ARM®-based 32-bit Cortex®-M4F MCU+FPU with 256 to 4032 KB Flash, sLib, dual QSPI, SDRAM, dual OTGFS, Ethernet, camera, 18 timers, 3 ADCs, 23 communication interfaces

Feature

- **Core: ARM®32-bit Cortex®-M4F CPU with FPU**
  - 288 MHz maximum frequency, with a Memory Protection Unit (MPU), single-cycle multiplication and hardware division
  - Floating Point Unit (FPU)
  - DSP instructions
- **Memories**
  - 256 to 4032 KBytes of Flash memory
  - sLib: configurable part of main Flash set as a library area with code executable but secured, non-readable
  - 384 to 512 KBytes of SRAM
  - External memory controller (XMC) with 16-bit data bus: supports CF, SRAM, PSRAM, NOR, NAND, SDRAM memories
  - Up to 2 x QSPI interfaces for external SPI Flash or SPI RAM extension, and memory mapping mode
- **LCD parallel interface, 8080/6800 modes**
- **Power control (PWC)**
  - 2.6 V ~ 3.6 V application supply
  - Power-on reset (POR)/ low-voltage reset (LVR), and power voltage monitor (PVM)
  - Low power: Sleep, Deepsleep, and Standby modes
  - VBAT supply for LEXT, ERTC and 20 x 32-bit battery power register (ERTC_BPR)
- **Clock and reset management (CRM)**
  - 4 to 25 MHz crystal oscillator (HEXT)
  - Internal 48 MHz factory-trimmed clock (HICK), accuracy 1% at T_A=25 °C, 2.5 % at T_A=-40 to +105 °C, with automatic clock calibration (ACC)
  - PLL with configurable frequency multiplication and division factor
  - 32.768 kHz crystal oscillator (LEXT)
  - Internal 40 kHz RC oscillator (LICK)
- **Analog**
  - 3 x 12-bit 5.33 MSPS A/D converters, up to 24 input channels, 12-bit/10-bit/8-bit/6-bit configurable resolution
  - Temperature sensor (V_{TS}), internal reference voltage (V_{INTR}), V_{BAT} battery voltage monitor ($V_{BAT}/4$)
  - 2 x 12-bit D/A converters
- **DMA:**
  - 2 x general-purpose DMAs and 1 x EDMA
  - 22 channels in all
- **Up to 116 Fast I/O Interfaces**
  - All mappable to 16 external interrupt vectors
  - Almost 5 V-tolerant
- **Up to 18 Timers (TMR)**
  - Up to 13 x 16-bit timers + 2 x 32-bit timers, each with 4 IC/OC/PWM or pulse counter channels
  - 2 x Watchdog timers (WDT and WWDT)
  - SysTick timer: 24-bit downcounter
- **ERTC: enhanced RTC with auto wakeup, alarm, subsecond precision, hardware calendar and calibration feature**
- **Up to 23 communication interfaces**
  - Up to 3 x I^2C interfaces (SMBus/PMBus)
  - Up to 4 x USARTs/4 x UARTs (ISO7816 interface, LIN, IrDA capability, modem control and RS485 drive enable, with interchangeable TX/RX)
  - Up to 4 x SPIs (36 Mbit/s), all with I^2S interface multiplexed, I^2S2/ I^2S3 support full-duplex
  - Up to 2 x CAN interfaces (2.0B Active)
  - Up to 2 x OTG FS controllers supporting crystal-less
  - Up to 2 x SDIO interfaces
  - Infrared transmitter (IRTMR)
  - 10/100M Ethernet MAC with dedicated DMA and SRAM(4 Kbytes): IEEE1588 hardware support, MII/RMII available (For AT32F437 only)
- **8~14 bit parallel digital camera interface (DVP)**
- **CRC Calculation Unit**
- **96-bit ID (UID)**
- **Debug mode**
  - SWD and JTAG interfaces
- **Temperature range: -40 to 105℃**
- **Packaging**
  - LQFP144 20 x 20 mm LQFP100 14 x 14 mm
  - LQFP64 10 x 10 mm LQFP48 7 x 7 mm
  - QFN48 6 x 6 mm

List of Models

Internal Flash	Model
4032 KBytes	AT32F435ZMT7, AT32F435VMT7,AT32F435RMT7, AT32F435CMT7, AT32F435CMU7, AT32F437ZMT7, AT32F437VMT7, AT32F437RMT7
1024 KBytes	AT32F435ZGT7, AT32F435VGT7, AT32F435RGT7, AT32F435CGT7, AT32F435CGU7, AT32F437ZGT7 AT32F437VGT7, AT32F437RGT7
448 KBytes	AT32F435ZDT7, AT32F435VDT7,AT32F435RDT7, AT32F435CDT7,AT32F435CDU7, AT32F437ZDT7 AT32F437VDT7, AT32F437RDT7
256 KBytes	AT32F435ZCT7, AT32F435VCT7,AT32F435RCT7, AT32F435CCT7,AT32F435CCU7, AT32F437ZCT7 AT32F437VCT7, AT32F437RCT7

Contents

1 System architecture 44

1.1 System overview 46

1.1.1 ARM Cortex™-M4F processor 46

1.1.2 BusMatrix 46

1.1.3 Bit band 48

1.1.4 Interrupt and exception vectors 50

1.1.5 System Tick (SysTick) 53

1.1.6 Reset 54

1.2 List of abbreviations for registers 56

1.3 Device characteristics information 56

1.3.1 Flash memory size register 56

1.3.2 Device electronic signature 56

2 Memory resources 57

2.1 Internal memory address map 57

2.2 Flash memory 57

2.3 SRAM memory 60

2.4 Peripheral address map 60

3 Power control (PWC) 63

3.1 Introduction 63

3.2 Main Features 63

3.3 POR/LVR 63

3.4 Power voltage monitor (PVM) 64

3.5 Power domain 64

3.6 Power saving modes 65

3.7 PWC registers 67

3.7.1 Power control register (PWC_CTRL) 68

3.7.2 Power control/status register (PWC_CTRLSTS) 68

3.7.3 LDO output voltage select register (PWC_LDOOV) 69

4 Clock and reset manage (CRM) 70

4.1 Clock 70
4.1.1 Clock sources 70
4.1.2 System clock 71
4.1.3 Peripheral clock 71
4.1.4 Clock fail detector 72
4.1.5 Auto step-by-step system clock switch 72
4.1.6 Clock output 72
4.1.7 Interrupts 72
4.2 Reset 72
4.2.1 System reset 72
4.2.2 Battery powered domain reset 73
4.3 CRM registers 73
4.3.1 Clock control register (CRM_CTRL) 74
4.3.2 PLL clock configuration register (CRM_PLLCFG) 75
4.3.3 Clock configuration register (CRM_CFG) 76
4.3.4 Clock interrupt register (CRM_CLKINT) 78
4.3.5 APB peripheral reset register1 (CRM_APBRST1) 79
4.3.6 APB peripheral reset register2 (CRM_APBRST2) 80
4.3.7 APB peripheral reset register3 (CRM_APBRST3) 80
4.3.8 APB1 peripheral reset register (CRM_APB1RST) 80
4.3.9 APB2 peripheral reset register (CRM_APB2RST) 82
4.3.10 APB peripheral clock enable register1 (CRM_AHBEN1) 83
4.3.11 APB peripheral clock enable register2 (CRM_AHBEN2) 84
4.3.12 APB1 peripheral clock enable register3 (CRM_AHBEN3) 84
4.3.13 APB1 peripheral clock enable register (CRM_APB1EN) 84
4.3.14 APB2 peripheral clock enable register (CRM_AHB2EN) 86
4.3.15 APB peripheral clock enable in low power mode register1 (CRM_AHBLPEN1) 87
4.3.16 APB peripheral clock enable in low power mode register2 (CRM_AHBLPEN2) 88
4.3.17 APB peripheral clock enable in low power mode register3 (CRM_AHBLPEN3) 89
4.3.18 APB1 peripheral clock enable in low power mode register (CRM_AHB1LPEN) 89

4.3.19 APB2 peripheral clock enable in low power mode register (CRM_AHB2LPEN) 90
4.3.20 Battery powered domain control register (CRM_BPDC) 91
4.3.21 Control/status register (CRM_CTRLSTS) 92
4.3.22 Additional register1 (CRM_MISC1) 93
4.3.23 Additional register2 (CRM_MISC2) 94

5 Flash memory controller (FLASH) 95

5.1 FLASH introduction 95
5.2 Flash memory operation 101
5.2.1 Unlock/lock 101
5.2.2 Erase operation 101
5.2.3 Programming operation 104
5.2.4 Read operation 105
5.3 User system data area operation 105
5.3.1 Unlock/lock 105
5.3.2 Erase operation 106
5.3.3 Programming operation 107
5.3.4 Read operation 108
5.4 Flash memory protection 108
5.4.1 Access protection 108
5.4.2 Erase/program protection 108
5.5 Special functions 108
5.5.1 Security library settings 108
5.6 Flash memory registers 109
5.6.1 Flash performance select register (FLASH_PSR) 110
5.6.2 Flash unlock register (FLASH_UNLOCK) 111
5.6.3 Flash user system data unlock register (FLASH_USD_UNLOCK) 111
5.6.4 Flash status register (FLASH_STS) 111
5.6.5 Flash control register (FLASH_CTRL) 111
5.6.6 Flash address register (FLASH_ADDR) 112
5.6.7 User system data register (FLASH_USD) 112
5.6.8 Erase/program protection status register0 (FLASH_EPPS0) 113
5.6.9 Erase/program protection status register1 (FLASH_EPPS1) 113
5.6.10 Flash unlock register2 (FLASH_UNLOCK2) 113

5.6.11 Flash status register2 (FLASH_STS2) 113
5.6.12 Flash control register2 (FLASH_CTRL2) 113
5.6.13 Flash address register2 (FLASH_ADDR2) 114
5.6.14 Flash continue read register (FLASH_CONTR) 114
5.6.15 Flash divider register (FLASH_DIVR) 114
5.6.16 Flash security library status register2 (SLIB_STS2) 115
5.6.17 Flash security library status register0 (SLIB_STS0) 115
5.6.18 Flash security library status register1 (SLIB_STS1) 115
5.6.19 Flash security library password clear register (SLIB_PWD_CLR) 115
5.6.20 Security library additional status register (SLIB_MISC_STS) 115
5.6.21 Security library password setting register (SLIB_SET_PWD) 116
5.6.22 Security library address setting register0 (SLIB_SET_RANGE0) 116
5.6.23 Security library address setting register1 (SLIB_SET_RANGE1) 116
5.6.24 Security library unlock register (SLIB_UNLOCK) 117
5.6.25 Flash CRC calibration control register (FLASH_CRC_CTRL) 117
5.6.26 Flash CRC check result register (FLASH_CRC_CHKR) 117

6 GPIOs and IOMUX 118

6.1 Introduction 118
6.2 Function overview 118
6.2.1 GPIO structure 118
6.2.2 GPIO reset status 118
6.2.3 General-purpose input configuration 119
6.2.4 Analog input/output configuration 119
6.2.5 General-purpose output configuration 119
6.2.6 I/O port protection 120
6.2.7 IOMUX structure 120
6.2.8 Multiplexed function pull-up/down configuration 120
6.2.9 IOMUX input/output 121
6.2.10 Peripheral MUX function configuration 135
6.2.11 IOMUX map priority 135
6.2.12 External interrupt/wake-up lines 136
6.3 GPIO registers 136
6.3.1 GPIO configuration register (GPIOx_CFGR) (x=A..H) 136
6.3.2 GPIO output mode register (GPIOx_OMODE) (x=A..H) 136

6.3.3 GPIO drive capability register (GPIOx_ODRVR) (x=A..H) 137
6.3.4 GPIO pull-up/pull-down register (GPIOx_PULL) (x=A..H) 137
6.3.5 GPIO input register (GPIOx_IDH) (x=A..H) 137
6.3.6 GPIO output register (GPIOx_IDH) (x=A..H) 137
6.3.7 GPIO set/clear register (GPIOx_SCR) (x=A..H) 137
6.3.8 GPIO write protection register (GPIOx_WPR) (x=A..H) 138
6.3.9 GPIO multiplexed function low register (GPIOx_MUXL) (x=A..H) ... 138
6.3.10 GPIO multiplexed function high register (GPIOx_MUXH) (x=A..H) . 139
6.3.11 GPIO port bit clear register (GPIOx_CLR) (x=A..H) 139
6.3.12 GPIO huge current control register (GPIOx_HDRV) (x=A..E) 139

7 System configuration controller (SCFG) 140

7.1 Introduction 140
7.2 IOMUX registers 140
7.2.1 SCFG configuration register1 (SCFG_CFG1) 140
7.2.2 SCFG configuration register2 (SCFG_CFG2) 141
7.2.3 SCFG external interrupt configuration register1 (SCFG_ EXINTC1) 141
7.2.4 SCFG external interrupt configuration register2 (SCFG_ EXINTC2) 142
7.2.5 SCFG external interrupt configuration register3 (SCFG_ EXINTC3) 143
7.2.6 SCFG external interrupt configuration register4 (SCFG_ EXINTC4) 144
7.2.7 SCFG ultra high sourcing/sinking strength (SCFG_UHDRV) 146

8 External interrupt/Event controller (EXINT) 148

8.1 EXINT introduction 148
8.2 Function overview and configuration procedure 148
8.3 EXINT registers 149
8.3.1 Interrupt enable register (EXINT_INTEN) 149
8.3.2 Event enable register (EXINT_EVTEN) 149
8.3.3 Polarity configuration register1 (EXINT_ POLCFG1) 149
8.3.4 Polarity configuration register2 (EXINT_ POLCFG2) 150
8.3.5 Software trigger register (EXINT_ SWTRG) 150
8.3.6 Interrupt status register (EXINT_ INTSTS) 150

9 DMA controller (DMA) 151

9.1 Introduction 151

9.2 Main features 151
9.3 Function overview 152
9.3.1 DMA configuration 152
9.3.2 Handshake mechanism 152
9.3.3 Arbiter 153
9.3.4 Programmable data transfer width 153
9.3.5 Errors 154
9.3.6 Interrupts 154
9.4 DMA multiplexer (DMAMUX) 154
9.4.1 DMAMUX functional overview 155
9.4.2 DMAMUX overflow interrupts 157
9.5 DMA registers 158
9.5.1 DMA interrupt status register (DMA_STS) 159
9.5.2 DMA interrupt flag clear register (DMA_CLR) 161
9.5.3 DMA channel-x configuration register (DMA_CxCTRL) (x = 1...7) 163
9.5.4 DMA channel-x number of data register (DMA_CxDTCNT) (x = 1...7) 164
9.5.5 DMA channel-x peripheral address register (DMA_CxPADDR) (x = 1...7) 164
9.5.6 DMA channel-x memory address register (DMA_CxMADDR) (x = 1...7) 164
9.5.7 DMAMUX selection register (DMA_MUXSEL) 164
9.5.8 DMAMUX channel-x control register (DMA_MUXCxCTRL) (x = 1...7) 164
9.5.9 DMAMUX generator x control register (DMA_MUXGxCTRL) (x = 1...4) 165
9.5.10 DMAMUX channel synchronization status register (DMA_MUXSYNCSTS) 166
9.5.11 DMAMUX channel interrupt clear flag register (DMA_MUXSYNCCLR) 166
9.5.12 DMAMUX generator interrupt status register (DMA_MUXGSTS) 166
9.5.13 DMAMUX generator interrupt flag clear register (DMA_MUXGCLR) 167

10 CRC calculation unit (CRC) 168

10.1 CRC introduction 168
10.2 CRC function description 168
10.3 CRC registers 169
10.3.1 Data register (CRC_DT) 169
10.3.2 Common data register (CRC_CDT) 169
10.3.3 Control register (CRC_CTRL) 170

10.3.4 Initialization register (CRC_IDT) 170

11 I²C interface 171

11.1 I²C introduction 171
11.2 I²C main features 171
11.3 I²C function overview 171
11.4 I²C interface 172
11.4.1 I²C timing control 174
11.4.2 Data transfer management 175
11.4.3 I²C master communication flow 176
11.4.4 I²C slave communication flow 181
11.4.5 SMBus 185
11.4.6 SMBus master communication flow 187
11.4.7 SMBus slave communication flow 191
11.4.8 Data transfer using DMA 195
11.4.9 Error management 195
11.5 I²C interrupt requests 197
11.6 I²C debug mode 197
11.7 I²C registers 197
11.7.1 Control register1 (I2C_CTRL1) 198
11.7.2 Control register2 (I2C_CTRL2) 199
11.7.3 Address register1 (I2C_OADDR1) 200
11.7.4 Own address register2 (I2C_OADDR2) 200
11.7.5 Timing register (I2C_CLKCTRL) 200
11.7.6 Timeout register (I2C_TIMEOUT) 200
11.7.7 Status register (I2C_STS) 201
11.7.8 Status clear register (I2C_CLR) 202
11.7.9 PEC register (I2C_PEC) 203
11.7.10 Receive data register (I2C_RXDT) 203
11.7.11 Transmit data register (I2C_TXDT) 203

12 Universal synchronous/asynchronous receiver/transmitter (USART) 204

12.1 USART introduction 204
12.2 Full-duplex/half-duplex selector 206
12.3 Mode selector 206

12.3.1 Introduction 206
12.3.2 Configuration procedure 206
12.4 USART frame format and configuration 209
12.5 DMA transfer introduction 211
12.5.1 Transmission using DMA 211
12.5.2 Reception using DMA 211
12.6 Baud rate generation 212
12.6.1 Introduction 212
12.6.2 Configuration 212
12.7 Transmitter 213
12.7.1 Transmitter introduction 213
12.7.2 Transmitter configuration 213
12.8 Receiver 214
12.8.1 Receiver introduction 214
12.8.2 Receiver configuration 214
12.8.3 Start bit and noise detection 215
12.9 Tx/Rx swap 216
12.10 Interrupt requests 217
12.11 I/O pin control 218
12.12 USART registers 218
12.12.1 Status register (USART_STS) 218
12.12.2 Data register (USART_DT) 219
12.12.3 Baud rate register (USART_BAUDR) 220
12.12.4 Control register1 (USART_CTRL1) 220
12.12.5 Control register2 (USART_CTRL2) 221
12.12.6 Control register3 (USART_CTRL3) 222
12.12.7 Guard time and divider register (USART_GDIV) 223

13 Serial peripheral interface (SPI) 224

13.1 SPI introduction 224
13.2 Function overview 224
13.2.1 SPI description 224
13.2.2 Full-duplex/half-duplex selector 225
13.2.3 Chip select controller 227

13.2.4 SPI_SCK controller 228
13.2.5 CRC 228
13.2.6 DMA transfer 229
13.2.7 TI mode 229
13.2.8 Transmitter 230
13.2.9 Receiver 230
13.2.10 Motorola mode 231
13.2.11 TI mode 233
13.2.12 Interrupts 234
13.2.13 IO pin control 234
13.2.14 Precautions 235
13.3 I²S functional description 235
13.3.1 I²S introduction 235
13.3.2 I²S full-duplex 236
13.3.3 Operating mode selector 236
13.3.4 Audio protocol selector 238
13.3.5 I2S_CLK controller 239
13.3.6 DMA transfer 241
13.3.7 Transmitter/Receiver 241
13.3.8 I²S communication timings 242
13.3.9 Interrupts 243
13.3.10 IO pin control 243
13.4 SPI registers 244
13.4.1 SPI control register1 (SPI_CTRL1) (Not used in I²S mode) 244
13.4.2 SPI control register2 (SPI_CTRL2) 245
13.4.3 SPI status register (SPI_STS) 246
13.4.4 SPI data register (SPI_DT) 247
13.4.5 SPICRC register (SPI_CPOLY) (Not used in I²S mode) 247
13.4.6 SPIRxCRC register (SPI_RCRC) (Not used in I²S mode) 247
13.4.7 SPITxCRC register (SPI_TCRC) 247
13.4.8 SPI_I2S register (SPI_I2SCTRL) 247
13.4.9 SPI_I2S prescaler register (SPI_I2SCLKP) 248
14 Timer 249
14.1 Basic timer (TMR6 and TMR7) 250

14.1.1 TMR6 and TMR7 introduction 250
14.1.2 TMR6 and TMR7 main features 250
14.1.3 TMR6 and TMR7 function overview 250
14.1.3.1 Count clock 250
14.1.3.2 Counting mode 250
14.1.3.3 Debug mode 252
14.1.4 TMR6 and TMR7 registers 252
14.1.4.1 TMR6 and TMR7 control register1 (TMRx_CTRL1) 252
14.1.4.2 TMR6 and TMR7 control register2 (TMRx_CTRL2) 253
14.1.4.3 TMR6 and TMR7 DMA/interrupt enable register (TMRx_IDEN) .. 253
14.1.4.4 TMR6 and TMR7 interrupt status register (TMRx_ISTS) 253
14.1.4.5 TMR6 and TMR7 software event register (TMRx_SWEVT) 253
14.1.4.6 TMR6 and TMR7 counter value (TMRx_CVAL) 253
14.1.4.7 TMR6 and TMR7 division (TMRx_DIV) 254
14.1.4.8 TMR6 and TMR7 period register (TMRx_PR) 254
14.2 General-purpose timer (TMR2 to TMR5) 254
14.2.1 TMR2 to TMR5 introduction 254
14.2.2 TMR2 to TMR5 main features 254
14.2.3 TMR2 to TMR5 functional overview 255
14.2.3.1 Count clock 255
14.2.3.2 Counting mode 258
14.2.3.3 TMR input function 261
14.2.3.4 TMR output function 263
14.2.3.5 TMR synchronization 266
14.2.3.6 Debug mode 269
14.2.4 TMRx registers 269
14.2.4.1 TMR2 to TMR5 control register1 (TMRx_CTRL1) 270
14.2.4.2 TMR2 to TMR5 control register2 (TMRx_CTRL2) 271
14.2.4.3 TMR2 to TMR5 slave timer control register (TMRx_STCTRL) 271
14.2.4.4 TMR2 to TMR5 DMA/interrupt enable register (TMRx_IDEN) 272
14.2.4.5 TMR2 to TMR5 interrupt status register (TMRx_ISTS) 273
14.2.4.6 TMR2 to TMR5 software event register (TMRx_SWEVT) 275
14.2.4.7 TMR2 to TMR5 channel mode register1 (TMRx_CM1) 275
14.2.4.8 TMR2 to TMR5 channel mode register2 (TMRx_CM2) 277
14.2.4.9 TMR2 to TMR5 channel control register (TMRx_CCTRL) 278
14.2.4.10 TMR2 to TMR5 counter value (TMRx_CVAL) 278
14.2.4.11 TMR2 to TMR5 division value (TMRx_DIV) 279

14.2.4.12 TMR2 to TMR5 period register (TMRx_PR) 279
14.2.4.13 TMR2 to TMR5 channel 1 data register (TMRx_C1DT) 279
14.2.4.14 TMR2 to TMR5 channel 2 data register (TMRx_C2DT) 279
14.2.4.15 TMR2 to TMR5 channel 3 data register (TMRx_C3DT) 279
14.2.4.16 TMR2 to TMR5 channel 4 data register (TMRx_C4DT) 280
14.2.4.17 TMR2 to TMR5 DMA control register (TMRx_DMACTRL) 280
14.2.4.18 TMR2 to TMR5 DMA data register (TMRx_DMADT)..... 280
14.2.4.19 TMR5 channel input remapping register (TMR2_RMP) 280
14.2.4.20 TMR2 channel input remapping register (TMR5_RMP) 280
14.3 General-purpose timer (TMR9 to TMR14) 281
14.3.1 TMR9 to TMR14 introduction.....281
14.3.2 TMR9 to TMR14 main features.....281
14.3.2.1 TMR9 and TMR12 main features 281
14.3.2.2 TMR10, TMR11, TMR13 and TMR14 main features 281
14.3.3 TMR9 to TMR14 functional overview 282
14.3.3.1 Count clock 282
14.3.3.2 Counting mode 285
14.3.3.3 TMR input function..... 286
14.3.3.4 TMR output function..... 289
14.3.3.5 TMR synchronization..... 291
14.3.3.6 Debug mode 292
14.3.4 TMR9 and TMR12 registers 292
14.3.4.1 TMR9 and TMR12 control register1 (TMRx_CTRL1)..... 292
14.3.4.2 TMR9 and TMR12 Slave timer control register (TMRx_STCTRL) 293
14.3.4.3 TMR9 and TMR12 DMA/interrupt enable register (TMRx_IDEN) 293
14.3.4.4 TMR9 and TMR12 interrupt status register (TMRx_ISTS) 294
14.3.4.5 TMR9 and TMR12 software event register (TMRx_SWEVT) 294
14.3.4.6 TMR9 and TMR12 channel mode register1 (TMRx_CM1) 295
14.3.4.7 TMR9 and TMR12 channel control register (TMRx_CCTRL)..... 297
14.3.4.8 TMR9 and TMR12 counter value (TMRx_CVAL)..... 297
14.3.4.9 TMR9 and TMR12 division value (TMRx_DIV) 297
14.3.4.10 TMR9 and TMR12 period register (TMRx_PR) 297
14.3.4.11 TMR9 and TMR12 channel 1 data register (TMRx_C1DT) 297
14.3.4.12 TMR9 and TMR12 channel 2 data register (TMRx_C2DT)..... 298
14.3.5 TMR10,TMR11, TMR13 and TMR14 registers.....298
14.3.5.1 TMR10, TMR11, TMR13 and TMR14 control register1 (TMRx_CTRL1) 298
14.3.5.2 TMR10, TMR11, TMR13 and TMR14 DMA/interrupt enable register

(TMRx_IDEN) 299
14.3.5.3 TMR10, TMR11, TMR13 and TMR14 interrupt status register (TMRx_ISTS) 299
14.3.5.4 TMR10, TMR11, TMR13 and TMR14 software event register (TMRx_SWEVT)..... 299
14.3.5.5 TMR10, TMR11, TMR13 and TMR14 channel mode register1 (TMRx_CM1) 300
14.3.5.6 TMR10, TMR11, TMR13 and TMR14 channel control register (TMRx_CCTRL) 301
14.3.5.7 TMR10, TMR11, TMR13 and TMR14 counter value (TMRx_CVAL)302
14.3.5.8 TMR10, TMR11, TMR13 and TMR14 division value (TMRx_DIV) 302
14.3.5.9 TMR10, TMR11, TMR13 and TMR14 period register (TMRx_PR)302
14.3.5.10 TMR10, TMR11, TMR13 and TMR14 channel 1 data register (TMRx_C1DT) 302
14.4 Advanced-control timers (TMR1,TMR8 and TMR20)..... 303
14.4.1 TMR1,TMR8 and TMR20 introduction 303
14.4.2 TMR1,TMR8 and TMR20 main features 303
14.4.3 TMR1, TMR8 and TMR20 functional overview 303
14.4.3.1 Count clock 303
14.4.3.2 Counting mode 307
14.4.3.3 TMR input function..... 311
14.4.3.4 TMR output function..... 313
14.4.3.5 TMR break function..... 317
14.4.3.6 TMR synchronization..... 319
14.4.3.7 Debug mode..... 320
14.4.4 TMR1, TMR8 and TM20 registers 321
14.4.4.1 TMR1, TMR8 and TMR20 control register1 (TMRx_CTRL1) 321
14.4.4.2 TMR1, TMR8 and TMR20 control register2 (TMRx_CTRL2) 322
14.4.4.3 TMR1, TMR8 and TMR20 slave timer control register (TMRx_STCTRL) 323
14.4.4.4 TMR1, TMR8 and TMR20 DMA/interrupt enable register (TMRx_IDEN) 324
14.4.4.5 TMR1, TMR8 and TMR20 interrupt status register (TMRx_ISTS) 325
14.4.4.6 TMR1, TMR8 and TMR20 software event register (TMRx_SWEVT)326
14.4.4.7 TMR1, TMR8 and TMR20 channel mode register1 (TMRx_CM1) 326
14.4.4.8 TMR1, TMR8 and TMR20 channel mode register2 (TMRx_CM2) 328
14.4.4.9 TMR1, TMR8 and TMR20 Channel control register (TMRx_CCTRL)329
14.4.4.10 TMR1, TMR8 and TMR20 counter value (TMRx_CVAL) 331
14.4.4.11 TMR1, TMR8 and TMR20 division value (TMRx_DIV) 331

14.4.4.12 TMR1, TMR8 and TMR20 period register (TMRx_PR)..... 331
14.4.4.13 TMR1, TMR8 and TMR20 repetition period register (TMRx_RPR)331
14.4.4.14 TMR1, TMR8 and TMR20 channel 1 data register (TMRx_C1DT)331
14.4.4.15 TMR1, TMR8 and TMR20 channel 2 data register (TMRx_C2DT)332
14.4.4.16 TMR1, TMR8 and TMR20 channel 3 data register (TMRx_C3DT)332
14.4.4.17 TMR1, TMR8 and TMR20 channel 4 data register (TMRx_C4DT)332
14.4.4.18 TMR1, TMR8 and TMR20 break register (TMRx_BRK)..... 332
14.4.4.19 TMR1, TMR8 and TMR20 DMA control register (TMRx_DMACTRL) 333
14.4.4.20 TMR1, TMR8 and TMR20 DMA data register (TMRx_DMADT)334
14.4.4.21 TMR1, TMR8 and TMR20 channel mode register3 (TMRx_ CM3)334
14.4.4.22 TMR1, TMR8 and TMR20 channel 5 data register (TMRx_C5DT)334

15 Window watchdog timer (WWDT) 335

15.1 WWDT introduction 335
15.2 WWDT main features 335
15.3 WWDT functional overview 335
15.4 Debug mode 336
15.5 WWDT registers 336
15.5.1 Control register (WWDT_CTRL) 336
15.5.2 Configuration register (WWDT_CFG) 336
15.5.3 Status register (WWDT_STS) 337

16 Watchdog timer (WDT) 338

16.1 WDT introduction 338
16.2 WDT main features 338
16.3 WDT functional overview 338
16.4 Debug mode 339
16.5 WDT registers 339
16.5.1 Command register (WDT_CMD) 340
16.5.2 Divider register (WDT_DIV) 340
16.5.3 Reload register (WDT_RLD) 340
16.5.4 Status register (WDT_STS) 340
16.5.5 Window register (WDT_WIN) 340

17 Enhanced real-time clock (ERTC) 341

17.1 ERTC introduction 341
17.2 ERTC main features 341
17.3 ERTC function overview 342
17.3.1 ERTC clock 342
17.3.2 ERTC initialization 342
17.3.3 Periodic automatic wakeup 344
17.3.4 ERTC calibration 345
17.3.5 Reference clock detection 346
17.3.6 Time stamp function 346
17.3.7 Tamper detection 346
17.3.8 Multiplexed function output 347
17.3.9 ERTC wakeup 347
17.4 ERTC registers 348
17.4.1 ERTC time register (ERTC_TIME) 349
17.4.2 ERTC date register (ERTC_DATE) 349
17.4.3 ERTC control register (ERTC_CTRL) 349
17.4.4 ERTC initialization and status register (ERTC_STS) 351
17.4.5 ERTC divider register (ERTC_DIV) 352
17.4.6 ERTC wakeup timer register (ERTC_WAT) 352
17.4.7 ERTC coarse calibration register (ERTC_CCAL) 352
17.4.8 ERTC alarm clock A register (ERTC_ALA) 353
17.4.9 ERTC alarm clock B register (ERTC_ALB) 353
17.4.10 ERTC write protection register (ERTC_WP) 354
17.4.11 ERTC subsecond register (ERTC_SBS) 354
17.4.12 ERTC time adjustment register (ERTC_TADJ) 354
17.4.13 ERTC time stamp time register (ERTC_TSTM) 354
17.4.14 ERTC time stamp date register (ERTC_TSDT) 355
17.4.15 ERTC time stamp subsecond register (ERTC_TSSBS) 355
17.4.16 ERTC smooth calibration register (ERTC_SCAL) 355
17.4.17 ERTC tamper configuration register (ERTC_TAMP) 355
17.4.18 ERTC alarm clock A subsecond register (ERTC_ALASBS) 357
17.4.19 ERTC alarm clock B subsecond register (ERTC_ALBSBS) 357
17.4.20 ERTC battery powered domain data register (ERTC_BPRx) 357

18 Analog-to-digital converter (ADC) 358

18.1 ADC introduction 358
18.2 ADC main features 358
18.3 ADC structure 358
18.4 ADC functional overview 359
18.4.1 Channel management 359
18.4.1.1 Internal temperature sensor 360
18.4.1.2 Internal reference voltage 360
18.4.1.3 Battery voltage 360
18.4.2 ADC operation process 360
18.4.2.1 Power-on and calibration 361
18.4.2.2 Trigger 362
18.4.2.3 Sampling and conversion sequence 363
18.4.3 Conversion sequence management 363
18.4.3.1 Sequence mode 363
18.4.3.2 Automatic preempted group conversion mode 364
18.4.3.3 Repetition mode 364
18.4.3.4 Partition mode 365
18.4.4 End of conversion 365
18.4.5 Oversampling 366
18.4.5.1 Oversampling of ordinary group of channels 367
18.4.5.2 Oversampling of preempted group of channels 368
18.4.6 Data management 368
18.4.6.1 Data alignment 368
18.4.6.2 Data read 369
18.4.7 Voltage monitoring 369
18.4.7.1 Status flag and interrupts 370
18.5 Master/Slave mode 370
18.5.1 Data management 371
18.5.2 Simultaneous mode 371
18.5.3 Alternate preempted trigger mode 372
18.5.4 Regular shift mode 373
18.6 ADC registers 374
18.6.1 ADC status register (ADC_STS) 376
18.6.2 ADC control register1 (ADC_CTRL1) 377
18.6.3 ADC control register2 (ADC_CTRL2) 378
18.6.4 ADC sampling time register 1 (ADC_SPT1) 380

18.6.5 ADC sampling time register 2 (ADC_SPT2) 381
18.6.6 ADC preempted channel data offset register x (ADC_ PCDTOx) (x=1..4) 383
18.6.7 ADC voltage monitor high threshold register (ADC_VWHB) 383
18.6.8 ADC voltage monitor low threshold register (ADC_ VWLB) 384
18.6.9 ADC ordinary sequence register 1 (ADC_ OSQ1) 384
18.6.10 ADC ordinary sequence register 2 (ADC_ OSQ2) 384
18.6.11 ADC ordinary sequence register 3 (ADC_ OSQ3) 384
18.6.12 ADC preempted sequence register (ADC_ PSQ) 385
18.6.13 ADC preempted data register x (ADC_ PDTx) (x=1..4) 385
18.6.14 ADC ordinary data register (ADC_ ODT) 385
18.6.15 ADC oversampling register (ADC_ OVSP) 385
18.6.16 ADC calibration value register (ADC_CALVAL) 386
18.6.17 ADC common status register (ADC_CSTS) 386
18.6.18 ADC common control register (ADC_CSTS) 388
18.6.19 ADC common data register (ADC_CODT) 390

19 Digital-to-analog converter (DAC) 391

19.1 DAC introduction 391
19.2 DAC main features 391
19.3 Design tips 391
19.4 Functional overview 392
19.4.1 Trigger events 392
19.4.2 Noise/Triangular-wave generation 392
19.4.3 DAC data alignment 393
19.5 DAC registers 394
19.5.1 DAC control register (DAC_CTRL) 394
19.5.2 DAC software trigger register (DAC_SWTRG) 396
19.5.3 DAC1 12-bit right-aligned data holding register (DAC_ D1DTH12R) 397
19.5.4 DAC1 12-bit left-aligned data holding register (DAC_ D1DTH12L) 397
19.5.5 DAC1 8-bit right-aligned data holding register (DAC_ D1DTH8R) 397
19.5.6 DAC2 12-bit right-aligned data holding register (DAC_ D2DTH12R) 397
19.5.7 DAC2 12-bit left-aligned data holding register (DAC_ D2DTH12L) 397
19.5.8 DAC2 8-bit right-aligned data holding register (DAC_ D2DTH8R) 397
19.5.9 Dual DAC 12-bit right-aligned data holding register (DAC_ DDTH12R) 397

19.5.10 Dual DAC 12-bit left-aligned data holding register (DAC_ DDTH12L)398
19.5.11 Dual DAC 8-bit right-aligned data holding register (DAC_ DDTH8R)398
19.5.12 DAC1 data output register (DAC_ D1ODT) 398
19.5.13 DAC2 data output register (DAC_ D2ODT) 398
19.5.14 DAC status register (DAC_STS) 398

20 CAN 399

20.1 CAN introduction 399
20.2 CAN main features..... 399
20.3 Baud rate 399
20.4 Interrupt management 402
20.5 Design tips 403
20.6 Functional overview 403
20.6.1 General description 403
20.6.2 Operating modes 404
20.6.3 Test modes 404
20.6.4 Message filtering 405
20.6.5 Message transmission 406
20.6.6 Message reception 408
20.6.7 Error management.....408
20.7 CAN registers 409
20.7.1 CAN control and status registers 410
20.7.1.1 CAN master control register (CAN_MCTRL) 410
20.7.1.2 CAN master status register (CAN_MSTS)..... 411
20.7.1.3 CAN transmit status register (CAN_TSTS) 412
20.7.1.4 CAN receive FIFO 0 register (CAN_RF0) 415
20.7.1.5 CAN receive FIFO 1 register (CAN_RF1) 415
20.7.1.6 CAN interrupt enable register (CAN_INTEN) 416
20.7.1.7 CAN error status register (CAN_ESTS)..... 417
20.7.1.8 CAN bit timing register (CAN_BTMG) 418
20.7.2 CAN mailbox registers 418
20.7.2.1 Transmit mailbox identifier register (CAN_TMIx) (x=0..2) 419
20.7.2.2 Transmit mailbox data length and time stamp register (CAN_TMCx) (x=0..2)..... 419
20.7.2.3 Transmit mailbox data low register (CAN_TMDTLx) (x=0..2) 420
20.7.2.4 Transmit mailbox data high register (CAN_TMDTHx) (x=0..2) ... 420

20.7.2.5 Receive FIFO mailbox identifier register (CAN_RFIx) (x=0..1) .. 420
20.7.2.6 Receive FIFO mailbox data length and time stamp register (CAN_RFCx) (x=0..1) 420
20.7.2.7 Receive FIFO mailbox data low register (CAN_RFDTLx) (x=0..1)421
20.7.2.8 Receive FIFO mailbox data high register (CAN_RFDTHx) (x=0..1)421
20.7.3 CAN filter registers 421
20.7.3.1 CAN filter control register (CAN_FCTRL) 421
20.7.3.2 CAN filter mode configuration register (CAN_FMCFG) 421
20.7.3.3 CAN filter bit width configuration register (CAN_ FBWCFG)..... 421
20.7.3.4 CAN filter FIFO association register (CAN_ FRF) 422
20.7.3.5 CAN filter activation control register (CAN_ FACFG)..... 422
20.7.3.6 CAN filter bank i filter bit register (CAN_ FiFBx) (i=0..13; x=1..2)422

21 Universal serial bus full-seed device interface (OTGFS)..... 423

21.1 USBFS structure..... 423
21.2 OTGFS functional description 423
21.3 OTGFS clock and pin configuration 424
21.3.1 OTGFS clock configuration 424
21.3.2 OTGFS pin configuration 424
21.4 OTGFS interrupts 425
21.5 OTGFS functional description 425
21.5.1 OTGFS initialization 425
21.5.2 OTGFS FIFO configuration 426
21.5.2.1 Device mode 426
21.5.2.2 Host mode..... 427
21.5.2.3 Refresh controller transmit FIFO 428
21.5.3 OTGFS host mode.....428
21.5.3.1 Host initialization 428
21.5.3.2 OTGFS channel initialization..... 429
21.5.3.3 Halting a channel..... 429
21.5.3.4 Queue depth..... 430
21.5.3.5 Special cases 431
21.5.3.6 Host HFIR feature 431
21.5.3.7 Initialize bulk and control IN transfers..... 433
21.5.3.8 Initialize bulk and control OUT/SETUP transfers..... 435
21.5.3.9 Initialize interrupt IN transfers..... 437
21.5.3.10 Initialize interrupt OUT transfers 439

21.5.3.11 Initialize synchronous IN transfers 441
21.5.3.12 Initialize synchronous OUT transfers 442
21.5.4 OTGFS device mode 444
21.5.4.1 Device initialization 444
21.5.4.2 Endpoint initialization on USB reset 444
21.5.4.3 Endpoint initialization on enumeration completion 445
21.5.4.4 Endpoint initialization on SetAddress command 445
21.5.4.5 Endpoint initialization on SetConfiguration/SetInterface command 445
21.5.4.6 Endpoint activation 445
21.5.4.7 USB endpoint deactivation 446
21.5.4.8 Control write transfers (SETUP/Data OUT/Status IN) 446
21.5.4.9 Control read transfers (SETUP/Data IN/Status OUT) 446
21.5.4.10 Control transfers (SETUP/Status IN) 447
21.5.4.11 Read FIFO packets 447
21.5.4.12 OUT data transfers 448
21.5.4.13 IN data transfers 450
21.5.4.14 Non-periodic (bulk and control) IN data transfers 451
21.5.4.15 Non-synchronous OUT data transfers 452
21.5.4.16 Synchronous OUT data transfers 454
21.5.4.17 Enable synchronous endpoints 455
21.5.4.18 Incomplete synchronous OUT data transfers 457
21.5.4.19 Incomplete synchronous IN data transfers 458
21.5.4.20 Periodic IN (interrupt and synchronous) data transfers 458
21.6 OTGFS control and status registers 460
21.6.1 CSR register map 460
21.6.2 OTGFS register address map 461
21.6.3 OTGFS global registers 465
21.6.3.1 OTGFS status and control register (OTGFS_GOTGCTL) 465
21.6.3.2 OTGFS interrupt status control register (OTGFS_GOTGINT) 465
21.6.3.3 OTGFS AHB configuration register (OTGFS_GAHBCFG) 465
21.6.3.4 OTGFS USB configuration register (OTGFS_GUSBCFG) 466
21.6.3.5 OTGFS reset register (OTGFS_GRSTCTL) 467
21.6.3.6 OTGFS interrupt register (OTGFS_GINTSTS) 469
21.6.3.7 OTGFS interrupt mask register (OTGFS_GINTMSK) 472
21.6.3.8 OTGFS receive status debug read/OTG status read and POP registers (OTGFS_GRXSTSR / OTGFS_GRXSTSP) 473
21.6.3.9 OTGFS receive FIFO size register (OTGFS_GRXFSIZ) 474
21.6.3.10 OTGFS non-periodic Tx FIFO size (OTGFS_GNPTXFSIZ)/Endpoint 0

Tx FIFO size registers (OTGFS_DIEPTXF0) 474
21.6.3.11 OTGFS non-periodic Tx FIFO size/request queue status register (OTGFS_GNPTXSTS) 475
21.6.3.12 OTGFS general controller configuration register (OTGFS_GCCFG) 475
21.6.3.13 OTGFS controller ID register (OTGFS_GUID) 476
21.6.3.14 OTGFS host periodic Tx FIFO size register (OTGFS_HPTXFSIZ)476
21.6.3.15 OTGFS device IN endpoint Tx FIFO size register (OTGFS_DIEPTXFn) (x=1...7, where n is the FIFO number) 476
21.6.4 Host-mode registers 476
21.6.4.1 OTGFS host mode configuration register (OTGFS_HCFG) 476
21.6.4.2 OTGFS host frame interval register (OTGFS_HFIR) 477
21.6.4.3 OTGFS host frame number/frame time remaining register (OTGFS_HFNUM) 477
21.6.4.4 OTGFS host periodic Tx FIFO/request queue register (OTGFS_HPTXSTS) 478
21.6.4.5 OTGFS host all channels interrupt register (OTGFS_HAINT) 478
21.6.4.6 OTGFS host all channels interrupt mask register (OTGFS_HAINTMSK) 478
21.6.4.7 OTGFS host port control and status register (OTGFS_HPRT) ... 479
21.6.4.8 OTGFS host channelx characteristics register (OTGFS_HCCHARx) (x = 0...15, where x= channel number)..... 480
21.6.4.9 OTGFS host channelx interrupt register (OTGFS_HCINTx) (x = 0...15, where x= channel number)..... 481
21.6.4.10 OTGFS host channelx interrupt mask register (OTGFS_HCINTMSKx) (x = 0...15, where x= channel number)..... 482
21.6.4.11 OTGFS host channelx transfer size register (OTGFS_HCTSIZx) (x = 0...15, where x= channel number)..... 482
21.6.5 Device-mode registers 483
21.6.5.1 OTGFS device configure register (OTGFS_DCFG) 483
21.6.5.2 OTGFS device control register (OTGFS_DCTL) 483
21.6.5.3 OTGFS device status register (OTGFS_DSTS) 485
21.6.5.4 OTGFS device OTGFSIN endpoint common interrupt mask register (OTGFS_DIEPMSK) 485
21.6.5.5 OTGFS device OUT endpoint common interrupt mask register (OTGFS_DOEPMSK)..... 486
21.6.5.6 OTGFS device all endpoints interrupt mask register (OTGFS_DAINT)..... 487
21.6.5.7 OTGFS all endpoints interrupt mask register (OTGFS_DAINTMSK)487
21.6.5.8 OTGFS device IN endpoint FIFO empty interrupt mask register

(OTGFS_DIEPEMPMSK) 487
21.6.5.9 OTGFS device control IN endpoint 0 control register (OTGFS_DIEPCTL0) 487
21.6.5.10 OTGFS device IN endpoint-x control register (OTGFS_DIEPCTLx) (x=x=1...7, where x is endpoint number) 489
21.6.5.11 OTGFS device control OUT endpoint 0 control register (OTGFS_DOEPCTL0) 491
21.6.5.12 OTGFS device control OUT endpoint-x control register (OTGFS_DOEPCTLx) (x= x=1...7, where x if endpoint number) 492
21.6.5.13 OTGFS device IN endpoint-x interrupt register (OTGFS_DIEPINTx) (x=0...7, where x if endpoint number) 494
21.6.5.14 OTGFS device OUT endpoint-x interrupt register (OTGFS_DOEPINTx) (x=0...7, where x if endpoint number) 494
21.6.5.15 OTGFS device IN endpoint 0 transfer size register (OTGFS_DIEPTSIZ0) 495
21.6.5.16 OTGFS device OUT endpoint 0 transfer size register (OTGFS_DOEPTSIZ0) 495
21.6.5.17 OTGFS device IN endpoint-x transfer size register (OTGFS_DIEPTSIZx) (x=1...7, where x is endpoint number) 496
21.6.5.18 OTGFS device IN endpoint transmit FIFO status register (OTGFS_DTXFSTSx) (x=1...7, where x is endpoint number) 496
21.6.5.19 OTGFS device OUT endpoint-x transfer size register (OTGFS_DOEPTSIZx) (x=1...7, where x is endpoint number) 497
21.6.6 Power and clock control registers 497
21.6.6.1 OTGFS power and clock gating control register (OTGFS_PCGCCTL) 497

22 HICK auto clock calibration (ACC) 498

22.1 ACC introduction 498
22.2 Main features 498
22.3 Interrupt requests 498
22.4 Functional description 498
22.5 Principle 499
22.6 Register description 500
22.6.1 ACC register map 501
22.6.2 Status register (ACC_STS) 501
22.6.3 Control register 1 (ACC_CTRL1) 501
22.6.4 Control register 2 (ACC_CTRL2) 502

22.6.5 Compare value 1 (ACC_C1) 502
22.6.6 Compare value 2 (ACC_C2) 503
22.6.7 Compare value 3 (ACC_C3) 503

23 Infrared timer (IRTMR) 504

24 External memory controller (XMC) 505

24.1 XMC introduction 505
24.2 XMC main features 505
24.3 XMC architecture 507
24.3.1 Block diagram 507
24.3.2 Address mapping 508
24.4 NOR/PSRAM 511
24.4.1 Operating mode 511
24.4.2 Access mode 512
24.4.2.1 Read/write operation with the same timings 513
24.4.2.2 Read/write operation with different timings 516
24.4.2.3 Multiplexed mode 524
24.4.2.4 Synchronous mode 526
24.4.3 NAND 528
24.4.4 Operating mode 528
24.4.5 Access timings 529
24.4.6 ECC computation 530
24.5 PC card 530
24.5.1 Operating mode 530
24.5.2 Access timings 531
24.6 SDRAM card 532
24.6.1 SDRAM access management 532
24.6.2 Self-refresh mode and Power-down mode 535
24.7 XMC registers 535
24.7.1 NOR Flash and PSRAM control registers 536
24.7.1.1 SRAM/NOR Flash chip select control register 1 (XMC_BK1CTRL1) 536
24.7.1.2 SRAM/NOR Flash chip select control register x (x=2, 3, 4) 537
24.7.1.3 SRAM/NOR Flash chip select timing register x (x=1,2,3,4) 539
24.7.1.4 SRAM/NOR Flash write timing register x (x=1,2,3,4) 539
24.7.1.5 SRAM/NOR Flash extra timing register x(XMC_EXTx) (x=1,2,3,4) 540

24.7.2 NAND Flash control registers 541
24.7.2.1 NAND Flash control register x (XMC_BKxCTRL) (x=2,3) 541
24.7.2.2 Interrupt enable and FIFO status register x (XMC_BKxIS) (x=2,3) 541
24.7.2.3 Regular memory timing register x (XMC_ BKxTMGRG) (x=2,3) . 542
24.7.2.4 Special memory timing register x (XMC_ BKxTMGSP) (x=2,3) .. 542
24.7.2.5 ECC value register x (XMC_ BKxCC) (x=2,3) 543
24.7.3 PC card controller registers 543
24.7.3.1 PC card control register (XMC_BK4CTRL) 543
24.7.3.2 Interrupt enable and FIFO status register 4 (XMC_BK4IS) 543
24.7.3.3 Common memory timing register 4 (XMC_ BK4TMGCM) 544
24.7.3.4 Attribute memory timing register 4 (XMC_ BK4TMGAT) 544
24.7.3.5 IO space timing register 4 (XMC_ BK4TMGIO) 545
24.7.4 SDRAM controller registers 545
24.7.4.1 SDRAM control register 1, 2 (SDRAM_CTRL1,SDRAM_CTRL2) 545
24.7.4.2 SDRAM timing register 1, 2 (SDRAM_TM1,SDRAM_TM2) 546
24.7.4.3 SDRAM command register (SDRAM_CMD) 548
24.7.4.4 SDRAM refresh timer register (SDRAM_RCNT) 548
24.7.4.5 SDRAM status register (SDRAM_STS) 549

25 SDIO interface 550

25.1 SDIO introduction 550
25.2 SDIO main features 550
25.3 Functional description 552
25.3.1 Card functional description 552
25.3.1.1 Card identification mode 552
25.3.1.2 Data transfer mode 553
25.3.1.3 Erase 554
25.3.1.4 Protection management 554
25.3.2 Commands and responses 557
25.3.2.1 Commands 557
25.3.2.2 Response formats 561
25.3.3 SDIO functional description 563
25.3.3.1 SDIO adapter 564
25.3.3.2 Data BUF 568
25.3.3.3 SDIO AHB interface 568
25.3.3.4 Hardware flow control 569
25.3.4 SDIO I/O card-specific operations 569

25.4 SDIO registers 570
25.4.1 SDIO power control register (SDIO_ PWRCTRL) 570
25.4.2 SDIO clock control register (SDIO_ CLKCTRL) 571
25.4.3 SDIO argument register (SDIO_ARG) 572
25.4.4 SDIO command register (SDIO_CMD) 572
25.4.5 SDIO command response register (SDIO_RSPCMD) 573
25.4.6 SDIO response 1..4 register (SDIO_RSPx) 573
25.4.7 SDIO data timer register (SDIO_DTTMR) 573
25.4.8 SDIO data length register (SDIO_DTLEN) 573
25.4.9 SDIO data control register (SDIO_DTCTRL) 574
25.4.10 SDIO data counter register (SDIO_DTCNTR) 575
25.4.11 SDIO status register (SDIO_STS) 575
25.4.12 SDIO clear interrupt register (SDIO_INTCLR) 576
25.4.13 SDIO interrupt mask register (SDIO_INTEN) 576
25.4.14 SDIOBUF counter register (SDIO_BUFCNTR) 578
25.4.15 SDIO data BUF register (SDIO_BUF) 578

26 Ethernet media access control (EMAC) 579

26.1 EMAC introduction 579
26.1.1 EMAC structure 579
26.1.2 EMAC main features 579
26.2 EMAC functional description 580
26.2.1 EMAC communication interfaces 580
26.2.2 EMAC frame communication 585
26.2.3 Ethernet frame transmission and reception using DMA 593
26.2.4 Enter and wake up EMAC power-down mode 604
26.2.5 IEEE1588 precision time protocol 605
26.2.6 EMAC interrupts 609
26.3 EMAC registers 610
26.3.1 Ethernet MAC configuration register (EMAC_MACCTRL) 612
26.3.2 Ethernet MAC frame filter register (EMAC_MACFRMF) 614
26.3.3 Ethernet MAC Hash table high register (EMAC_MACHTH) 616
26.3.4 Ethernet MAC Hash table low register (EMAC_MACHTL) 616
26.3.5 Ethernet MAC MII address register (EMAC_MACMIIADDR) 616
26.3.6 Ethernet MAC MII data register (EMAC_MACMIIDT) 617

26.3.7 Ethernet MAC flow control register (EMAC_MACFCTRL) 617
26.3.8 Ethernet MAC VLAN tag register (EMAC_MACVLT).....619
26.3.9 Ethernet MAC remote wakeup frame filter register (EMAC_MACRWFF)619
26.3.10 Ethernet MAC PMT control and status register (EMAC_MACPMTCTRLSTS) 620
26.3.11 Ethernet MAC interrupt status register (EMAC_MACISTS) 620
26.3.12 Ethernet MAC interrupt mask register (EMAC_MAIMR) 621
26.3.13 Ethernet MAC address 0 high register (EMAC_MACA0H).....621
26.3.14 Ethernet MAC address 0 low register (EMAC_MACA0L).....622
26.3.15 Ethernet MAC address 1 high register (EMAC_MACA1H).....622
26.3.16 Ethernet MAC address 1 low register (EMAC_MACA1H) 622
26.3.17 Ethernet MAC address 2 high register (EMAC_MACA2H).....623
26.3.18 Ethernet MAC address 2 low register (EMAC_MACA2L).....623
26.3.19 Ethernet MAC address 3 high register (EMAC_MACA3H).....623
26.3.20 Ethernet MAC address 3 low register (EMAC_MACA3L).....624
26.3.21 Ethernet DMA bus mode register (EMAC_DMABM).....624
26.3.22 Ethernet DMA transmit poll demand register (EMAC_DMATPD)..626
26.3.23 Ethernet DMA receive poll demand register (EMAC_DMARPD) ..626
26.3.24 Ethernet DMA receive descriptor list address register (EMAC_DMARDLADDR) 626
26.3.25 Ethernet DMA transmit descriptor list address register (EMAC_DMATDLADDR) 627
26.3.26 Ethernet DMA status register (EMAC_DMASTS).....627
26.3.27 Ethernet DMA operation mode register (EMAC_DMAOPM).....629
26.3.28 Ethernet DMA interrupt enable register (EMAC_DMAIE) 632
26.3.29 Ethernet DMA missed frame and buffer overflow counter register (EMAC_DMAMFBOCNT).....633
26.3.30 Ethernet DMA current transmit descriptor register (EMAC_DMACTD)634
26.3.31 Ethernet DMA current receive descriptor register (EMAC_DMACRD)634
26.3.32 Ethernet DMA current transmit buffer address register (EMAC_DMACTBADDR) 634
26.3.33 Ethernet DMA current receive buffer address register (EMAC_DMACRBADDR).....634
26.3.34 Ethernet MMC control register (EMAC_MMCCTRL) 634
26.3.35 Ethernet MMC receive interrupt register (EMAC_MMCRI) 635
26.3.36 Ethernet MMC transmit interrupt register (EMAC_MMCTI).....635

26.3.37 Ethernet MMC receive interrupt register (EMAC_MMCRIM) 636
26.3.38 Ethernet MMC transmit interrupt register (EMAC_MMCTIM) 636
26.3.39 Ethernet MMC transmitted good frame single collision counter register (EMAC_MMCTFSCC) 636
26.3.40 Ethernet MMC transmitted good frame more than a single collision counter register (EMAC_MMCTFMSCC) 637
26.3.41 Ethernet MMC transmitted good frames counter register (EMAC_MMCTFCNT) 637
26.3.42 Ethernet MMC received frames with CRC error counter register (EMAC_MMCRFCECR).....637
26.3.43 Ethernet MMC received frames with alignment error counter register (EMAC_MMCRFAECNT).....637
26.3.44 Ethernet MMC received good unicast frames counter register (EMAC_MMCRGUFCNT) 637
26.3.45 Ethernet PTP time stamp control register (EMAC_PTPTSCTRL) .637
26.3.46 Ethernet PTP subsecond increment register (EMAC_PTPSSINC)639
26.3.47 Ethernet PTP time stamp high register (EMAC_PTPTSH).....639
26.3.48 Ethernet PTP time stamp low register (EMAC_PTPTSL) 640
26.3.49 Ethernet PTP time stamp high update register (EMAC_PTPTSHUD)640
26.3.50 Ethernet PTP time stamp low update register (EMAC_PTPTSLUD)640
26.3.51 Ethernet PTP time stamp addend register (EMAC_PTPTSAD) 640
26.3.52 Ethernet PTP target time high register (EMAC_PTPTTH) 641
26.3.53 Ethernet PTP target time low register (EMAC_PTPTTL) 641
26.3.54 Ethernet PTP time stamp status register (EMAC_PTPTSSR) 641
26.3.55 Ethernet PTP PPS register (EMAC_PTPPPSCR) 641

27 Digital Video parallel interface (DVP)..... 643

27.1 Introduction 643
27.2 Introduction 643
27.3 Data capture and synchronization 643
27.3.1 Hardware synchronization mode.....644
27.3.2 Embedded data synchronization mode.....645
27.3.3 Data alignment.....646
27.3.4 Single frame and continuous capture modes 647
27.4 DMA access interface and data output packing 648
27.4.1 DMA access interface.....648

27.4.2 Data output packing 648
27.5 Interrupts and interrupt control 649
27.6 Functional overview 650
27.6.1 Frame rate control 650
27.6.2 Crop window 650
27.6.3 Image resizing 651
27.6.4 Grayscale image binarization conversion 654
27.7 Data formats 654
27.7.1 Common CMOS video camera output formats 654
27.7.2 JPEG compressed format 656
27.7.3 Enhanced data management 656
27.7.4 Data format conversion 656
27.8 Registers 657
27.8.1 DVP control register (DVP_CTRL) 657
27.8.2 DVP status register (DVP_STS) 659
27.8.3 DVP event status register (DVP_ESTS) 659
27.8.4 DVP interrupt enable register (DVP_IENA) 660
27.8.5 DVP interrupt status register (DVP_ISTS) 660
27.8.6 DVP interrupt clear register (DVP_ICLR) 660
27.8.7 DVP embedded synchronization code register (DVP_SCR) 661
27.8.8 DVP embedded synchronization unmask register (DVP_SUR) 661
27.8.9 DVP crop window start register (DVP_CWST) 662
27.8.10 DVP crop window size register (DVP_CWSZ) 662
27.8.11 DVP data register (DVP_DT) 662
27.8.12 DVP advanced control register (DVP_ACTRL) 662
27.8.13 DVP enhanced horizontal scaling factor register (DVP_HSCF) 664
27.8.14 DVP enhanced vertical scaling factor register (DVP_VSCF) 664
27.8.15 DVP enhanced frame rate control factor register (DVP_FRF) 664
27.8.16 DVP binarization threshold register (DVP_BTH) 664

28 Qud-SPI interface (QSPI) 665

28.1 Introduction 665
28.2 QSPI main features 665
28.3 QSPI command slave port 665
28.3.1 QSPI command slave port 665

28.3.2 CPU PIO mode 665
28.3.3 DMA handshake mode 665
28.3.4 XIP port (direct address mapping read/write) 666
28.3.5 XIP port prefetch 666
28.3.6 SPI device operation 666
28.4 QSPI registers 672
28.4.1 Command word 0 (CMD_W0) 672
28.4.2 Command word 1 (CMD_W1) 672
28.4.3 Command word 2 (CMD_W2) 673
28.4.4 Command word 3 (CMD_W3) 673
28.4.5 Control register (CTRL) 674
28.4.6 AC timing register (ACTR) 675
28.4.7 FIFO status register (FIFOSTS) 675
28.4.8 Control register 2 (CTRL2) 676
28.4.9 Command status register (CMDSTS) 676
28.4.10 Read status register (RSTS) 676
28.4.11 Flash size register (FSIZE) 677
28.4.12 XIP command word 0 (XIP CMD_W0) 677
28.4.13 XIP command word 1 (XIP CMD_W1) 677
28.4.14 XIP command word 2 (XIP CMD_W2) 678
28.4.15 XIP command word 3 (XIP CMD_W3) 679
28.4.16 Revision register (REV) 679
28.4.17 Data port register (DT) 679

29 EDMA controller (EDMA) 680

29.1 Introduction 680
29.2 Main features 680
29.3 Functional overview 681
29.3.1 Flow configuration 681
29.3.2 Channel selection and synchronizer 682
29.3.3 Handshake mechanism 682
29.3.4 Arbiter 682
29.3.5 Programmable data transfer width 683
29.3.6 FIFO and threshold 684
29.3.7 Linked table transfer mechanism 685

29.3.8 2D transfer mechanism 686
29.3.9 Errors 687
29.3.10 Interrupts 688
29.4 DMA multiplexer (DMAMUX) 688
29.4.1 DMAMUX functional overview 688
29.4.2 DMAMUX overflow interrupts 690
29.5 EDMA registers 692
29.5.1 DMA status register 1 (DMA_STS1) 694
29.5.2 DMA status register 2 (DMA_STS2) 695
29.5.3 DMA flag clear register 1 (DMA_CLR1) 696
29.5.4 DMA flag clear register 2 (DMA_CLR2) 697
29.5.5 DMA stream-x control register (DMA_SxCTRL) (x= 1...8) 698
29.5.6 DMA stream-x data register (DMA_SxDTCNT) (x= 1...8) 700
29.5.7 DMA stream-x peripheral address register (DMA_SxPADDR) (x= 1...8) 700
29.5.8 DMA stream-x memory 0 address register (DMA_SxM0ADDR) (x= 1...8) 701
29.5.9 DMA stream-x memory 1 address register (DMA_SxM1ADDR) (x= 1...8) 701
29.5.10 DMA stream-x FIFO control register (DMA_SxFCTRL) (x= 1...8) 701
29.5.11 DMA linked table control register (DMA_SxLLCTRL) 702
29.5.12 DMA linked table pointer register (DMA_SxLLP) (x = 1...8) 702
29.5.13 DMA 2D transfer control register (DMA_S2DCTRL) 702
29.5.14 DMA 2D transfer count register (DMA_S2DCNT) 702
29.5.15 DMA 2D transfer stride register (DMA_STRIDE)(x = 1...8) 703
29.5.16 DMA synchronization enable (DMA_SYNCEN) 703
29.5.17 DMAMUX table select (DMA_MUXSEL) 704
29.5.18 DMAMUX channel-x control register (DMA_MUXSxCTRL) (x = 1...8) 704
29.5.19 DMAMUX generator-x control register (DMA_MUXGxCTRL) (x = 1...8) 705
29.5.20 DMAMUX synchronization interrupt status register (DMA_MUXSYNCSTS) 705
29.5.21 DMAMUX synchronization interrupt clear flag register (BPR_MUXSYNCCLR) 705
29.5.22 DMAMUX generator interrupt status register (DMA_ MUXGSTS) 706

29.5.23 DMAMUX generator interrupt clear flag register (DMA_ MUXGCLR)706

30 Debug (DEBUG) 707

30.1 Debug introduction..... 707

30.2 Debug and Trace 707

30.3 I/O pin control..... 707

30.4 DEGUB registers 707

30.4.1 DEBUG device ID (DEBUG_IDCODE).....708

30.4.2 DEBUG control register (DEBUG_CTRL) 709

30.4.3 DEBUG APB1 pause register (DEBUG_ APB1_PAUSE) 709

30.4.4 DEBUG APB2 pause register (DEBUG_ APB2_PAUSE) 710

30.4.5 MCU SERIES ID register (DEBUG_ SER_ID) 711

31 Revision history..... 712

List of figures

Figure 1-1 AT32F435/437 Series microcontrollers system architecture 45
Figure 1-2 Internal block diagram of Cortex®-M4F 46
Figure 1-3 Internal block diagram of AHB BusMatrix. 47
Figure 1-4 Comparison between bit-band region and its alias region: image A 48
Figure 1-5 Comparison between bit-band region and its alias region: image B 48
Figure 1-6 Reset process 54
Figure 1-7 Example of MSP and PC initialization 55
Figure 2-1 AT32F435/437 address mapping 57
Figure 3-1 Block diagram of each power supply 63
Figure 3-2 Power-on reset/Low voltage reset waveform 64
Figure 3-3 PVM threshold and output 64
Figure 4-1 AT32F435/437 clock tree 70
Figure 4-2 System reset circuit 73
Figure 5-1 Flash memory sector erase process 102
Figure 5-2 Flash memory block erase process 103
Figure 5-3 Flash memory mass erase process 104
Figure 5-4 Flash memory programming process 105
Figure 5-5 System data area erase process 106
Figure 5-6 System data area programming process 107
Figure 6-1 GPIO basic structure 118
Figure 6-2 IOMUX structure 120
Figure 8-1 External interrupt/Event controller block diagram 148
Figure 9-1 DMA block diagram 151
Figure 9-2 Re-arbitrate after request/acknowledge 153
Figure 9-3 PWIDTH: byte, MWIDTH: half-word 153
Figure 9-4 PWIDTH: half-word, MWIDTH: word 154
Figure 9-5 PWIDTH: word, MWIDTH: byte 154
Figure 9-6 DMAMUX block diagram 155
Figure 9-7 DMAMUX request synchronized mode 157
Figure 9-8 DMAMUX event generation 158
Figure 10-1 CRC calculation unit block diagram 168
Figure 10-2 Diagram of byte reverse 169
Figure 11-1 I²C bus protocol 171
Figure 11-2 I²C function block diagram 172
Figure 11-3 Setup and hold time 174
Figure 11-4 I²C master transmission flow 178
Figure 11-5 Transfer sequence of I²C master transmitter 179
Figure 11-6 I²C master receive flow 179
Figure 11-7 Transfer sequence of I²C master receiver 180
Figure 11-8 10-bit address read access when READH10=1 180
Figure 11-9 10-bit address read access when READH10=0 180
Figure 11-10 I²C slave transmission flow 183
Figure 11-11 I²C slave transmission timing 183

Figure 11-12 I²C slave receive flow 184
Figure 11-13 I²C slave receive timing 184
Figure 11-14 SMBus master transmission flow 189
Figure 11-15 SMBus master transmission timing 190
Figure 11-16 SMBus master receive flow 190
Figure 11-17 SMBus master receive timing 191
Figure 11-18 SMBus slave transmission flow 193
Figure 11-19 SMBus slave transmission timing 193
Figure 11-20 SMBus slave receive flow 194
Figure 11-21 SMBus slave receive timing 194
Figure 12-1 USART block diagram 204
Figure 12-2 BFF and FERR detection in LIN mode 207
Figure 12-3 Smartcard frame format 207
Figure 12-4 IrDA DATA(3/16) – normal mode 208
Figure 12-5 Hardware flow control 208
Figure 12-6 Mute mode using Idle line or Address mark detection 209
Figure 12-7 8-bit format USART synchronous mode 209
Figure 12-8 Word length 210
Figure 12-9 Stop bit configuration 211
Figure 12-10 TDC/TDBE behavior when transmitting 214
Figure 12-11 Data sampling for noise detection 216
Figure 12-12 Tx/Rx swap 217
Figure 12-13 USART interrupt map diagram 218
Figure 13-1 SPI block diagram 224
Figure 13-2 SPI two-wire unidirectional full-duplex connection 225
Figure 13-3 Single-wire unidirectional receive only in SPI master mode 226
Figure 13-4 Single-wire unidirectional receive only in SPI slave mode 226
Figure 13-5 Single-wire bidirectional half-duplex mode 227
Figure 13-6 Master full-duplex communications 231
Figure 13-7 Slave full-duplex communications 232
Figure 13-8 Master half-duplex transmit 232
Figure 13-9 Slave half-duplex receive 232
Figure 13-10 Slave half-duplex transmit 233
Figure 13-11 Master half-duplex receive 233
Figure 13-12 TI mode continous transfer 233
Figure 13-13 TI mode continous transfer with dummy CLK 234
Figure 13-14 TI mode continous transfer with dummy CLK 234
Figure 13-15 SPI interrupts 234
Figure 13-16 I²S block diagram 235
Figure 13-17 I²S full-duplex structure 236
Figure 13-18 I²S slave device transmission 236
Figure 13-19 I²S slave device reception 237
Figure 13-20 I²S master device transmission 237
Figure 13-21 I²S master device reception 237
Figure 13-22 CK & MCK source in master mode 239

Figure 13-23 Audio standard timings 242
Figure 13-24 I²S interrupts 243
Figure 14-1 Basic timer block diagram 250
Figure 14-2 Control circuit with CK_INT divided by 1 250
Figure 14-3 Counter structure 251
Figure 14-4 Overflow event when PRBEN=0 251
Figure 14-5 Overflow event when PRBEN=1 251
Figure 14-6 Counter timing diagram with internal clock divided by 4 251
Figure 14-7 General-purpose timer block diagram 254
Figure 14-8 Count clock 255
Figure 14-9 Use CK_INT to drive counter with TMRx_DIV=0x0 and TMRx_PR=0x16 255
Figure 14-10 Block diagram of external clock mode A 256
Figure 14-11 Counting in external clock mode A, with PR=0x32 and DIV=0x0 256
Figure 14-12 Block diagram of external clock mode B 257
Figure 14-13 Counting in external clock mode B, with PR=0x32 and DIV=0x0 257
Figure 14-14 Counter timing with prescaler value changing from 1 to 4 258
Figure 14-15 Counter structure 258
Figure 14-16 Overflow event when PRBEN=0 259
Figure 14-17 Overflow event when PRBEN=1 259
Figure 14-18 Counter timing diagram with internal clock divided by 4 259
Figure 14-19 Counter timing diagram with internal clock divided by 1 and TMRx_PR=0x32 260
Figure 14-20 Encoder mode structure 260
Figure 14-21 Example of counter behavior in encoder interface mode (encoder mode C) 261
Figure 14-22 Input/output channel 1 main circuit 262
Figure 14-23 Channel 1 input stage 262
Figure 14-24 Example of PWM input mode configuration 263
Figure 14-25 PWM input mode 263
Figure 14-26 Capture/compare channel output stage (channel 1 to 4) 264
Figure 14-27 C1ORAW toggles when counter value matches the C1DT value 265
Figure 14-28 Upcounting mode and PWM mode A 265
Figure 14-29 Up/down counting mode and PWM mode A 265
Figure 14-30 One-pulse mode 266
Figure 14-31 Clearing CxORAW (PWM mode A) by EXT input 266
Figure 14-32 Example of reset mode 267
Figure 14-33 Example of suspend mode 267
Figure 14-34 Example of trigger mode 267
Figure 14-35 Master/slave timer connection 268
Figure 14-36 Using master timer to start slave timer 268
Figure 14-37 Starting master and slave timers synchronously by an external trigger 269
Figure 14-38 Block diagram of general-purpose TMR9/12 281
Figure 14-39 Block diagram of general-purpose TMR10/11/13/14 282
Figure 14-40 Count clock 282
Figure 14-41 Use CK_INT to drive counter, with TMRx_DIV=0x0 and TMRx_PR=0x16 282
Figure 14-42 Block diagram of external clock mode A 283
Figure 14-43 Counting in external clock mode A, with PR=0x32 and DIV=0x0 283

Figure 14-44 Counter timing with prescaler value changing from 1 to 4 284
Figure 14-45 Counter structure 285
Figure 14-46 Overflow event when PRBEN=0 286
Figure 14-47 Overflow event when PRBEN=1 286
Figure 14-48 Input/output channel 1 main circuit 287
Figure 14-49 Channel 1 input stage 287
Figure 14-50 Example of PWM input mode configuration 288
Figure 14-51 PWM input mode 288
Figure 14-52 Capture/compare channel output stage (channel 1) 289
Figure 14-53 C1ORAW toggles when counter value matches the C1DT value 290
Figure 14-54 Upcounting mode and PWM mode A 290
Figure 14-55 One-pulse mode 290
Figure 14-56 Example of reset mode 291
Figure 14-57 Example of suspend mode 291
Figure 14-58 Example of trigger mode 292
Figure 14-59 Block diagram of advanced-control timer 303
Figure 14-60 Count clock 304
Figure 14-61 Use CK_INT to drive counter, with TMRx_DIV=0x0 and TMRx_PR=0x16 304
Figure 14-62 Block diagram of external clock mode A 305
Figure 14-63 Counting in external clock mode A, with PR=0x32 and DIV=0x0 305
Figure 14-64 Block diagram of external clock mode B 306
Figure 14-65 Counting in external clock mode B, with PR=0x32 and DIV=0x0 306
Figure 14-66 Counter timing with prescaler value changing from 1 to 4 307
Figure 14-67 Counter structure 307
Figure 14-68 Overflow event when PRBEN=0 308
Figure 14-69 Overflow event when PRBEN=1 308
Figure 14-70 Counter timing diagram with internal clock divided by 4 308
Figure 14-71 Counter timing diagram with internal clock divided by 1 and TMRx_PR=0x32 309
Figure 14-72 OVFIF in upcounting mode and up/down counting mode 309
Figure 14-73 Encoder mode structure 310
Figure 14-74 Example of encoder interface mode C 311
Figure 14-75 Input/output channel 1 main circuit 311
Figure 14-76 Channel 1 input stage 312
Figure 14-77 Example of PWM input mode configuration 313
Figure 14-78 PWM input mode 313
Figure 14-79 Channel output stage (channel 1 to 3) 313
Figure 14-80 Channel 4 output stage 314
Figure 14-81 C1ORAW toggles when counter value matches the C1DT value 315
Figure 14-82 Upcounting mode and PWM mode A 315
Figure 14-83 Up/down counting mode and PWM mode A 316
Figure 14-84 One-pulse mode 316
Figure 14-85 Clearing CxORAW(PWM mode A) by EXT input 317
Figure 14-86 Complementary output with dead-time insertion 317
Figure 14-87 TMR output control 318
Figure 14-88 Example of TMR break function 319

Figure 14-89 Example of reset mode 319
Figure 14-90 Example of suspend mode 320
Figure 14-91 Example of trigger mode 320
Figure 15-1 Window watchdog block diagram 335
Figure 15-2 Window watchdog timing diagram 336
Figure 16-1 WDT block diagram 339
Figure 17-1 ERTC block diagram 341
Figure 18-1 ADC1 block diagram 359
Figure 18-2 ADC basic operation process 361
Figure 18-3 ADC power-on and calibration 362
Figure 18-4 Sequence mode 364
Figure 18-5 Preempted group auto conversion mode 364
Figure 18-6 Repetition mode 365
Figure 18-7 Partition mode 365
Figure 18-8 ADABRT timing diagram 366
Figure 18-9 Ordinary oversampling restart mode selection 367
Figure 18-10 Ordinary oversampling trigger mode 368
Figure 18-11 Oversampling of preempted group of channels 368
Figure 18-12 Data alignment 369
Figure 18-13 Block diagram of master/salve mode 370
Figure 18-14 Regular simultaneous mode 372
Figure 18-15 Regular simultaneous mode 372
Figure 18-16 Alternate preempted trigger mode 373
Figure 18-17 Regular shift mode 373
Figure 18-18 Regular shift mode and DMA mode 2 374
Figure 19-1 DAC1/DAC2 block diagram 391
Figure 19-2 LFSR register calculation algorithm 393
Figure 19-3 Triangular-wave generation 393
Figure 20-1 Bit timing 399
Figure 20-2 Transmit interrupt generation 401
Figure 20-3 Transmit interrupt generation 402
Figure 20-4 Receive interrupt 0 generation 402
Figure 20-5 Receive interrupt 1 generation 402
Figure 20-6 Status error interrupt generation 402
Figure 20-7 CAN block diagram 403
Figure 20-8 32-bit identifier mask mode 405
Figure 20-9 32-bit identifier list mode 405
Figure 20-10 16-bit identifier mask mode 405
Figure 20-11 16-bit identifier list mode 406
Figure 20-12 Transmit mailbox status 407
Figure 20-13 Receive FIFO status 408
Figure 20-14 Transmit and receive mailboxes 419
Figure 21-1 Block diagram of OTGFS structure 423
Figure 21-2 OTGFS interrupt hierarchy 425
Figure 21-3 Writing the transmit FIFO 430

Figure 21-4 Reading the receive FIFO 431
Figure 21-5 HFIR behavior when HFIRRLDCTRL=0x0 432
Figure 21-6 HFIR behavior when HFIRRLDCTRL=0x1 433
Figure 21-7 Example of common Bulk/Control OUT/SETUP and Bulk/Control IN transfer 436
Figure 21-8 Example of common interrupt OUT/IN transfers 440
Figure 21-9 Example of common synchronous OUT/IN transfers 443
Figure 21-10 Read receive FIFO 448
Figure 21-11 SETUP data packet flowchart 450
Figure 21-12 BULK OUT transfer block diagram 454
Figure 21-13 CSR memory map 460
Figure 22-1 ACC interrupt mapping diagram 498
Figure 22-2 ACC block diagram 499
Figure 22-3 Cross-return algorithm 500
Figure 23-1 IRTMR block diagram 504
Figure 24-1 XMC block diagram 507
Figure 24-2 XMC memory banks 509
Figure 24-3 NOR/PSRAM mode 1 read access 514
Figure 24-4 NOR/PSRAM mode 1 write access 514
Figure 24-5 NOR/PSRAM mode 2 read access 515
Figure 24-6 NOR/PSRAM mode 2 write access 516
Figure 24-7 NOR/PSRAM mode A read access 517
Figure 24-8 NOR/PSRAM mode A write access 518
Figure 24-9 NOR/PSRAM mode B read access 519
Figure 24-10 NOR/PSRAM mode B write access 520
Figure 24-11 NOR/PSRAM mode C read access 521
Figure 24-12 NOR/PSRAM mode C write access 522
Figure 24-13 NOR/PSRAM mode D read access 523
Figure 24-14 NOR/PSRAM mode D write access 524
Figure 24-15 NOR/PSRAM multiplexed mode read access 525
Figure 24-16 NOR/PSRAM multiplexed mode write access 526
Figure 24-17 NOR/PSRAM synchronous multiplexed mode read access 527
Figure 24-18 NOR/PSRAM synchronous multiplexed mode write access 528
Figure 24-19 NAND read access 529
Figure 24-20 NAND wait functionality 530
Figure 24-21 PC card read/write 532
Figure 24-22 SDRAM write access waveforms (Trcd=2, 9 consecutive write access) 533
Figure 24-23 SDRAM read access without using read FIFO (Trcd=2,CL=3, and 4 consecutive read accesses) 534
Figure 25-1 SDIO “no response” and “response” operations 551
Figure 25-2 SDIO multiple block read operation 551
Figure 25-3 SDIO multiple block write operation 551
Figure 25-4 SDIO sequential read operation 552
Figure 25-5 SDIO sequential write operation 552
Figure 25-6 SDIO block diagram 564
Figure 25-7 Command channel state machine (CCSM) 566

Figure 25-8 SDIO command transfer 567
Figure 25-9 Data channel state machine (DCSM) 567
Figure 26-1 Block diagram of EMAC 579
Figure 26-2 SMI interface signals 581
Figure 26-3 MII signals 582
Figure 26-4 Reduced media-independent interface signals 583
Figure 26-5 MII clock sources (provided by CLKOUT pin) 584
Figure 26-6 MII clock sources (provided by an external oscillator) 584
Figure 26-7 RMII clock sources (provided by an external crystal oscillator) 584
Figure 26-8 MAC frame format 585
Figure 26-9 Tagged MAC frame format 586
Figure 26-10 Descriptor for ring and chain structure 593
Figure 26-11 Transmit descriptors 596
Figure 26-12 RXDMA descriptor structure 601
Figure 26-13 Wakeup frame filter register 604
Figure 26-14 System time update using the fine correction method 607
Figure 26-15 PTP trigger output to TMR2 ITR1 connection 608
Figure 26-16 PPS output 609
Figure 26-17 Ethernet interrupts 610
Figure 26-18 Ethernet MAC remote wakeup frame filter register (EMAC_MACRWFF) 619
Figure 27-1 DVP block diagram 643
Figure 27-2 CMOS video camera output in Frame start type 644
Figure 27-3 CMOS video camera output in Frame valid type 644
Figure 27-4 FS/FE/LS/LE frame composition 645
Figure 27-5 SAV/EAV frame composition 646
Figure 27-6 Block diagram in single frame capture mode 647
Figure 27-7 Block diagram in continuous capture mode 648
Figure 27-8 PDL configuration and data output packing 649
Figure 27-9 Block diagram of frame rate control feature 650
Figure 27-10 Crop window block diagram 651
Figure 27-11 Image resizing block diagram 651
Figure 27-12 LCDC/LCDS and frame structure 652
Figure 27-13 PCDC/PCDS and line structure 653
Figure 27-14 RGB565-format data capture and packing 654
Figure 27-15 RGB555-format data capture and packing 655
Figure 27-16 YUV422-format data capture and packing 655
Figure 27-17 Y8 (Y-only)-format data capture and packing 656
Figure 27-18 YUV422 format to Y8 (Y-only) format 657
Figure 28-1 Function block diagram 665
Figure 28-2 DMA handshake mode 666
Figure 28-3 Write enable 667
Figure 28-4 Page programming 667
Figure 28-5 Status read 667
Figure 28-6 Data read 667
Figure 28-7 Quick read dual output command 668

Figure 28-8 Quick read dual-wire I/O 668
Figure 28-9 Quick read quad output 668
Figure 28-10 Quick read quad I/O command 669
Figure 29-1 Block diagram 680
Figure 29-2 Channel select and synchronizer 682
Figure 29-3 Re-arbitrate after Request/Acknowledge 683
Figure 29-4 Example of packing mechanism 683
Figure 29-5 Example of unpacking mechanism 684
Figure 29-6 Example of PINCOS 684
Figure 29-7 Descriptor format 685
Figure 29-8 Linked list pointers 685
Figure 29-9 Example of a 2D transfer (source side is managed by a peripheral controller) 686
Figure 29-10 Example of a 2D transfer (destination side is managed by a memory controller) 687
Figure 29-11 DMAMUX block diagram 689
Figure 29-12 DMAMUX synchronized mode 690
Figure 29-13 DMAMUX event generation 691

List of tables

Table 1-1 Bit-band address mapping in SRAM 49
Table 1-2 Bit-band address mapping in the peripheral area 49
Table 1-3 AT32F435/437 series vector table 50
Table 1-4 List of abbreviations for registers 56
Table 1-5 List of abbreviations for registers 56
Table 2-1 Peripheral boundary address 60
Table 3-1 PW register map and reset values 67
Table 4-1 CRM register map and reset values 73
Table 5-1 Flash memory architecture (4032 K) 95
Table 5-2 Flash memory architecture (1024 K) 96
Table 5-3 Flash memory architecture (448 K) 96
Table 5-4 Flash memory architecture (256 K) 97
Table 5-5 User system data area 97
Table 5-6 Extended system options 100
Table 5-7 Flash memory access limit 108
Table 5-8 Flash memory interface—Register map and reset value 110
Table 6-1 Port A multiplexed function configuration with GPIOA_MUX* register 121
Table 6-2 Port B multiplexed function configuration with GPIOB_MUX* register 123
Table 6-3 Port C multiplexed function configuration with GPIOC_MUX* register 125
Table 6-4 Port D multiplexed function configuration with GPIOD_MUX* register 127
Table 6-5 Port E multiplexed function configuration with GPIOE_MUX* register 129
Table 6-6 Port F multiplexed function configuration with GPIOF_MUX* register 131
Table 6-7 Port G multiplexed function configuration with GPIOG_MUX* register 133
Table 6-8 Port H multiplexed function configuration with GPIOH_MUX* register 135
Table 6-9 Pins owned by hardware 135
Table 6-10 GPIO register map and reset values 136
Table 7-1 SCFG register map and reset values 140
Table 8-1 External interrupt/Event controller register map and reset value 149
Table 9-1 DMA error event 154
Table 9-2 DMA interrupt requests 154
Table 9-3 Flexible DMA1/DMA2 request mapping 156
Table 9-4 DMAMUX EXINT LINE for trigger input and synchronized input 157
Table 9-5 DMA register map and reset value 158
Table 10-1 CRC register map and reset value 169
Table 11-1 I²C timing specifications 175
Table 11-2 I²C configuration table 176
Table 11-3 SMBus timeout specification 186
Table 11-4 SMBus timeout detection configuration 186
Table 11-5 SMBus mode configuration 187
Table 11-6 I²C error events 195
Table 11-7 I²C interrupt requests 197
Table 11-8 I²C register map and reset values 197
Table 12-1 Error calculation for programmed baud rate 212

