	65	CMP0 through EXTI Line detection interrupt	0x0000_0104
IRQ 50	66	CMP1 through EXTI Line detection interrupt	0x0000_0108
IRQ 51	67	FMU initializing timeout interrupt	0x0000_010C
IRQ 52	68	ADC1 interrupt	0x0000_0110
IRQ 53	69	Reserved	0x0000_0114
IRQ 54	70	DAC0 global interrupt	0x0000_0118
IRQ 55	71	STCM illegal triggered during running state	0x0000_011C
IRQ 56	72	DMA1 Channel0 global interrupt	0x0000_0120
IRQ 57	73	DMA1 Channel1 global interrupt	0x0000_0124
IRQ 58	74	DMA1 Channel2 global interrupt	0x0000_0128
IRQ 59	75	DMA1 Channel3 global interrupt	0x0000_012C
IRQ 60	76	DMA1 Channel4 global interrupt	0x0000_0130
IRQ 61	77	Reserved	0x0000_0134
IRQ 62	78	DTM_CAN1 interrupt0	0x0000_0138
IRQ 63	79	DTM_CAN1 Interrupt1	0x0000_013C
IRQ 64	80	DTM_CAN1 DMU Interrupt	0x0000_0140
IRQ 65	81	CTI Interrupt 0	0x0000_0144
IRQ 66	82	CTI Interrupt 1	0x0000_0148
IRQ 67	83	CAN0 ECC error interrupt	0x0000_014C
IRQ 68	84	CAN1 ECC error interrupt	0x0000_0150
IRQ 69	85	CAN2 ECC error interrupt	0x0000_0154
IRQ 70	86	Reserved	0x0000_0158
IRQ 71	87	CMP0 global interrupt	0x0000_015C
IRQ 72	88	CMP1 global interrupt	0x0000_0160
IRQ 73	89	Reserved	0x0000_0164
IRQ 74	90	Reserved	0x0000_0168
IRQ 75	91	Reserved	0x0000_016C
IRQ 76	92	Reserved	0x0000_0170
IRQ 77	93	Reserved	0x0000_0174
IRQ 78	94	DAC1 global interrupt	0x0000_0178
IRQ 79	95	HSM communication interrupt0	0x0000_017C
IRQ 80	96	HSM communication interrupt1	0x0000_0180
IRQ 81	97	FMU cation state interrupt request	0x0000_0184
IRQ 82	98	RCTL interrupt	0x0000_0188

Interrupt Number	Vector Number	Peripheral Interrupt Description	Vector Address
IRQ 83	99	Reserved	0x0000_018C
IRQ 84	100	SPI2 global interrupt	0x0000_0190
IRQ 85	101	SPI3 global interrupt	0x0000_0194
IRQ 86	102	SPI4 global interrupt	0x0000_0198
IRQ 87	103	SPI5 global interrupt	0x0000_019C
IRQ 88	104	Reserved	0x0000_01A0
IRQ 89	105	Reserved	0x0000_01A4
IRQ 90	106	QSPI global interrupt	0x0000_01A8
IRQ 91	107	Reserved	0x0000_01AC
IRQ 92	108	Reserved	0x0000_01B0
IRQ 93	109	Reserved	0x0000_01B4
IRQ 94	110	DTM_CAN2 DMU interrupt	0x0000_01B8
IRQ 95	111	DTM_CAN3 DMU interrupt	0x0000_01BC
IRQ 96	112	DTM_CAN4 DMU interrupt	0x0000_01C0
IRQ 97	113	DTM_CAN5 DMU interrupt	0x0000_01C4
IRQ 98	114	DTM_CAN6 DMU interrupt	0x0000_01C8
IRQ 99	115	DTM_CAN7 DMU interrupt	0x0000_01CC
IRQ 100	116	DTM_CAN2 Interrupt0	0x0000_01D0
IRQ 101	117	DTM_CAN2 Interrupt1	0x0000_01D4
IRQ 102	118	DTM_CAN3 Interrupt0	0x0000_01D8
IRQ 103	119	DTM_CAN3 Interrupt1	0x0000_01DC
IRQ 104	120	DTM_CAN4 Interrupt0	0x0000_01E0
IRQ 105	121	DTM_CAN4 Interrupt1	0x0000_01E4
IRQ 106	122	DTM_CAN5 Interrupt0	0x0000_01E8
IRQ 107	123	DTM_CAN5 Interrupt1	0x0000_01EC
IRQ 108	124	DTM_CAN6 Interrupt0	0x0000_01F0
IRQ 109	125	DTM_CAN6 Interrupt1	0x0000_01F4
IRQ 110	126	DTM_CAN7 Interrupt0	0x0000_01F8
IRQ 111	127	DTM_CAN7 Interrupt1	0x0000_01FC
IRQ 112	128	TIMER19 capture compare CH0 and MCH0 interrupt	0x0000_0200
IRQ 113	129	TIMER19 capture compare CH1 and MCH1 interrupt	0x0000_0204
IRQ 114	130	TIMER19 capture compare CH2 and MCH2 interrupt	0x0000_0208
IRQ 115	131	TIMER19 capture compare CH3 and MCH3 interrupt	0x0000_020C
IRQ 116	132	TIMER20 capture compare CH0 and MCH0 interrupt	0x0000_0210
IRQ 117	133	TIMER20 capture compare CH1 and MCH1 interrupt	0x0000_0214

Interrupt Number	Vector Number	Peripheral Interrupt Description	Vector Address
IRQ 118	134	TIMER20 capture compare CH2 and MCH2 interrupt	0x0000_0218
IRQ 119	135	TIMER20 capture compare CH3 and MCH3 interrupt	0x0000_021C
IRQ 120	136	TIMER60 capture compare CH0 and MCH0 interrupt	0x0000_0220
IRQ 121	137	TIMER60 capture compare CH1 and MCH1 interrupt	0x0000_0224
IRQ 122	138	TIMER60 capture compare CH2 and MCH2 interrupt	0x0000_0228
IRQ 123	139	TIMER60 capture compare CH3 and MCH3 interrupt	0x0000_022C
IRQ 124	140	TIMER61 capture compare CH0 and MCH0 interrupt	0x0000_0230
IRQ 125	141	TIMER61 capture compare CH1 and MCH1 interrupt	0x0000_0234
IRQ 126	142	TIMER61 capture compare CH2 and MCH2 interrupt	0x0000_0238
IRQ 127	143	TIMER61 capture compare CH3 and MCH3 interrupt	0x0000_023C
IRQ 128	144	TIMER62 capture compare CH0 and MCH0 interrupt	0x0000_0240
IRQ 129	145	TIMER62 capture compare CH1 and MCH1 interrupt	0x0000_0244
IRQ 130	146	TIMER62 capture compare CH2 and MCH2 interrupt	0x0000_0248
IRQ 131	147	TIMER62 capture compare CH3 and MCH3 interrupt	0x0000_024C
IRQ 132	148	TIMER63 capture compare CH0 and MCH0 interrupt	0x0000_0250
IRQ 133	149	TIMER63 capture compare CH1 and MCH1 interrupt	0x0000_0254
IRQ 134	150	TIMER63 capture compare CH2 and MCH2 interrupt	0x0000_0258
IRQ 135	151	TIMER63 capture compare CH3 and MCH3 interrupt	0x0000_025C
IRQ 136	152	DMA0 Channel7 global interrupt	0x0000_0260
IRQ 137	153	Reserved	0x0000_0264
IRQ 138	154	Reserved	0x0000_0268
IRQ 139	155	Reserved	0x0000_026C
IRQ 140	156	Reserved	0x0000_0270

Interrupt Number	Vector Number	Peripheral Interrupt Description	Vector Address
IRQ 141	157	Reserved	0x0000_0274
IRQ 142	158	Reserved	0x0000_0278
IRQ 143	159	Reserved	0x0000_027C
IRQ 144	160	Reserved	0x0000_0280
IRQ 145	161	DMA1 Channel5 global interrupt	0x0000_0284
IRQ 146	162	DMA1 Channel6 global interrupt	0x0000_0288
IRQ 147	163	DMA1 Channel7 global interrupt	0x0000_028C
IRQ 148	164	Reserved	0x0000_0290
IRQ 149	165	Reserved	0x0000_0294
IRQ 150	166	Reserved	0x0000_0298
IRQ 151	167	Reserved	0x0000_029C
IRQ 152	168	Reserved	0x0000_02A0
IRQ 153	169	Reserved	0x0000_02A4
IRQ 154	170	Reserved	0x0000_02A8
IRQ 155	171	Reserved	0x0000_02AC
IRQ 156	172	Reserved	0x0000_02B0
IRQ 157	173	Reserved	0x0000_02B4
IRQ 158	174	Reserved	0x0000_02B8
IRQ 159	175	Reserved	0x0000_02BC
IRQ 160	176	Reserved	0x0000_02C0
IRQ 161	177	Reserved	0x0000_02C4
IRQ 162	178	LINFlexD0 interrupt	0x0000_02C8
IRQ 163	179	LINFlexD1 interrupt	0x0000_02CC
IRQ 164	180	LINFlexD2 interrupt	0x0000_02D0
IRQ 165	181	LINFlexD3 interrupt	0x0000_02D4
IRQ 166	182	LINFlexD4 interrupt	0x0000_02D8
IRQ 167	183	LINFlexD5 interrupt	0x0000_02DC
IRQ 168	184	Reserved	0x0000_02E0
IRQ 169	185	Reserved	0x0000_02E4
IRQ 170	186	Reserved	0x0000_02E8
IRQ 171	187	Reserved	0x0000_02EC
IRQ 172	188	Reserved	0x0000_02F0
IRQ 173	189	Reserved	0x0000_02F4
IRQ 174	190	TIMER0 Capture Compare ch0 and mch0 interrupt	0x0000_02F8
IRQ 175	191	TIMER0 Capture Compare ch1 and mch1 interrupt	0x0000_02FC
IRQ 176	192	TIMER0 Capture Compare ch2 and mch2 interrupt	0x0000_0300
IRQ 177	193	TIMER0 Capture Compare ch3 and mch3 interrupt	0x0000_0304

Interrupt Number	Vector Number	Peripheral Interrupt Description	Vector Address
IRQ 178	194	TIMER7 Capture Compare ch0 and mch0 interrupt	0x0000_0308
IRQ 179	195	TIMER7 Capture Compare ch1 and mch1 interrupt	0x0000_030C
IRQ 180	196	TIMER7 Capture Compare ch2 and mch2 interrupt	0x0000_0310
IRQ 181	197	TIMER7 Capture Compare ch3 and mch3 interrupt	0x0000_0314
IRQ 182	198	TIMER1 Capture Compare ch0 interrupt	0x0000_0318
IRQ 183	199	TIMER1 Capture Compare ch1 interrupt	0x0000_031C
IRQ 184	200	TIMER1 Capture Compare ch2 interrupt	0x0000_0320
IRQ 185	201	TIMER1 Capture Compare ch3 interrupt	0x0000_0324
IRQ 186	202	TIMER2 Capture Compare ch0 interrupt	0x0000_0328
IRQ 187	203	TIMER2 Capture Compare ch1 interrupt	0x0000_032C
IRQ 188	204	TIMER2 Capture Compare ch2 interrupt	0x0000_0330
IRQ 189	205	TIMER2 Capture Compare ch3 interrupt	0x0000_0334
IRQ 190	206	FWDGT0 reset interrupt through EXTI Line detection interrupt	0x0000_0338
IRQ 191	207	Reserved	0x0000_033C
IRQ 192	208	Reserved	0x0000_0340
IRQ 193	209	Reserved	0x0000_0344
IRQ 194	210	Reserved	0x0000_0348
IRQ 195	211	Reserved	0x0000_034C
IRQ 196	212	RCU clock fail interrupt	0x0000_0350
IRQ 197	213	Reserved	0x0000_0354
IRQ 198	214	Reserved	0x0000_0358
IRQ 199	215	Reserved	0x0000_035C
IRQ 200	216	Reserved	0x0000_0360
IRQ 201	217	Reserved	0x0000_0364
IRQ 202	218	Reserved	0x0000_0368
IRQ 203	219	Reserved	0x0000_036C
IRQ 204	220	Reserved	0x0000_0370
IRQ 205	221	Reserved	0x0000_0374
IRQ 206	222	Reserved	0x0000_0378
IRQ 207	223	Reserved	0x0000_037C
IRQ 208	224	Reserved	0x0000_0380
IRQ 209	225	Reserved	0x0000_0384
IRQ 210	226	TIMER60 Break, update, trigger and commutation, dec interrupt	0x0000_0388
IRQ 211	227	TIMER61 Break, update, trigger and commutation, dec interrupt	0x0000_038C

Interrupt Number	Vector Number	Peripheral Interrupt Description	Vector Address
IRQ 212	228	TIMER62 Break, update, trigger and commutation, dec interrupt	0x0000_0390
IRQ 213	229	TIMER63 Break, update, trigger and commutation, dec interrupt	0x0000_0394
IRQ 214	230	TIMER5 channel0 update interrupt	0x0000_0398
IRQ 215	231	TIMER5 channel1 update interrupt	0x0000_039C
IRQ 216	232	TIMER5 channel2 update interrupt	0x0000_03A0
IRQ 217	233	TIMER5 channel3 update interrupt	0x0000_03A4
IRQ 218	234	TIMER6 channel0 update interrupt	0x0000_03A8
IRQ 219	235	TIMER6 channel1 update interrupt	0x0000_03AC
IRQ 220	236	TIMER6 channel2 update interrupt	0x0000_03B0
IRQ 221	237	TIMER6 channel3 update interrupt	0x0000_03B4
IRQ 222	238	Reserved	0x0000_03B8
IRQ 223	239	Reserved	0x0000_03BC
IRQ 224	240	Reserved	0x0000_03C0
IRQ 225	241	Reserved	0x0000_03C4
IRQ 226	242	Reserved	0x0000_03C8
IRQ 227	243	Reserved	0x0000_03CC
IRQ 228	244	Reserved	0x0000_03D0
IRQ 229	245	Reserved	0x0000_03D4
IRQ 230	246	Reserved	0x0000_03D8
IRQ 231	247	Reserved	0x0000_03DC
IRQ 232	248	Reserved	0x0000_03E0
IRQ 233	249	Reserved	0x0000_03E4
IRQ 234	250	Reserved	0x0000_03E8
IRQ 235	251	Reserved	0x0000_03EC
IRQ 236	252	Reserved	0x0000_03F0
IRQ 237	253	Reserved	0x0000_03F4
IRQ 238	254	Reserved	0x0000_03F8
IRQ 239	255	Reserved	0x0000_03FC

IRM allows monitoring the interrupt response latency on four interrupt requests to ensure that the process of critical interrupts is executed correctly within the expected time frame.

Each monitor performs its own registers configuration. Each monitor is configured as follows.

1. IRM_IRNSELx and IRM_CKTHx (x=0,1,2,3,corresponding to 4 monitors respectively) registers are first configured to select the interrupt source to be monitored and the maximum interrupt response latency.

   NOTE: Only defined interrupt sources can be monitored. For the interrupts in the interrupt vector table, IRQ48, IRQ65 and IRQ66 are not supported monitored. All other interrupts in the interrupt vector table are supported to be monitored. IRQ21~24 in the

corresponding direction are supported to be monitored, according to the direction selection register in IRM_IRQNSELx, x=0~3.

2. MOEN of IRM_CTL register is configured to enable the interrupt monitor.
3. CLKCNT[23:0] of IRM_CKCNTx (x=0,1,2,3) register is used to count the clock cycles between detecting an interrupt request and interrupt response, it is compared with the threshold value in IRM_CKTHx register. As long as IRM_CKTHx register is not 0 and the monitor error signal is low and the CLKCNT[23:0] is enabled, the CLKCNT[23:0] is incremented. When an interrupt request edge is detected, if IRM_CKTHx register is not 0 and the monitor error signal (monitor_error) is low, the IRM_CKCNTx will be reset to 1.
4. IRM_IRN register is configured as the interrupt source number of the interrupt response that occurs in the interrupt service routine.
5. If the interrupt response latency exceeds the set latency value, a monitor error is triggered, IRM_CKCNTx(x=0,1,2,3) stop counting, and the error signal (monitor_error) is sent to the fault monitor unit (FMU). And the status of the monitor monitoring to the error will be written to IRM_CMPOx register. By reading IRM_CMPOx register, the error information can be obtained.
6. The monitor status can be cleared by clearing the corresponding IRM_CKCNTx register.

   NOTE: When the source is not known its recommended to clear all IRM_CKCNTx.

Conditions for CLKCNT[23:0] enabling:

- Set MOEN bit to 1, and the rising edge of the request of the corresponding interrupt source selected by the monitor is captured.

Conditions for CLKCNT[23:0] disabling:

- System reset
- Set MOEN bit to 1, and the interrupt response time exceeded the threshold, a monitor error occurred.
- Set MOEN bit to 1, and the request of the corresponding interrupt source selected by the monitor is not asserted.
- The IRQRN[7:0] bits of the IRM_IRN register are consistent with the interrupt request number of the IRM_IRNSELx register, or the peripheral interrupt flag bits are cleared.

NOTE: Once CLKCNT[23:0] is enabled, set MOEN bit to 0 will not disable the CLKCNT[23:0].

26.5. IRM registers

IRM base address: 0x4002 3800

Table 26-3. IRM register overview

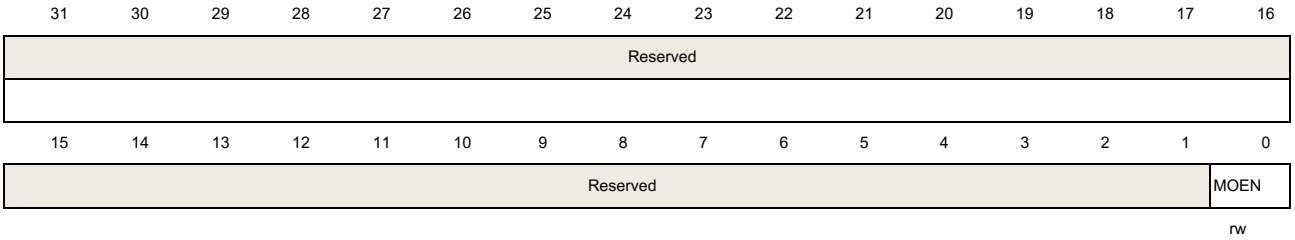
Register name	Symbol	Address offset	Reset value
Control register	IRM_CTL	0x00	0x0000 0000
Interrupt response number register	IRM_IRN	0x04	0x0000 00FF
Interrupt request number select register	IRM_IRQNSELx	0x08	0x0000 0000
Clock threshold register	IRM_CKTHx	0x0C	0x0000 0000
Clock counter register	IRM_CKCNTx	0x10	0x0000 0000
Comparison output register	IRM_CMPOx	0x14	0x0000 0000
Routing control register	IRM_RCTLx	0x48	0x0000 0005

26.5.1. Control register (IRM_CTL)

Address offset: 0x00

Reset value: 0x0000 0000

This register can be accessed by word (32-bit)



Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	MOEN	Monitor enable 0: Monitors disable 1: Monitors enable The MOEN bit controls whether the IRM monitors the latency of an interrupt response.

26.5.2. Interrupt response number register (IRM_IRN)

Address offset: 0x04

Reset value: 0x0000 00FF

This register has to be accessed by word (32-bit).



Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								IRQRN[7:0]							
								w							

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	IRQRN[7:0]	The number for interrupt response from the interrupt service routine. This write operation must provide the number of the processed interrupt, which is compared with the interrupt request number in each IRM_IRNSELx register and stops the corresponding CKCNTx when it found a match.

26.5.3. Interrupt request number select register (IRM_IRQNSELx) (x=0,1,2,3)

For GD32A711/A712 devices

Address offset: 0x08 + 0x10 * x
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								IRQSELN[7:0]							
								rw							

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7:0	IRQSELN[7:0]	The selected interrupt request number to monitor.

26.5.4. Clock threshold register (IRM_CKTHx) (x=0,1,2,3)

Address offset: 0x0C + 0x10 * x
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								MCT[23:16]							
								rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCT[15:0]															

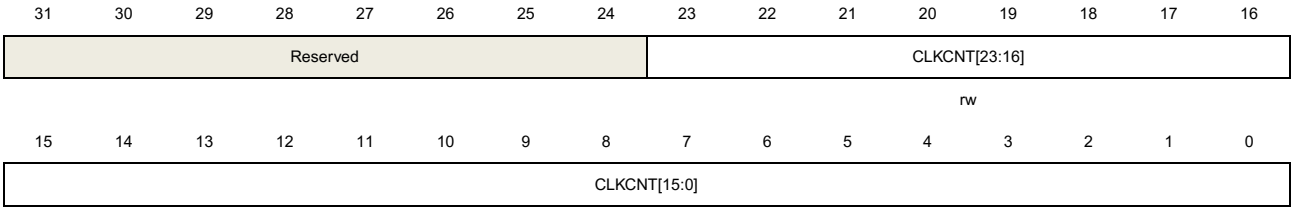
rw

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:0	MCT[23:0]	The maximum clock threshold from request to interrupt response for the selected monitored interrupt. Maximum programmed value should be no more than 0xFFFFFD to allow for proper timer error capture.

26.5.5. Clock counter register (IRM_CKCNTx) (x=0,1,2,3)

Address offset: 0x10 + 0x10 * x
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

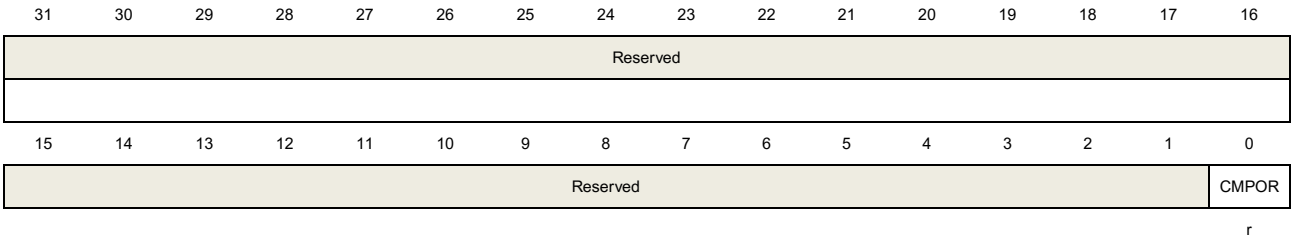


Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:0	CLKCNT[23:0]	IRM clock cycle counter. The counter value of the clock cycles between detected interrupt request and interrupt response.

26.5.6. Comparison output register (IRM_CMPOx) (x=0,1,2,3)

Address offset: 0x14 + 0x10 * x
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.

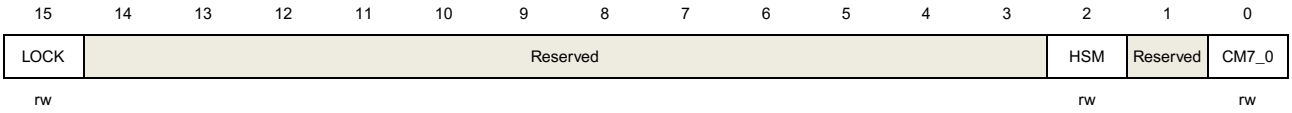
0	CMPOR	Comparison between the set interrupt response threshold of IRM_CKTH and the counter value of IRM_CKCNT. 0: IRM_CLKCNT value has not exceeded the IRM_CKTH value. 1: IRM_ CLKCNT value has exceeded the IRM_CKTH value.

26.5.7. Routing control register (IRM_RCTLx) (x=0..239)

Address offset: 0x4002 3848 + 0x2 * x

Reset value: 0x0000 0005

This register has to be accessed by halfword (16-bit).



Bits	Fields	Descriptions
15	LOCK	Lock the routing of the corresponding interrupt request. when you write 1 to this field, writes to IRM_RCTLx are invalid until the next reset writes 0 to the field. 0: Writes to IRM_RCTLx allowed 1: Writes to IRM_RCTLx ignored
14:3	Reserved	Must be kept at reset value.
2	HSM	HSM Cortex-M23 core Interrupt routing enable Enable or disable the corresponding interrupt request to route to HSM Cortex-M23 core. 0: Routing disabled 1: Routing enabled
1	Reserved	Must be kept at reset value.
0	CM7_0	Cortex-M7_0 core interrupt routing enable Enable or disable the corresponding interrupt request to route to Cortex-M7_0 core. 0: Routing disabled 1: Routing enabled

27. Client configuration (CLTCFG)

27.1. Overview

CLTCFG (client configuration) is a module to give initial configurations to other modules. This module begins to read data in option bytes before chip starts working. CLTCFG module can work in standby mode, and the registers in the CLTCFG module can be divided into three categories, each of which can be restored to the default value after the corresponding reset, namely, some registers can be reset when the system reset is generated, some registers can be reset when the subsystem reset is generated, and the other registers can be reset when the power-on reset is generated

27.2. Characteristics

- Load initial configuration from FMC option bytes.
- Support three types of register reset under the corresponding reset.
- Support the configuration of power watchdog.
- Support system reset information and interrupt.
- Support configuring some functions when standby exit happens.
- Support cpu into wait mode.
- Support the configuration of some function pins, such as reset pin, fault reaction pins and ADC standard channel pins.
- Lifecycle status and force lifecycle.

27.3. Function overview

The CLTCFG can indicate the current working status of some modules, such as the configuration of PMU and ADC channel, the pin function reuse status of GPIO, and chip life cycle control . And also, the CLTCFG can obtain the relevant configuration from option bytes to the register during chip startup, Please refer to the description of CLTCFG part in the ***Option bytes description*** for more details. In addition, user can also configure the work pattern of other modules directly through CLTCFG registers, it should be noticed that these registers of this division will be restored to default value as the chip is reset.

27.4. Register definition

CLTCFG base address: 0x4000 2400

Table 23-56. CLTCFG register overview

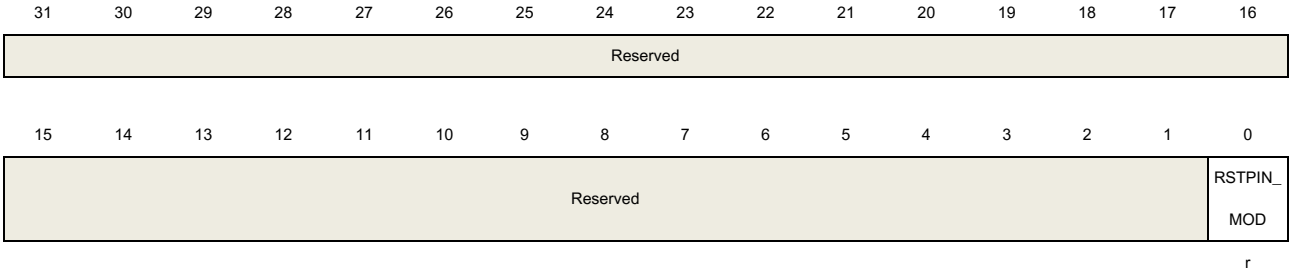
Register name	Symbol	Address offset	Reset value
System reset status register 1	CLTCFG_SR_STAT1	0x04	0x0000 0001
System reset status register 2	CLTCFG_SR_STAT2	0x08	0x0000 0000
Subsystem reset status register 1	CLTCFG_SUBSR_STAT1	0x10	0x0000 0001
Subsystem reset status register 2	CLTCFG_SUBSR_STAT2	0x14	0x0000 0000
Power-on reset control register 0	CLTCFG_POR_CTL0	0x18	0x0000 0400
Power-on reset control register 1	CLTCFG_POR_CTL1	0x1C	0x0000 0000
System reset control register 1	CLTCFG_SR_CTL1	0x24	0x0000 0000
Subsystem reset control register 0	CLTCFG_SUBSR_CTL0	0x28	0x0000 0000
Subsystem reset control register 1	CLTCFG_SUBSR_CTL1	0x2C	0x0000 0000
Subsystem reset control register 2	CLTCFG_SUBSR_CTL2	0x30	0x0000 0000
Triple modular redundancy control register	CLTCFG_TMR_CTL	0x34	0x0000 0000
Triple modular redundancy status register	CLTCFG_TMR_STAT	0x38	0x0000 0000
Triple modular redundancy flag clear register	CLTCFG_TMR_FC	0x3C	0x0000 0000
Life cycle control and status register	CLTCFG_LC_CS	0x40	0x0000 0077

27.4.1. System reset status register 1 (CLTCFG_SR_STAT1)

Address offset: 0x04

Reset value: 0x0000 0001

This register has to be accessed by word (32-bit).