Table 12-2 Data sampling over start bit and noise detection 215
Table 12-3 Data sampling over valid data and noise detection..... 216
Table 12-4 USART interrupt request 217
Table 12-5 USART register map and reset value..... 218
Table 13-1 Audio frequency precision using system clock..... 239
Table 13-2 SPI register map and reset value 244
Table 14-1 TMR functional comparison..... 249
Table 14-2 TMR6 and TMR7— register table and reset value..... 252
Table 14-3 TMRx internal trigger connection..... 257
Table 14-4 Counting direction versus encoder signals..... 261
Table 14-5 TMRx register map and reset value 269
Table 14-6 Standard CxOUT channel output control bit..... 278
Table 14-7 TMRx internal trigger connection..... 284
Table 14-8 TMRx register map and reset value 292
Table 14-9 Standard CxOUT channel output control bit..... 297
Table 14-10 TMRx register map and reset value 298
Table 14-11 Standard CxOUT channel output control bit..... 302
Table 14-12 TMRx internal trigger connection..... 306
Table 14-13 Counting direction versus encoder signals..... 310
Table 14-14 TMR1 and TMR8 register map and reset value 321
Table 14-15 Complementary output channel CxOUT and CxCOUT control bits with break function..... 330
Table 15-1 Minimum and maximum timeout value when PCLK1=72 MHz..... 336
Table 15-2 WWDT register map and reset value 336
Table 16-1 WDT timeout period (LICK=40kHz)..... 339
Table 16-2 WDT register and reset value..... 339
Table 17-1 RTC register map and reset values..... 342
Table 17-2 ERTC low-power mode wakeup..... 348
Table 17-3 Interrupt control bits..... 348
Table 17-4 ERTC register map and reset values 348
Table 18-1 Trigger sources for ordinary channels 362
Table 18-2 Trigger sources for preempted channels..... 363
Table 18-3 Correlation between maximum cumulative data, oversampling multiple and shift digits..... 366
Table 18-4 Master/slave DMA mode..... 371
Table 18-5 ADC register map and reset values..... 374
Table 19-1 Trigger source selection..... 392
Table 19-2 DAC register map and reset values 394
Table 20-1 CAN register map and reset values 409
Table 21-1 OTGFS input/output pins..... 424
Table 21-2 OTGFS transmit FIFO SRAM allocation 426
Table 21-3 OTGFS internal storage space allocation 428
Table 21-4 OTGFS register map and reset values 461
Table 21-5 Minimum duration for software disconnect..... 485
Table 22-1 ACC interrupt requests 498
Table 22-2 ACC register map and reset values..... 501
Table 24-1 NOR/PSRAM pins 507

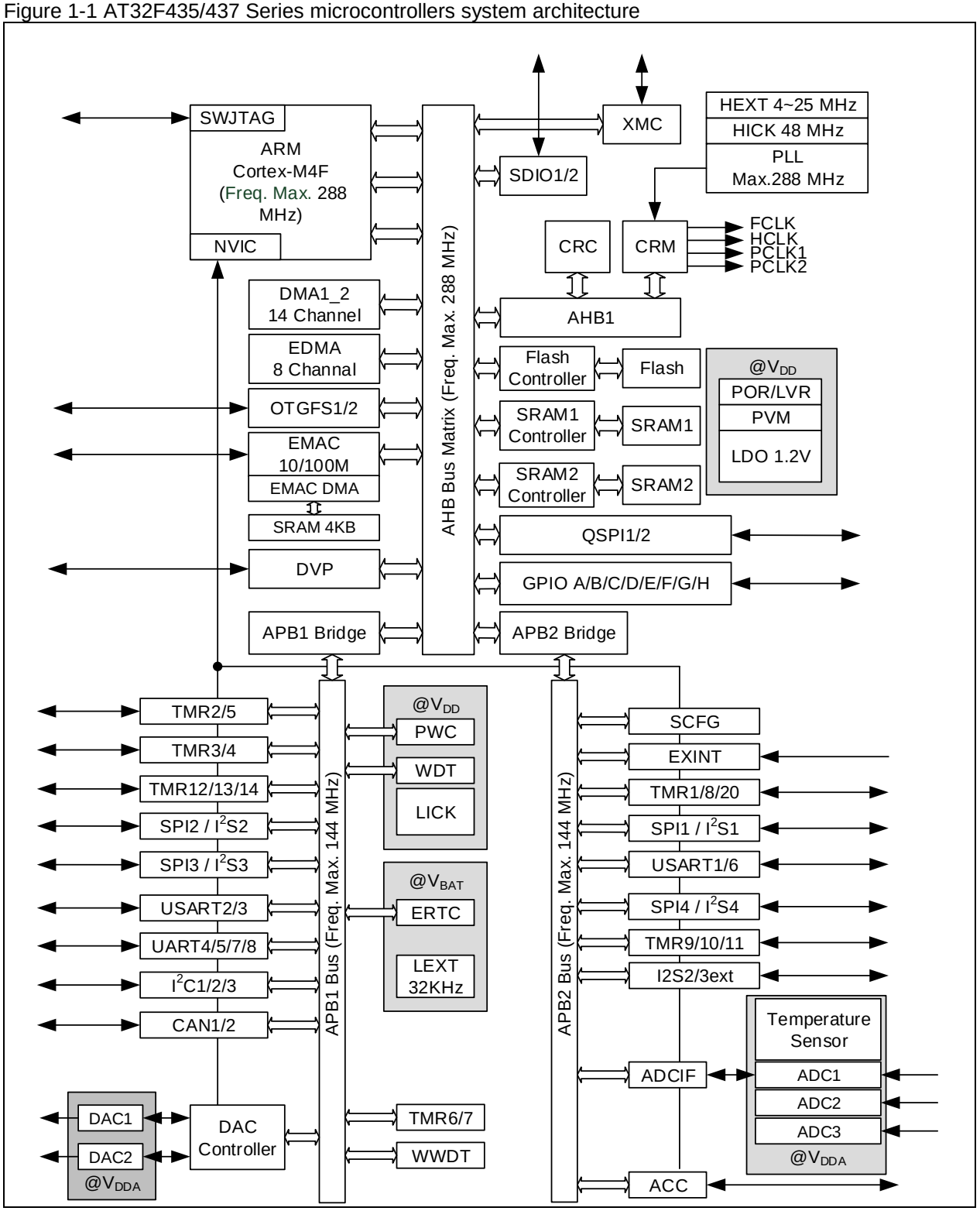
Table 24-2 NAND pins 508
Table 24-3 PC card pins 508
Table 24-4 SDRAM pins 508
Table 24-5 Memory bank selection 510
Table 24-6 8-bit SDRAM address mapping 510
Table 24-7 16-bit SDRAM address mapping 511
Table 24-8 Pin signals for NOR and PSRAM 511
Table 24-9 Address translation between HADDR and external memory 512
Table 24-10 Data access width vs. external memory data width 512
Table 24-11 NOR/PSRAM parameter registers 513
Table 24-12 Mode 1— SRAM/NOR Flash chip select control register (XMC_BK1CTRL) configuration 513
Table 24-13 Mode 1— SRAM/NOR Flash chip select timing register (XMC_ BK1TMG) configuration 513
Table 24-14 Mode 2 — SRAM/NOR Flash chip select control register 514
Table 24-15 Mode 2 — SRAM/NOR Flash chip select timing register 515
Table 24-16 Mode A— SRAM/NOR Flash chip select control register 516
Table 24-17 Mode A— SRAM/NOR Flash chip select timing register 517
Table 24-18 Mode A— SRAM/NOR Flash write timing register 517
Table 24-19 Mode B— SRAM/NOR Flash chip select register 518
Table 24-20 Mode B— SRAM/NOR Flash chip select timing register 519
Table 24-21 Mode B— SRAM/NOR Flash write timing register 519
Table 24-22 Mode C— SRAM/NOR Flash chip select register 520
Table 24-23 Mode C—SRAM/NOR Flash chip select timing register 521
Table 24-24 Mode C— SRAM/NOR Flash write timing register 521
Table 24-25 Mode D— SRAM/NOR Flash chip select register (XMC_BK1CTRL) configuration 522
Table 24-26 Mode D—SRAM/NOR Flash chip select timing register 523
Table 24-27 Mode D— SRAM/NOR Flash write timing register 523
Table 24-28 Multiplexed mode — SRAM/NOR Flash chip select control register 524
Table 24-29 Multiplexed mode—SRAM/NOR Flash chip select timing register (XMC_BK1TMG) configuration 525
Table 24-30 Synchronous mode — SRAM/NOR Flash chip select control register 526
Table 24-31 Synchronous mode—SRAM/NOR Flash chip select timing register (XMC_BK1TMG) 527
Table 24-32 Typical pin signals for NAND Flash 528
Table 24-33 Data access width vs. external memory data width 529
Table 24-34 NAND parameter registers 529
Table 24-35 lists the ECC result bits corresponding to the number of bytes 530
Table 24-36 Typical pin signals for PC card 531
Table 24-37 Access data width and PC card data width 531
Table 24-38 PC card parameter register 531
Table 24-39 XMC register address mapping 535
Table 25-1 Lock/unlock command structure 555
Table 25-2 Commands 557
Table 25-3 Data block read commands 558
Table 25-4 Data stream read/write commands 559
Table 25-5 Data block write commands 559
Table 25-6 Block-based write protect commands 559

Table 25-7 Erase commands 560
Table 25-8 I/O mode commands 560
Table 25-9 Card lock commands 560
Table 25-10 Application-specific commands 561
Table 25-11 R1 response 561
Table 25-12 R2 response 562
Table 25-13 R3 response 562
Table 25-14 R4 response 562
Table 25-15 R4b response 562
Table 25-16 R5 response 562
Table 25-17 R6 response 563
Table 25-18 SDIO pin definitions 564
Table 25-19 Command formats 565
Table 25-20 Short response format 565
Table 25-21 Long response format 565
Table 25-22 Command path status flags 566
Table 25-23 Data token formats 568
Table 25-24 A summary of the SDIO registers 570
Table 25-25 Response type and SDIO_RSPx register 573
Table 26-1 shows the clock range. 581
Table 26-2 Transmit interface signal encode 583
Table 26-3 Receive interface signal encode 583
Table 26-4 Ethernet peripheral pin configuration 585
Table 26-5 Destination address filtering 588
Table 26-6 Source address filtering 588
Table 26-7 Receive descriptor 0 602
Table 26-8 Ethernet register map and its reset values 610
Table 27-1 DVP pin use in hardware synchronization mode 645
Table 27-2 DVP pin use in embedded synchronization mode 646
Table 27-3 DVP register configuration and DVP_D pin use 647
Table 27-4 Enhanced features and DVP data formats supported 656
Table 27-5 DVP register map and reset values 657
Table 28-1 SPI register map and reset values 672
Table 29-1 DMA error events 687
Table 29-2 DMA interrupts 688
Table 29-3 EDMA flexible request mapping 689
Table 29-4 DMAMUX EXINT LINE for trigger input and synchronized input 690
Table 29-5 BPR register map and reset values 692
Table 30-1 DEBUG register address and reset value 707

1 System architecture

AT32F435/437 series microcontrollers incorporates a 32-bit ARM® Cortex®-M4F processor core, multiple 16-bit and 32-bit timers, Infrared Transmitter (IRTMR), DMA controller, EDMA controller, ERTC, communication interfaces such as SPI, QSPI, I2C, USART/UART and SDIO, CAN bus controller, external memory controller (XMC), USB2.0 full-speed interfaces, Ethernet MAC, parallel digital camera interface, HICK with automatic clock calibration (ACC), 12-bit ADC, 12-bit DAC, programmable voltage monitor (PVM) and other peripherals. Cortex®-M4F processer supports enhanced high-performance DSP instruction set, including extended single-cycle 16-bit/32-bit multiply accumulator (MAC), dual 16-bit MAC instructions, optimized 8-bit/16-bit SIMD operation and saturation operation instructions, and single-precision (IEEE-754) floating point unit (FPU), as shown in Figure 1-1:

Figure 1-1 AT32F435/437 Series microcontrollers system architecture



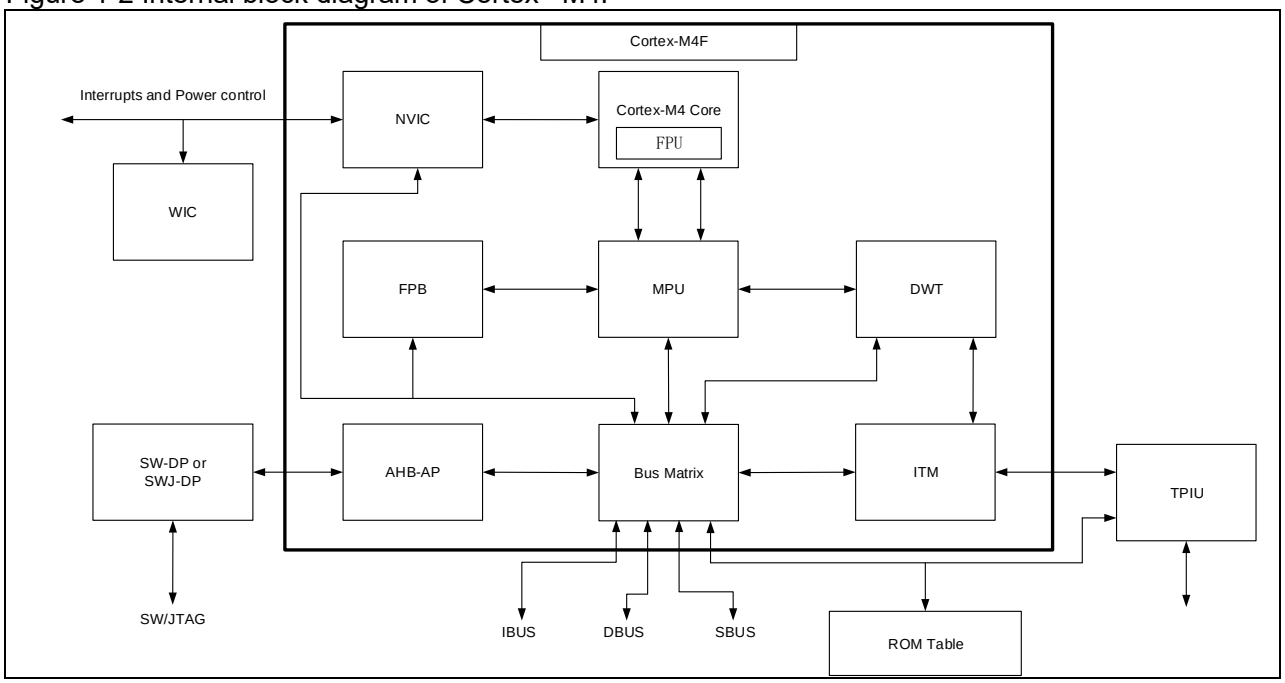
1.1 System overview

1.1.1 ARM Cortex™-M4F processor

Cortex®-M4F processor is a low-power consumption processor featuring low gate count, low interrupt latency, and low-cost debug. It supports DSP instruction set and FPU, and is applicable to deeply-embedded applications that require quicker response to interruption. Cortex®-M4F processor is based on ARMv7-M architecture, supporting both Thumb instruction set and DSP instruction set.

Figure 1-2 shows the internal block diagram of Cortex®-M4F processor. Please refer to *ARM Cortex® -M4 Technical Reference Manual* for more information.

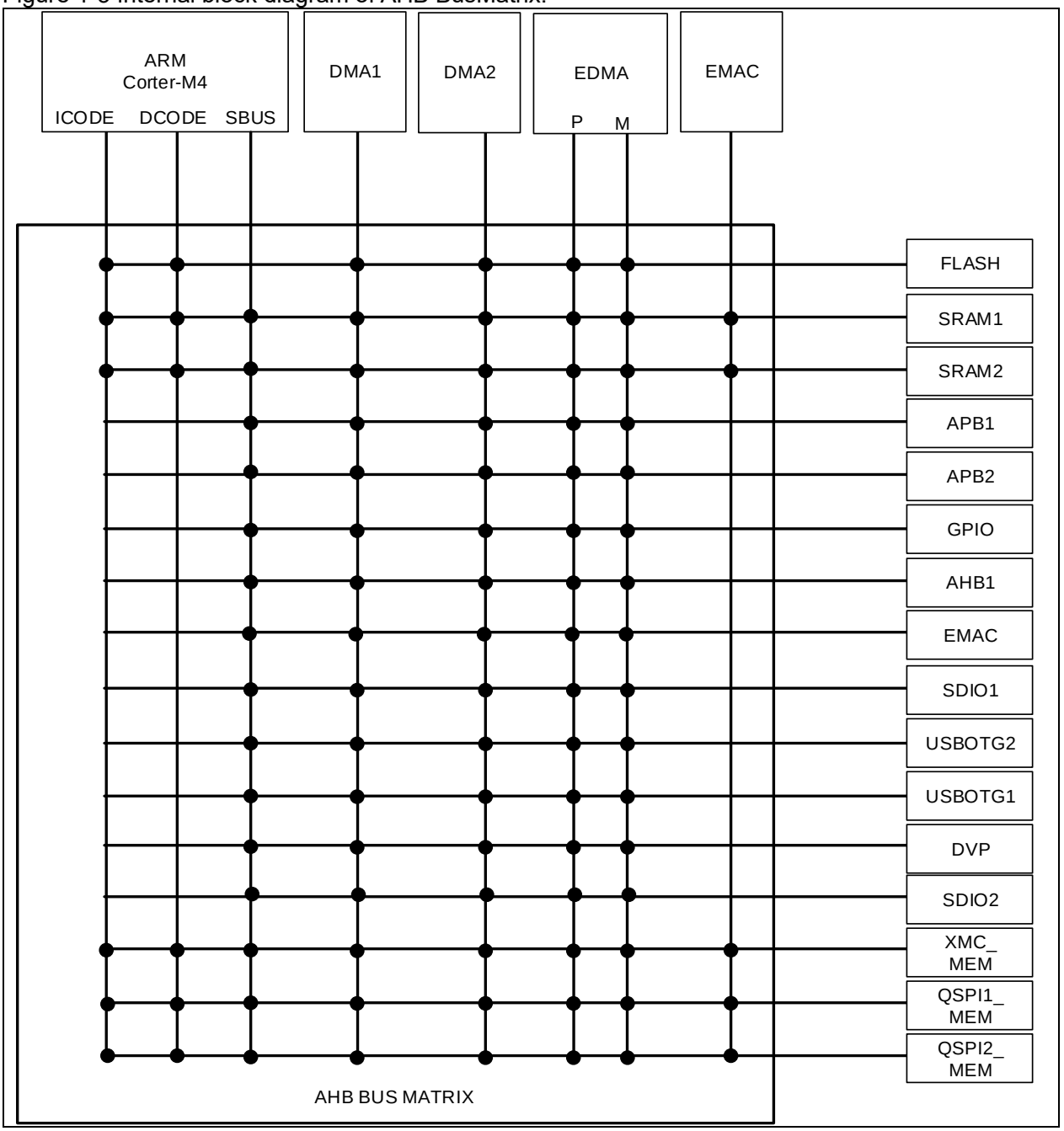
Figure 1-2 Internal block diagram of Cortex®-M4F



1.1.2 BusMatrix

Figure 1-3 shows the block diagram of AHB BusMatrix.

Figure 1-3 Internal block diagram of AHB BusMatrix.



1.1.3 Bit band

With the help of bit-band, read and write access to a single bit can be performed using common load/store operations. The Cortex®-M4F memory includes two bit-band regions: the least significant 1M bytes of SRAM and the least significant 1Mbytes of peripherals. In addition to access to bit-band addresses, their respective bit-band alias region can be used to access to any bit in these two bit-band regions. The bit-band alias region transforms each bit into a 32-bit word. Thus, accessing to a bit in an alias region has the same effect as read-modify-write operation on the corresponding bit in a bit-band region.

Figure 1-4 Comparison between bit-band region and its alias region: image A

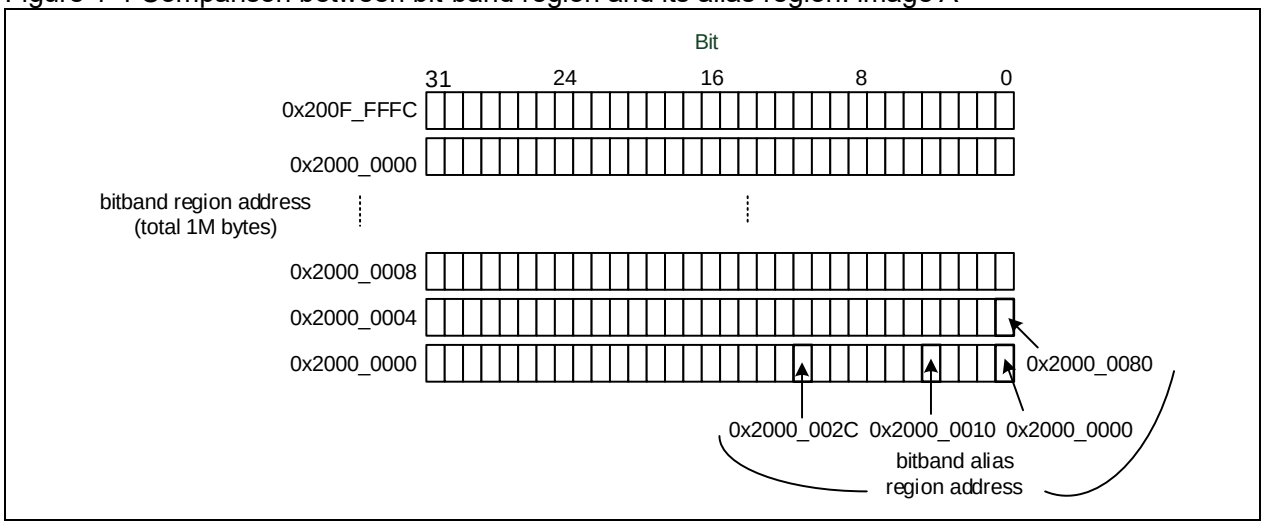
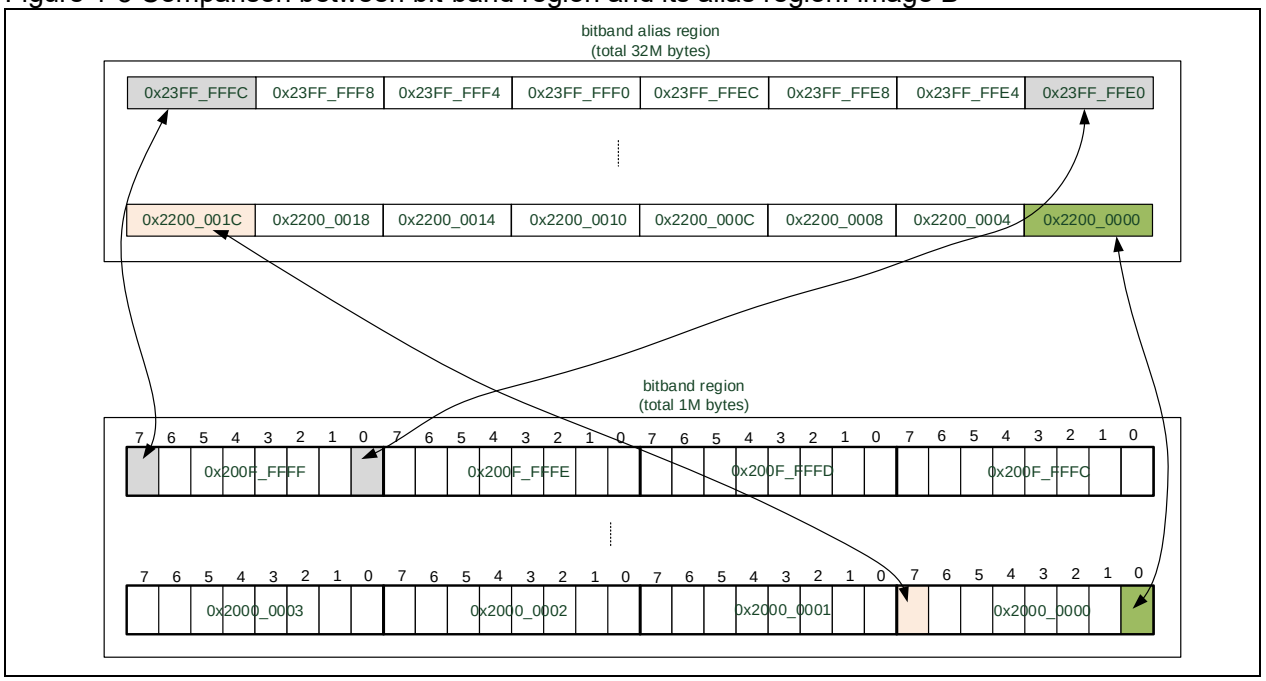


Figure 1-5 Comparison between bit-band region and its alias region: image B



Bit-band region: address region for bit-band operations

Bit-band alias region: access to the alias region has the same effect as read-modify-write operation on the bit-band region

Each bit in a bit-band region is mapped into a word (LSB) in an alias region. When accessing to the address in a bit-band alias region, such address is transformed into a bit-band address first. For a read operation, read one word in the bit-band region, and then move the targeted bit to the right to LSB before returning LSB. For a write operation, first move the targeted bit to the left to the corresponding bit number, then perform a read-modify-write operation on bit level.

The address ranges of two memories supporting bit-band operations:

The lowest 1 Mbyte of the SRAM: 0x2000_0000~0x200F_FFFF

The lowest 1 Mbyte of peripherals: 0x4000_0000~0x400F_FFFF

For a bit in the SRAM bit-band region, if the byte address is A, the bit number is n (0<=n<=7), then the alias address where the bit is:

AliasAddr = 0x2200_0000+ (A-0x2000_0000)*32+n*4

For a bit in the peripheral bit-band region, if the byte address is A, the bit number is n (0<=n<=7), then the alias address where the bit is:

AliasAddr = 0x4200_0000+ (A-0x4000_0000)*32+n*4

Table 1-1 shows the mapping between bit-band region and alias region in SRAM:

Table 1-1 Bit-band address mapping in SRAM

Bit-band region	Equivalent alias address
0x2000_0000.0	0x2200_0000.0
0x2000_0000.1	0x2200_0004.0
0x2000_0000.2	0x2200_0008.0
…	…
0x2000_0000.31	0x2200_007C.0
0x2000_0004.0	0x2200_0080.0
0x2000_0004.1	0x2200_0084.0
0x2000_0004.2	0x2200_0088.0
…	…
0x200F_FFFC.31	0x23FF_FFFC.0

Table 1-2 shows the mapping between bit-band region and alias region in the peripheral area:

Table 1-2 Bit-band address mapping in the peripheral area

Bit-band region	Equivalent alias address
0x4000_0000.0	0x4200_0000.0
0x4000_0000.1	0x4200_0004.0
0x4000_0000.2	0x4200_0008.0
…	…
0x4000_0000.31	0x4200_007C.0
0x4000_0004.0	0x4200_0080.0
0x4000_0004.1	0x4200_0084.0
0x4000_0004.2	0x4200_0088.0
…	…
0x400F_FFFC.31	0x43FF_FFFC.0

In terms of bit-band operation, one of the advantages is to control LED ON/OFF independently via GPIO pins. On the other hand, it brings great convenience for serial interface operations. In short, it is best suited to hardware I/O-intensive low-level applications.

In addition, bit-band operations can also simplify jump process. When jump operation is based on a bit level, the previous steps are:

- Read the whole register
- Mask the undesired bits
- Compare and jump

For now, you just need do:

- Read the bit status from the bit-band alias region
- Compare and jump

Apart from making code more concise, its important function is also reflected in multi-task environment. When it comes to multiple tasks, it turns the read-modify-write operations into a hardware-supported atomic operation to avoid the scenario where the read-modify-write operation is disrupted, resulting in disorder.

1.1.4 Interrupt and exception vectors

Table 1-3 AT32F435/437 series vector table

Pos.	Priority	Priority Type	Name	Description	Address
	-	-	-	Reserved	0x0000_0000
	-3	Fixed	Reset	Reset	0x0000_0004
	-2	Fixed	NMI	Non maskable interrupt CRM clock fail detector (CFD) is linked to NMI vector	0x0000_0008
	-1	Fixed	HardFault	All class of fault	0x0000_000C
	0	Configurable	MemoryManage	Memory management	0x0000_0010
	1	Configurable	BusFault	Pre-fetch fault, memory access fault	0x0000_0014
	2	Configurable	UsageFault	Undefined instruction or illegal state	0x0000_0018
	-	-	-	Reserved	0x0000_001C~ 0x0000_002B
	3	Configurable	SVCall	System service call via SWI instruction	0x0000_002C
	4	Configurable	Debug Monitor	Debug monitor	0x0000_0030
	-	-	-	Reserved	0x0000_0034
	5	Configurable	PendSV	Pendable request for system service	0x0000_0038
	6	Configurable	SysTick	System tick timer	0x0000_003C
0	7	Configurable	WWDT	Window watchdog timer	0x0000_0040
1	8	Configurable	PVM	PVM from EXINT interrupt	0x0000_0044
2	9	Configurable	TAMPER	Tamper interrupt	0x0000_0048
3	10	Configurable	ERTC_WKUP	ERTC wakeup interrupt	0x0000_004C
4	11	Configurable	FLASH	Flash global interrupt	0x0000_0050
5	12	Configurable	CRM	Clock and Reset manage (CRM) interrupt	0x0000_0054
6	13	Configurable	EXINT0	EXINT line0 interrupt	0x0000_0058
7	14	Configurable	EXINT1	EXINT line1 interrupt	0x0000_005C
8	15	Configurable	EXINT2	EXINT line2 interrupt	0x0000_0060
9	16	Configurable	EXINT3	EXINT line3 interrupt	0x0000_0064
10	17	Configurable	EXINT4	EXINT line4 interrupt	0x0000_0068
11	18	Configurable	EDMA data flow1	EDMA data flow1 global interrupt	0x0000_006C
12	19	Configurable	EDMA data flow2	EDMA data flow2 global interrupt	0x0000_0070
13	20	Configurable	EDMA data flow3	EDMA data flow3 global interrupt	0x0000_0074
14	21	Configurable	EDMA data flow4	EDMA data flow4 global interrupt	0x0000_0078

15	22	Configurable	EDMA data flow5	EDMA data flow5 global interrupt	0x0000_007C
16	23	Configurable	EDMA data flow6	EDMA data flow6 global interrupt	0x0000_0080
17	24	Configurable	EDMA data flow7	EDMA data flow7 global interrupt	0x0000_0084
18	25	Configurable	ADC1_2_3	ADC1, ADC2 and ADC3 global interrupt	0x0000_0088
19	26	Configurable	CAN1_TX	CAN1 TX interrupt	0x0000_008C
20	27	Configurable	CAN1_RX0	CAN1 RX0 interrupt	0x0000_0090
21	28	Configurable	CAN1_RX1	CAN1 RX1 interrupt	0x0000_0094
22	29	Configurable	CAN_SE	CAN state error interrupt	0x0000_0098
23	30	Configurable	EXINT9_5	EXINT line[9: 5] interrupt	0x0000_009C
24	31	Configurable	TMR1_BRK_TMR9	TMR1 break interrupt and TMR9 global interrupt	0x0000_00A0
25	32	Configurable	TMR1_OVF_TMR10	TMR1 overflow and TMR10 global interrupt	0x0000_00A4
26	33	Configurable	TMR1_TRG_HALL_TMR11	TMR1 trigger and HALL interrupt and TMR11 global interrupt	0x0000_00A8
27	34	Configurable	TMR1_CH	TMR1 channel interrupt	0x0000_00AC
28	35	Configurable	TMR2	TMR2 global interrupt	0x0000_00B0
29	36	Configurable	TMR3	TMR3 global interrupt	0x0000_00B4
30	37	Configurable	TMR4	TMR4 global interrupt	0x0000_00B8
31	38	Configurable	I2C1_EVT	I^2C1 event interrupt	0x0000_00BC
32	39	Configurable	I2C1_ERR	I^2C1 error interrupt	0x0000_00C0
33	40	Configurable	I2C2_EVT	I^2C2 event interrupt	0x0000_00C4
34	41	Configurable	I2C2_ERR	I^2C2 error interrupt	0x0000_00C8
35	42	Configurable	SPI1	SPI1 global interrupt	0x0000_00CC
36	43	Configurable	SPI2_I2S2EXT	SPI2 and IS2EXT global interrupt	0x0000_00D0
37	44	Configurable	USART1	USART1 global interrupt	0x0000_00D4
38	45	Configurable	USART2	USART2 global interrupt	0x0000_00D8
39	46	Configurable	USART3	USART3 global interrupt	0x0000_00DC
40	47	Configurable	EXINT15_10	EXINT line[15: 10] global interrupt	0x0000_00E0
41	48	Configurable	ERTCAlarm	ERTC alarm through EXINT interrupt	0x0000_00E4
42	49	Configurable	OTGFS1 wakeup	OTGFS1 standby wakeup through EXINT interrupt	0x0000_00E8
43	50	Configurable	TMR8_BRK_TMR12	TMR8 break interrupt and TMR12 global interrupt	0x0000_00EC
44	51	Configurable	TMR8_OVF_TMR13	TMR8 overflow interrupt and TMR13 global interrupt	0x0000_00F0
45	52	Configurable	TMR8_TRG_HALL_TMR14	TMR8 trigger and HALL interrupt and TMR14 global interrupt	0x0000_00F4
46	53	Configurable	TMR8_CH	TMR8 channel interrupt	0x0000_00F8
47	54	Configurable	EDMA data flow8	EDMA data flow8 global interrupt	0x0000_00FC
48	55	Configurable	XMC	XMC global interrupt	0x0000_0100

49	56	Configurable	SDIO1	SDIO1 global interrupt	0x0000_0104
50	57	Configurable	TMR5	TMR5 global interrupt	0x0000_0108
51	58	Configurable	SPI3_I2S3EXT	SPI3 and I2S3EXT global interrupt	0x0000_010C
52	59	Configurable	UART4	UART4 global interrupt	0x0000_0110
53	60	Configurable	UART5	UART5 global interrupt	0x0000_0114
54	61	Configurable	TMR6_DAC	TMR6 global interrupt DAC1 and DAC2 underflow error interrupt	0x0000_0118
55	62	Configurable	TMR7	TMR7 global interrupt	0x0000_011C
56	63	Configurable	DMA1 channel1	DMA1 channel1 global interrupt	0x0000_0120
57	64	Configurable	DMA1 channel2	DMA1 channel2 global interrupt	0x0000_0124
58	65	Configurable	DMA1 channel3	DMA1 channel3 global interrupt	0x0000_0128
59	66	Configurable	DMA1 channel4	DMA1 channel4 global interrupt	0x0000_012C
60	67	Configurable	DMA1 channel5	DMA1 channel5 global interrupt	0x0000_0130
61	68	Configurable	EMAC[2]	Ethernet global interrupt	0x0000_0134
62	69	Configurable	EMAC_WKUP[2]	Ethernet wakeup interrupt through EXINT	0x0000_0138
63	70	Configurable	CAN2_TX	CAN2 TX interrupt	0x0000_013C
64	71	Configurable	CAN2_RX0	CAN2 RX0 interrupt	0x0000_0140
65	72	Configurable	CAN2_RX1	CAN2 RX1 interrupt	0x0000_0144
66	73	Configurable	CAN2_SE	CAN2 status error interrupt	0x0000_0148
67	74	Configurable	OTGFS1	OTGFS1 CAN2 status error interrupt	0x0000_014C
68	75	Configurable	DMA1 channel6	DMA1 channel6 global interrupt	0x0000_0150
69	76	Configurable	DMA1 chaanel7	DMA1 channel7 global interrupt	0x0000_0154
70	77	Configurable	-	-	0x0000_0158
71	78	Configurable	USART6	USART6 global interrupt	0x0000_015C
72	79	Configurable	I2C3_EVT	I2C2 event interrupt	0x0000_0160
73	80	Configurable	I2C3_ERR	I2C2 error interrupt	0x0000_0164
74	81	Configurable	-	-	0x0000_0168
75	82	Configurable	-	-	0x0000_016C
76	83	Configurable	OTGFS2 wakeup	OTGFS2 standby wakeup interrupt through EXINT	0x0000_0170
77	84	Configurable	OTGFS2	OTGFS2 global interrupt	0x0000_0174
78	85	Configurable	DVP	DVP global interrupt	0x0000_0178
79	86	-	-	-	0x0000_017C
80	87	-	-	-	0x0000_0180
81	88	Configurable	FPU	FPU exception interrupt	0x0000_0184
82	89	Configurable	UART7	UART7 global interrupt	0x0000_0188
83	90	Configu	UART8	UART8 global interrupt	0x0000_018C

		rable			
84	91	Configurable	SPI4	SPI4 global interrupt	0x0000_0190
85	92	-	-	-	0x0000_0194
86	93	-	-	-	0x0000_0198
87	94	-	-	-	0x0000_019C
88	95	-	-	-	0x0000_01A0
89	96	-	-	-	0x0000_01A444
90	97	-	-	-	0x0000_01A8
91	98	Configurable	QSPI2	QSPI2 global interrupt	0x0000_01AC
92	99	Configurable	QSPI1	QSPI1 global interrupt	0x0000_01B0
93	100	-	-	-	0x0000_01B4
94	101	Configurable	DMAMUX	DMAMUX overflow interrupt	0x0000_01B8
95	102	-	-	-	0x0000_01BC
96	103	-	-	-	0x0000_01C0
97	104	-	-	-	0x0000_01C4
98	105	-	-	-	0x0000_01C8
99	106	-	-	-	0x0000_01CC
100	107	-	-	-	0x0000_01D0
101	108	-	-	-	0x0000_01D4
102	109	Configurable	SDIO2	SDIO2 global interrupt	0x0000_01D8
103	110	Configurable	ACC	ACC global interrupt	0x0000_01DC
104	111	Configurable	TMR20_BRK	TMR20 break interrupt	0x0000_01E0
105	112	Configurable	TMR20_OVF	TMR20 overflow interrupt	0x0000_01E4
106	113	Configurable	TMR20_TRG_HALL	TMR20 trigger and HALL interrupt	0x0000_01E8
107	114	Configurable	TMR20_CH	TMR20 channel interrupt	0x0000_01EC
108	115	Configurable	DMA2 channel0	DMA2 channel0 global interrupt	0x0000_01F0
109	116	Configurable	DMA2 channel1	DMA2 channel1 global interrupt	0x0000_01F4
110	117	Configurable	DMA2 channel2	DMA2 channel2 global interrupt	0x0000_01F8
111	118	Configurable	DMA2 channel3	DMA2 channel3 global interrupt	0x0000_01FC
112	119	Configurable	DMA2 channel4	DMA2 channel4 global interrupt	0x0000_0200
113	120	Configurable	DMA2 channel5	DMA2 channel5 global interrupt	0x0000_0204
114	121	Configurable	DMA2 channel6	DMA2 channel6global interrupt	0x0000_0208

Note: AT32F437 supports EMAC and EMAC_WKUP interrupts, but AT32F435 does not.

1.1.5 System Tick (SysTick)

The System Tick is a 24-bit downcounter. It will be reloaded with the initial value automatically when it is decremented to zero. It can generate periodic interrupts, so it is often used as multi-task scheduling counter for embedded operating system, and also to call the periodic tasks for non-embedded system.

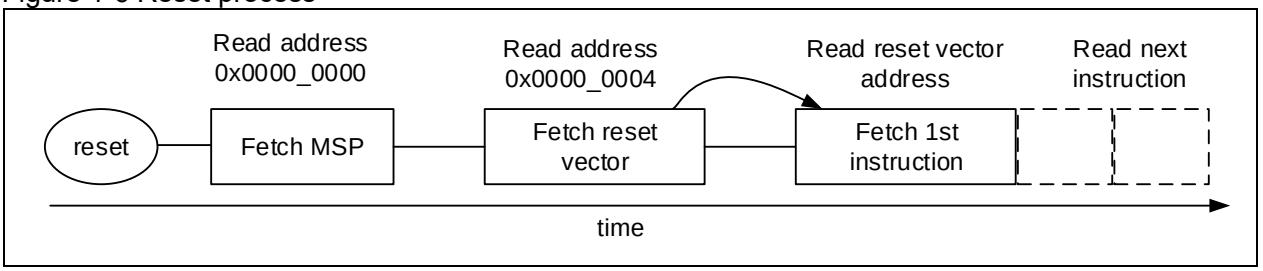
The System Tick calibration value is fixed to 9000, which gives a reference time base of 1 ms when the System Tick clock is set to 9 MHz.

1.1.6 Reset

The processor reads the first two words from the CODE memory after a system reset and before program execution.

- Get the initial value of the main stack pointer (MSP) from address 0x0000_0000
- Get the initial value of the program counter (PC) from address 0x0000_0004. This value is a reset vector and LSB must be 1. Then take the instructions from the address corresponding to this value.

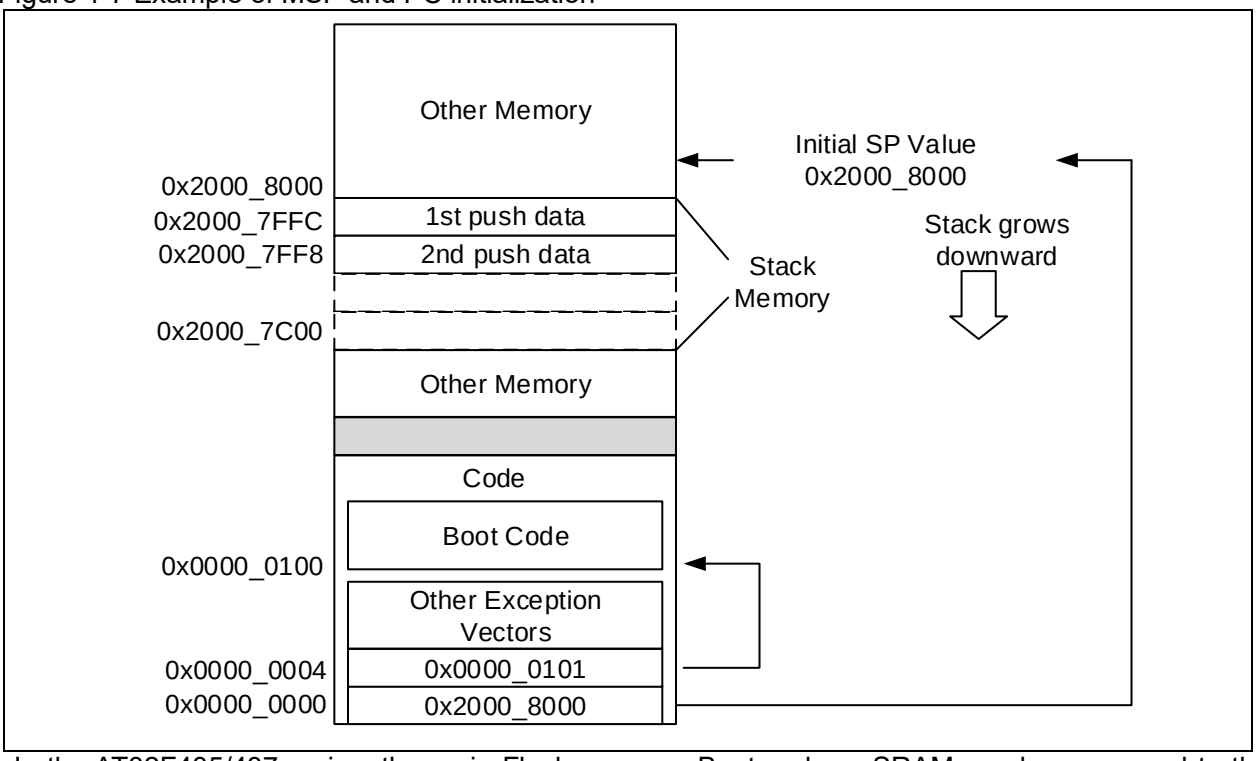
Figure 1-6 Reset process



Cortex™-M4F uses a full stack that increases downward, so the initial value of the main stack pointer (MSP) must be the end address of the stack memory plus 1. For example, if the stack area is set between 0x2000_7C00 and 0x2000_7FFF, then the initial value of MSP must be defined as 0x2000_8000.

The vector table follows the initial value of MSP. Cortex™-M4F operates in Thumb state, and thus each value in the vector table must set the LSB to 1. In Figure 1-6, 0x0000_0101 is used to represent 0x0000_0100. After the instruction at 0x0000_0100 is executed, the program starts running formally. Before that, it is a must for initializing MSP, because the first instruction may be interrupted by NMI or other faults before being executed. After the completion of MSP initialization, it is ready to prepare stack room for its service routines.

Figure 1-7 Example of MSP and PC initialization



In the AT32F435/437 series, the main Flash memory, Boot code or SRAM can be remapped to the code area between 0x0000_0000 and 0x07FF_FFFF. BOOT1 and BOOT0 are used to set the specific memory from which CODE starts.

{BOOT1, BOOT0}=00/10, CODE starts from the main Flash memory

{BOOT1, BOOT0}=01, CODE starts from Boot code

{BOOT1, BOOT0}=11, CODE starts from SRAM

After a system reset or when leaving from Standby mode, the pin values of both BOOT1 and BOOT0 will be relatched. If on-chip SRAM mode is used, the BOOT state is locked. In this case, it is impossible to select another boot mode, even after a system reset. A new boot mode can be selected only after the next power-on reset.

Boot code memory contains an embedded boot loader program that provides not only Flash programming function through USART1, USART2 or USB interface, but also provides extra firmware including communication protocol stacks that can be called for use by software developer through API.

1.2 List of abbreviations for registers

Table 1-4 List of abbreviations for registers

Register type	Description
rw	Software can read and write to this bit.
ro	Software can only read this bit.
wo	Software can only write to the bit. Reading it returns its reset value.
rrc	Software can read this bit. Reading this bit automatically clears it.
rw0c	Software can read this bit and clear it by writing 0. Writing 1 has no effect on this bit.
rw1c	Software can read this bit and clear it by writing 1. Writing 0 has no effect on this bit.
rw1s	Software can read this bit and set it by writing 1. Writing 0 has no effect on this bit.
tog	Software can read this bit and toggle it by writing 1. Writing 0 has no effect on this bit.
rwt	Software can read this bit. Writing any value will trigger an event.
resd	Reserved.

1.3 Device characteristics information

Table 1-5 List of abbreviations for registers

Register abbr.	Base address	Reset value
F_SIZE	0x1FFF F7E0	0xXXXX
UID[31: 0]	0x1FFF F7E8	0xXXXX XXXX
UID[63: 32]	0x1FFF F7EC	0xXXXX XXXX
UID[95: 64]	0x1FFF F7F0	0xXXXX XXXX

1.3.1 Flash memory size register

This register contains the information about Flash memory size.

Bit	Abbr.	Reset value	Type	Description
Bit 15: 0	F_SIZE	0xXXXX	ro	Flash size, in terms of KByte For example: 0x0080 = 128 KByte

1.3.2 Device electronic signature

The device electronic signature contains the memory size and the unique device ID (96 bits). It is stored in the information block of the Flash memory. The 96-bit ID is unique for any device, and cannot be altered by users. It can be used for the following:

- Serial number: such as USB string serial number
- Part of security keys

Bit	Abbr.	Reset value	Type	Description
Bit 31: 0	UID[31: 0]	0xXXXX XXXX	ro	UID for bit 31 to bit 0

Bit	Abbr.	Reset value	Type	Description
Bit 31: 0	UID[63: 32]	0xXXXX XXXX	ro	UID for bit 63 to bit 32

Bit	Abbr.	Reset value	Type	Description

Bit 31: 0	UID[95: 64]	0xXXXX XXXX	ro	UID for bit 95 to bit 64

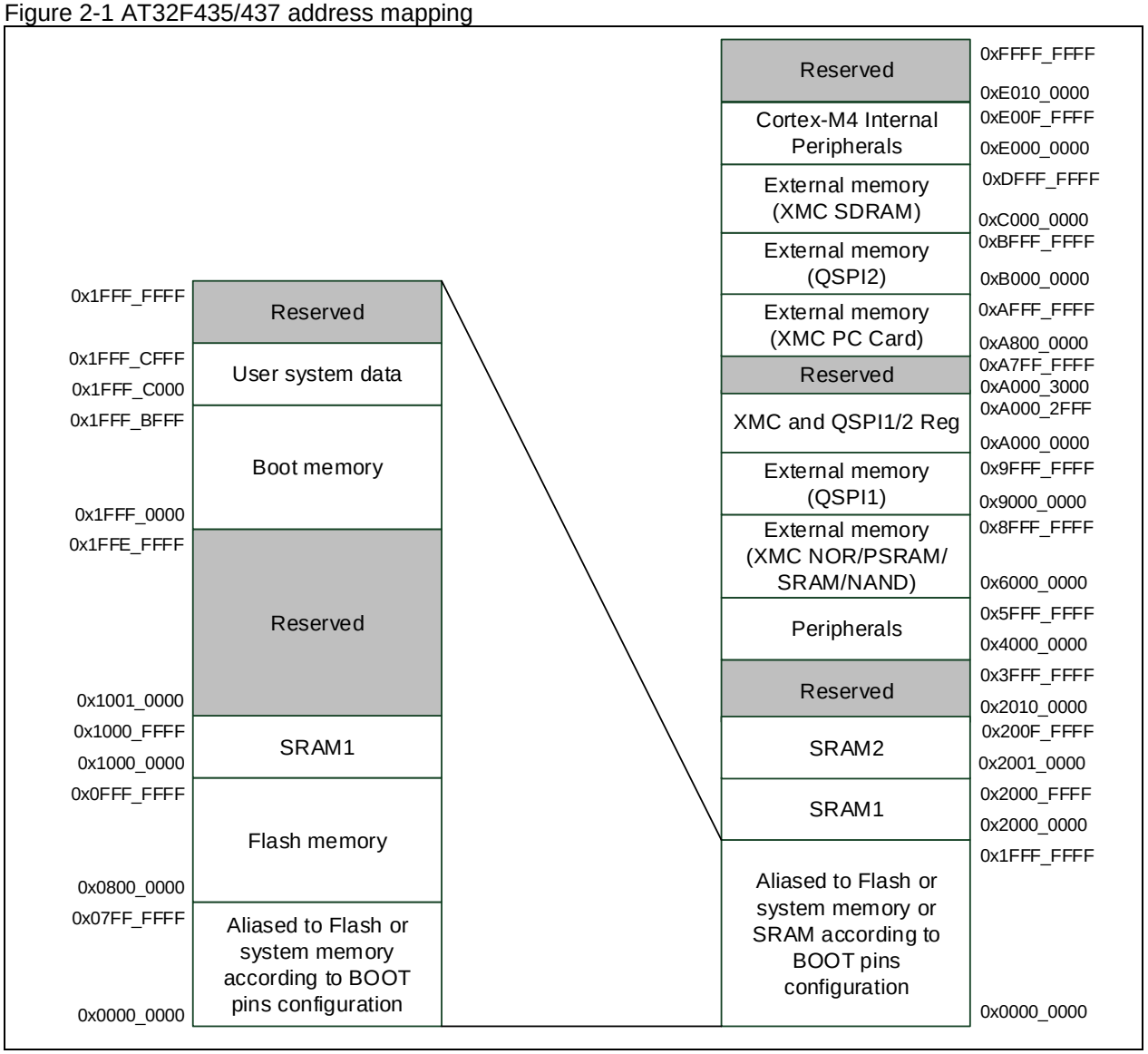
Note: UID[95:88] is series ID, which is 0x0D for AT32F435, and 0x0E for AT32F437.

2 Memory resources

2.1 Internal memory address map

Internal memory contains program memory (Flash), data memory (SRAM), peripheral registers and core registers. Their respective address mapping are shown in Figure 2-1.

Figure 2-1 AT32F435/437 address mapping



2.2 Flash memory

AT32F435/437 series provide up to 4032 KB of on-chip Flash memory, supporting a 0-wait state single cycle 32-bit read operation.

Refer to *Chapter 5* for more details about Flash memory controller and register configuration.

Flash memory organization (4032K)

The main memory is divided into bank 1 and bank 2:

Bank 1: 2048 Kbytes, including 32 blocks, 16 sectors per block, and 4 Kbytes per sector.

Bank 2: 1984 Kbytes, including 31 blocks, 16 sectors per block, and 4 Kbytes per sector.

User system data area is 4 Kbytes.

Bank		Name		Address range
Main memory	Bank 1 2048 KB	Block 0	Sector 0	0x0800 0000 - 0x0800 0FFF
			Sector 1	0x0800 1000 - 0x0800 1FFF
			Sector 2	0x0800 2000 - 0x0800 2FFF
			…	…
			Sector 15	0x0800 F000 - 0x0800 FFFF
		Block 1	Sector 16	0x0801 0000 - 0x0801 0FFF
			…	…
			Sector 31	0x0801 F000 - 0x0801 FFFF
		…	…	…
		Block 31	Sector 496	0x081F 0000 - 0x081F 0FFF
			…	…
			Sector 511	0x081F F000 - 0x081F FFFF
	Bank 2 1984 KB	Block 32	Sector 512	0x0820 0000 - 0x0820 0FFF
			…	…
			Sector 527	0x0820 F000 - 0x0820 FFFF
		…	…	…
		Block 62	Sector 992	0x083E 0000 - 0x083E 0FFF
			…	…
			Sector 1007	0x083E F000 - 0x083E FFFF
Information block		16 KB boot loader		0x1FFF 0000 - 0x1FFF 3FFF
		4 KB user system data		0x1FFF C000 - 0x1FFF CFFF

Flash memory organization (1024K)

The main memory features bank 1 and bank 2, with 512 Kbytes in each. Each bank is divided into 8 blocks, each of which includes 32 sectors and 2 Kbytes per sector.

User system data area is 512 bytes.

Bank		Name		Address range
Main memory	Bank 1 512 KB	Block 0	Sector 0	0x0800 0000 – 0x0800 07FF
			Sector 1	0x0800 0800 – 0x0800 0FFF
			Sector 2	0x0800 1000 – 0x0800 17FF
			…	…
			Sector 31	0x0800 F800 - 0x0800 FFFF
		Block 1	Sector 32	0x0801 0000 - 0x0801 07FF
			…	…
			Sector 63	0x0801 F800 - 0x0801 FFFF
		…	…	…
		Block 7	Sector 224	0x0807 0000 - 0x0807 07FF
			…	…
			Sector 255	0x0807 F800 – 0x0807 FFFF
	Bank 2 512 KB	Block 8	Sector 256	0x0808 0000 – 0x0808 07FF
			…	…
			Sector 287	0x0808 F800 - 0x0808 FFFF
		…	…	…
		Block 15	Sector 480	0x080F 0000 - 0x080F 07FF
			…	…
			Sector 511	0x080F F800 – 0x080F FFFF
Information block		16 KB boot loader		0x1FFF 0000 - 0x1FFF 3FFF
		512-byte user system data		0x1FFF C000 - 0x1FFF C1FF

Flash memory organization (448K)

Main Flash memory (448 KB) has only bank 1, which id divided into 7 blocks that include 16 sectors in each, and 4 K per sector. User system data area is 4 KB.

Bank		Name		Address range
Main memory	Bank1 448KB	Block 0	Sector 0	0x0800 0000 - 0x0800 0FFF
			Sector 1	0x0800 1000 - 0x0800 1FFF
			Sector 2	0x0800 2000 - 0x0800 2FFF
			…	…
			Sector 15	0x0800 F000 - 0x0800 FFFF
		Block 1	Sector 16	0x0801 0000 - 0x0801 0FFF
			…	…
			Sector 31	0x0801 F000 - 0x0801 FFFF
		…	…	…
		Block 6	Sector 96	0x0806 0000 - 0x0806 0FFF
			…	…
			Sector 111	0x0806 F000 - 0x0806 FFFF
Information block		16 KB boot memory		0x1FFF 0000 - 0x1FFF 3FFF
		4KB user system data		0x1FFF C000 - 0x1FFF CFFF

Flash memory organization (256K)

Main Flash memory (256 KB) has only bank 1, which is divided into 4 blocks that include 32 sectors in each, and 2 K per sector.

User system data area is 512 bytes.

Bank		Name		Address range
Main memory	Bank 1 512 KB	Block 0	Sector 0	0x0800 0000 – 0x0800 07FF
			Sector 1	0x0800 0800 – 0x0800 0FFF
			Sector 2	0x0800 1000 – 0x0800 17FF
			…	…
			Sector 31	0x0800 F800 - 0x0800 FFFF
		Block 1	Sector 32	0x0801 0000 - 0x0801 07FF
			…	…
			Sector 63	0x0801 F800 - 0x0801 FFFF
		Block 2	Sector 64	0x0802 0000 - 0x0802 07FF
			…	…
			Sector 95	0x0802 F800 - 0x0802 FFFF
		Block 7	Sector 96	0x0803 0000 - 0x0803 07FF
			…	…
			Sector 127	0x0803 F800 – 0x0803 FFFF
Information block		16 KB boot memory		0x1FFF 0000 - 0x1FFF 3FFF
		512-byte user system data		0x1FFF C000 - 0x1FFF C1FF

2.3 SRAM memory

The AT32F435/437 series contain a 384-KB on-chip SRAM which starts at the address 0x2000_0000. It can be accessed by bytes, half-words (16 bit) or words (32 bit). In addition, AT32F435/437 also provide a special mode that enables a dynamic switch between 128 KB (minimum) and 512 KB (maximum). This is done by setting the EOPB0 bit. In 512 KB mode, Flash memory size (zero wait sate) is limited to 128 KB, while in 128 KB extension mode, the zero-wait-state Flash size is limited to 512 KB. On-chip SRAM is divided into SRAM1 and SRAM2. The SRAM1 is fixed at 64 KB, and can be accessed through either address 0x2000_0000~0x2000_FFFF or 0x1000_0000~0x1000_FFFF. The SRAM2 size ranges from 64 KB to 448 KB, accessible through address 0x2001_0000~2007_FFF.

2.4 Peripheral address map

Table 2-1 Peripheral boundary address

Bus	Boundary address	Peripherals
AHB BusMatrix	0xC000 0000 - 0xDFFF FFFF	XMC_MEM(SDRAM)
	0xB000 0000 - 0xBFFF FFFF	QSPI2_MEM
	0xA800 0000 - 0xAFFF FFFF	XMC_MEM(PC CARD)
	0x9000 0000 - 0x9FFF FFFF	QSPI1_MEM
	0x6000 0000 - 0x8FFF FFFF	XMC_MEM
	0x5006 1000 - 0x5006 13FF	SDIO2
	0x5005 0000 - 0x5005 03FF	DVP
	0x5000 0000 - 0x5003 FFFF	OTG_FS1
	0x4004 0000 - 0x4007 FFFF	OTG_FS2
	0x4002 C400 - 0x4002 C7FF	SDIO1
	0x4002 8000 - 0x4002 9FFF	EMAC
	0x4002 1C00 - 0x4002 1FFF	GPIO port H

	0x4002 1800 - 0x4002 1BFF	GPIO port G
	0x4002 1400 - 0x4002 17FF	GPIO port F
	0x4002 1000 - 0x4002 13FF	GPIO port E
	0x4002 0C00 - 0x4002 0FFF	GPIO port D
	0x4002 0800 - 0x4002 0BFF	GPIO port C
	0x4002 0400 - 0x4002 07FF	GPIO port B
	0x4002 0000 - 0x4002 03FF	GPIO port A
AHB1	0xA000 2000 - 0xA000 2FFF	QSPI2_REG
	0xA000 1000 - 0xA000 1FFF	QSPI1_REG
	0xA000 0000 - 0xA000 0FFF	XMC_REG
	0x4002 6400 - 0x4002 67FF	DMA1_2
	0x4002 6000 - 0x4002 63FF	EDMA
	0x4002 4000 - 0x4002 5FFF	Reserved
	0x4002 3C00 - 0x4002 3FFF	Flash memory interface (FLASH)
	0x4002 3800 - 0x4002 3BFF	Clock and reset manage (CRM)
	0x4002 3400 - 0x4002 37FF	Reserved
	0x4002 3000 - 0x4002 33FF	CRC
APB2	0x4001 8000 - 0x4001 FFFF	Reserved
	0x4001 7C00 - 0x4001 7FFF	I2S3EXT
	0x4001 7800 - 0x4001 7BFF	I2S2EXT
	0x4001 7400 - 0x4001 77FF	ACC
	0x4001 4C00 - 0x4001 73FF	TMR20 timer
	0x4001 4800 - 0x4001 4BFF	TMR11 timer
	0x4001 4400 - 0x4001 47FF	TMR10 timer
	0x4001 4000 - 0x4001 43FF	TMR9 timer
	0x4001 3C00 - 0x4001 3FFF	EXINT
	0x4001 3800 - 0x4001 3BFF	SCFG
	0x4001 3400 - 0x4001 37FF	SPI4/I2S4
	0x4001 3000 - 0x4001 33FF	SPI1/I2S1
	0x4001 2400 - 0x4001 2FFF	Reserved
	0x4001 2000 - 0x4001 23FF	ADC1/2/3
	0x4001 1800 - 0x4001 1FFF	Reserved
	0x4001 1400 - 0x4001 17FF	USART6
	0x4001 1000 - 0x4001 13FF	USART1
	0x4001 0800 - 0x4001 0FFF	Reserved
	0x4001 0400 - 0x4001 07FF	TMR8 timer
	0x4001 0000 - 0x4001 03FF	TMR1 timer
APB1	0x4000 8000 - 0x4000 FFFF	Reserved
	0x4000 7C00 - 0x4000 7FFF	UART8
	0x4000 7800 - 0x4000 7BFF	UART7
	0x4000 7400 - 0x4000 77FF	DAC

0x4000 7000 - 0x4000 73FF	Power control (PWC)
0x4000 6C00 - 0x4000 6FFF	Reserved
0x4000 6800 - 0x4000 6BFF	CAN2
0x4000 6400 - 0x4000 67FF	CAN1
0x4000 6000 - 0x4000 63FF	Reserved
0x4000 5C00 - 0x4000 5FFF	I2C3
0x4000 5800 - 0x4000 5BFF	I2C2
0x4000 5400 - 0x4000 57FF	I2C1
0x4000 5000 - 0x4000 53FF	UART5
0x4000 4C00 - 0x4000 4FFF	UART4
0x4000 4800 - 0x4000 4BFF	USART3
0x4000 4400 - 0x4000 47FF	USART2
0x4000 4000 - 0x4000 43FF	Reserved
0x4000 3C00 - 0x4000 3FFF	SPI3/I2S3
0x4000 3800 - 0x4000 3BFF	SPI2/I2S2
0x4000 3400 - 0x4000 37FF	Reserved
0x4000 3000 - 0x4000 33FF	Watchdog timer (WDT)
0x4000 2C00 - 0x4000 2FFF	Window watchdog timer (WWDT)
0x4000 2800 - 0x4000 2BFF	ERTC
0x4000 2400 - 0x4000 27FF	Reserved
0x4000 2000 - 0x4000 23FF	TMR14 timer
0x4000 1C00 - 0x4000 1FFF	TMR13 timer
0x4000 1800 - 0x4000 1BFF	TMR12 timer
0x4000 1400 - 0x4000 17FF	TMR7 timer
0x4000 1000 - 0x4000 13FF	TMR6 timer
0x4000 0C00 - 0x4000 0FFF	TMR5 timer
0x4000 0800 - 0x4000 0BFF	TMR4 timer
0x4000 0400 - 0x4000 07FF	TMR3 timer
0x4000 0000 - 0x4000 03FF	TMR2 timer

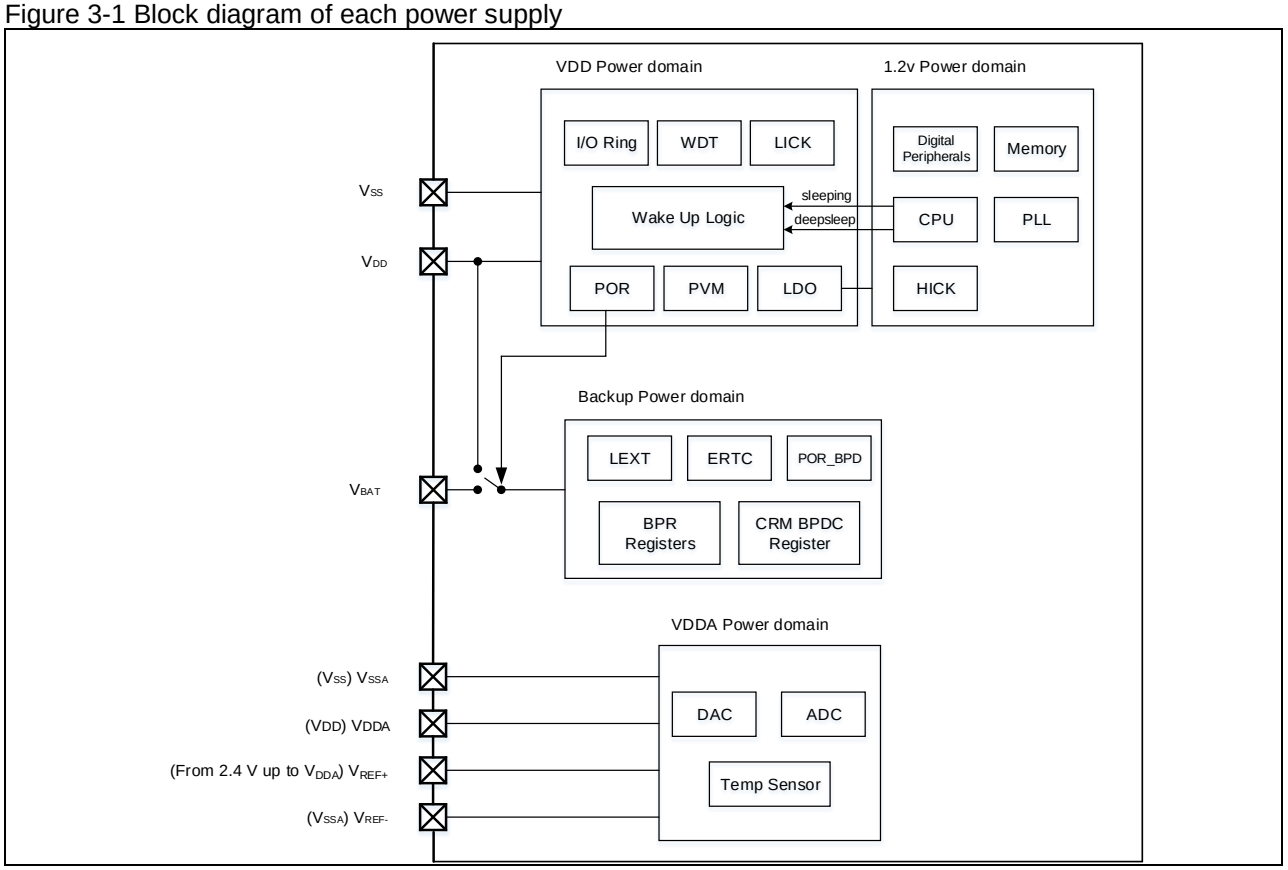
Note: Only AT32F437 supports EMAC module.

3 Power control (PWC)

3.1 Introduction

For AT32F435/437 series, its operating voltage supply is 2.6 V ~ 3.6 V, with a temperature range of -40~+105 °C. To reduce power consumption, this series provides three types of power saving modes, including Sleep, Deepsleep and Standby modes so as to achieve the best tradeoff among the conflicting demands of CPU operating time, speed and power consumption. The AT32F435/437 series have three power domains—VDD/VDDA domain, 1.2 V domain and battery powered domain. The VDD/VDDA domain is supplied directly by external power, the 1.2 V domain is powered by an embedded LDO in the VDD/VDDA domain, and the battery powered domain is supplied through a V_{BAT} pin.

Figure 3-1 Block diagram of each power supply



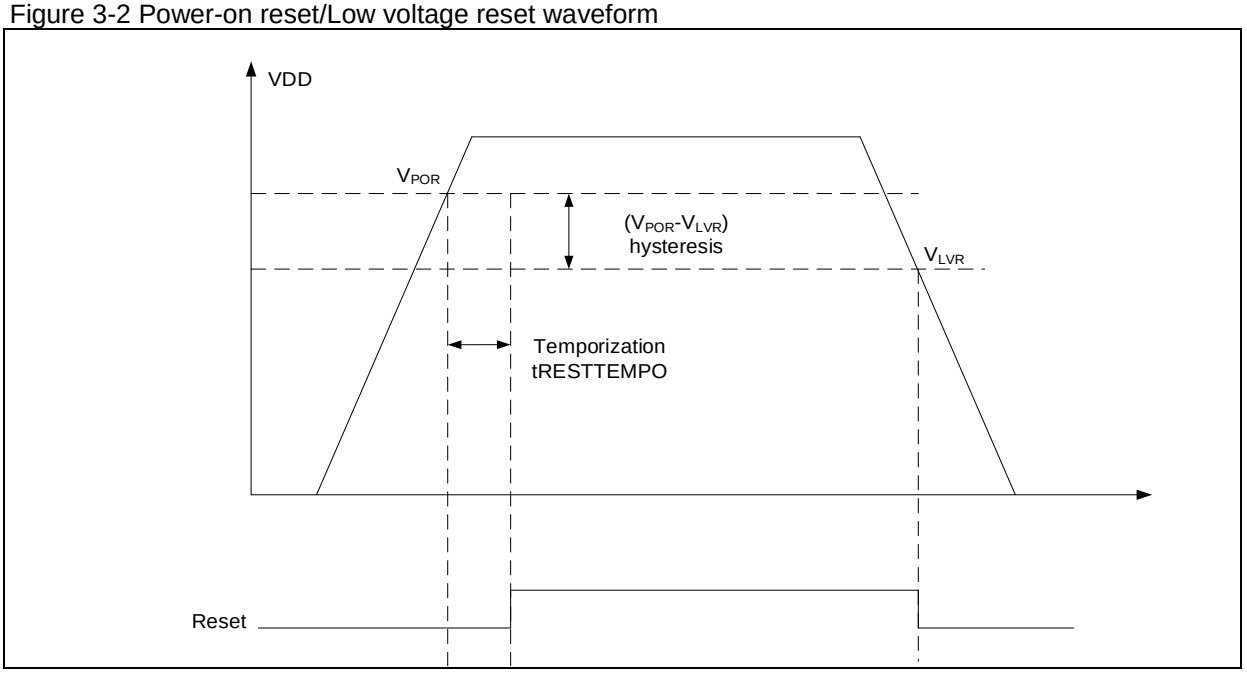
3.2 Main Features

- Three power domains: VDD/VDDA domain, 1.2 V domain and battery powered domain
- Three types of power saving modes: Sleep mode, Deepsleep mode, and Standby mode
- Internal voltage regulator supplies 1.2 V voltage source for the core domain
- Power voltage detector is provided to issue an interrupt when the supply voltage is lower or higher than a programmed threshold
- The battery powered domain is powered by V_{BAT} when VDD is powered off
- VDD/VDDA applies separated digital and analog module to reduce noise on external power

3.3 POR/LVR

A POR analog module embedded in the VDD/VDDA domain is used to generate a power reset. The power reset signal is released at V_{POR} when the VDD is increased from 0 V to the operating voltage, or it is triggered at V_{LVR} when the VDD drops from the operating voltage to 0 V. During the power-on reset period, the reset signal has certain amount of time delay compared to VDD boost process. At the same time, hysteresis occurs in power-on reset (POR) and low voltage reset (LVR).

Figure 3-2 Power-on reset/Low voltage reset waveform

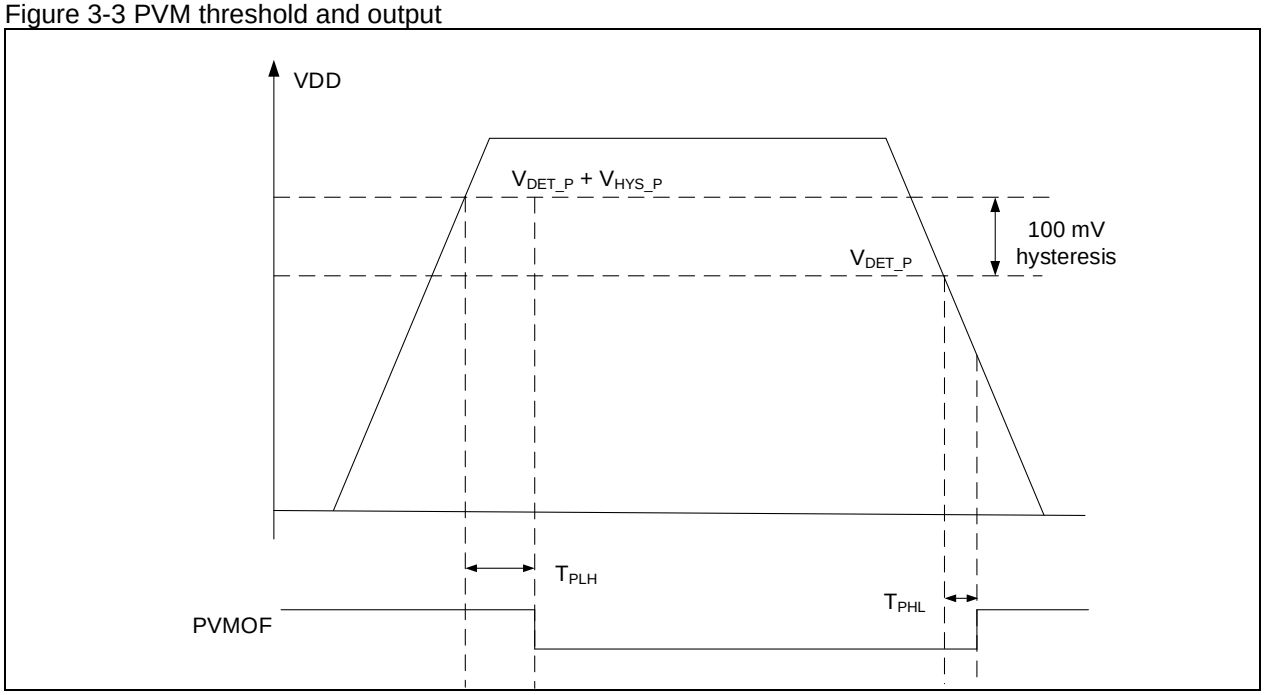


3.4 Power voltage monitor (PVM)

The PVM is used to monitor the power supply variations. It is enabled by setting the PVMEN bit in the power control register (PWC_CTRL), and the threshold value for voltage monitor is selected with the PVMSEL[2: 0].

After PVM is enabled, the comparison result between VDD and the programmed threshold is indicated by the PVMOF bit in the PWC_CTRLSTS register, with the hysteresis voltage VHYS_P being 100 mv. The PVM interrupt will be generated through the EXTI line 16 when VDD rises above the PVM threshold.

Figure 3-3 PVM threshold and output



3.5 Power domain

1.2 V domain

1.2 V core domain includes a CPU core, SRAM, embedded digital peripherals and Phase Locked Loop (PLL). Such power domain is supplied by LDO (voltage regulator).

VDD/VDDA domain

VDD/VDDA domain includes VDD domain and VDDA domain. The VDD domain contains I/O circuit, power-saving mode wakeup circuit, watchdog timer (WDT), power-on reset/low voltage reset (POR/LVR), LDO and all PAD circuits other than PC13, PC14 and PC15. The VDDA domain contains DAC/ADC (DA/AD converters), temperature sensor and so on.

Typically, to ensure a better accuracy of ADC/DAC at a low voltage, the digital circuit is supplied by VDD while the analog circuit is powered by VDDA. On 64-pin packages and packages with less pins, the external reference voltage VREF+ and VREF- are connected to the VDDA pin and VSSA pin, respectively.

In Run mode, the LDO supplies full power to the 1.2 V core domain and outputs 1.2 V voltage, by default. The LDO output voltage is selected through the PWC_LDOOV register. The maximum operating frequency for the system depends on the selected output voltage. Refer to AT32F435/437 datasheets for details. The LDO output voltage is changeable only when the HEXT or HICK is used as system clock.

LDO output voltage regulation

1) Select HICK or HEXT as system clock
2) Change LDO voltage (LDOOVSEL[2: 0])
3) Set the FLASH_DIVR register and NZW_BST bit in the FLASH_PSR register
4) Set the targeted frequency for PLL-related registers, enable PLL, and wait for PLL_STBL
5) Set pre-division factors for AHB and APB
6) Enable auto step-by-step frequency switch function when the PLL frequency is greater than 108 MHz
7) Switch the system clock to PLL

Battery powered domain

The battery powered domain contains ERTC circuit, LEXT oscillator, PC13, PC14 and PC15, which is powered by either VDD or VBAT pin. When the VDD is cut off, the battery powered domain is automatically switched to VBAT pin to ensure that ERTC can work normally.

1) When the battery powered domain is powered by VDD, the PC13 can be used as a general-purpose I/O, tamper pin, ERTC calibration clock, ERTC alarm or second output, while the PC14 and PC15 can be used as a GPIO or LEXT pin. (As an I/O port, PC13, PC14 and PC15 must be limited below 2 MHz, and to the maximum load of 30 pF, and these I/O ports must not be used as current sources)
2) When the battery powered domain is powered by VBAT, the PC13 can be used as a tamper pin, ERTC alarm or second output, while the PC14 and PC15 can only be used as a LEXT pin.

The switch of the battery powered domain will not be disconnected from VBAT because of the VDD being at its rising phrase or due to VDD low voltage reset. If the power switch has not been switched to the VDD when the VDD is powered on quickly, it is recommended to add a low voltage drop diode between VDD and VBAT in order to prevent the currents of VDD from being injected to VBAT. If there is no external battery in the application, it is better to connect the VBAT to a 100 nF ceramic filter capacitor that is externally connected to VDD.

3.6 Power saving modes

When the CPU does not need to be kept running, there are three types of low-power modes available (Sleep mode, Deepsleep mode and Standby mode) to save power. Users can select the mode that gives the best compromise according to the low-power consumption, short startup time, and available wakeup sources. In addition, the power consumption in Run mode can be reduced by slowing down the system clocks or gating the clocks to the APB and AHB peripherals when they are not used.

Sleep mode

The Sleep mode is entered by executing WFI or WFE instruction. There are two options to select the Sleep mode entry mechanism through the SLEEPONEXIT bit in the Cortex ® -M4F system control register.

SLEEP-NOW mode:

When SLEEPDEEP=0 and SLEEPONEXIT=0, the MCU enters Sleep mode as soon as WFI or WFE instruction is executed.

When SLEEPDEEP=0 and SLEEPONEXIT=1, by executing the WFI instruction, the MCU enters Sleep

mode as soon as the system exits the lowest-priority interrupt service routine.

In Sleep mode, all clocks and LDO work normally except CPU clocks (stopped), and all I/O pins keep the same state as in Run mode. The LDO provides a 1.2 V power (for CPU core, memory and embedded peripherals) as it is in normal power consumption mode. The LDO output voltage is configurable by the PWC_LDOOV register.

1) If the WFI is executed to enter Sleep mode, any peripheral interrupt can wake up the device from Sleep mode.
2) If the WFE is executed to enter Sleep mode, the MCU exits Sleep mode as soon as an event occurs. The wakeup event can be generated by the following:

- Enabling a peripheral interrupt (it is not enabled in the NVIC) and enabling the SEVONPEND bit.

When the MCU resumes, the peripheral interrupt pending bit and NVIC channel pending bit must be cleared.

- Configuring an internal EXINT line as an event mode to generate a wakeup event.
- The wakeup time required by a WFE instruction is the shortest, since no time is wasted on interrupt entry/exit.

Deepsleep Mode

Deepsleep mode is entered by setting the SLEEPDEEP bit in the Cortex™-M4F system control register and clearing the LPSEL bit in the power control register before WFI or WFE instructions.

The LDO status is selected by setting the VRSEL bit in the power control register (PWC_CTRL). When VRSEL=0, the LDO works in normal mode. When VRSEL=1, the LDO is set in low-power consumption mode.

In Deepsleep mode, all clocks in 1.2 V domain are stopped, and both HICK and HEXT oscillators are disabled. The LDO supplies power to the 1.2 V domain in normal mode or low-power mode. All I/O pins keep the same state as in Run mode. SRAM and register contents are preserved.

1) When the Sleep mode is entered by executing a WFI instruction, the interrupt generated on any external interrupt line in Interrupt mode can wake up the system from Deepsleep mode.
2) When the Sleep mode is entered by executing a WFE instruction, the interrupt generated on any external interrupt line in Event mode can wake up the system from Deepsleep mode.

When the MCU exits the Deepsleep mode, the HICK RC oscillator is enabled and selected as a system clock after stabilization. When the LDO operates in low-power mode, an additional wakeup delay is incurred for the reason that the LDO must be stabilized before the system is waken from the Deepsleep mode.

Low-power Deepsleep LDO voltage regulation process (note that Sleep and Standby modes have no limits)

1) Select HICK as system clock
2) Change LDO voltage to 1.0 V by setting the LDOOVSEL[2: 0] bit
3) Set the FLASH_DIVR register and NZW_BST bit in the FLASH_PSR register
4) Set the VRSEL bit to enable low-power mode for the LDO
5) System enters Deepsleep state
6) System exits Deepsleep state (If wakeup conditions are met)
7) Change LDO voltage by setting the LDOOVSEL[2: 0]
8) If the HEXT is used as PLL clock, enable HEXT and wait for HEXTSTBL
9) Set the targeted frequency for PLL-related registers
10) Enable PLL and wait for PLL_STBL
11) Set pre-division factors for AHB and APB
12) Enable auto step-by-step frequency switch function when the PLL frequency is greater than 108 MHz
13) Switch the system clock to the PLL

Note: If the clock, after low-power mode is waken up, needs to keep the same state as in low-power mode, the above-mentioned steps 3/9/11 can be ignored.

Standby Mode

Standby mode can achieve the lowest power consumption for the device. In this mode, the LDO is disabled. The whole 1.2 V domain, PLL, HICK and HEXT oscillators are also powered off. SRAM and register contents are lost. Only registers in the battery powered domain and standby circuitry remain supplied.

The Standby mode is entered by the following procedures:

- Set the SLEEPDEE bit in the Cortex™-M4F system control register
- Set the LPSEL bit in the power control register (PWC_CTRL)
- Clear the SWEF bit in the power control/status register (PWC_CTRLSTS)
- Execute a WFI/WFE instruction

In Standby mode, all I/O pins remain in a high-impedance state except reset pins, TAMPER pins that are set as anti-tamper or calibration output, and the wakeup pins enabled.

The MCU leaves the Standby mode when an external reset (NRST pin), a WDT reset, a rising edge on the WKUP pin or the rising edge of an RTC alarm even occurs.

Debug mode

By default, the debug connection is lost if the MCU enters Deepsleep mode or Standby mode while debugging. The reason is that the Cortex™-M4F core is no longer clocked. However, the software can be debugged even in the low-power mode by setting some configuration bits in the DEBUG register (DEBUG_CTRL).

3.7 PWC registers

The peripheral registers must be accessed by words (32 bits).

Table 3-1 PW register map and reset values

Register abbr.	Offset	Reset value
PWC_CTRL	0x00	0x0000 0000
PWC_CTRLSTS	0x04	0x0000 0000
PWC_LDOOV	0x10	0x000X 0X00

3.7.1 Power control register (PWC_CTRL)

Bit	Name	Reset value	Type	Description
Bit 31: 9	Reserved	0x000000	resd	Kept at its default value.
Bit 8	BPWEN	0x0	rw	Battery powered domain write enable 0: Disabled 1: Enabled Note: After reset, the battery powered domain write access is disabled. To write, this bit must be set.
Bit 7: 5	PVMSEL	0x0	rw	Power voltage monitoring boundary select 000: Unused, not configurable 001: 2.3 V 010: 2.4 V 011: 2.5 V 100: 2.6 V 101: 2.7 V 110: 2.8 V 111: 2.9 V
Bit 4	PVMEN	0x0	rw	Power voltage monitoring enable 0: Disabled 1: Enabled
Bit 3	CLSEF	0x0	wo	Clear SEF flag 0: No effect 1: Clear the SEF flag Note: This bit is cleared by hardware after clearing the SEF flag. Reading this bit at any time will return all zero.
Bit 2	CLSWEF	0x0	wo	Clear SWEF flag 0: No effect 1: Clear the SWEF flag Note: Clear the SWEF flag after two system clock cycles. This bit is cleared by hardware after clearing the SWEF flag. Reading this bit at any time will return all zero.
Bit 1	LPSEL	0x0	rw	Low power mode select when Cortex™-M4F sleepdeep 0: Enter DEEPSLEEP mode 1: Enter Standby mode
Bit 0	VRSEL	0x0	rw	LDO state select in Deepsleep mode 0: Enabled 1: Low-power consumption mode

3.7.2 Power control/status register (PWC_CTRLSTS)

Additional APB cycles are needed to read this register versus a standard APB read.