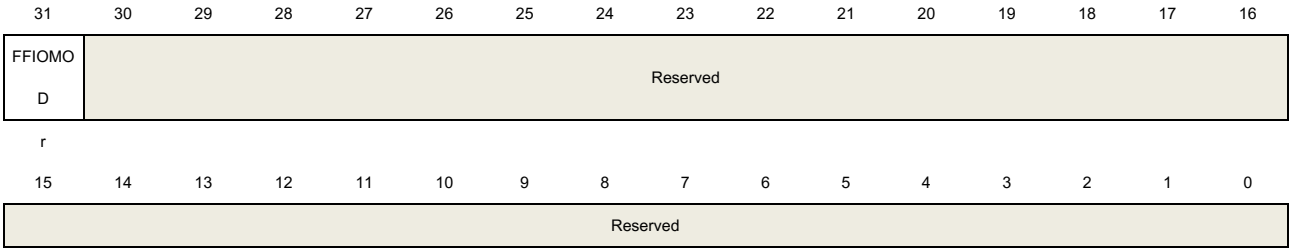
Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	RSTPIN_MOD	Specifies the reset pin mode. This bit is configured by bit0 of the option byte OB_CLTCFG[7:0]. 0: Reset pin acts as general GPIO pin multiplexed with non-reset function. 1: Reset pin acts as dedicated.

27.4.2. System reset status register 2 (CLTCFG_SR_STAT2)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



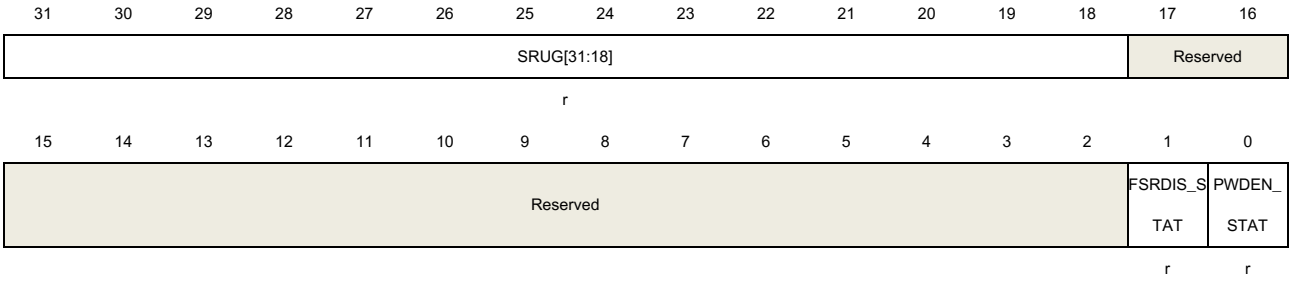
Bits	Fields	Descriptions
31	FFIOMOD	Indicates the configuration of the fatal fault pins mode in option byte This bit is configured by bit1 of the option byte OB_CLTCFG[7:0] 0: FFIO0 and FFIO1 acts as general GPIO pins. 1: FFIO0 and FFIO1 acts as dedicated fatal fault reaction pins. **Note**: For GD32A711 / A712 devices, when FFIO0 (PJ10) and FFIO1 (PJ11) are used as dedicated fatal fault reaction pins, it is necessary to ensure that the PJ10 and PJ11 pins are configured to alternate function mode and the AF0 function is selected.
30:0	Reserved	Must be kept at reset value.

27.4.3. Subsystem reset status register 1 (CLTCFG_SUBSR_STAT1)

Address offset: 0x10

Reset value: 0x00000001

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:18	SRUG[31:18]	These fields reflect the configuration of the system reset upgrade function as configured in option bytes. All SRUG[31:0] fileds are configured by option byte OB_CLTCFG[39:8]. And SRUG[31:18] fileds are configured by the option byte OB_CLTCFG[39:26]. SRUG[31]: Reserved SRUG[30]: The upgrade function of DBG_SYS_RST is enabled

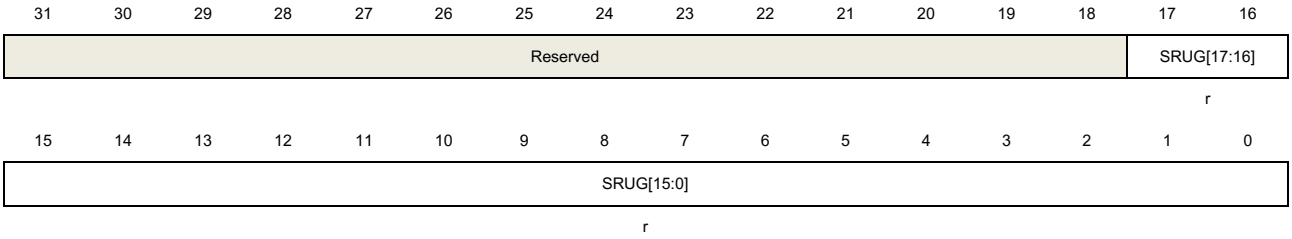
		SRUG[29]: The upgrade function of SW_SYS_RST is enabled SRUG[28:19]: Reserved SRUG[18]: The upgrade function of HSM_SYS_RST is enabled See SRUG[17:0] of CLTCFG_SUBSR_STAT2 register for more details.
17:2	Reserved	Must be kept at reset value.
1	FSRDIS_STAT	This bit indicates the final-step regulator automatic crossover from boot regulation feature is disabled or not. 0: Final-step auto crossover features is enabled. 1: Final-step auto crossover features is disabled.
0	PWDEN_STAT	This bit indicates the POR watchdog is enabled or not. This bit is configured by bit7 of the option byte OB_CLTCFG[7:0]. 0: POR watchdog is disabled. 1: POR watchdog is enabled.

27.4.4. Subsystem reset status register 2 (CLTCFG_SUBSR_STAT2)

Address offset: 0x14

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



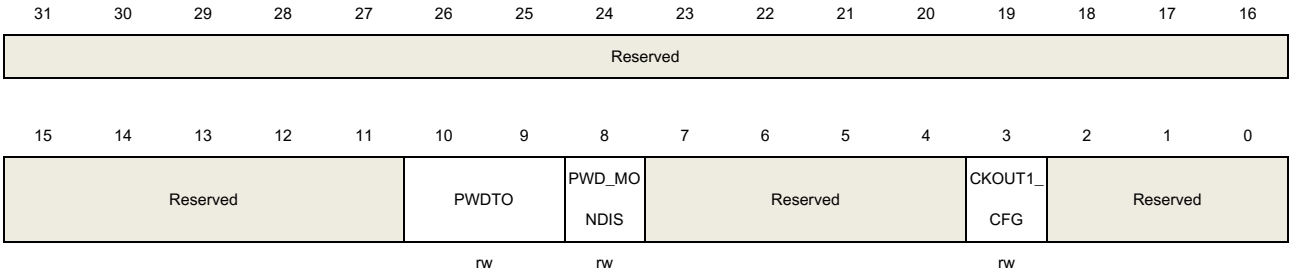
Bits	Fields	Descriptions
31:18	Reserved	Must be kept at reset value.
17:0	SRUG[17:0]	These fields reflect the configuration of the system reset upgrade function as configured in option bytes All SRUG[31:0] fileds are configured by option byte OB_CLTCFG[39:8]. And SRUG[17:0] fileds are configured by the option byte OB_CLTCFG[25:8]. SRUG[17:10]: Reserved SRUG[9]: The upgrade function of LOPRST is enabled SRUG[8]: The upgrade function of LOHRST is enabled SRUG[7]: Reserved SRUG[6]: The upgrade function of RCTL_SBSRU is enabled SRUG[5]: Reserved SRUG[4]: The upgrade function of STCM_REC_FAIL is enabled SRUG[3]: The upgrade function of FMU_REA_FAIL is enabled SRUG[2:0]: Reserved

Refer to the Option bytes description chapter and Reset control unit chapter for more details about configuration of system reset upgrade function.

27.4.5. Power-on reset control register 0 (CLTCFG_POR_CTL0)

Address offset: 0x18
Reset value: 0x0000 0400

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:11	Reserved	Must be kept at reset value.
10:9	PWDTO	Configure timeout value of the POR watch dog. 00: POR watchdog timeout value = 06.25ms 01: POR watchdog timeout value = 12.50ms 10: POR watchdog timeout value = 25.00ms 11: POR watchdog timeout value = 50.00ms
8	PWD_MONDIS	Disable the standby entry and exit monitor of the POR watchdog. 0: Enable 1: Disable
7:4	Reserved	Must be kept at reset value.
3	CKOUT1_CFG	Clock out standby expose over system reset and subsystem reset. This field controls the CK_OUT1 output in standby mode function availability while system reset or subsystem reset on PM4. 0: The CK_OUT1 output in standby mode function is not available while system reset or subsystem reset on PM4. 1: The CK_OUT1 output in standby mode function is available while system reset or subsystem reset on PM4.
2:0	Reserved	Must be kept at reset value.

27.4.6. Power-on reset control register 1 (CLTCFG_POR_CTL1)

Address offset: 0x1C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															FMU_MONDIS
															rw

Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	FMU_MONDIS	Disable FMU monitor. The internal security monitoring mechanism of the FMU is enabled by default. If it is triggered, the system reset will generated. If this bit is set to 1, the internal security monitoring mechanism of the FMU is disabled. 0: Enable FMU monitor 1: Disable FMU monitor

27.4.7. System reset control register 1 (CLTCFG_SR_CTL1)

Address offset: 0x24

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FRCFG	Reserved						
								rw							

Bits	Fields	Descriptions
31:8	Reserved	Must be kept at reset value.
7	FRCFG	Configure the fault reaction state during self test 0: Fault reaction state will change to high Z during selftest when device is under reset. 1: Fault reaction state will remain in fault state until this bit is cleared.
6:0	Reserved	Must be kept at reset value.

27.4.8. Subsystem reset control register 0 (CLTCFG_SUBSR_CTL0)

Address offset: 0x28

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															PAD_RET_CFG
															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value.
16	PAD_RET_CFG	Configure the IO state in the standby mode. 0: Enable IO pad retaining function before standby entry. 1: Disable IO pad retaining function after standby exit. Note: the pad keeping function must be disabled before changing the pad's mode.
15:0	Reserved	Must be kept at reset value.

27.4.9. Subsystem reset control register 1 (CLTCFG_SUBSR_CTL1)

Address offset: 0x2C

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														CORE0_WAIT	Reserved
														rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ADC1_CH23_DP	ADC1_CH22_DP	ADC1_CH15_DP	ADC1_CH14_DP	ADC0_CH9_DP	ADC0_CH8_DP	Reserved
									rw	rw	rw	rw	rw	rw	

Bits	Fields	Descriptions
31:18	Reserved	Must be kept at reset value.
17	CORE0_WAIT	Configuration for CM7 core 0 wait mode. 0: Disable CM7 core 0 wait mode. 1: Put CM7 core 0 into wait mode by CPUWAIT.
16:7	Reserved	Must be kept at reset value.
6	ADC1_CH23_DP	Drive pin selection for ADC1 channel 23. 0: Selects the default PG8 as drive pin.

		1: Selects the PA11 as drive pin.
5	ADC1_CH22_DP	Drive pin selection for ADC1 channel 22. 0: Selects the default PG6 as drive pin. 1: Selects the PA9 as drive pin.
4	ADC1_CH15_DP	Drive pin selection for ADC1 channel 15. 0: Selects the default PA13 as drive pin. 1: Selects the PB11 as drive pin.
3	ADC1_CH14_DP	Drive pin selection for ADC1 channel 14. 0: Selects the default PA15 as drive pin. 1: Selects the PJ5 as drive pin.
2	ADC0_CH9_DP	Drive pin selection for ADC0 channel 9. 0: Selects the default PG7 as drive pin. 1: Selects the PD11 as drive pin.
1	ADC0_CH8_DP	Drive pin selection for ADC0 channel 8. 0: Selects the default PC6 as drive pin. 1: Selects the PD13 as drive pin.
0	Reserved	Must be kept at reset value.

27.4.10. Subsystem reset control register 2 (CLTCFG_SUBSR_CTL2)

Address offset: 0x30

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BTADDR[30:15]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BTADDR[14:0]															BTMOD
rw															rw

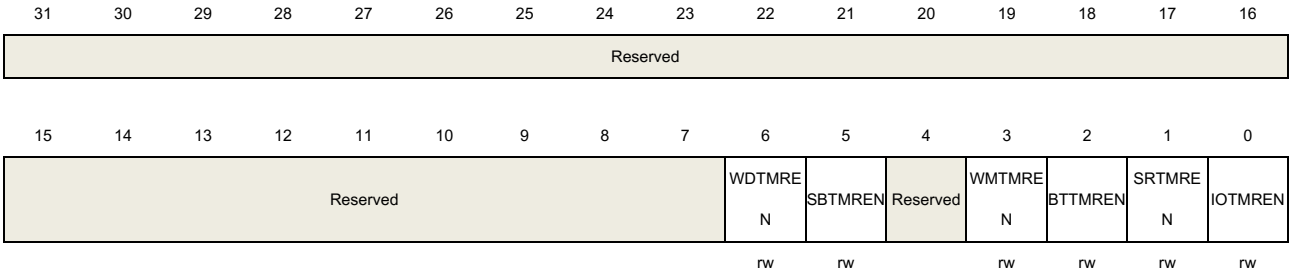
Bits	Fields	Descriptions
31:1	BTADDR	The SBAF application boot address of CM7_Core0 after exiting from standby mode. This address is used for the core boot of CM7_Core0 by SBAF when exiting from Standby mode (only valid for fast standby boot). Note: The startup address here uses bits [31:1]. Due to alignment requirements, bit 0 of the startup address is automatically padded with zero.
0	BTMOD	Configure boot mode after exiting from standby. 0: Normal boot mode. 1: Fast standby Boot.

27.4.11. Triple modular redundancy control register (CLTCFG_TMR_CTL)

Address offset: 0x34

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).



Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	WDTMREN	Enable Watch dog TMR error reporting to FMU 1: Enable watch dog triple modular redundancy. 0: Disable watch dog triple modular redundancy.
5	SBTMREN	Enable Standby mode TMR error reporting to FMU 1: Enable standby mode triple modular redundancy. 0: Disable standby mode triple modular redundancy.
4	Reserved	Must be kept at reset value.
3	WMTMREN	Enable Wait mode TMR error reporting to FMU 1: Enable WM triple modular redundancy. 0: Disable WM triple modular redundancy.
2	BTTMREN	Enable Boot mode TMR error reporting to FMU 1: Enable boot mode triple modular redundancy. 0: Disable boot mode triple modular redundancy.
1	SRTMREN	Enable System reset TMR error reporting to FMU 1: Enable system reset triple modular redundancy. 0: Disable system reset triple modular redundancy.
0	IOTMREN	Enable GPIO TMR error reporting to FMU 1: Enable GPIO triple modular redundancy. 0: Disable GPIO triple modular redundancy.

27.4.12. Triple modular redundancy status register (CLTCFG_TMR_STAT)

Address offset: 0x38

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									WDTMRERR	SBTMRERR	Reserved	WMTMRERR	BTTMRERR	SRTMRERR	IOTMRERR
									RR	R		RR	R	RR	R
									r	r		r	r	r	r

Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	WDTMRERR	Watch dog TMR error flag bit. 0: Watch dog TMR error is not generated. 1: Watch dog TMR error is generated.
5	SBTMRERR	Standby TMR error flag bit. 0: Standby TMR error is not generated. 1: Standby TMR error is generated.
4	Reserved	Must be kept at reset value.
3	WMTMRERR	Wait mode TMR error flag bit. 0: WM TMR error is not generated. 1: WM TMR error is generated.
2	BTTMRERR	Boot mode TMR error flag bit. 0: Boot mode TMR error is not generated. 1: Boot mode TMR error is generated.
1	SRTMRERR	System reset TMR error flag bit. 0: System reset TMR error is not generated. 1: System reset TMR error is generated.
0	IOTMRERR	GPIO TMR error flag bit. 0: GPIO TMR error is not generated. 1: GPIO TMR error is generated.

27.4.13. Triple modular redundancy flag clear register (CLTCFG_TMR_FC)

Address offset: 0x3C

Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	WDTMRFC	SBTMRFC	Reserved	WMTMRFC	BTTMRFC	SRTMRFC	IOTMRFC
	w	w		w	w	w	w

Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	WDTMRFC	Clear bit for Watch dog TMR error flag. 0: No effect. 1: Clear Watch dog TMR error flag.
5	SBTMRFC	Clear bit for Standby TMR error flag. 0: No effect. 1: Clear Standby TMR error flag.
4	Reserved	Must be kept at reset value.
3	WMTMRFC	Clear bit for Wait mode TMR error flag. 0: No effect. 1: Clear WM TMR error flag.
2	BTTMRFC	Clear bit for Boot mode TMR error flag. 0: No effect. 1: Clear Boot mode TMR error flag.
1	SRTMRFC	Clear bit for System reset TMR error flag. 0: No effect. 1: Clear System reset TMR error flag.
0	IOTMRFC	Clear bit for GPIO TMR error flag. 0: No effect. 1: Clear GPIO TMR error flag.

27.4.14. Life cycle control and status register (CLTCFG_LC_CS)

Address offset: 0x40
Reset value: 0x0000 0077

After reset, this register value is 0x0000 0077, and after scanning, it changes according to the programmed value.

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						FTNLCS		Reserved	RLCS			FLCEN	CLCS		
						rw			r			rw	r		

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:8	FTNLCS	Force to next life cycle stage Provides provisions for a temporary LC update. LC can be temporarily forced as follows: GD_PRODUCTION → CUSTOMER_DELIVERY (Force one stage forward) CUSTOMER_DELIVERY → OEM_PRODUCTION (Force one stage forward) CUSTOMER_DELIVERY → IN_FIELD (Force two stages forward) OEM_PRODUCTION → IN_FIELD (Force one stage forward) IN_FIELD → not allowed to move LC FA → not allowed to move LC **Note:** This field can be reset when the system reset is generated. 00: No force 01: Force LC to one next 10: Force LC to two next (only possible in CUSTOMER_DELIVERY stage) 11: Reserved (no effect)
7	Reserved	Must be kept at reset value.
6:4	RLCS	Real life cycle stage Indicates the real life cycle stage of the chip. The LC can move in this sequence: (110: GD_PRODUCTION) → (011: CUSTOMER_DELIVERY) → (010: OEM_PRODUCTION) → (111: IN_FIELD) → (000: FA) **Note:** This field can be reset as 111 when the power-on reset is generated, and then it changes according to the programmed value after scanning. 000: FA 001: Reserved 010: OEM_PRODUCTION 011: CUSTOMER_DELIVERY 100: Reserved 101: Reserved 110: GD_PRODUCTION 111: IN_FIELD
3	FLCEN	Force LC Indicates if the LC is normal or forced on the current LC. If the value of this field is 1, the value configured according to the CURRENT_LC is forced on the current LC whereas the real LC remains the same. **Note:** This field can be reset when the system reset is generated. 0: Normal 1: Forced on current LC
2:0	CLCS	Current life cycle stage Indicates the current life cycle stage of the chip. These bits must not be modified

during the processing of the flash program and erase

The LC can move in this sequence: (110: GD_PRODUCTION) → (011: CUSTOMER_DELIVERY) → (010: OEM_PRODUCTION) → (111: IN_FIELD) → (000: FA)

Note: This field can be reset as 111 when the subsystem reset is generated.

000: FA

001: Reserved

010: OEM_PRODUCTION

011: CUSTOMER_DELIVERY

100: Reserved

101: Reserved

110: GD_PRODUCTION

111: IN_FIELD

28. Hardware Security Module (HSM)

28.1. Overview

The HSM system is based on the M23 processor to achieve functional operation.

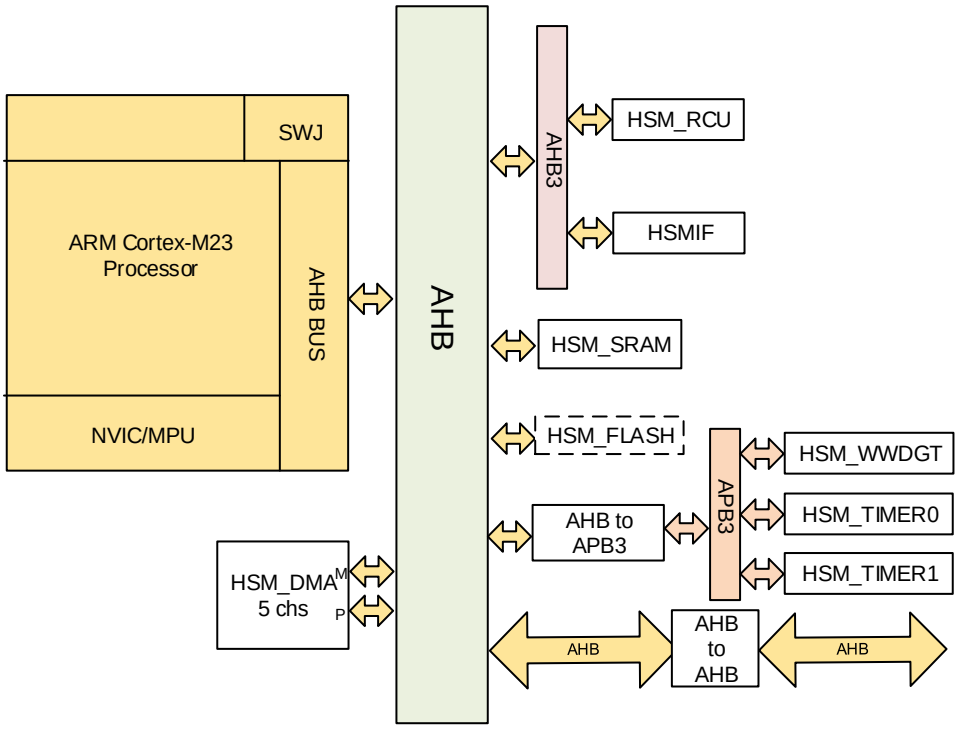
28.2. Characteristics

- On-chip SRAM and flash shared with host flash
- DMA transfer is supported.
- Two 16bits timer modules
- One watchdog timer module

28.3. Function overview

28.3.1. HSM system

Figure 28-1. HSM system architecture



HSM_RCU is used to control the operation clock and reset signals of HSM peripherals.

HSM_SRAM offers exclusive sram memory for HSM software operation. HSM_FLASH is located at Host flash. HSM_WWDGT and HSM_TIMER are integrated for HSM core normally working. A 32bit AHB master bus is connect to host bus matrix for large amount of data transfers. HSM has the ability to access almost all resources of whole chip. HSM_DMA is used to improve HSM system performance. And HSMIF is used to exchange messages between HSM and HOST.

The ARM Cortex®-M23 processor and the Nested Vectored Interrupt Controller (NVIC) prioritize and handle all exceptions in Handler Mode. The processor state is automatically stored to the stack on an exception and automatically restored from the stack at the end of the Interrupt Service Routine (ISR).

The vector is fetched in parallel to the state saving, enabling efficient interrupt entry. The processor supports tail-chaining, which enables back-to-back interrupts to be performed without the overhead of state saving and restoration. *Table 28-1. NVIC exception types in Cortex®-M23* and *Table 28-2. Interrupt vector table for HSM* list all exception types.

Table 28-1. NVIC exception types in Cortex®-M23

Exception type	Vector number	Priority (a)	Vector address	Description
-	0	-	0x0000_0000	Reserved
Reset	1	-3	0x0000_0004	Reset
NMI	2	-2	0x0000_0008	Non maskable interrupt
HardFault	3	-1	0x0000_000C	All class of fault
-	4-10	-	0x0000_0010 - 0x0000_002B	Reserved
SVCall	11	Programmable	0x0000_002C	System service call via SWI instruction
-	12-13	-	0x0000_0030 - 0x0000_0034	Reserved
PendSV	14	Programmable	0x0000_0038	Pendable request for systemservice
SysTick	15	Programmable	0x0000_003C	System tick timer

The SysTick calibration value is 10000 and SysTick clock frequency is fixed to HCLK*0.125. So this will give a 1ms SysTick interrupt if HCLK is configured to 80MHz.

Table 28-2. Interrupt vector table for HSM

Interrupt number	Vector number	Peripheral interrupt description	Vector Address
IRQ 0	16	DMA Channel0 global interrupt	0x0000_0040
IRQ 1	17	DMA Channel1 global interrupt	0x0000_0044
IRQ 2	18	DMA Channel2 global interrupt	0x0000_0048
IRQ 3	19	DMA Channel3 global interrupt	0x0000_004C
IRQ 4	20	DMA Channel4 global interrupt	0x0000_0050
IRQ 5	21	TIMER0 global interrupt	0x0000_0054

Interrupt number	Vector number	Peripheral interrupt description	Vector Address
IRQ 6	22	TIMER1 global interrupt	0x0000_0058
IRQ 7	23	WWDGT global interrupt	0x0000_005C
IRQ 8	24	JTAG interrupt	0x0000_0060
IRQ 9	25	HSM Sensor global interrupt	0x0000_0064
IRQ 10	26	HSM communication0 global interrupt	0x0000_0068
IRQ 11	27	HSM communication1 global interrupt	0x0000_006C
IRQ 12	28	HSM error interrupt	0x0000_0070
IRQ 13	29	HSM IRM0 interrupt	0x0000_0074
IRQ 14	30	HSM IRM1 interrupt	0x0000_0078
IRQ 15	31	HSM IRM2 interrupt	0x0000_007C
IRQ 16	32	HSM IRM3 interrupt	0x0000_0080
IRQ 17	33	HSM IRM4 interrupt	0x0000_0084
IRQ 18	34	HSM IRM5 interrupt	0x0000_0088
IRQ 19	35	HSM IRM6 interrupt	0x0000_008C
IRQ 20	36	HSM IRM7 interrupt	0x0000_0090
IRQ 24	40	DBG authentication request interrupt	0x0000_00A0
IRQ 25	41	CTI0 interrupt	0x0000_00A4
IRQ 26	42	CTI1 interrupt	0x0000_00A8

28.3.2. HSM RCU

Power reset

The Power reset is generated by either an external reset as Power On and Power Down reset (POR/PDR reset) or by the internal reset generator when exiting Standby mode.