Bit	Name	Reset value	Type	Description
Bit 31: 10	Reserved	0x000000	resd	Kept at its default value.
Bit 9	SWPEN2	0x0	rw	Standby wake-up pin2 enable 0: Disabled (this pin is used for general-purpose I/O) 1: Enabled (this pin is forced in input pull-down mode, and no longer used for general-purpose I/O) Note: This bit is cleared by hardware after system reset.
Bit 8	SWPEN1	0x0	rw	Standby wake-up pin1 enable 0: Disabled (this pin is used for general-purpose I/O) 1: Enabled (this pin is forced in input pull-down mode, and no longer used for general-purpose I/O) Note: This bit is cleared by hardware after system reset.
Bit 7: 3	Reserved	0x00	resd	Kept at its default value.
Bit 2	PVMOF	0x0	ro	Power voltage monitoring output flag 0: Power voltage is higher than the threshold 1: Power voltage is lower than the threshold Note: The power voltage monitor is stopped in Standby mode.
Bit 1	SEF	0x0	ro	Standby mode entry flag 0: Device is not in Standby mode

				1: Device is in Standby mode Note: This bit is set by hardware (enter Standby mode) and cleared by POR/LVR or by setting the CLSEF bit.
Bit 0	SWEF	0x0	ro	Standby wake-up event flag 0: No wakeup event occurred 1: A wakeup event occurred Note: This bit is set by hardware (on a wakeup event), and cleared by POR/LVR or by setting the CLSWEF bit. A wakeup event is generated by one of the following: When the rising edge on the Standby wakeup pin occurs; When the ERTC alarm event occurs; If the Standby wakeup pin is enabled when the Standby wakeup pin level is high.

3.7.3 LDO output voltage select register (PWC_LDOOV)

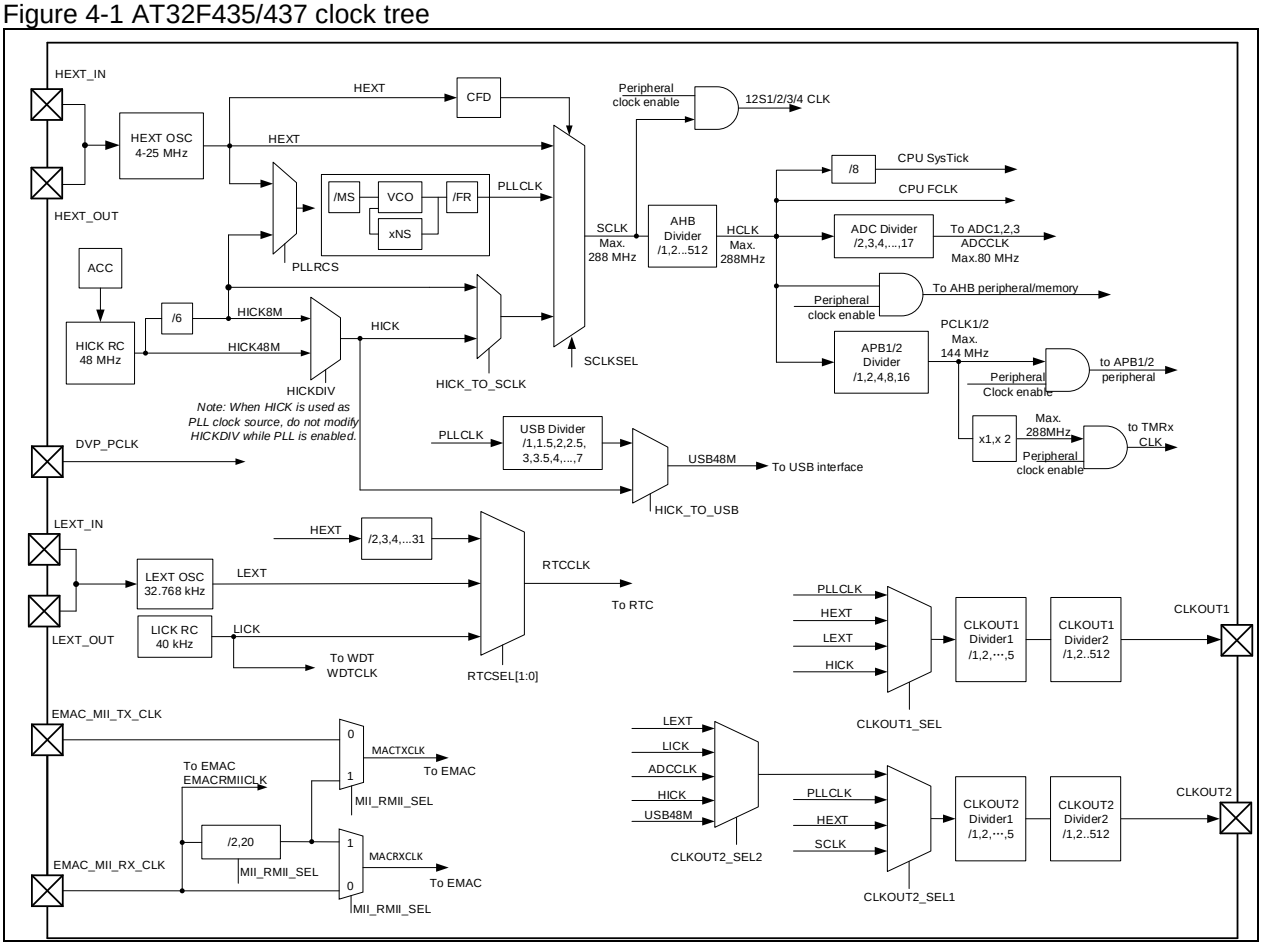
Bit	Name	Reset value	Type	Description
Bit 31: 3	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 2: 0	LDOOVSEL	0x0	rw	LDO output voltage select 000: 1.2V 001: 1.3V 010~011: Unused, not configurable 100: 1.1V 101: 1.0V 110~111: Unused, not configurable

4 Clock and reset manage (CRM)

4.1 Clock

AT32F435/437 series provide different clock sources: HEXT oscillator clock, HICK oscillator clock, PLL clock, LEXT oscillator and LICK oscillator.

Figure 4-1 AT32F435/437 clock tree



AHB, APB1 and APB2 all support multiple frequency division. The AHB domain has a maximum of 288 MHz, and both APB1 and APB2 are up to 144 MHz.

4.1.1 Clock sources

- High speed external oscillator (HEXT)

The HEXT includes two clock sources: crystal/ceramic resonator and bypass clock.

The HEXT crystal/ceramic resonator is connected externally to a 4~25 MHz HEXT crystal that produces a highly accurate clock for the system. The HEXT clock signal is not released until it becomes stable.

An external clock source can be provided by HEXT bypass. Its frequency can be up to 25 MHz. The external clock signal should be connected to the HEXT_IN pin while the HEXT_OUT pin should be left floating.

- High speed internal clock (HICK)

The HICK oscillator is clocked by a high-speed RC in the microcontroller. The internal frequency of the HICK clock is 48 MHz. Although it is less accurate, its startup time is shorter than the HEXT crystal oscillator. The HICK clock frequency of each device is calibrated by ARTERY to 1% accuracy (25°C) in factory. The factory calibration value is loaded in the HICKCAL[7: 0] bit of the clock control register. The RC oscillator speed may be affected by voltage or temperature variations. Thus the HICK frequency can be trimmed using the HICKTRIM[5: 0] bit in the clock control register.

The HICK clock signal is not released until it becomes stable.

- PLL clock

The HICK or HEXT clock can be used as an input clock source of the PLL. The PLL input clock, after being divided by a pre-divider internally, is sent to the VCO for frequency multiplication, and the VCO output frequency is output after being divided by a post-divider. At the same time, the clock after pre-divider must remain between 2 MHz and 16 MHz, and the VCO operating frequency must be kept between 500 MHz and 1200 MHz. The PLL must be configured before enabling it. The reason is that the configuration parameters cannot be changed once PLL is enabled. The PLL clock signal is not released before it becomes stable.

PLL formula:

PLL output clock = PLL input clock x PLL frequency multiplication factor / (PLL pre-divider factor x PLL post-divider factor)

500 MHz <= PLL input clock x PLL frequency multiplication factor / PLL pre-divider factor <= 1200 MHz

2 MHz <= PLL input clock / PLL pre-divider factor <= 16 MHz

For example, when the PLL input clock is 25 MHz, the PLL output frequency equals 25 x 192 / (5 x 4) = 240 MHz

- Low speed external oscillator (LEXT)

The LEXT oscillator provides two clock sources: LEXT crystal/ceramic resonator and LEXT bypass.

LEXT crystal/ceramic resonator:

The LEXT crystal/ceramic resonator provides a 32.768 KHz low-speed clock source. The LEXT clock signal is not released before it becomes stable.

- LEXT bypass clock

In this mode, an external clock source with a frequency of 32.768 kHz can be provided. The external clock signal should be connected to the LEXT_IN pin while the LEXT_OUT can be released for GPI control.

- Low speed internal RC oscillator (LICK)

The LICK oscillator is clocked by an internal low-speed RC oscillator. The clock frequency is between 30 kHz and 60 kHz. It acts as a low-power clock source that can be kept running in Deepsleep mode and Standby mode for watchdog and auto-wakeup unit.

The LICK clock signal is not released before it becomes stable.

4.1.2 System clock

After a system reset, the HICK oscillator is selected as system clock. The system clock can make flexible switch among HICK oscillator, HEXT oscillator and PLL clock. However, a switch from one clock source to another occurs only if the target clock source becomes stable. When the HICK oscillator is used directly or indirectly through the PLL as the system clock, it cannot be stopped.

4.1.3 Peripheral clock

Most peripherals use HCLK, PCLK1 or PCLK2 clock. The individual peripherals have their dedicated clocks.

System Tick timer (SysTick) is clocked by HCLK or HCLK/8.

ADCs are clocked by APB2 divided by 2, 3, 4, 5…17

The timers are clocked by APB1/2. In particular, if the APB prescaler is 1, the timer clock frequency is equal to that of APB1/2; otherwise, the timer clock frequency doubles that of the APB1/2 frequency.

The USB clock source can be switched between HICK and PLL frequency divider. If the HICK is selected as a clock source, the USB clock should be set as 48 MHz; If the PLL frequency divider is selected as a clock source, the USB frequency divider provides 48 MHz USBCLK, and thus the PLL must be set as 48*N*0.5 MHz (N=2,3,4,5…)

ERTC clock sources: divided HEXT oscillator, LEXT oscillator and LICK oscillator. Once the clock source is selected, it cannot be altered without resetting the battery powered domain. If the LEXT is used as an ERTC clock, the ERTC is not affected when the VDD is powered off. If the HEXT or LICK is selected as an ERTC clock, the ERTC state is not guaranteed when both HEXT and LICK are powered off.

Watchdog is clocked by LICK oscillator. If the watchdog is enabled by either hardware option or software access, the LICK oscillator is forced ON. The clock is provided to the watchdog only after the LICK

oscillator temporization.

4.1.4 Clock fail detector

The clock fail detector (CFD) is designed to respond to HEXT clock failure when the HEXT is used as a system clock, directly or indirectly. If a failure is detected on the HEXT clock, a clock failure event is sent to the break input of TMR1 and TMR8 and an interrupt is generated. This interrupt is directly linked to CPU NMI so that the software can perform rescue operations. The NMI interrupt keeps executing until the CFD interrupt pending bit is cleared. This is why the CFD interrupt has to be cleared in the NMI service routine. The HEXT clock failure will result in a switch of the system clock to the HICK clock, the CFD to be disabled, HEXT clock to be stopped, and even PLL to be disabled if the HEXT clock is selected as the system clock through PLL.

4.1.5 Auto step-by-step system clock switch

The automatic frequency switch is designed to ensure a smooth and stable switch of system frequency when the system clock source is switched from others to the PLL or when the AHB prescaler factor is changed from large to small. When the operational target is larger than 108 MHz, it is recommended to enable the automatic frequency switch. Once it is enabled, the AHB bus is halted by hardware till the completion of the switch. During this switch period, the DMA remain working, and the interrupt events are recorded and then handled by NVIC when the AHB bus resumes.

4.1.6 Clock output

The microcontroller allows the internal clock signal to be output to external CLKOUT1/2 pins. That is, ADCCLK, USB48M, SCLK, LICK, LEXT, HICK, HEXT, PLLCLK can be used as CLKOUT1/2 clocks. When being used as the CLKOUT1/2 clock output pin, the corresponding GPIO port registers must be configured accordingly.

4.1.7 Interrupts

The microcontroller specifies a stable flag for each clock source. As a result, when a clock source is enabled, it is possible to determine if the clock is stable by checking the flag pertaining to the clock source. An interrupt request is generated when the interrupt corresponding to the clock source is enabled.

If a failure is detected on the HEXT clock, the CFD interrupt is generated. Such interrupt is directly linked to CPU NMI.

4.2 Reset

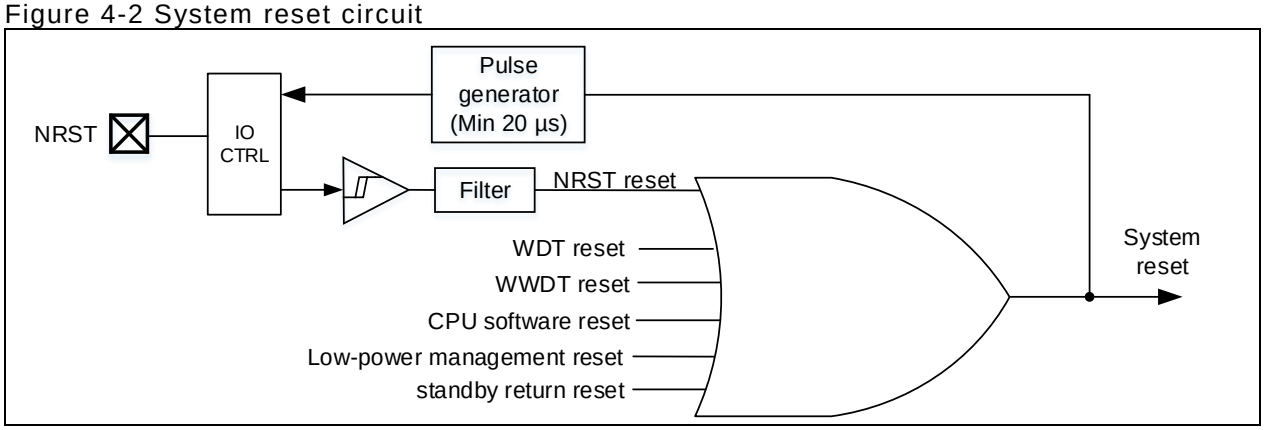
4.2.1 System reset

AT32F435/437 series provide the following system reset sources:

- NRST reset: on the external NRST pin
- WDT reset: watchdog overflow
- WWDT reset: window watchdog overflow
- CPU software reset: Cortex™-M4F software reset
- Low-power management reset: This type of reset is enabled when entering Standby mode (by clearing the nSTDBY_RST bit in the user system data area); this type of reset is also enabled when entering Deepsleep mode (by clearing the nDEPSLP_RST in the user system data area).
- POR reset: power-on reset
- LVR reset: low voltage reset
- When exiting Standby mode

NRST reset, WDT reset, WWDT reset, software reset and low-power management reset sets all registers to their reset values except the clock control/status register (CRM_CTRLSTS) and the battery powered domain; the power-on reset, low-voltage reset or reset generated when exiting Standby mode sets all registers to their reset values except the battery powered domain registers.

Figure 4-2 System reset circuit



4.2.2 Battery powered domain reset

Battery powered domain has two specific reset sources:

- Software reset: triggered by setting the BPDRST bit in the battery powered domain control register (CRM_BPDC)
- VDD or VBAT power on, if both supplies have been powered off.

Software reset affects only the battery powered domain.

4.3 CRM registers

These peripheral registers have to be accessed by bytes (8 bits), half words (16 bits) or words (32 bits).

Table 4-1 CRM register map and reset values

Register	Offset	Reset value
CRM_CTRL	0x000	0x0000 XX83
CRM_PLLCFG	0x004	0x0003 3002
CRM_CFG	0x008	0x0000 0000
CRM_CLKINT	0x00C	0x0000 0000
CRM_AHBRST1	0x010	0x0000 0000
CRM_AHBRST2	0x014	0x0000 0000
CRM_AHBRST3	0x018	0x0000 0000
CRM_APB1RST	0x020	0x0000 0000
CRM_APB2RST	0x024	0x0000 0000
CRM_AHBEN1	0x030	0x0000 0000
CRM_AHBEN2	0x034	0x0000 0000
CRM_AHBEN3	0x038	0x0000 0000
CRM_APB1EN	0x040	0x0000 0000
CRM_APB2EN	0x044	0x0000 0000
CRM_AHBLPEN1	0x050	0x3F63 90FF
CRM_AHBLPEN2	0x054	0x0000 8081
CRM_AHBLPEN3	0x058	0x0000 C003
CRM_APB1LPEN	0x060	0xF6FE E9FF
CRM_APB2LPEN	0x064	0x2017 7733
CRM_BPDC	0x070	0x0000 0000
CRM_CTRLSTS	0x074	0x0C00 0000
CRM_MISC1	0x0A0	0x0000 0000
CRM_MISC2	0x0A4	0x0000 000D

4.3.1 Clock control register (CRM_CTRL)

Bit	Name	Reset value	Type	Description
Bit 30: 26	Reserved	0x00	resd	Kept at its default value.
Bit 25	PLLSTBL	0x0	ro	PLL clock stable This bit is set by hardware after PLL is ready. 0: PLL clock is not ready. 1: PLL clock is ready.
Bit 24	PLLEN	0x0	rw	PLL enable This bit is set and cleared by software. It can also be cleared by hardware when entering Standby or Deepsleep mode. When the PLL clock is used as the system clock, this bit cannot be cleared. 0: PLL is OFF 1: PLL is ON.
Bit 23: 20	Reserved	0x0	resd	Kept at its default value.
Bit 19	CFDEN	0x0	rw	Clock failure detector enable 0: OFF 1: ON
Bit 18	HEXTBYPS	0x0	rw	High speed external crystal bypass This bit can be written only if the HEXT is disabled. 0: OFF 1: ON
Bit 17	HEXTSTBL	0x0	ro	High speed external crystal stable This bit is set by hardware after HEXT becomes stable. 0: HEXT is not ready. 1: HEXT is ready.
Bit 16	HEXTEN	0x0	rw	High speed external crystal enable This bit is set and cleared by software. It can also be cleared by hardware when entering Standby or Deepsleep mode. When the HEXT clock is used as the system clock, this bit cannot be cleared 0: OFF. 1: ON
Bit 15: 8	HICKCAL	0xXX	rw	High speed internal clock calibration The default value of this field is the initial factory calibration value. When the HICK output frequency is 48 MHz, it needs adjust 240 kHz (design value) based on this frequency for each HICKCAL value change; when HICK output frequency is 8 MHz (design value), it needs adjust 40 kHz based on this frequency for each HICKCAL value change. Note: This bit can be written only if the HICKCAL_KEY[7: 0] is set as 0x5A.
Bit 7: 2	HICKTRIM	0x20	rw	High speed internal clock trimming These bits work with the HICKCAL[7: 0] to determine the HICK oscillator frequency. The default value is 32, which can trim the HICK to be ±1%.
Bit 1	HICKSTBL	0x1	ro	High speed internal clock stable This bit is set by hardware after the HICK is ready. 0: Not ready 1: Ready

Bit 0	HICKEN	0x1	rw	High speed internal clock enable This bit is set and cleared by software. It can also be set by hardware when exiting Standby or Deepsleep mode. When a HEXT clock failure occurs. This bit can also be set. When the HICK is used as the system clock, this bit cannot be cleared. 0: Disabled 1: Enabled

4.3.2 PLL clock configuration register (CRM_PLLCFG)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value.
Bit 22	PLLRCS	0x0	rw	PLL reference clock select The PLL reference clock source is selected by setting or resetting this bit. This bit can be written only when the PLL is disabled. 0: HICK is used as PLL reference clock 1: HEXT is used as PLL reference clock
Bit 21: 19	Reserved	0x0	resd	Kept at its default value.
Bit 18: 16	PLL_FR	0x3	rw	PLL post-division PLL_FR range (2~5) 000: Reserved. Do not use. 001: Reserved. Do not use. 010: PLL post-division=4 011: PLL post-division=8 100: PLL post-division=16 101: PLL post-division=32 Others: Reserved. Do not use. Attention should be paid to the correlation between the PLL_FR value and post-division factor.
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14: 6	PLL_NS	0x0C0	rw	PLL multiplication factor PLL_NS range (31~500) 000000000 ~ 000011110: Forbidden 000011111: 31 000100000: 32 000100001: 33 …… 111110011: 499 111110100: 500 111110101~111111111: Forbidden
Bit 5: 4	Reserved	0x0	resd	Kept at its default value.
Bit 3: 0	PLL_MS	0x2	rw	PLL pre-division PLL_MS range (1~15) 0000: Forbidden 0001: 1 0010: 2 0011: 3 …… 1110: 14 1111: 15

Note: PLL clock formulas:

PLL output clock = PLL input clock x PLL frequency multiplication factor / (PLL pre-divider factor x PLL post-divider factor)

500 MHz <= PLL input clock x PLL frequency multiplication factor / PLL pre-divider factor <= 1200 MHz

2 MHz <= PLL input clock / PLL pre-divider factor <= 16 MHz

4.3.3 Clock configuration register (CRM_CFG)

Access: 0 to 2 wait states. 1 or 2 wait states are inserted only when the access occurs during a clock source switch.

Bit	Name	Reset value	Type	Description
Bit 31:30	CLKOUT2_SEL1	0x0	rw	Clock output2 selection 1 This field is set and cleared by software. 00: System clock (SCLK) selected 01: Secondary clock output selected by the CLKOUT2_SEL2 bit in the CRM_MISC1 register 10: External oscillator clock (HEXT) selected 11: PLL clock output Note: This clock out may be cut off during the startup and switch of CLKOUT2 clock source. While being used as an output to the CLKOUT2 pin, the system clock output frequency must be no more than 50 MHz (the maximum frequency of an IO port)
Bit 29: 27	CLKOUT2DIV1	0x0	rw	Clock output2 division1 0xx: CLKOUT2 100: CLKOUT2/2 101: CLKOUT2/3 110: CLKOUT2/4 111: CLKOUT2/5
Bit 26: 24	USBDIV	0x0	rw	Clock output1 division1 0xx: CLKOUT1 100: CLKOUT1/2 101: CLKOUT1/3 110: CLKOUT1/4 111: CLKOUT1/5
Bit 23	Reserved	0x0	resd	Kept at its default value.
Bit 22: 21	CLKOUT1_SEL	0x0	rw	Clock output1 selection This field is set and cleared by software. 00: HICK selected 01: LEXT selected 10: HEXT selected 11: PLL selected Note: This clock out may be cut off during the startup and switch of CLKOUT1 clock source. While being used as an output to the CLKOUT1 pin, the system clock output frequency must not exceed 50 MHz (the maximum frequency of an IO port)
Bit 20: 16	ERTCDIV	0x00	rw	HEXT division for ERTC clock This field is set and cleared by software to divide the HEXT for ERTC clock. These bits must be configured before selecting the ERTC clock source. 00000: Forbidden 00001: Forbidden 00010: HEXT/2 00011: HEXT/3 00100: HEXT/4 … 11110: HEXT/30 11111: HEXT/31

Bit 15: 13	APB2DIV	0x0	rw	APB2 division The divided HCLK is used as APB2 clock. 0xx: not divided 100: divided by 2 101: divided by 4 110: divided by 8 111: divided by 16 Note: The software must set these bits correctly to ensure that the APB2 clock frequency does not exceed 144 MHz.
Bit 12: 10	APB1DIV	0x0	rw	APB1 division The divided HCLK is used as APB1 clock. 0xx: not divided 100: divided by 2 101: divided by 4 110: divided by 8 111: divided by 16 Note: The software must set these bits correctly to ensure that the APB1 clock frequency does not exceed 144 MHz
Bit 9: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7: 4	AHBDIV	0x0	rw	AHB division 0xxx: SCLK not divided 1000: SCLK divided by 2 1100: SCLK divided by 64 1001: SCLK divided by 4 1101: SCLK divided by 128 1010: SCLK divided by 8 1110: SCLK divided by 256 1011: SCLK divided by 16 1111: SCLK divided by 512
Bit 3: 2	SCLKSTS	0x0	R0	System clock select status 00: HICK 01: HEXT 10: PLL 11: Reserved. Kept at its default value.
Bit 1: 0	SCLKSEL	0x0	rw	System clock select 00: HICK 01: HEXT 10: PLL 11: Reserved. Kept at its default value.

4.3.4 Clock interrupt register (CRM_CLKINT)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23	CFDFC	0x0	wo	Clock failure detection flag clear Writing 1 by software to clear CFDF. 0: No effect 1: Clear
Bit 22: 21	Reserved	0x0	resd	Kept at its default value.
Bit 20	PLLSTBLFC	0x0	wo	PLL stable flag clear Writing 1 by software to clear PLLSTBLF. 0: No effect 1: Clear
Bit 19	HEXTSTBLFC	0x0	wo	HEXT stable flag clear Writing 1 by software to clear HEXTSTBLF. 0: No effect 1: Clear
Bit 18	HICKSTBLFC	0x0	wo	HICK stable flag clear Writing 1 by software to clear HICKSTBLF. 0: No effect 1: Clear
Bit 17	LEXTSTBLFC	0x0	wo	LEXT stable flag clear Writing 1 by software to clear LEXTSTBLF. 0: No effect 1: Clear
Bit 16	LICKSTBLFC	0x0	wo	LICK stable flag clear Writing 1 by software to clear LICKSTBLF. 0: No effect 1: Clear
Bit 15: 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	PLLSTBLIEN	0x0	rw	PLL stable interrupt enable 0: Disabled 1: Enabled
Bit 11	HEXTSTBLIEN	0x0	rw	HEXT stable interrupt enable 0: Disabled 1: Enabled
Bit 10	HICKSTBLIEN	0x0	rw	HICK stable interrupt enable 0: Disabled 1: Enabled
Bit 9	LEXTSTBLIEN	0x0	rw	LEXT stable interrupt enable 0: Disabled 1: Enabled
Bit 8	LICKSTBLIEN	0x0	rw	LICK stable interrupt enable 0: Disabled 1: Enabled
Bit 7	CFDF	0x0	ro	Clock Failure Detection flag This bit is set by hardware when the HEXT clock failure occurs. 0: No clock failure 1: Clock failure
Bit 6: 5	Reserved	0x0	resd	Keep at its default value.

Bit 4	PLLSTBLF	0x0	ro	PLL stable flag Set by hardware. 0: PLL is not ready. 1: PLL is ready.
Bit 3	HEXTSTBLF	0x0	ro	HEXT stable flag Set by hardware. 0: HEXT is not ready. 1: HEXT is ready.
Bit 2	HICKSTBLF	0x0	ro	HICK stable flag Set by hardware. 0: HICK is not ready. 1: HICK is ready.
Bit 1	LEXTSTBLF	0x0	ro	LEXT stable flag Set by hardware. 0: LEXT is not ready. 1: LEXT is ready.
Bit 0	LICKSTBLF	0x0	ro	LICK stable interrupt flag Set by hardware. 0: LICK is not ready. 1: LICK is ready.

4.3.5 APB peripheral reset register1 (CRM_APBRST1)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value.
Bit 29	OTGFS2RST	0x0	rw	OTGFS2 reset 0: Does not reset the OTGFS2 1: Reset the OTGFS2
Bit 28: 26	Reserved	0x0	resd	Kept at its default value.
Bit 25	EMACRST	0x0	rw	EMAC reset 0: Does not reset the Ethernet MAC 1: Reset the Ethernet MAC
Bit 24	DMA2RST	0x0	rw	DMA2 reset 0: Does not reset DMA2 1: Reset DMA2
Bit 23	Reserved	0x0	resd	Kept at its default value.
Bit 22	DMA1RST	0x0	rw	DMA1 reset 0: Does not reset DMA1 1: Reset DMA1
Bit 21	EDMARST	0x0	rw	EDMA reset 0: Does not reset EDMA 1: Reset EDMA
Bit 20: 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	CRCRST	0x0	rw	CRC reset 0: Does not reset CRC 1: Reset CRC
Bit 11: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	GPIOHRST	0x0	rw	IO port H reset 0: Does not reset IO port H 1: Reset IO port H
Bit 6	GPIOGRST	0x0	rw	IO port G reset 0: Does not reset IO port G 1: Reset IO port G
Bit 5	GPIOFRST	0x0	rw	IO port F reset 0: Does not reset IO port F 1: Reset IO port F

Bit 4	GPIOERST	0x0	rw	IO port E reset 0: Does not reset IO port E 1: Reset IO port E
Bit 3	GPIODRST	0x0	rw	IO port D reset 0: Does not reset IO port D 1: Reset IO port D
Bit 2	GPIOCRST	0x0	rw	IO port C reset 0: Does not reset IO port C 1: Reset IO port C
Bit 1	GPIOBRST	0x0	rw	IO port B reset 0: Does not reset IO port B 1: Reset IO port B
Bit 0	GPIOARST	0x0	rw	IO port A reset 0: Does not reset IO port A 1: Reset IO port A

4.3.6 APB peripheral reset register2 (CRM_APBRST2)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31:16	Reserved	0x0	resd	Kept at its default value.
Bit 15	SDIO1RST	0x0	rw	SDIO1 reset 0: Does not reset SDIO1 1: Reset SDIO1
Bit 14:8	Reserved	0x0	resd	Kept at its default value.
Bit 7	OTGFS1RST	0x0	rw	OTGFS1 reset 0: Does not reset OTGFS1 1: Reset OTGFS1
Bit 6:1	Reserved	0x0	resd	Kept at its default value.
Bit 0	DVPRST	0x0	rw	DVP reset 0: Does not reset DVP 1: Reset DVP

4.3.7 APB peripheral reset register3 (CRM_APBRST3)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31:16	Reserved	0x0	resd	Kept at its default value.
Bit 15	SDIO2RST	0x0	rw	SDIO2 reset 0: Does not reset SDIO2 1: Reset SDIO2
Bit 14	QSPI2RST	0x0	rw	QSPI2 reset 0: Does not reset QSPI2 1: Reset QSPI2
Bit 13: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	QSPI1RST	0x0	rw	QSPI1 reset 0: Does not reset QSPI1 1: Reset QSPI1
Bit 0	XMCRST	0x0	rw	XMC reset 0: Does not reset XMC 1: Reset XMC

4.3.8 APB1 peripheral reset register (CRM_APB1RST)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31	UART8RST	0x0	rw	UART8 reset 0: Does not reset UART8

				1: Reset UART8
Bit 30	UART7RST	0x0	rw	UART7 reset 0: Does not reset UART7 1: Reset UART7
Bit 29	DACRST	0x0	rw	DAC interface reset 0: Does not reset DAC interface 1: Reset DAC interface
Bit 28	PWCRST	0x0	rw	Power interface reset 0: Does not reset Power interface 1: Reset Power interface
Bit 27	Reserved	0x0	resd	Kept at its default value.
Bit 26	CAN2RST	0x0	rw	CAN2 reset 0: Does not reset CAN2 1: Reset CAN2
Bit 25	CAN1RST	0x0	rw	CAN1 reset 0: Does not reset CAN1 1: Reset CAN1
Bit 24	Reserved	0x0	resd	Kept at its default value.
Bit 23	I2C3RST	0x0	rw	I2C3 reset 0: Does not reset I2C3 1: Reset I2C3
Bit 22	I2C2RST	0x0	rw	I2C2 reset 0: Does not reset I2C2 1: Reset I2C2
Bit 21	I2C1RST	0x0	rw	I2C1 reset 0: Does not reset I2C1 1: Reset I2C1
Bit 20	UART5RST	0x0	rw	UART5 reset 0: Does not reset UART5 1: Reset UART5
Bit 19	UART4RST	0x0	rw	UART4 reset 0: Does not reset UART4 1: Reset UART4
Bit 18	USART3RST	0x0	rw	USART3 reset Set and cleared by software. 0: Does not reset USART3 1: Reset USART3
Bit 17	USART2RST	0x0	rw	USART2 reset 0: Does not reset USART2 1: Reset USART2
Bit 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	SPI3RST	0x0	rw	SPI3 reset 0: Does not reset SPI3 1: Reset SPI3
Bit 14	SPI2RST	0x0	rw	SPI3 reset 0: Does not reset SPI2 1: Reset SPI2
Bit 13: 12	Reserved	0x0	resd	Kept at its default value.
Bit 11	WWDTRST	0x0	rw	Window watchdog reset 0: Does not reset Window watchdog 1: Reset Window watchdog
Bit 10: 9	Reserved	0x0	resd	Kept at its default value.
Bit 8	TMR14RST	0x0	rw	Timer14 reset 0: Does not reset Timer14 1: Reset Timer14
Bit 7	TMR13RST	0x0	rw	Timer13 reset 0: Does not reset Timer13 1: Reset Timer13
Bit 6	TMR12RST	0x0	rw	Timer12 reset 0: Does not reset Timer12 1: Reset Timer12
Bit 5	TMR7RST	0x0	rw	Timer7 reset 0: Does not reset Timer7 1: Reset Timer7

Bit 4	TMR6RST	0x0	rw	Timer6 reset 0: Does not reset Timer6 1: Reset Timer6
Bit 3	TMR5RST	0x0	rw	Timer5 reset 0: Does not reset Timer5 1: Reset Timer5
Bit 2	TMR4RST	0x0	rw	Timer4 reset 0: Does not reset Timer4 1: Reset Timer4
Bit 1	TMR3RST	0x0	rw	Timer3 reset 0: Does not reset Timer3 1: Reset Timer3
Bit 0	TMR2RST	0x0	rw	Timer2 reset 0: Does not reset Timer2 1: Reset Timer2

4.3.9 APB2 peripheral reset register (CRM_APB2RST)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value.
Bit 29	ACCRST	0x0	rw	ACC reset 0: Does not reset ACC 1: Reset ACC
Bit 28: 21	Reserved	0x00	resd	Kept at its default value.
Bit 20	TMR20RST	0x0	rw	Timer20 reset 0: Does not reset Timer20 1: Reset Timer20
Bit 19	Reserved	0x0	resd	Kept at its default value.
Bit 18	TMR11RST	0x0	rw	Timer11 reset 0: Does not reset Timer11 1: Reset Timer11
Bit 17	TMR10RST	0x0	rw	Timer10 reset 0: Does not reset Timer10 1: Reset Timer10
Bit 16	TMR9RST	0x0	rw	Timer9 reset 0: Does not reset Timer9 1: Reset Timer9
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	SCFGRST	0x0	rw	SCFG reset 0: Does not reset SCFG 1: Reset SCFG
Bit 13	SPI4RST	0x0	rw	SPI4 reset 0: Does not reset SPI4 1: Reset SPI4
Bit 12	SPI1RST	0x0	rw	SPI1 reset 0: Does not reset SPI1 1: Reset SPI1
Bit 11: 9	Reserved	0x0	resd	Kept at its default value.
Bit 8	ADCRST	0x0	rw	ADC interface reset 0: Does not reset ADC1 interface 1: Reset ADC1 interface
Bit 7: 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	USART6RST	0x0	rw	USART6 reset 0: Does not reset USART6 1: Reset USART6
Bit 4	USART1RST	0x0	rw	USART1 reset 0: Does not reset USART1 1: Reset USART1
Bit 3: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	TMR8RST	0x0	rw	TMR8 timer reset 0: Does not reset TMR8 1: Reset TMR8
Bit 0	TMR1RST	0x0	rw	TMR1 timer reset

				0: Does not reset TMR1 1: Reset TMR1

4.3.10 APB peripheral clock enable register1 (CRM_AHBEN1)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value.
Bit 29	OTGFS2EN	0x0	rw	OTGFS2 clock enable 0: Disabled 1: Enabled
Bit 28	EMACPTPEN	0x0	rw	EMAC PTP clock enable 0: Disabled 1: Enabled
Bit 27	EMACRXEN	0x0	rw	EMAC RX clock enable 0: Disabled 1: Enabled Note: In RMII mode, if this clock is enabled, then MAC RMII clock is enabled as well.
Bit 26	EMACTXEN	0x0	rw	EMAC TX clock enable 0: Disabled 1: Enabled Note: In RMII mode, if this clock is enabled, then MAC RMII clock is enabled as well.
Bit 25	EMACEN	0x0	rw	EMAC clock enable 0: Disabled 1: Enabled
Bit 24	DMA2EN	0x0	rw	DMA2 clock enable 0: Disabled 1: Enabled
Bit 23	Reserved	0x0	resd	Kept at its default value.
Bit 22	DMA1EN	0x0	rw	DMA1 clock enable 0: Disabled 1: Enabled
Bit 21	EDMAEN	0x0	rw	EDMA clock enable 0: Disabled 1: Enabled
Bit 20:13	Reserved	0x0	resd	Kept at its default value.
Bit 12	CRCEN	0x0	rw	CRC clock enable 0: Disabled 1: Enabled
Bit 11: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	GPIOHEN	0x0	rw	IO port H clock enable 0: Disabled 1: Enabled
Bit 6	GPIOGEN	0x0	rw	IO port G clock enable 0: Disabled 1: Enabled
Bit 5	GPIOFEN	0x0	rw	IO port F clock enable 0: Disabled 1: Enabled
Bit 4	GPIOEEN	0x0	rw	IO port E clock enable 0: Disabled 1: Enabled

Bit 3	GPIODEN	0x0	rw	IO port D clock enable 0: Disabled 1: Enabled
Bit 2	GPIOCEN	0x0	rw	IO port C clock enable 0: Disabled 1: Enabled
Bit 1	GPIOBEN	0x0	rw	IO port B clock enable 0: Disabled 1: Enabled
Bit 0	GPIOAEN	0x0	rw	IO port A clock enable 0: Disabled 1: Enabled

4.3.11 APB peripheral clock enable register2 (CRM_AHBEN2)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 16	Reserved	0x00	resd	Kept at its default value.
Bit 15	SDIO1EN	0x0	rw	SDIO1 clock enable 0: Disabled 1: Enabled
Bit 14: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	OTGFS1EN	0x0	rw	OTGFS1 clock enable 0: Disabled 1: Enabled
Bit 6:1	Reserved	0x0	resd	Kept at its default value.
Bit 0	DVPEN	0x0	rw	DVP clock enable 0: Disabled 1: Enabled

4.3.12 APB1 peripheral clock enable register3 (CRM_AHBEN3)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 16	Reserved	0x00	resd	Kept at its default value.
Bit 15	SDIO2EN	0x0	rw	SDIO2 clock enable 0: Disabled 1: Enabled
Bit 14	QSPI2EN	0x0	rw	QSPI2 clock enable 0: Disabled 1: Enabled
Bit 13: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	QSPI1EN	0x0	rw	QSPI1 clock enable 0: Disabled 1: Enabled
Bit 0	XMCEN	0x0	rw	XMC clock enable 0: Disabled 1: Enabled

4.3.13 APB1 peripheral clock enable register (CRM_APB1EN)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31	UART8EN	0x0	rw	UART8 clock enable 0: Disabled 1: Enabled

Bit 30	UART7EN	0x0	rw	UART7 clock enable 0: Disabled 1: Enabled
Bit 29	DACEN	0x0	rw	DAC interface clock enable 0: Disabled 1: Enabled
Bit 28	PWCEN	0x0	rw	Power interface clock enable 0: Disabled 1: Enabled
Bit 27	Reserved	0x0	resd	Kept at its default value.
Bit 26	CAC2EN	0x0	rw	CAN2 clock enable 0: Disabled 1: Enabled
Bit 25	CAC1EN	0x0	rw	CAN1 clock enable 0: Disabled 1: Enabled
Bit 24	Reserved	0x0	resd	Kept at its default value.
Bit 23	I2C3EN	0x0	rw	I2C3 clock enable 0: Disabled 1: Enabled
Bit 22	I2C2EN	0x0	rw	I2C2 clock enable 0: Disabled 1: Enabled
Bit 21	I2C1EN	0x0	rw	I2C1 clock enable 0: Disabled 1: Enabled
Bit 20	UART5EN	0x0	rw	UART5 clock enable 0: Disabled 1: Enabled
Bit 19	UART4EN	0x0	rw	UART4 clock enable 0: Disabled 1: Enabled
Bit 18	USART3EN	0x0	rw	USART3 clock enable 0: Disabled 1: Enabled
Bit 17	USART2EN	0x0	rw	USART2 clock enable 0: Disabled 1: Enabled
Bit 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	SPI3EN	0x0	rw	SPI3 clock enable 0: Disabled 1: Enabled
Bit 14	SPI2EN	0x0	rw	SPI2 clock enable 0: Disabled 1: Enabled
Bit 13: 12	Reserved	0x0	rw	Kept at its default value.
Bit 11	WWDTEN	0x0	rw	Window watchdog clock enable 0: Disabled 1: Enabled
Bit 10: 9	Reserved	0x0	rw	Kept at its default value.
Bit 8	TMR14EN	0x0	rw	Timer14 clock enable 0: Disabled 1: Enabled

Bit 7	TMR13EN	0x0	rw	Timer13 clock enable 0: Disabled 1: Enabled
Bit 6	TMR12EN	0x0	rw	Timer12 clock enable 0: Disabled 1: Enabled
Bit 5	TMR7EN	0x0	rw	Timer7 clock enable 0: Disabled 1: Enabled
Bit 4	TMR6EN	0x0	rw	Timer6 clock enable 0: Disabled 1: Enabled
Bit 3	TMR5EN	0x0	rw	Timer5 clock enable 0: Disabled 1: Enabled
Bit 2	TMR4EN	0x0	rw	Timer4 clock enable 0: Disabled 1: Enabled
Bit 1	TMR3EN	0x0	rw	Timer3 clock enable 0: Disabled 1: Enabled
Bit 0	TMR2EN	0x0	rw	Timer2 clock enable 0: Disabled 1: Enabled

4.3.14 APB2 peripheral clock enable register (CRM_AHB2EN)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value.
Bit 29	ACCEN	0x0	rw	ACC clock enable 0: Disabled 1: Enabled
Bit 28: 21	Reserved	0x0	resd	Kept at its default value.
Bit 20	TMR20EN	0x0	rw	Timer20 clock enable 0: Disabled 1: Enabled
Bit 19	Reserved	0x0	resd	Kept at its default value.
Bit 18	TMR11EN	0x0	rw	Timer11 clock enable 0: Disabled 1: Enabled
Bit 17	TMR10EN	0x0	rw	Timer10 clock enable 0: Disabled 1: Enabled
Bit 16	TMR9EN	0x0	rw	Timer9 clock enable 0: Disabled 1: Enabled
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	SCFGEN	0x0	rw	SCFG clock enable 0: Disabled 1: Enabled

Bit 13	SPI4EN	0x0	rw	SPI4 clock enable 0: Disabled 1: Enabled
Bit 12	SPI1EN	0x0	rw	SPI1 clock enable 0: Disabled 1: Enabled
Bit 11	Reserved	0x0	resd	Kept at its default value.
Bit 10	ADC3EN	0x0	rw	ADC3 interface clock enable 0: Disabled 1: Enabled
Bit 9	ADC2EN	0x0	rw	ADC2 interface clock enable 0: Disabled 1: Enabled
Bit 8	ADC1EN	0x0	rw	ADC1 interface clock enable 0: Disabled 1: Enabled
Bit 7: 6	Reserved	0x0	rw	Kept at its default value.
Bit 5	USART6EN	0x0	rw	USART6 clock enable 0: Disabled 1: Enabled
Bit 4	USART1EN	0x0	rw	USART1 clock enable 0: Disabled 1: Enabled
Bit 3: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	TMR8EN	0x0	rw	Timer8 clock enable 0: Disabled 1: Enabled
Bit 0	TMR1EN	0x0	rw	Timer1 clock enable 0: Disabled 1: Enabled

4.3.15 APB peripheral clock enable in low power mode register1 (CRM_AHBLPEN1)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value.
Bit 29	OTGFS2LPEN	0x1	rw	OTGFS2 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 28	EMACPTPLPEN	0x1	rw	EMAC PTP clock enable in sleep mode 0: Disabled 1: Enabled
Bit 27	EMACRXLPEN	0x1	rw	EMAC RX clock enable in sleep mode 0: Disabled 1: Enabled Note: In RMII mode, if this clock is enabled, then the MAC RMII clock is enabled as well.
Bit 26	EMACTXLPEN	0x1	rw	EMAC TX clock enable in sleep mode Set and cleared by software. 0: Disabled 1: Enabled Note: In RMII mode, if this clock is enabled, then the MAC RMII clock is enabled as well.
Bit25	EMACLPEN	0x1	rw	EMAC clock enable during sleep mode 0: Disabled 1: Enabled

Bit 24	DMA2LPEN	0x1	rw	DMA2 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 23	Reserved	0x0	resd	Kept at its default value.
Bit 22	DMA1LPEN	0x1	rw	DMA1 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 21	EDMALPEN	0x1	rw	EDMA clock enable in sleep mode 0: Disabled 1: Enabled
Bit 20: 18	Reserved	0x0	resd	Kept at its default value.
Bit 17	SRAM2LPEN	0x1	rw	SRAM2 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 16	SRAM1LPEN	0x1	rw	SRAM1 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 15	FLASHLPEN	0x1	rw	FLASH clock enable during sleep mode 0: Disabled 1: Enabled
Bit 14: 13	Reserved	0x0	resd	Kept at its default value.
Bit12	CRCLPEN	0x1	rw	CRC clock enable during sleep mode 0: Disabled 1: Enabled
Bit 11: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	GPIOHLPEN	0x1	rw	O port H clock enable in sleep mode 0: Disabled 1: Enabled
Bit 6	GPIOGLPEN	0x1	rw	IO port G clock enable in sleep mode 0: Disabled 1: Enabled
Bit 5	GPIOFLPEN	0x1	rw	IO port F clock enable in sleep mode 0: Disabled 1: Enabled
Bit 4	GPIOELPEN	0x1	rw	IO port E clock enable in sleep mode 0: Disabled 1: Enabled
Bit 3	GPIODLPEN	0x1	rw	IO port D clock enable in sleep mode 0: Disabled 1: Enabled
Bit 2	GPIOCLPEN	0x1	rw	IO port C clock enable in sleep mode 0: Disabled 1: Enabled
Bit 1	GPIOBLPEN	0x1	rw	IO port B clock enable in sleep mode 0: Disabled 1: Enabled
Bit 0	GPIOALPEN	0x1	rw	IO port A clock enable in sleep mode 0: Disabled 1: Enabled

4.3.16 APB peripheral clock enable in low power mode register2 (CRM_AHBLPEN2)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15	SDIO1LPEN	0x1	rw	SDIO1 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 14: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7	OTGFS1LPEN	0x1	rw	OTGFS1 clock enable in sleep mode 0: Disabled 1: Enabled

Bit 6: 1	Reserved	0x00	resd	Kept at its default value.
Bit 0	DVPLPEN	0x1	rw	DVP clock enable in sleep mode 0: Disabled 1: Enabled

4.3.17 APB peripheral clock enable in low power mode register3 (CRM_AHBLPEN3)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15	SDIO2LPEN	0x0	rw	SDIO2 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 14	QSPI2LPEN	0x0	rw	QSPI2 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 13: 2	Reserved	0x000	resd	Kept at its default value.
Bit 1	QSPI1LPEN	0x1	rw	QSPI1 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 0	XMCLPEN	0x1	rw	XMC clock enable in sleep mode 0: Disabled 1: Enabled

4.3.18 APB1 peripheral clock enable in low power mode register (CRM_AHB1LPEN)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31	UART8LPEN	0x1	rw	UART8 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 30	UART7LPEN	0x1	rw	UART7 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 29	DACLPEN	0x1	rw	DAC interface clock enable in sleep mode 0: Disabled 1: Enabled
Bit 28	PWCLPEN	0x1	rw	Power interface clock enable in sleep mode 0: Disabled 1: Enabled
Bit 27	Reserved	0x0	resd	Kept at its default value.
Bit 26	CAN2LPEN	0x1	rw	CAN2 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 25	CAN1LPEN	0x1	rw	CAN1 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 24	Reserved	0x0	resd	Kept at its default value.
Bit 23	I2C3LPEN	0x1	rw	I2C3 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 22	I2C2LPEN	0x1	rw	I2C2 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 21	I2C1LPEN	0x1	rw	I2C1 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 20	UART5LPEN	0x1	rw	UART5 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 19	UART4LPEN	0x1	rw	UART4 clock enable in sleep mode

				0: Disabled 1: Enabled
Bit 18	USART3LPEN	0x1	rw	USART3 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 17	USART2LPEN	0x1	rw	USART2 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	SPI3LPEN	0x1	rw	SPI3 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 14	SPI2LPEN	0x1	rw	SPI 2 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 13: 12	Reserved	0x2	resd	Kept at its default value.
Bit 11	WWDTLPEN	0x1	rw	Window watchdog clock enable in sleep mode 0: Disabled 1: Enabled
Bit 10: 9	Reserved	0x0	resd	Kept at its default value.
Bit 8	TMR14LPEN	0x1	rw	Timer14 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 7	TMR13LPEN	0x1	rw	Timer13 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 6	TMR12LPEN	0x1	rw	Timer12 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 5	TMR7LPEN	0x1	rw	Timer 7 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 4	TMR6LPEN	0x1	rw	Timer 6 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 3	TMR5LPEN	0x1	rw	Timer 5 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 2	TMR4LPEN	0x1	rw	Timer 4 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 1	TMR3LPEN	0x1	rw	Timer 3 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 0	TMR2LPEN	0x1	rw	Timer 2 clock enable in sleep mode 0: Disabled 1: Enabled

4.3.19 APB2 peripheral clock enable in low power mode register (CRM_AHB2LPEN)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value.
Bit 29	ACCLPEN	0x1	rw	ACC clock enable in sleep mode 0: Disabled 1: Enabled
Bit 28: 21	Reserved	0x00	resd	Kept at its default value.
Bit 20	TMR20LPEN	0x1	rw	Timer20 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 19	Reserved	0x0	resd	Kept at its default value.
Bit 18	TMR11LPEN	0x1	rw	Timer11 clock enable during sleep mode

				0: Disabled 1: Enabled
Bit 17	TMR10LPEN	0x1	rw	Timer10 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 16	TMR9LPEN	0x1	rw	Timer9 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	SCFGLPEN	0x1	rw	SCFG clock enable in sleep mode 0: Disabled 1: Enabled
Bit 13	SPI4LPEN	0x1	rw	SPI4 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 12	SPI1LPEN	0x1	rw	SPI1 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 11	Reserved	0x0	resd	Kept at its default value.
Bit 10	ADC3LPEN	0x1	rw	ADC3 interface clock enable in sleep mode 0: Disabled 1: Enabled
Bit 9	ADC2LPEN	0x1	rw	DC2 interface clock enable in sleep mode 0: Disabled 1: Enabled
Bit 8	ADC1LPEN	0x1	rw	ADC1 interface clock enable in sleep mode 0: Disabled 1: Enabled
Bit 7: 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	USART6LPEN	0x1	rw	USART6 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 4	USART1LPEN	0x1	rw	USART1 clock enable in sleep mode 0: Disabled 1: Enabled
Bit 3: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	TMR8LPEN	0x1	rw	TMR8 timer clock enable in sleep mode 0: Disabled 1: Enabled
Bit 0	TMR1LPEN	0x1	rw	TMR1 timer clock enable in sleep mode 0: Disabled 1: Enabled

4.3.20 Battery powered domain control register (CRM_BPDC)

Access: 0 to 3 wait states, accessible by words, half-words or bytes. Wait states are inserted in the case of consecutive accesses to this register.

Note: LEXTEN, LEXTBYPS, ERTCSEL, and ERTCEN bits of the battery powered domain control register CRM_BDC) are in the battery powered domain. As a result, these bits are write protected after reset, and can only be modified by setting the BPWEN bit in the power control register (PWR_CTRL). These bits could be reset only by battery powered domain reset. Any internal or external reset does not affect these bits.

Bit	Name	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	BPDRST	0x0	rw	Battery powered domain software reset 0: Do not reset battery powered domain software 1: Reset battery powered domain software
Bit 15	ERTCEN	0x0	rw	ERTC clock enable Set and cleared by software. 0: Disabled 1: Enabled

Bit 14: 10	Reserved	0x00	resd	Kept at its default value.
Bit 9: 8	eRTCSEL	0x0	rw	eRTC clock selection Once the ERTC clock source is selected, it cannot be changed until the BPDRST bit is reset. 00: No clock 01: LEXT 10: LICK 11: Divided HEXT (with the ERTC_DIV bit in the CRM_CFG)
Bit 7: 3	Reserved	0x00	resd	Kept at its default value.
Bit 2	LEXTBYPS	0x0	rw	Low speed external crystal bypass 0: Disabled 1: Enabled
Bit 1	LEXTSTBL	0x0	ro	Low speed external oscillator stable Set by hardware after the LEXT is ready. 0: LEXT is not ready. 1: LEXT is ready.
Bit 0	LEXTEN	0x0	rw	External low-speed oscillator enable 0: Disabled 1: Enabled

4.3.21 Control/status register (CRM_CTRLSTS)

Reset flag can only be cleared by power reset or by writing the RSTFC bit, while others are cleared by system reset.

Access: 0 to 3 wait states, accessible by words, half-words or bytes. Wait states are inserted in the case of consecutive accesses to this register.

Bit	Name	Reset value	Type	Description
Bit 31	LPRSTF	0x0	ro	Low-power reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No low-power reset occurs 1: Low-power reset occurs
Bit 30	WWDTRSTF	0x0	ro	Window watchdog timer reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No window watchdog timer reset occurs 1: Window watchdog timer reset occurs
Bit 29	WDTRSTF	0x0	ro	Watchdog timer reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No watchdog timer reset occurs 1: Watchdog timer reset occurs.
Bit 28	SWRSTF	0x0	ro	Software reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No software reset occurs 1: Software reset occurs.
Bit 27	PORRSTF	0x1	ro	POR/LVR reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No POR/LVR reset occurs 1: POR/LVR reset occurs.
Bit 26	NRSTF	0x1	rw	NRST pin reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No NRST pin reset occurs 1: NRST pin reset occurs

Bit 25	Reserved	0x0	resd	Kept at its default value.
Bit 24	RSTFC	0x0	rw	Reset flag clear Cleared by writing 1 through software. 0: No effect 1: Clear the reset flag.
Bit 23: 2	Reserved	0x000000	resd	Kept at its default value.
Bit 1	LICKSTBL	0x0	ro	LICK stable 0: LICK is not ready. 1: LICK is ready.
Bit 0	LICKEN	0x0	rw	LICK enable 0: Disabled 1: Enabled

4.3.22 Additional register1 (CRM_MISC1)

Access: 0 to 3 wait states, accessible by words, half-words or bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 28	CLKOUT2DIV2	0x0	rw	Clock output2 division2 0xxx: Clock output 1000: Clock output divided by 2 1001: Clock output divided by 4 1010: Clock output divided by 8 1011: Clock output divided by 16 1100: Clock output divided by 64 1101: Clock output divided by 128 1110: Clock output divided by 256 1111: Clock output divided by 512
Bit 27: 24	CLKOUT1DIV2	0x0	rw	Clock output1 division2 0xxx: Clock output 1000: Clock output divided by 2 1001: Clock output divided by 4 1010: Clock output divided by 8 1011: Clock output divided by 16 1100: Clock output divided by 64 1101: Clock output divided by 128 1110: Clock output divided by 256 1111: Clock output divided by 512
Bit 23: 20	Reserved	0x0	resd	Kept its default value.
Bit 19: 16	CLKOUT2_SEL2	0x0	rw	Clock output2 sel2 0000: USB clock output 0001: ADC clock output 0010: Internal RC oscillator clock (HICK) output frequency divider 0011: LICK clock output 0100: LEXT clock output 0101~1111: Reserved
Bit 15	Reserved	0x0	resd	Kept its default value.
Bit 14	HICK_TO_SCLK	0x0	rw	HICK as system clock frequency select When the HICK is selected as the clock source of SCLKSEL, the frequency of SCLK is: 0: Fixed 8 MHz, that is, HICK/6 1: 48 MHz or 8 MHz, depending on the HICKDIV
Bit 13	HICK_TO_USB	0x0	rw	USB 48 MHz clock source select 0: PLL or PLL division 1: HICK or HICK/6 Note: Since USB must work at 48 MHz, HICKDIV=1 must be guaranteed to ensure that the HICK 48 MHz is selected as the clock source of USB 48 MHz.
Bit 12	HICKDIV	0x0	rw	HICK 6 divider selection

				This bit is used to select HICK or HICK /6. If the HICK/6 is selected, the clock frequency is 8 MHz. Otherwise, the clock frequency is 48 MHz. 0: HICK/6 1: HICK Note: When HICK is used as PLL clock source, do not modify HICKDIV while PLL is enabled.
Bit 11: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7: 0	HICKCAL_KEY	0x00	rw	HICK calibration key The HICKCAL [7:0] can be written only when this field is set 0x5A.

4.3.23 Additional register2 (CRM_MISC2)

Bit	Name	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 12	USBDIV	0x0	rw	USB division The PLL clock frequency, after being divided, is used as USB clock. 0000: PLL clock divided by 1.5 0001: PLL clock not divided 0010: PLL clock divided by 2.5 0011: PLL clock divided by 2 0100: PLL clock divided by 3.5 0101: PLL clock divided by 3 0110: PLL clock divided by 4.5 0111: PLL clock divided by 4 1000: PLL clock divided by 5.5 1001: PLL clock divided by 5 1010: PLL clock divided by 6.5 1011: PLL clock divided by 6 1100: PLL clock divided by 7 1101~1111: Reserved
Bit 11: 10	Reserved	0x0	resd	Kept at its default value.
Bit 9	EMAC_PPS_SEL	0x0	rw	Ethernet pulse width select 0: Pulse width is 125 ms. 1: Pulse width is 1 system clock.
Bit 8	CLK1_TO_TMR	0x0	rw	CLKOUT1 internal connected to timer 10 channel 1 0: Disconnected 1: Connected
Bit 7: 6	Reserved	0x0	resd	Kept at its default value.
Bit 5: 4	AUTO_STEP_EN	0x0	rw	Auto step-by-step system clock switch enable When the system clock source is switched from others to the PLL or when the AHB prescaler is changed from large to small (system frequency is from small to large), it is recommended to enable the auto step-by-step system clock switch if the operational target is larger than 108 MHz,. Once it is enabled, the AHB bus is halted by hardware till the completion of the switch. During this switch period, the DMA remain working, and the interrupt events are recorded and then handled by NVIC when the AHB bus resumes. 00: Disabled 01: Reserved 10: Reserved 11: Enabled. When AHBDIV or SCLKSEL is modified, the auto step-by-step system clock switch is activated automatically.
Bit 3: 0	Reserved	0xd	resd	It is fixed to 0xd. Do not change.

5 Flash memory controller (FLASH)

5.1 FLASH introduction

Flash memory is divided into three parts: main Flash memory, information block and Flash memory registers.

- Main Flash memory is up to 4032 KB, including bank 1 and bank 2.
- Information block consists of 16 KB boot loader and the user system data area. The boot loader uses USART1, USART2 or USB (DFU) serial interface for ISP programming.

Main Flash memory (4032 KB) is divided into bank 1 and bank 2.

Bank 1: 2048 KB, including 32 blocks with 16 sectors in each, and 4 KB per sector.

Bank 2: 1984 KB, including 31 blocks with 16 sectors in each, and 4 KB per sector.

User system data area is 4 KB.

Table 5-1 Flash memory architecture (4032 K)

Bank		Name		Address range
Main Flash memory	Bank 1 2048 KB	Block 0	Sector 0	0x0800 0000 - 0x0800 0FFF
			Sector 1	0x0800 1000 - 0x0800 1FFF
			Sector 2	0x0800 2000 - 0x0800 2FFF
			…	…
			Sector 15	0x0800 F000 - 0x0800 FFFF
		Block 1	Sector 16	0x0801 0000 - 0x0801 0FFF
			…	…
			Sector 31	0x0801 F000 - 0x0801 FFFF
		…	…	…
		Block 31	Sector 496	0x081F 0000 - 0x081F 0FFF
			…	..
			Sector 511	0x081F F000 - 0x081F FFFF
	Bank 2 1984 KB	Block 32	Sector 512	0x0820 0000 - 0x0820 0FFF
			…	…
			Sector 527	0x0820 F000 - 0x0820 FFFF
		…	…	…
		Block 62	Sector 992	0x083E 0000 - 0x083E 0FFF
			…	…
			Sector 1007	0x083E F000 - 0x083E FFFF
Information block		16 KB Boot memory		0x1FFF 0000 - 0x1FFF 3FFF
		4 KB user system area		0x1FFF C000 - 0x1FFF CFFF

Main Flash memory (1024 KB) is divided into bank 1 and bank 2 with 512 KB in each. Each bank is sub-divided into 8 blocks that include 32 sectors in each, and 2 KB per sector.

User system data area is 512 B.

Table 5-2 Flash memory architecture (1024 K)

Bank		Name		Address range
Main Flash memory	Bank 1 512 KB	Block 0	Sector 0	0x0800 0000 – 0x0800 07FF
			Sector 1	0x0800 0800 – 0x0800 0FFF
			Sector 2	0x0800 1000 – 0x0800 17FF
			…	…
			Sector 31	0x0800 F800 - 0x0800 FFFF
		Block 1	Sector 32	0x0801 0000 - 0x0801 07FF
			…	…
			Sector 63	0x0801 F800 - 0x0801 FFFF
		…	…	…
		Block 7	Sector 224	0x0807 0000 - 0x0807 07FF
			…	…
			Sector 255	0x0807 F800 – 0x0807 FFFF
	Bank 2 512 KB	Block 8	Sector 256	0x0808 0000 – 0x0808 07FF
			…	…
			Sector 287	0x0808 F800 - 0x0808 FFFF
		…	…	…
		Block 15	Sector 480	0x080F 0000 - 0x080F 07FF
			…	…
			Sector 511	0x080F F800 – 0x080F FFFF
Information block		16 KB boot memory		0x1FFF 0000 - 0x1FFF 3FFF
		512 B user system area		0x1FFF C000 - 0x1FFF C1FF

Main Flash memory (448 KB) has only bank 1, which id divided into 7 blocks that include 16 sectors in each, and 4 K per sector.

User system data area is 4 KB.

Table 5-3 Flash memory architecture (448 K)

Bank		Name		Address range
Main Flash memory	Bank1 448 KB	Block 0	Sector 0	0x0800 0000 - 0x0800 0FFF
			Sector 1	0x0800 1000 - 0x0800 1FFF
			Sector 2	0x0800 2000 - 0x0800 2FFF
			…	…
			Sector 15	0x0800 F000 - 0x0800 FFFF
		Block 1	Sector 16	0x0801 0000 - 0x0801 0FFF
			…	…
			Sector 31	0x0801 F000 - 0x0801 FFFF
		…	…	…
		Block 6	Sector 96	0x0806 0000 - 0x0806 0FFF
			…	…
			Sector 111	0x0806 F000 - 0x0806 FFFF
Information block		16KB boot memory		0x1FFF 0000 - 0x1FFF 3FFF
		4KB user system area		0x1FFF C000 - 0x1FFF CFFF

Main Flash memory (256 KB) has only bank 1, which is divided into 4 blocks that include 32 sectors in each, and 2 K per sector.

User system data area is 512 B.

Table 5-4 Flash memory architecture (256 K)

Bank		Name		Address range
Main Flash memory	Bank 1 256 KB	Block 0	Sector 0	0x0800 0000 – 0x0800 07FF
			Sector 1	0x0800 0800 – 0x0800 0FFF
			Sector 2	0x0800 1000 – 0x0800 17FF
			…	…
			Sector 31	0x0800 F800 - 0x0800 FFFF
		Block 1	Sector 32	0x0801 0000 - 0x0801 07FF
			…	…
			Sector 63	0x0801 F800 - 0x0801 FFFF
		Block 2	Sector 64	0x0802 0000 - 0x0802 07FF
			…	…
			Sector 95	0x0802 F800 - 0x0802 FFFF
		Block 3	Sector 96	0x0803 0000 - 0x0803 07FF
			…	…
			Sector 127	0x0803 F800 – 0x0803 FFFF
Information block		16 KB Boot memory		0x1FFF 0000 - 0x1FFF 3FFF
		512 B user option bytes		0x1FFF C000 - 0x1FFF C1FF

User system data area

The system data will be read from the information block of Flash memory whenever a system reset occurs, and is saved in the user system data register (FLASH_USD) and erase programming protection status register (FLASH_EPPS).

Each system data occupies two bytes, where the low bytes corresponds to the contents in the system data area, and the high bytes represent the inverse code that is used to verify the correctness of the selected bit. When the high byte is not equal to the inverse code of the low byte (except when both high and low byte are all 0xFF), the system data loader will issue a system data error flag (USDERR) and the corresponding system data and their inverse codes are forced 0xFF.

Note: The update of the contents in the user system data area becomes effective only after a system reset.

Table 5-5 User system data area

Address	Bit	Description	
0x1FFF_C000	[7: 0]	FAP[7:0]: Flash memory access protection (Access protection enable/disable result is stored in the FLASH_USD[1] register) 0xA5: Disabled 0xFF: Enable Flash access protection (by erasing USD area. At this point, nFAP remains 0xFF) Others: Enabled	
	[15: 8]	nFAP[7: 0]: Inverse code of FAP[7: 0]	
	[23: 16]	SSB[7:0]: System configuration byte (it is stored in the FLASH_USD[9: 2] register)	
		Bit 7	Reserved
		Bit 6 (nSTDBY_WDT)	0: WDT stops counting while entering Standby mode 1: WDT does not stop counting while entering Standby mode
		Bit 5 (nDEPSLP_WDT)	0: WDT stops counting while entering Deepsleep mode 1: WDT does not stop counting while entering Deepsleep mode
		Bit 4	Reserved
		Bit 3 (BTOPT)	0: When booting from main Flash memory, if there is no boot loader in the bank 2, it will starts from bank 1, otherwise, bank 2. 1: When booting from main Flash memory, it starts from bank 1.
		Bit 2 (nSTDBY_RST)	0: Reset occurs when entering Standby mode 1: No reset occurs when entering

			Standby mode
		Bit 1 (nDEPSLP_RST)	0: Reset occurs when entering Deepsleep mode 1: No reset occurs when entering Deepsleep mode
		Bit 0 (nWDT_ATO_EN)	0: Watchdog is enabled 1: Watchdog is disabled
	[31: 24]	nSSB[7: 0]: Inverse code of SSB[7: 0]	
0x1FFF_C004	[7: 0]	Data0[7: 0]: User data 0 (It is stored in the FLASH_USD[17:10] register)	
	[15: 8]	nData0[7: 0]: Inverse code of Data0[7: 0]	
	[23: 16]	Data1[7: 0]: User data 1 (It is stored in the FLASH_USD[25: 18] register)	
	[31: 24]	nData1[7: 0]: Inverse code of Data1[7: 0]	
0x1FFF_C008	[7: 0]	EPP0[7: 0]: Flash erase/write protection byte 0 (in the FLASH_EPPS[7: 0]) This field is used to protect the 1st~32nd KB of main Flash memory. Each bit takes care of 4 KB sectors. 0: Erase/write protection is enabled 1: Erase/write protection is disabled	
	[15: 8]	nEPP0[7: 0]: Inverse code of EPP0[7: 0]	
	[23: 16]	EPP1[7: 0]: Flash erase/write protection byte 1 (stored in the FLASH_EPPS[15: 8]) This field is used to protect the 33rd~64th KB of main Flash memory. Each bit takes care of 4 KB sectors. 0: Erase/write protection is enabled 1: Erase/write protection is disabled	
	[31: 24]	nEPP1[7: 0]: Inverse code of EPP1[7: 0]	
0x1FFF_C00C	[7: 0]	EPP2[7: 0]: Flash erase/write protection byte 2 (stored in the FLASH_EPPS[23: 16]) This field is used to protect the 65th~96th KB of main Flash memory. Each bit takes care of 4 KB sectors. 0: Erase/write protection is enabled 1: Erase/write protection is disabled	
	[15: 8]	nEPP2[7: 0]: Inverse code of EPP2[7: 0]	
	[23: 16]	EPP3[7: 0]: Flash erase/write protection byte 3 (stored in the FLASH_EPPS[31: 24]) This field is used to protect the 97th~128th KB of main Flash memory. Each bit takes care of 4 KB sectors. 0: Erase/write protection is enabled 1: Erase/write protection is disabled	
	[31: 24]	nEPP3[7: 0]: Inverse code of EPP3[7: 0]	
0x1FFF_C010	[7: 0]	EOPB0[7: 0]: Extended system options Refer to Table 5-5 for more details.	
	[15: 8]	nEOPB0[7: 0]: Inverse code of EOPB0[7: 0]	
	[31: 16]	Reserved	
0x1FFF_C014	[7: 0]	EPP4[7: 0]: Flash erase/write protection byte 4 (stored in the FLASH_EPPS1[7: 0]) This field is used to protect the 129th~1152nd KB of main Flash memory. Each bit takes care of 128 KB sectors. 0: Erase/write protection is enabled 1: Erase/write protection is disabled	
	[15: 8]	nEPP4[7: 0]: Inverse code of EPP4 [7: 0]	
	[23: 16]	EPP5[7:0]: Flash erase/write protection byte 5 (stored in the FLASH_EPPS1[15: 8]) This field is used to protect the 1153rd~2176th KB of main Flash memory. Each bit takes care of 128 KB sectors. 0: Erase/write protection is enabled 1: Erase/write protection is disabled	
	[31: 24]	nEPP5 [7: 0]: Inverse code of EPP5[7: 0]	
0x1FFF_C018	[7: 0]	EPP6[7:0]: Flash erase/write protection byte 6 (stored in the FLASH_EPPS1[23: 16]) Bit 7 is reserved. Bit [6:0] is used to protect the 2177th~3200th KB of main Flash memory. Each bit takes care of 128 KB sectors.	

		0: Erase/write protection is enabled 1: Erase/write protection is disabled
	[15: 8]	nEPP6[7: 0]: Inverse code of EPP6[7: 0]
	[23: 16]	EPP7[7:0]: Flash erase/write protection byte 7 (stored in the FLASH_EPPS1[31: 24]) Bit 7 is reserved. Bit 6:0 is used to protect the 3201st~4032nd KB of main Flash memory. Each bit takes care of 128 KB sectors. 0: Erase/write protection is enabled
	[31: 24]	nEPP7 [7: 0]: Inverse code of EPP7[7: 0]
0x1FFF_C01C 0x1FFF_C020 0x1FFF_C024 0x1FFF_C028 0x1FFF_C02C 0x1FFF_C030	[31: 0]	Reserved
0x1FFF_C034	[7: 0]	QSPIKEY0 [7: 0]: QuadSPI (QSPI) ciphertext access area encryption key byte 0 The situations for non-encryption includes: Both QSPIKEYx and nQSPIKEYx are 0xFF (default erase status) Write 0x00 to QSPIKEYx Write 0xFF to QSPIKEYx That is, {nQSPIKEYx, QSPIKEYx} are all 0xFFFF, 0xFF00 and 0x00FF. Among them, QSPIKEY0-QSPIKEY3 are the encrypted key of QSPI1.
	[15: 8]	nQSPIKEY0[7: 0]: Inverse code of QSPIKEY0[7: 0]
	[23: 16]	QSPIKEY1[7: 0]: QuadSPI (QSPI) ciphertext encryption key byte 1
	[31: 24]	nQSPIKEY1[7: 0]: Inverse code of QSPIKEY1[7: 0]
0x1FFF_C038	[7: 0]	QSPIKEY2[7: 0]: QuardSPI (QSPI) ciphertext encryption key byte 2
	[15: 8]	nQSPIKEY2[7:0]: Inverse code of QSPIKEY2 [7: 0]
	[23: 16]	QSPIKEY3[7: 0]: QuardSPI (QSPI) ciphertext encryption key byte 3
	[31: 24]	nQSPIKEY3[7: 0]: Inverse code of QSPIKEY3[7: 0]
0x1FFF_C03C	[7: 0]	QSPIKEY4[7: 0]: QuardSPI (QSPI) ciphertext encryption key byte 4
	[15: 8]	nQSPIKEY4[7: 0]: Inverse code of QSPIKEY4[7: 0]
	[23: 16]	QSPIKEY5[7: 0]: QuadSPI (QSPI) ciphertext encryption key byte 5
	[31: 24]	nQSPIKEY5[7: 0]: Inverse code of QSPIKEY5[7: 0]
0x1FFF_C040	[7: 0]	QSPIKEY6[7: 0]: QuadSPI (QSPI) ciphertext encryption key byte 6
	[15: 8]	nQSPIKEY6[7: 0]: Inverse code of QSPIKEY6[7: 0]
	[23: 16]	EXT_FLASH_KEY7[7: 0]: External memory ciphertext encryption key byte 7
	[31: 24]	QSPIKEY7[7: 0]: QuadSPI (QSPI) ciphertext encryption key byte 7
0x1FFF_C044 0x1FFF_C048	[31: 0]	Reserved
0x1FFF_C04C	[7: 0]	Data2[7: 0]: User data2
	[15: 8]	nData2[7: 0]: Inverse code of Data2[7: 0]
	[23: 16]	Data3[7: 0]: User data3
	[31: 24]	nData3[7: 0]: Inverse code of Data3[7: 0]
. .	. .	. .
0x1FFF_C1FC	[7: 0]	Data218[7: 0]: User data218
	[15: 8]	nData218[7: 0]: Inverse code of Data218[7: 0]
	[23: 16]	Data219[7: 0]: User data219
	[31: 24]	nData219[7: 0]: Inverse code of Data219[7: 0]
. .	. .	. .
0x1FFF_CFFC	[7: 0]	Data2010[7: 0]: User data2010
	[15: 8]	nData2010[7: 0]: Inverse code of Data2010[7: 0]
	[23: 16]	Data2011[7: 0]: User data2011
	[31: 24]	nData2011[7: 0]: Inverse code of Data2011[7: 0]

Note: 1. For 1024 KB Flash memory and 256 KB Flash memory, there are 256 bytes of user system data, ranging from 0x1FFF_C000 to 0x1FFF_C1FF, including 220 bytes of user data.

2. For 4032 KB Flash memory and 448 KB Flash memory, there are 2048 bytes of user system data, ranging from 0x1FFF_C000 to 0x1FFF_CFFF, including 2012 bytes of user data.

3. For 448 KB Flash memory, only EPP0[7:0], EPP1[7:0], EPP2[7:0], EPP3[7:0] and EPP4[2:0] bits are configurable, and other EPPx bits must be fixed to 1.

Table 5-6 Extended system options

EOPB0[7: 0]: Extended system option		
256 K Flash	Bit 1: 0	00: On-chip 512 KB SRAM+128 KB zero-wait-state Flash 01: On-chip 448 KB SRAM+192 KB zero-wait-state Flash 10, 11: On-chip 384 KB SRAM+256 KB zero-wait-state Flash Note: Bit 1~0 can be altered only when security library is disabled.
	Bit 7: 2	Reserved
448 K Flash	Bit 2: 0	000: On-chip 512KB SRAM + 128KB zero-wait-state Flash 001: On-chip 448KB SRAM + 192KB zero-wait-state Flash 010: On-chip 384KB SRAM + 256KB (default) zero-wait-state Flash 011: On-chip 320KB SRAM + 320KB zero-wait-state Flash 100: On-chip 256KB SRAM + 384KB zero-wait-state Flash 101, 110, 111: On-chip 192KB SRAM + 448KB zero-wait-state Flash Note: Bit 2~0 can be altered only when security library is disabled.
	Bit 7: 3	Reserved
1024 K and above Flash	Bit 2: 0	000: On-chip 512 KB SRAM+128 KB zero-wait-state Flash 001: On-chip 448 KB SRAM+192 KB zero-wait-state Flash 010: On-chip 384 KB SRAM+256 KB zero-wait-state Flash 011: On-chip 320 KB SRAM+320 KB zero-wait-state Flash 100: On-chip 256 KB SRAM+384 KB zero-wait-state Flash 101: On-chip 192 KB SRAM+448 KB zero-wait-state Flash 110, 111: On-chip 128 KB SRAM+512 KB zero-wait-state Flash Note: Bit 2~0 can be altered only when security library is disabled.
	Bit 7: 3	Reserved

5.2 Flash memory operation

5.2.1 Unlock/lock

After reset, Flash memory is protected, by default. FLASH_CTRLx cannot be written. Write and erase operation can be performed only when the Flash memory is unlocked.

Unlock procedure:

Flash memory block can be unlocked by writing KEY1 (0x45670123) and KEY2 (0xCDEF89AB) to the FLASH_UNLOCKx register.

Note: Writing an incorrect key sequence leads to a bus error and the Flash memory is also locked until the next reset.

Lock procedure:

Flash memory block can be locked by setting the OPLK bit in the FLASH_CTRLx register.

5.2.2 Erase operation

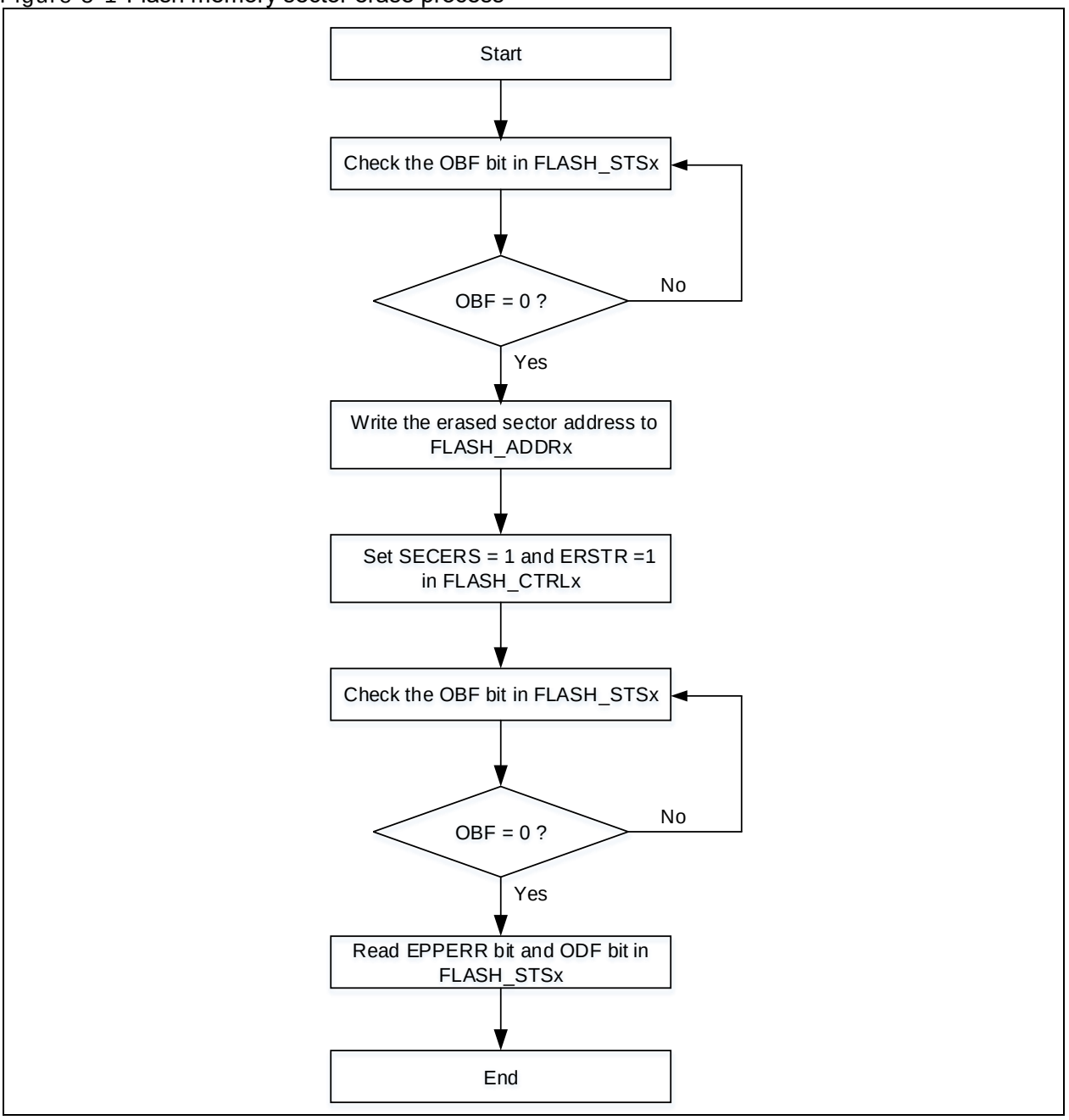
Erase operation must be done before programming. Flash memory erase includes sector erase, block erase and mass erase.

Sector erase

Any sector in the Flash memory can be erased with sector erase function independently. Below should be followed during se erase:

- Check the OBF bit in the FLASH_STSx register to confirm that there is no other programming operation in progress;
- Write the sector to be erased in the FLASH_ADDRx register;
- Set the SECERS and ERSTR bits in the FLASH_CTRLx register to enable sector erase;
- Wait until the OBF bit becomes “0” in the FLASH_STSx register. Read the EPPERR bit and ODF bit in the FLASH_STSx register to verify the erased sectors.

Figure 5-1 Flash memory sector erase process



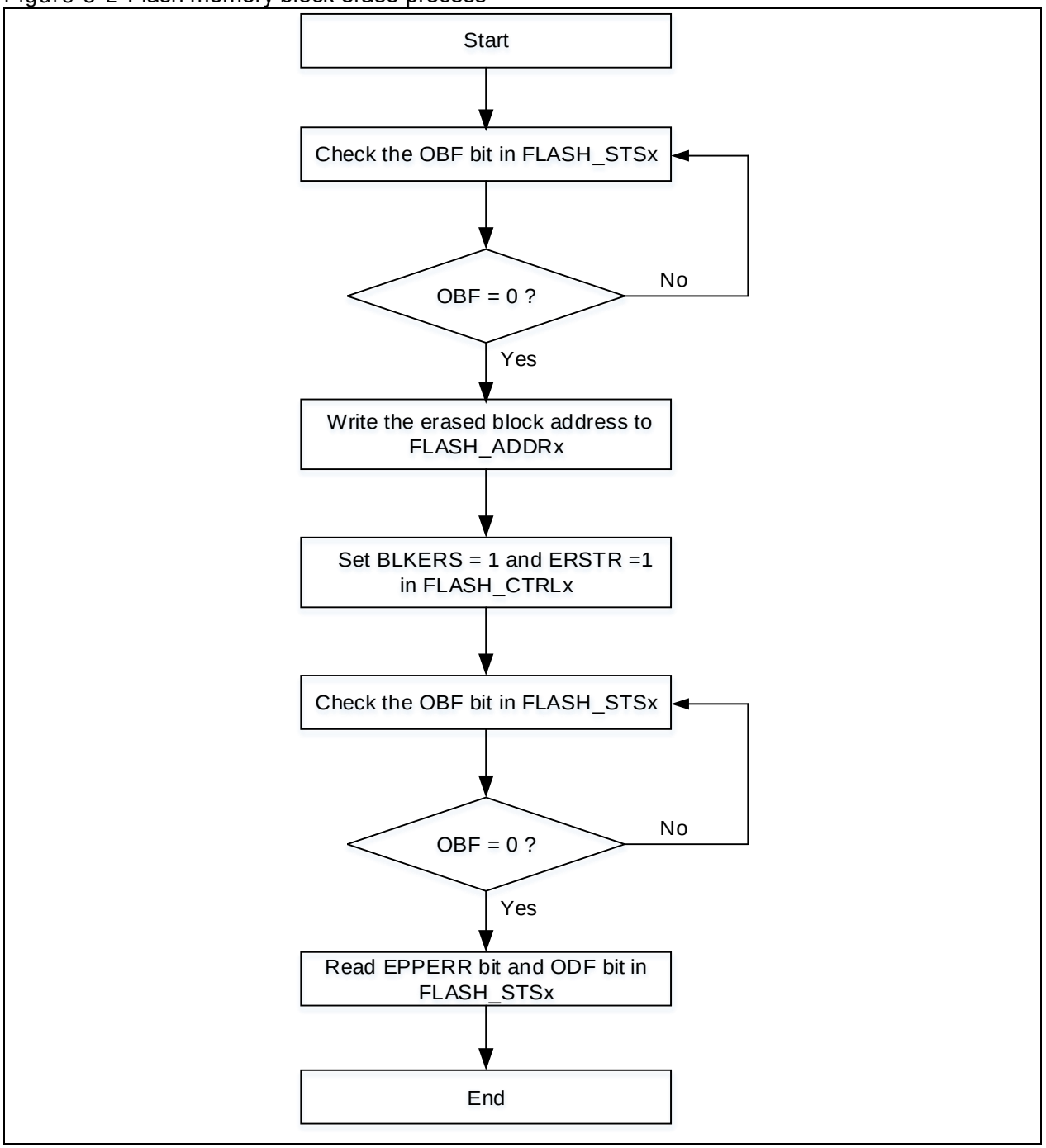
Block erase:

Any block of Flash memory can be erased independently.

The following process is recommended:

- Check the OBF bit in the FLASH_STSx register to confirm that there is no other programming operation in progress;
- Write the block to be erased in the FLASH_ADDRx register;
- Set the BANKERS and ERSTR bits in the FLASH_CTRLx register to enable block erase;
- Wait until the OBF bit becomes “0” in the FLASH_STSx register. Read the EPPERR bit and ODF bit in the FLASH_STSx register to verify the erased blocks.

Figure 5-2 Flash memory block erase process



Mass erase

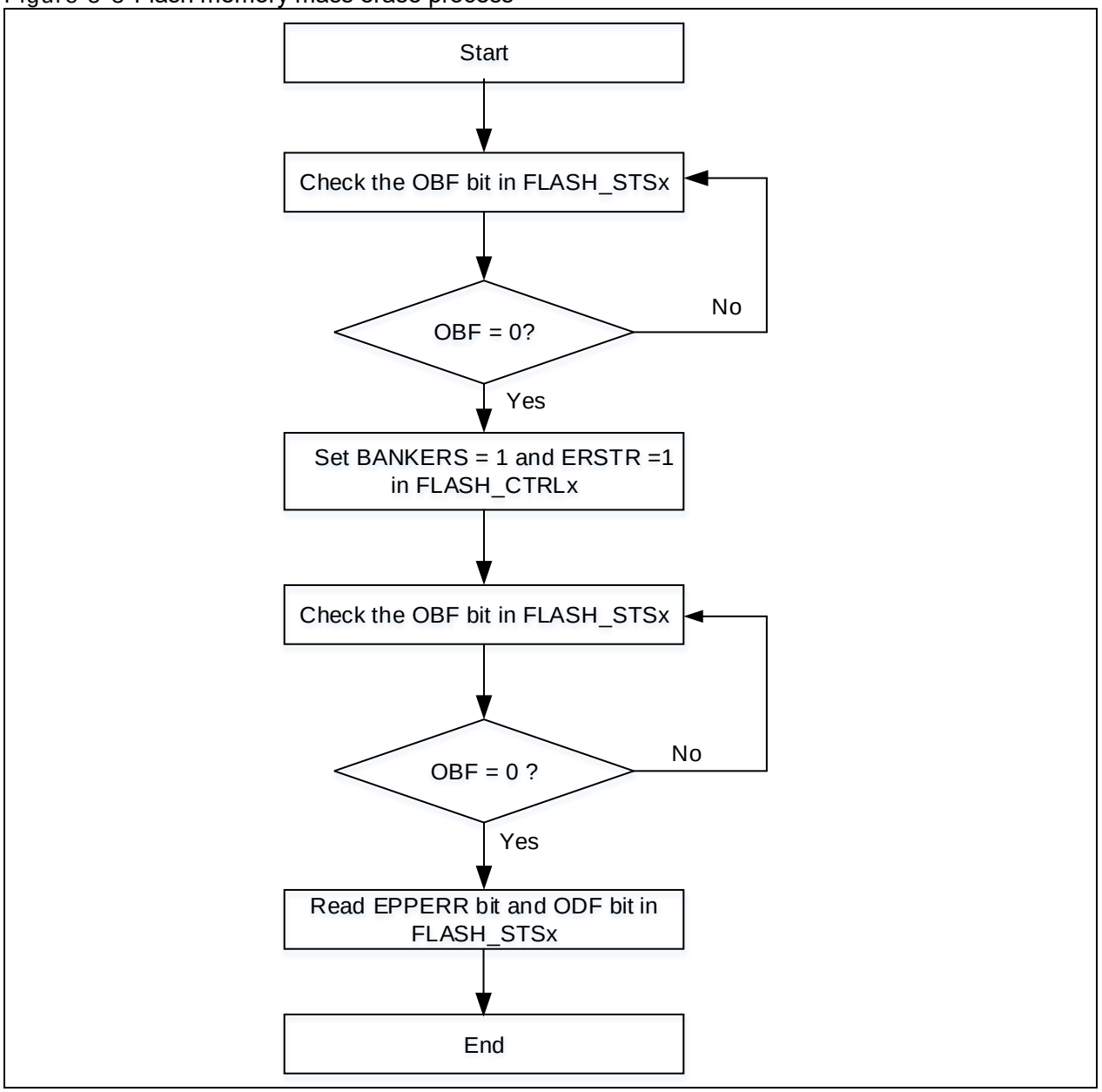
Mass erase function can erase all the Flash memory.

The following process is recommended:

- Check the OBF bit in the FLASH_STSx register to confirm that there is no other programming operation in progress;
- Set the BANKERS and ERSTR bits in the FLASH_CTRLx register to enable mass erase;
- Wait until the OBF bit becomes “0” in the FLASH_STSx register. Read the EPPERR bit and ODF bit in the FLASH_STSx register to verify the erased sectors.

Note: Read operation to the Flash memory during erase will halt CPU until the completion of erase.

Figure 5-3 Flash memory mass erase process



5.2.3 Programming operation

The Flash memory can be programmed with 32 bits, 16 bits or 8 bits at a time.

The following process is recommended:

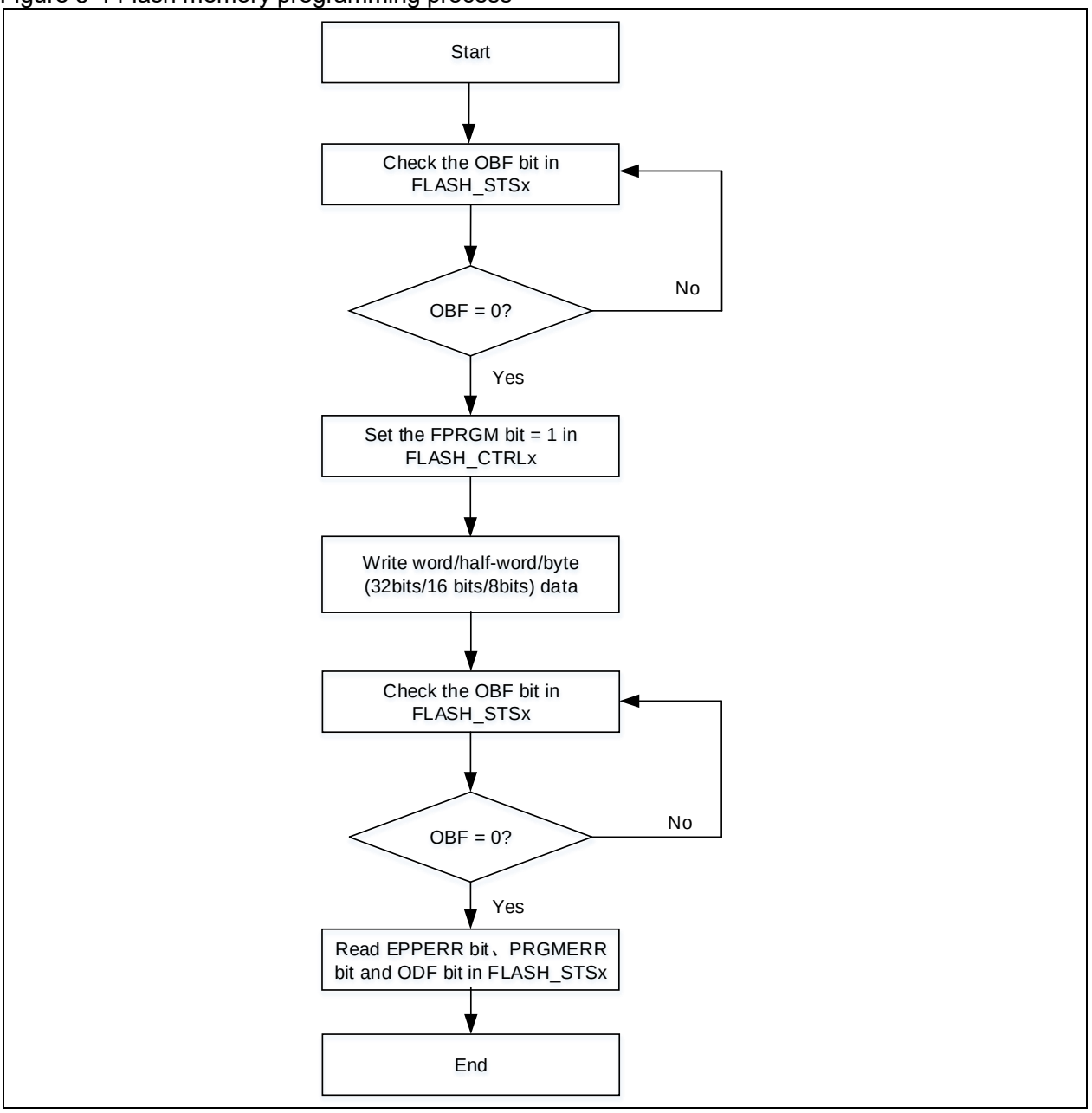
- Check the OBF bit in the FLASH_STSx register to confirm that there is no other programming operation in progress;
- Set the FPRGM bit in the FLASH_CTRLx register, so that the Flash memory programming instructions can be received;
- Write the data (word/half-word/byte) to be programmed to the designated address;
- Wait until the OBF bit in the FLASH_STSx register becomes “0”, read the EPPERR, PRGMERR and ODF bit to verify the programming result.

Note:

1. When the address to be written is not erased in advance, the programming operation is not executed unless the data to be written is all 0. In this case, a programming error is reported by the PRGMERR bit in the FLASH_STSx register.

2. Read operation to the Flash memory during programming will halt CPU until the completion of programming.

Figure 5-4 Flash memory programming process



5.2.4 Read operation

Flash memory can be accessed through AHB bus of the CPU.

5.3 User system data area operation

5.3.1 Unlock/lock

After reset, user system data area is protected, by default. Write and erase operations can be performed only after the Flash memory is unlocked before the unlock operation for the user system data area.

Unlock procedure:

Flash memory block can be unlocked by writing KEY1 (0x45670123) and KEY2 (0xCDEF89AB) to the FLASH_UNLOCK register;

When KEY1 (0x45670123) and KEY2 (0xCDEF89AB) is written to the FLASH_USD_UNLOCK register, the USDULKS bit in the FLASH_CTRL register will be automatically set by hardware, indicating that it supports write/erase operation to the user system data area.

Note: Writing an incorrect key sequence leads to bus error and the Flash memory is also locked until the next reset.

Lock procedure:

User system data area is locked by clearing the USDULKS bit in the FLASH_CTRL register by software.

5.3.2 Erase operation

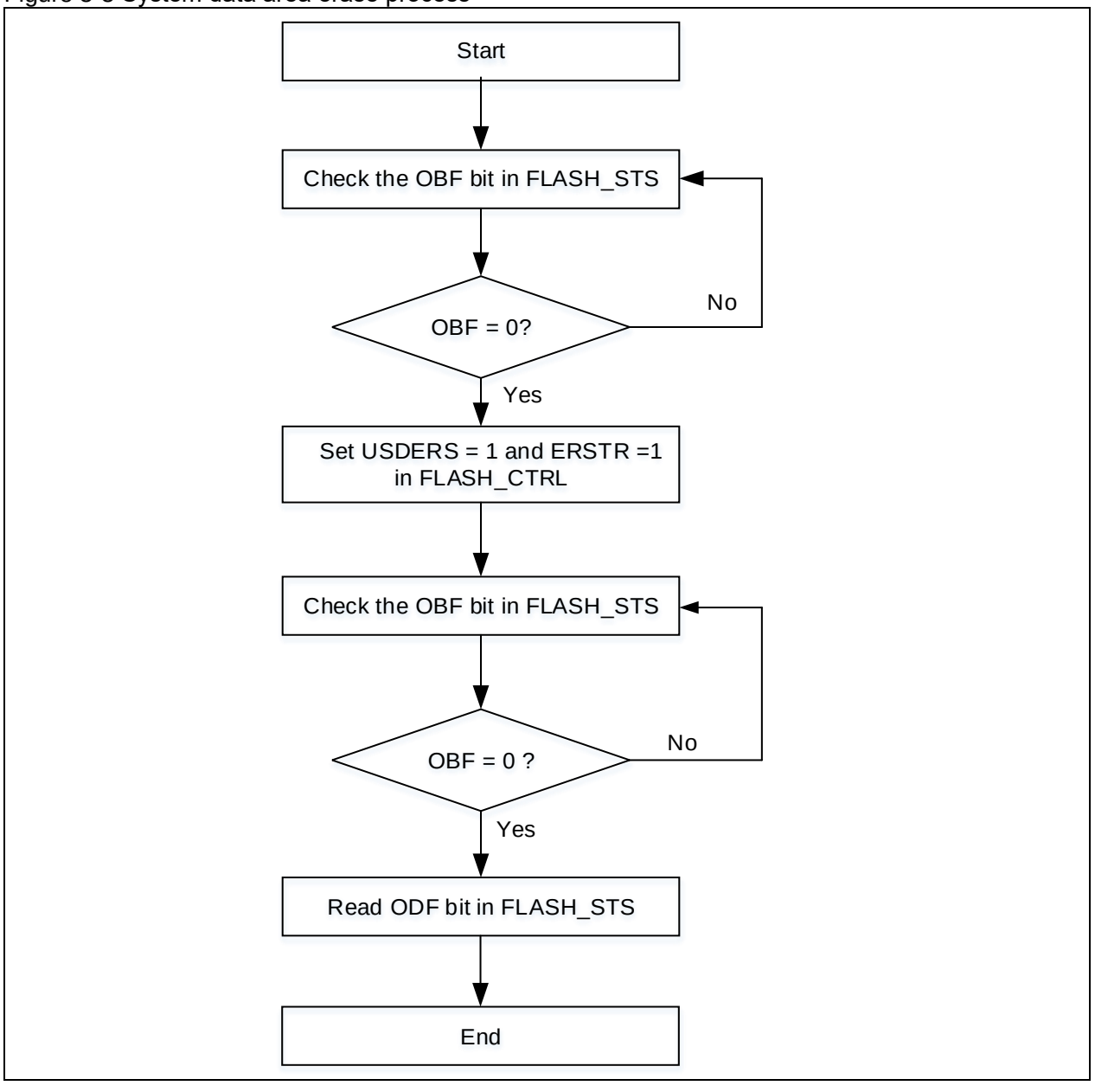
Erase operation must be done before programming. User system data area can perform erase operation independently.

Below should be followed during erase operation:

- Check the OBF bit in the FLASH_STS register to confirm that there is no other programming operation in progress;
- Set the USDERS and ERSTR bits in the FLASH_CTRL register to enable erase operation;
- Wait until the OBF bit becomes “0” in the FLASH_STS register. Read the ODF bit in the FLASH_STSx register to verify the erase result.

Note: Read operation to the Flash memory during programming will halt CPU until the completion of erase.

Figure 5-5 System data area erase process



5.3.3 Programming operation

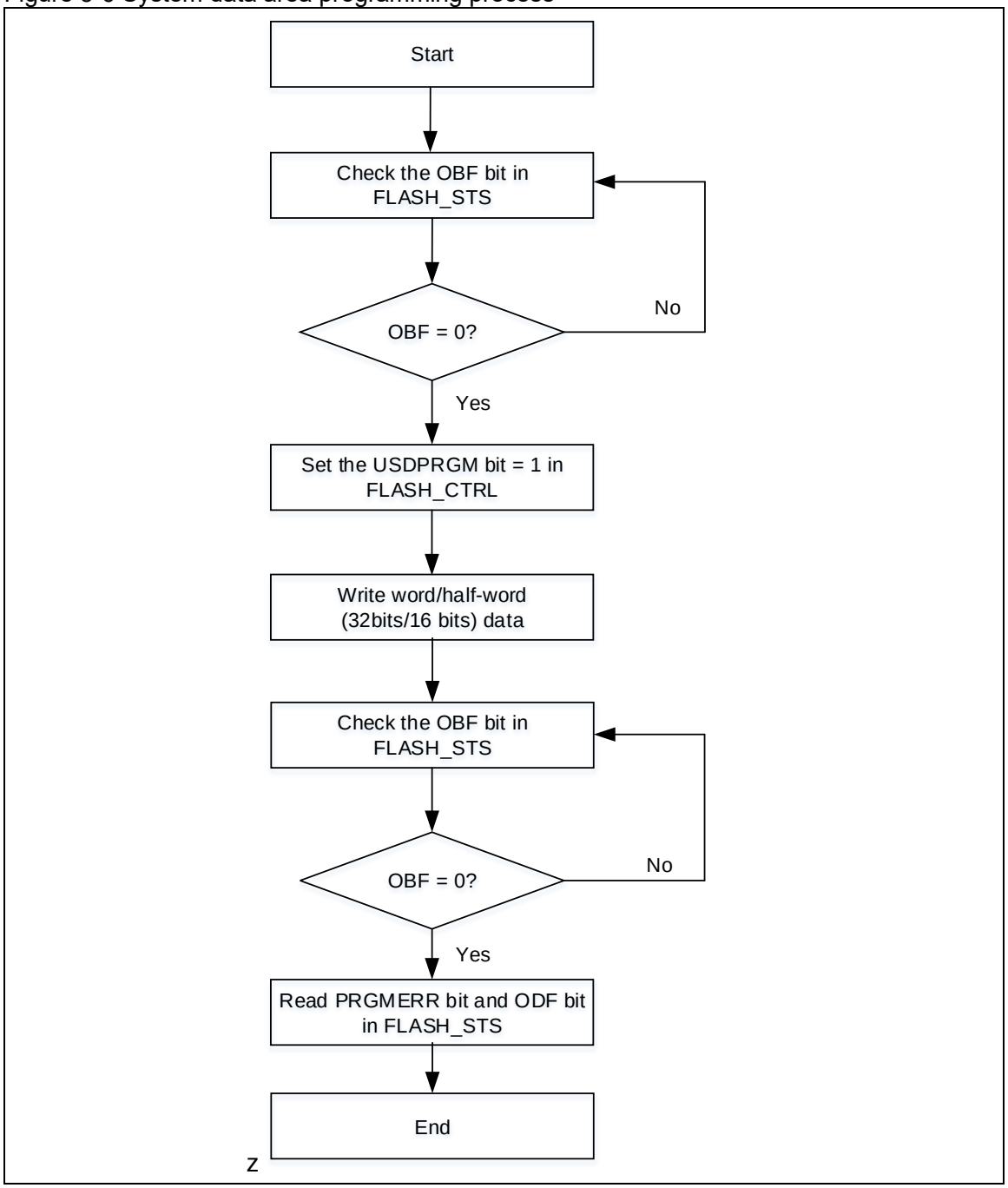
The User system data area can be programmed with 16 bits at a time.

The following process is recommended:

- Check the OBF bit in the FLASH_STS register to confirm that there is no other programming operation in progress;
- Set the USDPRGM bit in the FLASH_CTRL register, so that the programming instructions for the user system data area can be received;
- Write the data (half-word) to be programmed to the designated address;
- Wait until the OBF bit in the FLASH_STS register becomes “0”, read the PRGMERR and ODF bit to verify the programming result.

Note: Read operation to the Flash memory during programming will halt CPU until the completion of programming.

Figure 5-6 System data area programming process



5.3.4 Read operation

User system data area can be accessed through AHB bus of the CPU.

5.4 Flash memory protection

Flash memory includes access and erase/program protection.

5.4.1 Access protection

When the contents in the nFAP and FAP byte are different from 0x5A and 0xA5, the Flash memory will activate access protection after a system reset. In this case, only the Flash program is allowed to read Flash memory data. This read operation is not permitted in debug mode or by booting from non-Flash memory.

When the Flash access is protected, the user can re-erase the system data area, and unlock Flash access protection (switching from protected to unprotected state will trigger mass erase on the Flash memory automatically) by writing 0xA5 to FAP byte, and then perform a system reset. Subsequently, the system data loader will be reloaded with system data and updated with Flash memory access protection disable state (FAP byte)

Note: If the access protection bit is set in debug mode, then the debug mode has to be cleared by POR instead of system reset in order to resume access to Flash memory data.

Table 5-7 shows Flash memory access limits when Flash access protection is enabled.

Table 5-7 Flash memory access limit

Block	Access limits					
	In debug mode or boot from SRAM and boot memory			Boot from main Flash memory		
	Read	Write	Erase	Read	Write	Erase
Main Flash memory	Not allowed		Not allowed (1) (2)	Accessible		
User system data area	Not allowed	Accessible		Accessible		

(1)Main Flash memory is cleared automatically by hardware only when the access protection is disabled;

(2)Only sector erase and block erase are forbidden. Bank 1 and bank 2 and external memory mass erase are not affected.

5.4.2 Erase/program protection

This is used to protect the contents in the Flash memory against inadvertent operation when the program crash occurs.

Erase/program operation is not permitted under one of the following events, and the EPPERR bit is set accordingly:

- The sectors with erase/program protection enabled
- The blocks with erase/program protection enabled
- Bank1, bank2 with erase/program protection enabled
- When the Flash access protection is enabled, the first 4 KB in the main Flash memory will be protected against erase/program automatically,
- Once the Flash access protection is enabled, the main Flash memory is protected against erase/program when it is in debug mode or when it is started from non-main Flash memory.

5.5 Special functions

5.5.1 Security library settings

Security library is a defined area protected by a code in the main memory. This area is only executable but cannot be read (Except for I-Code and D-code buses), written, or deleted, unless a correct code is keyed in. Security library includes instruction security library and data security library.

Advantages of security library:

Security library is protected by codes so that solution providers can program core algorithm into this area;

Security library cannot be read or deleted (including ISP/IAP/SWD) but only executed unless code defined by the solution provider is keyed in;

The rest of the area can be used for secondary development by solution providers;

Solution providers can sell core algorithm with security library function and do not have to develop full solutions for every customer.

Security library helps prevent from deliberate damage or changing terminal application codes.

Note: Security library can only be located in the main Flash memory;

Security library code must be programmed by sector, with its start address aligned with the main memory address;

Only I-Code bus is allowed to read instruction security library;

Only I-Code and D-Code bus are allowed to read the read-only area;

In an attempt of writing or deleting security library code, a warning message will be issued by EPPERR=1 in the FLASH_STS register;

Executing mass erase in the main memory will not erase the security library.

By default, security library setting register is unreadable and write protected. To enable write access to this register, security library should be unlocked first, by writing 0xA35F6D24 to the SLIB_UNLOCK register, and checking the SLIB_ULKF bit in the SLIB_MISC_STS register to verify if it is unlocked successfully and then writing the programmed value into the security library setting register.

Optional CRC check for security library code is based on a sector level.

The steps to enable security library are as follows:

- Check the OBF bit in the FLASH_STS register to ensure that there is no other ongoing programming operation;
- Write 0xA35F6D24 to the SLIB_UNLOCK register to unlock security library;
- Check the SLIB_ULKF bit of SLIB_MISC_STS register to verify that it is unlocked successfully;
- Set a security password in the SLIB_SET_PWD register;
- Set the area to be protected in the SLIB_SET_RANGE0 and SLIB_SET_RANGE1 register;
- Enable the security library information settings by setting the SET_SLIB_STRT bit in the SLIB_SET_RANGE1 register;
- Wait until the OBF bit becomes “0”;
- Program the code to be saved in security library;
- Perform system reset, and then reload security library setting word;
- Read the SLIB_STS0/STS1/STS2 register to verify the security library settings.

Note: Security library should be enabled when the Flash access protection is not activated

Steps to unlock security library:

- Write the previously set security library password to the SLIB_PWD_CLR register;
- Wait until the OBF bit becomes “0”;
- Perform system reset, and then reload security library setting word;
- Read the SLIB_STS0 register to check the security library settings.

Note: Disabling the security library will automatically perform mass erase for the main memory and for security library setting block.

5.6 Flash memory registers

Table 5-8 lists Flash register map and their reset values.

These peripheral registers must be accessed by words (32 bits).

Table 5-8 Flash memory interface—Register map and reset value

Register	Offset	Reset value
FLASH_PSR	0x00	0x0000 0330
FLASH_UNLOCK	0x04	0xXXXX XXXX
FLASH_USD_UNLOCK	0x08	0xXXXX XXXX
FLASH_STS	0x0C	0x0000 0000
FLASH_CTRL	0x10	0x0000 0080
FLASH_ADDR	0x14	0x0000 0000
FLASH_USD	0x1C	0x03FF FFFC
FLASH_EPPS0	0x20	0xFFFF FFFF
FLASH_EPPS1	0x2C	0xFFFF FFFF
FLASH_UNLOCK2	0x44	0xXXXX XXXX
FLASH_STS2	0x4C	0x0000 0000
FLASH_CTRL2	0x50	0x0000 0080
FLASH_ADDR2	0x54	0x0000 0000
FLASH_CONTR	0x58	0x0000 0080
FLASH_DIVR	0x60	0x0000 0022
SLIB_STS2	0xC8	0x0000 FFFF
SLIB_STS0	0xCC	0x0000 0000
SLIB_STS1	0xD0	0xFFFF FFFF
SLIB_PWD_CLR	0xD4	0x0000 0000
SLIB_MISC_STS	0xD8	0x0100 0000
SLIB_SET_PWD	0xDC	0x0000 0000
SLIB_SET_RANGE0	0xE0	0x0000 0000
SLIB_SET_RANGE1	0xE4	0x0000 0000
SLIB_UNLOCK	0xF0	0x0000 0000
FLASH_CRC_CTRL	0xF4	0x0000 0000
FLASH_CRC_CHKR	0xF8	0x0000 0000

5.6.1 Flash performance select register (FLASH_PSR)

Bit	Abbr.	Reset value	Type	Description
Bit 31: 14	Reserved	0x00000	resd	Kept at its default value.
Bit 13	NZW_BST_STS	0x0	ro	Flash non-zero wait area boost status 0: Flash non-zero wait area boost status disabled 1: Flash non-zero wait area boost status enabled
Bit 12	NZW_BST	0x0	rw	Flash non-zero wait area boost 0: Flash non-zero wait area boost disabled 1: Flash non-zero wait area boost enabled Note: Enabling this feature will increase the operating efficiency of non-zero wait state Flash memory, but the system frequency will be limited. Refer to data sheet for more information. The setting code of this bit is located in the zero-wait state Flash.
Bit 11: 0	Reserved	0x330	resd	Kept at its default value.

5.6.2 Flash unlock register (FLASH_UNLOCK)

Only used in Flash memory bank 1.

Bit	Abbr.	Reset value	Type	Description
Bit 31: 0	UKVAL	0xXXXX XXXX	wo	Unlock key value This is used to unlock Flash memory bank 1.

Note: All these bits are write-only, and return 0 when being read.

5.6.3 Flash user system data unlock register (FLASH_USD_UNLOCK)

Bit	Abbr.	Reset value	Type	Description
Bit 31: 0	USD_UKVAL	0xXXXX XXXX	wo	User system data Unlock key value

Note: All these bits are write-only, and return 0 when being read.

5.6.4 Flash status register (FLASH_STS)

Only used in Flash memory bank 1.

Bit	Abbr.	Reset value	Type	Description
Bit 31: 6	Reserved	0x0000000	resd	Kept at its default value
Bit 5	ODF	0x0	rw	Operation done flag This bit is set by hardware when Flash memory operations (program/erase) is completed. It is cleared by writing “1”.
Bit 4	EPPERR	0x0	rw	Erase/program protection error This bit is set by hardware when programming the erase/program- protected Flash memory address. It is cleared by writing “1”.
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2	PRGMERR	0x0	rw	Programming error When the programming address is not “0xFFFF”, this bit is set by hardware. It is cleared by writing “1”.
Bit 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	OBF	0x0	ro	Operation busy flag When this bit is set, it indicates that Flash memory operation is in progress. It is cleared when operation is completed.

5.6.5 Flash control register (FLASH_CTRL)

Only used in Flash memory bank 1.

Bit	Register	Reset value	Type	Description
Bit 31: 13	Reserved	0x00000	resd	Kept at its default value
Bit 12	ODFIE	0x0	rw	Operation done flag interrupt enable 0: Interrupt is disabled; 1: Interrupt is enabled.
Bit 11,8	Reserved	0x0	resd	Kept its default value
Bit 10	ERRIE	0x0	rw	Error interrupt enable This bit enables EPPERR or PROGERR interrupt. 0: Interrupt is disabled; 1: Interrupt is enabled.
Bit 9	USDULKS	0x0	rw	User system data unlock success This bit is set by hardware when the user system data is unlocked properly, indicating that erase/program operation to the user system data is allowed. This bit is cleared by writing “0”, which will re-lock the user system data area.
Bit 7	OPLK	0x1	rw	Operation lock This bit is set by default, indicating that Flash memory is protected against operations. This bit is cleared by

				hardware after unlock, indicating that erase/program operation to Flash memory is allowed. Writing “1” can re-lock Flash memory operations.
Bit 6	ERSTR	0x0	rw	Erase start An erase operation is triggered when this bit is set. This bit is cleared by hardware after the completion of the erase operation.
Bit 5	USDERS	0x0	rw	User system data erase It indicates the user system data erase.
Bit 4	USDPRGM	0x0	rw	User system data program It indicates the user system data program.
Bit 3	BLKERS	0x0	rw	Block erase It indicates block erase operation.
Bit 2	BANKERS	0x0	rw	Bank erase It indicates bank erase operation.
Bit 1	SECERS	0x0	rw	Sector erase It indicates sector erase operation.
Bit 0	FPRGM	0x0	rw	Flash program It indicates Flash program operation.

5.6.6 Flash address register (FLASH_ADDR)

Only used in Flash memory bank 1.

Bit	Register	Reset value	Type	Description
Bit 31: 0	FA	0x0000 0000	wo	Flash address Select the address of the blocks/sectors to be erased.

5.6.7 User system data register (FLASH_USD)

Bit	Register	Reset value	Type	Description
Bit 31: 26	Reserved	0x00	resd	Kept at its default value
Bit 25: 18	USER_D1	0xFF	ro	User data 1
Bit 17: 10	USER_D0	0xFF	ro	User data 0
Bit 9: 2	SSB	0xFF	ro	System setting byte Includes the system setting bytes in the loaded user system data area Bit 9: Unused Bit 8: nSTDBY_WDT Bit 7: nDEPSLP_WDT Bit 6: Unused Bit 5: BTOPT Bit 4: nSTDBY_RST Bit 3: nDEPSLP_RST Bit 2: nWDT_ATO_EN
Bit 1	FAP	0x0	ro	Flash access protection Access to Flash memory is not allowed when this bit is set.
Bit 0	USDERR	0x0	ro	User system data error When this bit is set, it indicates that certain byte does not match its inverse code in the user system data area. At this point, this byte and its inverse code will be forced to 0xFF when being read.

5.6.8 Erase/program protection status register0 (FLASH_EPPS0)

Bit	Register	Reset value	Type	Description
Bit 31: 0	EPPS	0xFFFF FFFF	ro	Erase/Program protection status This register reflects the erase/program protection byte status in the loaded user system data.

5.6.9 Erase/program protection status register1 (FLASH_EPPS1)

Bit	Register	Reset value	Type	Description
Bit 31: 0	EPPS	0xFFFF FFFF	ro	Erase/Program protection status This register reflects the erase/program protection byte status in the loaded user system data.

5.6.10 Flash unlock register2 (FLASH_UNLOCK2)

Only used in Flash memory bank 2.

Bit	Register	Reset value	Type	Description
Bit 31: 0	UKVAL	0xXXXX XXXX	wo	Unlock key value This register is used to unlock Flash memory bank 2.

Note: All these bits are write-only, and return 0 when being read.

5.6.11 Flash status register2 (FLASH_STS2)

Only used in Flash memory bank 2.

Bit	Register	Reset value	Type	Description
Bit 31: 6	Reserved	0x0000000	resd	Kept at its default value
Bit 5	ODF	0x0	rw	Operation done flag This bit is set by hardware when Flash memory operations (program/erase) is completed. It is cleared by writing “1”.
Bit 4	EPPERR	0x0	rw	Erase/Program protection error This bit is set by hardware when programming the erase/program-protected Flash memory address. It is cleared by writing “1”
Bit 3	Reserved	0x0	resd	Kept at its default value
Bit 2	PRGMERR	0x0	rw	Program error When the programming address is not “0xFFFF”, this bit is set by hardware. It is cleared by writing “1”
Bit 1	Reserved	0x0	resd	Kept at its default value
Bit 0	OBF	0x0	ro	Operation busy flag When this bit is set, it indicates that Flash memory operation is in process. It is cleared when operation is completed.

5.6.12 Flash control register2 (FLASH_CTRL2)

Only used in Flash memory bank 2.

Bit	Register	Reset value	Type	Description
Bit 31: 13	Reserved	0x00000	resd	Kept its default value
Bit 12	ODFIE	0x0	rw	Operation done flag interrupt enable 0: Interrupt is disabled; 1: Interrupt is enabled.
Bit 11	Reserved	0x0	resd	Kept its default value
Bit 10	ERRIE	0x0	rw	Error interrupt enable This bit enables EPPERR or PROGERR interrupt. 0: Interrupt is disabled; 1: Interrupt is enabled.

Bit 9,8	Reserved	0x0	resd	Kept at its default value
Bit 7	OPLK	0x1	rw	Operation lock This bit is set by default, indicating that Flash memory is protected against operations. This bit is cleared by hardware after unlock, indicating that erase/program operation to Flash memory is allowed. Writing “1” can re-lock Flash memory operations.
Bit 6	ERSTR	0x0	rw	Erase start An erase operation is triggered when this bit is set. This bit is cleared by hardware after the completion of the erase operation.
Bit 5,4	Reserved	0x0	resd	Kept its default value
Bit 3	BLKERS	0x0	rw	Block erase It indicates block erase operation.
Bit 2	BANKERS	0x0	rw	Bank erase It indicates bank erase operation.
Bit 1	SECERS	0x0	rw	Sector erase It indicates sector erase operation.
Bit 0	FPRGM	0x0	rw	Flash program It indicates Flash program operation.

5.6.13 Flash address register2 (FLASH_ADDR2)

Only used in Flash memory bank 2.

Bit	Register	Reset value	Type	Description
Bit 31: 0	FA	0x0000 0000	wo	Flash address Select the address of the blocks/sectors to be erased.

5.6.14 Flash continue read register (FLASH_CONTR)

Bit	Register	Reset value	Type	Description
Bit 31: 0	FCONTR_EN	0x0	rw	Flash continue read enable 0: Flash continue read mode disabled 1: Flash continue read mode enabled Setting this bit will enable the CPU to read Flash at a faster speed, but will also increase power consumption of Flash at the same time.
Bit 30: 0	Reserved	0x0000 0080	resd	Kept at its default value.

5.6.15 Flash divider register (FLASH_DIVR)

Bit	Register	Reset value	Type	Description
Bit 31: 6	Reserved	0x0000000	resd	Kept at its default value.
Bit 5: 4	FDIV_STS	0x2	ro	Flash divider status This field indicates the division relationship between the current Flash interface and HCLK. 0: HCLK/2 1: HCLK/3 Others: HCLK/4
Bit 3: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1: 0	FDIV	0x2	rw	Flash divider This field is used to configure the division relationship between the Flash interface and HCLK. 0: HCLK/2 1: HCLK/3 Others: HCLK/4 Note: The Flash interface clock is not greater than 104MHz.

5.6.16 Flash security library status register2 (SLIB_STS2)

Only used in Flash security library.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 0	SLIB_INST_SS	0xFFFF	ro	sLib instruction start sector 0: Sector 0 1: Sector 1 2: Sector 2 … 0xFFFF: Non-sLib instruction

5.6.17 Flash security library status register0 (SLIB_STS0)

Only used in Flash security library.

Bit	Register	Reset value	Type	Description
Bit 31: 4	Reserved	0x0000000	resd	Kept at its default value
Bit 3	SLIB_ENF	0x0	ro	SLIB_ENF: sLib enable flag When this bit is set, it indicates that the main Flash memory is partially or completely (depending on the setting of SLIB_STS1) used as security library code.
Bit 2: 0	Reserved	0x0	resd	Keep at its default value

5.6.18 Flash security library status register1 (SLIB_STS1)

Only used in Flash security library.

Bit	Register	Reset value	Type	Description
Bit 31: 16	SLIB_ES	0xFFFF	ro	Security library end sector 0: Sector 0 1: Sector 1 2: Sector 2 …
Bit 15: 0	SLIB_SS	0xFFFF	ro	Security library start sector 0: Sector 0 1: Sector 1 2: Sector 2 …

5.6.19 Flash security library password clear register (SLIB_PWD_CLR)

Only used in Flash security library.

Bit	Register	Reset value	Type	Description
Bit 31:0	SLIB_PCLR_VAL	0x0000 0000	wo	Security library password clear value Entering correct security library password will unlock security library functions. The write status of this register is reflected in the bit 0 and bit 1 in the SLIB_MISC_STS register.

5.6.20 Security library additional status register (SLIB_MISC_STS)

Only used in Flash security library.

Bit	Register	Reset value	Type	Description
Bit 31:25	Reserved	0x00	resd	Kept at its default value
Bit 24: 16	SLIB_RCNT	0x100	ro	Security library remaining count It is decremented from 256 to 0.
Bit 15: 3	Reserved	0x0000	resd	Kept at its default value
Bit 2	SLIB_ULKF	0x0	ro	Security library unlock flag

				When this bit is set, it indicates that sLib-related setting registers can be configured.
Bit 1	SLIB_PWD_OK	0x0	ro	Security library password ok This bit is set by hardware when the password is correct.
Bit 0	SLIB_PWD_ERR	0x0	ro	Security library password error This bit is set by hardware when the password is incorrect and the setting value of the password clear register is different from 0xFFFF FFFF. Note: When this bit is set, the hardware will no longer agree to re-program the password clear register until the next reset.

5.6.21 Security library password setting register (SLIB_SET_PWD)

Only used in Flash security library.

Bit	Register	Reset value	Type	Description
Bit 31: 0	SLIB_PSET_VAL	0x0000 0000	wo	Security library password setting value Note: This register can be written only after unlocking security library lock. It is used to set up the startup password of security library. Values of 0xFFFF_FFFF and 0x0000_0000 are invalid.

Note: All these bits are write-only, and return 0 when being read.

5.6.22 Security library address setting register0 (SLIB_SET_RANGE0)

Only used for Flash security library address setting.

Bit	Register	Reset value	Type	Description
Bit 31: 16	SLIB_ES_SET	0x000	wo	Security library end sector setting Theses bits are used to set the security library end sector. 0: Sector 0 1: Sector 1 2: Sector 2 …
Bit 10: 0	SLIB_SS_SET	0x000	wo	Security library start sector setting These bits are used to set the security library start sector. 0: Sector 0 1: Sector 1 2: Sector 2 …

Note: All these bits are write-only, and return 0 when being read.

(1) This register can be written only after unlocking security library lock.

(2) Being out of the Flash address range is an invalid setting.

5.6.23 Security library address setting register1 (SLIB_SET_RANGE1)

Only used for Flash security library address setting.

Bit	Register	Reset value	Type	Description
Bit 31	SET_SLIB_STRT	0x0	wo	Setting sLib start This bit is used to enable security library information settings. It is cleared by hardware automatically when the security library configuration is enabled by hardware.
Bit 30: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 0	SLIB_ISS_SET	0x0000	wo	Security library instruction start sector setting These bits are used to set the security library instruction

				start sector. 0: Sector 0 1: Sector 1 2: Sector 2 … 0xFFFF: Non-sLib instruction

5.6.24 Security library unlock register (SLIB_UNLOCK)

Only used for Flash security library unlock setting.

Bit	Register	Reset value	Type	Description
Bit 31: 0	SLIB_UKVAL	0x0000 0000	wo	Security library unlock key value Fixed key value is 0xA35F_6D24, used for security library setting register unlock

Note: All these bits are write-only, and return 0 when being read.

5.6.25 Flash CRC calibration control register (FLASH_CRC_CTRL)

Only used in main Flash memory.

Bit	Register	Reset value	Type	Description
Bit 31	CRC_STRT	0x0	wo	CRC start Set this bit to enable user code or security library code CRC check. This bit is cleared automatically after the hardware enables CRC.
Bit 30: 24	Reserved	0x00	wo	Kept at its default value
Bit 23: 12	CRC_SN	0x000	wo	CRC sector number Set the number of the CRC calibration, in terms of sectors.
Bit 11: 0	CRC_SS	0x000	wo	CRC calibration start sector Set the start sector for this CRC calibration. 0x0: Sector 0 0x1: Sector 1 …

Note: All these bits are write-only, and return 0 when being read.

5.6.26 Flash CRC check result register (FLASH_CRC_CHKR)

Only used in Flash or security library.

Bit	Register	Reset value	Type	Description
Bit 31: 0	CRC_CHKR	0x0000 0000	ro	CRC check result

Note: All these bits are write-only, and return 0 when being read.

6 GPIOs and IOMUX

6.1 Introduction

AT32F435 series supports up to 116 bidirectional I/O pins, which are grouped as eight categories, namely PA0-PA15, PB0-PB15, PC0-PC15, PD0-PD15, PE0-PE15, PF0-PF15, PG0-PG15 and PH0-PH3. Each of these pins features communication, control and data collection. In addition, their main features also include:

- Supports general-purpose I/O (GPIO) or multiplexed function I/O (IOMUX)
- Each pin can be configured by software as floating input, pull-up/pull-down input, analog input/output, push-pull/open-drain output, multiplexed push-pull/open-drain output
- Each pin with individual weak pull-up/pull-down capability
- Each pin's output drive capability is configurable by software
- Each pin can be configured as external interrupt input
- Each pin can be locked

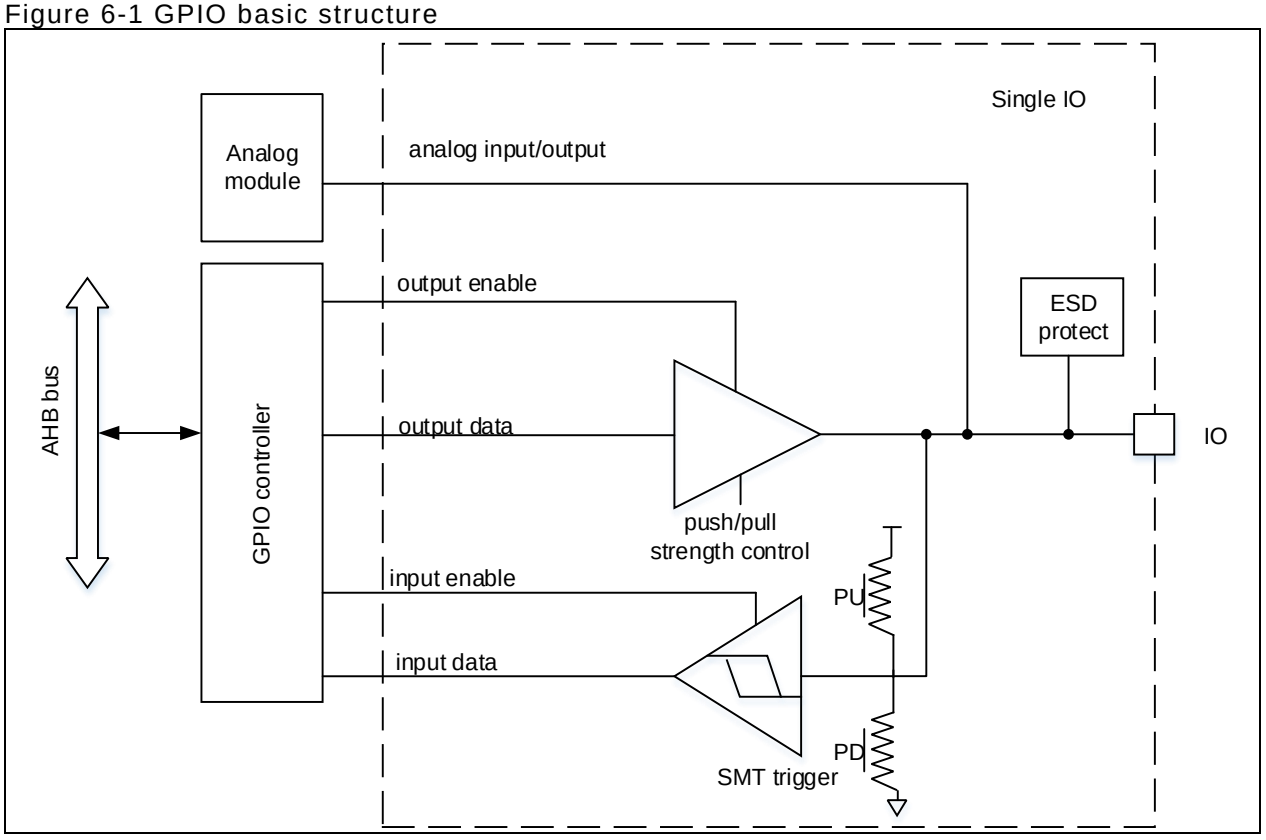
6.2 Function overview

6.2.1 GPIO structure

Each of the GPIO pins can be configured by software as four input modes (floating, pull-up/pull-down and analog input) and four output modes (open-drain, push-pull, alternate function push-pull/open-drain output)

Each I/O port bit can be programmed freely. However, I/O port registers must be accessed by half words or bytes.

Figure 6-1 GPIO basic structure



6.2.2 GPIO reset status

After power-on or system reset, all pins are configured as floating input mode except JATG-related pins. JTAG pin configuration are as follows:

- PA15/JTDI, PA13/JTMS and PB4/JNTRST in multiplexed pull-up mode;

- PA14/JTCK in multiplexed pull-down mode;
- PB3/TDO in multiplexed mode without pull-up/pull-down capability

6.2.3 General-purpose input configuration

Mode	IOMC	PUPD
Floating input	00	00
Pull-down input		10
Pull-up input		01

When I/O port is configured as input:

- Get I/O states by reading the input data register.
- Floating input, pull-up/pull-down input is configurable
- Schmitt-trigger input is activated.
- Output is disabled.

Note: In floating input mode, it is recommended to set the unused pins as analog input mode in order to avoid leakage caused by interference from unused pins in a complex environment.

6.2.4 Analog input/output configuration

Mode	IOMC	PUPD
Analog input/output	11	Unused

When I/O port is configured as analog input:

- Schmitt-trigger input is disabled.
- Digital input/output is disabled.
- Without any pull-up/pull-down resistor.

6.2.5 General-purpose output configuration

Mode	IOMC	OM	HDRV	ODRV[1: 0]	PUPD
Push-Pull without pull-up/pull-down	01	0	000: Output mode, normal sourcing/sinking strength 001: Output mode, large sourcing/sinking strength 010: Output mode, normal sourcing/sinking strength 011: Output mode, normal sourcing/sinking strength 1xx: Output mode, Maximum sourcing/sinking strength		00 or 11
Push-Pull with pull-up	01	0			01
Push-Pull with pull-down	01	0			10
Open-Drain without pull-up/pull-down	01	1			00 or 11
Open-Drain with pull-up	01	1			01
Open-Drain with pull-down	01	1			10

When I/O port is configured as output:

- Schmitt-trigger input is enabled
- Output through output register
- In open-drain mode, forced output 0, and use pull-up resistor to output 1
- In push-pull mode, output register is used to output 0/1
- GPIO set/clear register is used to set/clear the corresponding GPIO output data registers

Note: If both IOCB and IOSB bits are set in the GPIO set/clear register, the IOSB takes priority.

6.2.6 I/O port protection

Locking mechanism can freeze the I/O configuration for the purpose of protection. When LOCK is applied to a port bit, its configuration cannot be modified until the next reset or power on.

6.2.7 IOMUX structure

Several peripheral functions can be mapped on each IO pin. Peripheral input/output corresponding to an I/O pin is selected through IOMUX input/output table. Each I/O pin has up to 16 IOMUX mapping options for flexible selection, configured through the GPIOx_MUXL (for pin 0 to 7) and GPIOx_MUXH (for pin 8 to 15) registers.

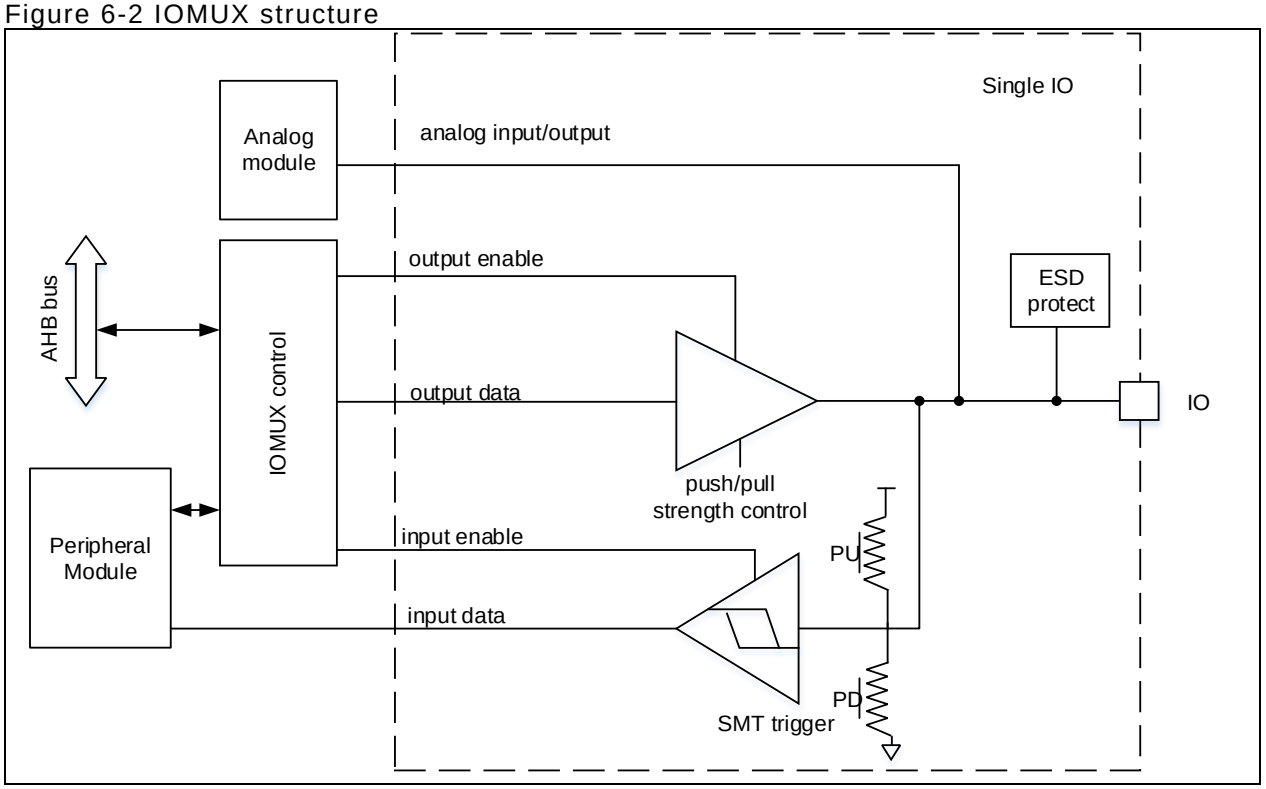
Each I/O pin is connected to only one peripheral's pin by setting the GPIOx_MUXL or GPIOx_MUXH register so that there can be no conflict between peripherals sharing the same pin.

While being used as multiplexed function input, the I/O port should be configured as input modes (floating, pull-up and pull-down input)

To enable multiplexed function output, the port is configured as multiplexed push-pull or open-drain mode by setting GPIOx_CFGR or GPIOx_OMODE register. In this case, the pins are disconnected from GPIO controller, and controlled by IOMUX controller, instead.

To achieve bidirectional multiplexed function, the port needs to be configured as multiplexed function mode (push-pull or open-drain), controlled by IOMUX controller.

Figure 6-2 IOMUX structure



6.2.8 Multiplexed function pull-up/down configuration

Mode	IOMC	PUPD
Multiplexed function floating	10	00
Multiplexed function pull-down		10
Multiplexed function pull-up		01

When an I/O port is configured as input:

- Get an I/O pin state by reading input data registers
- The pin be configured as floating input, pull-up or pull-down input
- Schmitt-trigger input is activated.

- GPIO pin output is disabled.

6.2.9 IOMUX input/output

The multiplexed function of each IO port line is configured through the GPIOx_MUXL (for pin 0 to 7) or GPIOx_MUXH (for pin 8 to 15) register.

Table 6-1 Port A multiplexed function configuration with GPIOA_MUX* register

Pin name	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PA0		TMR2_CH1 TMR2_EXT	TMR5_CH1	TMR8_EXT	I2C2_SCL			USART2_CTS
PA1		TMR2_CH2	TMR5_CH2		I2C2_SDA	SPI4_MOSI I2S4_SD		USART2_RTS_DE
PA2		TMR2_CH3	TMR5_CH3	TMR9_CH1				USART2_TX
PA3		TMR2_CH4	TMR5_CH4	TMR9_CH2		I2S2_MCK		USART2_RX
PA4						SPI1_CS I2S1_WS	SPI3_CS I2S3_WS	USART2_CK
PA5		TMR2_CH1 TMR2_EXT		TMR8_CH1C		SPI1_SCK I2S1_CK		
PA6		TMR1_BRK	TMR3_CH1	TMR8_BRK		SPI1_MISO	I2S2_MCK	USART3_CTS
PA7		TMR1_CH1C	TMR3_CH2	TMR8_CH1C		SPI1_MOSI I2S1_SD		
PA8	CLKOUT1	TMR1_CH1			I2C3_SCL			USART1_CK
PA9		TMR1_CH2			I2C3_SMBA	SPI2_SCK I2S2_CK		USART1_TX
PA10		TMR1_CH3				SPI2_MOSI I2S2_SD	I2S4_MCK	USART1_RX
PA11		TMR1_CH4			I2C2_SCL	SPI2_CS I2S2_WS	SPI4_MISO	USART1_CTS
PA12		TMR1_EXT			I2C2_SDA	SPI2_MISO		USART1_RTS_DE
PA13	JTMS SWDIO	IR_OUT					SPI3_MISO	
PA14	JTCK SWCLK						SPI3_MOSI I2S3_SD	
PA15	JTDI	TMR2_CH1 TMR2_EXT				SPI1_CS I2S1_WS	SPI3_CS I2S3_WS	USART1_TX

Pin name	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PA0	UART4_TX			EMAC_MII_CRS				EVENTOUT
PA1	UART4_RX	QSPI1_IO3		EMAC_MII_RX_CLK EMAC_RMII_REF_CLK				EVENTOUT
PA2			SDIO2_CK	EMAC_MDIO			XMC_D4	EVENTOUT
PA3		QSPI2_IO3	SDIO2_CMD	EMAC_MII_COL			XMC_D5	EVENTOUT
PA4	USART6_TX		SDIO2_D4	SDIO2_D0	OTG2_SOF	DVP_HSYNC	XMC_D6	EVENTOUT
PA5	USART6_RX	QSPI2_IO2	SDIO2_D5	SDIO2_D1			XMC_D7	EVENTOUT
PA6		TMR13_CH1	QSPI1_IO0	SDIO2_D2	SDIO1_CMD	DVP_PIXCLK	SDIO2_D6	EVENTOUT
PA7		TMR14_CH1	QSPI1_IO1	EMAC_MII_RX_DV EMAC_RMII_CRS_DV	XMC_SDNWE	SDIO2_D3	SDIO2_D7	EVENTOUT
PA8	USART2_TX		OTG1_SOF		SDIO1_D1		XMC_A4	EVENTOUT
PA9	I2C1_SCL		OTG1_VBUS		SDIO1_D2	DVP_D0		EVENTOUT
PA10	I2C1_SDA		OTG1_ID			DVP_D1		EVENTOUT
PA11	USART6_TX	CAN1_RX	OTG1_D-			DVP_D2		EVENTOUT
PA12	USART6_RX	CAN1_TX	OTG1_D+			DVP_D3		EVENTOUT
PA13			OTG1_OE					EVENTOUT
PA14	USART2_TX							EVENTOUT
PA15	USART2_RX	QSPI2_IO1	QSPI1_IO2		XMC_NE2 XMC_NCE3			EVENTOUT

Table 6-2 Port B multiplexed function configuration with GPIOB_MUX* register

Pin	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PB0		TMR1_CH2C	TMR3_CH3	TMR8_CH2C		I2S1_MCK	USART2_RX	SPI3_MOSI I2S3_SD
PB1		TMR1_CH3C	TMR3_CH4	TMR8_CH3C			SPI2_SCK I2S2_CK	
PB2		TMR2_CH4	TMR20_CH1		I2C3_SMBA			SPI3_MOSI I2S3_SD
PB3	JTDO SWO	TMR2_CH2			I2C2_SDA	SPI1_SCK I2S1_CK	SPI3_SCK I2S3_CK	USART1_RX
PB4	JNTRST		TMR3_CH1		I2C3_SDA	SPI1_MISO	SPI3_MISO	I2S3_SDEXT
PB5			TMR3_CH2		I2C1_SMBA	SPI1_MOSI I2S1_SD	SPI3_MOSI I2S3_SD	USART1_CK
PB6			TMR4_CH1		I2C1_SCL	I2S1_MCK	SPI4_CS I2S4_WS	USART1_TX
PB7			TMR4_CH2	TMR8_BRK	I2C1_SDA		SPI4_SCK I2S4_CK	USART1_RX
PB8		TMR2_CH1 TMR2_EXT	TMR4_CH3	TMR10_CH1	I2C1_SCL		SPI4_MISO	
PB9	IR_OUT	TMR2_CH2	TMR4_CH4	TMR11_CH1	I2C1_SDA	SPI2_CS I2S2_WS	SPI4_MOSI I2S4_SD	I2C2_SDA
PB10		TMR2_CH3			I2C2_SCL	SPI2_SCK I2S2_CK	I2S3_MCK	USART3_TX
PB11		TMR2_CH4	TMR5_CH4		I2C2_SDA			USART3_RX
PB12		TMR1_BRK	TMR5_CH1		I2C2_SMBA	SPI2_CS I2S2_WS	SPI4_CS I2S4_WS	SPI3_SCK I2S3_CK
PB13		TMR1_CH1C			I2C3_SMBA	SPI2_SCK I2S2_CK	SPI4_SCK I2S4_CK	I2C3_SCL
PB14		TMR1_CH2C		TMR8_CH2C	I2C3_SDA	SPI2_MISO	I2S2_SDEXT	USART3_RTS_DE
PB15	ERTC_REFI N	TMR1_CH3C		TMR8_CH3C	I2C3_SCL	SPI2_MOSI I2S2_SD		

Pin	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PB0	USART3_CK	QSPI2_IO0	QSPI1_IO0	EMAC_MII_RXD2	SDIO1_D1			EVENTOUT
PB1	USART3_RTS_DE	QSPI1_SCK	QSPI2_SCK	EMAC_MII_RXD3	SDIO1_D2			EVENTOUT
PB2		QSPI1_SCK			SDIO1_CK			EVENTOUT
PB3	UART7_RX		QSPI1_IO3			DVP_D4		EVENTOUT
PB4	UART7_TX				SDIO1_D0	DVP_D5		EVENTOUT
PB5	UART5_RX	CAN2_RX		EMAC_PPS_OUT	XMC_SDCKE1	DVP_D10	SDIO1_D3	EVENTOUT
PB6	UART5_TX	CAN2_TX	QSPI1_CS		XMC_SDCS1	DVP_D5	SDIO1_D0	EVENTOUT
PB7		QSPI2_IO1			XMC_NADV	DVP_VSYNC	SDIO1_D0	EVENTOUT
PB8	UART5_RX	CAN1_RX	QSPI2_CS	EMAC_MII_TXD3	SDIO1_D4	DVP_D6		EVENTOUT
PB9	UART5_TX	CAN1_TX	QSPI1_CS		SDIO1_D5	DVP_D7		EVENTOUT
PB10		QSPI1_CS	QSPI1_IO1	EMAC_MII_RX_ER	SDIO1_D7		XMC_NOE	EVENTOUT
PB11			QSPI1_IO0	EMAC_MII_TX_EN EMAC_RMII_TX_EN				EVENTOUT
PB12	USART3_CK	CAN2_RX		EMAC_MII_TXD0 EMAC_RMII_TXD0	OTG2_ID		XMC_D13	EVENTOUT
PB13	USART3_CTS	CAN2_TX		EMAC_MII_TXD1 EMAC_RMII_TXD1	OTG2_VBUS			EVENTOUT
PB14		TMR12_CH1			OTG2_D-	SDIO1_D6	XMC_D0	EVENTOUT
PB15		TMR12_CH2			OTG2_D+	SDIO1_CK		EVENTOUT

Table 6-3 Port C multiplexed function configuration with GPIOC_MUX* register

Pin	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PC0					I2C3_SCL			
PC1					I2C3_SDA	SPI3_MOSI I2S3_SDEXT		SPI2_MOSI I2S2_SDEXT
PC2			TMR20_CH2			SPI2_MISO	I2S2_SDEXT	
PC3						SPI2_MOSI I2S2_SDEXT		
PC4				TMR9_CH 1		I2S1_MCK		USART3_TX
PC5				TMR9_CH 2	I2C1_SMBA			USART3_RX
PC6			TMR3_CH1	TMR8_CH 1	I2C1_SCL	I2S2_MCK		
PC7			TMR3_CH2	TMR8_CH 2	I2C1_SDA	SPI2_SCK I2S2_CK	I2S3_MCK	
PC8			TMR3_CH3	TMR8_CH 3		I2S4_MCK	TMR20_CH3	UART8_TX
PC9	CLKOUT2		TMR3_CH4	TMR8_CH 4	I2C3_SDA			UART8_RX
PC10			TMR5_CH2				SPI3_SCK I2S3_CK	USART3_TX
PC11			TMR5_CH3			I2S3_SDEXT	SPI3_MISO	USART3_RX
PC12				TMR11_C H1	I2C2_SDA		SPI3_MOSI I2S3_SDEXT	USART3_CK
PC13								
PC14								
PC15								

Pin	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PC0	UART7_TX		SDIO2_D0		XMC_SDNWE			EVENTOUT
PC1	UART7_RX		SDIO2_D1	EMAC_MDC				EVENTOUT
PC2	UART8_TX		SDIO2_D2	EMAC_MII_TXD2	XMC_SDCS0		XMC_NWE	EVENTOUT
PC3	UART8_RX	QSPI2_IO1	SDIO2_D3	EMAC_MII_TX_CLK	XMC_SDCKE0		XMC_A0	EVENTOUT
PC4			QSPI1_IO2	EMAC_MII_RXD0 EMAC_RMII_RXD0	XMC_SDCS0	SDIO2_CK	XMC_NE4	EVENTOUT
PC5			QSPI1_IO3	EMAC_MII_RXD1 EMAC_RMII_RXD1	XMC_SDCKE0	SDIO2_CMD	XMC_NOE	EVENTOUT
PC6	USART6_TX		XMC_A0		SDIO1_D6	DVP_D0	XMC_D1	EVENTOUT
PC7	USART6_RX		XMC_A1		SDIO1_D7	DVP_D1		EVENTOUT
PC8	USART6_CK	QSPI1_IO2	XMC_A2		SDIO1_D0	DVP_D2		EVENTOUT
PC9		QSPI1_IO0	XMC_A3	OTG2_OE	SDIO1_D1	DVP_D3		EVENTOUT
PC10	UART4_TX	QSPI1_IO1			SDIO1_D2	DVP_D8		EVENTOUT
PC11	UART4_RX	QSPI1_CS			SDIO1_D3	DVP_D4	XMC_D2	EVENTOUT
PC12	UART5_TX				SDIO1_CK	DVP_D9	XMC_D3	EVENTOUT
PC13								EVENTOUT
PC14								EVENTOUT
PC15								EVENTOUT

Table 6-4 Port D multiplexed function configuration with GPIOD_MUX* register

Pin	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PD0						SPI4_MISO	SPI3_MOSI I2S3_SDEXT	SPI2_CS I2S2_WS
PD1							SPI2_SCK I2S2_CK	SPI2_CS I2S2_WS
PD2			TMR3_EXT					USART3_RTS_DE
PD3						SPI2_SCK I2S2_CK	SPI2_MISO	USART2_CTS
PD4							SPI2_MOSI I2S2_SDEXT	USART2_RTS_DE
PD5								USART2_TX
PD6						SPI3_MOSI I2S3_SDEXT		USART2_RX
PD7								USART2_CK
PD8								USART3_TX
PD9								USART3_RX
PD10								USART3_CK
PD11					I2C2_SMBA			USART3_CTS
PD12			TMR4_CH1		I2C2_SCL			USART3_RTS_DE
PD13			TMR4_CH2		I2C2_SDA			
PD14			TMR4_CH3		I2C3_SCL			
PD15			TMR4_CH4		I2C3_SDA			

Pin	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PD0		CAN1_RX	XMC_A5		XMC_D2			EVENTOUT
PD1		CAN1_TX	XMC_A6		XMC_D3			EVENTOUT
PD2	UART5_RX		XMC_A7		SDIO1_CMD	DVP_D11	XMC_NWE	EVENTOUT
PD3		QSPI1_SCK	XMC_A8		XMC_CLK	DVP_D5		EVENTOUT
PD4			XMC_A9		XMC_NOE			EVENTOUT
PD5			XMC_A10		XMC_NWE			EVENTOUT
PD6			XMC_A11		XMC_NWAIT	DVP_D10		EVENTOUT
PD7			XMC_A12		XMC_NE1 XMC_NCE2			EVENTOUT
PD8				EMAC_MII_RX_DV EMAC_RMII_CRS_DV	XMC_D13			EVENTOUT
PD9				EMAC_MII_RXD0 EMAC_RMII_RXD0	XMC_D14			EVENTOUT
PD10				EMAC_MII_RXD1 EMAC_RMII_RXD1	XMC_D15			EVENTOUT
PD11		QSPI1_IO0	XMC_A14 XMC_SDBA0	EMAC_MII_RXD2	XMC_A16 XMC CLE			EVENTOUT
PD12		QSPI1_IO1	XMC_A15 XMC_SDBA1	EMAC_MII_RXD3	XMC_A17 XMC ALE			EVENTOUT
PD13	UART8_TX	QSPI1_IO3	XMC_SD_CLK		XMC_A18			EVENTOUT
PD14	UART8_RX				XMC_D0			EVENTOUT
PD15					XMC_D1			EVENTOUT

Table 6-5 Port E multiplexed function configuration with GPIOE_MUX* register

Pin	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PE0			TMR4_EXT				TMR20_EXT	
PE1		TMR1_CH2C					TMR20_CH4	
PE2			TMR3_EXT			SPI4_SCK I2S4_CK	TMR20_CH1	
PE3			TMR3_CH1				TMR20_CH2	
PE4	CLKOUT1		TMR3_CH2			SPI4_CS I2S4_WS	TMR20_CH1C	
PE5			TMR3_CH3	TMR9_CH1		SPI4_MISO	TMR20_CH2C	
PE6			TMR3_CH4	TMR9_CH2		SPI4_MOSI I2S4_SDEXT	TMR20_CH3C	
PE7		TMR1_EXT						
PE8		TMR1_CH1C						UART4_TX
PE9		TMR1_CH1						UART4_RX
PE10		TMR1_CH2C						
PE11		TMR1_CH2				SPI4_CS I2S4_WS		
PE12		TMR1_CH3C			SPI1_CS I2S1_WS	SPI4_SCK I2S4_CK		
PE13		TMR1_CH3			SPI1_SCK I2S1_CK	SPI4_MISO		
PE14		TMR1_CH4			SPI1_MISO	SPI4_MOSI I2S4_SDEXT		
PE15		TMR1_BRK			SPI1_MOSI I2S1_SDEXT			

Pin	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PE0	UART8_RX				XMC_LB/XMC_SDDQML	DVP_D2		EVENTOUT
PE1	UART8_TX				XMC_UB/XMC_SDDQMH	DVP_D3		EVENTOUT
PE2		QSPI1_IO2	XMC_SDNCAS	EMAC_MII_TXD3	XMC_A23			EVENTOUT
PE3					XMC_A19	DVP_D9		EVENTOUT
PE4					XMC_A20	DVP_D4		EVENTOUT
PE5					XMC_A21	DVP_D6		EVENTOUT
PE6			XMC_SDNRAS		XMC_A22	DVP_D7		EVENTOUT
PE7	UART7_RX		QSPI2_IO0		XMC_D4			EVENTOUT
PE8	UART7_TX		QSPI2_IO1		XMC_D5			EVENTOUT
PE9			QSPI2_IO2		XMC_D6			EVENTOUT
PE10	UART5_TX		QSPI2_IO3		XMC_D7			EVENTOUT
PE11	UART5_RX				XMC_D8			EVENTOUT
PE12					XMC_D9			EVENTOUT
PE13					XMC_D10			EVENTOUT
PE14					XMC_D11			EVENTOUT
PE15					XMC_D12			EVENTOUT

Table 6-6 Port F multiplexed function configuration with GPIOF_MUX* register

Pin	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PF0					I2C2_SDA			
PF1					I2C2_SCL			
PF2			TMR20_CH3		I2C2_SMBA			
PF3			TMR20_CH4					
PF4			TMR20_CH1C					
PF5			TMR20_CH2C					
PF6			TMR20_CH4	TMR10_CH1				
PF7			TMR20_BRK	TMR11_CH1				
PF8								
PF9			TMR20_BRK					
PF10		TMR1_EXT	TMR5_CH4					
PF11			TMR20_EXT	TMR8_EXT				
PF12			TMR20_CH1	TMR8_BRK				
PF13			TMR20_CH2		I2C3_SMBA			
PF14			TMR20_CH3		I2C3_SCL			
PF15			TMR20_CH4		I2C3_SDA			

Pin	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PF0					XMC_A0			EVENTOUT
PF1					XMC_A1			EVENTOUT
PF2					XMC_A2			EVENTOUT
PF3					XMC_A3			EVENTOUT
PF4					XMC_A4			EVENTOUT
PF5					XMC_A5			EVENTOUT
PF6	UART7_RX	QSPI1_IO3			XMC_NIORD			EVENTOUT
PF7	UART7_TX	QSPI1_IO2			XMC_NREG			EVENTOUT
PF8		TMR13_CH1	QSPI1_IO0		XMC_NIOWR			EVENTOUT
PF9		TMR14_CH1	QSPI1_MOSI_IO1		XMC_CD			EVENTOUT
PF10		QSPI1_SCK			XMC_INTR	DVP_D11		EVENTOUT
PF11					XMC_SDNRAS	DVP_D12		EVENTOUT
PF12					XMC_A6			EVENTOUT
PF13					XMC_A7			EVENTOUT
PF14					XMC_A8			EVENTOUT
PF15					XMC_A9			EVENTOUT

Table 6-7 Port G multiplexed function configuration with GPIOG_MUX* register

Pin	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PG0			TMR20_CH1C			SPI1_MISO		
PG1			TMR20_CH2C			SPI1_MOSI I2S1_SDEXT		
PG2			TMR20_CH3C					
PG3			TMR20_BRK					
PG4								
PG5			TMR20_EXT					
PG6								
PG7								
PG8						QSPI2_CS		
PG9								
PG10						QSPI2_IO2		
PG11						QSPI2_IO3	SPI4_SCK I2S4_CK	
PG12						QSPI2_IO1	SPI4_MISO	
PG13						QSPI2_SCK	SPI4_MOSI I2S4_SDEXT	
PG14						QSPI2_IO0	SPI4_CS I2S4_WS	
PG15								

Pin	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PG0		CAN1_RX			XMC_A10			EVENTOUT
PG1		CAN1_TX			XMC_A11			EVENTOUT
PG2					XMC_A12			EVENTOUT
PG3					XMC_A13			EVENTOUT
PG4					XMC_A14 XMC_SDBA0			EVENTOUT
PG5					XMC_A15 XMC_SDBA1			EVENTOUT
PG6			QSPI1_CS		XMC_INT2	DVP_D12		EVENTOUT
PG7	USART6_CK				XMC_INT3	DVP_D13		EVENTOUT
PG8	USART6_RTS_DE			EMAC_PPS_OUT	XMC_SDCLK			EVENTOUT
PG9	USART6_RX	QSPI1_IO2			XMC_NE2 XMC_NCE3	DVP_VSYNC		EVENTOUT
PG10					XMC_NE3 XMC_NCE4_1	DVP_D2		EVENTOUT
PG11		CAN2_RX		EMAC_MII_TX_EN EMAC_RMII_TX_EN	XMC_NCE4_2	DVP_D3		EVENTOUT
PG12	USART6_RTS_DE	CAN2_TX			XMC_NE4			EVENTOUT
PG13	USART6_CTS			EMAC_MII_TXD0 EMAC_RMII_TXD0	XMC_A24			EVENTOUT
PG14	USART6_TX	QSPI1_IO3		EMAC_MII_TXD1 EMAC_RMII_TXD1	XMC_A25			EVENTOUT
PG15	USART6_CTS				XMC_SDNCAS	DVP_D13		EVENTOUT

Table 6-8 Port H multiplexed function configuration with GPIOH_MUX* register

Pin	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PH0					I2C1_SDA			
PH1					I2C1_SCL			
PH2			TMR5_CH1		I2C2_SCL			
PH3			TMR5_CH2		I2C2_SDA			

Pin	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PH0								EVENTOUT
PH1								EVENTOUT
PH2	UART4_RX		QSPI1_IO0					EVENTOUT
PH3	UART4_TX		QSPI1_IO1					EVENTOUT

Note: EVENTOUT is the Cortex-M TXEV signal.

6.2.10 Peripheral MUX function configuration

IOMUX function configuration as follows:

- To use a peripheral pin in MUX output, it is configured as multiplexed push-pull/open-drain output.
- To use a peripheral pin in MUX input, it is configured as multiplexed mode (floating/pull-up/pull-down).
- For ADC peripherals, the pins of analog channels should be configured as analog input/output mode.
- For I2C peripherals that intend to use pins as bidirectional functions, open-drain mode is required.
- For USB OTGFS_ID pin, configure corresponding IOMUX and enable corresponding clocks in CRM, there is no need of GPIO status configuration

6.2.11 IOMUX map priority

The unique peripheral multiplexed function can be configured through the GPIOx_MUXL/GPIOx_MUXH register, except individual pins that may be directly owned by hardware.

Some pins have been directly owned by specific hardware feature, whatever GPIO configuration.

Table 6-9 Pins owned by hardware

Pin	Enable bit	Description
PA0	PWC_CTRLSTS[8] =1	Once enabled, PA0 pin acts as WKUP1 function of PWC.
PA0	(ERTC_CTRL[11]=1& ERTC_TAMP[17]=1)| (ERTC_TAMP[0]=1& ERTC_TAMP[16]=1)| (ERTC_TAMP[3]=1)	Once enabled, PA0 pin is used as TAMPER2_BPR.
PA4	DAC_CTRL[2] =1	Once enabled, PA4 pin acts as DAC1 analog channel.
PA5	DAC_CTRL[18] =1	Once enabled, PA5 pin acts as DAC2 analog channel.
PC13	PWC_CTRLSTS[9] = 1	Once enabled, PA0 pin acts as WKUP2 function of PWC.
PC13	(ERTC_CTRL[23]=1)| (ERTC_CTRL[22:21]!=00)| (ERTC_CTRL[11]=1& ERTC_TAMP[17]=0)| (ERTC_TAMP[0]=1& ERTC_TAMP[16]=0)	Once enabled, PC13 pin acts as RTC channel.
PC14	CMR_BPDC[0]=1	Once enabled, PC14 pin acts as LEXT channel.
PC15	CMR_BPDC[0]=1 & CMR_BPDC[2]=0	Once enabled, PC15 pin acts as LEXT channel.
PH0	CMR_CTRL[16]=1	Once enabled, PH0 pin acts as HEXT channel.
PH1	CMR_CTRL[16]=1&	Once enabled, PH1 pin acts as HEXT channel.

CMR_CTRL[18]=0

Note: Either PA0 or PC13 cannot be used as TAMPER_BPR and WKUP of PWC simultaneously.

6.2.12 External interrupt/wake-up lines

Each pin can be used as an external interrupt input. The corresponding pin should be configured as input mode.

6.3 GPIO registers

Table 6-10 lists GPIO register map and their reset values. These peripheral registers must be accessed by bytes (8 bits), half-words (16 bits) or words (32 bits).

Table 6-10 GPIO register map and reset values

Register	Offset	Reset value
GPIOA_CFGR	0x00	0xA800 0000
GPIOx_CFGR(x =B,C,F)	0x00	0x0000 0280(B) 0x0000 0000
GPIOx_OMODER	0x04	0x0000 0000
GPIOx_ODRVR	0x08	0x0000 00C0(B) 0x0000 0000
GPIOA_PULL	0x0C	0x6400 0000(A)
GPIOx_PULL(x = B,C,F)	0x0C	0x0000 0100(B) 0x0000 0000
GPIOx_IDT	0x10	0x0000 XXXX
GPIOx_ODT	0x14	0x0000 0000
GPIOx_SCR	0x18	0x0000 0000
GPIOx_WPR	0x1C	0x0000 0000
GPIOx_MUXL	0x20	0x0000 0000
GPIOx_MUXH	0x24	0x0000 0000
GPIOx_CLR	0x28	0x0000 0000
GPIOx_HDRV	0x3C	0x0000 0000

6.3.1 GPIO configuration register (GPIOx_CFGR) (x=A..H)

Address offset: 0x00

Reset values: 0xa8000000 for port A, 0x0000 0280 for port B, 0x00000000 for other ports

Bit	Register	Reset value	Type	Description
Bit 2y+1: 2y	IOMCy	0xA800 0000	rw	GPIOx mode configuration (y=0~15) 00: Input mode (reset state) 01: General-purpose output mode 10: Multiplexed function mode 11: Analog mode

6.3.2 GPIO output mode register (GPIOx_OMODE) (x=A..H)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Always 0.
Bit 15: 0	OM	0x0000	rw	GPIOx output mode configuration These field is used to configure the output mode of the GPIOx: 0: Push-pull (reset state) 1: Open-drain

6.3.3 GPIO drive capability register (GPIOx_ODRVR) (x=A..H)

Address offset: 0x08

Reset values: 0x0000 00C0 for port B 0x00000000 for other ports

Bit	Register	Reset value	Type	Description
Bit 2y+1: 2y	ODRVy	0x0000 0000	rw	GPIOx drive capability (y=0…15) This field is used to configure the IO port drive capability. x0: Normal sourcing/sinking strength 01: Large sourcing/sinking strength 11: Normal sourcing/sinking strength

6.3.4 GPIO pull-up/pull-down register (GPIOx_PULL) (x=A..H)

Address offset: 0x0C

Reset values: 0x6400 0000 for port A, 0x0000 0100 for port B, 0x0000000 for other ports

Bit	Register	Reset value	Type	Description
Bit 2y+1: 2y	PULLy	0x6400 0000	rw	GPIOx pull-up/pull-down configuration (y=0…15) This field is used to configure the pull-up/pull-down of the IO port. 00: No pull-up, pull-down 01: Pull-up 10: Pull-down

6.3.5 GPIO input register (GPIOx_IDH) (x=A..H)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	IDT	0xXXXX	ro	GPIOx input data Indicates the input status of I/O port. Each bit corresponds to an I/O.

6.3.6 GPIO output register (GPIOx_IDH) (x=A..H)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	ODT	0x0000	rw	GPIOx output data Each bit represents an I/O port. It indicates the output status of I/O port. 0: Low 1: High

6.3.7 GPIO set/clear register (GPIOx_SCR) (x=A..H)

Bit	Register	Reset value	Type	Description
Bit 31: 16	IOCB	0x0000	wo	GPIOx clear bit The corresponding ODT register bit is cleared by writing “1” to these bits. Otherwise, the corresponding ODT register bit remains unchanged, which acts as ODT register bit operations. 0: No action to the corresponding ODT bits 1: Clear the corresponding ODT bits
Bit 15: 0	IOSB	0x0000	wo	GPIOx set bit The corresponding ODT register bit is set by writing “1” to these bits. Otherwise, the corresponding ODT register bit remains unchanged, which acts as ODT register bit operations. 0: No action to the corresponding ODT bits 1: Set the corresponding ODT bits

6.3.8 GPIO write protection register (GPIOx_WPR) (x=A..H)

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	WPSEQ	0x0	rw	Write protect sequence Write protect enable sequence bit and WPEN bit must be enabled at the same time to achieve write protection for some I/O bits. Write protect enable bit is executed four times in the order below: write “1” -> write “0” -> write “1” -> read. Note that the value of WPEN bit cannot be modified during this period.
Bit 15: 0	WPEN	0x0000	rw	Write protect enable Each bit corresponds to an I/O port. 0: No effect. 1: Write protect

6.3.9 GPIO multiplexed function low register (GPIOx_MUXL) (x=A..H)

Address offset: 0x20

Reset value: 0x00000000

Bit	Register	Reset value	Type	Description
Bit 4y+3: 4y	MUXLy	0x0	rw	Multiplexed function select for GPIOx pin y (y=0…7) This field is used to configure multiplexed function IOs. 0000: MUX0 0001: MUX1 0010: MUX2 0011: MUX3 0100: MUX4 0101: MUX5 0110: MUX6 0111: MUX7 1000: MUX8 1001: MUX9 1010: MUX10 1011: MUX11 1100: MUX12 1101: MUX13 1110: MUX14 1111: MUX15

6.3.10 GPIO multiplexed function high register (GPIOx_MUXH) (x=A..H)

Bit	Register	Reset value	Type	Description
Bit 4y+3: 4y	MUXHy	0x0	rw	Multiplexed function select for GPIOx pin y (y=8…15) This field is used to configure multiplexed function IOs 0000: MUX0 0001: MUX1 0010: MUX2 0011: MUX3 0100: MUX4 0101: MUX5 0110: MUX6 0111: MUX7 1000: MUX8 1001: MUX9 1010: MUX10 1011: MUX11 1100: MUX12 1101: MUX13 1110: MUX14 1111: MUX15

6.3.11 GPIO port bit clear register (GPIOx_CLR) (x=A..H)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	IOCB	0x0000	wo	GPIOx clear bit The corresponding ODT register bit is cleared by writing “1” to these bits. Otherwise, the corresponding ODT register bit remains unchanged, which acts as ODT register bit operations. 0: No action to the corresponding ODT bits 1: Clear the corresponding ODT bits

6.3.12 GPIO huge current control register (GPIOx_HDRV) (x=A..E)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15:0	HDRV	0x0000	rw	Huge sourcing/sinking strength control 0: Not active 1: GPIO is configured as maximum sourcing/sinking strength

7 System configuration controller (SCFG)

7.1 Introduction

This device contains a set of system configuration register. The system configuration controller is mainly used to:

- Manage the external interrupts connected to the GPIOs
- Control the memory mapping mode
- Manage IRTMR/EMAC GPIO configurations

7.2 IOMUX registers

Table 7-1 shows SCFG register map and their reset values.

These peripheral registers must be accessed by words (32 bits).

Table 7-1 SCFG register map and reset values

Register	Offset	Reset value
SCFG_CFG1	0x00	0x0000 000X
SCFG_CFG2	0x04	0x0000 0000
SCFG_EXINTC1	0x08	0x0000 0000
SCFG_EXINTC2	0x0C	0x0000 0000
SCFG_EXINTC3	0x10	0x0000 0000
SCFG_EXINTC4	0x14	0x0000 0000
SCFG_UHDRV	0x2C	0x0000 0000

7.2.1 SCFG configuration register1 (SCFG_CFG1)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value.
Bit 11: 10	SWAP_XMC	0x0	rw	XMC address mapping swap 00: No XMC address mapping swap 01: SDRAM addresses are mapped at 0x6000 0000 and 0x7000 0000. NOR/PSRAM /SRAM/NAND2 memory addresses are mapped at 0xC000 00000 and 0xD000 0000. 10: QSPI2 memory addresses are mapped at 0x8000 0000. NAND3 memory is mapped at 0xB000 0000. 11: SDRAM memory addresses are mapped at 0x6000 0000 and 0x7000 0000. NOR/PSRAM /SRAM/NAND2 memory addresses are mapped at 0xC000 00000 and 0xD000 0000. QSPI2 memory addresses are mapped at 0x8000 0000. NAND3 memory is mapped at 0xB000 0000.
Bit 9: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7: 6	IR_SRC_SEL	0x0	rw	Infrared modulation envelope signal source selection This field is used to select the infrared modulation envelope signal source. 00: TMR10 01: USART1 10: USART2 11: Reserved
Bit 5	IR_POL	0x0	rw	Infrared output polarity selection 0: Infrared output (IR_OUT) is not inversed 1: Infrared output (IR_OUT) is inversed
Bit 4: 3	Reserved	0x0	resd	Kept at its default value.
Bit 2: 0	MEM_MAP_SEL	0xX	rw	Memory address mapping selection This field defines the memory address mapping at 0x0000 0000. After reset, the reset value of this field is the same

				as that of the BOOT0 and BOOT1 pins. After changing this field, the user can decide which of the following memory to be mapped at 0x0000 0000. 000: Main Flash memory mapped at 0x0000 0000 001: Boot loader memory mapped at 0x0000 0000 010: XMC BANK1 mapped at 0x0000 0000 011: Embedded SRAM mapped at 0x0000 0000 100: XMC SDRAM BANK1 mapped at 0x0000 0000 Others: Reserved.

7.2.2 SCFG configuration register2 (SCFG_CFG2)

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23	MII_RMII_SEL	0x0	rw	MII or RMII selection This bit is used to select the Ethernet MII or RMII interface. 0: MII 1: RMII Note: This bit applies to AT32F437 only.
Bit 22: 0	Reserved	0x00000X	resd	Kept at its default value.