System reset

A system reset resets the processor core and peripheral IP components except for HSM RAM.

The clocks of HSM are derived from the host system clock, the maximum frequency of the AHB and Cortex®-M23 domains is 80 MHz.

28.3.3. HSM Power management

Power saving modes

Power management provides one type power saving mode, standby mode.

Standby mode

The Standby mode is based on the SLEEPDEEP mode of the Cortex®-M23. In Standby mode, the whole domain is power off.

Table 28-3. HSM low power mode

Host state	HSM state	Description	Entry	Wakeup
standby	standby	HSM power off	Host and HSM enter standby mode	Pin or other wake-up sources wake up MCU

28.3.4. HSM interface(HSMIF)

The HSMIF is an interface module between HSM sub-system and Host system. The main purpose of the HSMIF is to exchange information and messages between HSM and Host.

28.3.5. HSM memory

HSM SRAM

All of AHB SRAM support byte, half-word (16 bits), and word (32 bits) accesses. HSM SRAM support retention low power mode. HSM SRAM support ECC.HSM SRAM can be accessed by CPU and HSM DMA.

HSM Flash

HSM FMC (Flash Memory Controller) contains code and user data for HSM sub-system. The HSM Flash memory area is configured from Host flash memory. HSM FMC physically locates in Host system.

28.3.6. HSM timers

HSM WWDGT

The window watchdog timer of hardware security module (HSM_WWDGT) is used to detect hardware security module (system HSM) sub system failures due to software malfunctions. The watchdog timer also causes a reset when the counter is refreshed before the counter reached the window register value. So the software should refresh the counter in a limited window.

HSM TIMER

The HSM Timer includes two 16-bit basic timers which can be started and stopped individually. Each Timer consists of a 16-bit up counter with its own prescaler and reload value.

28.3.7. HSM DMA

HSM DMA is used to transfer encrypted or decrypted data between HSM and Host. There are 5 channels in the HSM DMA controller. Each channel is dedicated to manage memory

access requests from one or more peripherals. An arbiter is implemented inside to handle the priority among DMA requests.

The system bus is shared by the DMA controller and the Cortex®-M23 core. When the DMA and the CPU are targeting the same destination, the DMA access may stop the CPU access to the system bus for some bus cycles. Round-robin scheduling is implemented in the bus matrix to ensure at least half of the system bus bandwidth for the CPU.

28.3.8. HSM Debug

The Host debug system can access HSM internal resources, such as peripherals, local RAM, CPU related registers and HSMIF module registers. This operation is only allowed when the HSM debug is enabled. Besides, the debug module can track HSM data on the Host bus, and access the HSM specific memory region in the flash.

When the Host system debug is enabled, the HSMIF module is able to hold the TIMER0 / TIMER1 / WWDGT counter clock for debug when core halted.

29. Fault management unit (FMU)

29.1. Overview

Fault management unit (FMU) has a hardware fault management state machine (FMSM) for managing system faults. When a fault occurs, the fault management process does not require CPU intervention. FMU will automatically handle the fault and keep the device in a secure state.

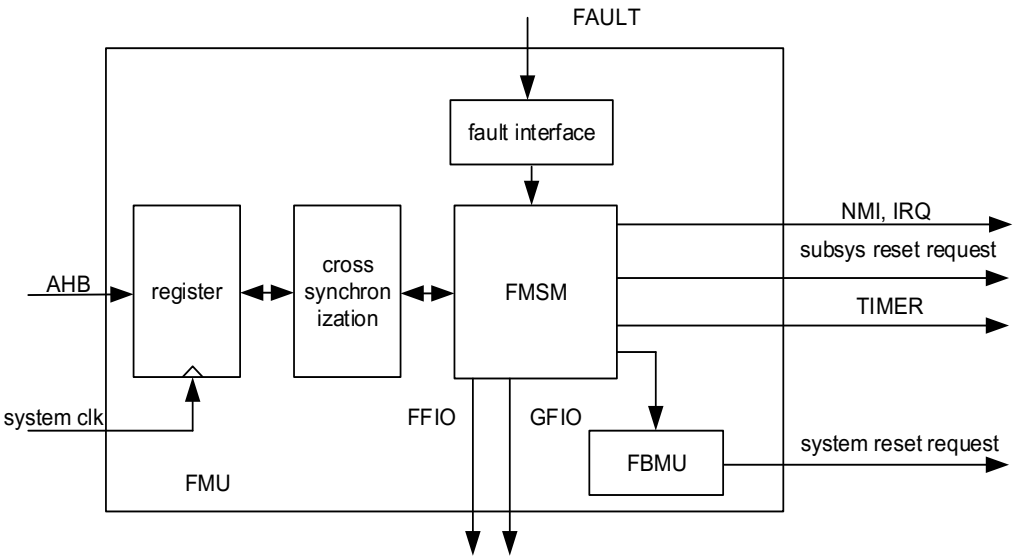
29.2. Characteristics

- Fault group (FG) fault management state machine.
- Fault recovery management, hardware recovery or software recovery.
- Support fault injection (software triggered fault).
- Initialization work mode management.
- Fatal fault I/O pin (FFIO) / general fault I/O pin (GFIO) signal output is used to indicate faults.
- CAUTION work mode internal interrupt request.
- 7 kinds configurable internal chip reaction for each FG.
- Support management all fault input into 12 FGs.
- Support FFIO protocol include time switching and bi-stable.
- Software configurable FFIO / GFIO state is used to indicate fault.

29.3. Function description

29.3.1. Block diagram

Figure 29-1. FMU module block diagram



The register of the FMU and the timers of the FMU have different clock source. The clock source of the FMU register is from the system clock. The clock source of the FMU timers is from the internal clock.

29.3.2. FMU clock

The FMU has two clock source inputs from the device, one is the system clock and the other is the IRC48MHz clock. The IRC48MHz clock from internal can be considered a more secure clock (SECURE_CLK).

29.3.3. FMU fault grouping

The device supports 12 FGs. Each FG has its own fault trigger sources. Usually, the fault trigger sources of each FG is a group of faults with similar properties. If one of these fault trigger sources is enabled by FMU_FGTSENx registers, and then the system occurs this fault, the corresponding interrupt request will be triggered. The *Table 29-1 FMU fault group definition* shows the details of FMU fault group.

Table 29-1 FMU fault group definition

Fault group	Category	Fault number	Fault source module(error type)
FG[0]	/	0	CM7_0 core lockup status
		1	Reserved

		2	HSM core lockup status
		3-9	Reserved
FG[1]	/	10	IMICU of BM1 safety alarm
		11	IMICU of BM2 safety alarm
		12	TCM 1:1 widget alarm status.
		13	Reserved
		14	H2H0 widget of TCM redundant monitor alarm status
		15	H2H0 widget of QSPI redundant monitor alarm status
		16	H2H0 widget of DMA0 mem redundant monitor alarm status
		17	H2H0 widget of DMA0 peri redundant monitor alarm status
		18	H2H0 widget of DMA1 mem redundant monitor alarm status
		19	H2H0 widget of DMA1 peri redundant monitor alarm status
		20	H2H1 widget of HSM redundant monitor alarm status
		21	Split0 widget of DMA0 mem redundant monitor alarm status
		22	Split0 widget of DMA0 peri redundant monitor alarm status
		23	Split0 widget of DMA1 mem redundant monitor alarm status
		24	Split0 widget of DMA1 peri redundant monitor alarm status
		25	Reserved
		26	Split1 widget of HSM redundant monitor alarm status
		27	CM7_0 AXIM port timeout and slaver ID check alarm status
		28	Reserved
		29	CM7_0 AHBP port timeout and slaver ID check alarm status
		30	Reserved
		31	DMA0 mem port timeout and slaver ID check alarm status
		32	DMA0 peri port timeout and slaver ID check alarm status
		33	DMA1 mem port timeout and slaver ID check alarm status

		34	DMA1 peri port timeout and slaver ID check alarm status
		35	HSM port timeout and slaver ID check alarm status
		36	Reserved
		37	ALL slaver port wdata safety alarm of EDC CHK
		38	ALL slaver port addr safety alarm of EDC CHK
		39	Reserved
		40	EDC CHK rdata of CM7_0 AXIM safety alarm
		41	Reserved
		42	EDC CHK rdata of CM7_0 AHBP safety alarm
		43	Reserved
		44	EDC CHK rdata of HSM safety alarm
		45	EDC CHK rdata of DMA0 mem safety alarm
		46	EDC CHK rdata of DMA0 peri safety alarm
		47	EDC CHK rdata of DMA1 mem safety alarm
		48	EDC CHK rdata of DMA1 peri safety alarm
		49	AHB1 redundant monitor alarm status
		50	AHB2 redundant monitor alarm status
		51	APB1 (including AHB to APB bridge) redundant monitor alarm status
		52	APB2 (including AHB to APB bridge) redundant monitor alarm status
		53	CM7_0 AXI to AHB1 bridge redundant monitor alarm status.
		54	Reserved
		55	FMC control logic redundant monitor error
		56	TCM0 control logic redundant monitor error
		57	Reserved
		58	SRAM control logic redundant monitor error
		59	Reserved
FG[2]	/	60	Multi bit ECC error from CM7_0 DCache data memory
		61	Multi bit ECC error from CM7_0 DCache tag memory
		62-63	Reserved
		64	Multi bit ECC error from CM7_0 ICache data memory
		65	Multi bit ECC error from CM7_0 ICache tag memory
		66-67	Reserved
		68	Single bit ECC error from CM7_0 DCache data memory.

		69	Single bit ECC error from CM7_0 DCache tag memory.
		70-71	Reserved
		72	Single bit ECC error from CM7_0 ICache data memory
		73	Single bit ECC error from CM7_0 ICache tag memory.
		74-75	Reserved
		76	CPU0 TCM uncorrectable data error
		77	CPU0 TCM address buffer overflow error
		78-79	Reserved
		80	SRAM uncorrectable data error
		81	SRAM Data overflow errors can be corrected
		82	FMC SRAM Uncorrectable data error in buffer
		83	FMC SRAM Correctable data overflow error in buffer
		84	FMC FLASH uncorrectable data error
		85	FMC FLASH Correctable data overflow error
		86	Reserved
FG[3]	/	87	FLASH load ECC error(include lifecycle error)
		88-89	Reserved
FG[4]	/	90	Low voltage detection from PMU
		91	High voltage detection from PMU
		92	LDO temperature overflow
FG[5]	/	93	STCM recovery fault happens during self-test
FG[6]	/	94	IRM monitored error
		95	Reserved
FG[7]	/	96	ADC0 error
		97	ADC1 error
		98-100	Reserved
		101	DMA0 error
		102	DMA1 error
		103	Reserved
FG[8]	TMR error	104	Reserved
		105	GPIO
		106	PMU
		107	QSPI
		108	DMA0
		109	DMA1
		110	SYSCFG
		111	HWSEM
		112	FMU
		113	STCM

		114	RCTL
		115	MTC
		116	CLTCFG
		117	Reserved
		118	WWDGT0
		119-127	Reserved
		128	Bus_Wrapper
		129	CCTL
		130	FMC
	memory correctable error	131	CPU0 TCM correctable data error
		132	Reserved
		133	Global SRAM correctable data error
		134	FMC SRAM correctable data error
		135	FMC FLASH correctable data error
FG[9]	Software monitor	136-143	SW triggered fault 0/1/2/3/4/5/6/7
FG[10]	Software monitor	144-151	SW triggered fault 0/1/2/3/4/5/6/7
FG[11]	Software monitor	152-159	SW triggered fault 0/1/2/3/4/5/6/7

29.3.4. Fault reactions

There are 7 fault reactions for all faults, the reactions are shown as follows:

- Interrupt to CPU.
- NMI to CPU.
- Subsystem reset request to RCU.
- System reset request to RCU (triggering system reset).
- Holding indication to GPOs of advanced timer and general level0 timer.
- General fault I/O pin (GFIO) to GPO.
- Fatal fault I/O pin (FFIO) to GPOs.

Each FG has its own fault trigger sources. By configuring registers, a fault trigger source of FG can trigger different fault reactions when the fault be triggered.

29.3.5. FMU behavior monitor unit

A FMU behavior monitor unit (FBMU) is supported. When a fault occurred, FBMU will monitor FMU's behavior over a period of time. FBMU's main purpose is to be able to trigger a system reset when the FMU fails and can not react fault properly. If FMU works normally, system reset will not be triggered by FBMU.

When FMU is in INIT work mode, a "silent signal" from FMU will output to FBMU. The "silent signal" indicates that the FMU will not react any fault occurred later, so monitor FMU's behavior is meaningless. At this time, FBMU will not capture fault until the work mode of FMU is transformed from INIT to other work mode.

FBMU can be disabled via the FMU_MONDIS bit in the CLTCFG_POR_CTL1 register.

FBMU timer

FBMU works base on a FBMU timer, and it's clock source is SECURE_CLK. The chip will set FBMU timer's timeout value (value is 0x69780) after system reset and this timeout value is the duration of the monitor window. Once a fault which is enabled occurred, the timer will automatically initialize and start count. The timer is unable to reenter, that means once FBMU timer is triggered by a fault "A", and start count, a new fault that occurred after fault "A" can not trigger FBMU timer again (include initialize and start count). When FMU is reacting a fault, and the reaction is one of followings, FBMU timer will stop:

- Subsystem reset
- Interrupt request to CPU (triggered by CAUTION work mode)
- NMI to CPU
- Holding indication to GPOs of advanced timer and general level0 timer
- FFIO out triggered (by FMU or by software)
- GFIO out triggered (by FMU or by software)

When FBMU timer is stopped, if the fault, which is captured by FBMU and has triggered FBMU timer running, is not recovery, then the fault can not trigger FBMU timer running again. Because once FBMU timer is stopped the fault capture logic will be cleared, FBMU timer will run again only when a new fault occurred.

FBMU has a failure indicator, and the indicator's output signal is connected to system reset of RCU. The output signal will be asserted, and a system reset will occur when FBMU timer is timeout. The FBMU module resets in the same way as the FMU module, when FBMU reset, its capture logic is cleared, FBMU timer is stopped and the failure indicator output signal is not asserted.

FMU in FAULT work mode signal

FMU in FAULT work mode signal (FIFS) can be output to FBMU from FMU. If FIFS is logic level 1, it indicates a fault is captured by FMU. FMU will assert and then deassert FIFS in one of follow states:

- When FOSCFG[1:0] bits are set to 0b01 which will make FFIO output low level always, FMU will assert this signal, until FFIO output are programmed not to be low level always.
- When FFIO output is not programmed to low level always by configure FOSCFG[1:0] bits, then if a FG fault occurred, and transforms FMU works mode to FAULT, FIFS will be asserted. FIFS will be deasserted, when FMU works mode transforms to other work mode.

29.3.6. Fault reaction programming

Table 29-2. FG reaction shows an example of FG fault recommend reaction, upgrade reaction and downgrade reaction. It is worth noting once downgrade fault reaction occurred, the functionality may not guarantee.

Table 29-2. FG reaction

FG	Recommend reaction	Upgrade reaction	Downgrade reaction
FG[0]	Subsystem reset	System reset	Caution interrupt / NMI
FG[1]	Subsystem reset	System reset	Caution interrupt / NMI
FG[2]	Subsystem reset	System reset	Caution interrupt / NMI
FG[3]	Subsystem reset	System reset	Caution interrupt / NMI
FG[4]	Caution interrupt	Subsystem reset / system reset / NMI	/
FG[5]	Caution interrupt	Subsystem reset / system reset / NMI	/
FG[6]	Caution interrupt	Subsystem reset / system reset / NMI	/
FG[7]	Caution interrupt	Subsystem reset / system reset / NMI	/

Once an enabled fault occurred, corresponding FG fault state bit in FMU_FGSTAT register will be set. The following programming steps are recommended for lockstep errors:

- A subsystem reset to RCU is the first thing worth considering.
- After subsystem reset the core will reboot, then need to initialize TCM and cache.
- Clear corresponding FG fault state bit，which can be realized by write to FMU_FGSTAT register.
- If the corresponding FG fault state bit is not set again, it means fault is cleared, fault error is recovered, then device work normal and nothing special need to do.
- If the corresponding FG fault state bit is set again, it means the cores have different values on the register. Reinitialize the Cortex®-M7 debug module or software trigger an system reset through PMU_MODCFG register, which will make them have same value and recover fault state.

Note: A system reset will make the previous configuration lost.

29.3.7. Recovery definitions

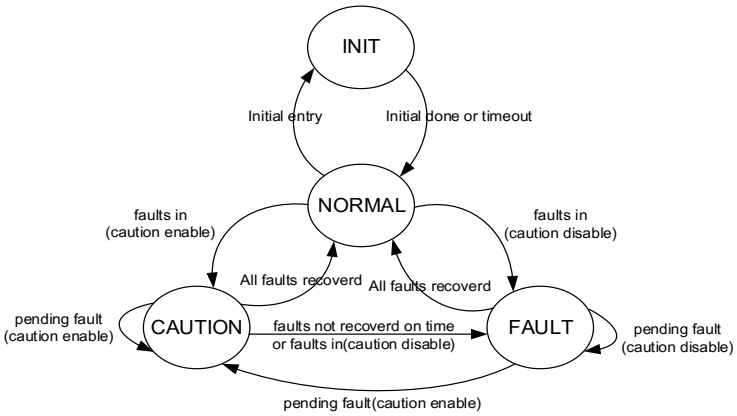
FMU supports two types of fault recovery management:

- Hardware recovery: The fault flag is set by fault signal. The fault will recovery when the fault signal has changed, or the fault will remain. For details, if the fault signal logic level is 0, which indicate a fault happened, once the fault signal logic level changed to 1, the fault will recovery. This recovery process is not need software intervene. FMU will transform work mode following the fault signal logic level change.
- Software recovery: The software recoverable fault is recovered by software when the fault's statue / flag bit is cleared. About how to recover a software recoverable fault can refer to ***Software recovery***. Hardware recoverable fault can be assumed a special type of software recoverable fault.

29.3.8. FMSM description

FMU has a fault management state machine (FMSM), when fault occurred it will control FMU works mode transition, *Figure 29-2. FMU works mode and transition shows* the main work mode and work mode transition path.

Figure 29-2. FMU works mode and transition



FMU works mode

Table 29-3. FMU works mode shows all work modes of the FMU.

Table 29-3. FMU works mode

Work mode	Description
NORMAL	After system reset, FMU will work in this mode, it shows there is no configuration or fault occurred on the device.
INIT	There exist some configuration registers, which can be used to initialize fault masking, timeout and reactions. In INIT work mode, there is a timeout timer. If the FMU works in the INIT work mode and the timeout timer is expiration, the FMU will exist the INIT work mode, and the relevant initialization registers will be set to reset values. Refer to *Table 29-6. Initialization registers* for initialization registers.
CAUTION	If a fault happened and caution state is enabled by FMU_FGCMTEN register. The work mode of FMU will transform to CAUTION. If caution interrupt is enabled by FMU_CMINTREN register, a caution interrupt will be triggered. CAUTION work mode also has a timeout timer and the timeout value can be configured by FMU_CMTVAL register.
FAULT	If a fault happened and caution state is disabled. The work mode of FMU will transform to FAULT. In FAULT work mode, FMU will generate the fault reaction which is defined in *Fault reactions*

- The FMU has a caution state. If the alarm caution is enabled, the FMU will enter the CAUTION work mode instead of the FAULT work mode when a fault occurs. The caution state can be enabled through the FMU_FGCMTEN register.

FMU works mode transition

There are 10 FMU works mode transition paths, which are shown in *Table 29-4. FMU works transition*.

Table 29-4. FMU works transition

Transition path	Work mode transition		Input trigger	Output change
	Start work mode	Destination work mode		
1	NORMAL	INIT	Software can make FMU works mode transform from NORMAL to INIT by following these steps: ■ Unlock temporary configuration lock by FMU_TCLOCK register ■ Set EOP[4:0] bits to transform FMU to INIT work mode ■ Configure initialization registers ■ Set EOP[4:0] bits can transform the work mode of FMU to NORMAL, which will make the configuration about initialization registers take effect. Refer to *Table 29-6. Initialization registers* for initialization registers	Configuration change
2	INIT	NORMAL	■ Timeout ■ Transform the work mode to NORMAL by the configuration of EOP[4:0] bits	■ Configuration put to reset value ■ Configuration take effect
3	NORMAL	CAUTION	Fault occurred with caution enabled and fault unmasked	Caution interrupt output asserted
4	NORMAL	FAULT	Fault occurred with caution disabled and fault unmasked	Depends on fault reaction configuration
5	CAUTION	FAULT	■ Faults not recovered on time ■ Faults in caution disabled	■ NMI output (fault upgrade) ■ Depends on fault reaction configuration
6	CAUTION	CAUTION	Pending fault with caution enabled	Caution interrupt output asserted
7	CAUTION	NORMAL	Input fault recovered	Caution interrupt output deasserted
8	FAULT	CAUTION	Pending fault with caution enabled	Caution interrupt output asserted

9	FAULT	FAULT	Pending fault with caution disabled	■ NMI output (fault upgrade) ■ Depends on fault reaction configuration
10	FAULT	NORMAL	Input fault recovered	Reactions deasserted

Note: The FMU_PCLOCK can be used to permanently lock the FMU. If the FMU is permanently locked, the FMU cannot enter the INIT work mode. If you want the FMU to enter INIT work mode again, you need to reset the chip.

29.3.9. Principles of fault reactions

Fault reacts usually follows the following principles:

- When FMU works in NORMAL work mode. If multi-faults occurred. the faults which will transform the work mode to FAULT will be firstly responded
- When FMU works in CAUTION work mode. At this time if a fault occurred. If the caution status of the fault is disabled but the fault is not masked. The fault will make the work mode of FMU transform to FAULT. FMU will react this fault preferentially. In this case, CAUTION timer will be stopped until the fault recovery (FMU leaves FAULT work mode).

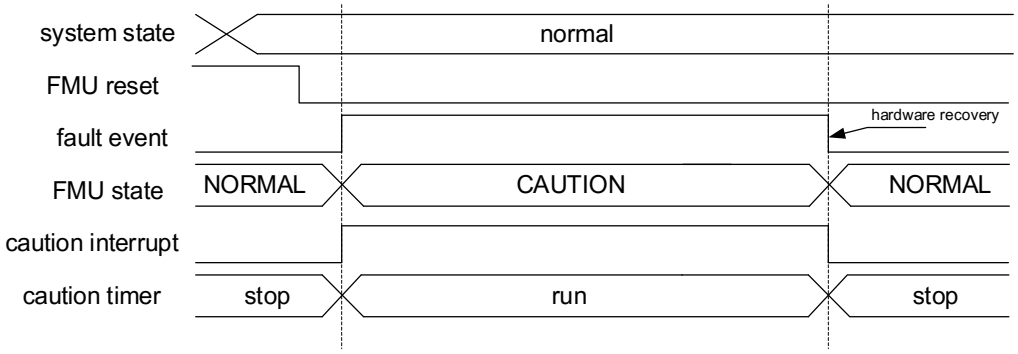
If the FMU is working in CAUTION work mode, another fault occurs and the caution timer of the fault is enabled, the caution timer will not reload and start again.

29.3.10. Fault recovery

Hardware recovery

Figure 29-3. FG fault hardware recovery shows a hardware recovery process. FMU works in NORMAL work mode. The system occurred a fault, the fault is unmasked and its caution timer is enabled. The work mode of FMU will transform to CAUTION. If the FAULT signal is no longer asserted within the timeout of the caution timer, the work mode of FMU will transform to NORMAL. During this process, the system can run normally without being affected by this fault.

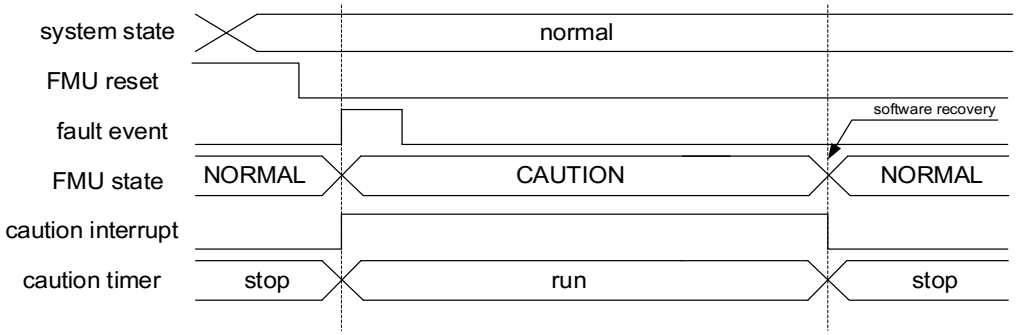
Figure 29-3. FG fault hardware recovery



Software recovery

Figure 29-4. FG fault software recovery shows a software recovery process. FMU works in NORMAL work mode. The system occurred a fault, the fault is unmasked and it's caution timer is enabled. The work mode of FMU will transform to CAUTION. If the FAULT signal is no longer asserted within the timeout of the caution timer, because of the software configuration recovered the fault. Then the work mode of FMU will transform to NORMAL. During this process, the system can run normally without being affected by this fault.

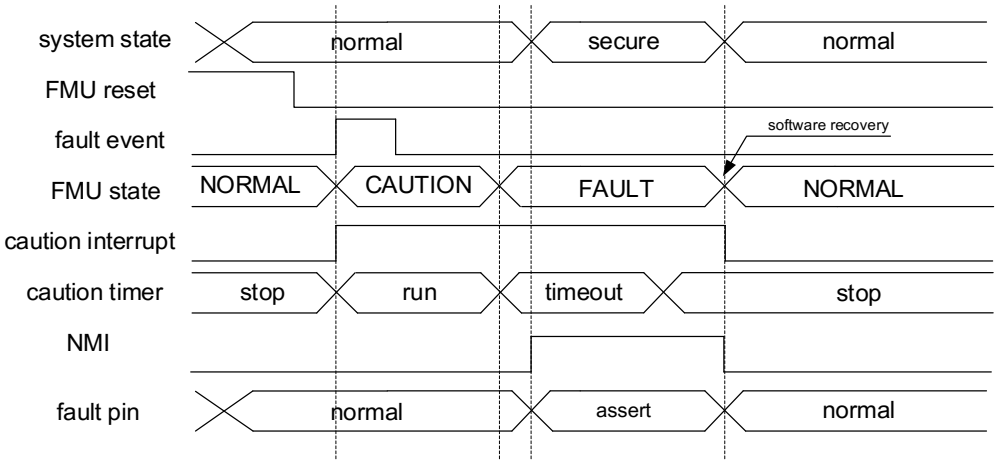
Figure 29-4. FG fault software recovery



CAUTION to FAULT work mode recovery

Figure 29-5. FG fault CAUTION to FAULT work mode recovery shows a CAUTION to FAULT work mode recovery process. The system occurred a fault, the fault is unmasked and it's caution timer is enabled. The work mode of FMU will transform to CAUTION. If the interrupt of the caution status is enabled, an interrupt request signal will be generated. Then if the caution timer timeout has occurred, the work mode of FMU will transform to FAULT. If NMI is enabled, the NMI request will be asserted. Then the system will work in a secure state to manage the fault. If the fault signal is no longer asserted or the fault is restored by the software configuration, the work mode of FMU will transform to NORMAL. Then the system can run normally again.