7.2.3 SCFG external interrupt configuration register1 (SCFG_EXINTC1)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 12	EXINT3	0x0000	rw	EXINT3 input source configuration These bits are used to select the input source for the EXINT3 external interrupt. 0000: GPIOA pin3 0001: GPIOB pin3 0010: GPIOC pin3 0011: GPIOD pin3 0100: GPIOE pin3 0101: GPIOF pin3 0110: GPIOG pin3 0111: GPIOH pin3 Others: Reserved
Bit 11: 8	EXINT2	0x0000	rw	EXINT2 input source configuration These bits are used to select the input source for the EXINT2 external interrupt. 0000: GPIOA pin2 0001: GPIOB pin2 0010: GPIOC pin2 0011: GPIOD pin2 0100: GPIOE pin2 0101: GPIOF pin2 0110: GPIOG pin2 0111: GPIOH pin2 Others: Reserved
Bit 7: 4	EXINT1	0x0000	rw	EXINT1 input source configuration These bits are used to select the input source for the EXINT1 external interrupt.

				0000: GPIOA pin1 0001: GPIOB pin1 0010: GPIOC pin1 0011: GPIOD pin1 0100: GPIOE pin1 0101: GPIOF pin1 0110: GPIOG pin1 0111: GPIOH pin1 Others: Reserved
Bit 3: 0	EXINT0	0x0000	rw	EXINT0 input source configuration These bits are used to select the input source for the EXINT0 external interrupt. 0000: GPIOA pin0 0001: GPIOB pin0 0010: GPIOC pin0 0011: GPIOD pin0 0100: GPIOE pin0 0101: GPIOF pin0 0110: GPIOG pin0 0111: GPIOH pin0 Others: Reserved

7.2.4 SCFG external interrupt configuration register2 (SCFG_EXINTC2)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 12	EXINT7	0x0000	rw	EXINT7 input source configuration These bits are used to select the input source for the EXINT7 external interrupt. 0000: GPIOA pin7 0001: GPIOB pin7 0010: GPIOC pin7 0011: GPIOD pin7 0100: GPIOE pin7 0101: GPIOF pin7 0110: GPIOG pin7 0111: GPIOH pin7 Others: Reserved
Bit 11: 8	EXINT6	0x0000	rw	EXINT6 input source configuration These bits are used to select the input source for the EXINT6 external interrupt. 0000: GPIOA pin6 0001: GPIOB pin6 0010: GPIOC pin6 0011: GPIOD pin6 0100: GPIOE pin6 0101: GPIOF pin6 0110: GPIOG pin6

				0111: GPIOH pin6 Others: Reserved
Bit 7: 4	EXINT5	0x0000	rw	EXINT5 input source configuration These bits are used to select the input source for the EXINT5 external interrupt. 0000: GPIOA pin5 0001: GPIOB pin5 0010: GPIOC pin5 0011: GPIOD pin5 0100: GPIOE pin5 0101: GPIOF pin5 0110: GPIOG pin5 0111: GPIOH pin5 Others: Reserved
Bit 3: 0	EXINT4	0x0000	rw	EXINT4 input source configuration These bits are used to select the input source for the EXINT4 external interrupt. 0000: GPIOA pin4 0001: GPIOB pin4 0010: GPIOC pin4 0011: GPIOD pin4 0100: GPIOE pin4 0101: GPIOF pin4 0110: GPIOG pin4 0111: GPIOH pin4 Others: Reserved

7.2.5 SCFG external interrupt configuration register3 (SCFG_EXINTC3)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 12	EXINT11	0x0000	rw	EXINT11 input source configuration These bits are used to select the input source for the EXINT11 external interrupt. 0000: GPIOA pin11 0001: GPIOB pin11 0010: GPIOC pin11 0011: GPIOD pin11 0100: GPIOE pin11 0101: GPIOF pin11 0110: GPIOG pin11 0111: GPIOH pin11 Others: Reserved
Bit 11: 8	EXINT10	0x0000	rw	EXINT10 input source configuration These bits are used to select the input source for the EXINT10 external interrupt. 0000: GPIOA pin10 0001: GPIOB pin10

				0010: GPIOC pin10 0011: GPIOD pin10 0100: GPIOE pin10 0101: GPIOF pin10 0110: GPIOG pin10 0111: GPIOH pin10 Others: Reserved
Bit 7: 4	EXINT9	0x0000	rw	EXINT9 input source configuration These bits are used to select the input source for the EXINT9 external interrupt. 0000: GPIOA pin9 0001: GPIOB pin9 0010: GPIOC pin9 0011: GPIOD pin9 0100: GPIOE pin9 0101: GPIOF pin9 0110: GPIOG pin9 0111: GPIOH pin9 Others: Reserved
Bit 3: 0	EXINT8	0x0000	rw	EXINT8 input source configuration These bits are used to select the input source for the EXINT8 external interrupt. 0000: GPIOA pin8 0001: GPIOB pin8 0010: GPIOC pin8 0011: GPIOD pin8 0100: GPIOE pin8 0101: GPIOF pin8 0110: GPIOG pin8 0111: GPIOH pin8 Others: Reserved

7.2.6 SCFG external interrupt configuration register4 (SCFG_EXINTC4)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 12	EXINT15	0x0000	rw	EXINT15 input source configuration These bits are used to select the input source for the EXINT15 external interrupt. 0000: GPIOA pin15 0001: GPIOB pin15 0010: GPIOC pin15 0011: GPIOD pin15 0100: GPIOE pin15 0101: GPIOF pin15 0110: GPIOG pin15 0111: GPIOH pin15 Others: Reserved

Bit 11: 8	EXINT14	0x0000	rw	EXINT14 input source configuration These bits are used to select the input source for the EXINT14 external interrupt. 0000: GPIOA pin14 0001: GPIOB pin14 0010: GPIOC pin14 0011: GPIOD pin14 0100: GPIOE pin14 0101: GPIOF pin14 0110: GPIOG pin14 0111: GPIOH pin14 Others: Reserved
Bit 7:4	EXINT13	0x0000	rw	EXINT13 input source configuration These bits are used to select the input source for the EXINT13 external interrupt. 0000: GPIOA pin13 0001: GPIOB pin13 0010: GPIOC pin13 0011: GPIOD pin13 0100: GPIOE pin13 0101: GPIOF pin13 0110: GPIOG pin13 0111: GPIOH pin13 Others: Reserved
Bit 3: 0	EXINT12	0x0000	rw	EXINT12 input source configuration These bits are used to select the input source for the EXINT12 external interrupt. 0000: GPIOA pin12 0001: GPIOB pin12 0010: GPIOC pin12 0011: GPIOD pin12 0100: GPIOE pin12 0101: GPIOF pin12 0110: GPIOG pin12 0111: GPIOH pin12 Others: Reserved

7.2.7 SCFG ultra high sourcing/sinking strength (SCFG_UHDRV)

Bit	Register	Reset value	Type	Description
Bit 31: 11	Reserved	0x0000 00	resd	Kept at its default value
Bit 10	PF15_UH	0x0	rw	PF15 Ultra high sourcing/sinking strength This bit is written by software to control the PF15 PAD sourcing/sinking strength. 0: Not active 1:Corresponding GPIO is switched to ultra-high sourcing/sinking strength When this bit is set, the control bits of GPIOx_OTYPER&GPIOx_HDRV become invalid.
Bit 9	PF14_UH	0x0	rw	PF14 Ultra high sourcing/sinking strength This bit is written by software to control the PF14 PAD sourcing/sinking strength. 0: Not active 1: Corresponding GPIO is switched to ultra-high sourcing/sinking strength When this bit is set, the control bits of GPIOx_OTYPER&GPIOx_HDRV become invalid.
Bit 8	PD15_UH	0x0	rw	PD15 Ultra high sourcing/sinking strength This bit is written by software to control the PD15 PAD sourcing/sinking strength. 0: Not active 1: Corresponding GPIO is switched to ultra-high sourcing/sinking strength When this bit is set, the control bits of GPIOx_OTYPER&GPIOx_HDRV become invalid.
Bit 7	PD14_UH	0x0	rw	PD14 Ultra high sourcing/sinking strength This bit is written by software to control the PD14 PAD sourcing/sinking strength. 0: Not active 1: Corresponding GPIO is switched to ultra-high sourcing/sinking strength When this bit is set, the control bits of GPIOx_OTYPER&GPIOx_HDRV become invalid.
Bit 6	PD13_UH	0x0	rw	PD13 Ultra high sourcing/sinking strength This bit is written by software to control the PD13 PAD sourcing/sinking strength. 0: Not active 1: Corresponding GPIO is switched to ultra-high sourcing/sinking strength When this bit is set, the control bits of GPIOx_OTYPER&GPIOx_HDRV become invalid.
Bit 5	PD12_UH	0x0	rw	PD12 Ultra high sourcing/sinking strength This bit is written by software to control the PD12 PAD sourcing/sinking strength. 0: Not active 1: Corresponding GPIO is switched to ultra-high sourcing/sinking strength When this bit is set, the control bits of GPIOx_OTYPER&GPIOx_HDRV become invalid.
Bit 4: 3	Reserved	0x0	rw	Kept at its default value
Bit 2	PB10_UH	0x0	rw	PB10 Ultra high sourcing/sinking strength This bit is written by software to control the PB10 PAD sourcing/sinking strength. 0: Not active 1: Corresponding GPIO is switched to ultra-high sourcing/sinking strength

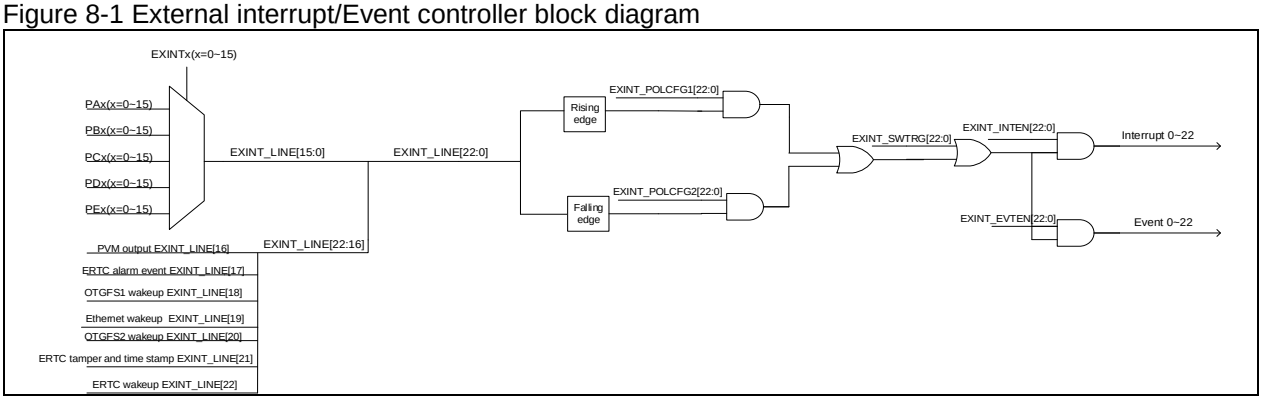
				When this bit is set, the control bits of GPIOx_OTYPER&GPIOx_HDRV become invalid.
Bit 1	PB9_UH	0x0	rw	PB9 Ultra high sourcing/sinking strength This bit is written by software to control the PB9 PAD sourcing/sinking strength. 0: Not active 1: Corresponding GPIO is switched to ultra-high sourcing/sinking strength When this bit is set, the control bits of GPIOx_OTYPER&GPIOx_HDRV become invalid.
Bit 0	PB3_UH	0x0	rw	PB3 Ultra high sourcing/sinking strength This bit is written by software to control the PB3 PAD sourcing/sinking strength. 0: Not active 1: Corresponding GPIO is switched to ultra-high sourcing/sinking strength When this bit is set, the control bits of GPIOx_OTYPER&GPIOx_HDRV become invalid.

8 External interrupt/Event controller (EXINT)

8.1 EXINT introduction

EXINT consists of 23 interrupt lines EXINT_LINE[22:0], each of which can generate an interrupt or event by edge detection trigger or software trigger. EXINT can enable or disable an interrupt or event independently through software configuration, and utilizes different edge detection modes (rising edge, falling edge or both edges) as well as trigger modes (edge detection, software trigger or both triggers) to respond to the trigger source in order to generate an interrupt or event.

Figure 8-1 External interrupt/Event controller block diagram



Main features:

- EXINT 0~15 mapping IO can be configured independently
- Independent trigger selection on each interrupt line
- Independent enable bit on each interrupt
- Independent enable bit on each event
- Up to 23 software trigger that can be generated and cleared independently
- Independent status bit on each interrupt
- Each interrupt can be cleared independently.

8.2 Function overview and configuration procedure

With up to 23 interrupt lines EXINT_LINE[22:0], EXINT can detect not only GPIO external interrupt sources but also seven internal sources such as PVM output, ERTC alarm events and time stamp events, ERTC wakeup events, OTGFS1/OTGFS2 events and Ethernet wakeup events through edge detection mechanism, where, GPIO interrupt sources can be selected with IOMUX_EXINTCx register. It should be noted that these input sources are mutually exclusive. For example, EXINT_LINE0 is allowed to select only one of PA0/PB0/PC0/PD0 pins, instead of taking both PA0 and PB0 as the input sources at the same time.

EXINT supports multiple edge detection modes, including rising edge, falling edge or both edges, selected by EXINT_POLCFG1 and EXINT_POLCFG2 register. Active edge trigger detected on the interrupt line can be used to generate an event or interrupt.

In addition, EXINT supports independent software trigger for the generation of an event or interrupt. This is achieved by setting the corresponding bits in the EXINT_SWTRG register.

EXINT can enable or disable an interrupt or event individually through software configuration such as EXINT_INTEN and EXINT_EVTEN register, indicating that the corresponding interrupt or event must be enabled prior to either edge detection or software trigger.

EXINT also features an independent interrupt status bit. Reading access to EXINT_INTSTS register can obtain the corresponding interrupt status. The status flag is cleared by writing “1” to this register.

Interrupt initialization procedure

- Select an interrupt source by setting IOMUX_EXINTCx register (This is required if GPIO is used as an interrupt source)
- Select an trigger mode by setting EXINT_POLCFG1 and EXINT_POLCFG2 register

- Enable interrupt or event by setting EXINT_INTEN and EXINT_EVTEN register
- Generate software trigger by setting EXINT_SWTRG register (This is applied to only software trigger interrupt)

Interrupt clear procedure

- Writing “1” to the EXINT_INTSTS register to clear the interrupts generated, and the corresponding bits in the EXINT_SWTRG register.

8.3 EXINT registers

These peripheral registers must be accessed by words (32 bits).

Table 8-1 shows EXINT register map and their reset value.

Table 8-1 External interrupt/Event controller register map and reset value

Register	Offset	Reset value
EXINT_INTEN	0x00	0x0000 0000
EXINT_EVTEN	0x04	0x0000 0000
EXINT_ POLCFG1	0x08	0x0000 0000
EXINT_ POLCFG2	0x0C	0x0000 0000
EXINT_ SWTRG	0x10	0x0000 0000
EXINT_ INTSTS	0x14	0x0000 0000

8.3.1 Interrupt enable register (EXINT_INTEN)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Forced to 0 by hardware.
Bit 22: 0	INTENx	0x00000	rw	Interrupt enable or disable on line x 0: Interrupt request is disabled. 1: Interrupt request is enabled. Note: Bit 19 is applied to only AT32F434/437A and is reserved otherwise.

8.3.2 Event enable register (EXINT_EVTEN)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Forced to 0 by hardware.
Bit 22: 0	EVTENx	0x00000	rw	Event enable or disable on line x 0: Event request is disabled. 1: Event request is enabled. Note: Bit 19 is applied to only AT32F437/437a and is reserved otherwise.

8.3.3 Polarity configuration register1 (EXINT_ POLCFG1)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Forced to 0 by hardware.
Bit 22: 0	RPx	0x00000	rw	Rising polarity configuration bit on line x These bits are used to select a rising edge to trigger an interrupt and event on line x. 0: Rising trigger on line x is disabled. 1: Rising trigger on line x is enable. Note: Bit 19 is applied to only AT32F437/437A and is reserved otherwise.

8.3.4 Polarity configuration register2 (EXINT_ POLCFG2)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Forced to be 0 by hardware.
Bit 22: 0	FPx	0x00000	rw	Falling polarity configuration bit on line x These bits are used to select a falling edge to trigger an interrupt and event on line x. 0: Falling trigger on line x is disabled. 1: Falling trigger on line x is enabled. Note: Bit 19 is applied to only AT32F437/437A and is reserved otherwise.

8.3.5 Software trigger register (EXINT_ SWTRG)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Forced to 0 by hardware.
Bit 22: 0	SWTx	0x00000	rw	Software trigger on line x If the corresponding bit in EXINT_INTEN register is 1, the software writes to this bit. The hardware sets the corresponding bit in the EXINT_INTSTS automatically to generate an interrupt. If the corresponding bit in the EXINT_EVTEN register is 1, the software writes to this bit. The hardware generates an event on the corresponding interrupt line automatically. 0: Default value 1: Software trigger generated Note: This bit is cleared by writing 1 to the corresponding bit in the EXINT_INTSTS register. Note: Bit 19 is applied to only AT32F437/437A and is reserved otherwise.

8.3.6 Interrupt status register (EXINT_ INTSTS)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Forced to 0 by hardware.
Bit 22: 0	LINEx	0x00000	Rw1c	Line x status bit 0: No interrupt occurred. 1: Interrupt occurred. Note: This bit is cleared by writing 1. Bit 19 is applied to only AT32F437/437A and is reserved otherwise.

9 DMA controller (DMA)

9.1 Introduction

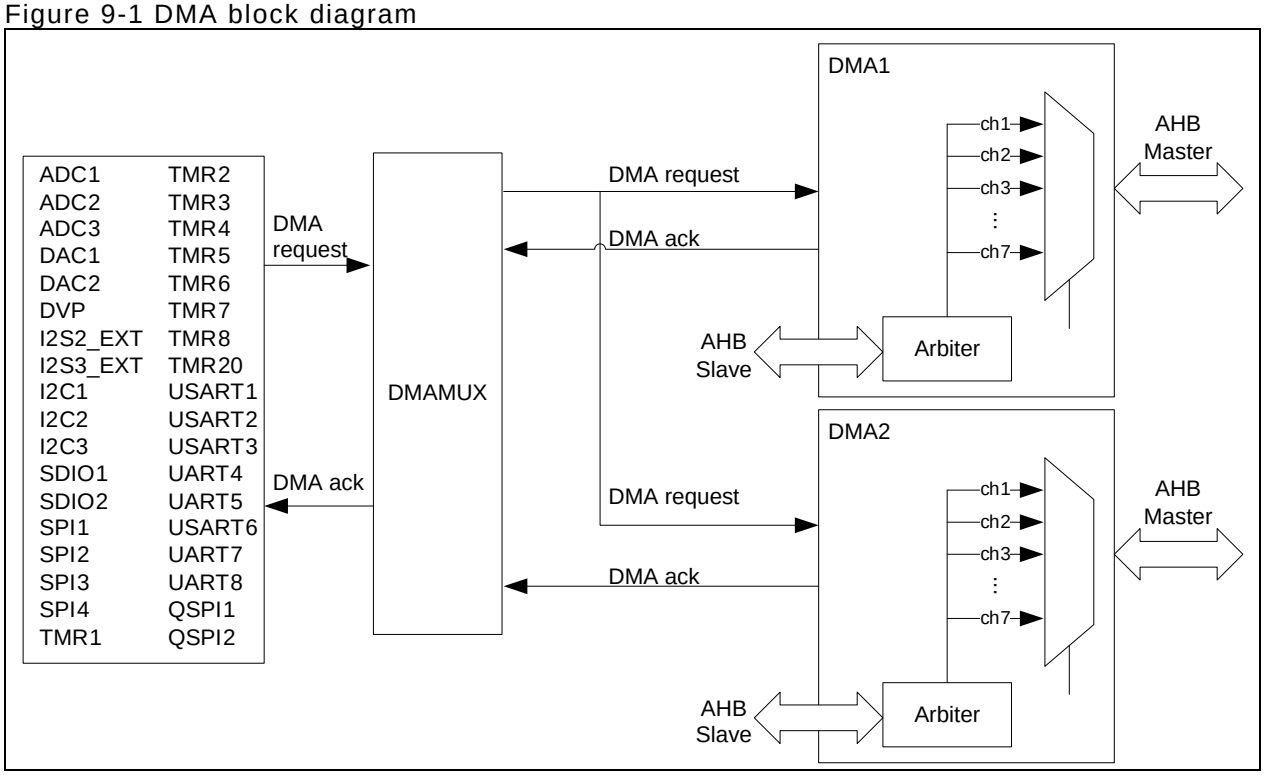
Direct memory access (DMA) controller is designed for 32-bit MCU applications with the aim of enhancing system performance and reducing the generation of interrupts.

There are two DMA controllers in the microcontroller. Each controller contains 7 DMA channels. Each channel manages memory access requests from one or more peripherals. An arbiter is available for coordinating the priority of each DMA request.

9.2 Main features

- AMBA compliant (Rev. 2.0)
- Only support AHB OKAY and ERROR responses
- HBUSREQ and HGRANT of AHB master interface are not supported
- Support 7 channels
- Peripheral-to-memory, memory-to-peripheral, and memory-to-memory transfers
- Support hardware handshake
- Support 8-bit, 16-bit and 32- bit data transfers
- Programmable amount of data to be transferred: up to 65535
- Support flexible mapping

Figure 9-1 DMA block diagram



Note: The number of DMA peripherals in Figure 9-1 may decrease depending on different models.

9.3 Function overview

9.3.1 DMA configuration

1. **Set the peripheral address in the DMA_CxPADDR register**
   The initial peripheral address for data transfer remains unchanged during transmission.
2. **Set the memory address in the DMA_CxMADDR register**
   The initial memory address for data transfer remains unchanged during transmission.
3. **Configure the amount of data to be transferred in the DMA_CxDTCNT register**
   Programmable data transfer size is up to 65535. This value is decremented after each data transfer.
4. **Configure the channel setting in the DMA_CxCTRL register**
   Including channel priority, data transfer direction/width, address incremented mode, circular mode and interrupt mode

- **Channel priority (CHPL)**

There are four levels, including very high priority, high priority, medium priority and low priority.
If the two channels have the same priority level, then the channel with lower number will get priority over the one with higher number. For example, channel 1 has priority over channel 2.

- **Data transfer direction (DTD)**

Memory-to-peripheral (M2P), peripheral-to-memory (P2M)

- **Address incremented mode (PINCM/MINCM)**

In incremented mode, the subsequent transfer address is the previous address plus transfer width (PWIDTH/MWIDTH).

- **Circular mode (LM)**

In circular mode, the contents in the DMA_CxDTCNT register is automatically reloaded with the initially programmed value after the completion of the last transfer.

- **Memory-to-memory mode (M2M)**

This mode indicates that DMA channels perform data transfer without requests from peripherals. Circular mode and memory-to-memory mode cannot be used at the same time.

5. **Enable DMA transfer by setting the CHEN bit in the DMA_CxCTRL register**

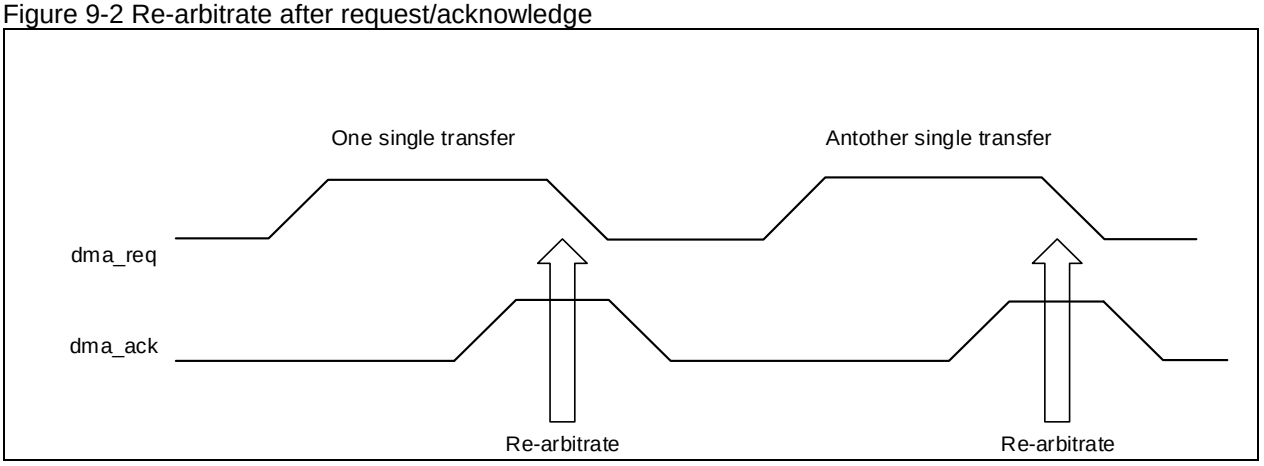
9.3.2 Handshake mechanism

In P2M and M2P mode, the peripherals need to send a request signal to the DMA controller. The DMA channel will send the peripheral transfer request (single) until the signal is acknowledged. After the completion of peripheral transmission, the DMA controller sends the acknowledge signal to the peripheral. The peripheral then releases its request as soon as it receives the acknowledge signal. At the same time, the DMA controller releases the acknowledge signal as well.

9.3.3 Arbiter

When several channels are enabled simultaneously, the arbiter will restart arbitration after full data transfer by the master controller. The channel with very high priority waits until the channel of the master controller has completed data transfers before taking control of it. The master controller will re-arbitrate to serve other channels as long as the channel completes a single transfer based on the master controller priority.

Figure 9-2 Re-arbitrate after request/acknowledge



9.3.4 Programmable data transfer width

Transfer width of the source data and destination data is programmable through the PWIDTH and MWIDTH bits in the DMA_CxCTRL register. When PWIDTH is not equal to MWIDTH, it can be aligned according to the settings of PWIDTH/ MWIDTH.

Figure 9-3 PWIDTH: byte, MWIDTH: half-word

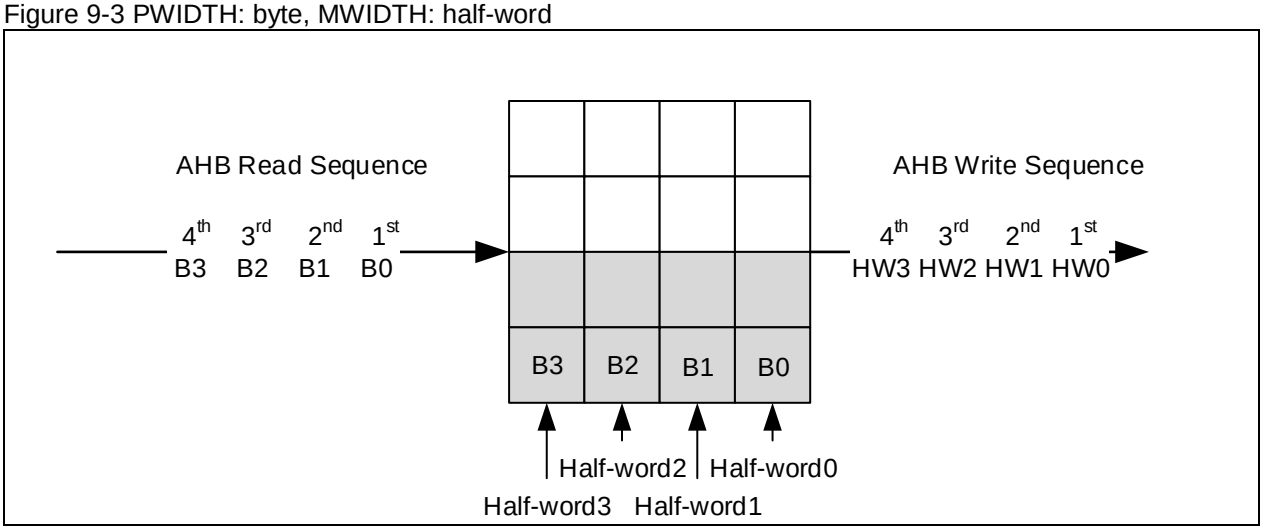


Figure 9-4 PWIDTH: half-word, MWIDTH: word

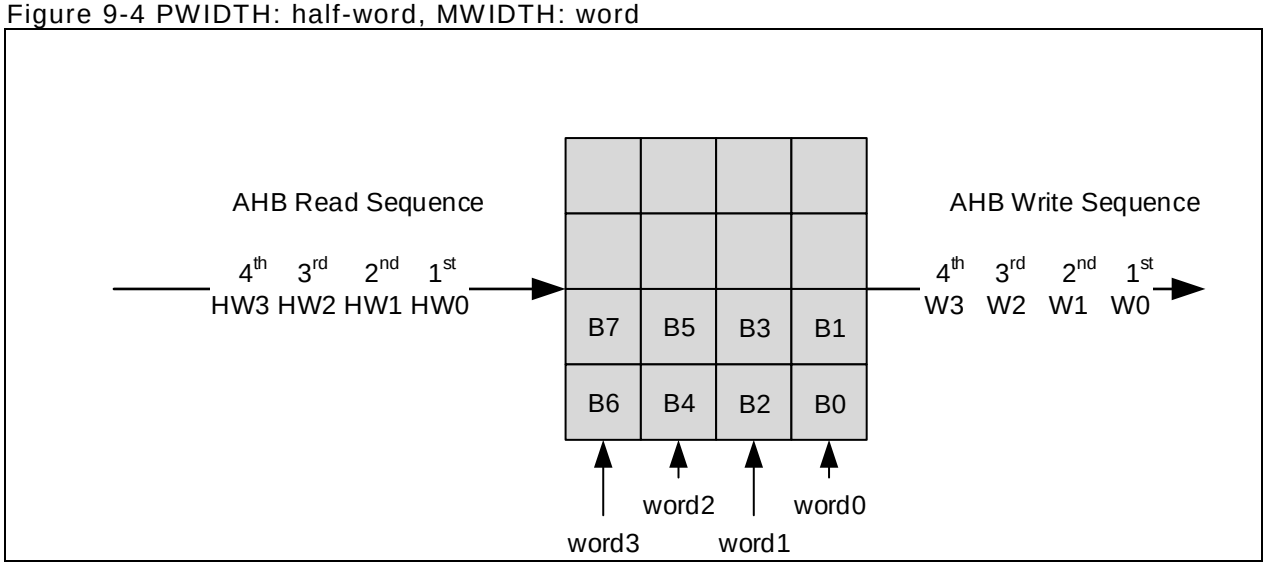
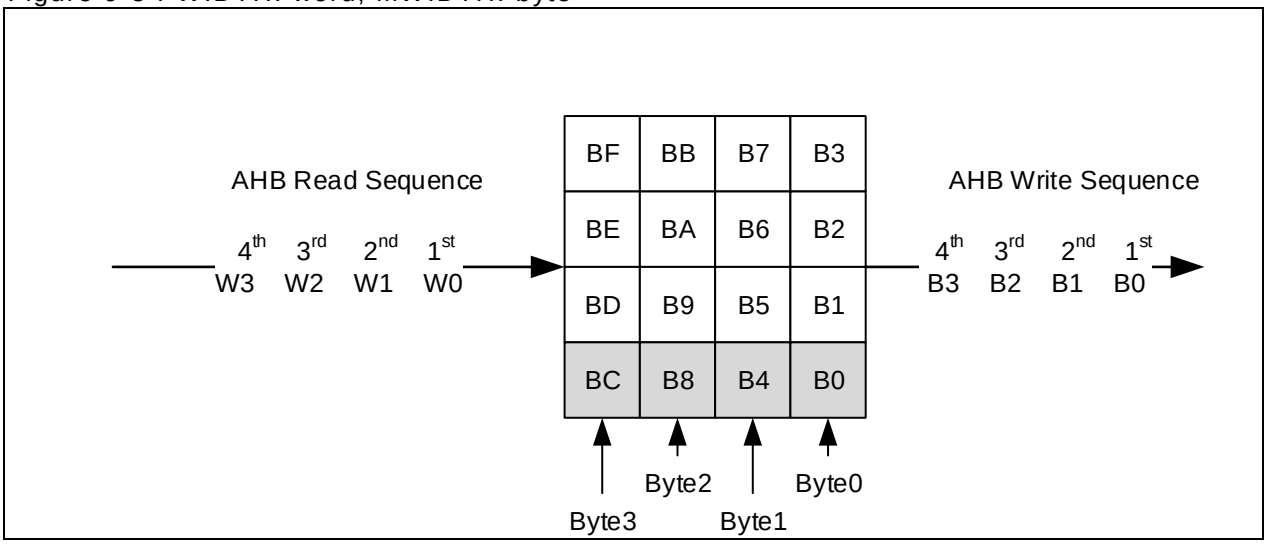


Figure 9-5 PWIDTH: word, MWIDTH: byte



9.3.5 Errors

Table 9-1 DMA error event

Error event	
Transfer error	AHB response error occurred during DMA read/write access

9.3.6 Interrupts

An interrupt can be generated on a DMA half-transfer, transfer complete and transfer error. Each channel has its specific interrupt flag, clear and enable bits, as shown in the table below.

Table 9-2 DMA interrupt requests

Interrupt event	Event flag bit	Clear control bit	Enable control bit
Half transfer	HDTF	HDTFC	HDTIEN
Transfer completed	FDTF	FDTFC	FDTIEN
Transfer error	DTERRF	DTERRFC	DTERRIEN

9.4 DMA multiplexer (DMAMUX)

DMAMUX manages DMA requests/acknowledge between peripherals and DMA controller.

The DMA controller selects the DMA mapping table with the TBL_SEL bit in the DMA_MUXSEL register. Each DMA controller stream selects only one DMA request from the flexible mapping table. In flexible mapping mode, each channel can bypass or synchronize 127 possible channel requests from peripherals or generators through the REQSEL [6: 0] bit in the DMA_MUXCxCTRL register.

EXINT LINE is used as the trigger input for request generators and the synchronized input for requests.

9.4.1 DMAMUX functional overview

The DMAMUX consists of a request generator and a request multiplexer.

Each of the DMAMUX generator channel x has a GEN enable bit in the DMA_MUXGxCTR register. The SIGSEL bit is used to select the trigger input of the DMAMUX generator. Typically, the number of DMA requests equals GREQCNT + 1. The GPOL bit is used in the DMA_MUXGxCTRL register to select a trigger event that can be on a rising edge, falling edge or either of them.

Each of the DMAMUX stream x comes from all_req [127: 1].

In flexible mapping mode, the SYNCEN bit in the DMA_MUXSxCTRL register is used to synchronize the selected DMA request input. In synchronous mode, the SYNCSEL bit in the DMA_MUXSxCTRL register is used to select synchronized input. The selected DMA request input will be transferred to chx_mux_req [7: 0] as soon as a valid edge of the synchronized input is detected by the SYNCPOL [1: 0] in the DMA_MUXSxCTRL register. In addition, when the EVTGEN bit is set in the DMA_MUXCxCTRL register, the programmable request counter (REQCNT) is used to generate a request output and event output.

Figure 9-6 DMAMUX block diagram

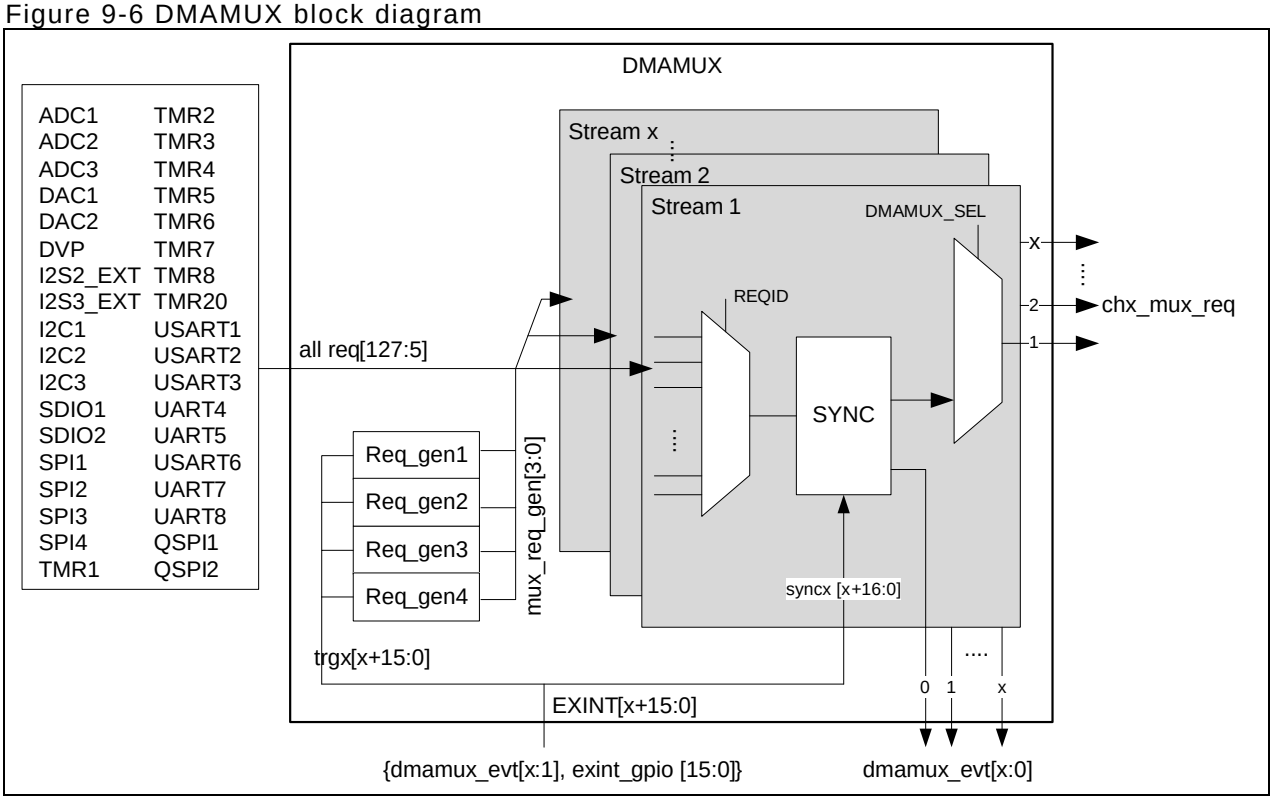


Table 9-3 Flexible DMA1/DMA2 request mapping

DMAMUX request	Source	DMAMUX request	Source	DMAMUX request	Source	DMAMUX request	Source
1	DMA_MUXREQG1	33	UART5_TX	65	TMR3_OVERFLOW	97	reserved
2	DMA_MUXREQG2	34	reserved	66	TMR3_TRIG	98	reserved
3	DMA_MUXREQG3	35	reserved	67	TMR4_CH1	99	reserved
4	DMA_MUXREQG4	36	ADC2	68	TMR4_CH2	100	reserved
5	ADC1	37	ADC3	69	TMR4_CH3	101	reserved
6	DAC1	38	reserved	70	TMR4_CH4	102	reserved
7	reserved	39	SDIO1	71	TMR4_OVERFLOW	103	SDIO2
8	TMR6_OVERFLOW	40	QSPI1	72	TMR5_CH1	104	QSPI2
9	TMR7_OVERFLOW	41	DAC2	73	TMR5_CH2	105	DVP
10	SPI1_RX	42	TMR1_CH1	74	TMR5_CH3	106	SPI4_RX
11	SPI1_TX	43	TMR1_CH2	75	TMR5_CH4	107	SPI4_TX
12	SPI2_RX	44	TMR1_CH3	76	TMR5_OVERFLOW	108	reserved
13	SPI2_TX	45	TMR1_CH4	77	TMR5_TRIG	109	reserved
14	SPI3_RX	46	TMR1_OVERFLOW	78	reserved	110	I2S2_EXT_RX
15	SPI3_TX	47	TMR1_TRIG	79	reserved	111	I2S2_EXT_TX
16	I2C1_RX	48	TMR1_HALL	80	reserved	112	I2S3_EXT_RX
17	I2C1_TX	49	TMR8_CH1	81	reserved	113	I2S3_EXT_TX
18	I2C2_RX	50	TMR8_CH2	82	reserved	114	USART6_RX
19	I2C2_TX	51	TMR8_CH3	83	reserved	115	USART6_TX
20	I2C3_RX	52	TMR8_CH4	84	reserved	116	UART7_RX
21	I2C3_TX	53	TMR8_OVERFLOW	85	reserved	117	UART7_TX
22	reserved	54	TMR8_TRIG	86	TMR20_CH1	118	UART8_RX
23	reserved	55	TMR8_HALL	87	TMR20_CH2	119	UART8_TX
24	USART1_RX	56	TMR2_CH1	88	TMR20_CH3	120	reserved
25	USART1_TX	57	TMR2_CH2	89	TMR20_CH4	121	reserved
26	USART2_RX	58	TMR2_CH3	90	TMR20_OVERFLOW	122	reserved
27	USART2_TX	59	TMR2_CH4	91	reserved	123	reserved
28	USART3_RX	60	TMR2_OVERFLOW	92	reserved	124	reserved
29	USART3_TX	61	TMR3_CH1	93	TMR20_TRIG	125	reserved
30	UART4_RX	62	TMR3_CH2	94	TMR20_HALL	126	TMR2_TRIG
31	UART4_TX	63	TMR3_CH3	95	reserved	127	reserved
32	UART5_RX	64	TMR3_CH4	96	reserved		

Table 9-4 DMAMUX EXINT LINE for trigger input and synchronized input

EXINT LINE	Source	EXINT LINE	Source	EXINT LINE	Source	EXINT LINE	Source
0	exint_gpio[0]	8	exint_gpio[8]	16	DMA_MUXevt1	24	reserved
1	exint_gpio[1]	9	exint_gpio[9]	17	DMA_MUXevt2	25	reserved
2	exint_gpio[2]	10	exint_gpio[10]	18	DMA_MUXevt3	26	reserved
3	exint_gpio[3]	11	exint_gpio[11]	19	DMA_MUXevt4	27	reserved
4	exint_gpio[4]	12	exint_gpio[12]	20	DMA_MUXevt5	28	reserved
5	exint_gpio[5]	13	exint_gpio[13]	21	DMA_MUXevt6	29	reserved
6	exint_gpio[6]	14	exint_gpio[14]	22	DMA_MUXevt7	30	reserved
7	exint_gpio[7]	15	exint_gpio[15]	23	reserved	31	reserved

9.4.2 DMAMUX overflow interrupts

During DMAMUX request generation, when a new trigger input occurs before the GREQCNT underflows, the TRGOVFx bit will be set in the DMA_MUXGSTS register. It is cleared by setting TRGOVFCx=1 in the DMA_MUXGCLR register. An interrupt will be generated if the interrupt enable bit TRGOVIEN is set in the DMA_MUXGxCTRL register.

In DMAMUX synchronous mode, when a new synchronized input occurs before the REQCNT underflows, the SYNCOVFx bit will be set in the DMA_MUXSYNCSTS register. It is cleared by setting the SYNCOVFCx bit in the DMA_MUXSYNCCLR register. An interrupt will be generated if the interrupt enable bit SYNCOVIEN is set in the DMA_MUXSxCTRL register.

Figure 9-7 DMAMUX request synchronized mode

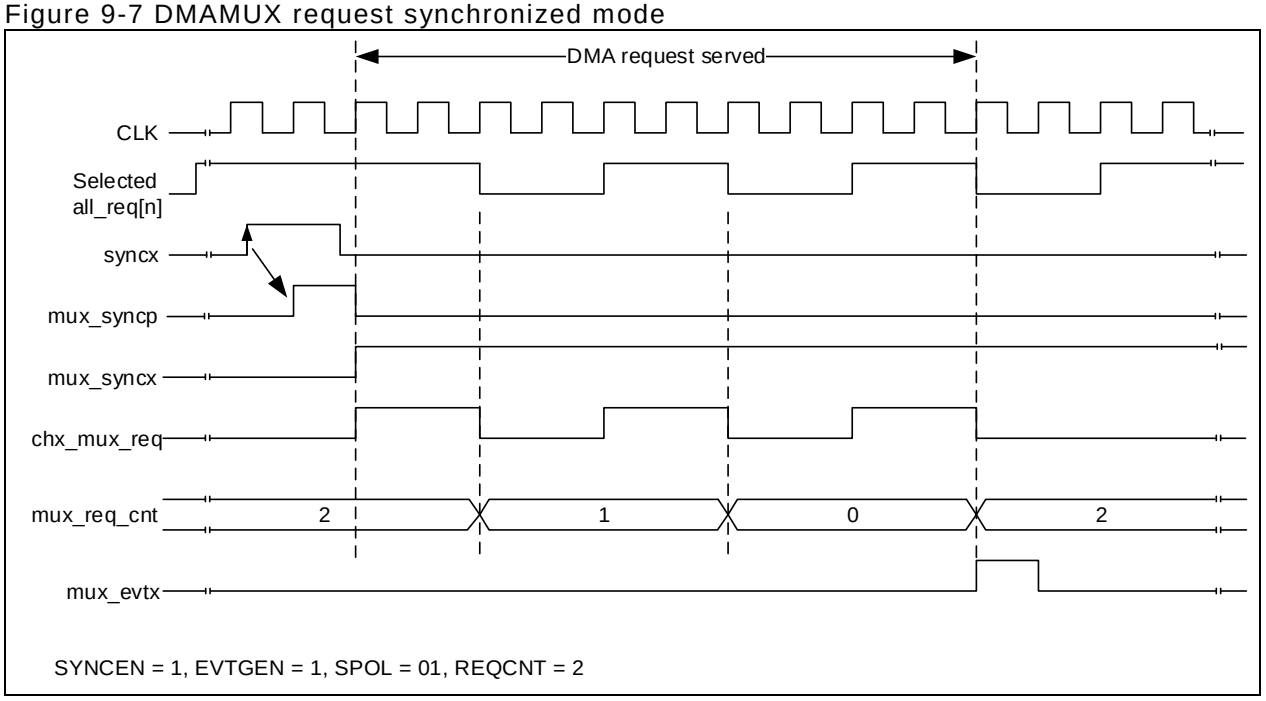
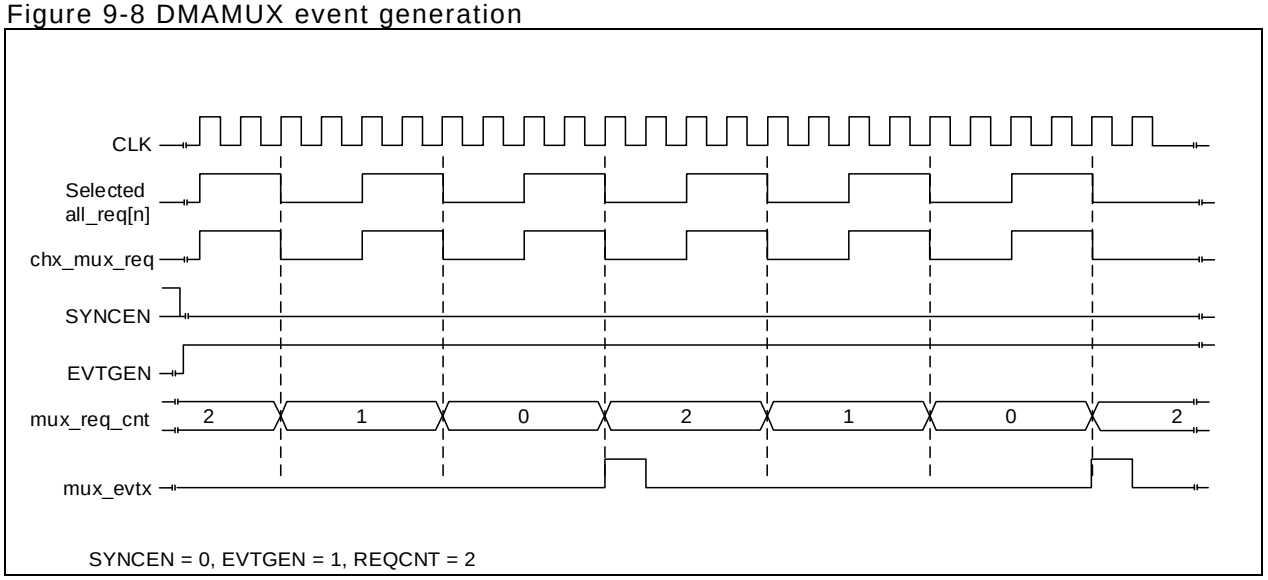


Figure 9-8 DMAMUX event generation



9.5 DMA registers

Table 9-5 shows DMA register map and their reset values.
These peripheral registers must be accessed by bytes (8 bits), half-words (16 bits) or words (32 bits).

Table 9-5 DMA register map and reset value

Register	Offset	Reset value
DMA_STS	0x00	0x0000 0000
DMA_CLR	0x04	0x0000 0000
DMA_C1CTRL	0x08	0x0000 0000
DMA_C1DTCNT	0x0c	0x0000 0000
DMA_C1PADDR	0x10	0x0000 0000
DMA_C1MADDR	0x14	0x0000 0000
DMA_C2CTRL	0x1c	0x0000 0000
DMA_C2DTCNT	0x20	0x0000 0000
DMA_C2PADDR	0x24	0x0000 0000
DMA_C2MADDR	0x28	0x0000 0000
DMA_C3CTRL	0x30	0x0000 0000
DMA_C3DTCNT	0x34	0x0000 0000
DMA_C3PADDR	0x38	0x0000 0000
DMA_C3MADDR	0x3c	0x0000 0000
DMA_C4CTRL	0x44	0x0000 0000
DMA_C4DTCNT	0x48	0x0000 0000
DMA_C4PADDR	0x4c	0x0000 0000
DMA_C4MADDR	0x50	0x0000 0000
DMA_C5CTRL	0x58	0x0000 0000
DMA_C5DTCNT	0x5c	0x0000 0000
DMA_C5PADDR	0x60	0x0000 0000
DMA_C5MADDR	0x64	0x0000 0000
DMA_C6CTRL	0x6c	0x0000 0000

	DMA_C6DTCNT	0x70	0x0000 0000
	DMA_C6PADDR	0x74	0x0000 0000
	DMA_C6MADDR	0x78	0x0000 0000
	DMA_C7CTRL	0x80	0x0000 0000
	DMA_C7DTCNT	0x84	0x0000 0000
	DMA_C7PADDR	0x88	0x0000 0000
	DMA_C7MADDR	0x8c	0x0000 0000
	DMA_MUXSEL	0x100	0x0000 0000
	DMA_MUXC1CTRL	0x104	0x0000 0000
	DMA_MUXC2CTRL	0x108	0x0000 0000
	DMA_MUXC3CTRL	0x10c	0x0000 0000
	DMA_MUXC4CTRL	0x110	0x0000 0000
	DMA_MUXC5CTRL	0x114	0x0000 0000
	DMA_MUXC6CTRL	0x118	0x0000 0000
	DMA_MUXC7CTRL	0x11c	0x0000 0000
	DMA_MUXG1CTRL	0x120	0x0000 0000
	DMA_MUXG2CTRL	0x124	0x0000 0000
	DMA_MUXG3CTRL	0x128	0x0000 0000
	DMA_MUXG4CTRL	0x12c	0x0000 0000
	DMA_MUXSYNCSTS	0x130	0x0000 0000
	DMA_MUXSYNCCLR	0x134	0x0000 0000
	DMA_MUXGSTS	0x138	0x0000 0000
	DMA_MUXGCLR	0x13c	0x0000 0000

9.5.1 DMA interrupt status register (DMA_STS)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
31: 28	Reserved	0x0	resd	Kept at its default value.
Bit 27	DTERRF7	0x0	ro	Channel 7 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 26	HDTF7	0x0	ro	Channel7 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 25	FDTF7	0x0	ro	Channel 7 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 24	GF7	0x0	ro	Channel7 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event occurred.
Bit 23	DTERRF6	0x0	ro	Channel 6 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.

Bit 22	HDTF6	0x0	ro	Channel 6 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 21	FDTF6	0x0	ro	Channel 6 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 20	GF6	0x0	ro	Channel 6 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event
Bit 19	DTERRF5	0x0	ro	Channel 5 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 18	HDTF5	0x0	ro	Channel 5 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 17	FDTF5	0x0	ro	Channel 5 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 16	GF5	0x0	ro	Channel 5 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event
Bit 15	DTERRF4	0x0	ro	Channel 4 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 14	HDTF4	0x0	ro	Channel 4 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 13	FDTF4	0x0	ro	Channel 4 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 12	GF4	0x0	ro	Channel 4 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event
Bit 11	DTERRF3	0x0	ro	Channel 3 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 10	HDTF3	0x0	ro	Channel 3 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 9	FDTF3	0x0	ro	Channel 3 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 8	GF3	0x0	ro	Channel 3 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event
Bit 7	DTERRF2	0x0	ro	Channel 2 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 6	HDTF2	0x0	ro	Channel 2 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.

Bit 5	FDTF2	0x0	ro	Channel 2 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 4	GF2	0x0	ro	Channel 2 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event
Bit 3	DTERRF1	0x0	ro	Channel 1 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 2	HDTF1	0x0	ro	Channel 1 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 1	FDTF1	0x0	ro	Channel 1 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 0	GF1	0x0	ro	Channel 1 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event

9.5.2 DMA interrupt flag clear register (DMA_CLR)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
31: 28	Reserved	0x0	resd	Kept at its default value.
Bit 27	DTERRFC7	0x0	rw1c	Channel 7 data transfer error flag clear 0: No effect 1: Clear the DTERRF flag in the DMA_STS register
Bit 26	HDTFC7	0x0	rw1c	Channel 7 half transfer flag clear 0: No effect 1: Clear the HDTF7 flag in the DMA_STS register
Bit 25	FDTFC7	0x0	rw1c	Channel 7 transfer complete flag clear 0: No effect 1: Clear the FDTF7 flag in the DMA_STS register
Bit 24	GFC7	0x0	rw1c	Channel 7 global interrupt flag clear 0: No effect 1: Clear the DTERRF7, HDTF7, FDTF7 and GF7 flag in the DMA_STS register
Bit 23	DTERRFC6	0x0	rw1c	Channel 6 data transfer error flag clear 0: No effect 1: Clear the DTERRF6 flag in the DMA_STS register
Bit 22	HDTFC6	0x0	rw1c	Channel 6 half transfer flag clear 0: No effect 1: Clear the HDTF6 flag in the DMA_STS register
Bit 21	FDTFC6	0x0	rw1c	Channel 6 transfer complete flag clear 0: No effect 1: Clear the FDTF6 flag in the DMA_STS register
Bit 20	GFC6	0x0	rw1c	Channel 6 global interrupt flag clear 0: No effect 1: Clear the DTERRF6, HDTF6, FDTF6 and GF6 flag in the DMA_STS register
Bit 19	DTERRFC5	0x0	rw1c	Channel 5 data transfer error flag clear 0: No effect 1: Clear the DTERRF5 flag in the DMA_STS register

Bit 18	HDTFC5	0x0	rw1c	Channel 5 half transfer flag clear 0: No effect 1: Clear the HDTF5 flag in the DMA_STS register
Bit 17	FDTFC5	0x0	rw1c	Channel 5 transfer complete flag clear 0: No effect 1: Clear the FDTF5 flag in the DMA_STS register
Bit 16	GFC5	0x0	rw1c	Channel 5 global interrupt flag clear 0: No effect 1: Clear the DTERRF5, HDTF5, FDTF5 and GF5 in the DMA_STS register
Bit 15	DTERRFC4	0x0	rw1c	Channel 4 data transfer error flag clear 0: No effect 1: Clear the DTERRF4 flag in the DMA_STS register
Bit 14	HDTFC4	0x0	rw1c	Channel 4 half transfer flag clear 0: No effect 1: Clear the HDTF4 flag in the DMA_STS register
Bit 13	FDTFC4	0x0	rw1c	Channel 4 transfer complete flag clear 0: No effect 1: Clear the FDTF4 flag in the DMA_STS register
Bit 12	GFC4	0x0	rw1c	Channel 4 global interrupt flag clear 0: No effect 1: Clear the DTERRF4, HDTF4, FDTF4 and GF4 flag in the DMA_STS register
Bit 11	DTERRFC3	0x0	rw1c	Channel 7 data transfer error flag clear 0: No effect 1: Clear the DTERRF7 flag in the DMA_STS register
Bit 10	HDTFC3	0x0	rw1c	Channel 7 half transfer flag clear 0: No effect 1: Clear the HDTF7 flag in the DMA_STS register
Bit 9	FDTFC3	0x0	rw1c	Channel 3 transfer complete flag clear 0: No effect 1: Clear the FDTF3 flag in the DMA_STS register
Bit 8	GFC3	0x0	rw1c	Channel 3 global interrupt flag clear 0: No effect 1: Clear the DTERRF3, HDTF3, FDTF3 and GF3 flag in the DMA_STS register
Bit 7	DTERRFC2	0x0	rw1c	Channel 2 data transfer error flag clear 0: No effect 1: Clear the DTERRF2 flag in the DMA_STS register
Bit 6	HDTFC2	0x0	rw1c	Channel 2 half transfer flag clear 0: No effect 1: Clear the HDTF2 flag in the DMA_STS register
Bit 5	FDTFC2	0x0	rw1c	Channel 2 transfer complete flag clear 0: No effect 1: Clear the FDTF2 flag in the DMA_STS register
Bit 4	GFC2	0x0	rw1c	Channel 2 global interrupt flag clear 0: No effect 1: Clear the DTERRF2, HDTF2, FDTF2 and GF2 in the DMA_STS register
Bit 3	DTERRFC1	0x0	rw1c	Channel 1 data transfer error flag clear 0: No effect 1: Clear the DTERRF1 flag in the DMA_STS register
Bit 2	HDTFC1	0x0	rw1c	Channel 1 half transfer flag clear 0: No effect 1: Clear the HDTF1 flag in the DMA_STS register
Bit 1	FDTFC1	0x0	rw1c	Channel 1 transfer complete flag clear 0: No effect 1: Clear the FDTF1 flag in the DMA_STS register

Bit 0	GFC1	0x0	rw1c	Channel 1 global interrupt flag clear 0: No effect 1: Clear the DTERRF1, HDTF1, FDTF1 and GF1 in the DMA_STS register

9.5.3 DMA channel-x configuration register (DMA_CxCTRL) (x = 1…7)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 15	Reserved	0x00000	resd	Kept at its default value.
Bit 14	M2M	0x0	rw	Memory to memory mode 0: Disabled 1: Enabled.
Bit 13: 12	CHPL	0x0	rw	Channel priority level 00: Low 01: Medium 10: High 11: Very high
Bit 11: 10	MWIDTH	0x0	rw	Memory data bit width 00: 8 bits 01: 16 bits 10: 32 bits 11: Reserved
Bit 9: 8	PWIDTH	0x0	rw	Peripheral data bit width 00: 8 bits 01: 16 bits 10: 32 bits 1: Reserved
Bit 7	MINCM	0x0	rw	Memory address increment mode 0: Disabled 1: Enabled.
Bit 6	PINCM	0x0	rw	Peripheral address increment mode 0: Disabled 1: Enabled.
Bit 5	LM	0x0	rw	Circular mode 0: Disabled 1: Enabled.
Bit 4	DTD	0x0	rw	Data transfer direction 0: Read from peripherals 1: Read from memory
Bit 3	DTERRIEN	0x0	rw	Data transfer error interrupt enable 0: Disabled 1: Enabled.
Bit 2	HDTIEN	0x0	rw	Half-transfer interrupt enable 0: Disabled 1: Enabled.
Bit 1	FDTIEN	0x0	rw	Transfer complete interrupt enable 0: Disabled 1: Enabled.
Bit 0	CHEN	0x0	rw	Channel enable 0: Disabled 1: Enabled.

9.5.4 DMA channel-x number of data register (DMA_CxDTCNT) (x = 1…7)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	CNT	0x0000	rw	Number of data to transfer The number of data to transfer is from 0x0 to 0xFFFF. This register can only written when the CHEN bit in the corresponding channel is set 0. The value is decremented after each DMA transfer. Note: This register holds the number of data to transfer, instead of transfer size. The transfer size is calculated by data width.

9.5.5 DMA channel-x peripheral address register (DMA_CxPADDR) (x = 1…7)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	PADDR	0x0000 0000	rw	Peripheral base address Base address of peripheral data register is the source or destination of data transfer. Note: The register can only be written when the CHEN bit in the corresponding channel is set 0.

9.5.6 DMA channel-x memory address register (DMA_CxMADDR) (x = 1…7)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	MADDR	0x0000 0000	rw	Memory base address Memory address is the source or destination of data transfer. Note: The register can only be written when the CHEN bit in the corresponding channel is set 0.

9.5.7 DMAMUX selection register (DMA_MUXSEL)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 0	TBL_SEL	0x0	rw	Multiplexer table select 0x1: Flexible mapping table

9.5.8 DMAMUX channel-x control register (DMA_MUXCxCTRL) (x = 1…7)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 25	Reserved	0x00	resd	Kept at its default value.
Bit 28: 24	SYNCSEL	0x00	rw	Synchronization select

Bit 23: 19	REQCNT	0x00	rw	DMA request count These bits indicate the number of DMA requests sent to the DMA controller after synchronization is enabled, and/or DMA request count before event output is generated. These bits are reserved only when both SYNCEN and EVTGEN bits are low.
Bit 18: 17	SYNCPOL	0x0	rw	Synchronization polarity This field defines the polarity of the selected synchronization input. 0x0: No events 0x1: Rising edge 0x2: Falling edge 0x3: Rising edge and falling edge
Bit 16	SYNCEN	0x0	rw	Synchronization enable 0: Synchronization disabled 1: Synchronization enabled
Bit 15: 10	Reserved	0x00	resd	Kept at its default value.
Bit 9	EVTGEN	0x0		Event generation enable 0: Event generation is disabled 1: Event generation is enabled
Bit 8	SYNCOVIEN	0x0		Synchronization overrun interrupt enable 0: Interrupt disabled 1: Interrupt enabled
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6: 0	REQSEL	0x00		DMA request select Select DMA request. Refer to DMAMUX table for more information.

9.5.9 DMAMUX generator x control register (DMA_MUXGxCTRL) (x = 1...4)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23: 19	GREQCNT	0x00	rw	DMA request generation count These bits define the number of DMA requests (GNBREQ + 1) to be generated when a trigger event occurs. This field is reserved only when the GEN bit is disabled.
Bit 18: 17	GPOL	0x0	rw	DMA request generation polarity This field defines the polarity of the selected trigger input. 0x0: No events 0x1: Rising edge 0x2: Falling edge 0x3: Rising and falling edges

Bit 16	GEN	0x0	rw	DMA request generation enable 0: DMA request generation is disabled 1: DMA request generation is enabled
Bit 15: 9	Reserved	0x00	resd	Kept at its default value.
Bit 8	TRGOVIEN	0x0	rw	Trigger overrun interrupt enable 0: Interrupt disabled 1: Interrupt enabled
Bit 7: 5	Reserved	0x0	resd	Kept at its default value.
Bit 4: 0	SIGSEL	0x00	rw	Signal select This field is used to select the DMA trigger input for DMA request generation.

9.5.10 DMAMUX channel synchronization status register (DMA_MUXSYNCSTS)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0000 00	resd	Kept at its default value.
Bit 7: 0	SYNCOVF	0x00	ro	Synchronization overrun interrupt flag When the DMA request count is less than REQCNT, this bit is set while a new synchronization event occurs.

9.5.11 DMAMUX channel interrupt clear flag register (DMA_MUXSYNCCLR)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0000 00	resd	Kept at its default value.
Bit 7: 0	SYNCOVFC	0x00	rw1c	Synchronization overrun interrupt flag clear Writing 1 to the corresponding bit can clear the SYNCOVF flag in the MUXSYNCST register.

9.5.12 DMAMUX generator interrupt status register (DMA_MUXGSTS)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 4	Reserved	0x0000 000	resd	Kept at its default value.
Bit 3: 0	TRGOVF	0x00	ro	Trigger overrun interrupt flag When the DMA request count is lower than GREQCNT, this field is set while a new trigger event occurs.

9.5.13 DMAMUX generator interrupt flag clear register (DMA_MUXGCLR)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit	**Register**	**Reset value**	**Type**	**Description**
Bit 3: 0	TRGOVFC	0x00	rw1c	Trigger overrun interrupt flag clear Writing 1 to the corresponding bit can clear the TRGOVF flag in the DMA_MUXGSTS register.

10 CRC calculation unit (CRC)

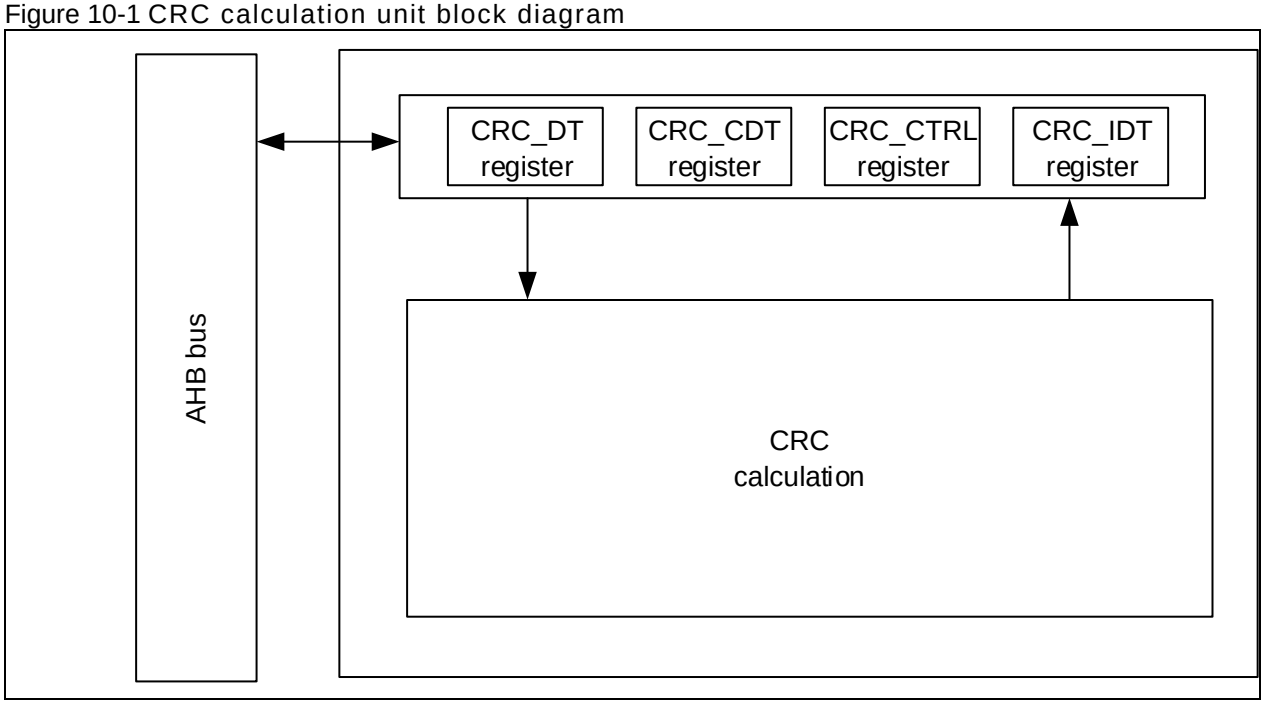
10.1 CRC introduction

The Cyclic Redundancy Check (CRC) is an independent peripheral with CRC check feature. It follows CRC32/MPEG-2 standard.

The CRC_CTRL register is used to select output data toggle (word, REVOD=1) or input data toggle (byte, REVID=01; half-word, REVID=0; word: REVID=11). CRC calculation unit is also equipped with initialization function. After each CRC reset, the value in the CRC_IDT register is written into the data register (CRC_DT) by CRC.

Users can write the data to go through CRC check and read the calculated result through CRC_DT register. Note that the calculation result is the combination of the previous result and the current value to be calculated.

Figure 10-1 CRC calculation unit block diagram



Main features

- Use CRC-32 code
- 4 HCLK cycles for each CRC calculation
- Support input/output data format toggle
- Perform write/read operation through CRC_DT register
- Set an initialization value with the CRC_IDT register. The value is loaded into CRC_DT register after each CRC reset.

10.2 CRC function description

According to CRC calculation principle, the input data is taken as dividend, and the generator polynominal as a division. Using mod 2 division logic, the input data divided by the generator polynominal gets a remainder, that is, the CRC value.

CRC calculation procedure

- Input data reverse. After data input, reverse input data depending on the REVID value in the CRC_CTRL register.
- Initialization. The first data input needs to be XOR-ed with the initial value defined in the CRC_IDT register. If f it is not the first data input, the initial value is the previously calculated result.
- CRC calculation. Dividing the input data by the generator polynominal (0x4C11DB7) using mod 2

division method produces a remainder, that is, CRC value.

- Output data toggle. Select whether to perform word toggle before output CRC value through the REVOD bit in the CRC_CTRL register.

CRC-32/MPEG-2 parameters

- Generator polynominal: 0x4C11DB7,

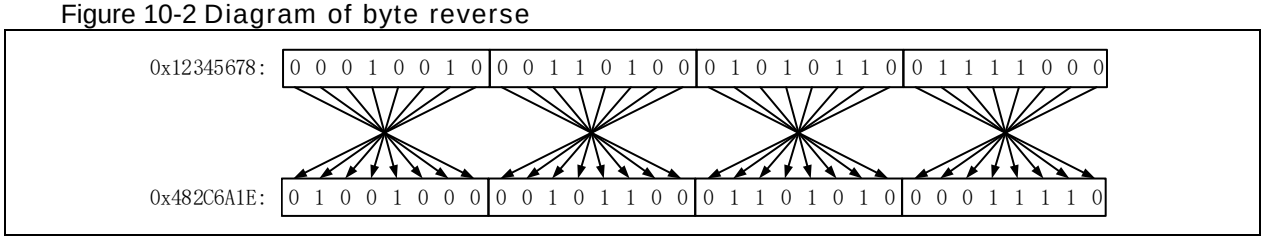
that is, X^32+X^26+X^23+X^22+X^16+X^12+X^11+X^10+X^8+X^7+X^5+X^4+X^2+X+1

- Initial value: 0xFFFF FFFF, in order to avoid that 1-byte 0x00 data to be calculated has the same result as that of multi-byte 0x00.
- XOR-ed value: 0x0000 0000, indicating that the CRC result will not be XOR-ed.

Toggle function

- Byte reverse, 8 bits in a group, and sequence is reversed within the group. As shown in the figure below, if the original data is 0x12345678, it is reversed as 0x482C6A1E.
- Half-word reverse, 16 bits in a group, and sequence is reversed within the group.
- Word reverse, 32 bits in a group, and sequence is reversed within the group.

Figure 10-2 Diagram of byte reverse



10.3 CRC registers

CRC_DT register can be accessed by bytes (8 bits), half-words (16 bits) or words (32 bits). Other registers have to be accessed by words (32 bits).

Table 10-1 CRC register map and reset value

Register	Offset	Reset value
CRC_DT	0x00	0xFFFF FFFF
CRC_CDT	0x04	0x0000 0000
CRC_CTRL	0x08	0x0000 0000
CRC_IDT	0x10	0xFFFF FFFF

10.3.1 Data register (CRC_DT)

Bit	Register	Reset value	Type	Description
Bit 31: 0	DT	0xFFFF FFFF	rw	Data value Used as input register when writing new data into the CRC calculator. It returns CRC calculation results when it is read.

10.3.2 Common data register (CRC_CDT)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7: 0	CDT	0x00	rw	Common 8-bit data value This field is used to store one byte data temporarily. This register is not affected by the CRC reset generated by the RST bit in the CRC_CTRL register.

10.3.3 Control register (CRC_CTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7	REVOD	0x0	resd	Reverse output data Set and cleared by software. This bit is used to control whether or not to reverse output data. 0: No effect 1: Word reverse
Bit 6: 5	REVID	0x0	rw	Reverse input data Set and cleared by software. This bit is used to control how to reverse input data. 00: No effect 01: Byte reverse 10: Half-word reverse 11: Word reverse
Bit 4: 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	RST	0x0	rw	Reset CRC calculation unit Set by software. Cleared by hardware. To reset CRC calculation unit, the data register is set as 0xFFFF FFFF. 0: No effect 1: Reset

10.3.4 Initialization register (CRC_IDT)

Bit	Register	Reset value	Type	Description
Bit 31: 0	IDT	0xFFFF FFFF	rw	Initialization data register When CRC reset is triggered by the RST bit in the CRC_CTRL register, the value in the initialization register is written into the CRC_DT register as an initial value.

11 I²C interface

11.1 I²C introduction

I²C (inter-integrated circuit) bus interface manages the communication between the microcontroller and serial I²C bus. It supports master and slave modes, with up to 1 Mbit/s of communication speed (enhanced edition).

11.2 I²C main features

- I2C bus
  - Master and slave modes
  - Multimaster capability
  - Stand mode (100 kHz), fast mode(400 kHz) and fast mode plus (1 MHz)
  - 7-bit and 10-bit address modes
  - Two 7-bit slave addresses (two addresses, one of them can be masked)
  - Broadcast call mode
  - Programmable data setup and hold time
  - Clock stretching capability
- Support DMA transfer
- Programmable digital noise filter
- Support SMBus2.protocol
  - PEC generation and verification
  - Acknowledgement control for command and data
  - ARP (address resolution protocol)
  - Master capability
  - Device capability
  - SMBus reminder capability
  - Timeout detection
  - Idle detection
- PMBus

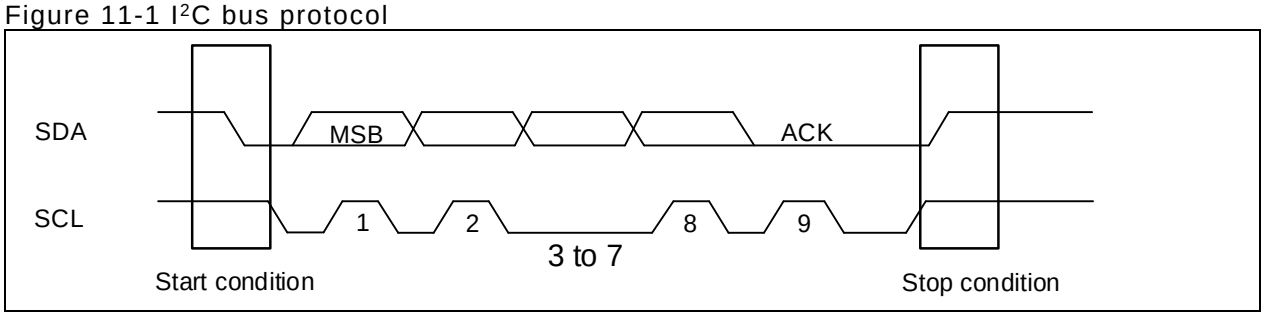
11.3 I²C function overview

I²C bus consists of a data line (SDA) and clock line (SCL). It can achieve a maximum of 100 kHz speed in standard mode, whereas up to 400kHz in fast mode. A frame of data transmission begins with a Start condition and ends with a Stop condition. The bus is kept in busy state after receiving the Start condition, and becomes idle as long as it receives the Stop condition.

Start condition: SDA switches from high to low when SCL is set high.

Stop condition: SDA switches from low to high when SCL is set high.

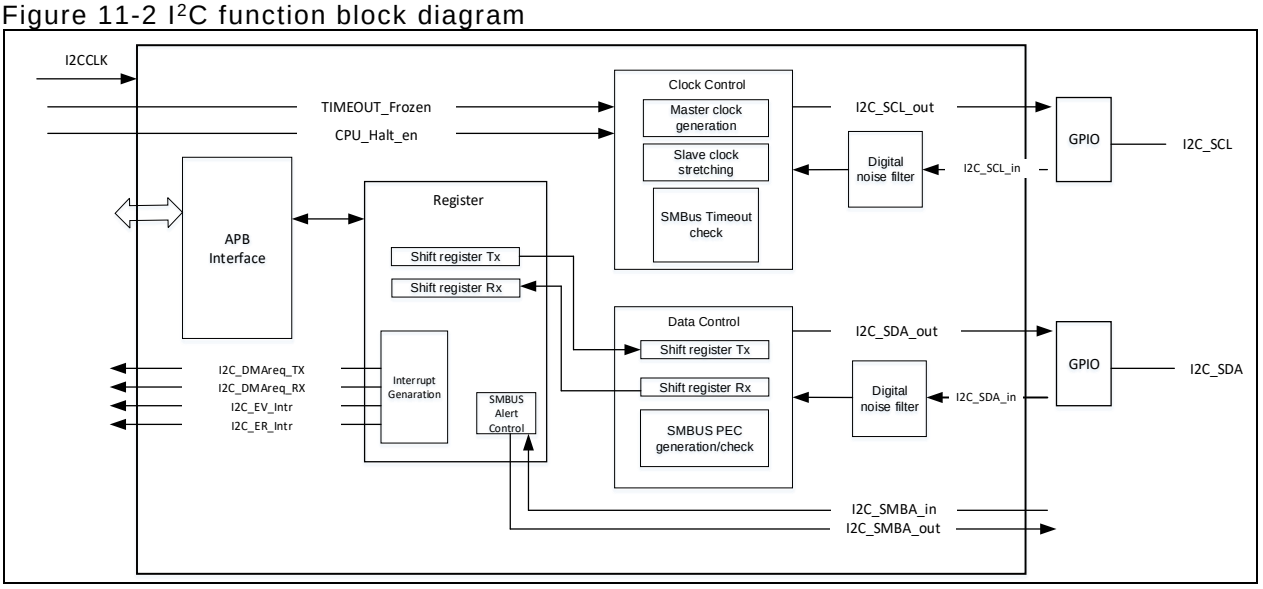
Figure 11-1 I²C bus protocol



11.4 I²C interface

Figure 11-2 shows the block diagram of I²C function.

Figure 11-2 I²C function block diagram



1. Operating mode

I²C bus interface can operate both in master mode and slave mode. Switching from master mode to slave mode, vice versa, is supported as well. By default, the interface operates in slave mode. When GENSTART=1 is set (Start condition is activated), the I²C bus interface switches from slave mode to master mode, and returns to slave mode automatically at the end of data transfer (Stop condition is triggered).

2. Communication process

- Master mode communication:
  1. Start condition generation
  2. Address transmission
  3. Data Tx or Rx
  4. Stop condition generation
  5. End of communication
- Slave mode communication:
  1. Wait until the address is matched.
  2. Data Tx or Rx
  3. Wait for the generation of Stop condition
  4. End of communication

3. Digital filter capability

The digital filter is available on both SCL and SDA lines. It is enabled by setting the DFLT[3: 0] bit in the I2C_CTRL1 register to reduce noise on bus on a large scale. The filter time is DLFT x t_{I2C_CLK}.

The digital filter is not allowed to be altered when the I²C is enabled.

4. Address control

Both master and slave support 7-bit and 10-bit addressing modes.

Slave address mode:

- In 7-bit mode (ADDR1MODE=0)
  - ADDR1EN=1, ADDR2EN=0 stands for a single address mode: only matches OADDR1
  - ADDR1EN=1, ADDR2EN=1stands for dual address mode: matches OADDR1 and OADDR2
- In 10-bit mode (ADDR1MODE=1)
  - Only supports a single address mode (ADDR1EN=1, ADDR2EN=0), matches OADDR1

Slave address masking capability

The Slave address 2 (OADDR2) is maskable, which is done by setting the ADDR2MASK[2: 0].

- 0: Address bit [7: 1]
- 1: Address bit [7: 2]
- 2: Address bit [7: 3]
- 3: Address bit [7: 4]
- 4: Address bit [7: 5]
- 5: Address bit [7: 6]
- 6: Address bit [7]
- 7: All addresses, excluding those reserved by I²C

Support special slave address:

- Broadcast call address (0b0000000x): This address is enabled when GCAEN=1.
- SMBus device default address (0b1100001x): This address is enabled for SMBus address resolution protocol in SMBus device mode (DEVADDREN = 1).
- SMBus master default address (0b0001000x): This address is enabled for SMBus master notification protocol in SMBus master mode (HADDREN = 1).
- SMBus alert address (0b0001100x): This address is enabled for SMBus alert response address protocol in SMBus master mode when SMBALERT = 1

Refer to SMBus2.0 protocol for more information.

Slave address matching procedure:

- Receive a Start condition
- Address matching
- The slave sends an ACK if address is matched.
- ADDR7F is set, with SIDR indicating the transmission direction
  - When SIDR =0, slave enters receiver mode, starting receiving data.
  - When SIDR =1, slave enters transmitter mode, starting transmitting data

5. Clock stretching capability

Clock stretching is enabled by default (STRETCH=0 in the I2C_CTRL1 register). The slave can hold the SCL line low for software operation. If the clock stretching capability is not supported by master, then the STRETCH must be set in the I2C_CTRL1 register. It should be noted that the clock stretching capability of I²C slave must be configured before the I²C peripherals are enabled.

Clock stretching capability enabled

I²C slave stretches the SCL clock in one of the following conditions:

- Address reception: When the address received by slave matches the local address enabled (ADDRF=1 in the I2C_STS), the SCL line is pulled down until the ADDRF is cleared by setting the ADDRC in the I2C_CLR
- Data reception: When the shift register has received another new byte before the data in the I2C_RXDT register is read, the I2C will hold the SCL bus low to wait for the software to read I2C_RXDT register
- Data transmission: If no data is written when the ADDRF is cleared, TDBE= 1 in the I2C_STS, then the SCL line will be pulled down until the data is written to the I2C_TXDT
- Data transmission: If no data is written to the I2C_TXDT after the completion of the previous data transfer, the SCL line will be pulled down until data is written to the I2C_TXDT
- When slave data control mode is selected (SCTRL=1 in the I2C_CTRL1) and RLDEN=1 in the I2C_CTRL2 register, if TCRLD = 1, indicating the completion of the last data transfer, then the TCRLD will be cleared by hardware so as to release the SCL line before a non-zero value is written to the CNT bit in the I2C_CTRL2 register

Clock stretching capability disabled

The SCL clock is disabled when STRETCH=1 in the I2C_CTRL1 register, with the following conditions worth our notice:

- Address reception: The SCL clock is not stretched when the address received by slave matches the local address enabled (ADDRF=1 in the I2C_STS)
- Data reception: If there is data to be read in the I2C_RXDT register before the next ACK signal, an overflow will occur, and the OUF bit will also be set in the I2C_STS register
- Data transmission: If no data is written to the I2C_TXDT register after the completion of the previous data transfer, an underflow will occur, and the OUF will also be set in the I2C_STS register

11.4.1 I²C timing control

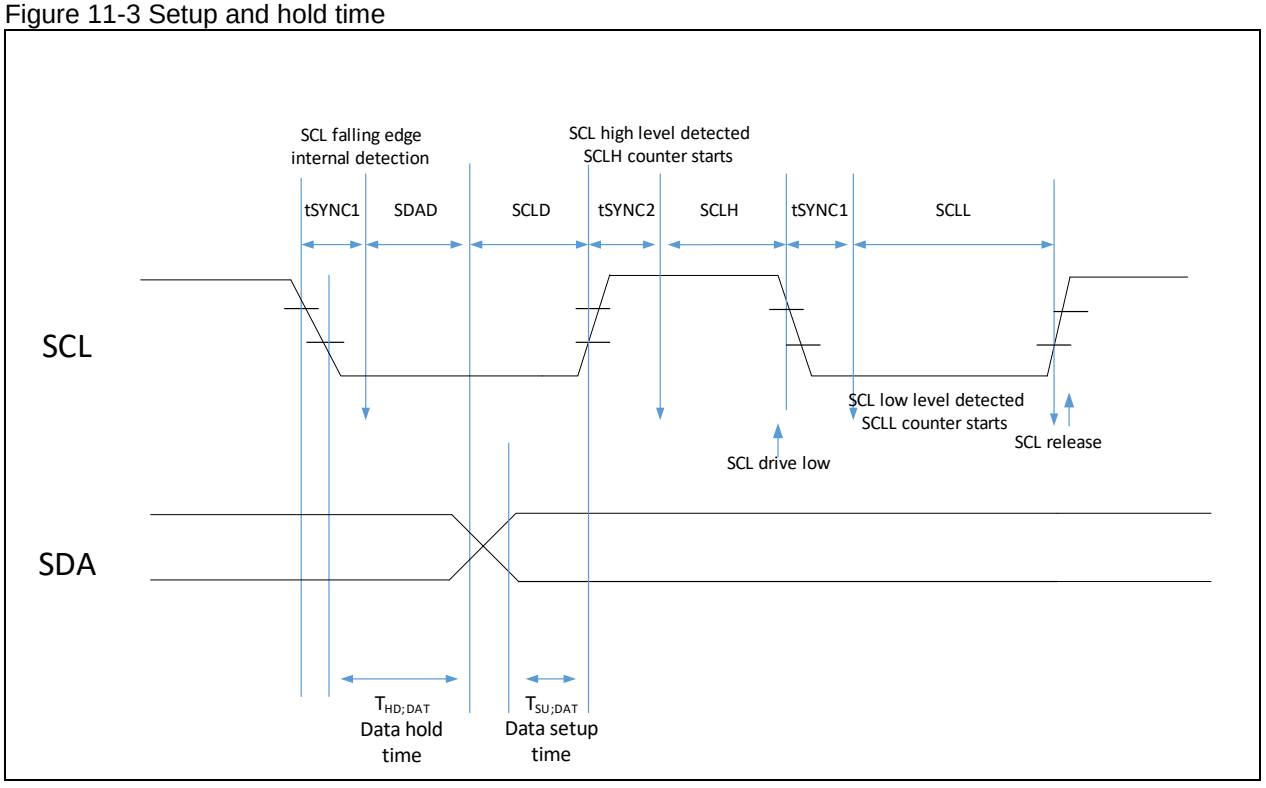
I²C core is clocked by I2C_CLK whereas the I2C_CLK is clocked by PCLK1. The PCLK1 should be set to be less than 4/3 SCL cycles.