Figure 29-5. FG fault CAUTION to FAULT work mode recovery



29.3.11. Fatal fault pin signal

There are two ways to use FFIO as a dedicated fatal fault feedback pin:

- When PJ10 and PJ11 are used as FFIO pins, set FFIOMOD to 1, refer to ***Table 2-12. OB_CLTCFG option bytes fields description***;
- Set the SMC bit of the GPIOx_SMCCTL register corresponding to FFIO to 1, refer to ***Table 8-8. Pin behavior when FMU fault in run mode when the fault reaction is not reset***.

There are 3 work states in FMU which can be indicated by FPO0 and FPO1 bits in FMU_FPCTL, these work states are:

- Faulty: FMU is in FAULT work mode and FFIO timer is running.
- Non-faulty: FMU is in CAUTION or NORMAL work mode and FFIO timer is not running.
- Initialization: FMU is in INIT work mode and FFIO timer is not running.

FFIO timer in bi-stable work mode

FFIO can run in bi-stable work mode by configuring FOMSET[2:0] bits in FMU_CFG.

FFIO timer will start when FFIO run in bi-stable work mode and all of the following conditions are true:

- FFIO timer is not running before.
- FMU works in FAULT work mode.
- FOSCFG[1:0] bits are not set to 0b01 which will make FFIO output low level always.

If FFIO timer is running before and a new fault occurs:

- If FMU is working in INIT work mode, a new occurred fault can not restart FFIO timer.
- If FMU is working in NORMAL or CAUTION work mode, a new occurred fault whose caution is disabled will transform the work mode of FMU to FAULT and restart FFIO timer. if the fault's caution state is enabled, it will transform the work mode of FMU to CAUTION firstly, and can not restart FFIO timer.

FFIO timer will be stopped and reinitialized by FMU when all of the following conditions are true:

- Timeout occurred which is configured by FOT[13:0] bits in FMU_FOT.
- When all faults have been recovered which will transform the work mode of FMU to NORMAL.

Further introduction about FFIO signal

FOMSET[2:0] bits can be used to configure FFIO work in different mode, Bi-Stable or time-switching, and FOPCFG bit can be used to set FFIO polarity.

If FFIO signal is configured to indicate a fault state or software request FFIO signal indicate a state. And the state has a minimum output time value T, and the time T can be set by

FMU_FOT. The FFIO will always indicate the signal for a period of T, even if the software clears the signal before this period of time is over.

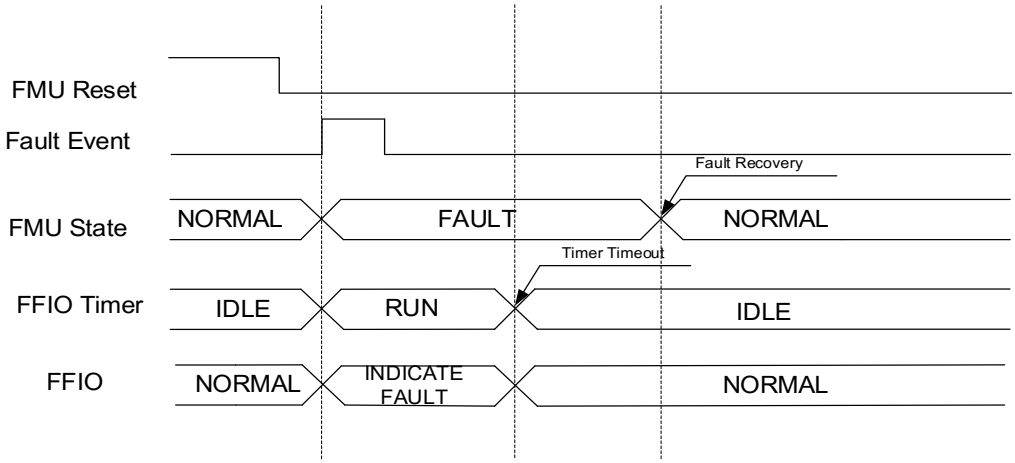
If software configures FFIO to always low by FOSCFG[1:0], the FFIO will be low level in time T. The value of T can be set by FMU_FOT. At this time FMSM will transform work mode i independently. Once the time is over T, FFIO will be driven by the work mode of FMU. as long as the timer is running, software can not make the signal of FFIO to high by transform FMSM to INIT work mode. When the time value of T is over and FMSM is in NORMAL work mode, FFIO signal will be driven to high. If during the time of T, a new fault has happened and requested FFIO signal, the FFIO timer will rerun.

When the FFIO signal is configured for FMSM control, the FFIO signal fault indication is only controlled by the time threshold "T".

During the "T" time period, the FFIO signal remains in the fault indication state. When the FFIO timer times out, the FFIO signal returns to normal.

The time "T" can be configured via FMU_FOT, with a maximum configurable duration of 16633 μs.

Table 29-6. FFIO signal indicating fault status in GD32A711/A712 products is solely controlled by the FFIO timer



Bi-stable protocol

Table 29-5. Bi-stable mode and FFIO signal

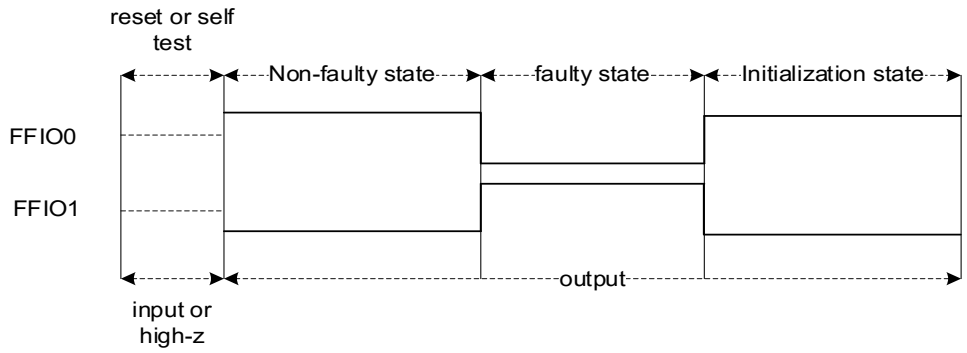
mode	FFIO[1:0] (FOPCFG is 0)	FFIO[1:0] (FOPCFG is 1)
Non-faulty	0b01	0b 10
Faulty	0b10	0b01
Reset	high-impedance[1]	high-impedance
Initialization (FOMCFG is 1 and FOSEN is 0)	high-impedance[2]	high-impedance

mode	FFIO[1:0] (FOPCFG is 0)	FFIO[1:0] (FOPCFG is 1)
Initialization (When FOMCFG is 1 and FOSEN is 1)	0b01	0b10

1. Device can configure output value.
2. If FFIO signals are not active (FOSEN is 0), FFIO signals will stay in a high-impedance state after reset lifts.

Figure 29-7. Bi-stable protocol shows FMU state and FFIO output.

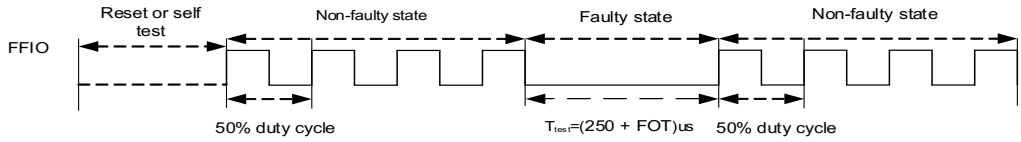
Figure 29-7. Bi-stable protocol



Time-switching protocol

FFIO is toggled between logic level 0 and logic level 1 with a defined frequency which configured via FOPRE[3:0]. This frequency modulation protocol will be violated when the FMU enters the faulty state. *Figure 29-8. Time-switching protocol* shows FMU state and FFIO toggle output in time-switching protocol.

Figure 29-8. Time-switching protocol



29.3.12. FMU initialization preprogramming

FMU initialization preprogramming is that configure INIT work mode and then transform the work mode of FMU to INIT, which will prepare for FMU initialization.

Note for FMU initialization preprogramming

For FMU initialization the following matters should be noted:

- After reset FMU is temporarily locked. The FMU_TCLOCK register can be used to unlock FMU. Then the work mode of FMU can transform to INIT as set.
- The FMU should be in NORMAL work mode before transform the work mode of FMU to

INIT.

- When FMU is in INIT work mode, the initialize value did not really save to initialization registers. In order to save the initialize value, you should transform the work mode of FMU to NORMAL, then the value of initialization registers will be saved voluntary.
- If FMU works in INIT work mode and then the initialization timer is expires, the work mode of FMU will transform to NORMAL. At this time, FMU will set the value of FMU_CFG to the timeout value of initialization timer, and FMU will set other initialization registers to reset value. *Table 29-6. Initialization registers* shows related initialization registers.

Table 29-6. Initialization registers

Register	Reset value
FMU_CFG	0x0000 0000
FMU_FGCFG	0x0000 FFFF
FMU_FGSUBSRCFG	0x0000 0000
FMU_FGEN	0x0000 0000
FMU_FGCMTEN	0x0000 0000
FMU_FGCMTVAL	0x0000 0000
FMU_IMTVAL	0x0000 0005
FMU_CMINTREN	0x0000 0000
FMU_FMNMIINTREN	0x0000 0000
FMU_FPOEN	0x0000 0000
FMU_GPOEN	0x0000 0000
FMU_FGTSENx	Refer to section 35.4.33

FMU transform to INIT work mode programming

1. Configure FMU_TCLOCK register to unlock temporary lock state.
2. Read FCSTAT[2:0] bits to get FMU current work mode. If FMU is in INIT work mode, nothing needs to do. If FMU is in NORMAL work mode execute step 3. If FMU is in CAUTION or FAULT work mode, all faults need to be recovered first and then execute step 2 again.
3. Execute EOP1 operation, refer to *Execute operations*.
4. Read OPSTAT[1:0] bits to get operation result, if the operation result of EOP1 is successful. Then work mode of FMU will transform to INIT. If not, FMU_PCLOCK register may be in permanent lock state. It is recommended to reset FMU and then go to execute step 1 again.

29.3.13. FMU initialization programming

If you want FMU to work as expected, you should first initialize FMU.

Note for FMU initialization

When initialize FMU, if a FG fault is enabled but all reactions of this group is disabled. No reaction will happen. If the reactions of FG is enabled, but the FG is disabled no work mode transformation will happen.

Initialization programming

The main matters about FMU FG initialization programming are shown as below:

1. Use FMU_FGCFG to configure fault recovery type.
2. 3 types of fault reaction exist, at least one of them should be enabled:
   - FMU_FGSUBSRCFG can configure fault reaction to subsystem reset.
   - FMU_FMNMIINTREN can enable NMI interrupt request as fault reaction.
   - FMU_FPOEN can enable FFIO signal as fault reaction.
3. Configure the FMU_FGCMTVAL. register to set the timeout value of CAUTION work mode.
4. FMU_FGCMTEN can be used to enable CAUTION work mode, which will make FMU transform to CAUTION work mode firstly.
5. FMU_CMINTREN can be used to enable CAUTION work mode interrupt request.
6. FMU_FGEN can be used to enable group fault monitor.

Note:

- If subsystem reset is set to fault reaction. NMI interrupt request or FFIO signal fault reaction should be enable, or CAUTION work mode should be enabled in step 4.
- CAUTION work mode timeout value should not over FBMU timeout value, if not FBMU will reset chip when a fault occur, FBMU's timeout value is 0x69780.

FMU INIT work mode initialization

Two initialization registers can use to INIT work mode:

- FMU_IMTVAL can be used to configure the timeout value that FMU can stay in INIT work mode;
- FMU_IMTINTREN can be used to enable a INIT work mode timeout interrupt.

29.3.14. Transform FMU to NORMAL work mode

Transform FMU work mode from INIT to NORMAL which can save configuration register, and when a fault occurred, the work mode of FMU will transform to CAUTION or FAULT. Unmasked faults will be reacted.

Note for transform FMU to NORMAL work state

When FMU is in INIT work mode and try to lock configuration register, it will take effect when FMU leave INIT work mode, whether the FMU actively exited INIT work mode or passively exited INIT work mode (INIT work mode timeout event happened).

If FMU is permanently locked in INIT work mode, and then FMU transform to NORMAL. Then the work mode of FMU can not transform to INIT work mode unless reset.

Transform to NORMAL work mode programming

1. Read FMU_STAT to get current FMU work mode.
   - If the FMU works mode is INIT, turn to step 2.
   - If the FMU works mode is CAUTION or FAULT, turn to step 4.
   - If the FMU works mode is NORMAL, go to step 5.
2. Execute the EOP2 operation, refer to ***Execute operations***.
3. Read FMU_CTL to get operation result, if the operation result of EOP2 is successful, the work mode of FMU will transform to NORMAL. If not you should execute EOP2 again.
4. Recovery fault and then turn to step 1.
5. Configuration lock register can prevent configuration later, there are temporary lock and permanent lock register.

29.3.15. Manage software recoverable faults

Once FMU has be initialized, the work mode of FMU will transform as configuration to manage the fault. If there is unrecovered fault, FMU_FGSTAT register can be used to get FG state. The steps below are recommended to recover a software recoverable FG fault.

1. Deal the reason which causes FG fault.
2. Write FMU_FGKEY register to unlock FMU_STAT register.
3. Write FMU_STAT register will clear FG fault status, the EOP12 operation will be executed by FMU automatically.
4. Read FMU_CTL register to get operation status, if the operation result of EOP12 is not successful, turn to step 2; if the operation result of EOP12 is successful turn to step 5; if operation is on going turn to step 4.
5. Read FMU_STAT, if there is no fault then fault software recovery is successful, if there is fault exists turn to step 2.

Note: If more than one fault state want to be recover, the faults should be recover one by one.

29.3.16. Hold status clean

Execute operation 13 (EOP13) will clean hold-status, if the operation result is not successful, EOP13 should be executed again.

29.3.17. Execute operations

Write to FMU_CTL register can execute some operations such as transform FMU to INIT work mode or configure operation status. Operation will not be executed when operation status is on going, there are some operations which can be executed only unlock firstly, and when operation is executed it will be locked again. To execute an operation the steps below are recommended:

1. Read FMU_CTL to get current operation state.
2. If operation state is on going turn to step 1.
3. Write FMU_CTLKEY register to unlock if the operation to be executed needs to be unlocked first.
4. Write FMU_CTL register to execute an operation.
5. Get operation result by read FMU_CTL register.
6. If operation is on going, turn to step 5.
7. The operation result will be one of idle / on going / success / not success.

29.4. Register definition

FMU base address: 0x4002 3400

Table 29-7. FMU register overview

Register name	Symbol	Address offset	Reset value
Control register	FMU_CTL	0x00	0x0000 00C0
Control key register	FMU_CTLKEY	0x04	0x0000 0000
Configuration register	FMU_CFG	0x08	0x0000 0000
FG configuration register	FMU_FGCFG	0x1C	0x0000 0FFF
FG subsystem reset configuration register	FMU_FGSUBSRCFG	0x4C	0x0000 0000
FG status register	FMU_FGSTAT	0x80	0x0000 0000
FG key register	FMU_FGKEY	0x90	0x0000 0000
FG enable register	FMU_FGEN	0x94	0x0000 0000
FG CAUTION work mode timeout enable register	FMU_FGCMTEN	0xA4	0x0000 0FFF
FG CAUTION work mode timeout value register	FMU_FGCMTVAL	0xB4	0x0003 A980
INIT work mode timeout value register	FMU_IMTVAL	0xB8	0x0000 0005
Fault pin control register	FMU_FPCTL	0xBC	0x0000 0000
Status register	FMU_STAT	0xC0	0x0000 0050
NORMAL transform to CAUTION work mode hold status register	FMU_NTCHSTAT	0xC4	0x0000 0000
CAUTION transform to FAULT work mode hold status register	FMU_CTFHSTAT	0xC8	0x0000 0000
NORMAL transform to FAULT work mode hold status register	FMU_NTFHSTAT	0xCC	0x0000 0000
FAULT transform to CAUTION work mode hold status register	FMU_FTCHSTAT	0xD0	0x0000 0000
Fake FG fault number configuration register	FMU_FFGNCFG	0xDC	0x0000 0100
Interrupt request pending register	FMU_INTRPD	0xE0	0x0000 0000
INIT work mode timeout interrupt request enable register	FMU_IMTINTREN	0xE4	0x0000 0000
Temporary configuration lock register	FMU_TCLOCK	0xF0	0x0000 0000
Permanent configuration lock register	FMU_PCLOCK	0xF4	0x0000 0000
Fault output timer register	FMU_FOT	0xF8	0x0000 0000
FG CAUTION work mode interrupt request enable register	FMU_CMINTREN	0xFC	0x0000 0000
FG FAULT work mode NMI interrupt request enable register	FMU_FMNMIINTREN	0x10C	0x0000 0000
FG FFIO pin output enable register	FMU_FPOEN	0x11C	0x0000 0000

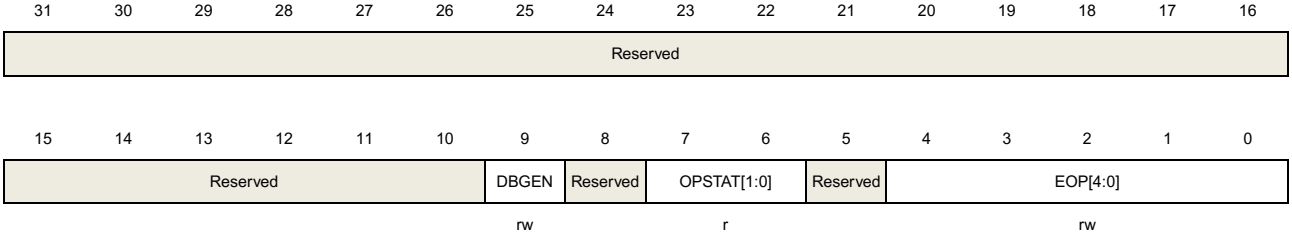
Register name	Symbol	Address offset	Reset value
FG GFIO pin output enable register	FMU_GPOEN	0x120	0x0000 0000
Fault holding timer output enable register	FMU_FHTOEN	0x124	0x0000 0000
FG CAUTION work mode counter register	FMU_CMCNT	0x12C	0x0000 0000
FG INIT work mode counter register	FMU_IMCNT	0x134	0x0000 0000
Fatal fault output counter register	FMU_FOCNT	0x138	0x0000 0000
General fault output counter register	FMU_GOCNT	0x13C	0x0000 0000
FG trigger source enable register 0	FMU_FGTSEN0	0x140	0xFFFF FFFF
FG trigger source enable register 1	FMU_FGTSEN1	0x144	0xFF9F FFFF
FG trigger source enable register 2	FMU_FGTSEN2	0x148	0xFFFF FFFF
FG trigger source enable register 3	FMU_FGTSEN3	0x14C	0xFFFF FFFF
FG trigger source enable register 4	FMU_FGTSEN4	0x150	0xFFFF FFFF
FG trigger source status register	FMU_FGTSSTATx(x=0...4)	0x160+4*x(x=0…4)	0x0000 0000
FG software triggered fault enable register	FMU_FGSWTENx(x=0...2)	0x180+4*x(x=0...2)	0x0000 0000
TMR enable and status register	FMU_TMRES	0x240	0x0000 0015

29.4.1. Control register (FMU_CTL)

Address offset: 0x00

Reset value: 0x0000 00C0

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9	DBGEN	FMU debug mode enable bit This bit is set and cleared by software. If FMU debug mode is enabled, FMU will hold its state when chip in debug state. 0: FMU debug mode is disabled 1: FMU debug mode is enabled **Note:** FMU will hold state in debug mode, but FBMU will not hold state in debug mode. So when FMU is in debug mode, FBMU can still trigger a system reset.
8	Reserved	Must be kept at reset value.
7:6	OPSTAT[1:0]	Operation status bits

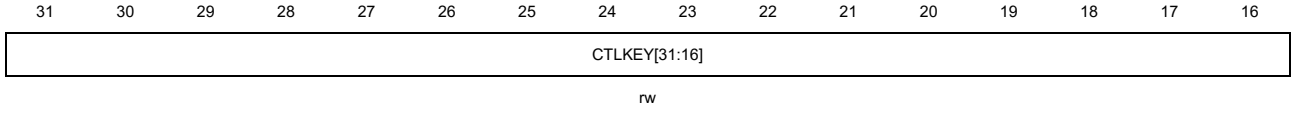
These bits can be read by software, and can be cleared by EOP15 operation.
00: Idle
01: On going
10: Not success
11: Success

5	Reserved	Must be kept at reset value.
4:0	EOP[4:0]	Execute operation bits These bits can execute some operations such as transform the work mode of FMU to INIT or configure operation status. 00000: No execute operation (EOP0) 00001: Execute operation which will transform FMU to INIT work mode (EOP1) 00010: Execute operation which will transform FMU to NORMAL work mode (EOP2) …. 01011: Reserved 01100: Execute clean a FG fault flag in FMU_FGSTAT register operation is on going (EOP12), the behavior of FMU, do not set it. 01101: Execute operation which will clean the hold status register (EOP13) 01110: INIT work mode timeout is on going(EOP14), the behavior of FMU, do not set it. 01111: Execute operation which will set OPSTAT[1:0] to idle (0b00) (EOP15) 10000…10100: Reserved 10101…11110: Execute operation which will set OPSTAT[1:0] to not success, should not write these values. 11111: Reserved **Note:** ■ If one operation is on going, write to these bits will be ignored. ■ If one operation is success executed, these bits will be set to EOP0(0b00000). ■ These behavior below will set OPSTAT[1:0] to not success: – If not unlock FMU_FGKEY first, then write to FMU_FGSTAT register – FMU is not in the NORMAL work mode or the configuration not be unlocked, then execute EOP1 – If not unlock FMU_CTLKEY first, then execute EOP1 or EOP2 operation

29.4.2. Control key register (FMU_CTLKEY)

Address offset: 0x04
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTLKEY [15:0]															
rw															

Bits	Fields	Descriptions
31:0	CTLKEY[31:0]	Unlock operation control key These bits are set and cleared by software. Write specific value to this register can unlock operation EOP1 and EOP2, other execute operation not need unlock. 0x85A670B9: Unlock EOP1 0x4C30765D: Unlock EOP2

29.4.3. Configuration register (FMU_CFG)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						GOSEN	FOSEN	FOSCFG[1:0]		GOSCFG[1:0]		FOPRE[3:0]			
						rw	rw	rw		rw					rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				FOMCFG	GOPCFG	FOPCFG	FOMSET[2:0]			GOMSET[2:0]			Reserved		
				rw	rw	rw	rw			rw					

Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25	GOSEN	General fault I/O output pin (GFIO) signal enable bit This bit is set and cleared by software. 0: Disable general fault output pin signals 1: Enable general fault output pin signals **Note:** General fault output pin signals will be high-impedance state if general fault output signals are disabled.
24	FOSEN	Fatal fault I/O output pin (FFIO) signals enable bit This bit is set and cleared by software. 0: Disable fatal fault output pin signals 1: Enable fatal fault output pin signals **Note:** Fatal fault output pin signals will be high-impedance state if fatal fault output signals are disabled.
23:22	FOSCFG[1:0]	Fatal fault I/O output pin (FFIO) signals configuration bits These bits are set and cleared by software. Configuration fatal fault output pin signals are controlled by FMU's FMSM or not when fatal fault output signals work in bi-stable and time switching mode. 00 / 10: Fatal fault output pin signals are controlled by FMU's FMSM

		01: Fatal fault output pin signals are low 11: Before a fault occurs even the fault is not enable fatal fault output pin is high, once fault occurred it will be controlled by FMU's FMSM. **Note:** When fatal fault output pin timer is running, write to these bits will be ignored.
21:20	GOSCFG[1:0]	General fault I/O output pin signal configuration bits These bits are set and cleared by software. Configuration general fault output pin signals are controlled by FMU's FMSM or not when general fault output signals work in fault output mode. 00 / 10: General fault output pin signals are controlled by FMU's FMSM 01: General fault output pin signals are low 11: Before a fault occurs even the fault is not enable general fault output pin is high, once fault occurred it will be controlled by FMU's FMSM. **Note:** When general fault output pin timer is running write to these bits will be ignored.
19:16	FOPRE[3:0]	Fatal fault output pin signals time switching clock prescaler configuration bits These bits are set and cleared by software. The clock source is SECURE_CLK. 0000: Configure to SECURE_CLK / 2 0001: Configure to SECURE_CLK / 4 0010: Configure to SECURE_CLK / 8 0011: Configure to SECURE_CLK / 16 0100: Configure to SECURE_CLK / 32 0101: Configure to SECURE_CLK / 64 0110: Configure to SECURE_CLK / 128 0111: Configure to SECURE_CLK / 256 1000: Configure to SECURE_CLK / 512 1001: Configure to SECURE_CLK / 1024 1010: Configure to SECURE_CLK / 2048 1011: Configure to SECURE_CLK / 4096 Others value are reserved.
15:12	Reserved	Must be kept at reset value.
11	FOMCFG	Fatal fault output pin (FFIO) signals work mode configuration bit This bit is set and cleared by software. Controlled fatal fault output pin signals work mode is consistent with FMU in non-faulty state or not when FMU is in INIT work mode. 0: Not consistent with FMU in non-faulty state 1: Consistent with FMU in non-faulty state
10	GOPCFG	General fault output pin (GFIO) signal polarity configuration bit This bit is set and cleared by software. When general fault output pin work in fault output mode configure it polarity. 0: When FG fault occurred GFIO0 is low.

		1: When FG fault occurred GFIO0 is high.
9	FOPCFG	Fatal fault output pin (FFIO) signals polarity configuration bit This bit is set and cleared by software. When fatal fault output pin work in bi-stable mode configure it polarity. 0: When FG fault occurred FFIO1 is high, and FFIO0 is low. 1: When FG fault occurred FFIO1 is low, and FFIO0 is high.
8:6	FOMSET[2:0]	Fatal fault output pin (FFIO) signals work mode set bits These bits are set and cleared by software. Control the pin signal's protocol. 000 / 001: Reserved 010: Work mode set to bi-stable protocol 011: Work mode set to time switching protocol 100: Reserved 101: Test 0 (FFIO1 is an output; FFIO0 is an input, configured by FMU_FPCTL) 110: Test 1 (FFIO1 is an output; FFIO0 is an output, configured by FMU_FPCTL) 111: Test 2 (FFIO1 is an input; FFIO0 is an output, configured by FMU_FPCTL)
5:3	GOMSET[2:0]	General fault output pin (GFIO) signal work mode set bits These bits are set and cleared by software. Control the pin signal's protocol. 000 / 001: Reserved 010: Work mode set to fault output mode 011: Reserved 100: Reserved 101: Test 0 (GFIO0 is an output, configured by FMU_FPCTL) 110: Reserved 111: Test 1 (GFIO0 is an input, configured by FMU_FPCTL)
2:0	Reserved	Must be kept at reset value.

29.4.4. FG configuration register (FMU_FGCFG)

Address offset: 0x1C

Reset value: 0x0000 0FFF

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				FGCFG11	FGCFG10	FGCFG9	FGCFG8	FGCFG7	FGCFG6	FGCFG5	FGCFG4	FGCFG3	FGCFG2	FGCFG1	FGCFG0
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.

11:0	FGCFGx	FG fault recovery type configuration bit x (x=0..11) These bits are set and cleared by software. 0: Configure fault recovery type to hardware recovery 1: Configure fault recovery type to software recovery **Note:** When FMU is in INIT work mode configuration is effective, if INIT work mode timeout event happened the value of these bits will be set to reset value by FMU.

29.4.5. FG subsystem reset configuration register (FMU_FGSUBSRCFG)

Address offset: 0x4C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								RST11[1:0]		RST10[1:0]		RST9[1:0]		RST8[1:0]	
								rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RST7[1:0]		RST6[1:0]		RST5[1:0]		RST4[1:0]		RST3[1:0]		RST2[1:0]		RST1[1:0]		RST0[1:0]	
rw		rw		rw		rw		rw		rw		rw		rw	

Bits	Fields	Descriptions
31:24	Reserved	Must be kept at reset value.
23:22	RST11[1:0]	FG subsystem reset reaction configuration 11 bits These bits are set and cleared by software. Refer to RST0[1:0] description
21:20	RST10[1:0]	FG subsystem reset reaction configuration 10 bits These bits are set and cleared by software. Refer to RST0[1:0] description
19:18	RST9[1:0]	FG subsystem reset reaction configuration 9 bits These bits are set and cleared by software. Refer to RST0[1:0] description
17:16	RST8[1:0]	FG subsystem reset reaction configuration 8 bits These bits are set and cleared by software. Refer to RST0[1:0] description
15:14	RST7[1:0]	FG subsystem reset reaction configuration 7 bits These bits are set and cleared by software. Refer to RST0[1:0] description
13:12	RST6[1:0]	FG subsystem reset reaction configuration 6 bits These bits are set and cleared by software. Refer to RST0[1:0] description
11:10	RST5[1:0]	FG subsystem reset reaction configuration 5 bits

		These bits are set and cleared by software. Refer to RST0[1:0] description
9:8	RST4[1:0]	FG subsystem reset reaction configuration 4 bits These bits are set and cleared by software. Refer to RST0[1:0] description
7:6	RST3[1:0]	FG subsystem reset reaction configuration 3 bits These bits are set and cleared by software. Refer to RST0[1:0] description
5:4	RST2[1:0]	FG subsystem reset reaction configuration 2 bits These bits are set and cleared by software. Refer to RST0[1:0] description
3:2	RST1[1:0]	FG subsystem reset reaction configuration 1 bits These bits are set and cleared by software. Refer to RST0[1:0] description
1:0	RST0[1:0]	FG subsystem reset reaction configuration 0 bits These bits are set and cleared by software. Control if subsystem reset will as the reaction when a FG fault happened. 00 / 11: Configure subsystem reset not as a reaction of a FG fault 01: Configure subsystem reset as a reaction of a FG fault 10: Reserved **Note:** When FMU is in INIT work mode configuration is effective, if INIT work mode timeout event happened the value of these bits will be set to reset value by FMU.

29.4.6. FG status register (FMU_FGSTAT)

Address offset: 0x80

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				FGSTAT11	FGSTAT10	FGSTAT9	FGSTAT8	FGSTAT7	FGSTAT6	FGSTAT5	FGSTAT4	FGSTAT3	FGSTAT2	FGSTAT1	FGSTAT0
				rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	FGSTATx	FG fault status bit x (x=0..11) When FG fault occurred corresponding bits will set, for software recoverable fault

clean these bits may recovery fault, for hardware recoverable fault FMU will clean fault status, if software clean a hardware recoverable fault status bit, FMU not recovery the fault and the fault status will be not effective. About software recovery fault refer to ***Software recovery***.

0: No FG fault happened

1: FG fault happened and not recovered

29.4.7. FG key register (FMU_FGKEY)

Address offset: 0x90

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FGKEY[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FGKEY[15:0]															
rw															

Bits	Fields	Descriptions
31:0	FGKEY[31:0]	Unlock FG status register key These bits are set by software. Before write FMU_FGSTAT should write this register for unlock first. After unlock should write FMU_FGSTAT immediately, because FMU_FGSTAT may be locked soon. 0xDA3778FB: Unlock FMU_FGSTAT register Other value: Reserved

29.4.8. FG enable register (FMU_FGEN)

Address offset: 0x94

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				FGEN11	FGEN10	FGEN9	FGEN8	FGEN7	FGEN6	FGEN5	FGEN4	FGEN3	FGEN2	FGEN1	FGEN0
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	FGENx	FG enable bit x (x=0..11)

These bits are set and cleared by software. When FG fault is enabled and a fault occurred which is in the source of this FG fault will transform FMU works mode to FAULT or CAUTION.

0: Disable FG

1: Enable FG

Note: When FMU is in INIT work mode configuration is effective, if INIT work mode timeout event happened the value of these bits will be set to reset value by FMU.

29.4.9. FG CAUTION work mode timeout enable register (FMU_FGCMTEN)

Address offset: 0xA4

Reset value: 0x0000 0FFF

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				FGCMTEN11	FGCMTEN10	FGCMTEN9	FGCMTEN8	FGCMTEN7	FGCMTEN6	FGCMTEN5	FGCMTEN4	FGCMTEN3	FGCMTEN2	FGCMTEN1	FGCMTEN0
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	FGCMTENx	FG fault caution work mode timeout enable bit x (x=0..11) These bits are set and cleared by software. If enabled FMU will transform work mode to caution state when a unmasked fault occurs, otherwise FMU transform to FAULT state. 0: Disable FG fault caution state timeout 1: Enable FG fault caution state timeout **Note:** When FMU is in INIT work mode, the configuration is effective, if INIT work mode timeout event happened the value of these bits will be set to reset value by FMU.

29.4.10. FG CAUTION work mode timeout value register (FMU_FGCMTVAL)

Address offset: 0xB4

Reset value: 0x0003 A980

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FGCMTVAL[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

FGCMTVAL[15:0]
rw

Bits	Fields	Descriptions
31:0	FGCMTVAL[31:0]	FG caution work mode timeout value bits These bits are set and cleared by software. These bits set the timeout value (FGCMTVAL * T_{SECURE_CLK}) that FMU transform to CAUTION and stay in CAUTION work mode, once the time is over and the fault is not recovery FMU will transform to FAULT work mode. **Note:** ■ When FMU is in INIT work mode, the configuration is effective, if INIT work mode timeout event happened the value of these bits will be set to reset value by FMU. ■ CAUTION work mode timeout value should not over FBMU timeout value or FBMU reset chip when a fault occur, FBMU's timeout value is 0x69780.

29.4.11. INIT work mode timeout value register (FMU_IMTVAL)

Address offset: 0xB8
Reset value: 0x0000 0005

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													IMTVAL[2:0]		
													rw		

Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2:0	IMTVAL[2:0]	FMU INIT work mode timeout value bits These bits are set and cleared by software. These bits set the timeout value (T_{SECURE_CLK} * $2^{(IMTVAL + 13)}$) that FMU transform to INIT and stay in INIT work mode, once the time is over FMU will transform to NORMAL work mode. **Note:** ■ When FMU is in INIT work mode, the configuration is effective, if INIT work mode timeout event happened the value of these bits will be set to reset value by FMU. ■ Not accessible when INIT work mode timeout is on going (EOP14). ■ If INIT work mode timeout event happen the value of FMU_CFG will be set to this register.

29.4.12. Fault pin control register (FMU_FPCTL)

Address offset: 0xBC
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

The FMU_FPCTL register allows the following operations typically in the NORMAL state:

When FMU in NORMAL work mode FMU_FPCTL register can control the pin like below:

- When set FOMSET[2:0] to 0b101 or 0b110 FMU fault out pin signal will work in test0 or test1 mode, in this mode FMU_FPCTL can control FFIO1 output level.
- When set FOMSET[2:0] to 0b110 or 0b111 FMU fault out pin signal will work in test1 or test2 mode, in this mode FMU_FPCTL can control FFIO0 output level.
- Read FMU_FPCTL can get the state of FFIO0 and FFIO1.

The table *Table 29-8. Test mode bi-stable encoding* shows bi-stable encoding.

Table 29-8. Test mode bi-stable encoding

Mode = FMU_CFG[FOMSET]	FFIO0	FFIO1
Test0	input	output
Test1	output	output
Test2	output	input

Note:

Because of the resynchronization stage of the FFIO interface, a latency of some SECURE_CLK cycles is exist. For example write FMU_FPCTL register to set FPO0 bit to 0, and read immediately, the read result may not 0, to get the expectation value need wait some SECURE_CLK cycles.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									GPI	FPI1	FPI0	Reserved	GPO	FPO1	FPO0
									r	r	r		rw	rw	rw

Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	GPI	GFIO signal input status This bit is read by software. significant when GOSEN is set. 0: GFIO signal input status is low 1: GFIO signal input status is high
5	FPI1	FFIO1 signal input status This bit is read by software. significant when FOSEN is set.

0: FFIO1 signal input status is low
1: FFIO1 signal input status is high

4	FPI0	FFIO0 signal input status This bit is read by software. significant when FOSEN is set. 0: FFIO0 signal input status is low 1: FFIO0 signal input status is high
3	Reserved	Must be kept at reset value.
2	GPO	GFIO output signal configuration bit This bit is set and cleared by software. 0: Configure GFIO to low level 1: Configure GFIO to high level **Note:** ■ If INIT work mode timeout event happened the value of these bits will be set to reset value by FMU. ■ When set GOMSET[2:0] to configure GFIO0 work in output mode, the configuration is significant
1	FPO1	FFIO1 output signal configuration bit This bit is set and cleared by software. 0: Configure FFIO1 to low level 1: Configure FFIO1 to high level **Note:** ■ If INIT work mode timeout event happened the value of these bits will be set to reset value by FMU. ■ When set FOMSET[2:0] to configure FFIO1 work in output mode, the configuration is significant
0	FPO0	FFIO0 output signal configuration bit This bit is set and cleared by software. 0: Configure FFIO0 to low level 1: Configure FFIO0 to high level **Note:** ■ If INIT work mode timeout event happened the value of these bits will be set to reset value by FMU. ■ When set FOMSET[2:0] to configure FFIO0 work in output mode, the configuration is significant.

29.4.13. Status register (FMU_STAT)

Address offset: 0xC0
Reset value: 0x0000 0050

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									GPSTAT	FPSTAT[1:0]		FSTAT	FCSTAT[2:0]		
									r	r		r	r		

Bits	Fields	Descriptions
31:7	Reserved	Must be kept at reset value.
6	GPSTAT	GFIO signal states (fault drive signal) When GOSEN is set this bit is show GFIO signal states. 0: GFIO is low 1: GFIO is high
5:4	FPSTAT[1:0]	FFIO signal states (fault drive signals) When FOSEN is set these bits are show FFIO signal states. 00: FFIO1 is low; FFIO0 is low. 01: FFIO1 is low; FFIO0 is high. 10: FFIO1 is high; FFIO0 is low. 11: FFIO1 is high; FFIO0 is high.
3	FSTAT	FMU faulty state bit This bit is show whether FMU is in faulty state. 0: FMU is not in faulty state 1: FMU is in faulty state
2:0	FCSTAT[2:0]	FMU current work mode state bits These bits show the current work mode state of FMU. 000: FMU current work mode is NORMAL 001: FMU current work mode is INIT 010: FMU current work mode is CAUTION 011: FMU current work mode is FAULT Others: Reserved

29.4.14. NORMAL transform to CAUTION work mode hold status register (FMU_NTCHSTAT)

Address offset: 0xC4

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				NTCHSTA	NTCHSTA	NTCHSTA	NTCHSTA	NTCHSTA	NTCHSTA	NTCHSTA	NTCHSTA	NTCHSTA	NTCHSTA	NTCHSTA	NTCHSTA

	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0
	r	r	r	r	r	r	r	r	r	r	r	r

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	NTCHSTATx	NORMAL transform to CAUTION work mode hold status bit x (x=0...11) These bits can be cleared by software, how to clear can refer to ***Hold status clean***. After these bits are cleared if a FG fault occurred and transform FMU works mode from NORMAL to CAUTION, these bits will be set and indicate the fault source. Only used for testing and debugging.

29.4.15. CAUTION transform to FAULT work mode hold status register (FMU_CTFHSTAT)

Address offset: 0xC8

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				CTFHSTAT11	CTFHSTAT10	CTFHSTAT9	CTFHSTAT8	CTFHSTAT7	CTFHSTAT6	CTFHSTAT5	CTFHSTAT4	CTFHSTAT3	CTFHSTAT2	CTFHSTAT1	CTFHSTAT0
				r	r	r	r	r	r	r	r	r	r	r	r

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	CTFHSTATx	CAUTION transform to FAULT work mode hold status bit x (x=0...11) These bits can be cleared by software, how to clear can refer to ***Hold status clean***. After these bits are cleared, if a FG fault occurred and transform the works mode of FMU from CAUTION to FAULT, these bits will be set and indicate the fault source. Only used for testing and debugging.

29.4.16. NORMAL transform to FAULT work mode hold status register (FMU_NTFHSTAT)

Address offset: 0xCC

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				NTFHSTAT11	NTFHSTAT10	NTFHSTAT9	NTFHSTAT8	NTFHSTAT7	NTFHSTAT6	NTFHSTAT5	NTFHSTAT4	NTFHSTAT3	NTFHSTAT2	NTFHSTAT1	NTFHSTAT0
				r	r	r	r	r	r	r	r	r	r	r	r

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	NTFHSTATx	NORMAL transform to FAULT work mode hold status bit x (x=0...11) These bits can be cleared by software, how to clear can refer to ***Hold status clean***. After these bits are cleared, if a FG fault occurred and transform FMU works mode from NORMAL to FAULT, these bits will be set and indicate the fault source. Only used for testing and debugging.

29.4.17. FAULT transform to CAUTION work mode hold status register (FMU_FTCHSTAT)

Address offset: 0xD0

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				FTCHSTAT11	FTCHSTAT10	FTCHSTAT9	FTCHSTAT8	FTCHSTAT7	FTCHSTAT6	FTCHSTAT5	FTCHSTAT4	FTCHSTAT3	FTCHSTAT2	FTCHSTAT1	FTCHSTAT0
				r	r	r	r	r	r	r	r	r	r	r	r

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	FTCHSTATx	FAULT transform to CAUTION work mode hold status bit x (x=0...11) These bits can be cleared by software, how to clear can refer to ***Hold status clean***. After these bits are cleared, if a FG fault occurred and transform FMU works mode from FAULT to CAUTION, these bits will be set and indicate the fault source, for details about fault sources can refer to ***Table 29-1 FMU fault group definition***.. Only used for testing and debugging.

29.4.18. Fake FG fault number configuration register (FMU_FFGNCFG)

Address offset: 0xDC

Reset value: 0x0000 0100

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						FFGEN[1:0]		NUM[7:0]							
						rw		w							

Bits	Fields	Descriptions
31:10	Reserved	Must be kept at reset value.
9:8	FFGEN[1:0]	Fake FG fault enable configure bits These bits are set by software. 00 / 11: Reserved 01: Disable fake FG fault 10: Enable fake FG fault
7:0	NUM[7:0]	Injection fake FG fault number configuration bits These bits are set by software, and the value correspondence FG fault. The fake FG fault is just like real fault and can trigger system reaction. 00000000: Injection fake FG fault number 0 (FG0) 00000001: Injection fake FG fault number 1 (FG1) ….. 00001011: Injection fake FG fault number 11 (FG11) Others: Reserved

29.4.19. Interrupt request pending register (FMU_INTRPD)

Address offset: 0xE0

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													NMIINTRPD	CAUINTRPD	INITINTRPD
													r	r	rc_w1

Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2	NMIINTRPD	NMI interrupt request pending bit

		This bit is set and cleared with FMSM. 0: NMI request is not pending 1: NMI request is pending.
1	CAUINTRPD	CAUTION work mode interrupt request pending bit This bit is set and cleared with FMSM. 0: CAUTION work mode interrupt request is not pending 1: CAUTION work mode interrupt request is pending.
0	INITINTRPD	INIT work mode timeout interrupt request pending bit This bit is set by hardware, and cleared to 0 by writing 1 to it. 0: INIT work mode timeout interrupt request is not pending 1: INIT work mode timeout interrupt request is pending.

29.4.20. INIT work mode timeout interrupt request enable register (FMU_IMTINTREN)

Address offset: 0xE4
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITTIE
															rw

Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	INITTIE	INIT work mode timeout interrupt enable bit This bit is set and cleared by software. 0: Disable INIT work mode timeout interrupt 1: Enable INIT work mode timeout interrupt

29.4.21. Temporary configuration lock register (FMU_TCLOCK)

Address offset: 0xF0
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TCLK[8:0]								
							rw								

Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8:0	TCLK[8:0]	INIT work mode transition temporary lock bits These bits are set and cleared by software, running in the privilege mode. 0xDA: Unlock INIT work mode transition Others: Lock INIT work mode transition **Note:** This register can be written in any FMU works mode, but it will take effect only when FMU is in NORMAL work mode.

29.4.22. Permanent configuration lock register (FMU_PCLOCK)

Address offset: 0xF4
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							PCLK[8:0]								
							rw								

Bits	Fields	Descriptions
31:9	Reserved	Must be kept at reset value.
8:0	PCLK[8:0]	INIT work mode transition permanent lock bits These bits are set and cleared by software which running in the privilege mode. 0xFF: Lock INIT work mode transition Others: No effect **Note:** This register can be written in any FMU works mode, but it will take effect only when FMU is in NORMAL work mode. Once INIT work mode transition is permanent locked FMU can not transform to INIT work mode until reset.

29.4.23. Fault output timer register (FMU_FOT)

Address offset: 0xF8
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		FOT[13:0]													
		rw													

Bits	Fields	Descriptions
31:14	Reserved	Must be kept at reset value.
13:0	FOT[13:0]	Fault output(FFIO) timer configure bits These bits are set and cleared by software. The shortest fault output T_{est} (T_{est} = (250+ FOT) * (48000 / SECURE_CLK)) (FOT is measured in "us" and SECURE_CLK is measured in" KHz")

29.4.24. FG CAUTION work mode interrupt request enable register (FMU_CMINTREN)

Address offset: 0xFC
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				CMINTREN11	CMINTREN10	CMINTREN9	CMINTREN8	CMINTREN7	CMINTREN6	CMINTREN5	CMINTREN4	CMINTREN3	CMINTREN2	CMINTREN1	CMINTREN0
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	CMINTRENx	CAUTION work mode interrupt request enable bit x (x=0...11) 0: Disable FG CAUTION work mode interrupt request 1: Enable FG CAUTION work mode interrupt request **Note:** When FMU is in INIT work mode configuration is effective, if INIT work mode timeout event happened the value of these bits will be set to reset value by FMU.

29.4.25. FG FAULT work mode NMI interrupt request enable register (FMU_FMNMIINTREN)

Address offset: 0x10C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				FMNMIINTREN11	FMNMIINTREN10	FMNMIINTREN9	FMNMIINTREN8	FMNMIINTREN7	FMNMIINTREN6	FMNMIINTREN5	FMNMIINTREN4	FMNMIINTREN3	FMNMIINTREN2	FMNMIINTREN1	FMNMIINTREN0
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	FMNMIINTRENx	FAULT work mode NMI interrupt request enable bit x (x=0..11) 0: Disable FG fault work mode NMI interrupt request 1: Enable FG fault work mode NMI interrupt request **Note:** When FMU is in INIT work mode configuration is effective, if INIT work mode timeout event happened the value of these bits will be set to reset value by FMU. Effective when the FMUNMIIE bit in the SYSCFG_CFG register is enabled.

29.4.26. FG FFIO pin output enable register (FMU_FPOEN)

Address offset: 0x11C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				FPOEN11	FPOEN10	FPOEN9	FPOEN8	FPOEN7	FPOEN6	FPOEN5	FPOEN4	FPOEN3	FPOEN2	FPOEN1	FPOEN0
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	FPOENx	FAULT work mode FFIO signaling enable bit x (x=0...11) 0: In Bi-Stable fault output mode, FFIO signaling and FIFS assertion are disabled; in other fault-output modes, FFIO signaling is enabled and FIFS assertion is disabled. 1: In all FFIO output mode FFIO signaling and FIFS assertion are enabled **Note:** ■ When FMU is in INIT work mode configuration is effective, if INIT work mode timeout event happened the value of these bits will be set to reset value by FMU.

- If FIFS assertion is disabled, at this time if a fault happened FMU will transform work mode to FAULT and FBMU may generate a system reset.

29.4.27. FG GFIO pin output enable register (FMU_GPOEN)

Address offset: 0x120
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				GPOEN11	GPOEN10	GPOEN9	GPOEN8	GPOEN7	GPOEN6	GPOEN5	GPOEN4	GPOEN3	GPOEN2	GPOEN1	GPOEN0
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:12	Reserved	Must be kept at reset value.
11:0	GPOENx	FAULT work mode GFIO signaling enable bit x (x=0…11) 0: In fault-output mode GFIO signaling is enabled and FIFS assertion is disabled. 1: In fault-output mode GFIO signaling and FIFS assertion are enabled **Note:** ■ When FMU is in INIT work mode configuration is effective, if INIT work mode timeout event happened the value of these bits will be set to reset value by FMU. ■ If FIFS assertion is disabled, at this time if a fault happened FMU will transform work mode to FAULT and FBMU may generate a system reset.

29.4.28. Fault holding timer output enable register (FMU_FHTOEN)

Address offset: 0x124
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				FHTOEN11	FHTOEN10	FHTOEN9	FHTOEN8	FHTOEN7	FHTOEN6	FHTOEN5	FHTOEN4	FHTOEN3	FHTOEN2	FHTOEN1	FHTOEN0
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions

31:12	Reserved	Must be kept at reset value.
11:0	FHTOENx	FAULT work states holding timer output enable bit x (x=0…11) Can write only when FMU is in the INIT work mode. This bit enables or disables the GPOs holding output status indication of the advanced timer and general level0 timer when the FMU enters the FAULT work states. 0: Disable FAULT work mode holding timer output 1: Enable FAULT work mode holding timer output

29.4.29. FG CAUTION work mode counter register (FMU_CMCNT)

Address offset: 0x12C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMCNT[31:16]															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMCNT[15:0]															
r															

Bits	Fields	Descriptions
31:0	CMCNT[31:0]	CAUTION work mode counter. Time base on SECURE_CLK periods.

29.4.30. FG INIT work mode counter register (FMU_IMCNT)

Address offset: 0x134

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IMCNT[31:16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IMCNT[15:0]															
rw															

Bits	Fields	Descriptions
31:0	IMCNT[31:0]	INIT work mode counter. Time base on SECURE_CLK periods.

29.4.31. Fatal fault output counter register (FMU_FOCNT)

Address offset: 0x138

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FOCNT[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FOCNT [15:0]															
r															

Bits	Fields	Descriptions
31:0	FOCNT[31:0]	FFIO fault output counter. Time base on SECURE_CLK periods. Once FFIO starts output, the counter increments from 0.

29.4.32. General fault output counter register (FMU_GOCNT)

Address offset: 0x13C
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GOCNT[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GOCNT [15:0]															
r															

Bits	Fields	Descriptions
31:0	GOCNT[31:0]	GFIO fault output timer counter. Time base on SECURE_CLK periods. Once GFIO starts output, the counter increments from 0.

29.4.33. FG trigger source enable register (FMU_FGTSENx) (x=0...4)

Address offset: 0x140 + 4 * x (x=0…4)
Reset value: 0xFFFF FFFF (x=0, 2, 3, 4), 0xFF9F FFFF (x=1)

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FGTSEN 31	FGTSEN3 0	FGTSEN2 9	FGTSEN2 8	FGTSEN2 7	FGTSEN2 6	FGTSEN2 5	FGTSEN2 4	FGTSEN2 3	FGTSEN2 2	FGTSEN2 1	FGTSEN2 0	FGTSEN1 9	FGTSEN1 8	FGTSEN1 7	FGTSEN1 6
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FGTSEN 15	FGTSEN 14	FGTSEN 13	FGTSEN 12	FGTSEN 11	FGTSEN 10	FGTSEN 9	FGTSEN 8	FGTSEN 7	FGTSEN 6	FGTSEN 5	FGTSEN 4	FGTSEN 3	FGTSEN 2	FGTSEN 1	FGTSEN 0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:0	FGTSENy	FG trigger source enable bit y (y=0…31), about fault number can refer to ***Table 29-1 FMU fault group definition***. 0: Disable FG trigger source 1: Enabled FG trigger source **Note**: When FMU is in INIT work mode configuration is effective, if INIT work mode timeout event happened the value of these bits will be set to reset value by FMU.

29.4.34. FG trigger source status register (FMU_FGTSSTATx) (x=0...4)

Address offset: 0x160 + 4 * x(x=0…4)
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FGTSSTAT31	FGTSSTAT30	FGTSSTAT29	FGTSSTAT28	FGTSSTAT27	FGTSSTAT26	FGTSSTAT25	FGTSSTAT24	FGTSSTAT23	FGTSSTAT22	FGTSSTAT21	FGTSSTAT20	FGTSSTAT19	FGTSSTAT18	FGTSSTAT17	FGTSSTAT16
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FGTSSTAT15	FGTSSTAT14	FGTSSTAT13	FGTSSTAT12	FGTSSTAT11	FGTSSTAT10	FGTSSTAT9	FGTSSTAT8	FGTSSTAT7	FGTSSTAT6	FGTSSTAT5	FGTSSTAT4	FGTSSTAT3	FGTSSTAT2	FGTSSTAT1	FGTSSTAT0
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bits	Fields	Descriptions
31:0	FGTSSTATy	FG trigger source status bit y (y=0..31) , about fault number can refer to ***Table 29-1 FMU fault group definition***. 0: FG trigger source is not triggered 1: FG trigger source is triggered

29.4.35. FG software triggered fault enable register (FMU_FGSWTENx) (x=0...2)

Address offset: 0x180 + 4 * x
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FGSWTEN7[1:0]		FGSWTEN6[1:0]		FGSWTEN5[1:0]		FGSWTEN4[1:0]		FGSWTEN3[1:0]		FGSWTEN2[1:0]		FGSWTEN1[1:0]		FGSWTEN0[1:0]	
rw		rw		rw		rw		rw		rw		rw		rw	

Bits	Fields	Descriptions

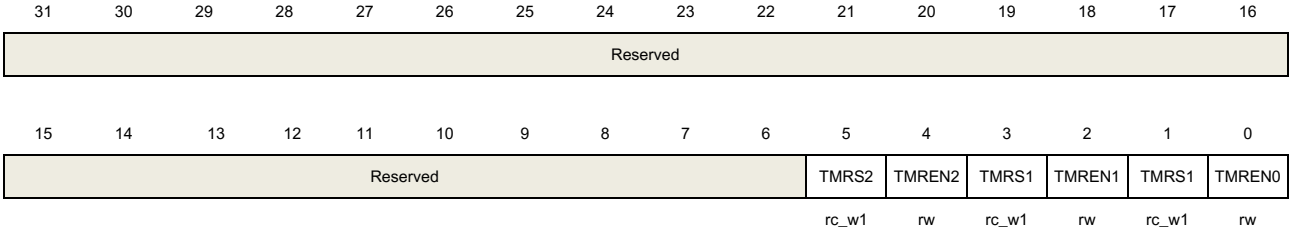
31:16	Reserved	Must be kept at reset value.
15:0	FGSWTENy[1:0]	Software triggered fault enable bit y (y=0...7) 00: Disable software triggered fault 01: Enable software triggered fault 10: Reserved 11: Disable software triggered fault

29.4.36. TMR enable and status register (FMU_TMRES)

Address offset: 0x240

Reset value: 0x0000 0015

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:6	Reserved	Must be kept at reset value.
5	TMRS2	FMU_FGCFG and FMU_FGEN registers TMR error status (TMR error status bit 2) 0: Normal 1: Error
4	TMREN2	TMR error output enable bit 2 0: Disable 1: Enable
3	TMRS1	FOSEN, FOSCFG[1:0], GOSEN and GOSCFG[1:0] bits TMR error status (TMR error status bit 1) 0: Normal 1: Error
2	TMREN1	TMR error output enable bit 1 0: Disable 1: Enable
1	TMRS0	DBGEN register TMR error status (TMR error status bit 0) 0: Normal 1: Error

0	TMREN0	TMR error output enable bit 0 0: Disable 1: Enable

30. Debug system (DBGSYS)

30.1. Overview

The chip's architectural design encompasses both debug and trace modules. For specific details about the debug and trace capabilities supported by Cortex-M7 core clusters, please refer to the following section for detailed description.