The corresponding bits in the I2C_CLKCTRL register are used for timing configuration.

— DIV[7: 0]: I²C clock divider
— SDAD[3: 0]: Data hold time ($t_{HD;DAT}$)
— SCLD[3: 0]: Data setup time ($t_{SU;DAT}$)
— SCLH[7: 0]: SCL high
— SCLL[7: 0]: SCL low

Note: Timing configuration cannot be modified once the I²C is enabled.

Figure 11-3 Setup and hold time



It is possible to configure data hold time ($t_{HD;DAT}$) and data setup time ($t_{SU;DAT}$) freely by setting the DIV[7: 0], SDAD[3: 0] and SCLD[3: 0] in the I2C_CLKCTRL register.

- Data hold time ($t_{HD;DAT}$): refers to the duration from SCL falling edge to SDA output

  $t_{HD;DAT} = t_{SDAD} + t_{SYNC}$

  $t_{SDAD} = SDAD \times (DIV + 1) \times t_{I2C_CLK}$

  $t_{SYNC} = (DLFT + 3) \times t_{I2C_CLK} - t_f$

  t_{SYNC} consists of three parts:
  — SCL falling edge time t_f
  — Digital filter input latency ($DLFT \times t_{I2C_CLK}$)
  — Synchronization delay between SCL and I2C_CLK (2~3 I2C_CLK cycles)
- Data setup time ($t_{SU;DAT}$): refers to the duration from SDA output to SCL rising edge

  $t_{SU;DAT} = SCLD \times (DIV+1) \times t_{I2C_CLK} - t_r$

In master mode, the width of SCL signals (high and low) can be configured freely by setting the DIV[7:

0], SCLH[7: 0] and SCLL[7: 0] in the I2C_CLKCTRL register.

SCL low: When the SCL low signal is detected, the internal SCLL counter starts counting until it reaches the SCLL value. At this point, the SCL line is released and become high.

SCL high: When the SCL high signal is detected, the internal SCLH counter starts counting. When the counter value reaches the SCLH value, the SCL line is pulled low. In the process of SCL remaining high, if it is pulled low by external bus, the internal SCLH counter will stop counting and start counting in SCL low mode, laying the foundation for clock synchronization.

- SCL high signal width

  t_{HIGH} = (SCLH + 1) x (DIV + 1) x t_{I2C_CLK}
- SCL low signal width

  T_{Low} = (SCLL + 1) x (DIV + 1) x t_{I2C_CLK}

Table 11-1 I²C timing specifications

Parameter		Standard mode		Fast mode		Fast mode plus		SMBus	
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
f_{SCL} (kHz)	SCL clock frequency		100		400		1000		100
t_{LOW} (us)	SCL clock low	4.7		1.3		0.5		4.7	
t_{HIGH} (us)	SCL clock high	4.0		0.6		0.26		4.0	50
$t_{HD;DAT}$ (us)	Data hold time	0		0	0.9	0	0.45	300	
$t_{SU;DAT}$ (ns)	Data setup time	250		100		50		250	
tr (ns)	SCL/SDA rising edge		1000		300		120		1000
tf (ns)	SCL/SDA falling edge		300		300		120		300

11.4.2 Data transfer management

Data transfer counter is available in the I²C interface to control communication flow. It is mainly used for:

- NACK transmission: master reception mode
- STOP transmission: master reception/transmission modes
- RESTART generation: master reception/transmission modes
- ACK control: slave mode (SMBus)
- PEC transmission/reception: master/slave modes

Generally, the data transfer management counter (by setting the CNT[7:0] in the I2C_CTRL2) is applicable to master mode. It is disabled in slave mode. This counter is used only in SMBus mode for the ACK control and PEC reception of each byte by the slave. In SMBus mode, the slave enables data counter with the SCTRL bit in the I2C_CTRL2 register.

Byte control through master

The CNT[7:0] bit in the I2C_CTRL2 register is used to configure the number of bytes to be transferred, ranging from 1 to 255. If the number of data to be transferred is greater than 255, then the RLDEN bit has to be set in the I2C_CTRL2 register to enable reload mode. The following configuration processes are described in two aspects:

- ≤255 bytes, for example, the number of data to be transferred is 100 bytes
  - Step 1: Disable reload mode by setting RLDEN=0
  - Step 2: Set CNT[7:0]=100
- >255 bytes, for example, the number of data to be transferred is 600 bytes
  - Step 1: Enable reload mode by setting RLDEN=1
  - Step 2: Set CNT[7:0]=255, the remaining bytes are 600-255=345 bytes
  - Step 3: After the completion of 255-byte data transfer, the TCRLD is set in the I2C_STS register, and then configure CNT[7:0]=255 for continuous transfer, the remaining bytes are 345-255=90

- Step 4: After the completion of the second 255-byte data transfer, the TCRLD is set in the I2C_STS register, and then set RLDEN=0 to disable reload mode before setting CNT[7:0]=90 for continuous transfer.

There are two ways to stop the last data transfer (RLDEN=0, reload mode is disabled)

- Stop data transfer automatically (ASTOPEN=1 in the I2C_CTRL2)
  - When the number of data programmed in the CNT[7:0] bit has been fully transferred, the master will automatically send a STOP condition
- Stop data transfer by software (ASTOPEN=0 in the I2C_CTRL2)
  - When the number of data programmed in the CNT[7:0] has been fully transferred, the TDC will be set in the I2C_STS register, and the SCL, at this point, will be pulled low, an interrupt generated if TDCIEN is enabled. In this case, it is possible to send a STOP condition by setting GENSTOP=1 in the I2C_CTRL2 register, or send a RESTART condition by setting GENSTART=1 in the I2C_CTRL2 register, before clearing TDC flag by software.

Byte control through slave

This feature is enabled by setting the SCTRL bit in the I2C_CTRL2 register so that the slave is able to control ACK/NACK signals of each byte independently.

- Proceed as below:
  - Set SCTRL=1 to enable Byte Control Through Slave
  - After the slave address is matched (ADDRF=1), enable reload mode by setting RLDEN=1, and then set CNT[7:0]=1
  - When a byte is received, the TCRLD is set in the I2C_STS register, and the slave will pull the SCL bus low between the 8th and 9th clock edges. At this point, the user can read the RXDT register and generate an ACK or NACK signal through the NACKEN bit in the I2C_CTRL2 register
  - When an NACK signal is generated, it indicates the end of communication
  - When an ACK signal is generated, the communication flow keeps going on. At this point, set CNT[7:0]=1, the TCRLD flag is cleared automatically by hardware, and the SCL bus is released for the reception of the next byte

As we know, the value in the CNT[7:0] bit is not limited to 1. If you want to receive 8 data, for example, but just want to control the ACK/NACK signals of the 8th data. Proceed as below: set CNT[7:0]=8, the slave will receive 7 consecutive data, with ACK signals sent. Once the 8th data reception is completed, the SCL bus is pulled low, and then proceed as above to select whether to send an ACK or NACK.

It should be noted that the clock stretching capability must be enabled (STRETCH=0 in the I2C_CTRL1 register) before selecting Byte Control Mode Through Slave.

Table 11-2 I^{2}C configuration table

Description	RLDEN	ASTOPEN	SCTRL
Master transmit/receive RESTART	0	0	×
Master transmit/receive STOP	0	1	×
Slave receive (control ACK/NACK of each byte)	1	×	1
Slave transmit/receive (ACK response to all bytes)	×	×	0

11.4.3 I^{2}C master communication flow

1. **I^{2}C clock initialization (by setting the I2C_CLKCTRL register)**
   - I^{2}C clock divider: DIV[7: 0]
   - Data hold time ($t_{HD;DAT}$): SDAD[3: 0]
   - Data setup time ($t_{SU;DAT}$): SCLD[3: 0]
   - SCL high duration: SCLH[7: 0]
   - SCL low duration: SCLL[7: 0]

   Artery_I2C_Timing_Configuration can be used to configure this register.

2. **Set the number of bytes to be transferred**
   - ≤255 bytes
     Disable reload mode by setting RLDEN=0 in the I2C_CTRL2 register
     Set CNT[7:0]=N in the I2C_CTRL2 register
   - >255 bytes
     Enable reload mode by setting RLDEN=1 in the I2C_CTRL register
     Set CNT[7:0]=255 in the I2C_CTRL2 register
     Remaining bytes N=N-255
3. **End of data transfer**
   - ASTOPEN=0: stop data transfer by software. After the completion of data transfer, the TDC is set in the I2C_STS register, and GENSTOP=1 or GENSTART=1 is written by software to send a STOP or START condition
   - ASTOPEN=1: data transfer is stopped automatically. A STOP condition is sent at the end of data transfer
4. **Set slave address**
   - Set slave address value (by setting the SADDR bit in the I2C_CTRL2 register)
   - Set slave address mode (by setting the ADDR10 bit in the I2C_CTRL2 register)
     ADDR10=0: 7-bit address mode
     ADDR10=1: 10-bit address mode
5. **Set transfer direction (by setting the DIR bit in the I2C_CTRL2 register)**
   - DIR=0: Master reception
   - DIR=1: Master transmission
6. **Start data transfer**
   When GENSTART=1 in the I2C_CTRL2 register, the master starts sending a START condition and slave address. After receiving the ACK from the slave, ADDRF=1 is asserted in the I2C_STS register. The ADDRF flag can be cleared by setting ADDRC=1 in the I2C_CLR register, and then data transfer starts.
7. **Master transmit**
   1. I2C_TXDT data register is empty, the shift register is empty, TDIS=1 in the I2C_STS register
   2. Writing 1 to the TXDT register, and data is immediately moved to the shift register
   3. TXDT register becomes empty, TDIS=1 again
   4. Writing 2 to the TXDT register, TDIS is cleared
   5. Repeat step 2 and 3 until the data in the CNT[7:0] is sent
   6. If TCRLD=1 (reload mode) in the I2C_STS register, the following two circumstances should be noted:
      Remaining bytes N>255: write 255 to the CNT bit, N=N-255, TCRLD is cleared, and data transfer continues
      Remaining bytes N≤255: Disable reload mode (RLDEN=0), write N to the CTN bit, TCRLD is cleared, and data transfer continues
8. **Master receive**
   1. After the slave address is matched, ADDRF=1 in the I2C_STS register, clear ADDRF flag by setting ADDRC=1 in the I2C_CLR register, and then it starts sending data
   2. After the reception of data, RDBF=1, read the RXDT register will clear the RDBF automatically
   3. Repeat step 2 until the reception of data programmed in the CNT[7:0] bit
   4. If TCRLD=1 (reload mode) in the I2C_STS, the following two circumstances should be noted:
      Remaining bytes N>255: write 255 to the CNT bit, N=N-255, TCRLD is cleared, and data transfer continues
      Remaining bytes N≤255: Disable reload mode (RLDEN=0), write N to the CTN bit, TCRLD is cleared, and data transfer continues
   5. After the reception of the last data, an NACK signal will be sent by master
9. **STOP condition**
   - STOP condition generation:
     ASTOPEN=0: TDC=1 in the I2C_STS register, set GENSTOP=1 to generate a STOP condition
     ASTOPEN=1: A STOP condition is generated automatically
   - Wait for the generation of a STOP condition, when a STOP condition is generated, STOPF=1

is asserted in the I2C_STS register. The STOPF flag can be cleared by setting STOPC=1 in the I2C_CLR register, and then transfer stops

When the host receives an NACK signal during transmission, then ACKFAIL is set in the I2C_STS register, and a STOP condition is sent to stop communication, whatever mode (either ASTOPEN=0 or ASTOPEN=1).

Master transmitter

Figure 11-4 I²C master transmission flow

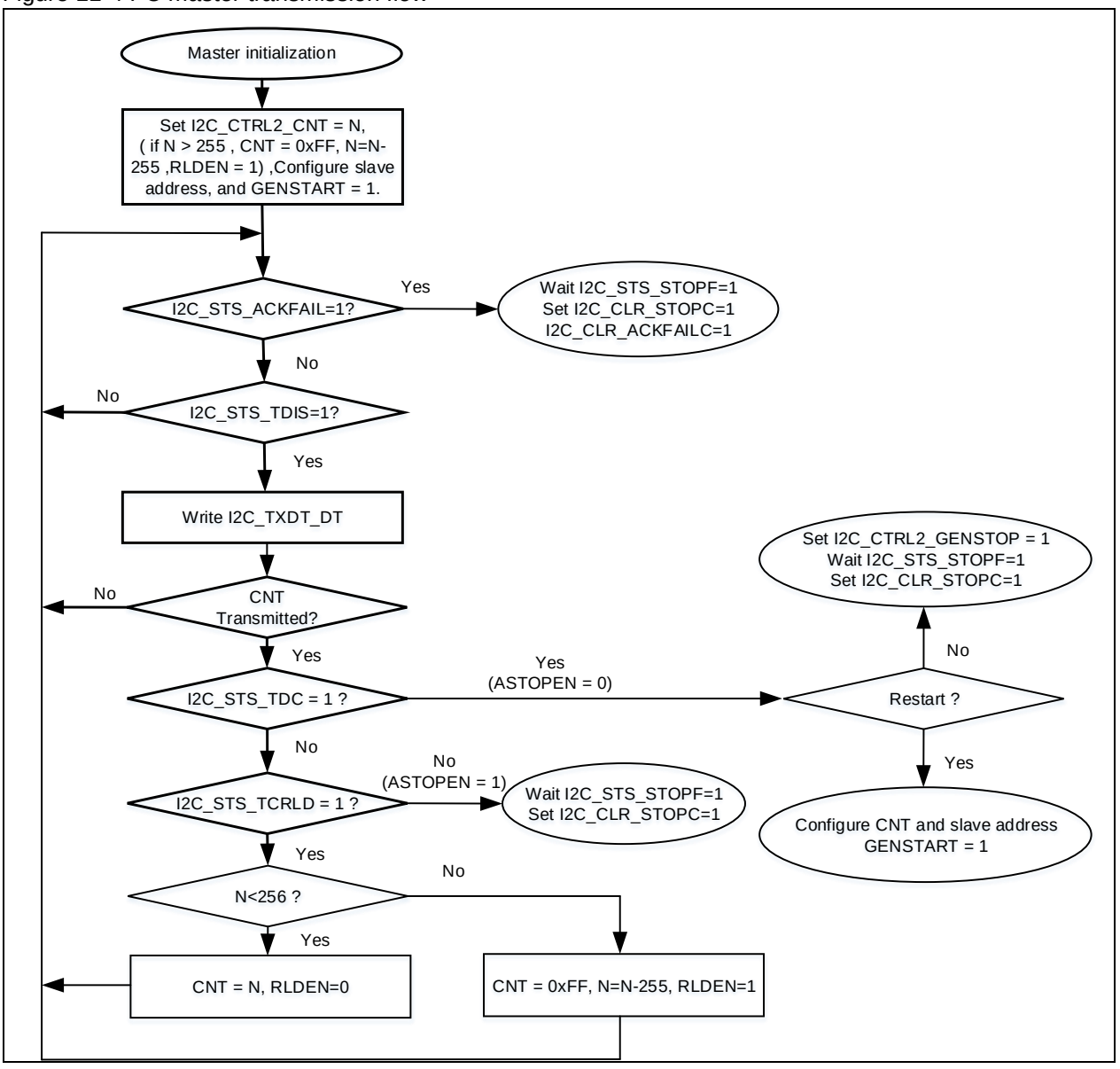
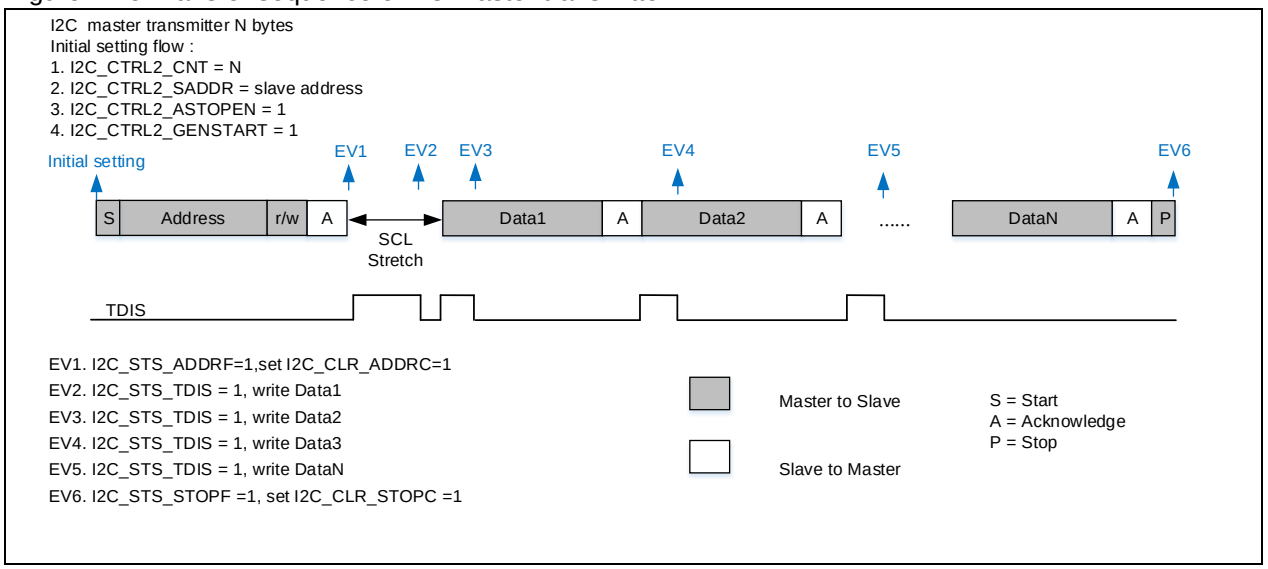


Figure 11-5 Transfer sequence of I²C master transmitter



Master receiver

Figure 11-6 I²C master receive flow

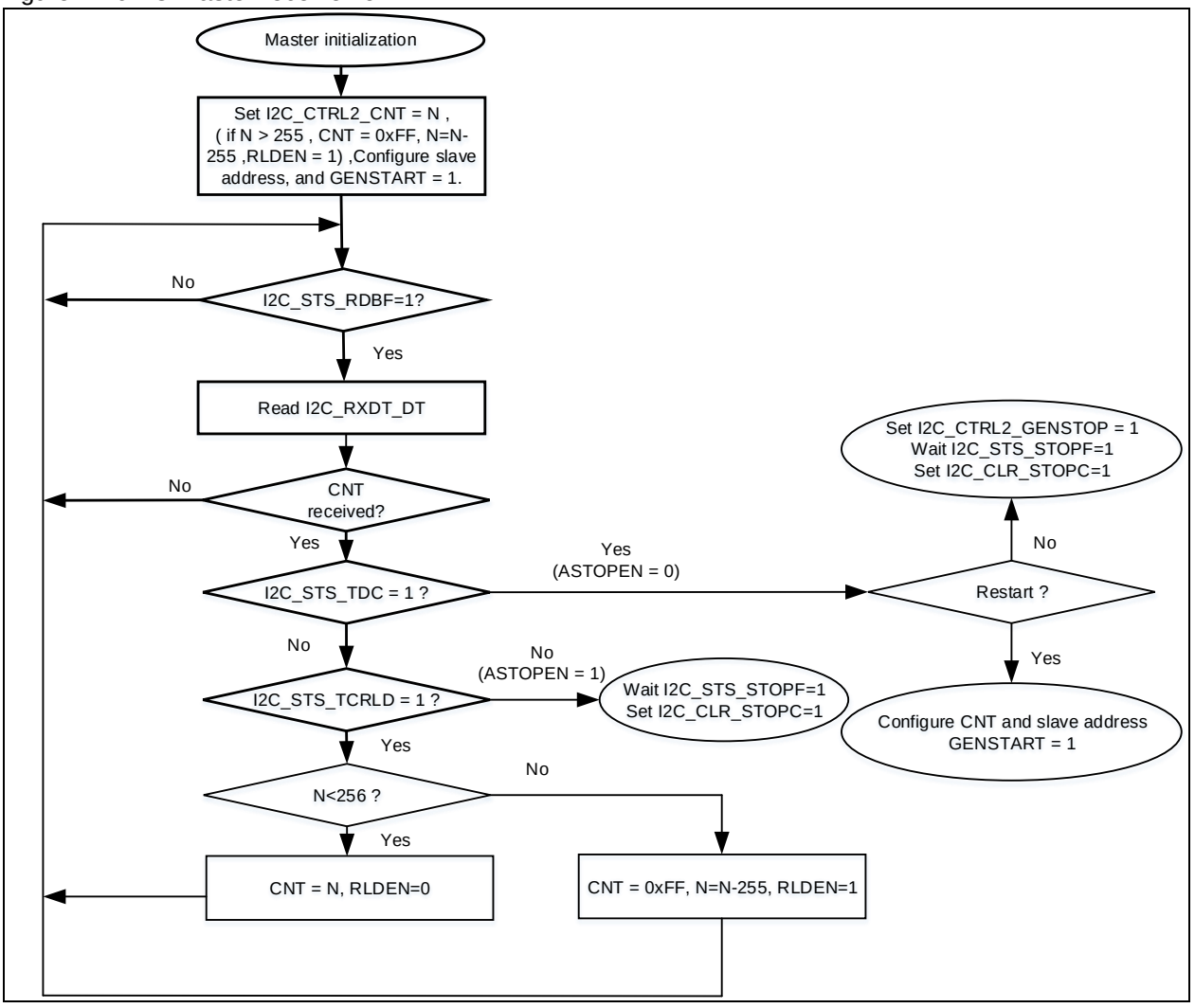
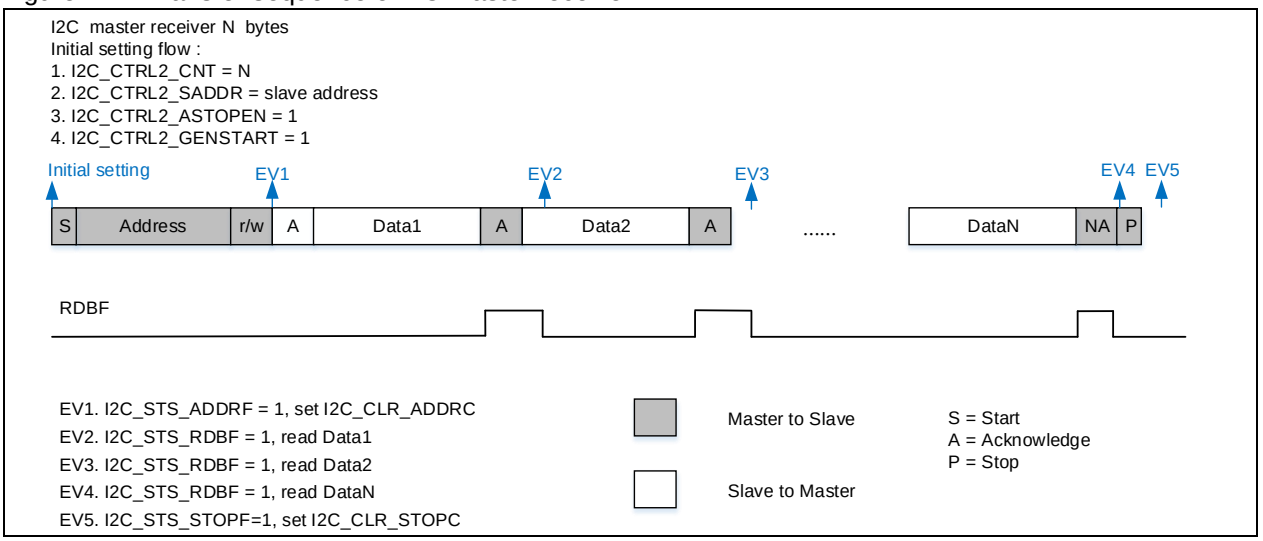


Figure 11-7 Transfer sequence of I²C master receiver



Master special transfer sequence

In 10-bit addressing mode, the READH10 bit of the I2C_CTRL2 register is used to generate a special timing. When READH10=1, the master sends data to the slave before read access to the slave, as shown in the figure below:

Operating method:

When ASTOPEN = 0, data is transferred from the master to the slave. At the end of data transfer, READH10=0 is asserted, and then the master starts receiving data from the slave.

Figure 11-8 10-bit address read access when READH10=1

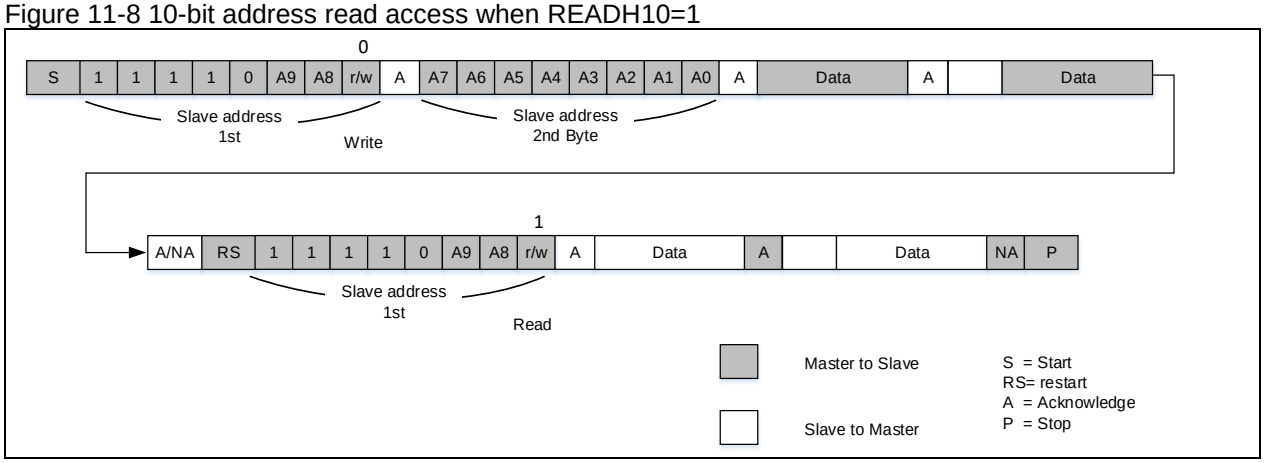
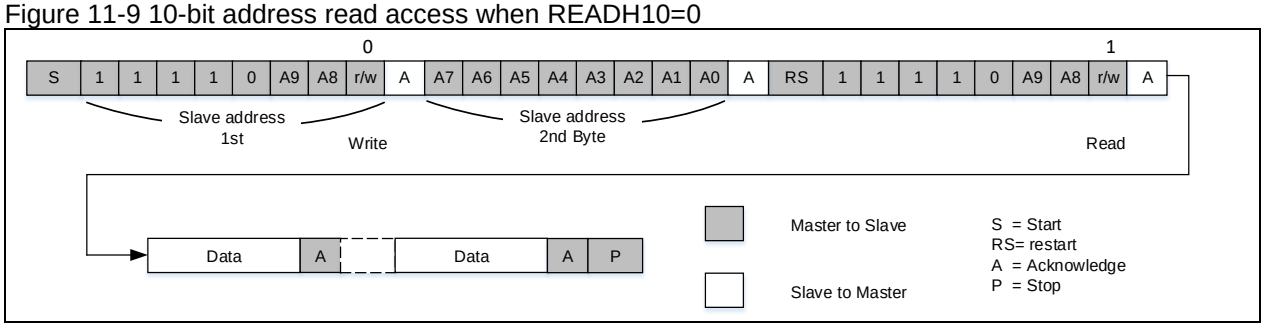


Figure 11-9 10-bit address read access when READH10=0



11.4.4 I²C slave communication flow

1. **I²C clock initialization (by setting the I2C_CLKCTRL register)**
   - I²C clock divider: DIV[7: 0]
   - Data hold time ($t_{HD;DAT}$): SDAD[3: 0]
   - Data setup time ($t_{SU;DAT}$): SCLD[3: 0]

   This register can be configured by means of Artery_I2C_Timing_Configuration tool.
2. **Set local address 1**
   - Set address mode:

     7-bit address: by setting ADDR1MODE = 0 in the I2C_OADDR register

     10-bit address: by setting ADDR1MODE = 1 in the I2C_OADDR register
   - Set address 1: by setting the ADDR1 bit in the I2C_OADDR1 register
   - Enable address 1: by setting ADDR1EN=1 in the I2C_OADDR1 register
3. **Set local address 2**
   - Set address 2: by setting the ADDR2 bit in the I2C_OADDR2 register
   - Set address 2 mask bit: by setting the ADDR2MASK bit in the I2C_OADDR2 register
   - Enable address 2: by setting ADDR2EN=1 in the I2C_OADDR2 register

   Note: Only 7-bit address mode is available in the address 2 mode. The ADDR2MASK bit is used to mask some address bits freely so that the slave can respond to some specific addresses. Refer to Section 14.2 for more information about the ADDR2MASK bit.

   In the case of using only one address, only address 1 needs to be configured, without the need of address 2 mode.
4. **Wait for address matching**

   When the local address is received, the ADDRF bit is set in the I2C_STS register. The data transfer direction can be obtained by read access to the SDIR bit in the I2C_STS register. When SDIR=0, it indicates that the slave is receiving data, whereas SDIR=1 indicates that the slave is sending data. The ADDR[6:0] bit of the I2C_STS register indicates what kind of address has been received, which is particularly helpful in the case when the dual address mode is used and the address 2 mode mask bit is set.

   Data transfer starts when the ADDRF is cleared by setting ADDRC=1 of the I2C_CLR register.
5. **Data transfer (slave transmission, clock stretching enabled, STRETCH=0)**

   After address matching:

   1. I2C_TXDT data register becomes empty, the shift register becomes empty, and TDIS=1 in the I2C_STS register
   2. Data is then transferred to the shift register after writing 1 to the TXDT register
   3. The TXDT register then becomes empty, and the TDIS is set again
   4. TDIS is cleared by writing 2 to the TXDT register
   5. Repeat step 3 and 4 until the completion of data transfer
   6. Wait for the generation of an NACK signal. Once received, the ACKFAILF is set in the I2C_STS register. The ACKFAILF flag is cleared by writing 1 to the ACKFAILC
   7. Wait for the generation of a STOP condition. Once received, the STOPF is set in the I2C_STS register. At the end of data transfer, the STOPF is cleared by writing 1 to the STOPC, transmission ends.

   In the case of the clock stretching being disabled (STRETCH=1), if data has not yet been written to the TXDT register before the transmission of the first bit of the to-be-transferred data (that is, before the generation of SDA edge), an underrun error may occur, and the OUF bit is set in the I2C_STS register, sending 0xFF to the bus.

   In order to write data in time, data must be written to the DT register first before communication, in two different ways:

   - Write operation through software: Clear the TXDT register by setting the TDBE bit through

software, and then write the first data to the TXDT register, the TDBE is cleared

- Write operation through interrupts or DMA: Clear the TXDT register by setting the TDBE bit through software, then set the TDIS bit to generate a TDIS event, which generates an interrupt or DMA request. At this point, data is written to the TXDT register using DMA or interrupt functions.

6. Data transfer (slave receive, clock stretching enabled, STRETCH=0)

After address matching:

1. I2C_RXDT register becomes empty, the shift register becomes empty, and RDBF=0 in the I2C_STS register
2. Upon the receipt of data, RDBF=1; The RDBF is cleared by read operation to the RXDT register
3. Repeat step 2 until the completion of all data transfer
4. Wait for the generation of a STOP condition. Once received, the STOPF is set in the I2C_STS register. The STOPF can be cleared by writing 1 to the STOPC bit in the I2C_CLR register, transfer ends.

In slave receive mode, the slave byte control mode can be used for data reception. This mode allows to control ACK/NACK signals of each byte received. This mode is typically available in SMBus protocol. Refer to Section 114.2 for more information about this mode.

Note that the slave must read the received data in the case of the clock stretching being disabled (STRETCH=1). If one-byte data has been received and data is not read yet before the end of the next data reception, an overrun error occurs, setting the OUF bit in the I2C_STS register, and sending NCAK.

An interrupt will be generated if the corresponding interrupt enable bit is enabled. For more information about interrupt generation, refer to the interrupt chapter.

Slave transmission

Figure 11-10 I²C slave transmission flow

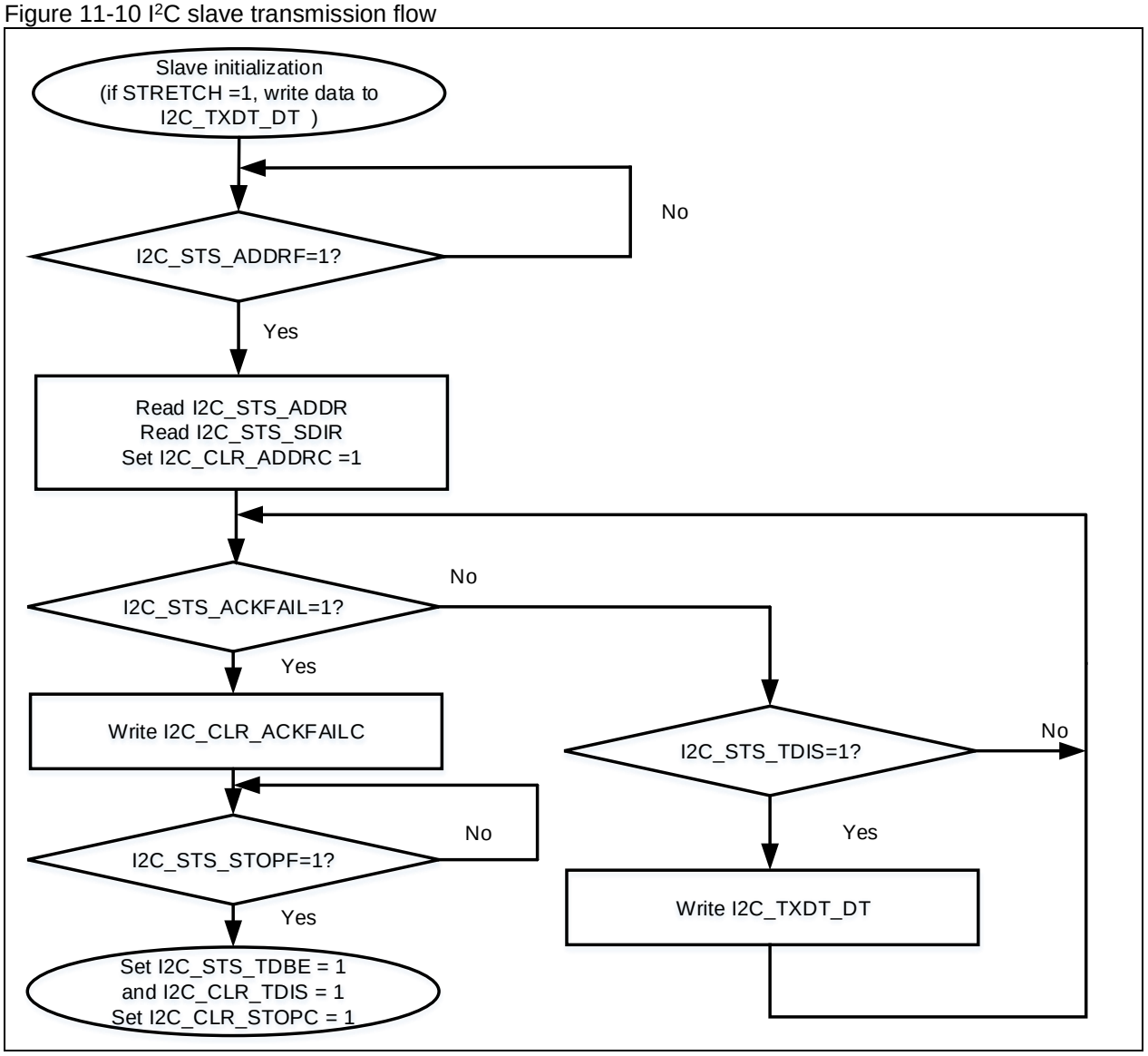
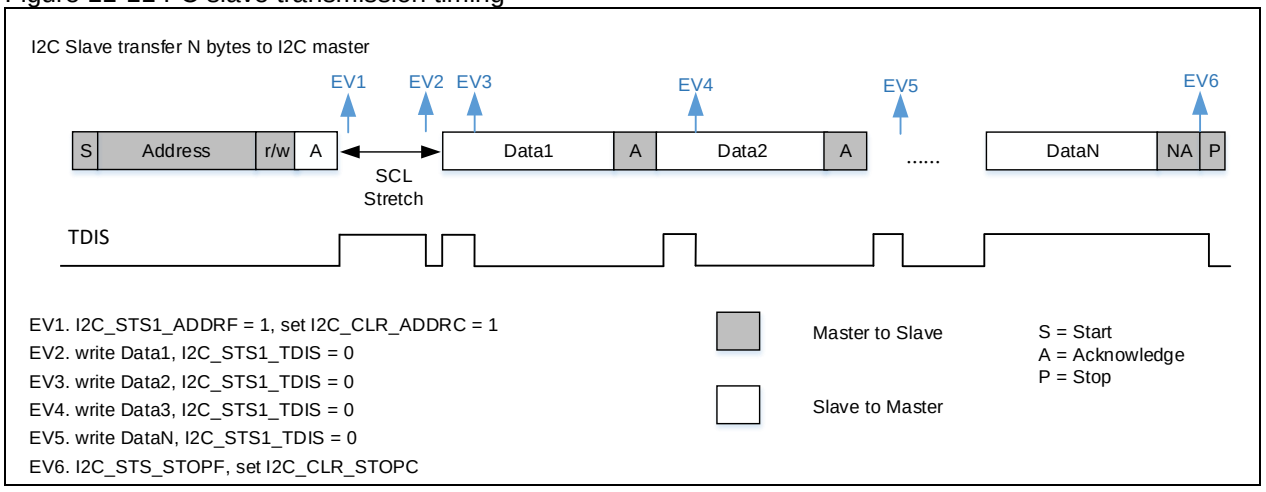


Figure 11-11 I²C slave transmission timing



Slave receive

Figure 11-12 I²C slave receive flow

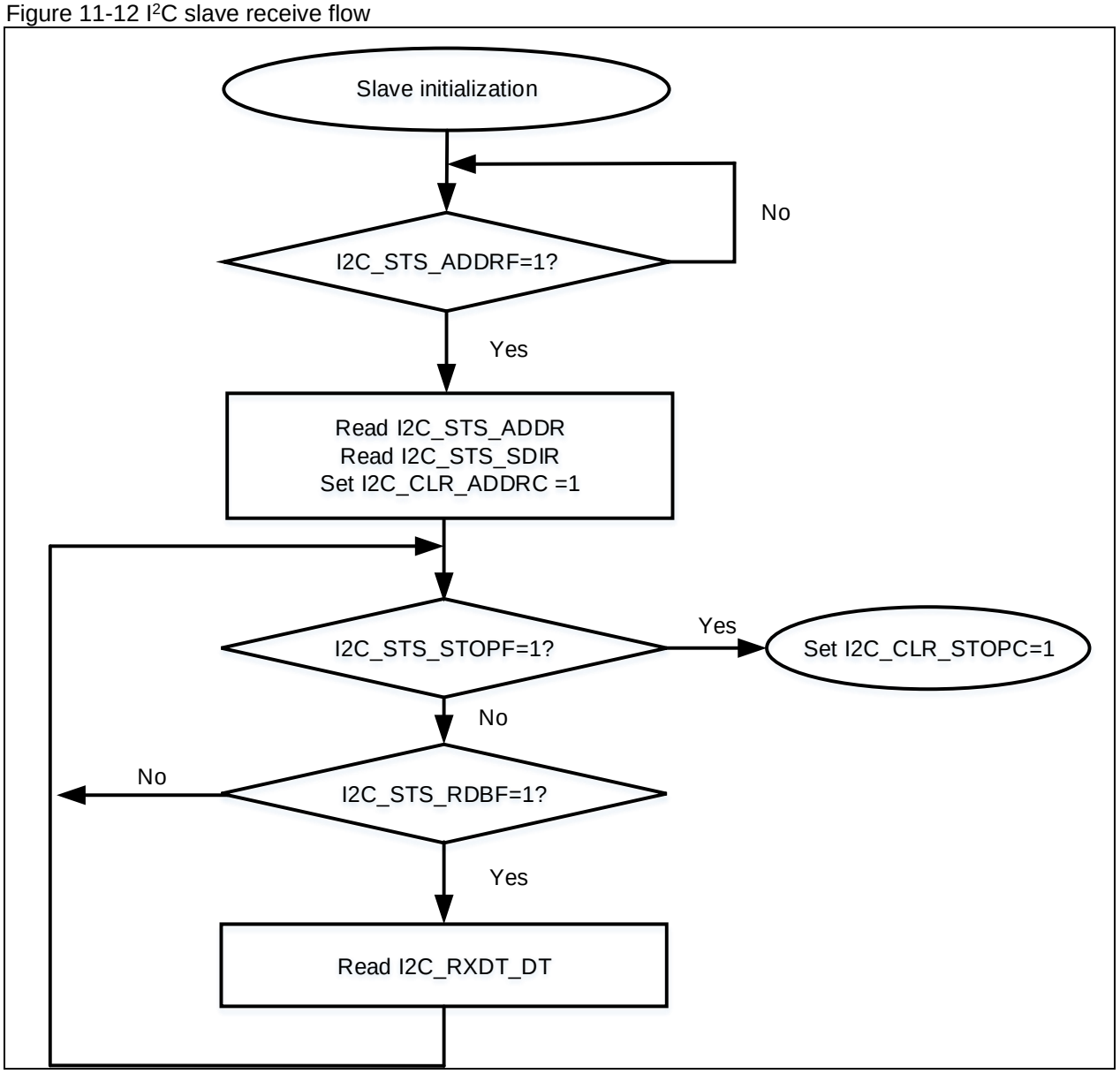
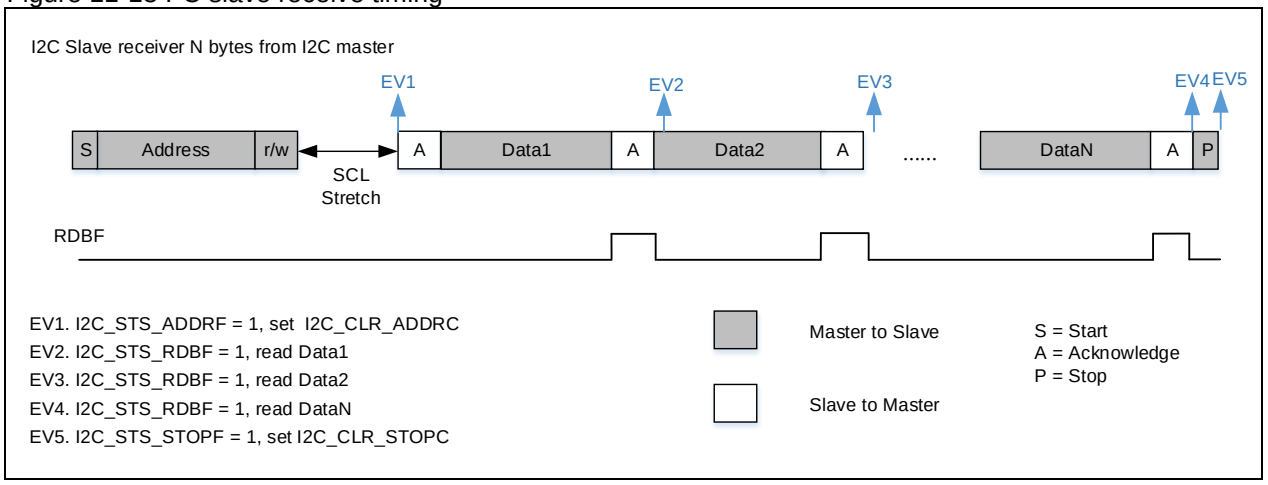


Figure 11-13 I²C slave receive timing



11.4.5 SMBus

The System Management Bus (SMBus) is a two-wire interface through which various devices can communicate with each other. It is based on I²C. With SMBus, the device can provide manufacturer information, tell the system its model/part number, report different types of errors and accept control parameters and so on. For more information, refer to SMBus 2.0 protocol.

Differences between SMBus and I²C

1. SMBus requires a minimum speed of 10 kHz for the purpose of management and monitor. It is quite easy to know whether the bus is in Idle state or not as long as a parameter is input while running on a certain transmission speed, without the need of detecting the STOP signals one after another, or even keeping STOP and other parameter monitor. There is no limit for I²C.
2. SMBus transmission speed ranges from 10 kHz to 100 kHz. In contrast, I²C has no minimum requirement, and its maximum speed varies from one mode to another, namely, 100 kHz in standard mode and 400 kHz in fast mode.
3. After reset, SMBus needs timeout, but there is no limit for I²C in this regard.

SMBus address resolution protocol (ARP)

SMBus address conflicts can be resolved by dynamically assigning a new unique address to each device. Refer to SMBus 2.0 protocol for more information about ARP.

Setting the DEVADDREN bit in the I2C_CTRL1 register can enable the I²C interface to recognize the default device address (0b1100001x). However, unique device identifier (UDID) and the detailed protocol implementation should be handled by software.

SMBus host notify protocol

The slave device can send data to the master device through SMBus host notify protocol. For example, the slave can notify the host to implement ARP with this protocol. Refer to SMBus 2.0 protocol for details on SMBus host notify protocol.

In host mode (HADDREN =1), the I²C interface is enabled to recognize the 0b0001000x (default host address)

SMBus Alert

SMBALERT is an optional signal that connects the ALERT pin between the host and the salve. With this signal, the slave notifies the host to access the slave. SMBALERT is a wired-AND signal. For more information about SMBus Alert, refer to SMBus2.0 protocol.

The detailed sequences are as follows:

SMBus host:

1. Enable SMBus Alert mode by setting SMBALERT=1
2. Enable ALERT interrupt if necessary
3. When an alert event occurs on the ALERT pin (ALERT pin changes from high to low)
4. The host will generate ALERT interrupt if enabled
5. The host then processes the interrupt and accesses to all devices through ARA (Alert Response Address 0001100x) so as to get the slave addresses. Only the devices with pulled-down SMBALERT can acknowledge ARA.
6. The host then continues to operate based on the slave addresses available.

SMBus slave:

1. When an alert event occurs and the ALERT pin changes from high to low (SMBALERT=1), the slave responses to ARA (Alert Response Address) address (0001100x)
2. Wait until the host gets the slave addresses through ARA
3. Report its own address, but it continues to wait if the arbitration is lost.
4. Address is reported properly, and the ALERT pin is released (SMBALERT=0).

Packet error checking (PEC)

Packet error checking (PEC) is used to guarantee the correctness and integrity of data transfer. This is done by using CRC-8 polynomial:

$C(x) = x^8 + x^2 + x + 1$

PEC calculation is enabled when PECEN=1 to check address and data.

PEC transfer:

- Host: PEC transfer is enabled by setting PECTEN=1 in the I2C_CTRL2 register. The host sends a PEC as soon as the number of data transfer reaches N-1 (CNT=N)
- Slave: PEC transfer is enabled by setting PECTEN=1 in the I2C_CTRL2 register. When the number of data transfer reaches N-1 (CNT=N), the slave will consider the Nth data as a PEC and check it. A NACK will be sent if the PEC checking result is not correct, setting the PECERR flag in the I2C_STS register. In case of slave transmission mode, a NACK must follow the PEC whatever the checking result

SMBus timeout

The SMBus protocol specifies three timeout detection modes:

- Low level timeout ($t_{TIMEOUT}$): The time duration when the SCL is kept low in a single mode (taking into account master/slave device, however actively or passively pulled low)
- Cumulative timeout for a slave device at low level ($t_{LOW:SEXT}$): The cumulative time duration when the SCL is pulled low by a slave device during the period from a START condition to a STOP condition
- Cumulative timeout for a master device at low level ($t_{LOW:MEX}$): The cumulative time duration when the SCL is pulled low by a master device during the period from the ACK of the last byte to the 8th bit of the next byte (a single byte)

It should be noted that both $t_{LOW:SEXT}$ and $t_{LOW:MEXT}$ only deal with the time when they set themselves low level, excluding the time when they are pulled low by external sources. In contrast, both of these cases are considered in the calculation of $t_{TIMEOUT}$.

Table 11-3 SMBus timeout specification

Type of timeout	Min	Max	Unit
$t_{TIMEOUT}$	25	35	ms
$t_{LOW:SEXT}$	-	25	ms
$t_{LOW:MEXT}$	-	10	ms

The I²C peripherals embeds two counters for timeout detection, which can be configured through the I2C_TIMEOUT register. When a timeout event occurs, the TMOUT is set in the I2C_STS register. The TMOUT bit can be cleared by writing 1 to the TMOUTC bit in the I2C_CLR register.

- EXTTIME: This is used to the cumulative timeout detection for master/slave devices at low level
  Timeout duration= (EXTTIME + 1) x 2048 x T_{I2C_CLK}
- TOTIME: This is used for clock level timeout detection, selected through the TOMODE bit.
  TOMODE=0: Low level timeout detection, timeout duration =(TOTIME + 1) x 2048 x T_{I2C_CLK}
  TOMODE=1: High level timeout detection, timeout duration =(TOTIME + 1) x 4 x T_{I2C_CLK}

Table 11-4 SMBus timeout detection configuration

Type of timeout	Other configurations	Enable bit	Timeout calculation
$t_{TIMEOUT}$	TOMODE=0	TOEN=1	(TOTIME + 1) x 2048 x T_{I2C_CLK}
$t_{LOW:SEXT}$	-	EXTEN=1	(EXTTIME + 1) x 2048 x T_{I2C_CLK}
$t_{LOW:MEXT}$	-	EXTEN=1	(EXTTIME + 1) x 2048 x T_{I2C_CLK}

Slave receive byte control

In slave receive mode, the slave receive byte control mode (SCTRL=1) can be used to control ACK/NACK signals of each received byte. Refer to section 11.4.2 for more information.

Table 11-5 SMBus mode configuration

Transfer mode	PECEN	PECTEN	RLDEN	ASTOPEN	SCTRL
Master receive/transmit+STOP	1	1	0	1	-
Master receive/transmit +RESTART	1	1	0	0	-
Slave receive	1	1	1	-	1
Slave transmit	1	1	0	-	-

How to use the interface in SMBus mode

1. Set SMBus default address acknowledgement:
   HADDREN=1: Master default address acknowledged (0b0001000x)
   DEVADDREN=1: Device default address acknowledged (0b1100001x)
2. Configure PEC
3. Slave receive byte control mode can be enabled (with SCTRL bit in the I2C_CTRL1) in slave mode, if necessary
4. Other configurations follow the I^2C

However, the detailed SMBus protocol implementation should be handled by software, since the I^2C interface is only enabled to recognize the addresses of SMBus protocols.

11.4.6 SMBus master communication flow

The SMBus is similar to the I^2C in terms of master communication flow.

1. I^2C clock initialization (by setting the I2C_CLKCTRL register)

- I^2C clock divider: DIV[7: 0]
- Data hold time ($t_{HD;DAT}$): SDAD[3: 0]
- Data setup time ($t_{SU;DAT}$): SCLD[3: 0]
- SCL high duration: SCLH[7: 0]
- SCL low duration: SCLL[7: 0]

The register can be configured by means of Artery_I2C_Timing_Configuration tool.

2. SMBus-related initialization

- Select SMBus host: host default address acknowledged (0b0001000x) by setting HADDREN=1
- Enable PEC calculation: Set PECEN=1 in the I2C_CTRL1 register
- Enable PEC transfer: Set PECTEN=1 in the I2C_CTRL2 register

3. Set the number of bytes to be transferred

- Disable reload mode by setting RLDEN=0 in the I2C_CTRL2 register
- Set CNT[7:0]=N in the I2C_CTRL2 register

The number of bytes to be transferred is <255 in SMBus mode at one time.

4. End of data transfer

- ASTOPEN=0: stop data transfer by software. After the completion of data transfer, the TDC is set in the I2C_STS register, and GENSTOP=1 or GENSTART=1 is written by software to send a STOP or START condition
- ASTOPEN=1: data transfer is stopped automatically. A STOP condition is sent at the end of data transfer

5. Set slave address

- Set slave address value (by setting the SADDR bit in the I2C_CTRL2 register)

- Set 7-bit slave address mode (by setting the ADDR10=0 in the I2C_CTRL2 register)

6. Set transfer direction (by setting the DIR bit in the I2C_CTRL2 register)

- DIR=0: Master reception
- DIR=1: Master transmission

7. Start data transfer

In case of GENSTART=1 in the I2C_CTRL2 register, the master starts sending a START condition and slave address. After receiving the ACK from the slave, ADDRF=1 is asserted in the I2C_STS register. The ADDRF flag can be cleared by setting ADDRC=1 in the I2C_CLR register, and then data transfer starts.

8. Master transmit

1. I2C_TXDT data register is empty, the shift register is empty, TDIS=1 in the I2C_STS register
2. Writing 1 to the TXDT register, and data is immediately moved to the shift register
3. TXDT register becomes empty, TDIS=1 again
4. Writing 2 to the TXDT register, TDIS is cleared
5. Repeat step 2 and 3 until the specified data (N-1) is sent
6. The master will automatically transmit the Nth data, that is, PEC.

9. Master receive

1. After the reception of data, RDBF=1, read the RXDT register will clear the RDBF automatically
2. Repeat step 1 until the reception of the specified data (N). The Nth data is set as PEC. A NACK is automatically sent after the receipt of the Nth data (PEC) whatever the PEC result.

10. STOP condition

- STOP condition generation:

  ASTOPEN=0: TDC=1 in the I2C_STS register, set GENSTOP=1 to generate a STOP condition

  ASTOPEN=1: A STOP condition is generated automatically
- Wait for the generation of a STOP condition, when a STOP condition is generated, STOPF=1 is asserted in the I2C_STS register. The STOPF flag can be cleared by setting STOPC=1 in the I2C_CLR register, and then transfer stops

SMBus master transmission flow

Figure 11-14 SMBus master transmission flow

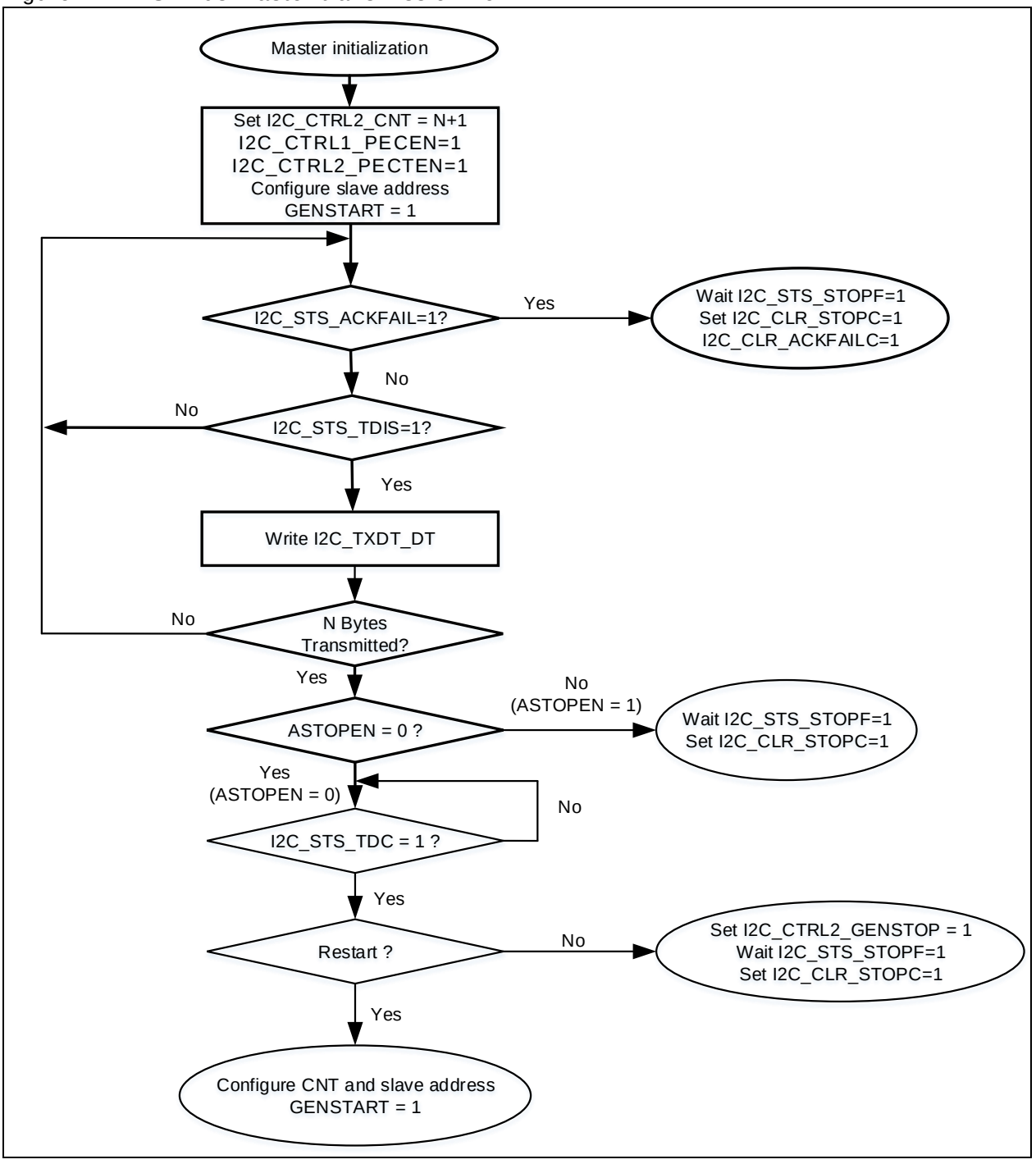
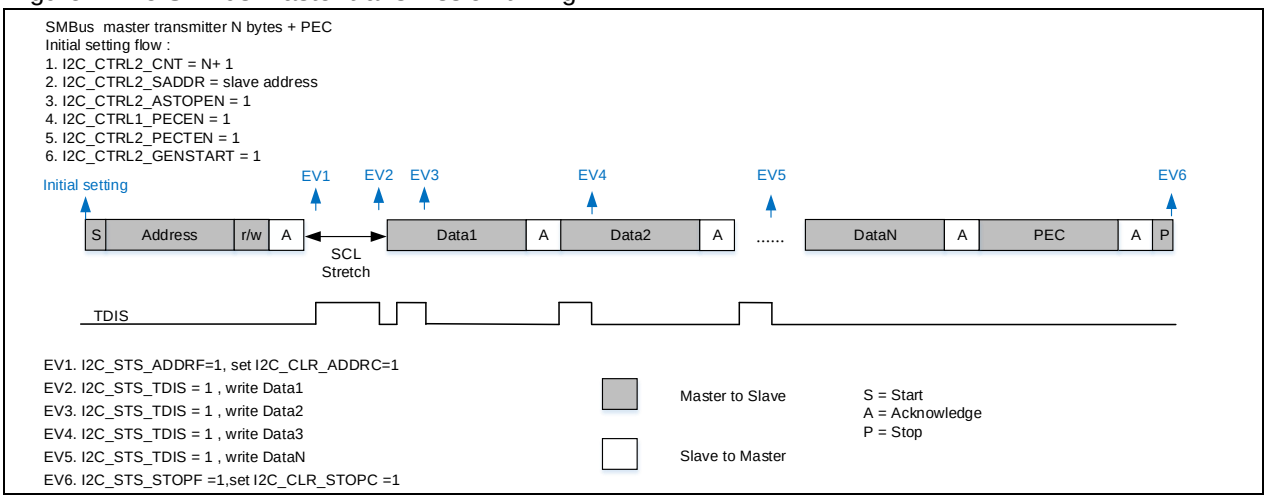


Figure 11-15 SMBus master transmission timing



SMBus master receive flow

Figure 11-16 SMBus master receive flow

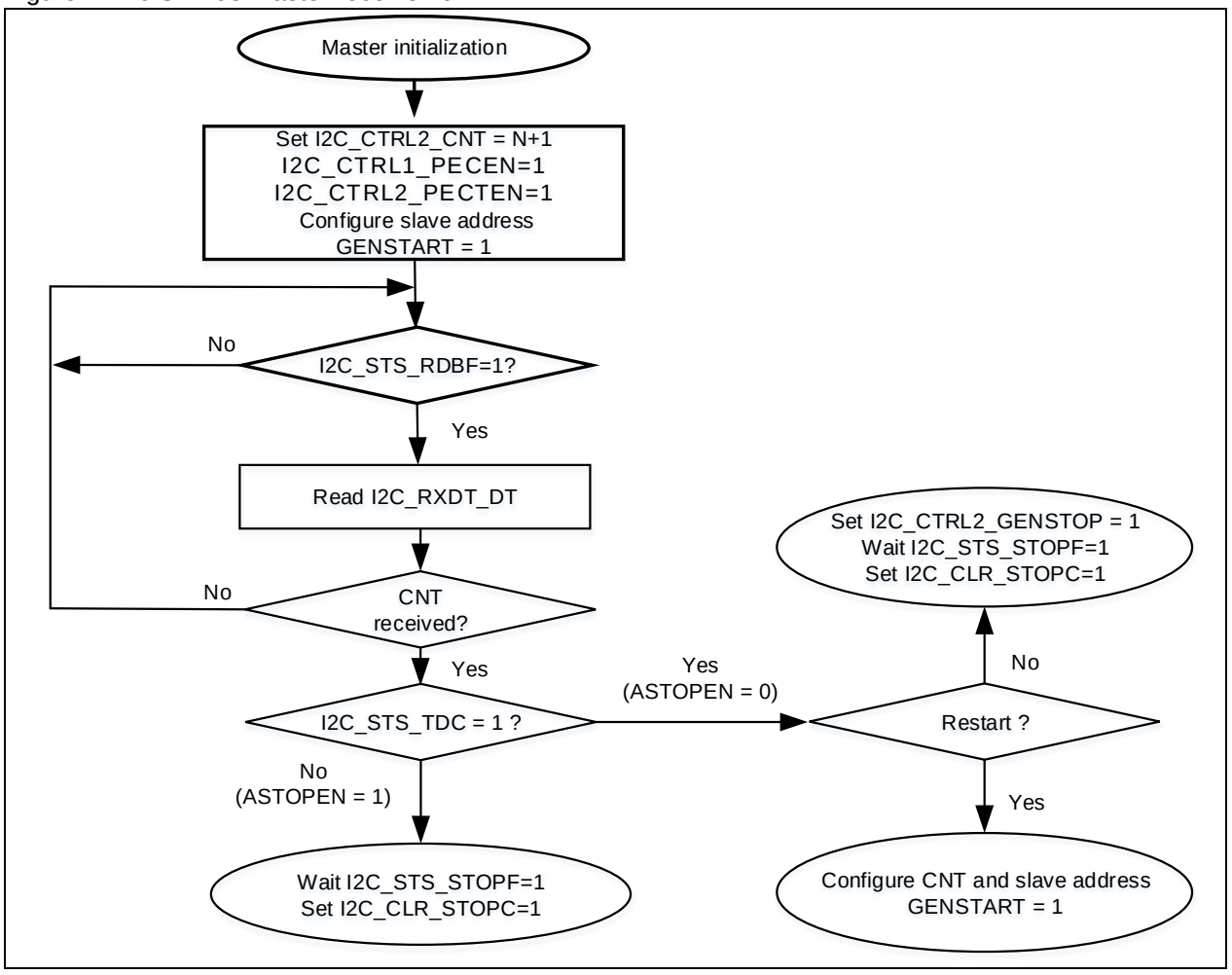
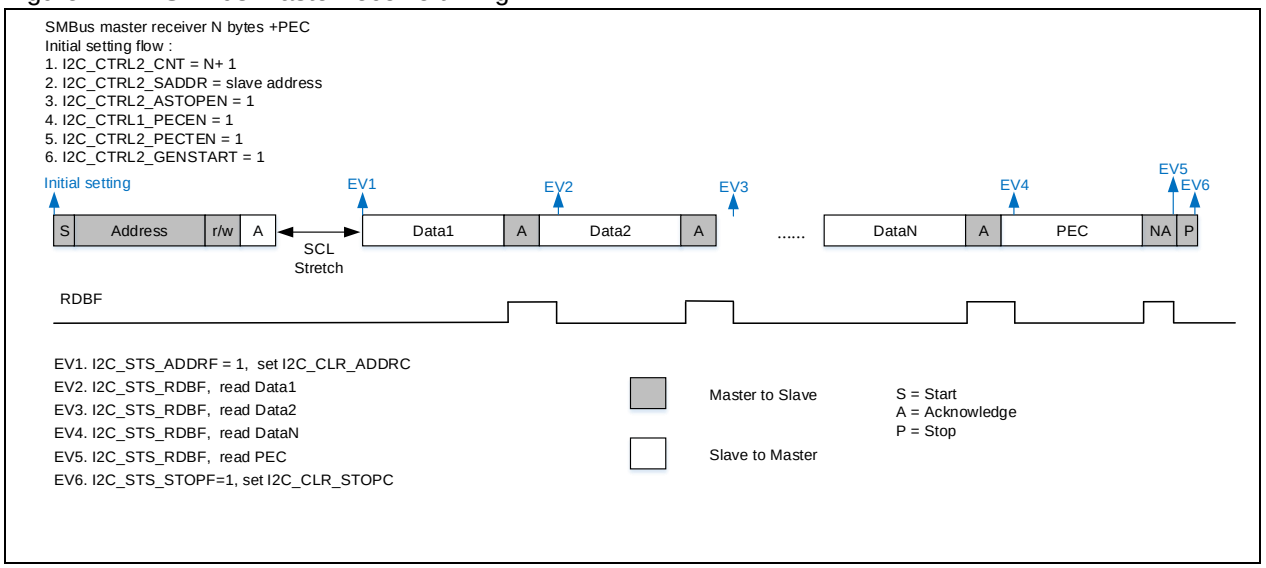


Figure 11-17 SMBus master receive timing



11.4.7 SMBus slave communication flow

The SMBus is similar to the I²C in terms of slave communication flow.

1. **I²C clock initialization (by setting the I2C_CLKCTRL register)**
   - I²C clock divider: DIV[7: 0]
   - Data hold time ($t_{HD;DAT}$): SDAD[3: 0]
   - Data setup time ($t_{SU;DAT}$): SCLD[3: 0]

   The register can be configured by means of Artery_I2C_Timing_Configuration tool.
2. **Set local address**
   - Set 7-bit address mode: by setting ADDR1MODE = 0 in the I2C_OADDR register
   - Set address 1: by setting the ADDR1 bit in the I2C_OADDR1 register
   - Enable address 1: by setting ADDR1EN=1 in the I2C_OADDR1 register
3. **SMBus-related initialization**
   - Select SMBus host: device default address acknowledged (0b1100001x) by setting DEVADDREN=1
   - Enable PEC calculation: Set PECEN=1 in the I2C_CTRL1 register
   - Set slave byte control mode:

     Slave transmit: disable byte control mode by setting SCTRL=0 in the I2C_CTRL1 register

     Slave receive: enable byte control mode by setting SCTRL=1 in the I2C_CTRL1 register
4. **Wait for address matching**

   When the local address is received, the ADDRF bit is set in the I2C_STS register. The data transfer direction can be obtained by read access to the SDIR bit in the I2C_STS register. When SDIR=0, it indicates that the slave is receiving data, whereas SDIR=1 indicates that the slave is sending data. The ADDR[6:0 bit of the I2C_STS register indicates what kind of address has been received, which is particularly helpful in the case when the dual address mode is used and the address 2 mode mask bit is set.

   Enable PEC transfer: by setting PECTEN=1 in the I2C_CTRL2 register

   Set the number of data to be transferred:
   - Slave transmit: by setting CNT=N in the I2C_CTRL2 register
   - Slave receive: by setting CNT=1 in the I2C_CTRL2 register

   Set reload mode:
   - Slave transmit: by setting RLDEN=0 in the I2C_CTRL2 register
   - Slave receive: by setting RLDEN=1 in the I2C_CTRL2 register

The ADDRF flag can be cleared by setting ADDRC=1 in the I2C_CLR register, and then data transfer starts.

5. **Data transfer (slave transmission, clock stretching enabled, STRETCH=0)**

   After address matching:

   1. I2C_TXDT data register becomes empty, the shift register becomes empty, and TDIS=1 in the I2C_STS register
   2. Data is then transferred to the shift register after writing 1 to the TXDT register
   3. The TXDT register then becomes empty, and the TDIS is set again
   4. TDIS is cleared by writing 2 to the TXDT register
   5. Repeat step 3 and 4 until data (N-1) is sent
   6. The slave will automatically transmit the Nth data, that is, PEC
   7. Wait for the generation of an NACK signal. Once received, the ACKFAILF is set in the I2C_STS register. The ACKFAILF flag is cleared by writing 1 to the ACKFAILC
   8. Wait for the generation of a STOP condition. Once received, the STOPF is set in the I2C_STS register. At the end of data transfer, the STOPF is cleared by writing 1 to the STOPC, transmission ends.

6. **Data transfer (slave receive, clock stretching enabled, STRETCH=0)**

   After address matching:

   1. I2C_RXDT register becomes empty, the shift register becomes empty, and RDBF=0 in the I2C_STS register
   2. Upon the receipt of one-byte data, RDBF=1 and TCRLD=1, then the SCL is pulled low by the slave
   3. The RDBF is cleared by read operation to the RXDT register
   4. NACKEN bit of the I2C_CTRL register can be configured to generate an ACK or NACK, if needed

      If a NACK is detected, it indicates the completion of communication

      If an ACK is detected, communication continues. Writing CNT=1 will automatically clear the TCRLD flag by hardware, and the SCL is released by the slave for the reception of the next data
   5. Repeat step 2/3/4 until the completion of data reception (N-1)
   6. Set RLDEN=0 of the I2C_CTRL2 register to disable reload mode. Set CNT=1 to repeat step 2/3 to receive a PEC. The PECERR bit will be set if a PEC error occurs
   7. Wait for the generation of a STOP condition. Once received, the STOPF is set in the I2C_STS register. The STOPF can be cleared by writing 1 to the STOPC bit in the I2C_CLR register, transfer ends.

SMBus slave transmission

Figure 11-18 SMBus slave transmission flow

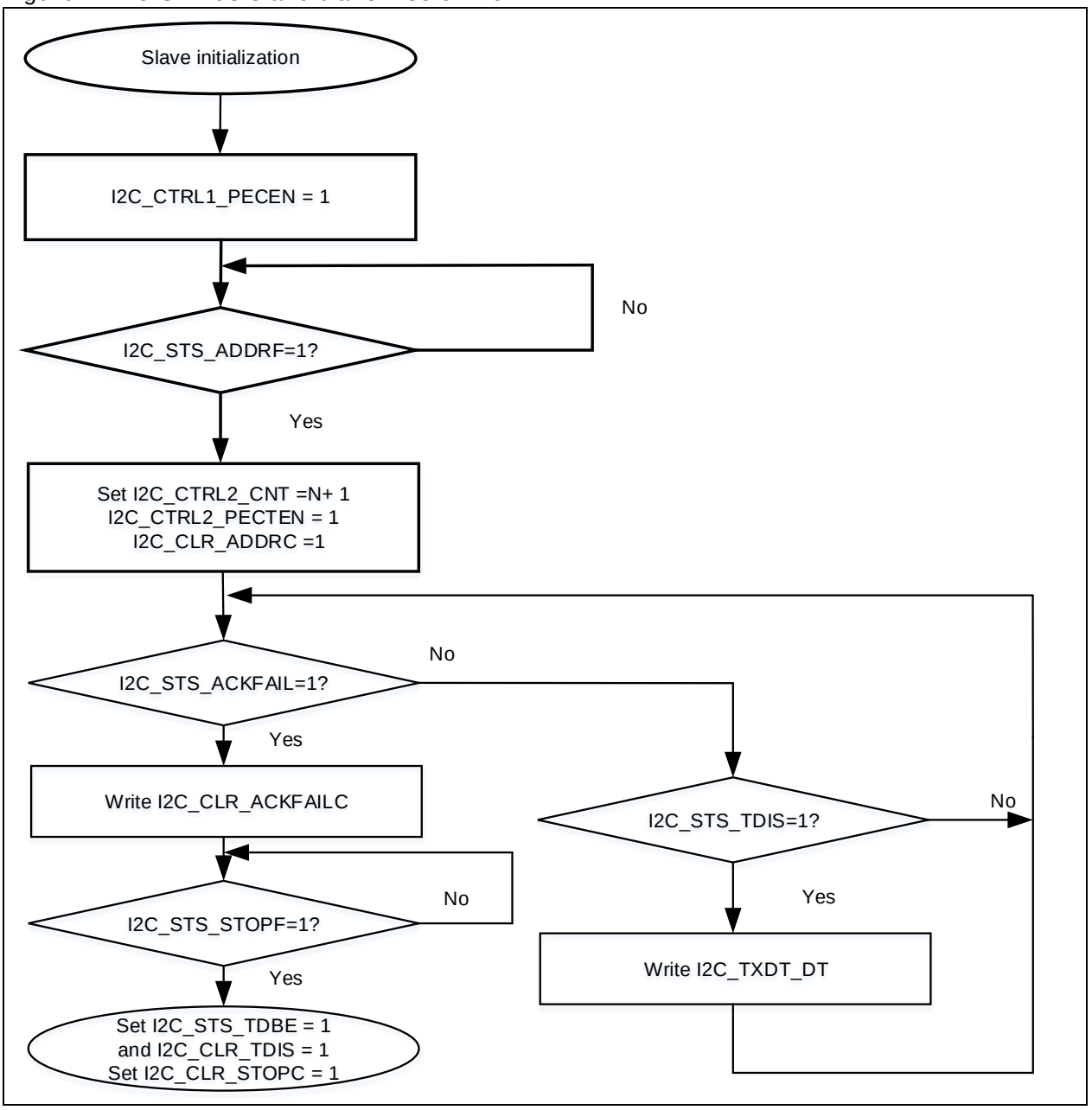
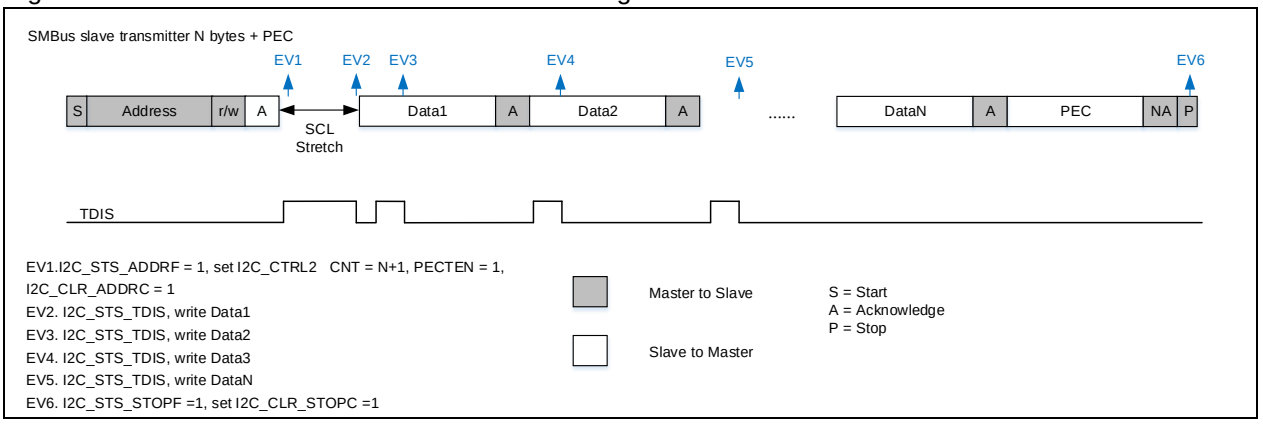


Figure 11-19 SMBus slave transmission timing



SMBus slave receive

Figure 11-20 SMBus slave receive flow

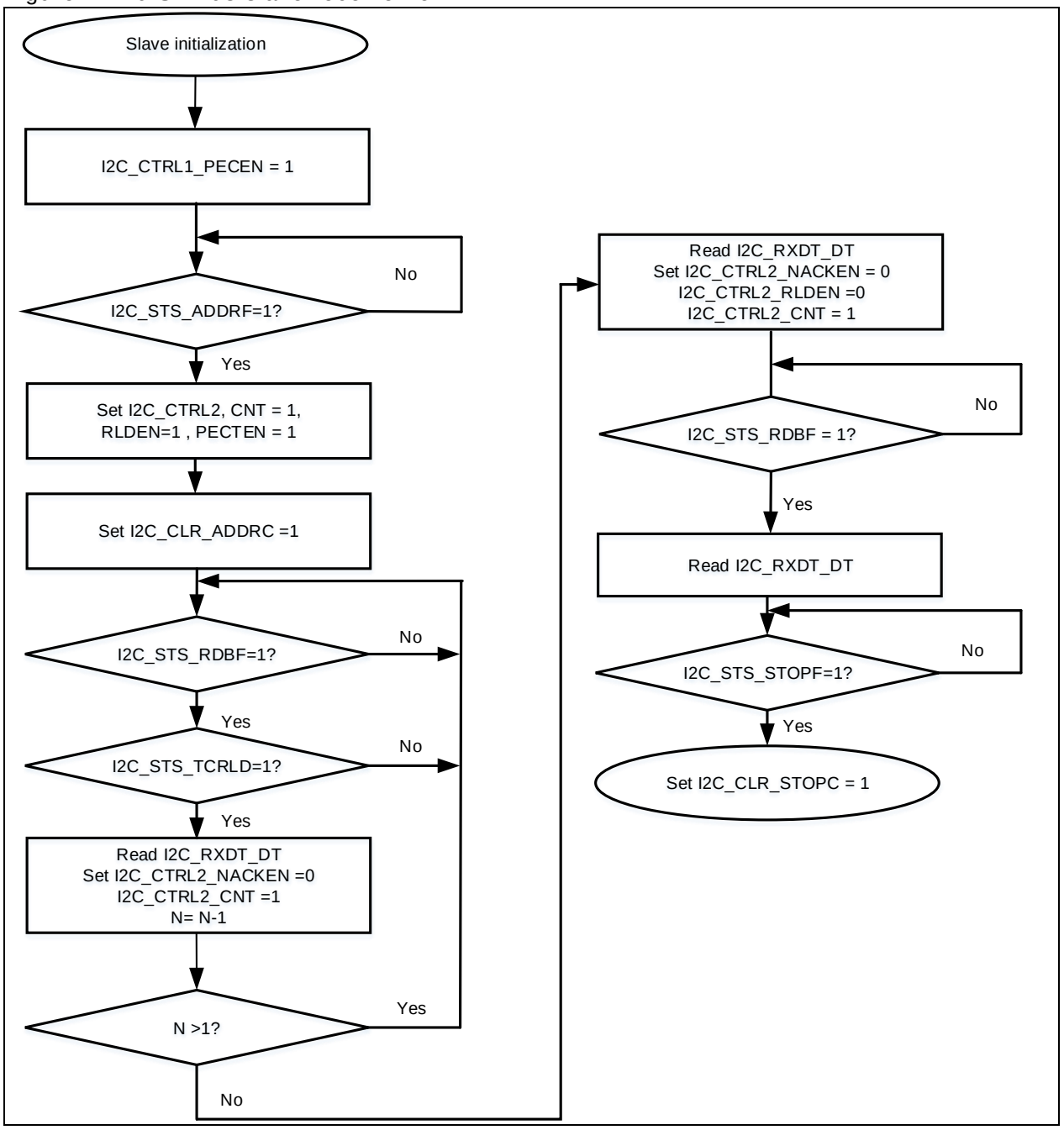
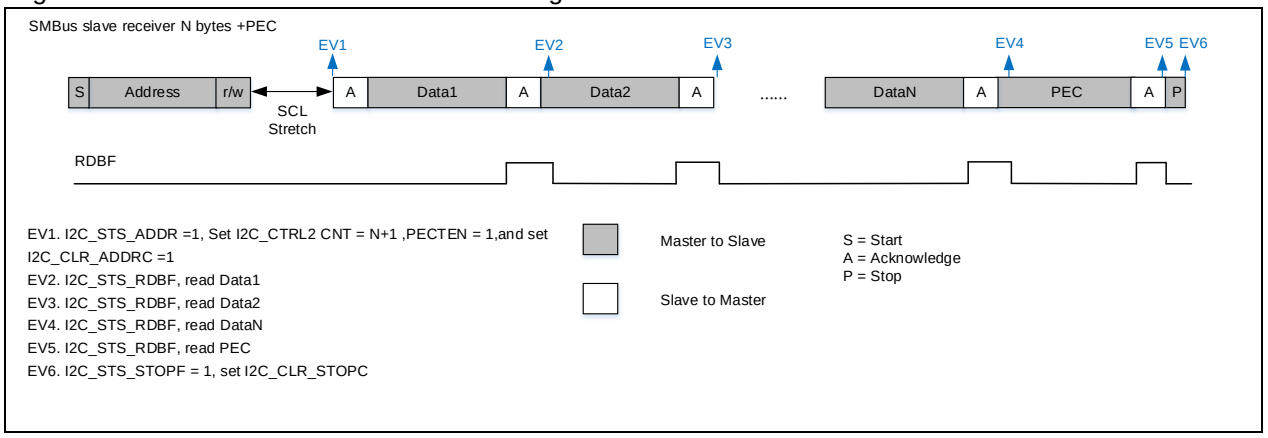


Figure 11-21 SMBus slave receive timing



11.4.8 Data transfer using DMA

I²C data transfer can be done using DMA controller so as to reduce the burden on the CPU. The TDIEN and RDIEN must be set 0 when using DMA for data transfer.

Transmission using DMA (DMATEN=1)

1. Set the peripheral address (DMA_CxPADDR= I2C_TXDT address)
2. Set the memory address (DMA_CxMADDR=data memory address)
3. The transmission direction is set from memory to peripheral (DTD=1 in the DMA_CHCTRL register)
4. Configure the total number of bytes to be transferred in the DMA_CxDTCNT register
5. Configure other parameters such as priority, memory data width, peripheral data width, interrupts, etc. in the DMA_CHCTRL register
6. Enable the DMA channel by setting CHEN=1 in the DMA_CxCTRL register
7. Enable I²C DMA request by setting DMAEN=1 in the I2C_CTRL2 register. Once the TDBE bit in the I2C_STS1 register is set, the data is loaded from the programmed memory to the I2C_DT register through DMA
8. When the number of data transfers, programmed in the DMA controller, is reached (DMA_CxDTCNT=0), the data transfer is complete (An interrupt is generated if enabled).
9. Master transmitter: Once the TDC flag is set, the STOP condition is generated, indicating that transfer is complete.

   Slave transmitter: Once the ACKFAIL flag is set, clear the ACKFAIL flag, transfer is complete.

Reception using DMA (DMAREN=1)

1. Set the peripheral address (DMA_CxPADDR = I2C_RXDT address)
2. Set the memory address (DMA_CxMADDR = memory address)
3. The transmission directions set from peripheral to memory (DTD=0 in the DMA_CHCTRL register)
4. Configure the total number of bytes to be transferred in the DMA_CxDTCNT register
5. Configure other parameters such as priority, memory data width, peripheral data width, interrupts, etc. in the DMA_CHCTRL register
6. Enable the DMA channel by setting CHEN=1 in the DMA_CxCTRL register
7. Enable I²C DMA request by setting DMAEN=1 in the I2C_CTRL2 register. Once the RDBE bit in the I2C_STS1 register is set, the data is loaded from the I2C_DT register to the programmed memory through DMA
8. When the number of data transfers, programmed in the DMA controller, is reached (DMA_CxDTCNT=0), the data transfer is complete (An interrupt is generated if enabled).
9. Master receiver: Clear the ACKFAIL flag, the STOP condition is generated, indicating that the transfer is complete (when the number of bytes to be transferred is greater >=2 and DMAEND=1, the NACK signal is generated automatically after transfer complete (DMA_CxDTCNT=0))

   Slave receiver: Once the STOPF flag is set, clear the STOPF flag, and the transfer is complete.

11.4.9 Error management

The error management feature included in the I²C provides a guarantee for the reliability of communication. Table 11-6 presents the manageable error events:

Table 11-6 I²C error events

Error event	Event Flag	Enable control bit	Clear bit
SMBus alert	ALERTF	ERRIEN	ALERTC
Timeout error	TMOUT	ERRIEN	TMOUTC
PEC error	PECERR	ERRIEN	PECERRC

Overrun/underrun	OUF	ERRIEN	OUFC
Arbitration lost	ARLOST	ERRIEN	ARLOSTC
Bus error	BUSERR	ERRIEN	BUSERRC

Overrun/Underrun (OUF)

In slave mode, an underrun/overrun may appear if the clock stretching feature is disabled (STRETCH=1 in the I2C_CTRL1 register)

In slave transmit mode: if data has not yet been written to the TXDT register before the transmission of the first bit of the to-be-transferred data (that is, before the generation of SDA edge), an underrun error may occur, and the OUF bit is set in the I2C_STS register, sending 0xFF to the bus.