Access to the debug components supported by the Cortex-M7 core is facilitated through the Arm DAP controller-based architecture. The DAP controller operates in parallel with the system JTAGC, with both controllers sharing the JTAG port and JTAG instruction set.

The JTAG Data Communication (JDC) module provides the transfer of register data between the APB and JTAG domains.

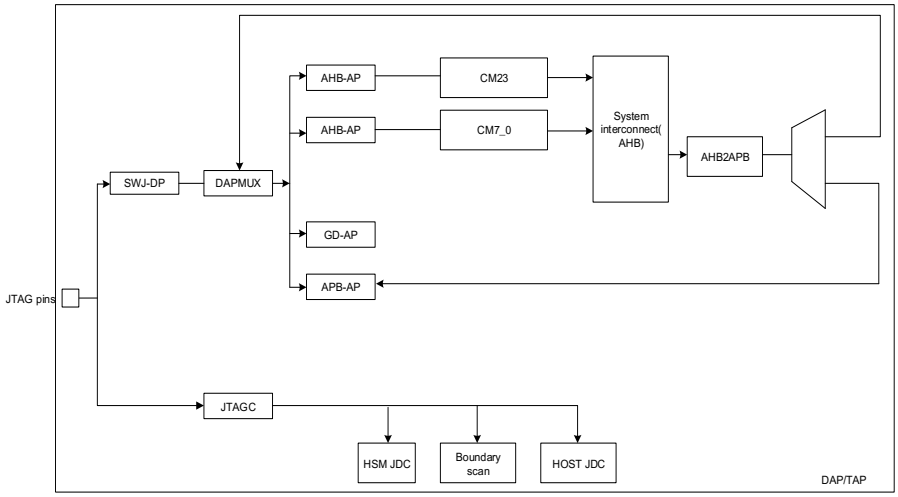
30.2. Characteristics

- Debug and test control
  - Complies with IEEE 1149.1 standard to implement debug control and test control
  - Support stopping points, starting points, breakpoints, and watchpoints to implement debugging and run control
  - Cortex-M7 core has eight instruction comparators and a watchpoint unit with four watchpoints
  - Ability to view and modify all memory-mapped areas that are not otherwise blocked
- Cross-triggering
  - Controls the run-control options of the cores based on the other cores, the cores include Cortex-M7 and Cortex-M23
- Security and safety
  - The DAP/TAP interface that controls all the debug features and is gated by Cortex M23 JDC module
- Support ITM trace

30.3. DAP architecture

The *Figure 30-1. Debug access port architecture of GD32A711 / A712* depict the GD32A711/A712 DAP (Debug Access Port) architecture.

Figure 30-1. Debug access port architecture of GD32A711 / A712



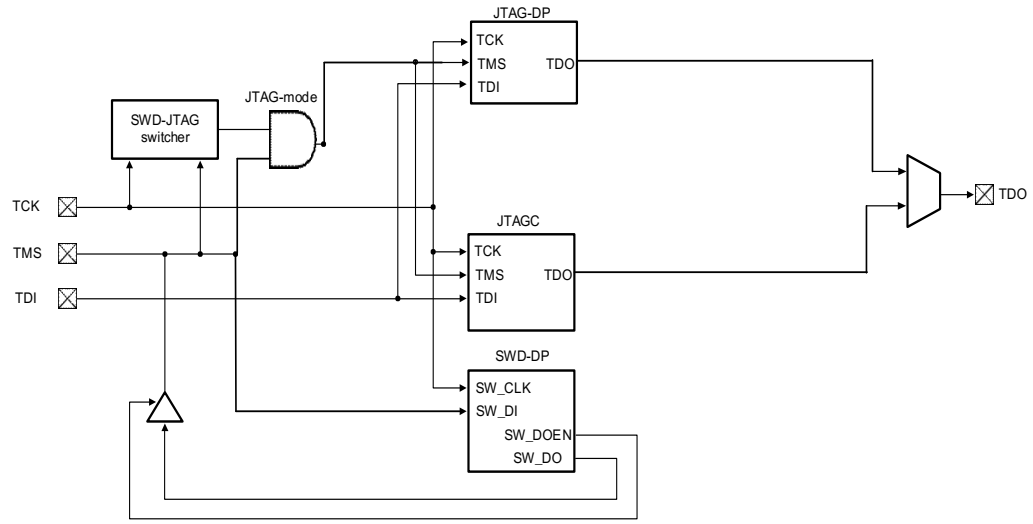
Note: Debug components other than the Cortex-M23 core can be accessed through a software path (CORE > AHB > APB > DAPMUX > APs), which means specifying memory-mapped addresses directly without the need for challenge-response procedures.

30.4. Debug

30.4.1. Test access port(TAP)

The ***Figure 30-2. TAP connectivity block diagram*** provides a detailed view of TAP(Test Access Port) connectivity. SWD/JTAG SELECT, SWDP, and JTAG-DP are integral components of the ARM DAP.

Figure 30-2. TAP connectivity block diagram



The debug port initially enters a standard JTAG mode upon reset. To transition it into SWD mode, follow the JTAG-to-SWD change sequence described below. Once the mode has been

successfully changed, any unused debug pins can be reconfigured for their alternative multiplexed functions.

JTAG-to-SWD change sequence:

1. Transmit more than 50 TCK cycles with TMS (SWDIO) set to 1.
2. Send the 16-bit sequence on TMS (SWDIO) as 0111_1001_1110_0111 (with the MSB transmitted first).
3. Follow this with more than 50 TCK cycles with TMS (SWDIO) set to 1.

To switch from SWD to JTAG operation:

1. Transmit more than 50 TCK cycles with TMS (SWDIO) set to 1.
2. Send the 16-bit sequence on TMS (SWDIO) as 0011_1100_1110_0111 (with the MSB transmitted first).
3. Follow this with more than 5 TCK cycles with TMS (SWDIO) set to 1

30.4.2. Debug access port(DAP)

DAP provides a way to access the system buses through an AHB master interface. It also allows external access ports to connect to the internal DAP bus. For more details about DAP TAP, please refer to the Arm Debug Interface Architecture Specification document.

In this chip, the following features are supported:

- The debugger can access all the memory units and registers in the system through the AHB slave ports of Core CM7_0 and CM23.
- The debugger can see a consistent view of the memory map for each core
- The debugger needs to use the corresponding AHB_AP port to access the AHB-S port of each subsystem
- A core can access the debug components of another core through a DAPMUX that connects to the bus interconnect
- The DAP bus that is exposed to external access ports contains AHB_AP and GD_AP.
  - GD_AP, which contains system-level JTAG status and control registers. These registers can be used for miscellaneous control and status operations.
  - APB_AP, which uses an APSEL value of 0h to access any debug component that are mapped to the APB bus.

Debug Access Port (DAP) is used to access registers within the DAP and in the connected system. The DAP Instruction Register (DAP IR) shares the same space with the system JTAG Instruction Register (JTAGC IR). *Table 30-1. DAP instruction* provides an overview of DAP instructions.

Table 30-1. DAP instruction

9-bit IR instruction value	Data register	Description
0b11111000	ABORT	Access the AP Abort Register in the DP, to force an AP abort.
0b11111010	DPACC	Initiate a DP access, to access a DP register.
0b11111011	APACC	Initiate a AP access, to access a AP register.
0b11111110	IDCODE	Device identification. The Device ID Code value enables a debugger to identify the Debug

		Port to which it is connected.
0b11111111	BYPASS	Bypasses the device, by providing a direct path between DBGTDI and DBGTDO

Table 30-2. DAP master address mapping of GD32A711 / A712 display the decoding of APSEL.

Table 30-2. DAP master address mapping of GD32A711 / A712

DAP master	DAP component	DAPBUS address	System memory map address
0	APB-AP	0x00000000	0x40250000
1	AHB-AP(CM7_0)	0x01000000	0x40250100
2	AHB-AP(CM23)	0x02000000	0x40250200
4	GD_AP	0x04000000	0x40250400

Note: Simultaneous access to GD_AP from both the core and the debugger is not allowed. Attempting to do so may result in unpredictable responses to core-initiated transactions, such as incorrect data reads, failed attempts to write to registers, or transfer errors. Debugger-initiated transactions, on the other hand, will proceed as expected.

30.4.3. Application cores debugging from first instruction

The MCU supports application cores debugging from the first instruction in the following cases:

- MCU power up
- Exit from standby mode
- System reset and subsystem reset

When MCU power up or system reset occurs, the timing of debugging from the first instruction is shown in *Figure 30-3. Debugging from first instruction sequence*.

Figure 30-3. Debugging from first instruction sequence

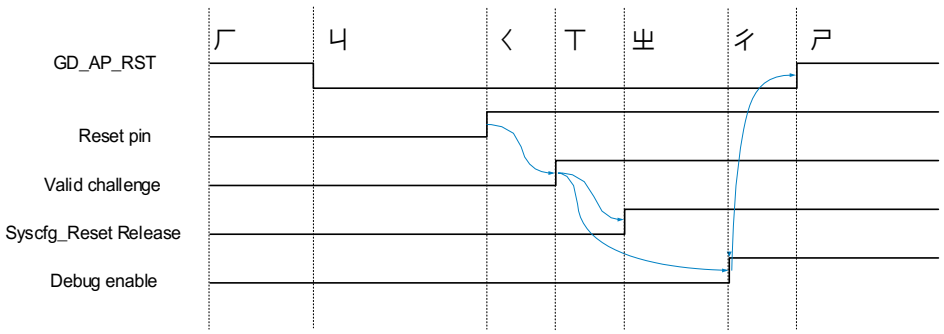


Table 30-3. Debugging from first instruction step description

Step name	Description
①	The debugger maintains the chip in a reset state by activating the reset pin
②	The debugger sets up to retain the reset state of the application core.
③	The debugger releases the chip's reset pin, but the application core remains in a reset state

Step name	Description
④	BAF writes valid status information to GD_AP.
⑤	BAF release Syscfg reset
⑥	Debugger provides expected response, challenge-response pass
⑦	Debugger configures the core's debug and releases the core's resets

After chip power-up or system reset, debugging from first instruction, the following operation need to be enabled:

1. Configure mcu support debugging from the first instruction mode, the steps are as follows
   (1) CM7_0_RST bit of GD_AP_RST register are set to 1
   (2) CM7_0_RST bit of GD_AP_RST register are set to 0

Note: Start debugging with the first instruction during the low power debugging protocol, WAITRST bit of GD_AP_LPCTL1 register is set to 1.

2. The HSM core booting, when the reset pin is released
3. Start debugger challenge-response (CR)
4. BAF writes Syscfg reset release
5. CM7_0_RST bit of GD_AP_RST register is set to 1 to release cpu.

30.4.4. Debug authentication

To execute the SWJ-DP sequence for debugging authentication, the debugger:

1. Keep getting the CHLGRDY bit of GD_AP_AUTHN_STAT register until it enters the challenge ready state.
2. Get the challenge value of GD_AP_KEYCHLGx (x = 0...3) registers, create a 128-bit response value and write it to GD_AP_KEYRESPx (x = 0...3) registers.
3. Configure the AUTHREQ bit of GD_AP_AUTHN_CTL register to indicates the response is ready.
4. Read and check the GD_AP_AUTHN_STAT register to determine whether the authentication was successful.

If the authentication process is successful and the debugger has written access to Debug enable register (GD_AP_DBGEN), the challenge or response is considered successful too.

30.4.5. Debug in low-power mode

If debugger handshake is enabled, the debugger must enter or exit Standby mode in the following order:

1. Turn on DAP, set up the debugger connection, and set up the debug system for the required operations.
2. Write 1 to LPDBGEN bit in GD_AP_LPCTL0 register gate entry into Standby mode with the debugger handshake.
3. Write 1 to PWD_MONDIS bit in CLTCFG_POR_CTL0 register to disable POR_WDG for monitoring the entry or exit from Standby mode. This is needed for low-power debug handshake because debugger configurations may take longer than the POR_WDG

threshold levels and cause a false POR_WDG event.

4. Write 1 to CLTCFG_SUBSR_CTL0[16] in the CLTCFG. This disables padkeeping on standby entry itself without depending on any software. This is necessary for low-power debug because otherwise padkeeping on TDO would block debugger communication. In other cases, CLTCFG_SUBSR_CTL0 [16] should be set as 0 before standby entry.
5. Read RCU_LPDBGSTAT[0] field to check whether low-power debug is enabled on low-power entry.

■ If low-power debug and TPIU are both enabled, trace flush begins, and the debugger confirms the following:
- Low-power debug traces
- DAP-related configuration reception and context saving.

After this, the chip goes into Standby mode

■ If TPIU is off:
- The debugger confirms DAP-related configuration reception and context saving by writing 1 to RST_BLK bit in low power control register 1(GD_AP_LPCTL1). After this, the chip goes into Standby mode

If low-power debug is off, the chip goes into Standby mode without waiting for debugger confirmation.

On any wakeup event, the chip starts the Standby mode exit sequence described in the PMU chapter. After the chip leaves Standby mode, the debugger connection is resumed. By this time, the debugger knows that it has enabled the low-power debug handshake. In this case, the debugger must poll SYSRST bit in GD_AP_STAT register to see whether:

- The debug infrastructure has been reset.

- The DAP connection can be set up.

After DAP is turned on (the debugger connection is set up), the debugger:

a. Resets the debug and trace attributes using the fields in GD_AP_DBGEN register.

b. Writes 1 to WAITRST bit in GD_AP_LPCTL1 register after the debugger trace context is restored.

The chip leaves Standby mode. If low-power debug is not set up, the chip leaves Standby mode without waiting for the debugger to write to WAITRST bit in GD_AP_LPCTL1 register.

6. Reset the debug and trace configuration in GD_AP register descriptions after the chip leaves Standby mode.

Note: If the chip awakens from Standby mode due to reset pin, the debugger needs to execute a Challenge-Response (CR) procedure once more.

30.4.6. Debug resets

The reset results caused by different reset types are shown in ***Table 30-4. Reset results for different reset types***.

Table 30-4. Reset results for different reset types

Reset type	Result description
Power reset	Reset all debug system
System reset	Reset all debug system except JTAGC

The method to generate a debug subsystem reset is shown in *Table 30-5. Generates different reset methods*.

Table 30-5. Generates different reset methods

Generate reset type	Method
Subsystem reset	Set SUBSYSRST bit of GD_AP_CTL register to 1
System reset	Set SYSRST bit of GD_AP_CTL register to 1

Note:

1. The debugger will loss connect when debugger provide the system reset to the mcu.

30.4.7. Debug across device lifecycles

Debug access to the system is available during the early lifecycles and subsequently depends on the OPT configuration settings.

Once debugging is configured as 'Trust,' system access is permitted only after a successful authorization step between the debugger and the system.

Authorization provisions are in place to allow debugging for both Cortex M23 and Cortex M7 cores. The *Table 30-6. Debug access* illustrates debug access based on various configuration bits.

Table 30-6. Debug access

Lifecycle	Debug access
FA	Open(debug is always possible)
GD_PROD	Open(debug is always possible)
CUST_DEL	Open(debug is always possible)
OEM_PROD	Trust(debug is possible after successful authentication, challenge/response handshake with correct credentials)
IN_FIELD	Trust(debug is possible after successful authentication, challenge/response handshake with correct credentials)

30.5. JTAG controller(JTAGC)

30.5.1. Overview

JTAGC and the TAP controller (DAP's JTAG-DP) are in a parallel relationship in terms of connection method. The chip fully complies with the JTAG standard and appears as a single TAP in the JTAG chain. The instruction register (IR) of JTAGC is 9 bits in length. The IR

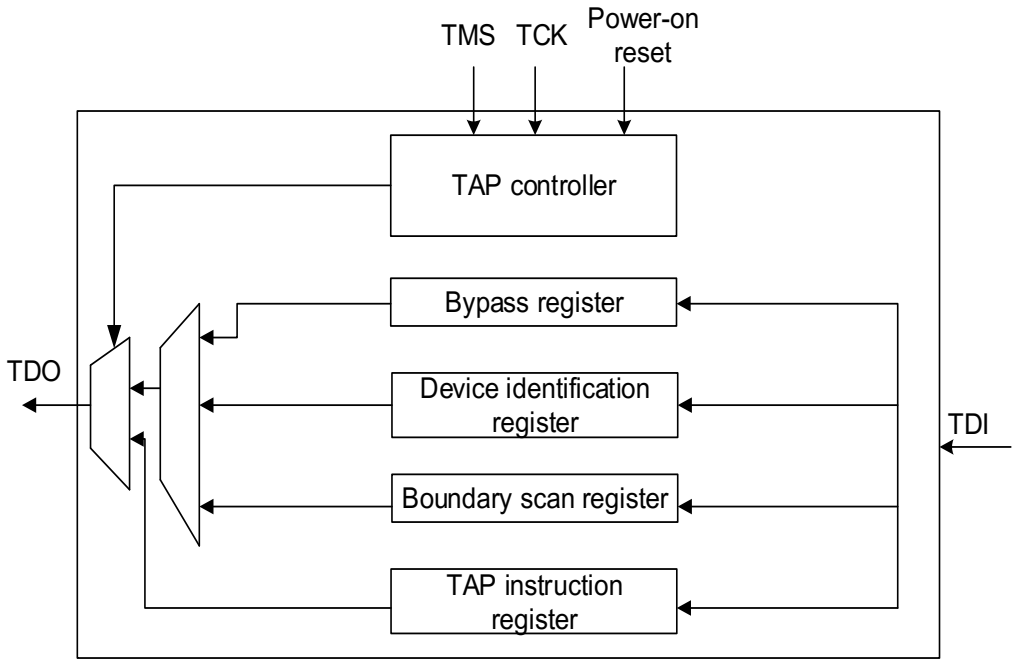
instructions of JTAGC are partially the same as those of the DAP controller. Four instructions are used in DAP, while nine instructions are used in JTAGC. Depending on the IR instruction selected, TAP outputs (TPOs) operate differently. Refer to ***Figure 30-2. TAP connectivity block diagram***.

30.5.2. JTAGC architecture

The JTAGC module provides a method to test the chip functionality and connectivity without affecting the system logic in non-test mode. The testing process uses the boundary scan technique, which is a testing method defined by the IEEE 1149.1-2001 standard. The input and output data of the JTAGC module are transmitted in serial format.

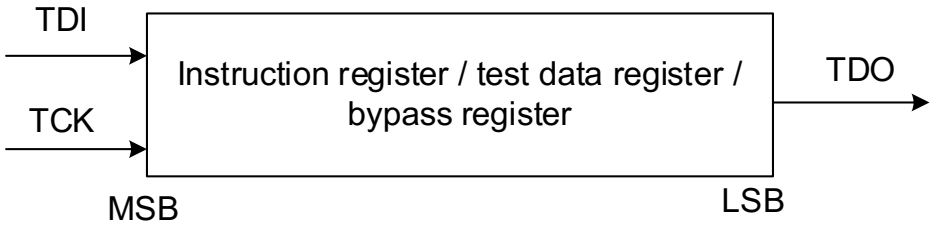
The ***Figure 30-4. JTAG Controller block*** is a simplified block diagram of the JTAG Controller (JTAGC) block

Figure 30-4. JTAG Controller block



In the selected register, data moves from TDI to TDO, starting from the least significant bit,, as depicted in the ***Figure 30-5. Data transfer shift***. This procedure is applicable to the instruction register, test data registers, and the bypass register.

Figure 30-5. Data transfer shift



30.5.3. JTAGC IR codes

The JTAGC IR codes shown in *Table 30-7. JTAG instruction register*.

Table 30-7. JTAG instruction register

9-bit IR instruction value	Instruction register
0b00000_xxxx	Boundary Scan
0b00000_xxxx	SCAN/MBIST
0b01001_1111	HSM JDC
0b01010_1111	HOST JDC
0b11111_1000	ABORT (Arm)
0b11111_1010	DPACC (Arm)
0b11111_1011	APACC (Arm)
0b11111_1110	IDCODE (Arm)
0b11111_1111	BYPASS
Others	Reserved

30.5.4. JTAGC reset and bypass mode

The JTAGC block will enter reset mode in the following two situations:

- Power-on reset
- TMS input stays high for enough consecutive rising edges of TCK to move the TAP controller state machine to the Test-Logic-Reset state.

Activating power-on reset causes an asynchronous entry to the reset state. When a reset occurs, the following behavior occurs in JTAGC

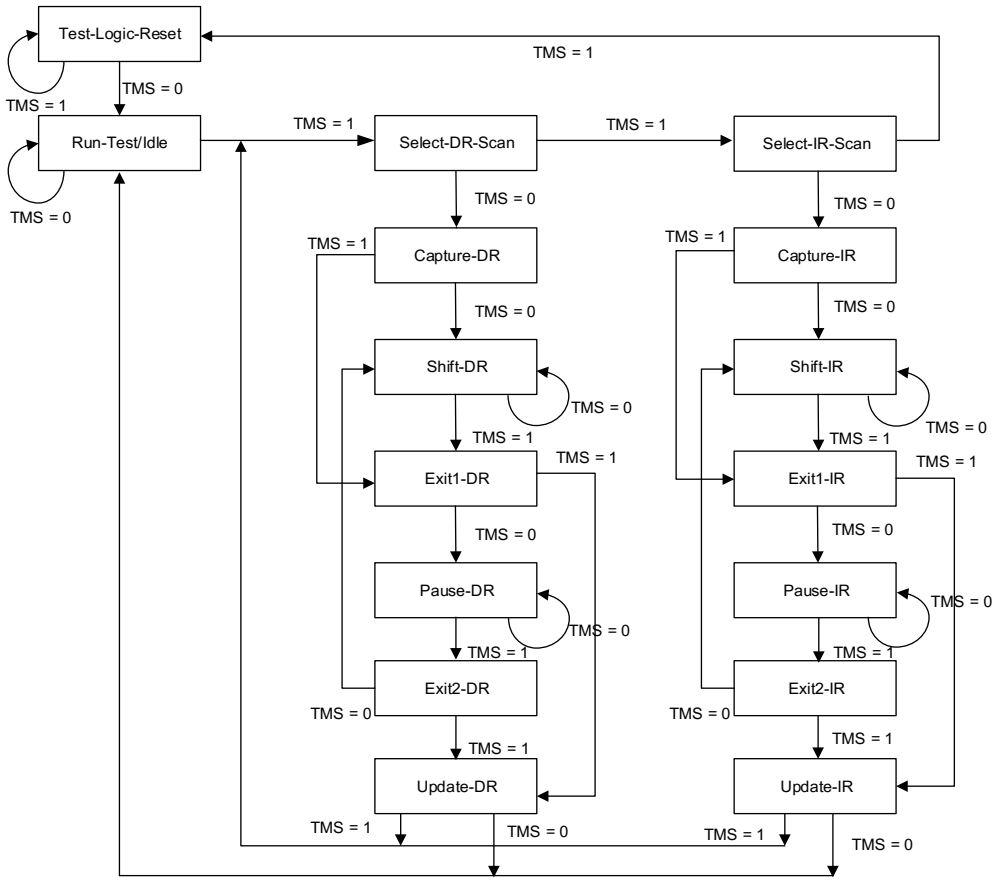
- The TAP controller is placed into the Test-Logic-Reset state, which deactivates the test logic and permits uninterrupted operation of the on-chip system logic.
- The instruction register is loaded with the IDCODE instruction.

When there is no need for a test operation, please load the BYPASS instruction to place the JTAGC block into bypass mode. In bypass mode, the single-bit bypass shift register is utilized to establish a minimal-length serial pathway for shifting data between TDI and TDO.

30.5.5. TAP controller state machine

The TAP controller is a synchronizer that controls the test logic based on the voltage changes on the TMS pin. It has 16 states, the states of this machine are illustrated in the *Figure 30-6. TAP controller state machine*. The numbers next to each state indicate the voltage level of the TMS pin when the TCK signal rises. If the TMS pin is kept high and the TCK signal rises multiple times, the TAP controller will return to the test logic reset state.

Figure 30-6. TAP controller state machine



30.5.6. JTAGC signal description

The JTAGC is comprised of a signal set that interfaces with external development tools, enabling access to test support functions. Details of the JTAGC signals can be found in the *Table 30-8. JTAG signal description*.

Table 30-8. JTAG signal description

Signal	I/O State	Reset state	Function description
TCK	Input	Weak pulldown	Test clock
TMS	Input	Weak pullup	Test mode select.
TDI	Input	Weak pullup	Test data in
TDO	Output	High impedance	Test data out

30.6. JTAG Data Communication(JDC)

30.6.1. Overview

The JTAG Data Communication (JDC) module enables the transfer of register data between the APB and JTAG domains. This functionality allows for communication between internal

resources that utilize memory-mapped register space and an external tool that interfaces with the JTAG port.

30.6.2. JDC architecture

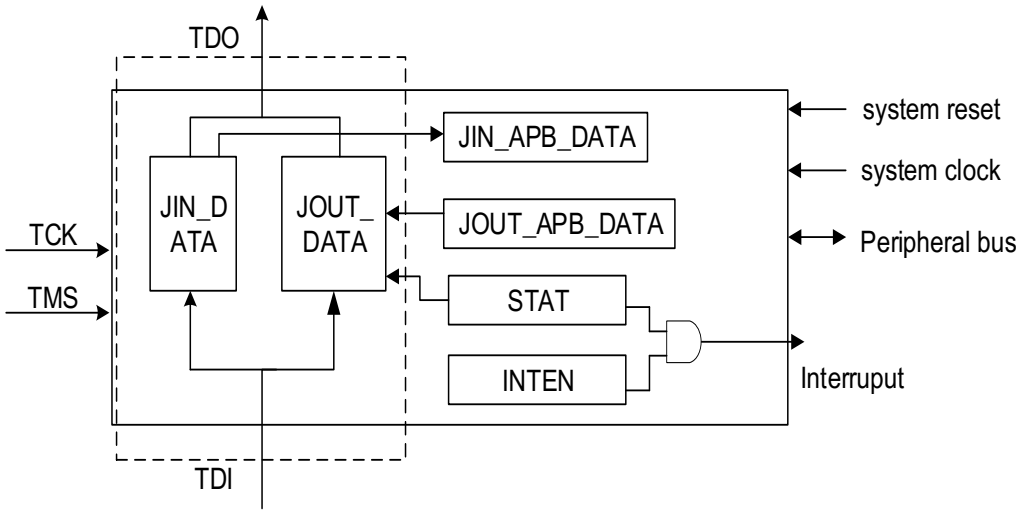
The JDC module consists of APB accessible registers, JTAG accessible registers, and associated logic in order to coordinate movement of data from one register domain to another

The JDC has the following APB data registers, which have separate memory space:

1. JTAG output data register (JDC_JOUT_APB_DATA)
   - This is a 32-bit memory-mapped register accessible for both read and write operations via the Advanced Peripheral Bus (APB).
   - The data within this register is routed out for capture into a designated JTAG register (JDC_JOUT_DATA), which can then be accessed through the JTAG port.
2. JTAG input data register (JDC_JIN_APB_DATA)
   - This is a 32-bit memory-mapped register that can solely be read through the Advanced Peripheral Bus (APB).
   - Its content is loaded from a specific JTAG register (JDC_JIN_DATA).
   - When new data has been shifted in through the JTAG port and is available for reading from the JTAG Input Data Register (JDC_JIN_APB_DATA) register.

A Status Register (JDC_STAT) captures the state of these flags and also feeds the flags into a JTAG register (JDC_JOUT_DATA) to make them visible to tools. There is a single bus interface for sending register data to the JTAGC and another single bus interface for receiving data from the JTAGC. The architecture is depicted in the *Figure 30-7. Block diagram of JDC*.

Figure 30-7. Block diagram of JDC



30.6.3. JDC functional overview

The JDC (JTAG Data Communication) module offers the capability to perform data transfers between the JTAG port and memory-mapped registers accessible via APB. It supports two

main operations: shifting in data from the JTAG port into a register and capturing data from a memory-mapped register into a JTAG shift register for output through the JTAG port. Here is an overview of its functionality:

1. Shifting out data (register to JTAG):

   - When software writes to the JDC_JOUT_APB_DATA register, it sets the JOUTSET flag bit, indicating that new data is available for reading via the JTAG port.
   - The JOUTSET status is reflected in the Status Register (JDC_STAT) and is also transmitted to the JDC_JOUT_DATA register for visibility.
   - A JTAG read operation of the JDC_JOUT_DATA register using the JOUT_READ instruction with a JOUTSET bit value of logic 1 indicates that the register contains fresh data.
   - The JOUTSET flag bit is cleared when exiting the Capture-DR JTAG state during the execution of the JOUT_READ instruction. Its clearance signifies to software that a new data value can be written to the JDC_JOUT_APB_DATA register.

2. Shifting in data (JTAG to register):

   - When a JTAG write operation is performed on the JDC_JIN_DATA register using the JIN_WRITE JTAG instruction, it updates the contents of the JDC_IN_APB_DATA register upon exiting the Update-DR state.
   - An update of the JDC_JIN_APB_DATA register sets the JINSET flag bit, signifying the availability of new data for reading via the APB interface.
   - The JINSET status is mirrored in the status register and is also transmitted to the JDC_JOUT_DATA register.
   - The JINSET flag bit is cleared when software reads the JDC_JIN_APB_DATA register via the APB interface.
   - A JTAG read operation on the JDC_JOUT_DATA register with a JINSET value of logic 0 indicates that new data can be written to the JDC_JIN_DATA register.

In summary, the JDC module facilitates bidirectional data communication between the JTAG port and memory-mapped registers via APB. It uses flag bits (JOUTSET and JINSET) and specific JTAG instructions to manage the transfer of data and ensure synchronization between the JTAG interface and the memory-mapped registers.

30.6.4. JDC Signal description

The JDC contains TCK、TMS、TDI、TDO signals, which are described in detail in the ***Table 30-9. JDC signal description***.

Table 30-9. JDC signal description

Signal	I/O State	Reset state	Function description
TCK	Input	-	Test clock input
TMS	Input	0	Test mode select
TDI	Input	0	Test data in
TDO	Output	High impedance	Test data out

30.6.5. JDC instruction

The JDC instructions are as follows:

■ Reserved (Code:0b00001)

This instruction is reserved for factory debug and not specified for regular use.

■ JOUT_READ (Code: 0b00010)

This instruction selects the JDC_JOUT_DATA data register. The JDC captures data from JDC_JOUT_APB_DATA into the JDC_JOUT_DATA register when JOUT_READ is active and the system enters the Capture-DR state.

■ JIN_WRITE (Code: 0b01110)

This instruction selects the JIN data register. The JDC captures data from JDC_JIN_DATA into JDC_JIN_APB_DATA when JIN_WRITE is active and the system exits the Update-DR state.

■ BYPASS (Code: 0b11111)

This instruction selects the bypass register for data operations.

■ Reserved (All other opcodes)

All other opcodes that are not specified are decoded to select the bypass register.

30.6.6. JDC secure challenge/response connection procedure

A summary of how the JDC can be used to interact with a security module to perform challenge/response password authorization is as follows:

1. Initialization:
   - All registers value are reset value when exit of reset.
   - JINSET = 0 allows tool to write to JDC_JIN_DATA.
   - JOUTSET = 0 allows software to write to JDC_JOUT_APB_DATA.
2. Interrupt configuration:
   - Set JDC_INTEN to enable JOUT and JIN interrupts, if needed.
3. Authorization initiation:
   - Tool writes to JDC_JIN_DATA via JTAGC JIN_WRITE instruction when ready.
4. Authorization start trigger:
   - Software monitors JINSET bit or enables JIN interrupt.
   - Interrupt triggers the authorization process.
5. Initial authorization data:
   - Software writes the first JOUT value to JDC_JOUT_APB_DATA, setting JOUTSET to 1.

6. Challenge handling:
   - Tool polls JOUT to detect software's challenge word.
   - Upon reading JOUT with JOUTSET bit set, the tool records the data for response generation.
   - Reading JOUT clears JOUTSET, possibly triggering a JOUT interrupt if JDC_INTEN[JOUTIE] is set.

7. Next challenge:
   - After JOUTSET is cleared, software provides the next challenge word.

8. Response generation:
   - When the tool generates a response, it checks JINSET in the JOUT register to know when to write the response to JDC_JIN_DATA.

9. Iterative process:
   - Security module may provide additional challenges and read corresponding responses as required, repeating the process.

10. Password input (Non-Challenge/Response):
    - For straightforward password input
    - Tool provides 32-bit password segments via JDC_JIN_DATA.
    - Monitor JINSET in the JOUT register to know when the next JDC_JIN_DATA value can be written.
    - No need to write JDC_JOUT_APB_DATA in this scenario.

30.7. Trace

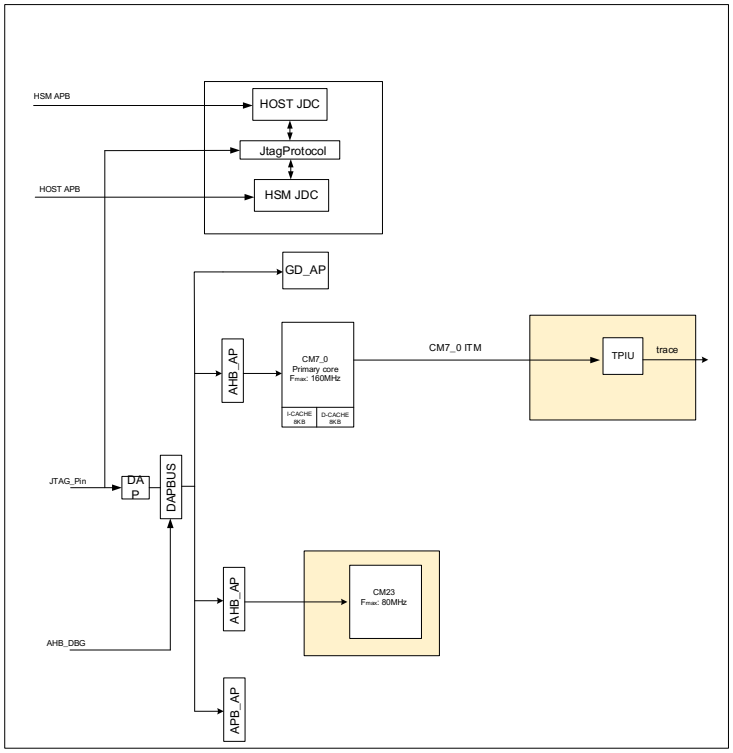
30.7.1. Overview

The trace subsystem combines trace data from all internal sources that generate trace information. The system consists of ATB, Debug APB and TPIU components. The Trace subsystem supports parallel tracing with multiple options for trace output. Trace information can be accessed from the trace interface or the DAP interface.

30.7.2. Trace architecture

The trace architecture is shown in *Figure 30-8. Block diagram of trace for GD32A711 / A712 devices*.

Figure 30-8. Block diagram of trace for GD32A711 / A712 devices



Note：For GD32A711 / A712 devices, the TPIU address is 0xE0040000.

30.7.3. TPIU

The debug subsystem has a standard 16-bit parallel TPIU integrated into it. This can be achieved with the following pin and frequency combinations:

- Four high-speed data + one clock pads @80 MHz (160 Mbit/s per pad) totaling 80 MB/s
- 16 low-speed data + one clock pads @20 MHz (40 Mbit/s per pad) totaling 80 MB/s

To transition the chip into Standby mode, software initiates a WFI (Wait for Interrupt) instruction to signal Standby entry. When the chip enters Standby mode, the PMU (Power Management Unit) sends a trace flush request to the TPIU and awaits a trace flush completion signal from the TPIU before proceeding. Subsequently, both the TPIU and the debug infrastructure clocks can be gated. It's essential to note that because the TPIU derives its clock from the system clock, clock gating should only occur after this stage.

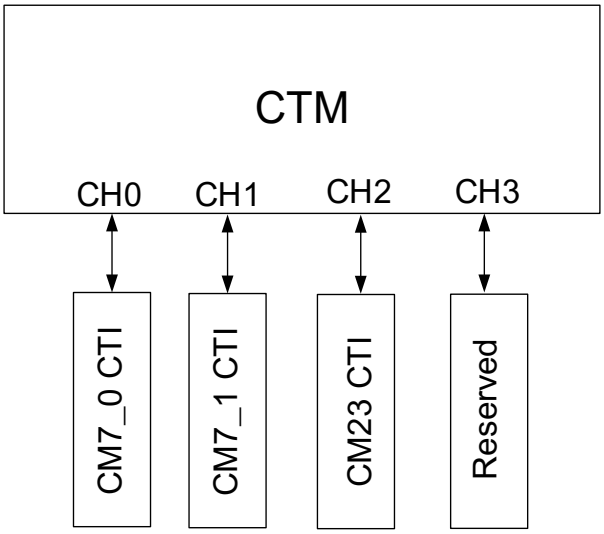
30.8. Embedded cross trigger (ECT)

The ECT enables multi-core run control and trace cross-triggering, such as synchronized start-stop operations for all cores or triggering trace events from one core or module based on triggers from another core. For comprehensive details about ECT, please refer to the CoreSight Components Technical Reference Manual.

The ECT architecture encompasses CTM (Cross Trigger Matrix) and CTIs (Cross Trigger

Interfaces). These CTIs offer a cross-triggering interface connecting the cores with other debug and trace modules. The interconnection of these CTI channels is facilitated through CTMs, as depicted in the *Figure 30-9. CTM and CTI interconnection diagram of* .

Figure 30-9. CTM and CTI interconnection diagram of GD32A711 / A712



The details of CTI assignments is shown in *Table 30-10. CTI assignments of GD32A711 / A712*.

Table 30-10. CTI assignments of GD32A711 / A712

CTI instance	Trigger number	Trigger in	Trigger out
M7_0 CTI	7	Reserved	Processor restart
	3	DWT comparator output 2	Reserved
	2	DWT comparator output 1	interrupt request1
	1	DWT comparator output 0	interrupt request0
	0	Processor halted	Processor debug request
CM23 CTI	7	Reserved	Reserved
	6	Reserved	Reserved
	5	Reserved	Reserved
	4	DWT comparator output 3	Reserved
	3	DWT comparator output 2	interrupt request1
	2	DWT comparator output 1	interrupt request0
	1	DWT comparator output 0	Processor restart
	0	Processor halted	Processor debug request

30.9. GD_AP register definition

GD_AP base address: 0x4025 0400

Table 30-11. GD_AP register overview

Register Name	Symbol	Address Offset	Reset Value
Authentication status register	GD_AP_AUTH_STAT	0x00	0x0000 0000
Authentication control register	GD_AP_AUTH_CTL	0x04	0x0000 0000
Status register	GD_AP_STAT	0x08	0x3800 0006
Control register	GD_AP_CTL	0x0C	0x0040 0000
Key challenge register x	GD_AP_KEYCHLG0 ~ GD_AP_KEYCHLG3	0x10 + 0x04 * x	0x0000 0000
Low power control register 0	GD_AP_LPCTL0	0x30	0x0000 0000
Low power control register 1	GD_AP_LPCTL1	0x38	0x0000 0000
Key response register x	GD_AP_KEYRESPx ~ GD_AP_KEYRESP3	0x40 + 0x04 * x	0x0000 0000
Debug enable register	GD_AP_DBGEN	0x80	0x0000 0000
Reset register	GD_AP_RST	0x90	0x0600 0000
ID code register	GD_AP_ID	0xFC	0xXXXX XXXX

30.9.1. Authentication status register (GD_AP_AUTH_STAT)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HSM_DAUTH	HOST_DAUTH	HOST_SAUTH	Reserved												
r	r	r													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												SWACC	Reserved	CHLGMOD	CHLGRDY
												r		r	r

Bits	Fields	Descriptions
31	HSM_DAUTH	HSM dynamic authentication status 0: HSM dynamic authentication failed 1: HSM dynamic authentication success
30	HOST_DAUTH	Host dynamic authentication success 0: HOST dynamic authentication failed 1: HOST dynamic authentication success
29	HOST_SAUTH	Host static authentication success 0: HOST static authentication failed 1: HOST static authentication success

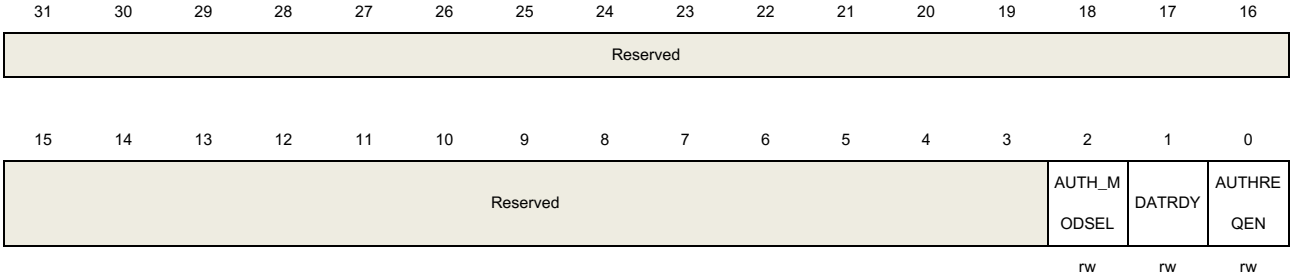
28:4	Reserved	Must be kept at reset value.
3	SWACC	Debug software access enable 0: Debug software access disable 1: Debug software access enable
2	Reserved	Must be kept at reset value.
1	CHLGMOD	Challenge mode status 0: Dynamic authentication 1: Static authentication
0	CHLGRDY	Challenge ready 0: Challenge is not ready 1: Challenge is ready

30.9.2. Authentication control register (GD_AP_AUTH_CTL)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



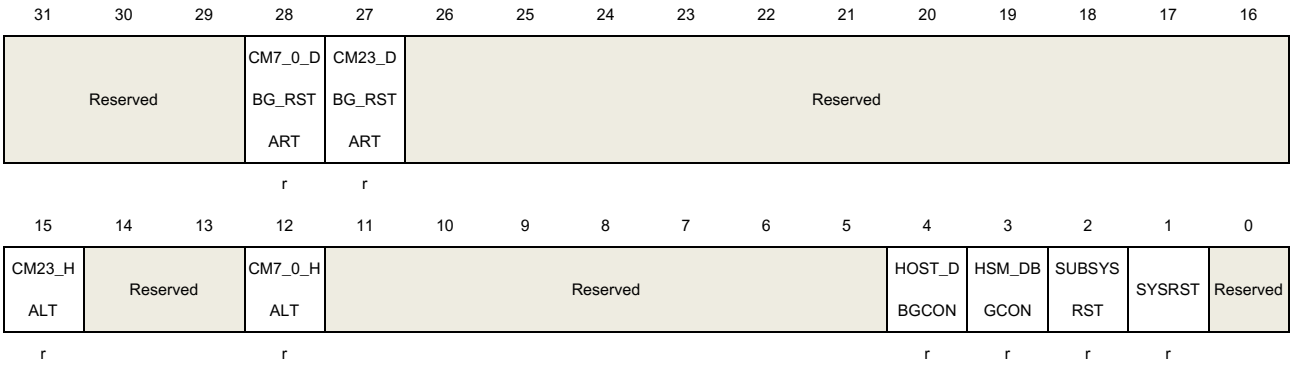
Bits	Fields	Descriptions
31:3	Reserved	Must be kept at reset value.
2	AUTH_MODSEL	Authentication mode selection 0: Host authentication 1: HSM authentication
1	DATRDY	Indicates the data is ready for static authentication After key response write this bit, the software can only write 1, and reset by power reset. 0: The data is not ready 1: The data is ready
0	AUTHREQEN	Debug authentication request enable 0: Debug authentication request disable 1: Debug authentication request enable

30.9.3. Status register (GD_AP_STAT)

Address offset: 0x08

Reset value: 0x1800 0006

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:29	Reserved	Must be kept at reset value.
28	CM7_0_DBG_RSTART	Core CM7_0 debug restarted 0: Core CM7_0 is debug state 1: Core CM7_0 is normal state(return to normal state from debug state)
27	CM23_DBG_RSTART	Core CM23 debug restarted 0: Core CM23 is debug state 1: Core CM23 is normal state(return to normal state from debug state)
26:16	Reserved	Must be kept at reset value.
15	CM23_HALT	MCU entry debug mode, core CM23 is halted 0: Core CM23 is not halted 1: Core CM23 is halted
14:13	Reserved	Must be kept at reset value.
12	CM7_0_HALT	MCU entry debug mode, core CM7_0 is halted 0: Core CM7_0 is not halted 1: Core CM7_0 is halted
11:5	Reserved	Must be kept at reset value.
4	HOST_DBGCON	HOST debug connect status Indicates the system can debug host cpu. 0: Not connected 1: Connected
3	HSM_DBGCON	HSM debug connect status Indicates the system can debug HSM cpu. 0: Not connected

		1: Connected
2	SUBSYSRST	Subsystem reset state 0: MCU is not in subsystem reset state 1: MCU is in subsystem reset state
1	SYSRST	System reset state 0: MCU is not in system reset state 1: MCU is in system reset state
0	Reserved	Must be kept at reset value.

30.9.4. Control register (GD_AP_CTL)

Address offset: 0x0C

Reset value: 0x0040 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRACE_CLKEN	Reserved		CM7_0_DBG_RSTARTEN	CM23_DBG_RSTARTEN	Reserved				SWOOVRDEN	Reserved	TPIUOVRDEN	Reserved			
rw			rw	rw					rw		rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PORWDGTEN	Reserved						CM7_0_DBGREQEN	CM23_DBGREQEN	Reserved	SUBSYSRSTEN	SYSRSTEN	Reserved			
rw							rw	rw		rw	rw				

Bits	Fields	Descriptions
31	TRACE_CLKEN	Trace clock enable 0: Trace clock disable 1: Trace clock enable
30:29	Reserved	Must be kept at reset value.
28	CM7_0_DBG_RSTARTEN	Core CM7_0 debug restarted enable 0: Disable 1: Enable
27	CM23_DBG_RSTARTEN	Core CM23 debug restarted enable 0: Disable 1: Enable
26:23	Reserved	Must be kept at reset value.
22	SWOOVRDEN	SWO override enable 0: Disable

		1: Enable
21	Reserved	Must be kept at reset value.
20	TPIUOVRDEN	TPIU override enable 0: Disable 1: Enable
19:16	Reserved	Must be kept at reset value.
15	PORWDGTEN	Power watchdog 0: Power watchdog disable 1: Power watchdog enable **Note**: When PORWDGTEN is disabled, the power watchdog reset generated by the subsystem reset request from SUBSYSRSTEN is prohibited.
14:9	Reserved	Must be kept at reset value.
8	CM7_0_DBGREQEN	Core CM7_0 debug request enable 0: Core CM7_0 debug request disable 1: Core CM7_0 debug request enable
7	CM23_DBGREQEN	Core CM23 debug request enable 0: Core CM23 debug request disable 1: Core CM23 debug request enable
6	Reserved	Must be kept at reset value.
5	SUBSYSRSTEN	Subsystem reset enable 0: Disable 1: Enable **Note**: When using SUBSYSRSTEN to implement the subsystem reset function, it is necessary to first configure enable and then configure disable; otherwise, the subsystem reset will be continuously triggered.
4	SYSRSTEN	System reset enable 0: Disable 1: Enable
3:0	Reserved	Must be kept at reset value.

30.9.5. Key challenge register x (GD_AP_KEYCHLGx) (x = 0...3)

Address offset: 0x10 + 0x4 * x
Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEYCHLG[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEYCHLG[15:0]															
r															

Bits	Fields	Descriptions
31:0	KEYCHLG[31:0]	Debug enablement key challenge

30.9.6. Low power control register 0(GD_AP_LPCTL0)

Address offset: 0x30

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															LPDBGEN
															rw

Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	LPDBGEN	Low power debug mode enable 0: Disable 1: Enable

30.9.7. Low power control register 1(GD_AP_LPCTL1)

Address offset: 0x38

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WAITRST
															rw

Bits	Fields	Descriptions
31:1	Reserved	Must be kept at reset value.
0	WAITRST	Support for wait NRST pin

On exiting Standby mode, RCTL waits until the debugger writes to another field in the GD_AP register to allow it to exit reset.

0:Not support

1:Support

30.9.8. Key response register x(GD_AP_KEYRESPx) (x = 0...3)

Address offset: 0x40 + 0x4 * x

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEYRESP[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEYRESP[15:0]															
rw															

Bits	Fields	Descriptions
31:0	KEYRESP[31:0]	Debug enablement key response

30.9.9. Debug enable register (GD_AP_DBGEN)

Address offset: 0x80

Reset value: 0x0000 0000 (In OEM_PROD and IN_FIELD phase)

0x3000 003C (In GD_PROD, CUST_DEL and IN_FIELD phase)

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		CM7_CORE_NIDEN	CM7_CORE_DBGEN	Reserved											
		rw	rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										CM7_COMP_NIDEN	CM7_COMP_DBGEN	CM23_NIDEN	CM23_DBGEN	Reserved	
										rw	rw	rw	rw		

Bits	Fields	Descriptions
31:30	Reserved	Must be kept at reset value.
29	CM7_CORE_NIDEN	Core CM7 non-invasive debug enable 0: Disable 1: Enable

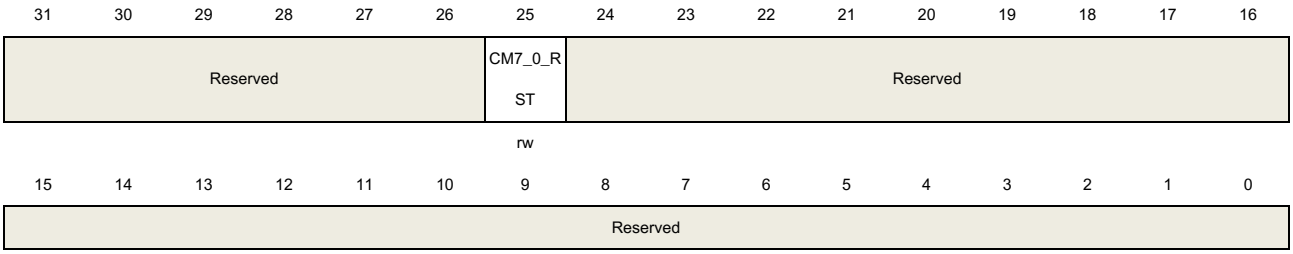
28	CM7_CORE_DBGEN	Core CM7 debug enable 0: Disable 1: Enable
27:6	Reserved	Must be kept at reset value.
5	CM7_COMP_NIDEN	CM7 subsystem, ITM, ETM and CTI component non-invasive debug enable 0: Disable 1: Enable
4	CM7_COMP_DBGEN	CM7 subsystem, ITM, ETM and CTI component debug enable 0: Disable 1: Enable
3	CM23_NIDEN	Core CM23 non-invasive debug enable 0: Disable 1: Enable
2	CM23_DBGEN	Core CM23 debug enable 0: Disable 1: Enable
1:0	Reserved	Must be kept at reset value.

30.9.10. Reset register (GD_AP_RST)

Address offset: 0x90

Reset value: 0x0600 0000

This register has to be accessed by word (32-bit).



Bits	Fields	Descriptions
31:26	Reserved	Must be kept at reset value.
25	CM7_0_RST	Core CM7_0 reset release 0: Core CM7_0 reset enable 1: Core CM7_0 reset release
24:0	Reserved	Must be kept at reset value.

30.9.11. ID code register (GD_AP_ID)

Address offset: 0xFC

Reset value: 0xXXXX XXXX

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ID_CODE[31:16]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID_CODE[15:0]															
r															

Bits	Fields	Descriptions
31:0	ID_CODE[31:0]	JTAG ID code These bits are read by software.