In slave receive mode: The slave must read the received data in the case of the clock stretching being disabled (STRETCH=1). If one-byte data has been received and data is not read yet before the end of the next data reception, an overrun error occurs, setting the OUF bit in the I2C_STS register, and sending NCAK.

Arbitration lost (ARLOST)

An arbitration lost may occur when the device controls the SDA line to output high level but the actual bus output is low.

- Master transmit: An arbitration may occur during an address transfer and a data transfer
- Master receive: An arbitration may occur during an address transfer and an ACK response
- Slave transmit: An arbitration may occur during a data transfer
- Slave receive: An arbitration may occur during an ACK response

Once an arbitration lost is detected, the ARLOST is set by hardware in the I2C_STS register. The SCL and SDA buses will be released and go automatically back to slave mode.

Bus error (BUSERR)

The SDA line, during a data transfer, must be kept in a stable state when the SCL is in high level. The SDA can be changed only when the SCL signal becomes low, otherwise, a bus error may appear. When the SCL is high:

- SDA changes from 1 to 0: a misplaced START condition
- SDA changes from 0 to 1: a misplaced STOP condition

Both of these conditions above may trigger a bus error. Once it occurs, the BUSERR is set by hardware in the I2C_STS register.

Packet error checking (PECERR)

The PEC is available only in SMBus mode. In master receive and slave receive modes, a PEC error may appear if the received PEC is not equal to the internally calculated PEC. In this case, the PECERR bit is set by hardware in the I2C_STS register

In slave receive mode, an NACK is sent when a PEC error is detected.

In master receive mode, an NACK is always sent, whatever the PEC check result.

SMBus alert (ALERTF)

The SMBus alert feature is present when HADDREN=1 (SMBus master mode) and SMBALERT=1 (SMBus alert mode).Once an alert event is detected on the ALERT pin (ALERT pin changes from high to low), the ALERTF bit is set by hardware in the I2C_STS register.

Timeout error (TMOUT)

SMBus defines a timeout mechanism for the improvement of the system stability, preventing the bus from being pulled down in the case of a master or slave failure. Once a timeout event (defined in SMBus chapter) is detected, the TMOUT is set by hardware in the I2C_STS register. If a timeout error occurs in slave mode, the SCL and SDA buses are immediately released; if a timeout error occurs in master mode, a STOP condition is automatically by host to abort the communication

11.5 I²C interrupt requests

The following table lists all the I²C interrupt requests.

Table 11-7 I²C interrupt requests

Interrupt event	Event flag	Enable control bit
Address matched	ADDRF	ADDRIEN
Acknowledge failure	ACKFAIL	ACKFAILIEN
Stop condition received	STOPF	STOPIEN
Transmit interrupt state	TDIS	TDIEN
Receive data buffer full	RDBF	RDIEN
Transfer complete, wait for loading data	TCRLD	TDCIEN
Data transfer complete	TDC	
SMBus alert	ALERTF	ERRIEN
Timeout error	TMOUT	
PEC error	PECERR	
Overrun/Underrun	OUF	
Arbitration lost	ARLOST	
Bus error	BUSERR	

11.6 I²C debug mode

When the microcontroller enters debug mode (Cortex™-M4 halted), the SMBUS timeout either continues to work or stops, depending on the I2Cx_SMBUS_TIMEOUT configuration bit in the DEBUG module.

11.7 I²C registers

These peripheral registers must be accessed by words (32 bits).

Table 11-8 I²C register map and reset values

Register	Offset	Reset value
I2C_CTRL1	0x00	0x00000000
I2C_CTRL2	0x04	0x00000000
I2C_OADDR1	0x08	0x00000000
I2C_OADDR2	0x0C	0x00000000
I2C_CLKCTRL	0x10	0x00000000
I2C_TIMEOUT	0x14	0x00000000
I2C_STS	0x18	0x00000000
I2C_CLR	0x1C	0x00000000
I2C_PEC	0x20	0x00000000
I2C_RXDT	0x24	0x00000000
I2C_TXDT	0x28	0x00000000

11.7.1 Control register1 (I2C_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 31:24	Reserved	0x00	res	Kept at its default value.
Bit 23	PECEN	0x0	rw	PEC calculation enable 0: PEC calculation disabled 1: PEC calculation enabled
Bit 22	SMBALERT	0x0	rw	SMBus alert enable / pin set To enable SMBus master alert feature: 0: SMBus alert disabled 1: SMBus alert enabled To enable SMBus slave alert address: 0: Pin high 1: Pin low, response address 0001100x
Bit 21	DEVADDREN	0x0	rw	SMBus device default address enable 0: SMBus device default address disabled 1: SMBus device default address enabled, response device default address 1100001x
Bit 20	HADDREN	0x0	rw	SMBus host address enable 0: SMBus host address disabled 1: SMBus host address enabled, response host address 0001000x
Bit 19	GCAEN	0x0	rw	General call address enable 0: General call address disabled 1: General call address enabled, response 0000000x
Bit 18	Reserved	0x0	res	Kept at its default value.
Bit 17	STRETCH	0x0	rw	Clock stretching mode 0: Clock stretching mode enabled 1: Clock stretching mode disabled Note: This feature is valid in slave mode only.
Bit 16	SCTRL	0x0	rw	Slave receive data control 0: Slave receive data disabled 1: Slave receive data enabled
Bit 15	DMAREN	0x0	rw	DMA receive data request enable 0: DMA receive data request disabled 1: DMA receive data request enabled
Bit 14	DMATEN	0x0	rw	DMA Transmit data request enable 0: DMA Transmit data request disabled 1: DMA Transmit data request enabled
Bit 13: 12	Reserved	0x0	resd	Kept at its default value.
Bit 11: 8	DELT	0x0	rw	Digital filter value Filter time = DFLT x T_{I2C_CLK} The glitches less than the filter time on the SCL bus will be filtered.
Bit 7	ERRIEN	0x0	rw	Error interrupt enable 0: Error interrupt disabled 1: Error interrupt enabled
Bit 6	TDCIEN	0x0	rw	Data transfer complete interrupt enable 0: Data transfer complete interrupt disabled 1: Data transfer complete interrupt enabled
Bit 5	STOPIEN	0x0	rw	Stop generation complete interrupt enable 0: Stop generation complete interrupt disabled 1: Stop generation complete interrupt enabled

Bit 4	ACKFAILIEN	0x0	rw	Acknowledge fail interrupt enable 0: Acknowledge fail interrupt disabled 1: Acknowledge fail interrupt enabled
Bit 3	ADDRIEN	0x0	rw	Address match interrupt enable 0: Address match interrupt disabled 1: Address match interrupt enabled
Bit 2	RDIEN	0x0	resd	Data receive interrupt enable 0: Data receive interrupt disabled 1: Data receive interrupt enabled
Bit 1	TDIEN	0x0	rw	Data transmit interrupt enable 0: Data transmit interrupt disabled 1: Data transmit interrupt enabled
Bit 0	I2CEN	0x0	rw	I^2C peripheral enable 0: Disabled 1: Enabled

11.7.2 Control register2 (I2C_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 31: 27	Reserved	0x00	res	Kept at its default value.
Bit 26	PECTEN	0x0	rw	Request PEC transmission enable 0: Transmission disabled 1: Transmission enabled
Bit 25	ASTOPEN	0x0	rw	Automatically send stop condition enable 0: Disabled (Software sends STOP condition) 1: Enabled (Automatically send STOP condition)
Bit 24	RLDEN	0x0	rw	Send data reload mode enable 0: Send data reload mode disable 1: Send data reload mode enabled
Bit 23: 16	CNT[7: 0]	0x00	rw	Transmit data counter
Bit 15	NACKEN	0x0	rw	Not acknowledge enable 0: Acknowledge enabled 1: Acknowledge disabled
Bit 14	GENSTOP	0x0	rw	Generate stop condition 0: No stop generation 1: stop generation
Bit 13	GENSTART	0x0	rw	Generate start condition 0: No start generation 1: Start generation
Bit 12	READH10	0x0	rw	10-bit address header read enable 0: 10-bit address header read disabled 1: 10-bit address header read enabled
Bit 11	ADDR10	0x0	rw	Host sends 10-bit address mode enable 0: 7-bit address mode 1: 10-bit address mode
Bit 10	DIR	0x0	rw	Master data transfer direction 0: Receive 1: Transmit
Bit 9: 0	SADDR[9: 0]	0x000	rw	Slave address sent by the master In 7-bit address mode, BIT0 and BIT[9: 8] don't care.

11.7.3 Address register1 (I2C_OADDR1)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	rw	Kept at its default value.
Bit 15	ADDR1EN	0x0	rw	Own Address 1 enable 0: Own Address 1 disabled 1: Own Address 1 enabled
Bit 14: 11	Reserved	0x0	res	Kept at its default value.
Bit 10	ADDR1MODE	0x0	rw	Own Address mode 0: 7-bit address mode 1: 10-bit address mode
Bit 9: 0	ADDR1[9: 0]	0x000	rw	Own address1 In 7-bit address mode, bit 0 and bit [9: 8] don't care.

11.7.4 Own address register2 (I2C_OADDR2)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x000	res	Kept at its default value.
Bit 15	ADDR2EN	0x0	rw	Own address 2 enable 0: Own address 2 disabled 1: Own address 2 enabled
Bit 14: 11	Reserved	0x0	res	Kept at its default value
Bit 10: 8	ADDR2MASK[2: 0]	0x0	rw	Own address 2-bit mask 000: Match Address bit [7: 1] 001: Match Address bit [7: 2] 010: Match address bit [7: 3] 011: Match address bit [7: 4] 100: Match address bit [7: 5] 101: Match address bit [7: 6] 110: Match address bit [7] 111: Response all addresses other than those reserved for I2C
Bit 7: 1	ADDR2[7: 1]	0x00	rw	Own address 2 7-bit address mode
Bit 0	Reserved	0x0	res	Kept at its default value.

11.7.5 Timing register (I2C_CLKCTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 28	DIVL[3: 0]	0x0	rw	Low 4 bits of clock divider value
Bit 27: 24	DIVH[7: 4]	0x0	rw	High 4 bits of clock divider value DIV = (DIVH << 4) + DIVL
Bit 23: 20	SCLD[3: 0]	0x0	rw	SCL output delay T_{SCLD} = (SCLD + 1) x (DIV + 1) x T_{I2C_CLK}
Bit 19: 16	SDAD[3: 0]	0x0	rw	SDA output delay T_{SDAD} = (SDAD + 1) x (DIV + 1) x T_{I2C_CLK}
Bit 15: 8	SCLH[7: 0]	0x00	rw	SCL high level T_{SCLH} = (SCLH + 1) x (DIV + 1) x T_{I2C_CLK}
Bit 7: 0	SCLL[7: 0]	0x00	rw	SCL low level T_{SCLL} = (SCLL + 1) x (DIV + 1) x T_{I2C_CLK}

11.7.6 Timeout register (I2C_TIMEOUT)

Bit	Register	Reset value	Type	Description
Bit 31	EXTEN	0x0	rw	Cumulative clock low extend timeout enable 0: Cumulative clock low extend timeout disabled 1: Cumulative clock low extend timeout enabled Corresponds to $T_{LOW:SEXT}$ / $T_{LOW:MEXT}$ in SMBus
Bit 30: 28	Reserved	0x0	res	Kept at its default value.
Bit 27: 16	EXTTIME[11:0]	0x000	rw	Cumulative clock low extend timeout value Timeout duration = (EXTTIME + 1) x 2048 x T_{I2C_CLK}.
Bit 15	TOEN	0x0	rw	Detect clock low/high timeout enable 0: Clock low/high timeout detection disabled 1: clock low/high timeout detection enabled

				Corresponds to $T_{TIMEOUT}$ in SMBus.
Bit 14: 13	Reserved	0x0	res	Kept at its default value.
Bit 12	TOMODE	0x0	rw	Clock timeout detection mode 0: Clock low level detection 1: Clock high level detection
Bit 11: 0	TOTIME[11:0]	0x000	rw	Clock timeout detection time For clock low level detection (TOMODE = 0): Timeout duration = (TOTIME + 1) x 2048 x T_{I2C_CLK} For clock high level detection (TOMODE = 1): Timeout duration = (TOTIME + 1) x 4 x T_{I2C_CLK}

11.7.7 Status register (I2C_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	res	Kept at its default value.
Bit 23: 17	ADDR[6: 0]	0x00	r	Slave address matching value In 7-bit address mode: Slave address received In 10-bit address mode: 10-bit slave address header received
Bit 16	SDIR	0x0	r	Slave data transfer direction 0: Receive data 1: Transmit data
Bit 15	BUSYF	0x0	r	Bus busy flag transmission mode 0: Bus idle 1: Bus busy Once a START condition is detected, this bit is set; Once a STOP condition is detected, this bit is automatically cleared.
Bit 14	Reserved	0x00	res	Kept at its default value.
Bit 13	ALERTF	0x0	r	SMBus alert flag SMBus host: This bit indicates the reception of an alert signal (ALERT pin changes from high to low) 0: No alert signal received 1: Alert signal received
Bit 12	TMOUT	0x0	r	SMBus timeout flag 0: No timeout 1: Timeout
Bit 11	PECERR	0x0	r	PEC receive error flag 0: No PEC error 1: PEC error
Bit 10	OUF	0x0	r	Overrun or underrun flag In transmission mode: 0: No overrun or underrun 1: Underrun In reception mode: 0: No overrun or underrun 1: Overrun
Bit 9	ARLOST	0x0	r	Arbitration lost flag 0: No arbitration lost detected. 1: Arbitration lost detected.
Bit 8	BUSERR	0x0	rw0c	Bus error flag 0: No Bus error occurred 1: Bus error occurred
Bit 7	TCRLD	0x0	r	Data transfer complete, waiting for data load 0: Data transfer is not complete yet 1: Data transfer is complete This bit is set when data transfer is complete (CNT=1) and reload mode is enabled (RLDEN=1). It is automatically cleared when writing a CNT value. This bit is applicable in master mode or when SCTRL=1 in slave mode.
Bit 6	TDC	0x0	r	Data transfer complete flag 0: Data transfer is not completed yet (the shift register still holds data)

				1: Data transfer is completed (shift register become empty and all data has been sent to the bus) This bit is set when ASTOPEN = 0, RLDEN = 0, CNT = 0. It is automatically cleared after a START or a STOP condition is received.
Bit 5	STOPF	0x0	r	Stop condition generation complete flag 0: No Stop condition detected. 1: Stop condition detected.
Bit 4	ACKFAILF	0x0	r	Acknowledge failure flag 0: No acknowledge failure 1: Acknowledge failure
Bit 3	ADDRHF	0x0	r	0~7 bit address head match flag 0: 0~7 bit address head mismatch 1: 0~7 bit address head match
Bit 2	RDBF	0x0	r	Receive data buffer full flag 0: Data register has not received data yet 1: Data register has received data
Bit 1	TDIS	0x0	rw1s	Transmit data interrupt status 0: Data has been written to the I2C_TXDT 1: Data has been sent from the I2C_TXDT to the shift register. I2C_TXDT become empty, and thus the to-be-transferred data must be written to the I2C_TXDT. When the clock stretching mode is disabled, a TDIS event is generated by writing 1 so that data is written to the I2C_TXDT register in advance.
Bit 0	TDBE	0x0	rw1s	Transmit data buffer empty flag 0: I2C_TXDT not empty 1: I2C_TXDT empty This bit is only used to indicate the current status of the I2C_TXDT register. The I2C_TXDT register can be cleared by writing 1 through software.

11.7.8 Status clear register (I2C_CLR)

Bit	Register	Reset value	Type	Description
Bit 31: 14	Reserved	0x00000	res	Kept at its default value.
Bit 13	ALERTC	0x0	w	Clear SMBus alert flag SMBus alert flag is cleared by writing 1.
Bit 12	TMOUTC	0x0	w	Clear SMBus timeout flag SMBus timeout flag is cleared by writing 1.
Bit 11	PECERRC	0x0	w	Clear PEC receive error flag PEC receive error flag is cleared by writing 1.
Bit 10	OUFC	0x0	w	Clear overload / underload flag Overload / underload flag is cleared by writing 1.
Bit 9	ARLOSTC	0x0	w	Clear arbitration lost flag Arbitration lost flag is cleared by writing 1.
Bit 8	BUSERRC	0x0	w	Clear bus error flag Bus error flag is cleared by writing 1
Bit 7: 6	Reserved	0x0	res	Kept at its default value.
Bit 5	STOPC	0x0	w	Clear stop condition generation complete flag Stop condition generation complete flag is cleared by writing 1.
Bit 4	ACKFAILC	0x0	w	Clear acknowledge failure flag Acknowledge failure flag is cleared by writing 1.
Bit 3	ADDRC	0x0	w	Clear 0~7 bit address match flag 0~7 bit address match flag is cleared by writing 1.
Bit 2: 0	Reserved	0x0	res	Kept at its default value.

11.7.9 PEC register (I2C_PEC)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	res	Kept at its default value.
Bit 7: 0	PECVAL[7: 0]	0x00	r	PEC value

11.7.10 Receive data register (I2C_RXDT)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	res	Kept at its default value.
Bit 7: 0	DT[7: 0]	0x00	r	Receive data register

11.7.11 Transmit data register (I2C_TXDT)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	res	Kept at its default value.
Bit 7: 0	DT[7: 0]	0x00	rw	Transmit data register

12 Universal synchronous/asynchronous receiver/transmitter (USART)

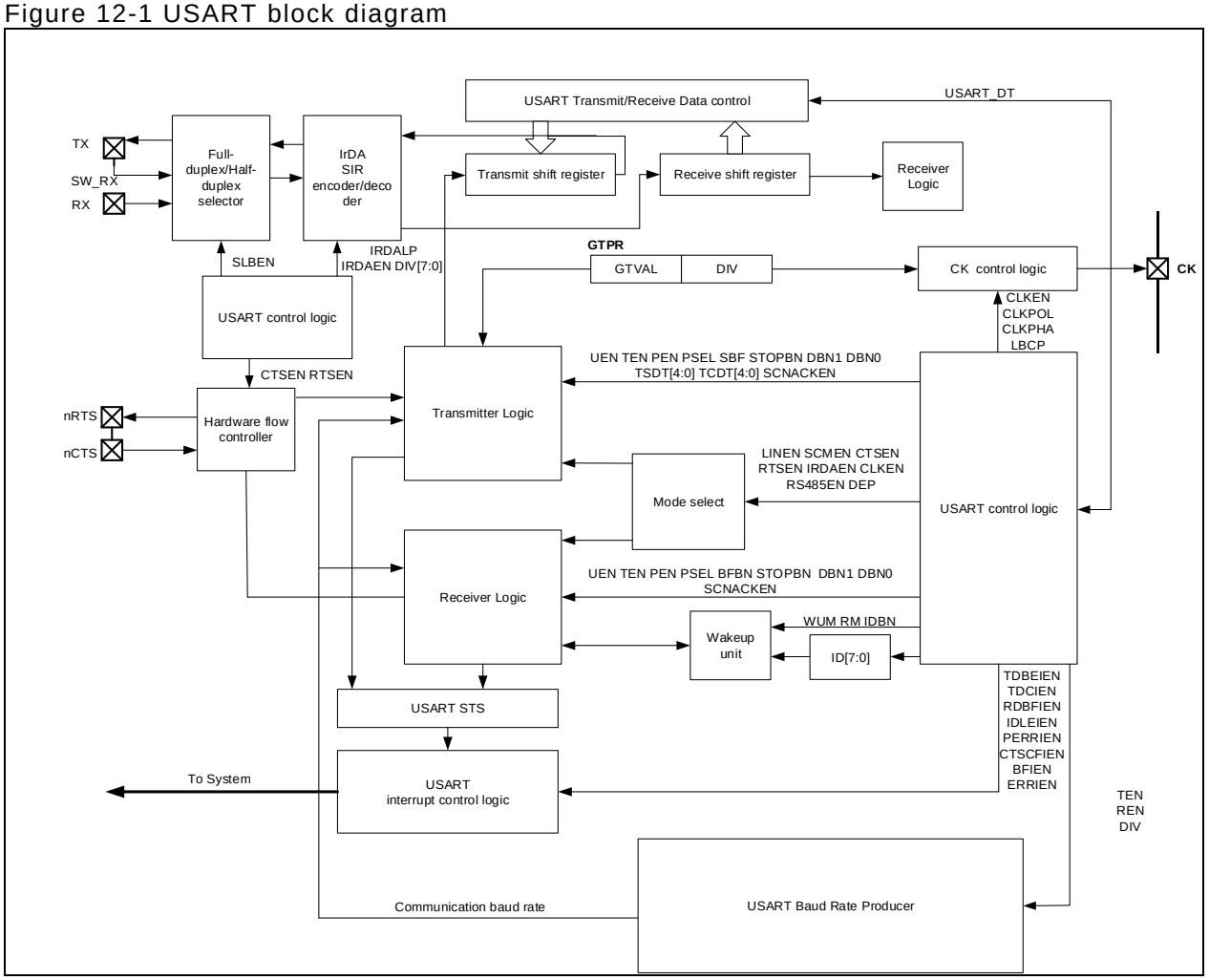
12.1 USART introduction

The universal synchronous/asynchronous receiver/transmitter (USART) serves an interface for communication by means of various configurations and peripherals with different data formats. It supports asynchronous full-duplex and half-duplex as well as synchronous transfer. With a programmable baud rate generator, USART offers up to 9 MBits/s of baud rate by setting the system frequency and frequency divider, which is also convenient for users to configure the required communication frequency.

In addition to standard NRZ asynchronous and synchronous receiver/transmitter communication protocols, USART also supports widely-used serial communication protocols such as LIN (Local Interconnection Network), IrDA (Infrared Data Association) SIRENDEC specification, Asynchronous SmartCard protocol defined in ISO7816-3 standard, and CTS/RTS (Clear To Send/Request To Send) hardware flow operation.

It also allows mutli-processor communication, and supports silent mode waken up by idle frames or ID matching to build up a USART network. Meanwhile, high-speed communication is possible by using DMA.

Figure 12-1 USART block diagram



USART main features:

- Programmable full-duplex or half-duplex communication
  - Full-duplex, asynchronous communication
  - Half-duplex, single communication
- Programmable communication modes
  - NRZ standard format (Mark/Space)
  - LIN (Local Interconnection Network)
  - IrDA SIR
  - Asynchronous SmartCard protocol defined in ISO7816-3 standard: Support 0.5 or 1.5 stop bits in Smartcard mode
  - RS-232 CTS/RTS (Clear To Send/Request To Send) hardware flow operation
  - RS-485
  - Multi-processor communication with silent mode (waken up by configuring ID match and bus idle frame)
  - Synchronous mode
- Programmable baud rate generator
  - Shared by transmission and reception, up to 9 MBits/s
- Programmable frame format
  - Programmable data word length (7 bits, 8 bits or 9 bits)
  - Programmable stop bits-support 1 or 2 stop bits
  - Programmable parity control: transmitter with parity bit transmission capability, and receiver with received data parity check capability
- Programmable DMA multi-processor communication
- Programmable separate enable bits for transmitter and receiver
- Programmable output CLK phase, polarity and frequency
- Detection flags
  - Receive buffer full
  - Transmit buffer empty
  - Transfer complete flag
- Four error detection flags
  - Overrun error
  - Noise error
  - Framing error
  - Parity error
- Programmable 10 interrupt sources with flags
  - CTSF changes
  - LIN break detection
  - Transmit data register empty
  - Transmission complete
  - Receive data register full
  - Idle bus detected
  - Overrun error
  - Framing error
  - Noise error

— Parity error

12.2 Full-duplex/half-duplex selector

The full-duplex and half-duplex selector enables USART to perform data exchanges with peripherals in full-duplex or half-duplex mode, which is achieved by setting the corresponding registers.

In two-wire unidirectional full-duplex mode (by default), TX pin is used for data output, while the RX pin is used for data input. Since the transmitter and receiver are independent of each other, USART is allowed to send/receive data at the same time so as to achieve full-duplex communication.

When the HALFSEL is set 1, the single-wire bidirectional half-duplex mode is selected for communication. In this case, the LINEN, CLKEN, SCMEN and IRDAEN bits must be set 0. RX pin is inactive, while TX and SW_RX are interconnected inside the USART. For the USART part, TX pins is used for data output, and SW_RX for data input. For the peripheral part, bidirectional data transfer is executed through IO mapped by TX pin.

12.3 Mode selector

12.3.1 Introduction

USART mode selector allows USART to work in different operation modes through software configuration so as to enable data exchanges between USART and peripherals with different communication protocols.

USART supports NRZ standard format (Mark/Space), by default. It also supports LIN (Local Interconnection Network), IrDA SIR (Serial Infrared), Asynchronous Smartcard protocol in ISO7816-3 standard, RS-232 CTS/RTS (Clear To Send/Request To Send) hardware flow operation, silent mode and synchronous mode, depending on USART mode selection configuration.

12.3.2 Configuration procedure

Selection of operation mode is done by following the configuration process listed below. In addition, such configuration method, along with those of receiver and transmitter described in the subsequent sections, are used to make USART initialization configuration.

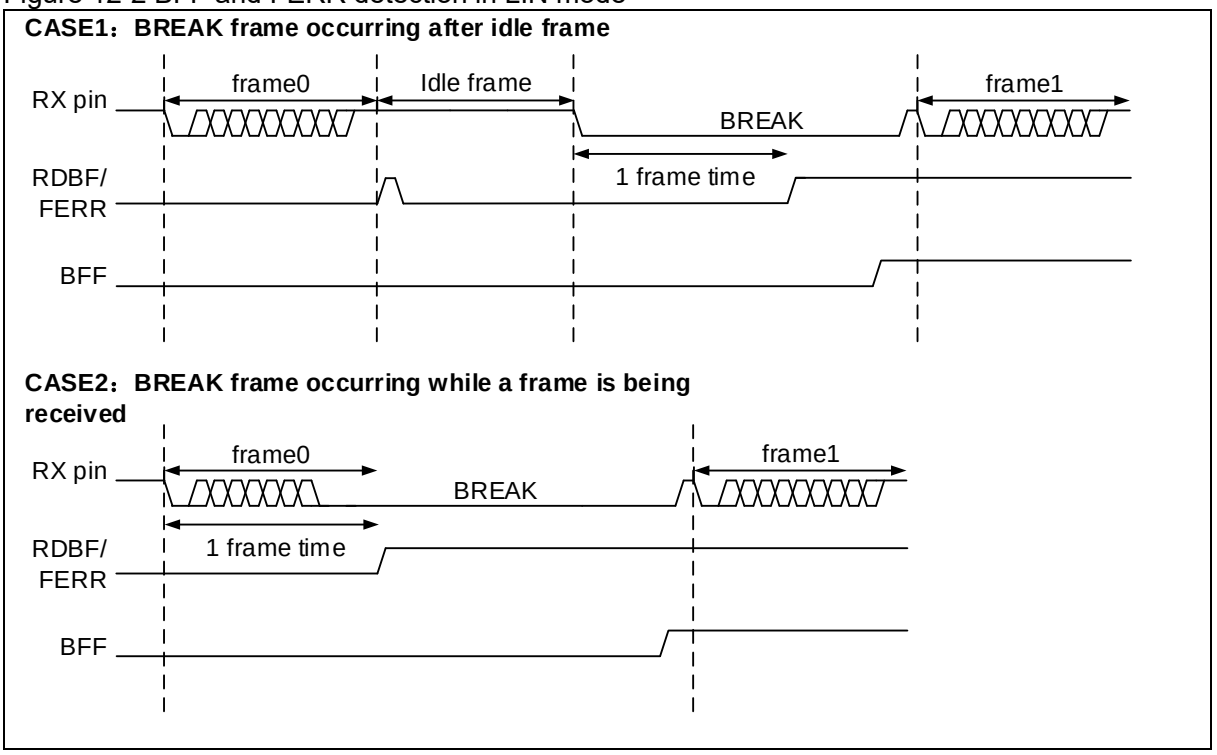
1. **LIN mode:**

   Parameters configuration: LINEN=1, CLKEN=0, STOPBN[1: 0]=0, SCMEN=0, SLHDEN=0, IRDAEN=0 and DBN[1: 0]=0.

   LIN master has break frame transmission capability, and thus it is able to send 13-bit low-level LIN synchronous break frame by setting SBF=1.

   Similarly, LIN slave has break frame detection capability, and thus it is able to detect 11-bit or 12-bit break frame, depending on whether BFBN=1 or BFBN=0.

Figure 12-2 BFF and FERR detection in LIN mode



2. **Smartcard mode:**

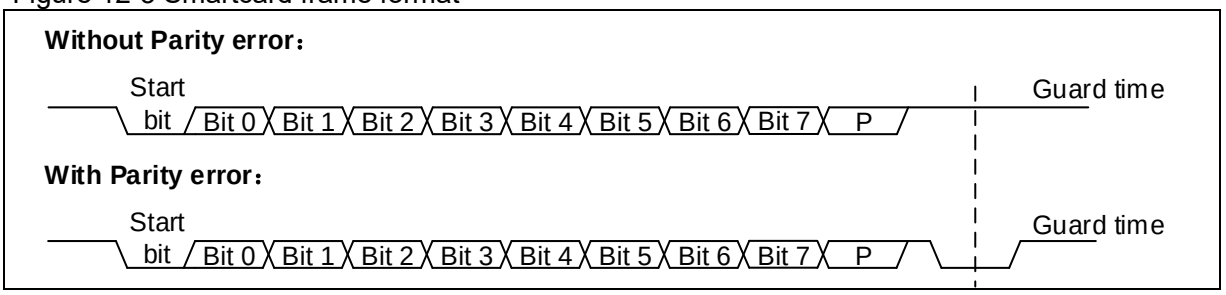
Parameters configuration: SCMEN=1, LINEN=0, SLHDEN=0, IRDAEN=0, CLKEN=1, DBN[1: 0]=1, PEN=1, and STOPBN[1: 0]=11.

The polarity, phase and pulse number of the clock can be configured by setting the CLKPOL, CLKPHA and LBCP bits (Refer to Synchronous mode for details).

The assertion of the TDC flag can be delayed by setting the SCGT[7: 0] bit (guard time bit). The TDF bit can be asserted high after the guard time counter reaches the value programmed in the SCGT[7: 0] bit.

The Smartcard is a single-wire half duplex communication protocol. The SCNACKEN bit is used to select whether to send NACK when a parity error occurs. This is to indicate to the Smarcard that the data has not been correctly received.

Figure 12-3 Smartcard frame format

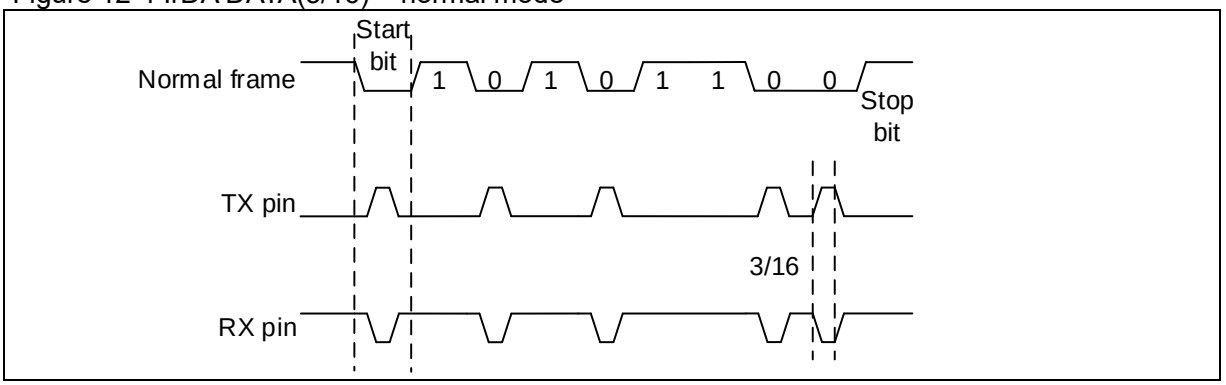


3. **Infrared mode:**

Parameters configuration: IRDAEN=1, CLKEN=0, STOPBN[1: 0]=0, SCMEN=0 and SLHDEN=0.

The infrared low-power mode can be enabled by setting IRDALP=1. In normal mode the transmitted pulse width is specified as 3/16 bit. In infrared low-power mode, the pulse width can be configurable. And the ISDIV[7:0] bit can be used to achieve the desired low-power frequency.

Figure 12-4 IrDA DATA(3/16) – normal mode



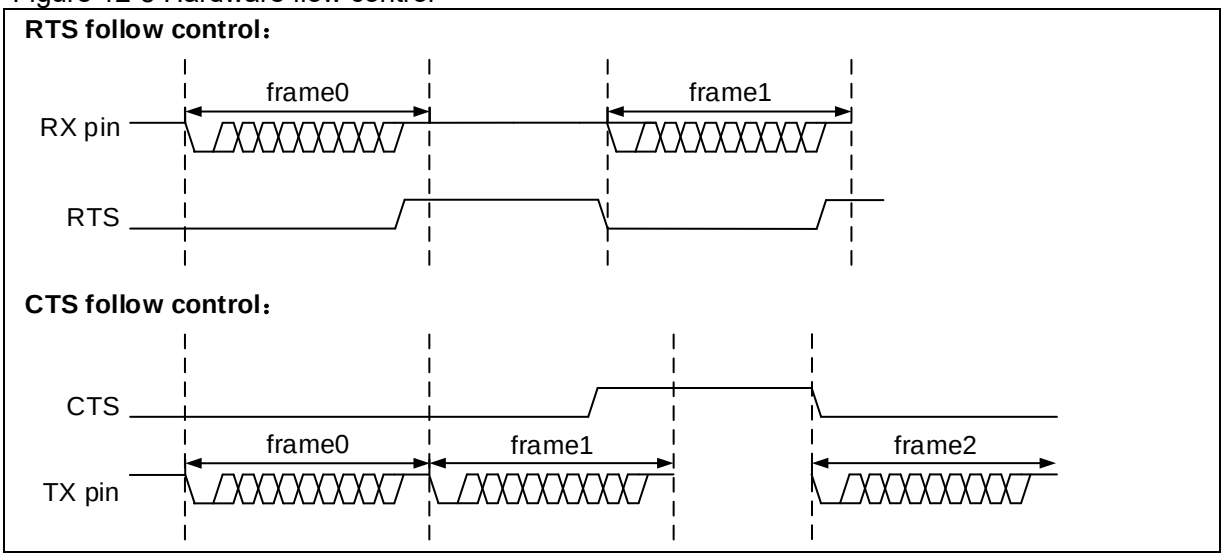
4. **Hardware flow control mode:**

RTS and CTS flow control can be enabled by setting RTSEN=1 and CTSEN=1, respectively.

RTS: the RTS becomes active (pull-down means low) as soon as the USART receiver is ready to receive a data. When the data has arrived (starts at each STOP bit) in the receive register, the RTS bit is set, indicating request to stop data transfer at the end of current frame.

CTS: the USART transmitter checks the CTS input before sending next frame. The next data is sent if CTS is active (when low); if CTS becomes inactive (when high) during transmission, it stops sending at the end of current transfer.

Figure 12-5 Hardware flow control



5. **RS485 mode:**

This mode is enabled by setting RS485EN=1. The enable signal is output on the RTS pin. The DEP bit is used to select the polarity of the DE signal. The TSDT[4: 0] bit is used to define the latency before the transmission of the start bit on the transmitter side, while the TCDT[4: 0] is used to define the latency before the TC flag is set following the stop bit at the end of the last data.

6. **Silent mode:**

Silent mode can be entered by setting RM=1. It is possible to wake up from silent mode by setting WUM=1 (ID match) and WUM=0 (idle bus), respectively. The ID[7: 0] is configurable. Select ID[7: 0] or ID[3: 0] by setting the IDBN bit. When ID match is selected, if the MSB of data bit is set, it indicates that the current data stands for ID.

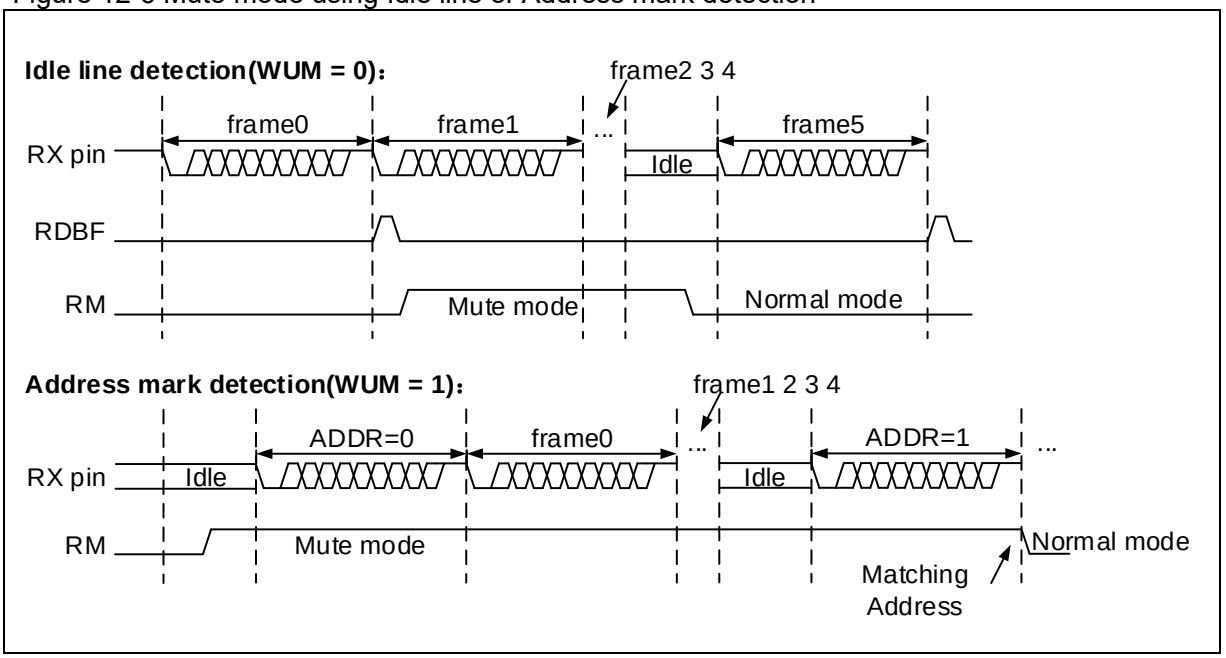
When parity check is disabled, if DBN[1:0]=10, the MSB refers to the USART_DT[6]; if DBN[1:0]=00, the MSB refers to the USART_DT[7]; if DBN[1:0]=01, the MSB stands for USART_DT[8].

When parity check is enabled, if DBN[1:0]=10, the MSB stands for USART_DT[5]; if DBN[1:0]=00, the MSB stands for USART_DT[6]; if DBN[1:0]=01, the MSB stands for USART_DT[7].

When the ID[3: 0] bit is selected, the four LSB bits indicate the ID value; When the ID[7:0] bit is selected, all of the LSB bits indicates the ID value, except for the above parity check bits and MSB

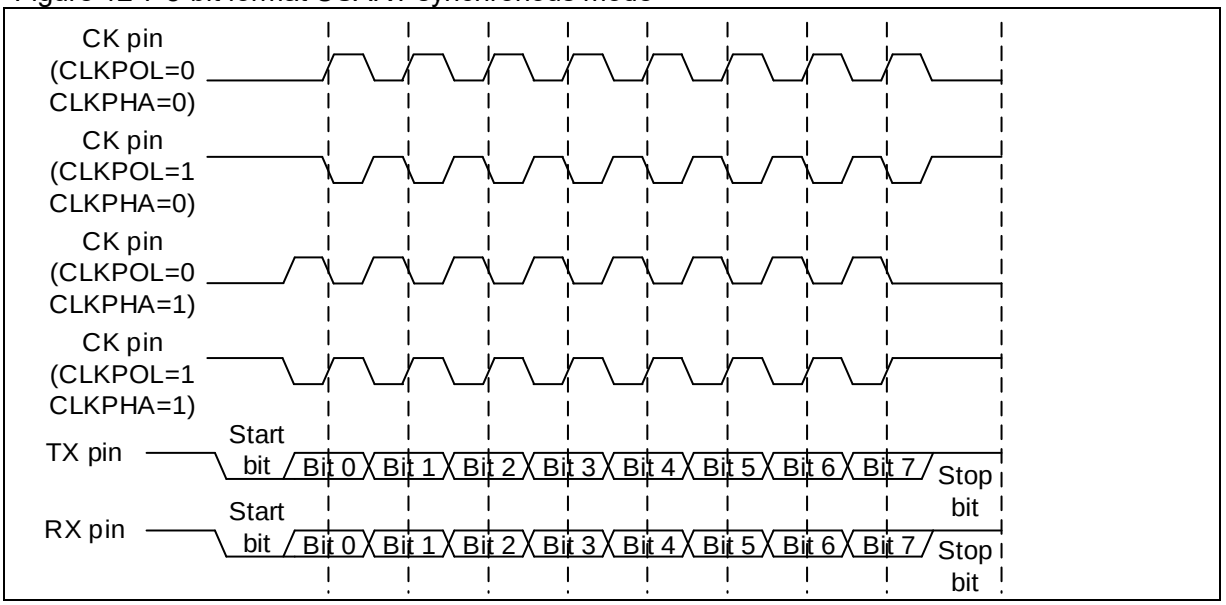
bits.

Figure 12-6 Mute mode using Idle line or Address mark detection



7. **Synchronous mode:**
By setting the CLKEN bit to 1, synchronous mode and clock pin output are enabled. Select CK pin high or low in idle state by setting the CLKPOL bit (1 or 0). Whether to sample data on the second or first edge of the clock depends on the CLKPHA bit (1 or 0). The LBCP bit (1 or 0) is used to select whether to output clock on the last data bit. And the ISDIV[4: 0] is used to select the required clock output frequency.

Figure 12-7 8-bit format USART synchronous mode



12.4 USART frame format and configuration

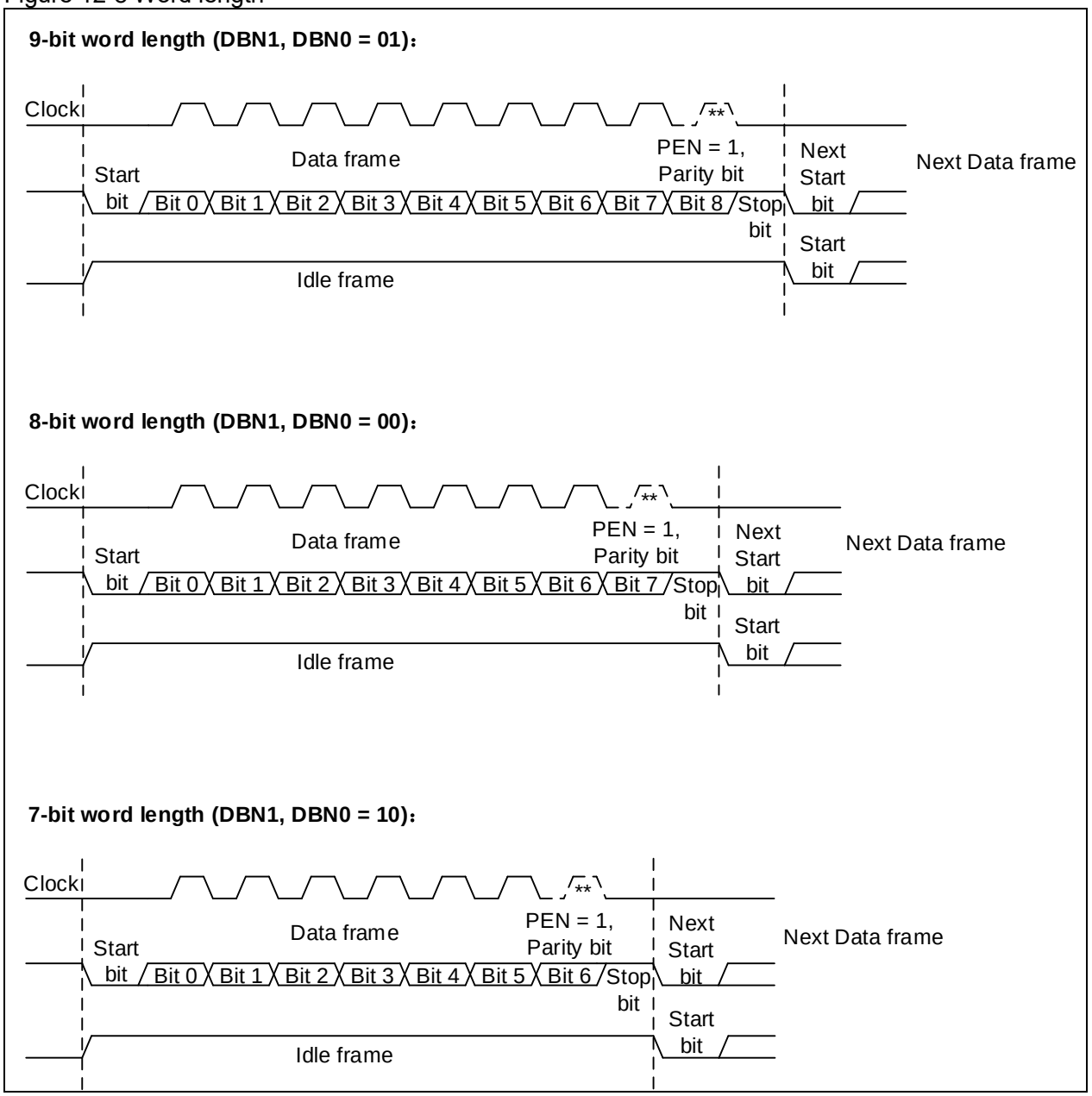
USART data frame consists of start bit, data bit and stop bit, with the last data bit being as a parity bit.

USART idle frame size is equal to that of the data frame under current configuration, but all bits are 1.

USART break frame size is the current data frame size plus its stop bit. All bits before the stop bit are 0. In non-LIN mode, a break frame transmission and detection must be in line with this rule. For instance, if DBN[1:0]=00, the break frame size for transmission and detection should be 10-bit low level plus its stop bit. In LIN mode, refer to Mode selector and configuration process for more details.

The DBN1 and DBN0 bits are used to program 7-bit (DBN[1:0]=10), 8-bit (DBN[1:0]=00) or 9-bit (DBN[1:0]=01) data bits.

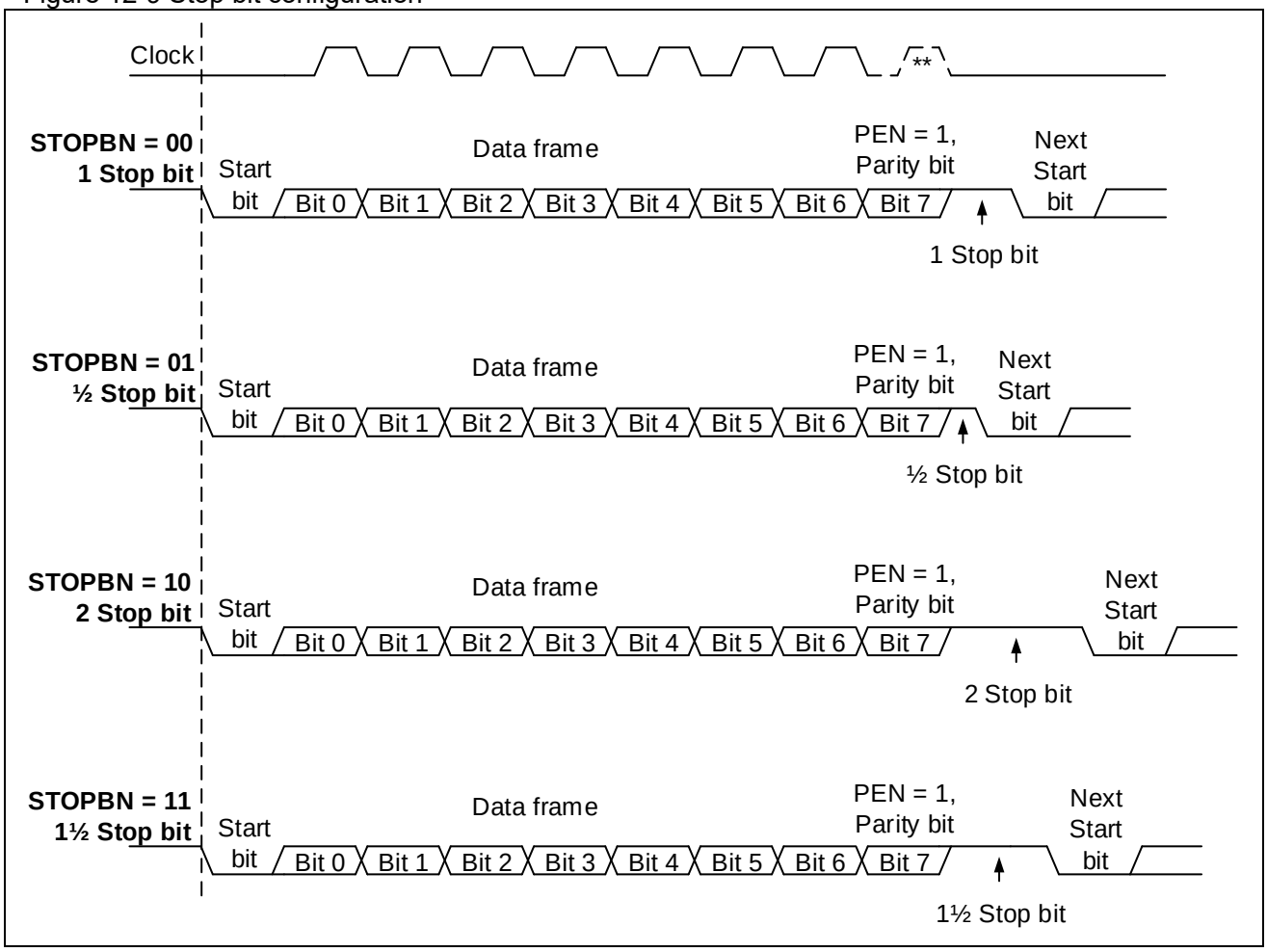
Figure 12-8 Word length



The STOPBN bit is used to program one bit (STOPBN=00), 0.5-bit (STOPBN=01), two-bit (STOPBN=10) and 1.5-bit (STOPBN=11) stop bits.

Set the PEN=1 will enable parity control. PSEL=1 indicates Odd parity, while PSEL=0 for Even parity. Once the parity control is enabled, the MSB of the data bit will be replaced with parity bit, that is, the significant bits is reduced by one bit.

Figure 12-9 Stop bit configuration



12.5 DMA transfer introduction

Enable transmit data buffer and receive data buffer using DMA to achieve continuous high-speed transmission for USART, which is detailed in subsequent sections. For more information on specific DMA configuration, refer to DMA chapter.

12.5.1 Transmission using DMA

1. Select a DMA channel: Select a DMA channel from DMA channel map table described in DMA chapter.
2. Configure the destination of DMA transfer: Configure the USART_DT register address as the destination address bit of DMA transfer in the DMA control register. Data will be sent to this address after transmit request is received by DMA.
3. Configure the source of DMA transfer: Configure the memory address as the source of DMA transfer in the DMA control register. Data will be loaded into the USART_DT register from the memory address after transmit request is received by DMA.
4. Configure the total number of bytes to be transferred in the DMA control register.
5. Configure the channel priority of DMA transfer in the DMA control register.
6. Configure DMA interrupt generation after half or full transfer in the DMA control register.
7. Enable DMA transfer channel in the DMA control register.

12.5.2 Reception using DMA

1. Select a DMA transfer channel: Select a DMA channel from DMA channel map table described in DMA chapter.
2. Configure the destination of DMA transfer: Configure the memory address as the destination of DMA

transfer in the DMA control register. Data will be loaded from the USART_DT register to the programmed destination after reception request is received by DMA.

3. Configure the source of DMA transfer: Configure the USART_DT register address as the source of DMA transfer in the DMA control register. Data will be loaded from the USART_DT register to the programmed destination after reception request is received by DMA.
4. Configure the total number of bytes to be transferred in the DMA control register.
5. Configure the channel priority of DMA transfer in the DMA control register.
6. Configure DMA interrupt generation after half or full transfer in the DMA control register.
7. Enable a DMA transfer channel in the DMA control register.

12.6 Baud rate generation

12.6.1 Introduction

USART baud rate generator uses an internal counter based on PCLK. The DIV (USART_BAUDR [15:0] register) represents the overflow value of the counter. Each time the counter is full, it denotes one-bit data. Thus each data bit width refers to PCLK cycles x DIV.

The receiver and transmitter of USART share the same baud rate generator, and the receiver splits each data bit into 16 equal parts to achieve oversampling, so the data bit width should not be less than 16 PCLK periods, that is, the DIV value must be equal to or greater than 16.

12.6.2 Configuration

User can program the desired baud rate by setting different system clocks and writing different values into the USART_BAUDR register. The calculation format is as follows:

$$\frac{\mathrm{TX}}{\mathrm{RX}} baud\ rate = \frac{f_{CK}}{\mathrm{DIV}}$$

Where, f_{CK} refers to the system clock of USART (i.e. PCLK1/PCLK2)

Note:

1. *Write access to the USART_BAUDR register before UEN. The baud rate register value should not be altered when UEN=1.*
2. *When USART receiver or transmitter is disabled, the internal counter will be reset, and baud rate interrupt will occur.*

Table 12-1 Error calculation for programmed baud rate

Baud		fPCLK=36MHz			fPCLK=72MHz		
No.	Kbps	Actual	Value programmed in the baud register	Error%	Actual	Value programmed in the baud register	Error%
1	2.4	2.4	15000	0%	2.4	30000	0%
2	9.6	9.6	3750	0%	9.6	7500	0%
3	19.2	19.2	1875	0%	19.2	3750	0%
4	57.6	57.6	625	0%	57.6	1250	0%
5	115.2	115.384	312	0.15%	115.2	625	0%
6	230.4	230.769	156	0.16%	230.769	312	0.16%
7	460.8	461.538	78	0.16%	461.538	156	0.16%
8	921.6	923.076	39	0.16%	923.076	78	0.16%
9	2250	2250	16	0%	2250	32	0%
10	4500	NA	NA	NA	4500	16	0%

Taking a baud rate of 115.2Kbps as an example, if fPCLK=36MHz, the value in the baud register should be set to 312(0x38). Based on formula, the calculated baud rate (actual) is 36000000 / 312 = 115384 =

115.384Kbps.

The % error between the desired and actual value is calculated based on the formula: (Calculated actual result-Desired)/desired baud rate*100%, that is, (115.384 - 115.2) / 115.2 * 100% = 0.15%.

12.7 Transmitter

12.7.1 Transmitter introduction

USART transmitter has its individual TEN control bit. The transmitter and receiver share the same baud rate that is programmable. There is a transmit data buffer (TDR) and a transmit shift register in the USART. The TDBE bit is set whenever the TDR is empty, and an interrupt is generated if the TDBEIEN is set.

The data written by software is stored in the TDR register. When the shift register is empty, the data will be moved from the TDR register to the shift register so that the data in the transmit shift register is output on the TX pin in LSB mode. The output format depends on the programmed frame format.

If synchronous transfer or clock output is selected, the clock pulse is output on the CK pin. If the hardware flow control is selected, the control signal is input on the CTS pin.

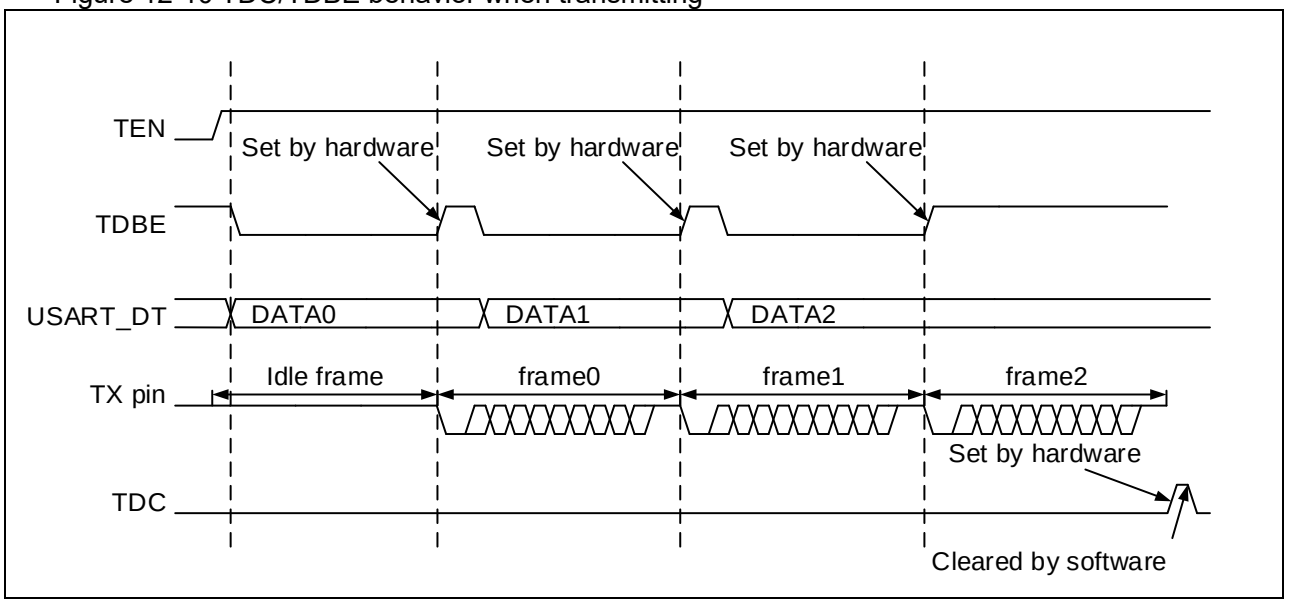
Note: 1. The TEN bit cannot be reset during data transfer, or the data on the TX pin will be corrupted.

2. After the TEN bit is enabled, the USART will automatically send an idle frame.

12.7.2 Transmitter configuration

1. USART enable: Set the UEN bit.
2. Full-duplex/half-duplex configuration: Refer to *12.2 Full-duplex/half-duplex selector.*
3. Mode configuration: Refer to *12.3 Mode selector*.
4. Frame format configuration: Refer to *12.4 USART frame format and configuration*.
5. Interrupt configuration: Refer to *12.10 Interrupt requests.*
6. DMA transmission configuration: If the DMA mode is selected, the DMATEN bit (bit 7 in the USART_CTRL3register) is set, and configure DMA register accordingly.
7. Baud rate configuration: Refer to *12.6 Baud rate generation*.
8. Transmitter enable: When the TEN bit is set, the USART transmitter will send an idle frame.
9. Write operation: Wait until the TDBE bit is set, the data to be transferred will be loaded into the USART_DT register (This operation will clear the TDBE bit). Repeat this step in non-DMA mode.
10. After the last data expected to be transferred is written, wait until the TDC is set, indicating the end of transfer. The USART cannot be disabled before the flag is set, or transfer error will occur.
11. When TDC=1, read access to the USART_STS register and write access to the USART_DT register will clear the TDC bit; This bit can also be cleared by writing “0”, but this is valid only in DMA mode.

Figure 12-10 TDC/TDBE behavior when transmitting



12.8 Receiver

12.8.1 Receiver introduction

USART receiver has its individual REN control bit (bit 2 in the USART_CTRL1 register). The transmitter and receiver share the same baud rate that is programmable. There is a receive data buffer (RDR) and a receive shift register in the USART.

The data is input on the RX pin of the USART. When a valid start bit is detected, the receiver ports the data received into the receive shift register in LSB mode. After a full data frame is received, based on the programmed frame format, it will be moved from the receive shift register to the receive data buffer, and the RDBF is set accordingly. An interrupt is generated if the RDBFIEN is set.

If hardware flow control is selected, the control signal is output on the RTS pin.

During data reception, the USART receiver will detect whether there are errors to occur, including framing error, overrun error, parity check error or noise error, depending on software configuration, and whether there are interrupts to generate using the interrupt enable bits.

12.8.2 Receiver configuration

Configuration procedure:

1. USART enable: UEN bit is set.
2. Full-duplex/half-duplex configuration: Refer to *12.2 Full-duplex/half-duplex selector*.
3. Mode configuration: Refer to *12.3 Mode selector*.
4. Frame format configuration: Refer to *12.4 USART frame format and configuration*.
5. Interrupt configuration: Refer to *12.10 Interrupt requests*.
6. Reception using DMA: If the DMA mode is selected, the DMAREN bit is set, and configure DMA register accordingly.
7. Baud rate configuration: Refer to *12.6 Baud rate generation*.
8. Receiver enable: REN bit is set.

Character reception:

- The RDBF bit is set. It indicates that the content of the shift register is transferred to the RDR (Receiver Data Register). In other words, data is received and can be read (including its associated error flags)
- An interrupt is generated when the RDBFIEN is set.

- The error flag is set when a framing error, noise error or overrun error is detected during reception.
- In DMA mode, the RDNE bit is set after every byte is received, and it is cleared when the data register is read by DMA.
- In non-DMA mode, the RDBF bit is cleared when read access to the USART_DT register by software. The RDBF flag can also be cleared by writing 0 to it. The RDBF bit must be cleared before the end of next frame reception to avoid overrun error.

Break frame reception:

- Non-LIN mode: It is handled as a framing error, and the FERR is set. An interrupt is generated if the corresponding interrupt bit is enabled. Refer to framing error described below for details.
- LIN mode: It is handled as a break frame, and the BFF bit is set. An interrupt is generated if the BFIEN is set.

Idle frame reception:

- It is handled as a data frame, and the IDLEF bit is set. An interrupt is generated if the IDLEIEN is set.

When a framing error occurs:

- The FERR bit is set.
- The USART receiver moves the invalid data from the receive shift register to the receive data buffer.
- In non-DMA mode, both FERR and RDBF are set at the same time. The latter will generate an interrupt. In DMA mode, an interrupt is generated if the ERRIEN.

When an overrun error occurs:

- The ROERR bit is set.
- The data in the receive data buffer is not lost. The previous data is still available when the USART_DT register is read.
- The content in the receive shift register is overwritten. Afterwards, any data received will be lost.
- An interrupt is generated if the RDBFIEN is set or both ERRIEN and DMAREN are set.
- The ROERR bit is cleared by reading the USART_STS register and then USART_DT register in order.

Note: If ROERR is set, it indicates that at least one piece of data is lost, with two possibilities:

If RDBF=1, it indicates that the last valid data is still stored in the receive data buffer, and can be read.

If RDBF=0, it indicates that the last valid data in the receive data buffer has already been read.

Note: The REN bit cannot be reset during data reception, or the byte that is currently being received will be lost.

12.8.3 Start bit and noise detection

A start bit detection occurs when the REN bit is set. With the oversampling techniques, the USART receiver samples data on the 3rd, 5th, 7th, 8th, 9th and 10th bits to detect the valid start bit and noise. Table 12-2 shows the data sampling over start bit and noise detection.

Table 12-2 Data sampling over start bit and noise detection

Sampled value (3·5·7)	Sampled value (8·9·10)	NERR bit	Start bit validity
000	000	0	Valid
001/010/100	001/010/100	1	Valid
001/010/100	000	1	Valid
000	001/010/100	1	Valid
111/110/101/011	Any value	1	Invalid
Any value	111/110/101/011	1	Invalid

Note: If the sampling values on the 3rd, 5th, 7th, 8th, 9th, and 10th bits do not match the above

mentioned requirements, the USART receiver does not think that a correct start bit is received, and thus it will abort the start bit detection and return to idle state waiting for a falling edge.

The USART receiver has the ability to detect noise. In the non-synchronous mode, the USART receiver samples data on the 7th, 8th and 9th bits, with its oversampling techniques, to distinguish valid data input from noise based on different sampling values, and recover data as well as set NERR (Noise Error Flag) bit.

Table 12-3 Data sampling over valid data and noise detection

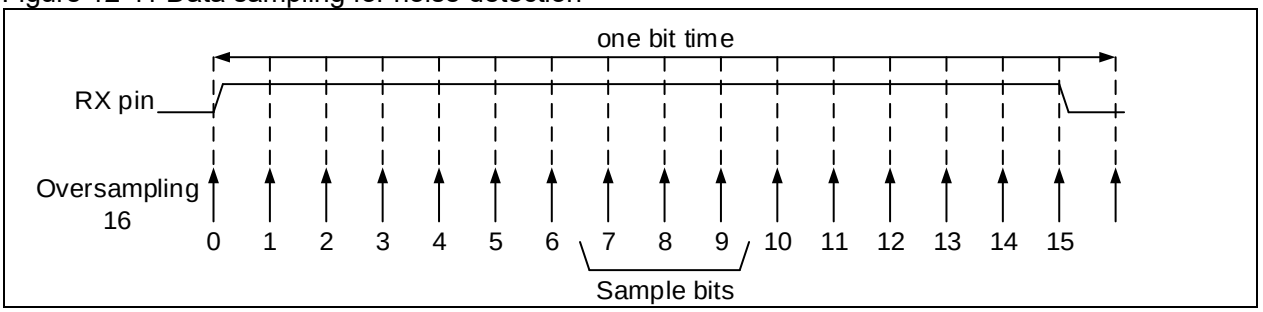
Sampled value	NERR bit	Received bit value	Data validity
000	0	0	Valid
001	1	0	Invalid
010	1	0	Invalid
011	1	1	Invalid
100	1	0	Invalid
101	1	1	Invalid
110	1	1	Invalid
111	0	1	Valid

When noise is detected in a data frame:

- The NERR bit is set at the same time as the RDBF bit
- The invalid data is transferred from the receive shift register to the receive data buffer.
- No interrupt is generated in non-DMA mode. However, since the NERR bit is set at the same time as the RDBF bit, the RDBF bit will generate an interrupt. In DMA mode, an interrupt will be issued if the ERRIEN is set.

The NERR bit is cleared by read access to USART_STS register followed by the USART_DT read operation.

Figure 12-11 Data sampling for noise detection

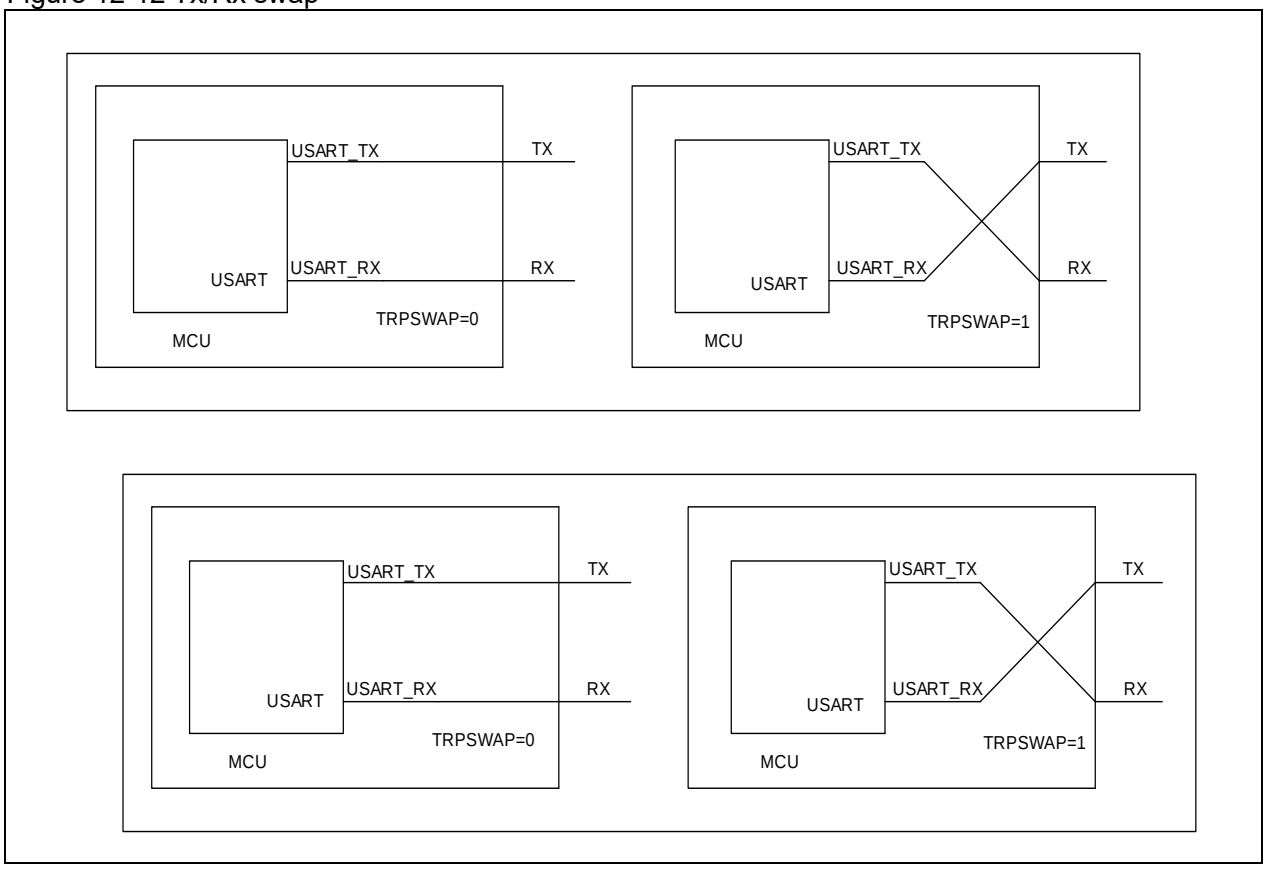


12.9 Tx/Rx swap

When the TRPSWAP bit (USART_CTRL2[15]) is set, Tx/Rx pin can be swapped. Two common scenes are listed below:

- If the Tx/Rx were reversed while the user attempts to connect the device externally to a RS-232 chip, they can be swapped through the TRPSWAP bit, without the need of hardware intervention.
- If the user only connected the master Tx to the slave Rx in full-duplex mode, the Tx/Rx can be interchangeable with the TRPSWAP bit, after the master and slave are swapped, without the need of hardware intervention.

Figure 12-12 Tx/Rx swap



Note: The SWAP (USART_CTRL2[15]) can be modified only when the USART is disabled (UEN=0)

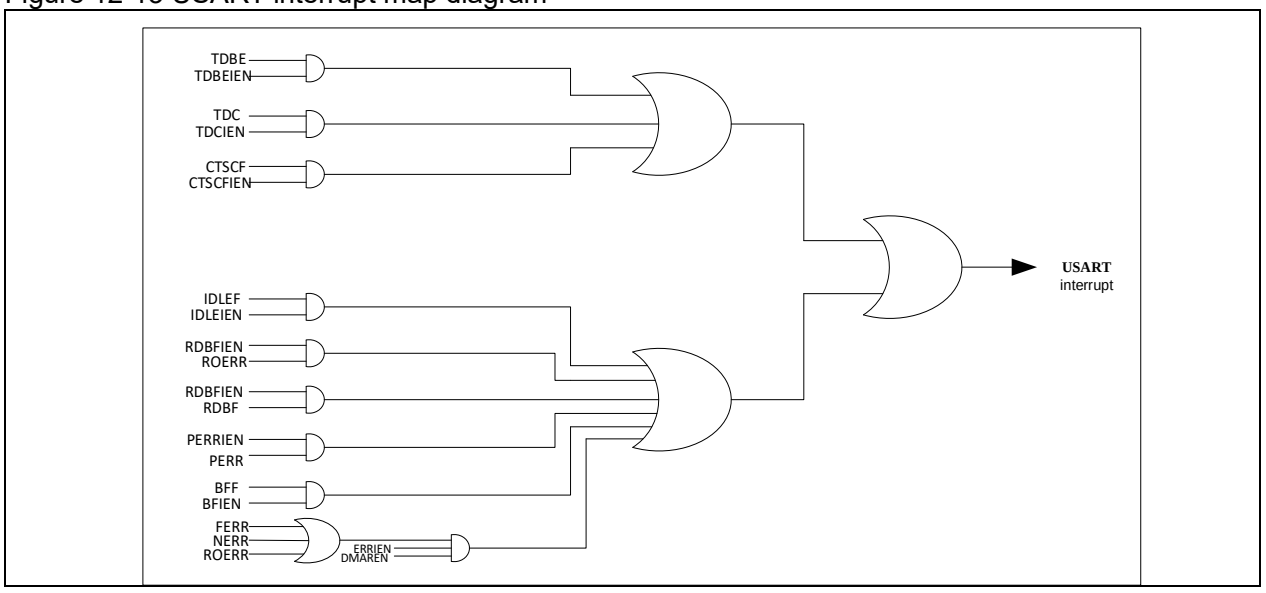
12.10 Interrupt requests

USART interrupt generator serves as a control center of USART interrupts. It is used to monitor the interrupt source inside the USART in real time and the generation of interrupts according to the programmed interrupt control bits. Table 12-4 shows the USART interrupt source and interrupt enable control bit. An interrupt will be generated over an event when the corresponding interrupt enable bit is set.

Table 12-4 USART interrupt request

Interrupt event	Event flag	Enable bit
Transmit data register empty	TDBE	TDBEIEN
CTS flag	CTSCF	CTSCFIEN
Transmit data complete	TDC	TDCIEN
Receive data buffer full	RDBF	RDBFIEN
Receiver overflow error	ROERR	
Idle flag	IDLEF	IDLEIEN
Parity error	PERR	PERRIEN
Break frame flag	BFF	BFIEN
Noise error, overflow error or framing error	NERR or ROERR or FERR	ERRIEN(1)

Figure 12-13 USART interrupt map diagram



12.11 I/O pin control

The following five interfaces are used for USART communication.

RX: Serial data input.

TX: Serial data output. In single-wire half-duplex and Smartcard mode, the TX pin is used as an I/O for data transmission and reception.

CK: Transmitter clock output. The output CLK phase, polarity and frequency can be programmable.

CTS: Transmitter input. Send enable signal in hardware flow control mode.

RTS: Receiver output. Send request signal in hardware flow control mode.

12.12 USART registers

These peripheral registers must be accessed by words (32 bits).

Table 12-5 USART register map and reset value

Register	Offset	Reset value
USART_STS	0x00	0x0000 00C0
USART_DT	0x04	0x0000 0000
USART_BAUDR	0x08	0x0000 0000
USART_CTRL1	0x0C	0x0000 0000
USART_CTRL2	0x10	0x0000 0000
USART_CTRL3	0x14	0x0000 0000
USART_GTP	0x18	0x0000 0000

12.12.1 Status register (USART_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 10	Reserved	0x000000	resd	Forced 0 by hardware.
Bit 9	CTSCF	0x0	rw0c	CTS change flag This bit is set by hardware when the CTS status line changes. It is cleared by software. 0: No change on the CTS status line 1: A change occurs on the CTS status line.
Bit 8	BFF	0x0	rw0c	Break frame flag This bit is set by hardware when a break frame is detected. It is cleared by software. 0: Break frame is not detected.

				1: Break frame is detected.
Bit 7	TDBE	0x1	ro	Transmit data buffer empty This bit is set by hardware when the transmit data buffer is empty. It is cleared by a USART_DT register write operation. 0: Data is not transferred to the shift register. 1: Data is transferred to the shift register.
Bit 6	TDC	0x1	rw0c	Transmit data complete This bit is set by hardware at the end of transmission. It is cleared by software. (Option 1: read access to USART_STS register followed by a USART_DT write operation; Option 2: Write “0” to this bit) 0: Transmission is not completed. 1: Transmission is completed.
Bit 5	RDBF	0x0	rw0c	Receive data buffer full This bit is set by hardware when the data is transferred from the shift register to the USART_DT register. It is cleared by software. (Option 1: read USART_DT register; Option 2: write “0” to this bit) 0: Data is not received. 1: Data is received.
Bit 4	IDLEF	0x0	ro	Idle flag This bit is set by hardware when an idle line is detected. It is cleared by software. (Read USART_DT register followed by a USART_DT read operation) 0: No idle line is detected. 1: Idle line is detected.
Bit 3	ROERR	0x0	ro	Receiver overflow error This bit is set by hardware when the data is received while the RDNE is still set. It is cleared by software. (Read USART_STS register followed by a USART_DT read operation) 0: No overflow error 1: Overflow error is detected. Note: When this bit is set, the DT register content will not be lost, but the subsequent data will be overwritten.
Bit 2	NERR	0x0	ro	Noise error This bit is set by hardware when noise is detect on a received frame. It is cleared by software. (Read USART_STS register followed by a USART_DT read operation) 0: No noise is detected. 1: Noise is detected.
Bit 1	FERR	0x0	ro	Framing error This bit is set by hardware when a stop bit error (low), excessive noise or break frame is detected. It is cleared by software. USART_STS register followed by a USART_DT read operation) 0: No framing error is detected. 1: Framing error is detected.
Bit 0	PERR	0x0	ro	Parity error This bit is set by hardware when parity error occurs. It is cleared by software. USART_STS register followed by a USART_DT read operation) 0: No parity error occurs. 1: Parity error occurs.

12.12.2 Data register (USART_DT)

Bit	Register	Reset value	Type	Description
Bit 31: 9	Reserved	0x000000	resd	Kept at its default value.
Bit 8: 0	DT	0x00	rw	Data value This register provides read and write function. When transmitting with the parity bit enabled, the value written in the MSB bit will be replaced by the parity bit. When

				receiving with the parity bit enabled, the value in the MSB bit is the received parity bit.

12.12.3 Baud rate register (USART_BAUDR)

Note: If the TE and RE are both disabled, the baud counter stops counting.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	DIV	0x0000	rw	Divider This field define the USART divider.

12.12.4 Control register1 (USART_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 31: 29	Reserved	0x0	resd	Kept at its default value.
Bit 28	DBN1	0x0	rw	Data bit num This bit, along with the DBN0 bit, is used to program the number of data bits. 10: 7 data bits 00: 8 data bits 01: 9 data bits 11: Write operation forbidden.
Bit 27: 26	Reserved	0x0	resd	Kept at its default value.
Bit 25 : 21	TSDT	0x00	rw	Transmit start delay time In RS485 mode, the first data (in sequential transmit mode) is transmitted after a period of time of being written so as to ensure that the transfer direction of the external transmitter/receiver to switch back to transmit. This time depends on the TSDT value, in unit of 1/16 baud rate.
Bit 20 : 16	TCDT	0x00	rw	transmit complete delay time In RS485 mode, a period of time (delay) is needed before the last data transfer is complete even if the last STOP bit has been transferred. This time duration allows the transfer direction of the external receiver/transmitter to switch back to receive. This time depends on the TCDT value, in unit of 1/16 baud rate.
Bit 15: 14	Reserved	0x0	resd	Kept at its default value.
Bit 13	UEN	0x0	rw	USART enable 0: USART is disabled. 1: USART is enable.
Bit 12	DBN0	0x0	rw	Data bit num This bit, along with DBN1, is used to program the number of data bits. 10: 7 data bits 00: 8 data bits 01: 9 data bits 11: Write operation forbidden.
Bit 11	WUM	0x0	rw	Wakeup mode This bit determines the way to wake up silent mode. 0: Waken up by idle line 1: Waken up by ID match
Bit 10	PEN	0x0	rw	Parity enable This bit is used to enable hardware parity control (generation of parity bit for transmission; detection of parity bit for reception). When this bit is enabled, the MSB bit of the transmitted data is replaced with the parity bit; Check whether the parity bit of the received data is correct. 0: Parity control is disabled. 1: Parity control is enabled.
Bit 9	PSEL	0x0	rw	Parity selection This bit selects the odd or even parity after the parity control is enabled. 0: Even parity 1: Odd parity

Bit 8	PERRIEN	0x0	rw	PERR interrupt enable 0: Interrupt is disabled. 1: Interrupt is enabled.
Bit 7	TDBEIEN	0x0	rw	TDBE interrupt enable 0: Interrupt is disabled. 1: Interrupt is enabled.
Bit 6	TDCIEN	0x0	rw	TDC interrupt enable 0: Interrupt is disabled. 1: Interrupt is enabled.
Bit 5	RDBFIEN	0x0	rw	RDBF interrupt enable 0: Interrupt is disabled. 1: Interrupt is enabled.
Bit 4	IDLEIEN	0x0	rw	IDLE interrupt enable 0: Interrupt is disabled. 1: Interrupt is enabled.
Bit 3	TEN	0x0	rw	Transmitter enable This bit enables the transmitter. 0: Transmitter is disabled. 1: Transmitter is enabled.
Bit 2	REN	0x0	rw	Receiver enable This bit enables the receiver. 0: Receiver is disabled. 1: Receiver is enabled.
Bit 1	RM	0x0	rw	Receiver mute This bit determines if the receiver is in mute mode or not. It is set or cleared by software. When the idle line is used to wake up from mute mode, this bit is cleared by hardware after wake up. When the address match is used to wake up from mute mode, it is cleared by hardware after wake up. When address mismatches, this bit is set by hardware to enter mute mode again. 0: Receiver is in active mode. 1: Receiver is in mute mode.
Bit 0	SBF	0x0	rw	Send break frame This bit is used to send a break frame. It can be set or cleared by software. Generally speaking, it is set by software and cleared by hardware at the end of break frame transmission. 0: No break frame is transmitted. 1: Break frame is transmitted.

12.12.5 Control register2 (USART_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 31: 28	IDH	0x0	rw	USART identification This field holds the upper four bits of USART ID. It is configurable.
Bit 27: 16	Reserved	0x000	resd	Kept at its default value.
Bit 15	TRPSWAP	0x0	rw	Transmit/receive pin swap 0: Transmit/receive pin is not swappable 1: Transmit/receive pin is swappable
Bit 14	LINEN	0x0	rw	LIN mode enable 0: LIN mode is disabled. 1: LIN mode is enabled.
Bit 13: 12	STOPBN	0x0	rw	STOP bit num These bits are used to program the number of stop bits. 00: 1 stop bit 01: 0.5 stop bit 10: 2 stop bits 11: 1.5 stop bits
Bit 11	CLKEN	0x0	rw	Clock enable This bit is used to enable the clock pin for synchronous mode or Smartcard mode. 0: Clock is disabled. 1: Clock is enabled.

Bit 10	CLKPOL	0x0	rw	Clock polarity In synchronous mode or Smartcard mode, this bit is used to select the polarity of the clock output on the clock pin in idle state. 0: Clock output low 1: Clock output high
Bit 9	CLKPHA	0x0	rw	Clock phase This bit is used to select the phase of the clock output on the clock pin in synchronous mode or Smartcard mode. 0: Data capture is done on the first clock edge. 1: Data capture is done on the second clock edge.
Bit 8	LBCP	0x0	rw	Last bit clock pulse This bit is used to select whether the clock pulse of the last data bit transmitted is output on the clock pin in synchronous mode. 0: The clock pulse of the last data bit is no output on the clock pin. 1: The clock pulse of the last data bit is output on the clock pin.
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6	BFIEN	0x0	rw	Break frame interrupt enable 0: Disabled 1: Enabled
Bit 5	BFBN	0x0	rw	Break frame bit num This bit is used to select 11-bit or 10-bit break frame. 0: 10-bit break frame 1: 11-bit break frame
Bit 4	IDBN	0x0	rw	Identification bit num This bit is used to select ID bit number. 0: 4 bit 1: Data bit - 1 bit Note: When this bit is set, in 7, 8 or 8-bit data mode, the ID bit number is the lower 6, 7 or 8 bit, respectively.
Bit 3: 0	IDL	0x0	rw	USART identification This field holds the lower four bits of USART ID. It is configurable.

Note: These three bits (CLKPOL, CLKPHA and LBCP) should not be changed while the transmission is enabled.

12.12.6 Control register3 (USART_CTRL3)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Forced 0 by hardware.
Bit 15	DEP	0x0	rw	DE polarity selection 0: High level active 1: Low level active
Bit 14	RS485EN	0x0	rw	RS485 enable This bit is used to enable RS485 mode. In RS485 mode, the USART controls the transfer direction of the external receiver/transmitter through the DE signal. 0: RS485 mode disabled. The control signal DE output is disabled. RTS pin is used in RS232 mode. 1: RS485 mode enabled. The control signal DE outputs on the RTS pin.
Bit 13 : 11	Reserved	0x0	resd	Forced 0 by hardware.
Bit 10	CTSCFIEN	0x0	rw	CTSCF interrupt enable 0: CTSCF interrupt disabled 1: CTSCF interrupt enabled
Bit 9	CTSEN	0x0	rw	CTS enable 0: CTS is disabled. 1: CTS is enabled.
Bit 8	RTSEN	0x0	rw	RTS enable 0: RTS is disabled. 1: RTS is enabled.

Bit 7	DMATEN	0x0	rw	DMA transmitter enable 0: DMA transmitter is disabled. 1: DMA transmitter is enabled.
Bit 6	DMAREN	0x0	rw	DMA receiver enable 0: DMA receiver is disabled. 1: DMA receiver is enabled.
Bit 5	SCMEN	0x0	rw	Smartcard mode enable 0: Smartcard mode is disabled. 1: Smartcard mode is enabled.
Bit 4	SCNACKEN	0x0	rw	Smartcard NACK enable This bit is used to send NACK when parity error occurs. 0: NACK is disabled when parity error occurs. 1: NACK is enabled when parity error occurs.
Bit 3	SLBEN	0x0	rw	Single-wire bidirectional half-duplex enable 0: Single-wire bidirectional half-duplex is disabled. 1: Single-wire bidirectional half-duplex is enabled.
Bit 2	IRDALP	0x0	rw	IrDA low-power mode This bit is used to configure IrDA low-power mode. 0: IrDA low-power mode is disabled. 1: IrDA low-power mode is enabled.
Bit 1	IRDAEN	0x0	rw	IrDA enable 0: IrDA is disabled. 1: IrDA is enabled.
Bit 0	ERRIEN	0x0	rw	Error interrupt enable An interrupt is generated when a framing error, overflow error or noise error occurs. 0: Error interrupt is disabled. 1: Error interrupt is enabled.