30.10. JDC register definition

HSM JDC base address: 0x5000 FC00

HOST JDC base address: 0x4000 EC00

Table 30-12. GD_AP register overview

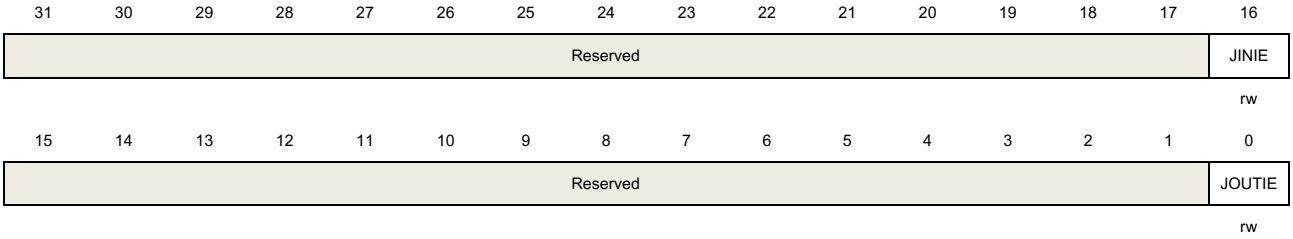
Register Name	Symbol	Address Offset	Reset Value
Interrupt enable register	JDC_INTEN	0x00	0x0000 0000
Status register	JDC_STAT	0x04	0x0000 0000
JOUT APB data register	JDC_JOUT_APB_DATA	0x08	0x0000 0000
JIN APB data register	JDC_JIN_APB_DATA	0x0C	0x0040 0000

30.10.1. Interrupt enable register (JDC_INTEN)

Address offset: 0x00

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



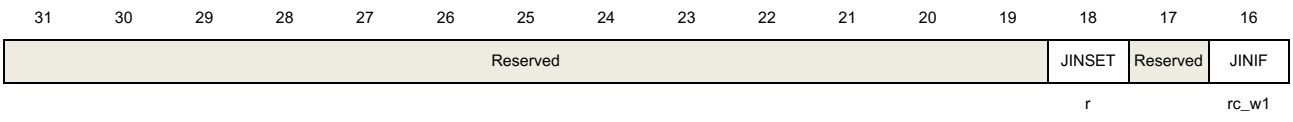
Bits	Fields	Descriptions
31:17	Reserved	Must be kept at reset value.
16	JINIE	JIN interrupt enable 0: Disable JIN interrupt 1: Enable JIN interrupt
15:1	Reserved	Must be kept at reset value.
0	JOUTIE	JOUT interrupt enable 0: Disable JOUT interrupt 1: Enable JOUT interrupt

30.10.2. Status register (JDC_STAT)

Address offset: 0x04

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													JOUTSET	Reserved	JOUTIF
													r		rc_w1

Bits	Fields	Descriptions
31:19	Reserved	Must be kept at reset value.
18	JINSET	JIN set 0: Software read JDC_JIN_APB_DATA via APB 1: The new data is written to the JDC_JIN_APB_DATA register
17	Reserved	Must be kept at reset value.
16	JINIF	The new data is written to the JDC_JIN_APB_DATA register interrupt flag 0: No write operation 1: The new data is written to the JDC_JIN_APB_DATA
15:3	Reserved	Must be kept at reset value.
2	JOUTSET	JOUT set 0: The tool through JTAG interface read of JDC_JOUT_DATA register 1: The new data is written to the JDC_JOUT_APB_DATA register
1	Reserved	Must be kept at reset value.
0	JOUTIF	The JOUTSET bit is cleared by tool reading JDC_JOUT_DATA register interrupt flag 0: No read operation 1: The JOUTSET bit is cleared by tool reading JDC_JOUT_DATA register

30.10.3. JOUT APB data register (JDC_JOUT_APB_DATA)

Address offset: 0x08

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
JOUT_APB_DATA[31:16]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JOUT_APB_DATA[15:0]															
rw															

Bits	Fields	Descriptions
31:0	JOUT_APB_DATA	The JDC_JOUT_APB_DATA register stores data that has been written via APB. The JDC module captures the contents of the JDC_JOUT_APB_DATA register and transfers it to the JDC_JOUT_DATA register, which can be accessed through the

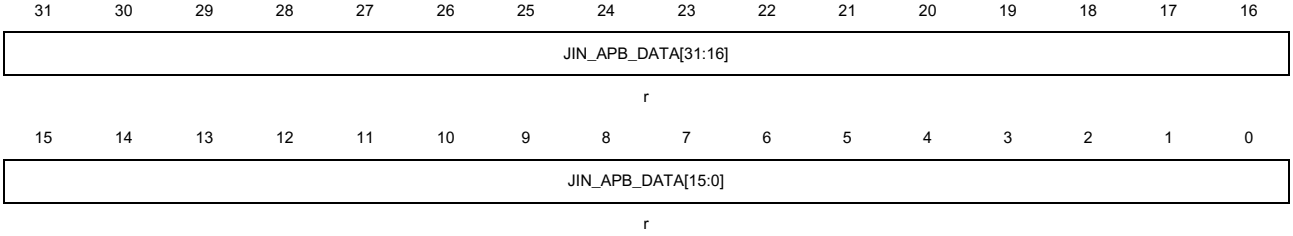
JTAG port. It's important to note that this register is reset during a system reset.

30.10.4. JIN APB data register (JDC_JIN_APB_DATA)

Address offset: 0x0C

Reset value: 0x0000 0000

This register has to be accessed by word (32-bit).

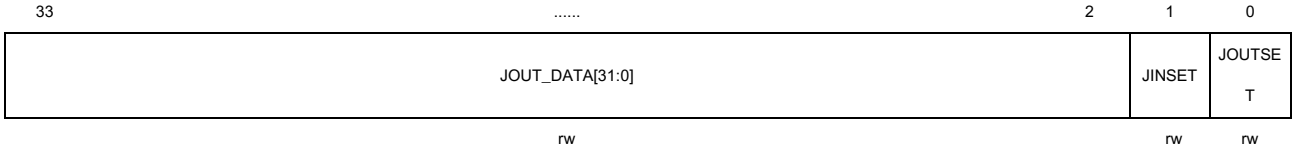


Bits	Fields	Descriptions
31:0	JIN_APB_DATA	The JDC_JIN_APB_DATA register contains data written to the JTAG input data register (JDC_JIN_DATA) through the JTAG port. This data can then be read via APB. Any attempt to write to the JDC_JIN_APB_DATA register via APB will result in a transfer error. It's important to note that this register is reset during a system reset.

30.10.5. JOUT data register (JDC_JOUT_DATA)

This register is JTAG accessible register that are not memory-mapped.

The JDC_JOUT_DATA register collects data from the JDC_JOUT_APB_DATA register when the JOUT_READ JTAG instruction is executed. Additionally, it maintains the status bits JINSET and JOUTSET. Please note that both system reset and JTAG reset will reset this register, and its initial value after reset is 0.

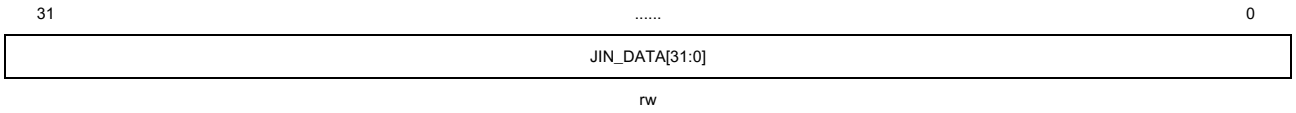


Bits	Fields	Descriptions
33:2	JOUT_DATA[31:0]	The value retrieve from JDC_JOUT_APB_DATA register
1	JINSET	JINSET state from JDC_STAT register
0	JOUTSET	JOUTSET state from JDC_STAT register

30.10.6. JIN data register (JDC_JIN_DATA)

JTAG accessible registers that are not memory-mapped.

The external tool communicates data to the JDC_JIN_DATA register through JTAG when the JIN_WRITE JTAG instruction is executed. Subsequently, the JDC module captures the data from JDC_JIN_DATA and stores it in the JDC_JIN_APB_DATA register, allowing it to be accessed via APB. It's important to note that this register is reset by both system reset and JTAG reset. After reset, the JDC_JIN_DATA register initializes to a value of 0.



Bits	Fields	Descriptions
31:0	JIN_DATA[31:0]	In JIN_WRITE Jtag instruction, the contains data can be captured in JDC_JIN_APB_DATA register when exiting the Update-DR state.

31. Appendix

31.1. List of abbreviations used in register

Table 31-1. List of abbreviations used in register

abbreviations for registers	Descriptions
read/write (rw)	Software can read and write to this bit.
read-only (r)	Software can only read this bit.
write-only (w)	Software can only write to this bit. Reading this bit returns the reset value.
read/clear write 1 (rc_w1)	Software can read as well as clear this bit by writing 1. Writing 0 has no effect on the bit value.
read/clear write 0 (rc_w0)	Software can read as well as clear this bit by writing 0. Writing 1 has no effect on the bit value.
toggle (t)	Software can toggle this bit by writing 1. Writing 0 has no effect.
read-only/set by write 1 (rt_w1)	Software can only read to this bit. Writing 1 triggers the event but has no effect on the bit value.
read/set (rs)	Software can read as well as set this bit to 1. Writing 0 has no effect on the bit value.
read/clear by read (rc_r)	Software can read this bit. Reading this bit automatically clears it to '0'. Writing '0' has no effect on the bit value.
read/set by read (rs_r)	Software can read this bit, and set this bit by reading. Writing has no effect on the bit value.
read/write once (rwo)	Software can only write once to this bit and can also read it at any time. Only a reset can return the bit to its reset value.
read/clear write (rc_w)	Software can read as well as clear this bit by writing. Writing 0 or 1 has the same effect to this bit.
read-only/write trigger (rt_w)	Software can read this bit. Writing 0 or 1 triggers an event but has no effect on the bit value.

31.2. List of terms

Table 31-2. List of terms

Glossary	Descriptions
Word	Data of 32-bit length.
Half-word	Data of 16-bit length.
Byte	Data of 8-bit length.
IAP (in-application programming)	Writing 0 has no effect IAP is the ability to re-program the Flash memory of a microcontroller while the user program is running.
ICP (in-circuit	ICP is the ability to program the Flash memory of a microcontroller using the

Glossary	Descriptions
programming)	JTAG protocol, the SWD protocol or the boot loader while the device is mounted on the user application board.
Option bytes	Product configuration bits stored in the Flash memory.
AHB	Advanced high-performance bus.
APB	Advanced peripheral bus.
RAZ	Read-as-zero.
WI	Writes ignored.
RAZ/WI	Read-as-zero, writes ignored.

31.3. Available peripherals

For availability of peripherals and their number across all MCU series types, refer to the corresponding device data datasheet.

32. Revision history

Table 32-1. Revision history

Revision No.	Description	Date
0.0	Initial Release	June 19, 2025
0.1	1. Update the description of the Gain and Offset section in the ***Analog-to-digital converter (ADC)*** chapter. 2. Update the reset value of the BKP_TPCTL register in the ***Backup registers (BKP)*** chapter. 3. Update Figure 23-13 and Table 23-1 in the ***Modular controller area network (M_CAN)***, and update the description of the DBTP and NBTP registers. 4. Add Figure 36-11 , update Table 36-13 and update the description of the GD_AP_CTL register in the ***Debug system (DBGSYS)*** chapter. 5. Update the description of the Mass erase section in the ***Flash memory controller (FMC)*** chapter. 6. Update the description of the Programming model in master transmitting mode section, update Table 20-6, and update the description of the I2C_STAT in the ***Inter-integrated circuit interface (I2C)*** chapter. 7. Update the description of the Header Trigger section, update the description of the UARTCR and LIN_STCTL registers in the ***Local Interconnect Network Flexible Driver (LINFlexD)*** chapter. 8. Update the description of the Overview section, update RCU_CTL register reset value, update RCU_SYSRSTUCNT, RCU_SUBSRSTCNT, RCU_SYSRSTCNT, RCU_APB2RST_ADD, RCU_APB2EN_ADD registers description in the ***Reset and clock unit (RCU)*** chapter. 9. Update the description of the I2S clock section and CRC function section, update the description of SPI_MREQCFG register in the ***Serial peripheral interface/Inter-IC sound (SPI/I2S)*** chapter. 10. Update Table 1-2, Table 1-3, Table 1-13, update the description in Host recovery mode section, update EXTMPU_CS0, TCM_BUFVLD and TCM_OVF registers description in the ***System and memory architecture*** chapter. 11. In ***TIMER*** chapter, add an abort mode chapter to the General level0 timer, add the BRKIE bit to the TIMERx_DMAINTEN0 register, the BRKIF bit to the	August 29, 2025

Revision No.	Description	Date
	TIMERx_INTF register, and the OAEN, BRKEN, and PROT[1:0] bits to the TIMERx_CCHP register. 12. Update Internal connect section in the ***Trigger selection controller (TRIGSEL)*** chapter.	
1.0	1. Added detailed description of analog multiplexer channels and updated Self test section in the ***Analog-to-digital converter (ADC)*** chapter. 2. Updated CLTCFG_SR_STAT0 register bit0 attribute to read-only in the ***Client configuration (CLTCFG)*** chapter. 3. Removed low power mode debug support functionality in the ***Debug (DBG)*** chapter. 4. Updated Table 35-1 FMU fault group definition in the ***Fault management unit (FMU)*** chapter. 5. Removed GPIOK from GPIOx_TMRASTAT register in the ***General-purpose and alternate-function I/Os (GPIO and AFIO)*** chapter. 6. Updated Memory test flow section related descriptions in the ***Memory test controller (MTC)***. 7. Updated IRC48M and IRC40K oscillator descriptions and multiple register bit field descriptions in the ***Reset and clock unit (RCU)*** chapter. 8. Updated SPI master receive delay compensation, separate FIFO, and CRC function descriptions in the ***Serial peripheral interface/Inter-IC sound (SPI/I2S)*** chapter. 9. Added FAST_MOD bit field to MBISTx phase configuration register in the ***Self-test control module (STCM) chapter***. 10. Updated GD32A711/A712 device memory mapping table and boot configuration description in the ***System and memory architecture*** chapter. 11. Updated Break mode and separate dead time insertion function descriptions, as well as complementary channel protection register bit field descriptions in the ***TIMER*** chapter.	October 21, 2025
1.1	1. In the ***Reset and clock unit(RCU),*** (1) Remove GD32A714/A72x/A74x-related information. (2) Remove Section 5.4 entirely. (3) Modify Sections 5.5.17 and 5.5.18 to add the description of ΔCKFM = 3. Correct errors in Table 5-3 regarding CKFM1 and CKFM2 descriptions. (4) Add the supported crystal oscillator types for different packages in section 5.2.3 LXTAL.	January 21, 2026

Revision No.	Description	Date
	(5) Add frequency differentiation for GD32A711_712AxxxxB and GD32A711_712BxxxxB chips in ClockTree, and include descriptions of LXTAL support for crystal oscillators in different packages. 2. Modify the reset value of the FWDGT_RLD register to 1FFF, and add the status bit RLD_BSY to the FWDGT_STAT register in the Section 15.1.4 of ***Watchdog timer(WDGT)*** chapter. 3. Add PA2 as a trigger source in the Table 6-1 of ***Interrupt / event controller(EXTI)*** chapter . 4. Update the interrupt vector table for CAN interrupts in the Table26-2 ***Interrupt requests management(IRM)*** chapter. ***5.*** In ***Local Interconnect Network Flexible Driver (LINFlexD)*** chapter. (1) Add deviation range description in the Section 18.3.7. (2) Modify the interrupt table, add the LIN_STCTL register, add 3 FIFO interrupts; modify the interrupt diagram, add interrupt sources in the Section 18.3.11. (3) Add base address definitions for LINFlexD12 and LINFlexD13 in the Section 18.4. (4) For the DMATXE register and DMARXE register, modify the bit allocation of DRENn and DTENn to 0 bits in the Section 18.4.24. (5) The DMATXE register and DMARXE register have been modified to set the bit allocation of DRENn and DTENn to 0 bits in the Section 18.4.25. (6) The LIN_STCTL register has added bit descriptions for FIFO interrupt sources and FIFO status flags in the Section 18.4.26. (7) Remove the redundant controller channel register addresses for A72x_A74x.in the Section 18.4. (8) Change LINTCSR.CNT to LINTCSR.CNT in the Section 18.3.3. (9) Add rw attribute to bits 0:5 of the register. in the Section 18.4.20. ***6.*** In ***Inter-integrated circuit interface (I2C)*** chapter. (1) Remove the register address information of GD32A714 /A72x /A74x I2C. in the Section 19.4. (2) Modify the description of OUERR. The OUERR error occurs in both master and slave modes when overflow or underflow happens in the Section 19.4.7 (3) Update the description of the conditions under which	

Revision No.	Description	Date
	OUERR is generated. in the Section 19.3.7. ***7.*** Remove the functional descriptions and register information related to GD32A714/ A72x/ A74x chips in the ***Trigger selection controller (TRIGSEL)*** chapter . 8. Remove the functional descriptions and register information related to GD32A714/ A72x/ A74x chips in the ***Direct memory access controller (DMA)*** chapter . 9. Remove the functional descriptions and register information related to GD32A714/ A72x/ A74x chips in the ***DMA request multiplexer (DMAMUX)*** chapter . 10. Add precautions for using the data register in the Section 20.5.4 of ***Serial peripheral circuit interface (SPI)*** chapter. 11. In ***General-purpose and alternate-function I/Os(GPIO)*** chapter. (1) Remove GD32A714/A72x/A74x-related information. (2) Add usage description for the backup function in the Section 8.3.3. (3) Modify the number of IOs based on GD32A711/A712 in the Section 8. (4) Remove Ethernet-related content in the Section 8.4.2. (5) Add description and notes on the usage of pad retention in the Section 8.3.13. 12. In ***System and memory architecture*** chapter. (1) Remove descriptions and registers related to GD32A714/A72x/A74x about SYSCFG、SRAM、TCM、CLTCFG. (2) Update the description of main features in the Section 1.7.2. (3) Remove Table 1-24: Unlock keys and corresponding peripheral registers memory mapping for GD32A714/A72x/A74x series in the 1.8.3. 13. In ***Fault management unit(FMU)*** chapter. (1) Remove faults supported only by the GD32A714/A72x/A74x series in Figure29-1. (2) More information about FFIO signals: Remove the description of GD32A714/A72x/A74x FFIO being controlled by timers. in the Section 29.3.11 (3) Remove the description of newly added registers for GD32A714/A72x/A74x in the Section 29.4.33 and 29.4.34. 14. In ***Debug (DBG)*** chapter. (1) Add GPIO idle state description. Add boundary scan description and entry reference in the Section 12.2.	

Revision No.	Description	Date
	(2) Remove GD32A714/A72x/A74x related registers in the Section 12.4. 15. In ***Debug system(DBGSYS)*** chapter (1) Remove GD32A714/A72x/A74x-related information. (2) Remove the descriptions related to Funnel in section 30.7.3. (3) Remove the content related to GD32A714/A72x/A74x ECT in section 30.8. (4) Remove the description of GD AP-related registers for GD32A714/A72x/A74x in section 30.10.1. 16. In the ***Flash memory controller(FMC)*** chapter. (1) Remove GD32A714/A72x/A74x-related information. (2) Remove the description of FMC-related registers for GD32A714/A72x/A74x in section 2.4. 17. Modify the chip operations in standby mode and the content regarding clock enablement in standby mode in the ***Power management unit(PMU)*** chapter. 18. Remove the content description regarding ADC2 in the ***Analog-to-digital converter(ADC)*** chapter. 19. In the ***Memory test controller(MTC)*** chapter. Remove GD32A714/A72x/A74x-related information. 20. In the ***Self-test control module(STCM)*** chapter. Remove GD32A714/A72x/A74x-related information. 21. Delete the original Chapter 9: MFCOM (only available for GD32A714/A72x/A74x). 22. Delete the original Chapter 24: SENT (only available for GD32A714/A72x/A74x). 23. Delete the original Chapter 25: ENET (only available for GD32A714/A72x/A74x). 24. Delete the original Chapter 27: HWSEM (only available for GD32A714/A72x/A74x). 25. Delete the original Chapter 28: MCMU (only available for GD32A714/A72x/A74x). 26. Delete the original Chapter 32: IOC (only available for GD32A714/A72x/A74x).	
1.2	1. Update the register overview tables for all modules. 2. ***Analog-to-Digital Converter (ADC)*** chapter: Correct the errors in the description of the ADC_CTL register, the self-test configuration register, the self-test error data register name, and the ADC_STAT register values. 3. ***Comparator (CMP)*** chapter: Add supplementary descriptions for CMP function usage. ;Add note for RRTINIT fields and revise the description of config cycle	Sep 10, 2026

Revision No.	Description	Date
	time;Add note for RRTVAL fields. 4. ***Power Management Unit (PMU)*** chapter: Remove LDO_RUN from the PMU section, and change "Low-Voltage Operation" to "Low-Power Operation." Add LVD precautions to the Voltage Monitor subsection. Remove the VDDB description from the hardware standby-mode entry sequence. Remove the BJT description from SW2. Refine the main core configuration description in SW1. 5. ***Self-Test Control Module (STCM)*** chapter: Revise the STCM register descriptions. 6. ***DMA Request Multiplexer (DMAMUX)*** chapter: Modify the number of peripheral trigger sources for DMAMUX. Add a note on the precautions when EXTI is used as a trigger input source and as a synchronization input source: the EXTI pending flag must be cleared before the next trigger can occur; 7. ***Fault Management Unit (FMU)*** chapter: Correct the fault numbering errors. 8. ***Serial Peripheral Interface (SPI)*** chapter: updated the software configuration requirements for SAMPDLY, revised the restrictions on configuring hardware single-frame/continuous-frame modes in slave mode, and updated the standard MSB/LSB timing diagrams for I2S. 9. ***Reset and Clock Unit (RCU)*** chapter: Update the description of the RCUSUBSERSTAT register. Added a note clarifying that SRAM, TCM, and CPU cache in the GD32A711/A712 require software initialization after reset, as needed by the application, to prevent spurious ECC error reports; Add notes for modify PLL configuration. 10. ***System and Memory Architecture*** chapter: Modify the description of HSM_SPLT_WGT_CHKEN. Revise the description regarding the processor core, changing it from dual-core to single-core. Change the minimum granularity of the MPU memory region size to 128 bytes in chapter1.7.2. 11. ***TIMER*** chapter: modify the BTIMER bit width.Remove the input of fmu in the Figure Advanced timer block diagram. 12. ***DBG*** chapter: Remove the low-power debugging feature description 13. ***Client configuration (CLTCFG)*** chapter: Delete CLTCFG_SRSTAT0; Update the TMR description of CLTCFG_TMR_CTL; Add note for CLTCFG_SUBSR_CTL2;	

Revision No.	Description	Date
	14. ***Memory Test Controller (MTC)*** chapter: Update the description of "Functional Description"; Update the error injection channel table: ECC Data Bit Injection, ECC Address Injection, EDC Data Bit Injection, ECC Injection after EDC; 15. ***Direct memory access controller (DMA)*** chapter: Fix a typo in Section 10.3.3: change "peripheral" to "memory". 16. ***Flash memory controller (FMC)*** chapter: 1. Standardized the terminology for Flash read buffers in the Function overview and Buffer and prefetch function sections, and updated the Master Buffer / Prefetch table; 2. Clarified that the ECC error enable bits control error reporting to the FMU, rather than directly generating CPU interrupts; updated the related ECC descriptions and register bit definitions. 17. ***System and Memory Architecture*** chapter: Standardize the descriptions of boot banks. Align the AN name with the revisions made in the system documentation. Correct the error regarding the core controlled in recovery mode.	

Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the "Product"), is owned by the Company according to the laws of the People's Republic of China and other applicable laws. The Company reserves all rights under such laws and no Intellectual Property Rights are transferred (either wholly or partially) or licensed by the Company (either expressly or impliedly) herein. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

To the maximum extent permitted by applicable law, the Company makes no representations or warranties of any kind, express or implied, with regard to the merchantability and the fitness for a particular purpose of the Product, nor does the Company assume any liability arising out of the application or use of any Product. Any information provided in this document is provided only for reference purposes. It is the sole responsibility of the user of this document to determine whether the Product is suitable and fit for its applications and products planned, and properly design, program, and test the functionality and safety of its applications and products planned using the Product. The Product is designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only, and the Product is not designed or intended for use in "Unintended Uses" as specified in the datasheet or user manual of the Product. Customers shall take any and all actions to ensure the Product meets the applicable laws and regulations. The Company is not liable for, in whole or in part, and customers shall hereby release the Company as well as its suppliers and/or distributors from, any claim, damage, or other liability arising from or related to all Unintended Uses of the Product. Customers shall indemnify and hold the Company, and its officers, employees, subsidiaries, affiliates as well as its suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Product. Information in this document is provided solely in connection with the Product. The Company reserves the right to make changes, corrections, modifications or improvements to this document and the Product described herein at any time without notice. The Company shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

This document is provided solely for the purpose of obtaining general information relating to the Product. Accordingly, you hereby agree to make use of this document solely for the purpose of obtaining general information relating to the Product. You further acknowledge and agree that this document may not be used in or in connection with any legal or administrative proceeding in any court, arbitration, agency, commission or other tribunal or in connection with any action, cause of action, litigation, claim, allegation, demand or dispute of any kind. You further acknowledge and agree that this document shall not be construed as an admission, acknowledgment or evidence of any kind, including, without limitation, as to the liability, fault or responsibility whatsoever of the Company, or as to the accuracy or validity of the information contained herein, or concerning any alleged product issue, failure, or defect. The Company does not promise that this document is accurate or error free and specifically disclaims all warranties, express or implied, as to the accuracy of the information contained herein. Accordingly, you agree that in no event will the Company be liable to you for any direct, indirect, consequential, exemplary, incidental, punitive, or other damages, including lost profits, arising from or relating to your reliance upon or use of this document.

THIS DOCUMENT IS CONFIDENTIAL INFORMATION WITH A CONFIDENTIAL AND PROPRIETARY NATURE PROVIDED BY THE COMPANY, IN TANGIBLE OR INTANGIBLE FORM ("CONFIDENTIAL INFORMATION").

BY DOWNLOADING, REVIEWING, ACCESSING OR OTHERWISE USING THIS DOCUMENT OR ANY PART THEREOF (AND THE RELATED INFORMATION), YOU AGREE TO BE BOUND BY THE CONFIDENTIALITY OBLIGATION PROVIDED HEREIN, AND YOU PROMISE AND AGREE TO MAINTAIN THE CONFIDENTIAL INFORMATION IN CONFIDENCE, AND TO PROTECT AND SAFEGUARD CONFIDENTIAL INFORMATION AGAINST UNAUTHORIZED USE OR DISCLOSURE USING AT LEAST THE SAME DEGREE OF CARE AS YOU

ACCORDS TO YOUR OWN CONFIDENTIAL INFORMATION OF LIKE IMPORTANCE, BUT IN NO CASE LESS THAN REASONABLE CARE. IF YOU DO NOT AGREE TO COMPLY WITH THE CONFIDENTIALITY OBLIGATION PROVIED HEREIN, DO NOT DOWNLOAD, REVIEW, ACCESS OR OTHERWISE USE THIS DOCUMENT.

You further agree that at all times you will comply with the following securities rules: (1) Do not copy or reproduce all or part of this document. (2) Keep this document locked away. (3) Further copies can be provided on a "need to know basis". Please contact the local Sales Office or document writer of the Company. (4) Disclosure of this document to any non-authorized party must be previously authorized by the Company only under the provision of a proper confidentiality contractual arrangement executed between the Company and you and must be treat as strictly confidential.

Information furnished is believed to be accurate and reliable. However, the Company assumes no responsibility for the consequences of use of such information nor for any infringement of patents or other rights of third parties which may result from its use.

Resale of the Product with provisions different from the information set forth herein shall void any warranty granted by the Company for the Product.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2026 GigaDevice Semiconductor Inc. - All rights reserved