12.12.7 Guard time and divider register (USART_GDIV)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Forced 0 by hardware.
Bit 15: 8	SCGT	0x00	rw	Smartcard guard time value This field specifies the guard time value. The transmission complete flag is set after this guard time in smartcard mode.
Bit 7: 0	ISDIV	0x00	rw	IrDA/smartcard division In IrDA mode: 8 bit [7: 0] is valid. It is valid in common mode and must be set to 00000001. In low-power mode, it divides the peripheral clock to serve as the period base of the pulse width; 00000000: Reserved–Do not write. 00000001: Divided by 1 00000010: Divided by 2 …… Smartcard mode: The lower 5 bit [4: 0] is valid. This division is used to divide the peripheral clock to provide clock for the Smartcard. Configured as follows: 00000: Reserved–Do not write. 00001: Divided by 2 00010: Divided by 4 00011: Divided by 6 ……

13 Serial peripheral interface (SPI)

13.1 SPI introduction

The SPI interface supports either the SPI protocol or the I²S protocol, depending on software configuration. This chapter gives an introduction of the main features and configuration procedure of SPI used as SPI or I²S.

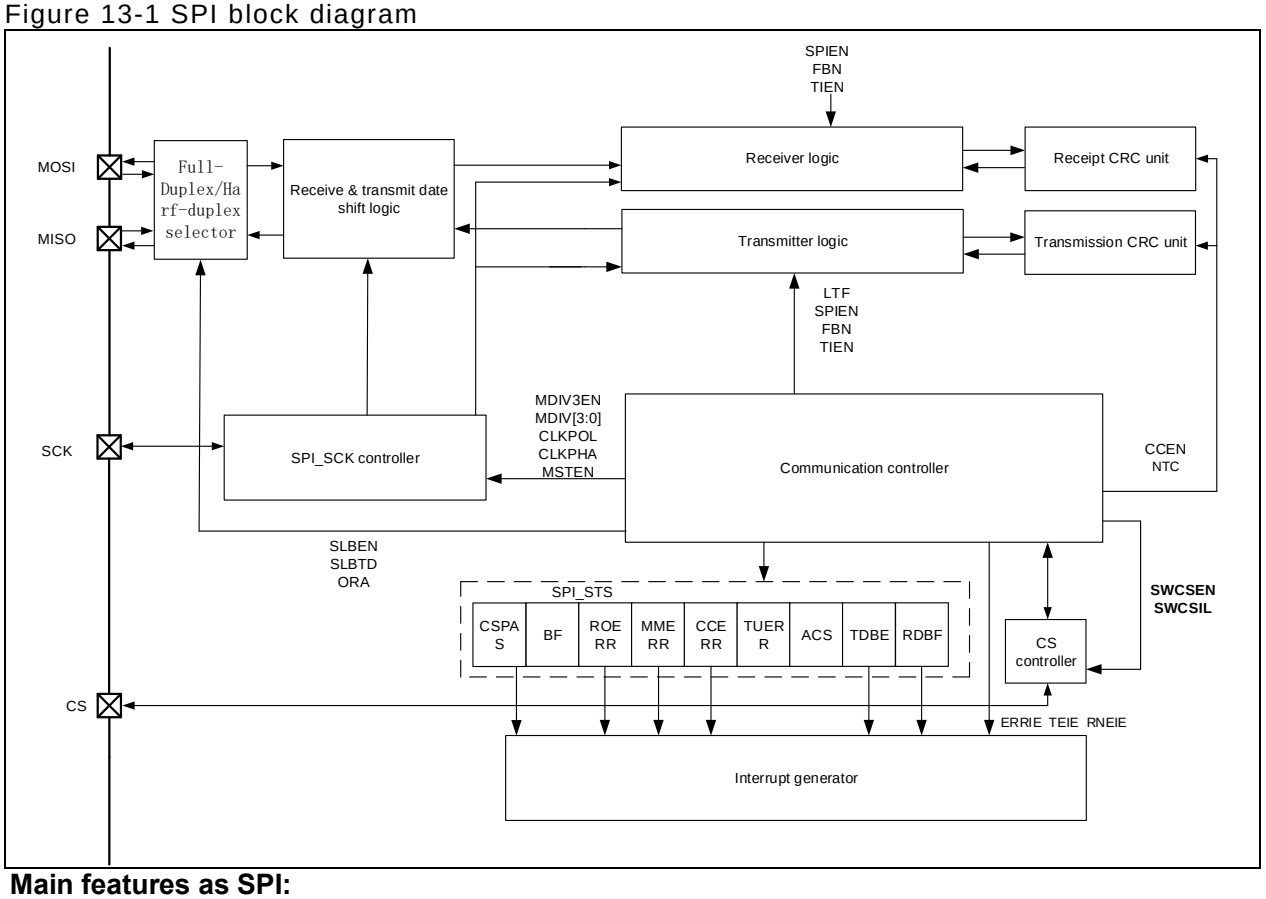
13.2 Function overview

13.2.1 SPI description

The SPI can be configured as host or slave based on software configuration, supporting full-duplex, reception-only full-duplex and transmission-only/reception-only half-duplex modes, DMA transfer, and automatic CRC function of SPI internal hardware. In the meantime, the SPI interface can be compatible with the TI protocol through software configurations.

SPI block diagram:

Figure 13-1 SPI block diagram



Main features as SPI:

- Full-duplex or half-duplex communication
  - — Full-duplex synchronous communication (supporting reception-only mode to release IO for transmission)
  - — Half-duplex synchronous communication (transfer direction is configurable: receive or transmit)
- Master or slave mode
- CS signal processing mode
  - — CS signal processing by hardware
  - — CS signal processing by software
- 8-bit or 16-bit frame format
- Communication frequency and prescalers (prescalers up to $f_{PCLK}/2$)

- Programmable clock polarity and phase
- Programmable data transfer order (MSB-first or LSB-first)
- Programmable error interrupt flags (CS pulse error, receiver overflow error, master mode error and CRC error)
- Programmable transmit data buffer empty interrupt and receive data buffer full interrupt
- Support transmission and reception using DMA
- Support hardware CRC transmission and error checking
- Busy status flag
- Compatible with the TI protocol

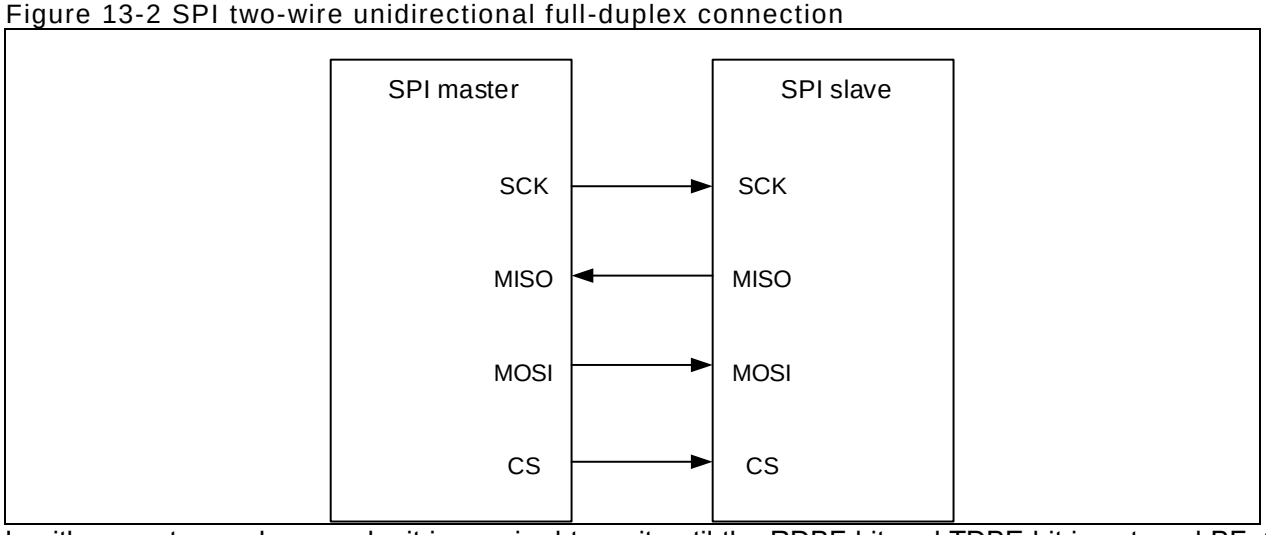
13.2.2 Full-duplex/half-duplex selector

When used as an SPI interface, it supports four synchronous modes: two-wire unidirectional full-duplex, single-wire unidirectional receive only, single-wire bidirectional half-duplex transmit and single-wire bidirectional half-duplex receive.

Figure 13-2 shows the two-wire unidirectional full-duplex mode and SPI IO connection:

The SPI operates in two-wire unidirectional full-duplex mode when the SLBEN bit and the ORA bit is both 0. In this case, the SPI supports data transmission and reception at the same time. IO connection is as follows:

Figure 13-2 SPI two-wire unidirectional full-duplex connection



In either master or slave mode, it is required to wait until the RDBF bit and TDBE bit is set, and BF=0 before disabling the SPI or entering power-saving mode (or disabling SPI system clock).

Figure 13-3 shows the single-wire unidirectional receive-only mode and SPI IO connection

The SPI operates in single-wire unidirectional receive-only mode when the SLBEN is 0 and the ORA is set. In this case, the SPI can be used only for data reception (transmission is not supported). The MISO pin transmits data in slave mode and receives data in master mode. The MOSI pin transmits data in master mode and receives data in salve mode.

Figure 13-3 Single-wire unidirectional receive only in SPI master mode

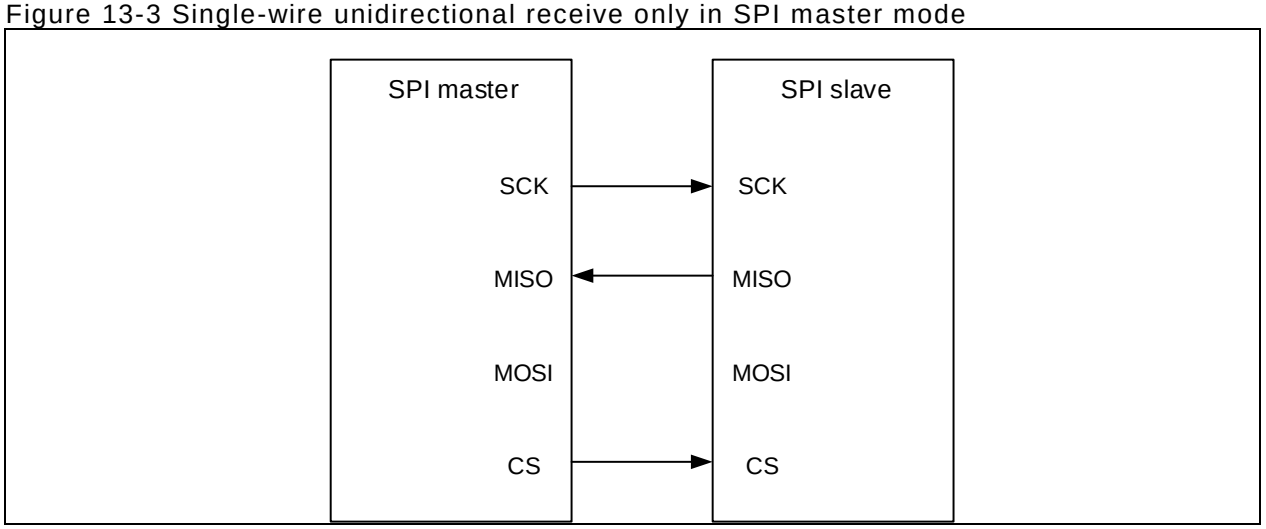
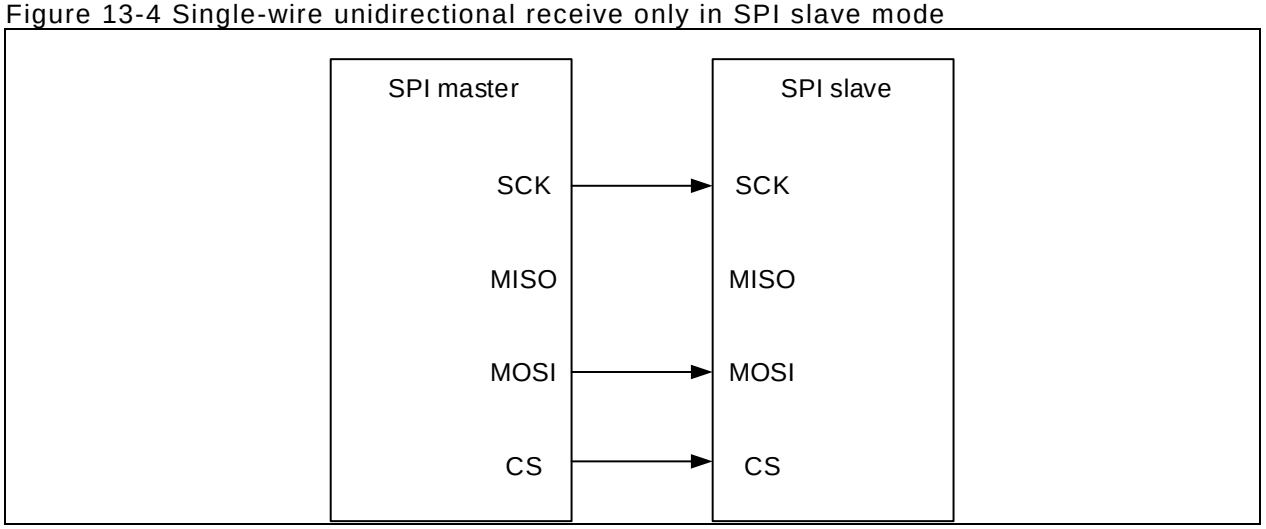


Figure 13-4 Single-wire unidirectional receive only in SPI slave mode



In master mode, it is necessary to wait until the second-to-last RDBF bit is set and then another SPI_CPK period before disabling the SPI. The last RDBF must be set before entering power-saving mode (or disabling SPI system clock).

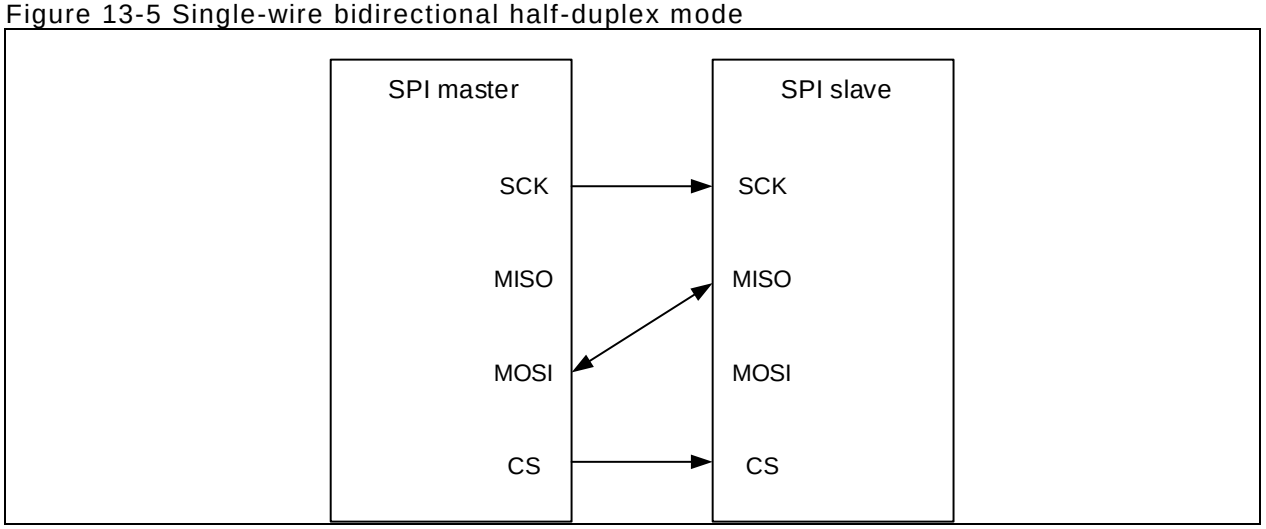
In slave mode, there is no need to check any flag before disabling the SPI. However, it is required to wait until the BF becomes 0 before entering power-saving mode.

Figure 13-5 shows single-wire bidirectional half-duplex mode and SPI IO connection

When the SLBEN is set, the SPI operates in single-wire bidirectional half-duplex mode. In this case, the SPI supports data reception and transmission alternately. In master mode, the MOSI pin transmits or receives data in master mode, while the MISO pin is released. In slave mode, the MISO pin transmits or receives data, but the MOSI pin is released.

The SLBTD bit is used by software to configure transfer direction. When the SLBTD bit is set, the SPI can be used only for data transmission; when the SLBTD bit is 0, the SPI can be used only for data reception.

Figure 13-5 Single-wire bidirectional half-duplex mode



When the SPI is selected for data transmission in single-wire bidirectional half-duplex mode (master or slave), the TDBE bit must be set, and the BF must be 0 before disabling the SPI. The power-saving mode (or disabling SPI system clock) cannot be entered unless the SPI is disabled.

In master mode, when the SPI is selected for data reception in single-wire bidirectional half-duplex mode, it is required to wait until the second-to-last RDBF is set and then another SPI_SCK period before disabling the SPI. And the last RDBF must be set before entering power-saving mode (or disabling SPI system clock).

In slave mode, when the SPI is selected for data reception in single-wire bidirectional half-duplex mode, there is no need to check any flags before disabling the SPI. However, the BT must be 0 before entering power-saving mode (or disabling SPI system clock).

13.2.3 Chip select controller

The Chip select controller (CS) is used to enable hardware or software control for chip select signals through software configuration. This controller is used to select master/slave device in multi-processor mode, and to avoid conflicts on the data lines by enabling the SCK signal output followed by CS signal. The hardware and software configuration procedure is detailed as follows, along with their respective input/output in master and slave mode.

CS hardware configuration procedure:

In master mode with CS being as an output, HWCSOE=1, SWCSEN=0, the CS hardware control is enabled. If the SPI is enabled, low level is output on the CS pin. The CS signal is then released after the SPI is disabled and the transmission is complete.

In master mode with CS being as an input, HWCSOE=0, SWCSEN=0, the CS hardware control is enabled. At this point, the SPI is automatically disabled by hardware and enters slave mode as soon as the CS pin low is detected by master SPI. The mode error flag (MMERR bit) is set at the same time. An interrupt is generated if ERRIE=1. When the MMERR is set, the SPIEN and MSTEN cannot be set by software. The MMERR is cleared by read or write access to the SPI_STS register followed by write operation to the SPI_CTRL1 register.

In slave mode with CS being as an input, HWCSOE=0, SWCSEN=0, the CS hardware control is enabled. The slave selects whether to transmit / receive data based on the level on the CS pin. The slave is selected for data reception and transmission only when the CS pin is low.

CS software configuration procedure:

In master mode with CS being as an input, SWCSEN=1, the CS software control is enabled. When SWCSIL=0, the SPI is automatically disabled by hardware and enters slave mode. The mode error flag (MMERR bit) is set at this time. An interrupt is generated if ERRIE=1. When the MMERR bit is set, the SPIEN and MSTEN bits cannot be set by software. The MMERR bit is cleared by read or write access to the SPI_STS register followed by write operation to the SPI_CTRL1 register.

In slave mode with CS being as an input, SWCSEN=1, the CS software control is enabled. The SPI judges the CS signal with the SWCSIL bit, instead of CS pin. When SWCSIL=0, the slave is selected for data reception and transmission.

13.2.4 SPI_SCK controller

The SPI protocol adopts synchronous transmission. In master mode with the SPI being used as SPI, it is required to generate a communication clock for data reception and transmission on the SPI, and the communication clock should be output to the slave via IO for data reception and transmission. In slave mode, the communication clock is provided by peripherals, and is input to the SPI via IO. In all, the SPI_SCK controller is used for the generation and distribution of SPI_SCK, with the configuration procedure detailed as follows:

SPI_SCK controller configuration procedure:

- Clock polarity and clock phase selection: It is selected by setting the CLKPOL and CLKPHA bit.
- Clock prescaler selection: Select the desired PCLK frequency by setting the CRM bit. Select the desired prescaler by setting the MDIV[3: 0] bit.
- Master/slave selection: Select SPI as master or slave by setting the MSTEN bit.

Note that the clock output is activated after the SPI is enabled in master reception-only mode, and it remains output until when the SPI is disabled and the reception is complete.

13.2.5 CRC

There is an independent transmission and reception CRC calculation unit in the SPI. When used as SPI through software configuration, the SPI enables CRC calculation and CRC check automatically while the user is reading or writing through DMA or CPU. During the transmission, if the received data is not consistent with, detected by hardware, the data in the SPI_RCRC register, and such data is exactly the CRC value, then the CCERR bit will be set. An interrupt is generated if ERRIE=1.

The CRC function and configuration procedure of the SPI are described as follows.

CRC configuration procedure

- CRC calculation polynomial is configured by setting the SPI_CPOLY register.
- CRC enable: The CRC calculation is enabled by setting the CCEN bit. This operation will reset the SPI_RCRC and SPI_TCRC registers.
- Select if or when the NTC bit is set, depending on DMA or CPU data register. See the following descriptions.

Transmission using DMA

When DMA is used to write the data to be transmitted, if the CCEN bit is enabled, the hardware calculates the CRC value automatically according to the value in the SPI_CPOLY register and each transmitted data, and sends the CRC value at the end of the last data transmission. This result is regarded as the value of the SPI_TCRC register.

Reception using DMA

When DMA is used to read the data to be received, if the CCEN bit is enabled, the hardware calculates the CRC value automatically according to the value in the SPI_CPOLY register and each received data, and waits until the completion of CRC data reception at the end of the last data reception before comparing the received CRC value with the value of the SPI_RCRC register. If check error occurs, the CCERR flag is set. An interrupt is generated if the ERRIE bit is enabled.

Transmission using CPU

Unlike DMA mode, after writing the last data to be transmitted, the CPU mode requires the NTC bit to be set by software before the end of the last data transmission.

Reception using CPU

In two-wire unidirectional full-duplex mode, follow CPU transmission mode to operate the NTC bit, the CRC calculation and check in CPU reception mode will be completed automatically.

In single-wire unidirectional reception-only mode and single-wire bidirectional reception-only mode, it is required to set the NTC bit before the software receives the last data when the second-to-last data is received.

13.2.6 DMA transfer

The SPI supports write and read operations with DMA. Refer to the following configuration procedure.

Special attention should be paid to: when the CRC calculation and check is enabled, the number of data transferred by DMA is configured as the number of the data to be transferred. The number of data read with DMA is configured as the number of the data to be received. In this case, the hardware will send CRC automatically at the end of full transfer, and the receiver will also perform CRC check. Note that the received CRC data will be moved into the SPI_DT register by hardware, with the RDBF being set, and the DMA read request will be sent if then DAM transfer is enabled. Hence, it is recommended to read the SPI_DT register to get the CRC value at the end of CRC reception in order to avoid the upcoming transfer error.

Transmission with DMA

- Select DMA channel: Select a DMA channel for the current SPI transfer from DMA flexible request map table described in DMA chapter.
- Configure the destination of DMA transfer: Configure the SPI_DT register address as the destination address bit of DMA transfer in the DMA control register. Data will be sent to this address after transmit request is received by DMA.
- Configure the source of DMA transfer: Configure the memory address as the source of DMA transfer in the DMA control register. Data will be loaded into the SPI_DT register from the memory address after transmit request is received by DMA.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure the channel priority of DMA transfer in the DMA control register.
- Configure DMA interrupt generation after half or full transfer in the DMA control register.
- Enable DMA transfer channel in the DMA control register.

Reception with DMA

- Select DMA transfer channel: Select a DMA channel for the current SPI from DMA flexible request map table described in DMA chapter.
- Configure the destination of DMA transfer: Configure the memory address as the destination of DMA transfer in the DMA control register. Data will be loaded from the SPI_DT register to the programmed destination after reception request is received by DMA.
- Configure the source of DMA transfer: Configure the SPI_DT register address as the source of DMA transfer in the DMA control register. Data will be loaded from the SPI_DT register to the programmed destination after reception request is received by DMA.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure DMA interrupt generation after half or full transfer in the DMA control register
- Enable DMA transfer channel in the DMA control register.

13.2.7 TI mode

The SPI interface is compatible with the TI protocol. The TI mode is enabled by setting the TIEN bit.

In this mode, the SPI interface will generates a communication clock SPI_CLK in accordance with the TI protocol. This means that the SPI_CLK polarity and phase are forced to conform to the TI protocol requirements, without the need of the intervention of CLKPOL and CLKPHA bits. Thus the CLKPOL and CLKPHA bits cannot be used to change the polarity and phase of the SPI_CLK either.

In this mode, the SPI interface will generate a CS signal in accordance with the TI protocol, meaning that the CS input and output are forced to conform to the TI protocol requirements, without the need of the intervention of SWCSEN, SWCSIL and HWCSOE bits. Thus, the SWCSEN, SWCSIL and HWCSOE bits cannot be used for CS signal management either.

In slave mode, once the TI mode is enabled, the SPI slave controls the MISO pin only during data transmission, meaning that the MISO pin state remains Hi-Z in idle state.

In slave mode, once the TI mode is enabled, the SPI interface is capable of detecting CS pulse errors during data transmission, and setting the CSPAS bit (It is cleared by reading the SPI_STS) as soon as

a CS pulse error is detected. At this point, the detected pulse error will be discarded by the SPI. However, since there is something wrong with the CS signal, the software should disable the SPI slave and re-configure the SPI master before re-enabling the SPI slave for communication.

13.2.8 Transmitter

The SPI transmitter is clocked by SPI_SCK controller. It can output different data frame formats, depending on software configuration. There is a SPI_DT register available in the SPI that is used to be written with the data to be transmitted. When the transmitter is clocked, the contents in the SPI_DT register are copied into the data buffer (Unlike SPI_DT, it is driven by SPI_SCK, and controlled by hardware, instead of software), and sent out in order based on the programmed frame format.

Both DMA and CPU can be used for write operation. For DMA transfer, refer to DMA transfer section for more details. For CPU transfer, attention should be paid to the TDBE bit. The reset value of this bit is 1, indicating that the SPI_DT register is empty. If the TDBEIE bit is set, an interrupt is generated. After the data is written, the TDBE is pulled low until the data is moved to the transmit data buffer before the TDBE is set once again. This means that the user can be allowed to write the data to be transmitted only when the TDBE is set.

After the transmitter is configured and the SPI is enabled, the SPI is ready for data transmission. Before going forward, it is necessary for the users to refer to full-duplex / half-duplex chapter to get detailed configuration information, go to the Chip select controller chapter for specific chip select mode, check the SPI_SCK controller chapter for information on communication clock, and refer to CRC and DMA transfer chapter to configure CRC and DMA (if necessary). The recommended configuration procedure are as follows.

Transmitter configuration procedure:

- Configure full-duplex/half-duplex selector
- Configure chip select controller
- Configure SPI_SCK controller
- Configure CRC (if necessary)
- Configure DMA transfer (if necessary)
- If the DMA transfer mode is not used, the software will check whether to enable transmit data interrupt (TDBEIE =1) through the TDBE bit.
- Configure frame format: select MSB/LSB mode with the LTF bit, and select 8/16-bit data with the FBN bit
- Enable SPI by setting the SPIEN

13.2.9 Receiver

The SPI receiver is clocked by the SPI_SCK controller. It can output different data frame formats through software configuration. There is a receive data buffer register, driven by the SPI_SCK, in the SPI receiver.

At the last CLK of each transfer, the data is moved from the shift register to the receive data buffer register. Then the transmitter sets the receive data complete flag to the SPI logic. When the flag is detected by the SPI logic, the data in the receive data buffer is copied into the SPI_DT register, with the RDBF being set. This means that the data is received, and it is already stored into the SPI_DT. In this case, read access to the SPI_DT register will clear the RDBF bit.

Both DMA and CPU can be used for read operation. For DMA transfer, refer to DMA transfer section for more details. For CPU transfer, attention should be paid to the RDBE bit. The reset value of this bit is 0, indicating that the SPI_DT register is empty. If the data is received and moved into the SPI_DT, the RDBF is set, meaning that there are some data to be read in the SPI_DT register. An interrupt is generated if the RDBFIE bit is set.

When the next received data is ready to be moved to the SPI_DT register, if the previous received data is still not read (RDBF=1), then the data overflow occurs. The previous receive data is not lost, but the next received data will do. At this point, the ROERR is set. An interrupt is generated if the ERRIE is set. Read SPI_DT register and then the SPI_STS register will clear the ROERR bit. The recommended configuration procedure is as follows.

Receiver configuration procedure:

- Configure full-duplex/half-duplex selector
- Configure chip select controller
- Configure SPI_SCK controller
- Configure CRC (if necessary)
- Configure DMA transfer (if necessary)
- If the DMA transfer mode is not used, the software will check whether to enable receive data interrupt (RDBEIE =1) through the RDBE bit.
- Configure frame format: select MSB/LSB mode with the LTF bit, and select 8/16-bit data with the FBN bit
- Enable SPI by setting the SPIEN

13.2.10 Motorola mode

This section describes the SPI communication timings, which includes full-duplex and half-duplex master/slave timings.

Full-duplex communication – master mode

Configured as follows:

MSTEN=1: Master enable

SLBEN=0: Full-duplex mode

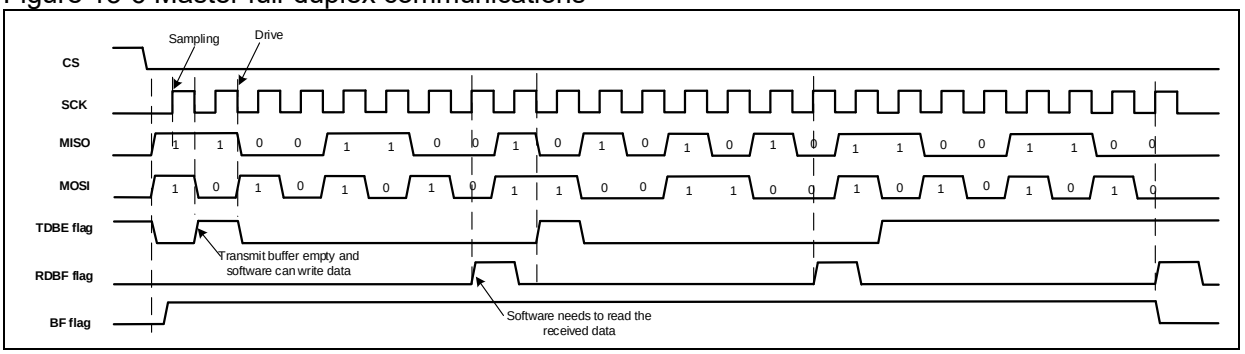
CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Master transmit (MOSI): 0xaa, 0xcc, 0xaa

Slave transmit (MISO): 0xcc, 0xaa, 0xcc

Figure 13-6 Master full-duplex communications



Full-duplex communication – slave mode

Configured as follows:

MSTEN=0: Slave enable

SLBEN=0: Full-duplex mode

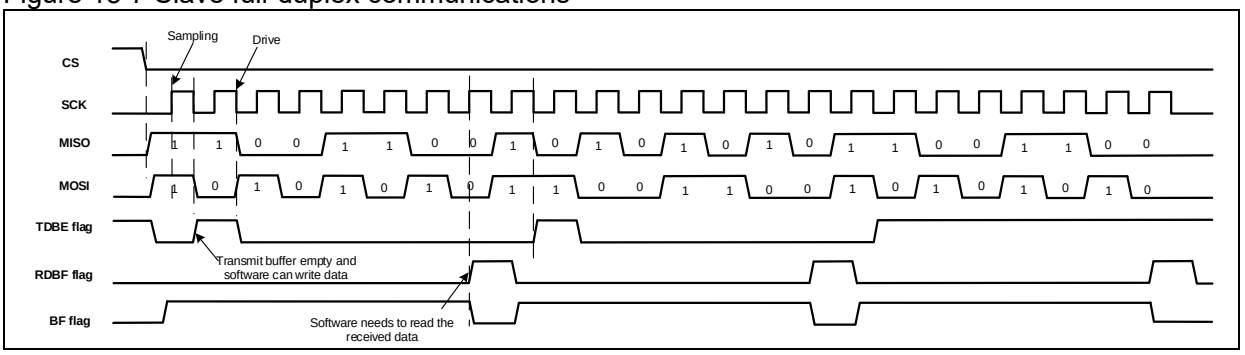
CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Master transmit (MOSI): 0xaa, 0xcc, 0xaa

Slave transmit (MISO): 0xcc, 0xaa, 0xcc

Figure 13-7 Slave full-duplex communications



Half-duplex communication – master transmit

Configured as follows:

MSTEN=1: Master enable

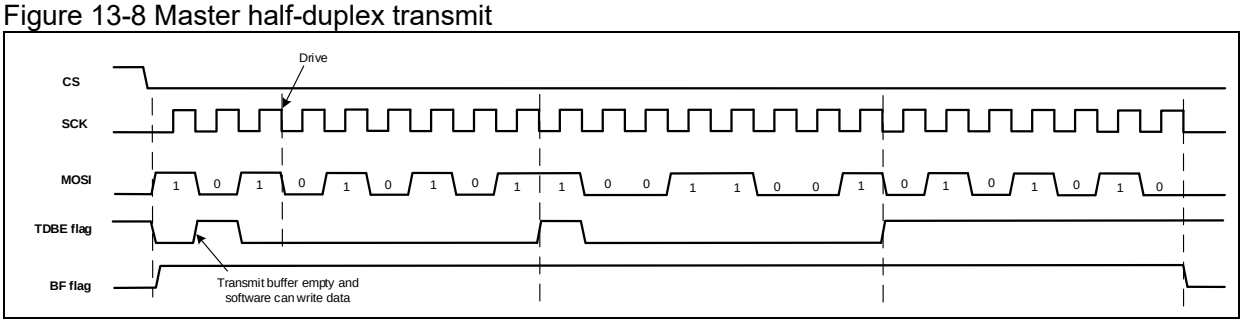
SLBEN=1: Single line bidirectional mode

CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Master transmit (MOSI): 0xaa, 0xcc, 0xaa

Figure 13-8 Master half-duplex transmit



Half-duplex communication – slave receive

Configured as follows:

MSTEN=0: Slave enable

SLBEN=1: Single line bidirectional mode

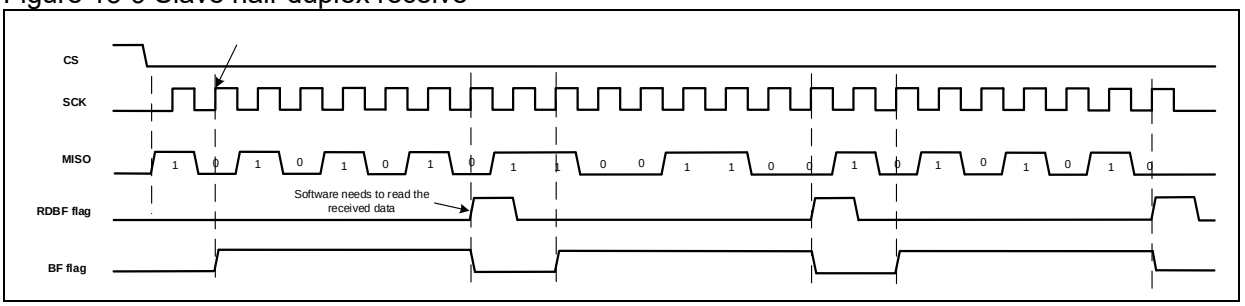
SLBTD=0: Receive mode

CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Slave receive: 0xaa, 0xcc, 0xaa

Figure 13-9 Slave half-duplex receive



Half-duplex communication – slave transmit

Configured as follows:

MSTEN=0: Slave enable

SLBEN=1: Single line bidirectional mode

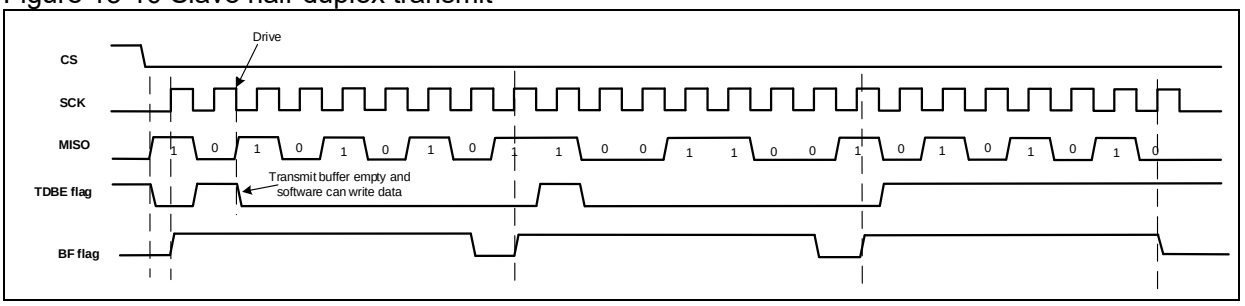
SLBTD=1: Transmit enable

CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Slave transmit: 0xaa, 0xcc, 0xaa

Figure 13-10 Slave half-duplex transmit



Half-duplex communication – master receive

Configured as follows:

MSTEN=1: Master enable

SLBEN=1: Single line bidirectional mode

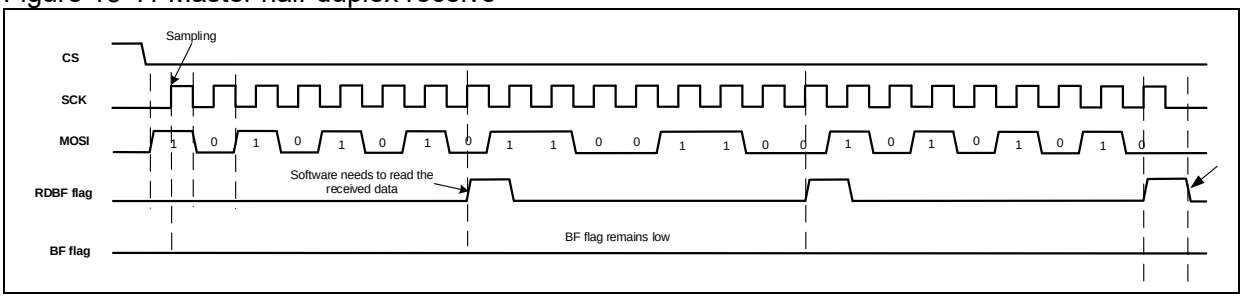
SLBTD=0: Receive enable

CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Master receive: 0xaa, 0xcc, 0xaa

Figure 13-11 Master half-duplex receive

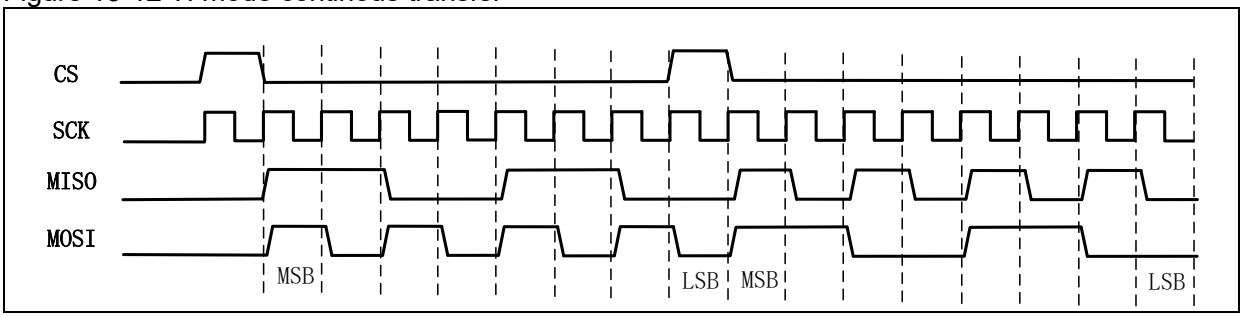


13.2.11 TI mode

The SPI interface supports TI mode. The TIEN bit can be set to enable SPI TI mode.

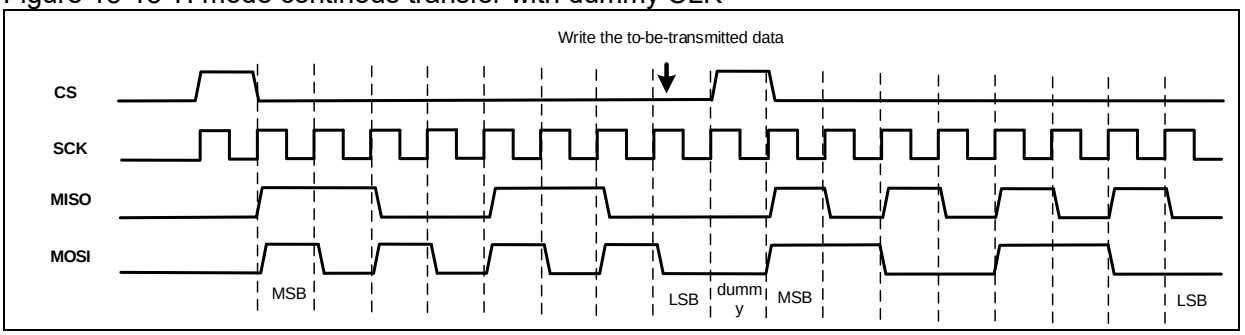
In TI mode, a bit of different is present between continuous and discontinuous communication timings. When the to-be transmitted data is written before the rising SCK edge corresponding to the last data of the current transmit frame, it is a continuous communication, without dummy CLK between data, and the host sends a valid CS pulse while transmitting the last data of the current frame.

Figure 13-12 TI mode continous transfer



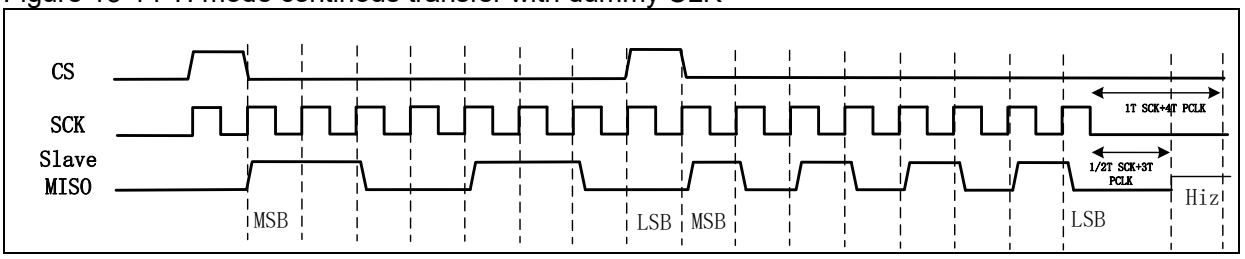
When the to-be-transmitted data is written between the rising and falling SCK edge corresponding to the last data of the current transmit frame, a dummy CLK exists between data.

Figure 13-13 TI mode continous transfer with dummy CLK



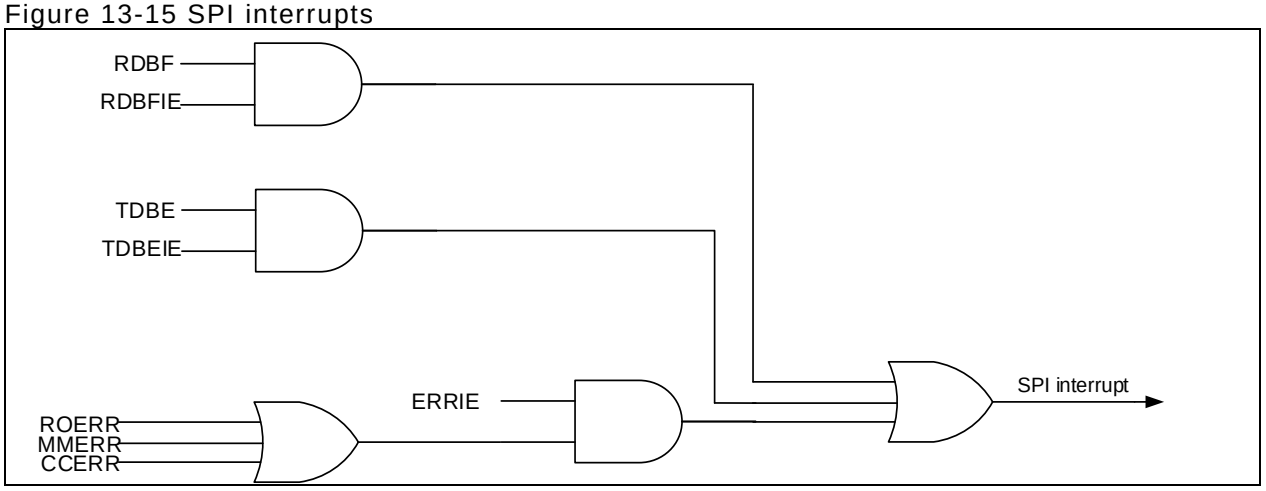
When the to-be-transmitted data is written after the falling SCK edge corresponding to the last data of the current transmit frame, the host always issues a valid SCK clock after 1T SCK + 4T PCLK. If the slave still does not detect a valid CS pulse at the end of the current data reception, it disables MISO output after 1/2T SCK + 3T PCLK to control MISO floating.

Figure 13-14 TI mode continous transfer with dummy CLK



13.2.12 Interrupts

Figure 13-15 SPI interrupts



13.2.13 IO pin control

Usually, the SPI is connected to external devices through four pins.

- MISO: Master In/Slave Out. The pin receives data in master mode, and transmits data in slave mode.
- MOSI: Master Out/Slave In. The pin transmits data in master mode, and receives data in slave mode.
- SCK: SPI communication clock. The pin serves as output in master mode, and input in slave mode.
- CS: Chip Select. This is an optional pin which selects master/slave mode.

Note: Some of SPI1/I²S1 and SPI3/I²S3 are shared with JTAG pins (SPIx_CS/I2Sx_WS shared with JTDI, SPIx_SCK/I2Sx_CK with JTDO), so these pins are not controlled by IO controller, and they are used as JTAG by default after each reset. To configure them as SPIx/I²Sx, the JTAG should be disabled (during debugging) and switched to SWD interface, or both the JTAG and SWD are disabled (during normal run)

13.2.14 Precautions

- CRC value is obtained by software reading DT register at the end of CRC reception
- In the case of CPOL=1 and CPHA=1, the clock divided by 3 that is generated inside the SPI must be less than 32 MHz. To achieve a greater communication frequency, it is necessary to use a clock divided by 2, and adjust the corresponding HCLK and PCLK frequencies. The SPI frequencies must not exceed the maximum value programmed in the corresponding datasheet.

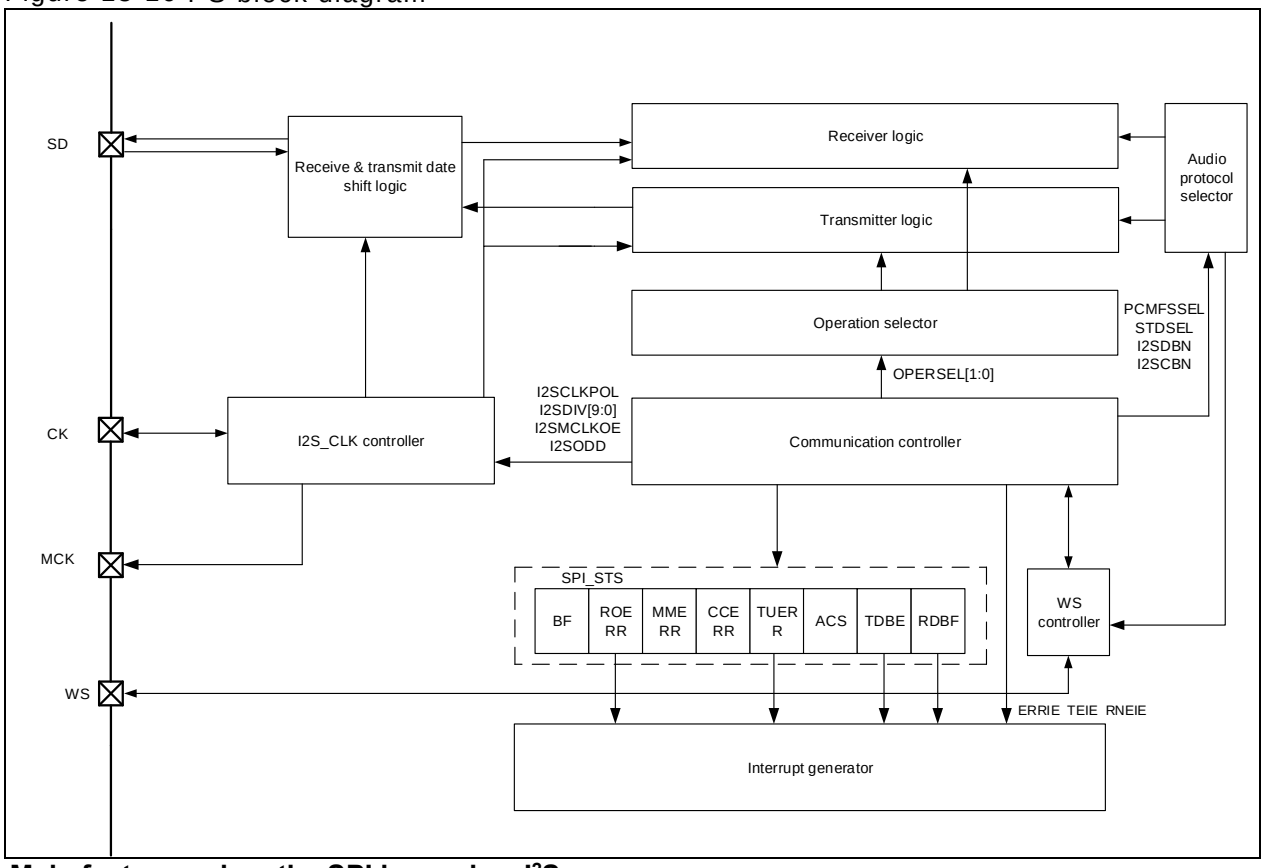
13.3 I²S functional description

13.3.1 I²S introduction

The I²S can be configured by software as master reception/transmission, and slave reception/transmission, supporting four kinds of audio protocols including Philips standard, MSB-aligned standard, LSB-aligned standard and PCM standard, respectively. The DMA transfer is also supported.

A single I²S supports half-duplex. However, it can work with two additional instantiated I²S modules (I²S2EXT and I²S3EXT) to achieve full-duplex mode. In other words, combining the I²S2 with the I²S2EXT enables the I²S2 to support full-duplex mode. This is true for the I²S3 through the combination of the I²S3 with the I²S3EXT. Refer to I²S full-duplex section for more information.

Figure 13-16 I²S block diagram



Main features when the SPI is used as I²S:

- Programmable operation mode
  - Slave device transmission
  - Slave device reception
  - Master device transmission
  - Master device reception
- Programmable clock polarity
- Programmable clock frequency (8 KHz to 192 KHz)
- Programmable data bits (16 bit, 24 bit, 32 bit)
- Programmable channel bits (16 bit, 32 bit)

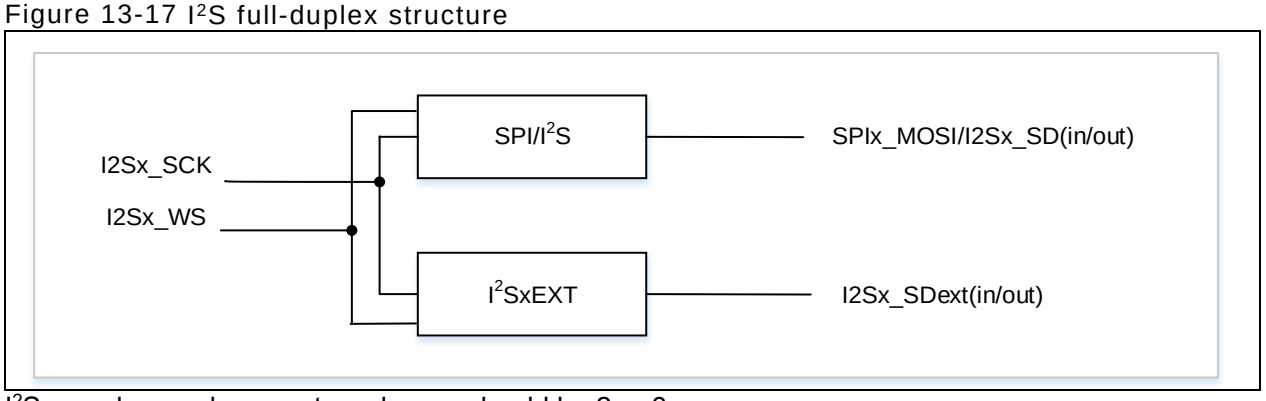
- Programmable audio protocol
  - I²S Philips standard
  - MSB-aligned standard (left-aligned)
  - LSB-aligned standard (right-aligned)
  - PCM standard (long or short frame)
- I²S full-duplex
- DMA transfer
- Main peripheral clock with a fixed frequency of 256x Fs (audio sampling frequency)

13.3.2 I²S full-duplex

Two extra I²S modules (I²S2EXT and I²S3EXT) are used to support I²S full-duplex mode. Combine the I²S2 with the I²S2EXT, and I²S3 with I²S3EXT to support full-duplex mode.

Note: I²S2EXT and I²S3EXT only used for I²S full-duplex mode.

Figure 13-17 I²S full-duplex structure



I²Sx can be used as master, where x should be 2 or 3

- In half-duplex mode, only I²Sx can output SCK and WS
- In full-duplex mode, only I²Sx can output SCK and WS

I²SxEXT is only used for full-duplex mode, and I²SxEXT only for slave mode

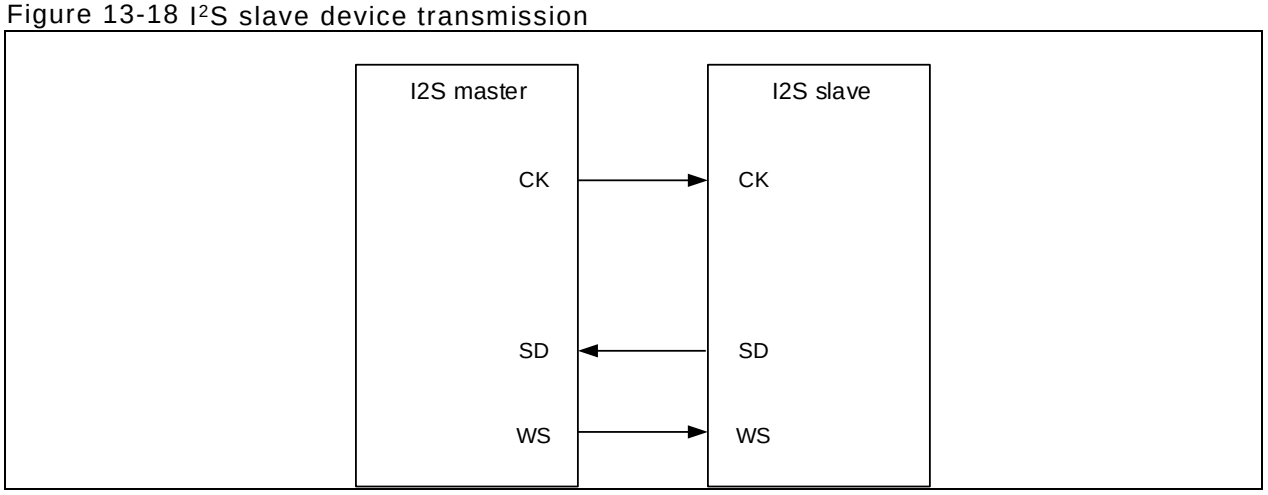
Both the I²Sx and I²SxEXT can be configured as transmit or receive mode.

13.3.3 Operating mode selector

The SPI, used as I²S selector, offers multiple operating modes for selection, namely, slave device transmission, slave device reception, master device transmission and master device reception. This is done by software configuration.

Slave device transmission:
Set the I2SMSEL bit, and OPERSEL[1:0] =00, the I²S will work in slave device transmission mode.

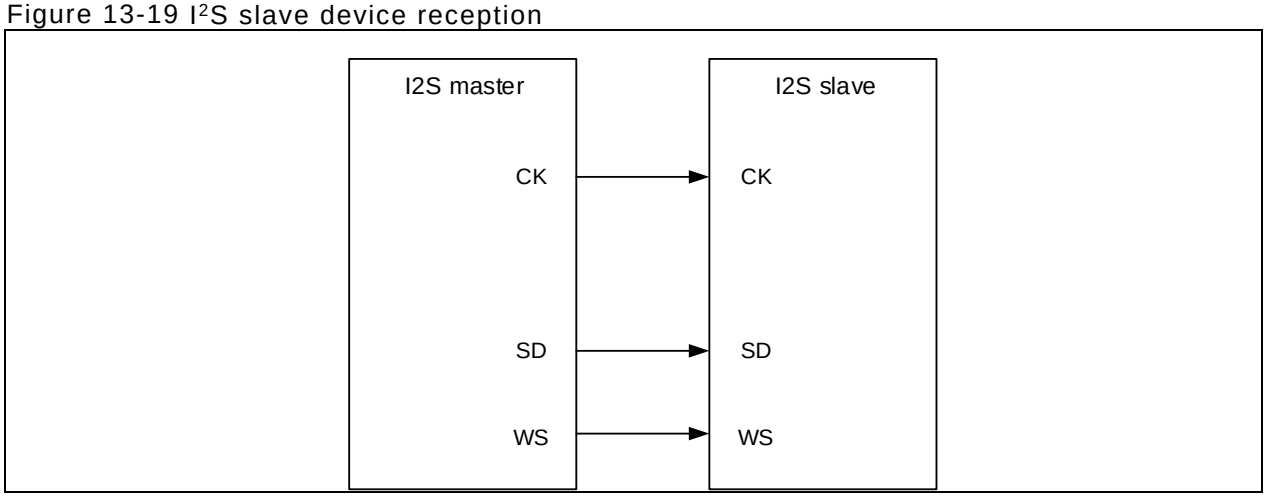
Figure 13-18 I²S slave device transmission



Slave device reception:

Set the I2SMSEL bit, and OPERSEL[1:0]=01, the I²S will work in slave device reception mode.

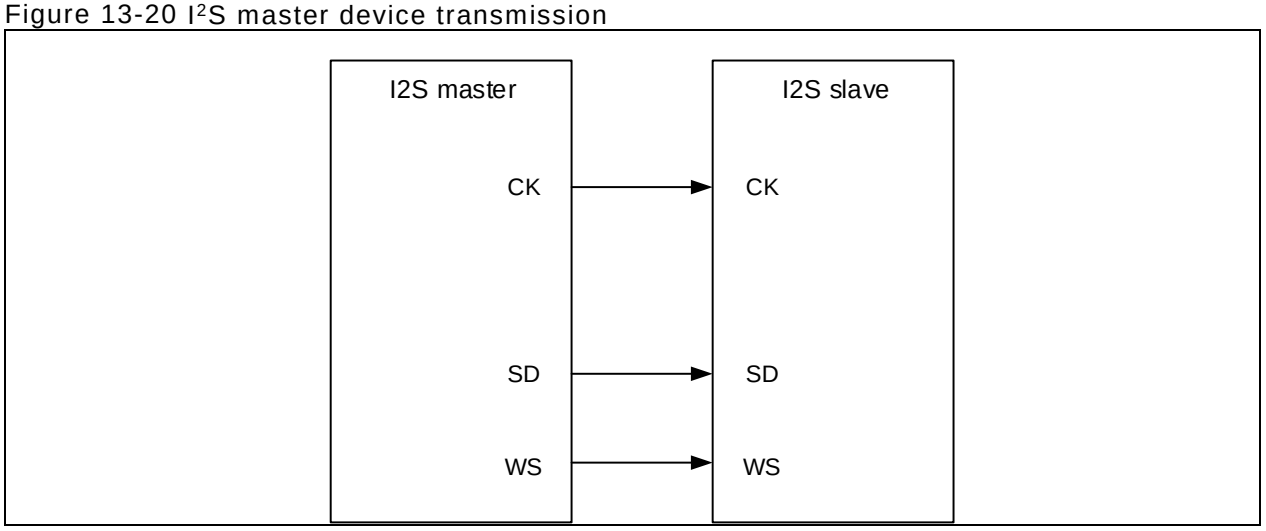
Figure 13-19 I²S slave device reception



Master device transmission:

Set the I2SMSEL bit, and OPERSEL[1:0]=10, the I²S will work in master device transmission mode.

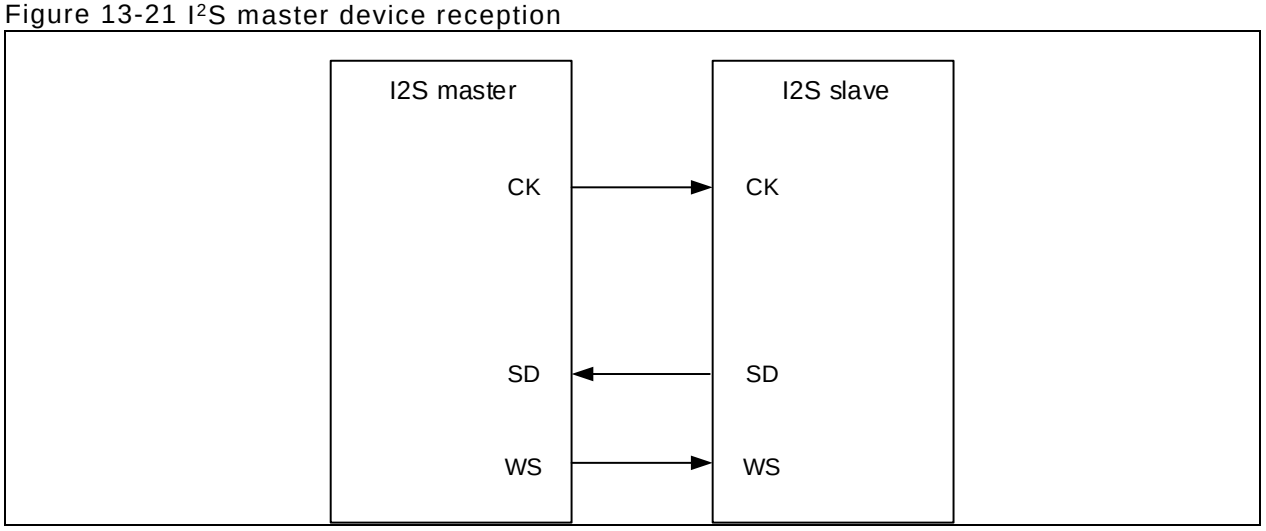
Figure 13-20 I²S master device transmission



Master device reception:

Set the I2SMSEL bit, and OPERSEL[1: 0]=11, the I²S will work in master device reception mode.

Figure 13-21 I²S master device reception



13.3.4 Audio protocol selector

While being used as I²S, the SPI supports multiple audio protocols. The user can control the audio protocol selector through software configuration to select the desired audio protocol, with the data bits and channel bits being controlled by the audio protocol selector. Besides, the user can also select the data bits and channel bits through software configuration. Meanwhile, the audio protocol selector manages the WS controller, output or detect the WS signal that meets the protocol requirements.

- Select audio protocol by setting the STDSEL bit

STDSLE=00: Philips standard
STDSLE=01: MSB-aligned standard (left-aligned)
STDSLE=10: LSB-aligned standard (right-aligned)
STDSLE=11: PCM standard

- Select PCM frame synchronization format: PCMFSSEL=1 for PCM long frame synchronization, PCMFSSEL=0 for short frame synchronization (this step is required when selecting PCM protocol)
- Select data bits by setting the I2SDBN bit

I2SDBN=00: 16 bit
I2SDBN =01: 24 bit
I2SDBN =10: 32 bit

- Select channel bits by setting the I2SCBN bit

I2SCBN =0: 16 bit
I2SCBN =1: 32 bit

Note: Read/Write operation mode depends on the selected audio protocols, data bits and channel bits. The following lists all possible configuration combinations and their respective read and write operation mode.

- Philips standard, PCM standard, MSB-aligned or LSB-aligned standard, 16-bit data and 16-bit channel

The data bit is the same as the channel bit. Each channel requires one read/write operation from/ to the SPI_DT register, and the number of DMA transfer is 1.

- Philips standard, PCM standard or MSB-aligned standard, 16-bit data and 32-bit channel

The data bit is different from the channel bit. Each channel requires one read/write operation from/to the SPI_DT register, and the number of DMA transfer is 1. The first 16 bits (MSB) are the significant bits, and the 16-bit LSB is forced to 0 by hardware.

- Philips standard, PCM standard or MSB-aligned standard, 24-bit data and 32-bit channel

The data bit is different from the channel bit. Each channel requires two read/write operations from/to the SPI_DT register, and the number of DMA transfer is 2. The 16-bit MSB transmits and receives the first 16-bit data, the 16-bit LSB transmits and receives the 8-bit MSB data, with 8-bit LSB data being forced to 0 by hardware.

- Philips standard, PCM standard, MSB-aligned or LSB-aligned standard, 32-bit data and 32-bit channel

The data bit is the same as the channel bit. Each channel requires two read/write operations from/to the SPI_DT register, and the number of DMA transfer is 2. These 32-bit data are proceeded in two times, with 16-bit data each time.

- LSB-aligned standard, 16-bit data and 32-bit channel

The data bit is different from the channel bit. Each channel requires one read/write operation from/to the SPI_DT register, and the number of DMA transfer is 1. The 16 bits (LSB) are the significant bits while the first 16-bit data (MSB) are forced to 0 by hardware.

- LSB-aligned standard, 24-bit data and 32-bit channel

The data bit is different from the channel bit. Each channel requires two read/write operations from/to the SPI_DT register, and the number of DMA transfer is 2. For the first 16-bit data, its 8-bit LSB are the significant bits, with the 8-bit MSB forced to 0 by hardware; the subsequent 16 bits transmit and receive the second 16-bit data.

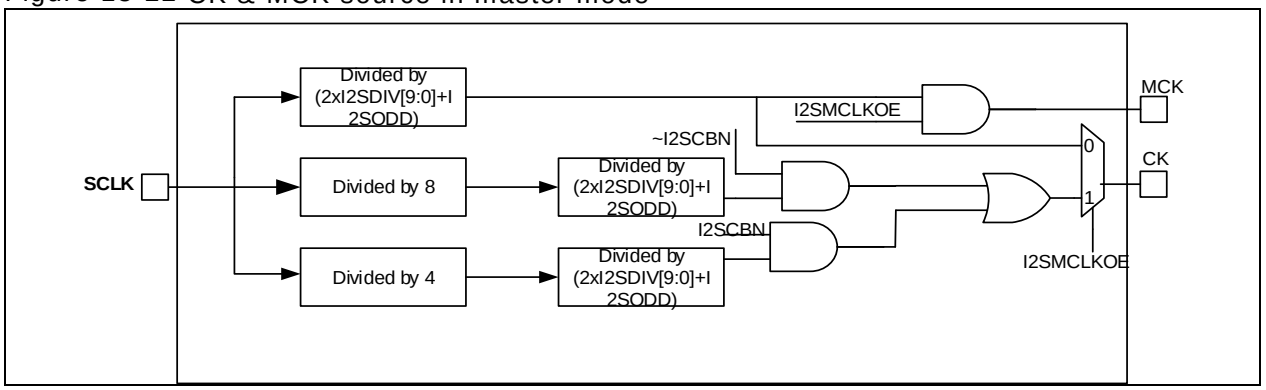
13.3.5 I2S_CLK controller

The audio protocols the SPI supports adopts synchronous transmission. In master mode, it is required to generate a communication clock for data reception and transmission on the SPI, and the communication clock should be output to the slave via IO for data reception and transmission. In slave mode, the communication clock is provided by master, and is input to the SPI via IO. In all, the I2S_SCK controller is used for the generation and distribution of I2S_SCK, with the configuration procedure detailed as follows:

When used as I2S master, the SPI can provide communication clock (CK) and main peripheral clock (MCK) shown in Figure 13-13. The CK and MCK are generated by HCLK divider, with the prescaler of the MCK determined by I2SDIV and I2SODD. The calculation formula is seen in Figure 13-13.

The prescaler of the CK depends on whether to provide the main clock for peripherals. To ensure that the main clock is always 256 times larger than the audio sampling frequency, the channel bits should be taken into account. When the main clock is needed, the CK should be divided by 8 (I2SCBN=0) or 4 (I2SCBN=1), then divided again by the same prescaler as that of the MCK, that is the final communication clock; When the main clock is not needed, the prescaler of the CK is determined by I2SDIV and I2SODD, shown in Figure 13-13.

Figure 13-22 CK & MCK source in master mode



Apart from the above-mentioned configuration, the following table lists the values of I2SDIV and I2SODD corresponding to some specific frequencies, as well as their respective error for the users to configure the I2SDIV and I2SODD.

Table 13-1 Audio frequency precision using system clock

SysCLK (MHz)	MCLK	Target Fs (Hz)	16bit				32bit			
			I2S DIV	I2S_ODD	RealFs	Error	I2S DIV	I2S_ODD	RealFs	Error
240	NO	192000	19	1	192307	1.2%	10	0	187500	2.34%
240	NO	96000	39	0	96153	0.16%	19	1	96153	0.16%
240	NO	48000	78	0	48076	0.16%	39	0	48076	0.16%
240	NO	44100	85	0	44117	0.04%	42	1	44117	0.04%
240	NO	32000	117	0	32051	0.16%	58	1	32051	0.16%
240	NO	22050	170	0	22058	0.04%	85	0	22058	0.04%
240	NO	16000	234	1	15991	0.05%	117	0	16025	0.16%
240	NO	11025	340	0	11029	0.04%	170	0	11029	0.04%
240	NO	8000	469	0	7995	0.05%	234	1	7995	0.05%
240	YES	192000	2	1	187500	2.34%	2	1	187500	2.34%
240	YES	96000	5	0	93750	2.34%	5	0	93750	2.34%
240	YES	48000	10	0	46875	2.34%	10	0	46875	2.34%
240	YES	44100	10	1	44642	1.23%	10	1	44642	1.23%
240	YES	32000	14	1	32327	1.02%	14	1	32327	1.02%
240	YES	22050	21	1	21802	1.12%	21	1	21802	1.12%
240	YES	16000	29	1	15889	0.68%	29	1	15889	0.68%

240	YES	11025	42	1	11029	0.04%	42	1	11029	0.04%
240	YES	8000	58	1	8012	0.16%	58	1	8012	0.16%
200	No	192000	16	1	189393.9	1.36%	8	0	195312.5	1.73%
200	No	96000	32	1	96153.85	0.16%	16	1	94696.97	1.36%
200	No	48000	65	0	48076.92	0.16%	32	1	48076.92	0.16%
200	No	44100	71	0	44014.08	0.19%	35	1	44014.08	0.19%
200	No	32000	97	1	32051.28	0.16%	49	0	31887.76	0.35%
200	No	22050	141	1	22084.81	0.16%	71	0	22007.04	0.19%
200	No	16000	195	1	15984.65	0.10%	97	1	16025.64	0.16%
200	No	11025	283	1	11022.93	0.02%	141	1	11042.4	0.16%
200	No	8000	390	1	8002.561	0.03%	195	1	7992.327	0.10%
200	Yes	192000	3	0	130208.3	32.18%	3	0	130208.3	32.18%
200	Yes	96000	4	0	97656.25	1.73%	4	0	97656.25	1.73%
200	Yes	48000	8	0	48828.13	1.73%	8	0	48828.13	1.73%
200	Yes	44100	9	0	43402.78	1.58%	9	0	43402.78	1.58%
200	Yes	32000	12	0	32552.08	1.73%	12	0	32552.08	1.73%
200	Yes	22050	17	1	22321.43	1.23%	17	1	22321.43	1.23%
200	Yes	16000	24	1	15943.88	0.35%	24	1	15943.88	0.35%
200	Yes	11025	35	1	11003.52	0.19%	35	1	11003.52	0.19%
200	Yes	8000	49	0	7971.939	0.35%	49	0	7971.939	0.35%
100	No	192000	8	0	195312.5	1.73%	4	0	195312.5	1.73%
100	No	96000	16	1	94696.97	1.36%	8	0	97656.25	1.73%
100	No	48000	32	1	48076.92	0.16%	16	1	47348.48	1.36%
100	No	44100	35	1	44014.08	0.19%	17	1	44642.86	1.23%
100	No	32000	49	0	31887.76	0.35%	24	1	31887.76	0.35%
100	No	22050	71	0	22007.04	0.19%	35	1	22007.04	0.19%
100	No	16000	97	1	16025.64	0.16%	49	0	15943.88	0.35%
100	No	11025	141	1	11042.4	0.16%	71	0	11003.52	0.19%
100	No	8000	195	1	7992.327	0.10%	97	1	8012.821	0.16%
100	Yes	96000	2	0	97656.25	1.73%	2	0	97656.25	1.73%
100	Yes	48000	4	0	48828.13	1.73%	4	0	48828.13	1.73%
100	Yes	44100	4	1	43402.78	1.58%	4	1	43402.78	1.58%
100	Yes	32000	6	0	32552.08	1.73%	6	0	32552.08	1.73%
100	Yes	22050	9	0	21701.39	1.58%	9	0	21701.39	1.58%
100	Yes	16000	12	0	16276.04	1.73%	12	0	16276.04	1.73%
100	Yes	11025	17	1	11160.71	1.23%	17	1	11160.71	1.23%
100	Yes	8000	24	1	7971.939	0.35%	24	1	7971.939	0.35%
72	No	192000	6	0	187500	2.34%	3	0	187500	2.34%
72	No	96000	11	1	97826.09	1.90%	6	0	93750	2.34%
72	No	48000	32	1	34615.38	27.88%	11	1	48913.04	1.90%
72	No	44100	25	1	44117.65	0.04%	13	0	43269.23	1.88%
72	No	32000	35	0	32142.86	0.45%	17	1	32142.86	0.45%
72	No	22050	51	0	22058.82	0.04%	25	1	22058.82	0.04%
72	No	16000	70	1	15957.45	0.27%	35	0	16071.43	0.45%
72	No	11025	102	0	11029.41	0.04%	51	0	11029.41	0.04%
72	No	8000	140	1	8007.117	0.09%	70	1	7978.723	0.27%
72	Yes	96000	2	0	70312.5	26.76%	2	0	70312.5	26.76%
72	Yes	48000	3	0	46875	2.34%	3	0	46875	2.34%
72	Yes	44100	3	0	46875	6.29%	3	0	46875	6.29%
72	Yes	32000	4	1	31250	2.34%	4	1	31250	2.34%
72	Yes	22050	6	1	21634.62	1.88%	6	1	21634.62	1.88%
72	Yes	16000	9	0	15625	2.34%	9	0	15625	2.34%

72	Yes	11025	13	0	10817.31	1.88%	13	0	10817.31	1.88%
72	Yes	8000	17	1	8035.714	0.45%	17	1	8035.714	0.45%

13.3.6 DMA transfer

The SPI supports write and read operations with DMA. Whether used as SPI or I²S, read/write request using DMA comes from the same peripheral. As a result, their configuration procedure are the same, described as follows.

Transmission with DMA

- Select DMA channel: Select a DMA channel for the current SPI from DMA channel map table described in DMA chapter.
- Configure the destination of DMA transfer: Configure the SPI_DT register address as the destination address bit of DMA transfer in the DMA control register. Data will be sent to this address after transmit request is received by DMA.
- Configure the source of DMA transfer: Configure the memory address as the source of DMA transfer in the DMA control register. Data will be loaded into the SPI_DT register from the memory address after transmit request is received by DMA.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure the channel priority of DMA transfer in the DMA control register.
- Configure DMA interrupt generation after half or full transfer in the DMA control register.
- Enable DMA transfer channel in the DMA control register.

Reception with DMA

- Select DMA transfer channel: Select a DMA channel for the current SPI from DMA channel map table described in DMA chapter.
- Configure the destination of DMA transfer: Configure the memory address as the destination of DMA transfer in the DMA control register. Data will be loaded from the SPI_DT register to the programmed destination after reception request is received by DMA.
- Configure the source of DMA transfer: Configure the SPI_DT register address as the source of DMA transfer in the DMA control register. Data will be loaded from the SPI_DT register to the programmed destination after reception request is received by DMA.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure DMA interrupt generation after half or full transfer in the DMA control register
- Enable DMA transfer channel in the DMA control register.

13.3.7 Transmitter/Receiver

Whether being used as SPI or I2S, there is no difference for CPU. The SPI (in whatever mode) shares the same base address, the same SPI_DT register, the same transmitter and receiver. The SPI transmitter and receiver is responsible for sending and receiving the desired data frame according to the configuration of the communication controller. Thus their status flags such as TDBE, RDBF and ROERR, and their interrupt enable bits including TDBEIE, RDBFIE and ERRIE are identical.

Special attention must be paid to:

- CRC check is not available on the I²S. Any operation linked to CRC, including CCERR flag and the corresponding interrupts, is not supported.
- I²S protocol needs decode the current channel status. The ACS bit is used to judge whether the current transfer occurs on the left channel (ACS=0) or the right channel (ACS=1).
- TUERR bit indicates whether an underrun occurs. TUERR=1 means an underrun error occurs on the transmitter. An interrupt is generated when the ERRIE is set.
- Read/write operation to the SPI_DT register is different under different audio protocols, data bits and channel bits. Refer to the audio protocol selector section for more information.
- Pay more attention to the I²S disable operation under different configurations, shown as follows:
  - I2SDBN=00, I2SCBN=1, STDSLE=10: wait for the second-to-last RDBF=1 and 17 CK periods

before disabling the I²S.

— I2SDBN=00, I2SCBN=1, STDSLE=00 or STDSLE=01 or STDSLE=11: wait for the last RDBF=1 and one CK period before the I²S.

— I2SDBN, I2SCBN,STDSLE combination: wait for the second-to-last RDBF=1 and one CK period before disabling the I²S.

I²S transmitter configuration procedure:

- Configure operation mode selector
- Configure audio protocol selector
- Configure I2S_SCK controller
- Configure DMA transfer (if necessary)
- Set the I2SEN bit to enable I²S
- Follow above steps to configure the I²SxEXT (For I²S full-duplex mode)

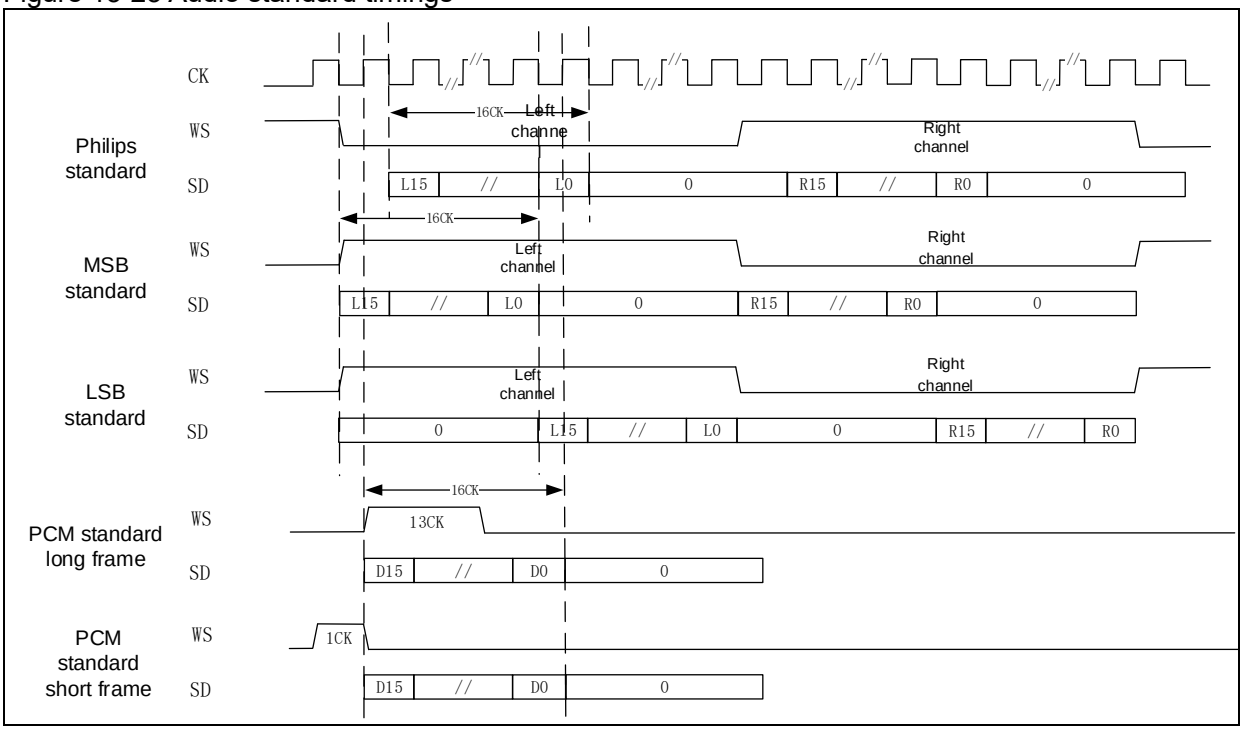
I²S receiver configuration procedure:

- Configure operation mode selector
- Configure audio protocol selector
- Configure I2S_SCK controller
- Configure DMA transfer (if necessary)
- Set the I2SEN bit to enable I²S
- Follow above steps to configure the I²SxEXT (For I²S full-duplex mode)

13.3.8 I²S communication timings

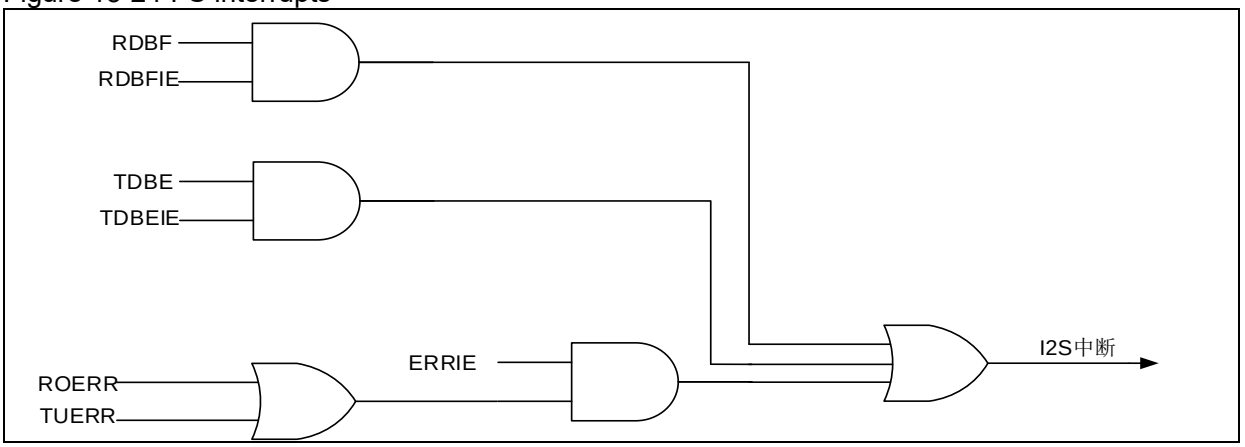
I2S can address four different audio standards: Philips standard, the most significant byte (left-aligned) and the least significant byte (right-aligned) standards, and the PCM standard. Figure 13-23 shows their respective timings.

Figure 13-23 Audio standard timings



13.3.9 Interrupts

Figure 13-24 I²S interrupts



13.3.10 IO pin control

The I²S needs three pins for transfer operation, namely, the SD, WS and CK. The MCLK pin is also required if need to provide main clock for peripherals. The I²S shares some pins with the SPI, described as follows:

- SD: Serial data (mapped on the MOSI pin) for bidirectional data transmission and reception.
- WS: Word select (mapped on the CS pin) for data control signal output in master mode, and input in slave mode.
- CK: Communication clock (mapped on the SCK pin) as clock signal output in master mode, and input in slave mode.
- MCLK: Master clock (mapped independently) is used to provide main clock for peripherals. The frequency of output clock signal is set to 256x Fs (audio sampling frequency)

13.4 SPI registers

These peripheral registers must be accessed by half-word (16 bits) or word (32 bits).

Table 13-2 SPI register map and reset value

Register	Offset	Reset value
SPI_CTRL1	0x00	0x0000
SPI_CTRL2	0x04	0x0000
SPI_STS	0x08	0x0002
SPI_DT	0x0C	0x0000
SPI_CPOLY	0x10	0x0007
SPI_RCRC	0x14	0x0000
SPI_TCRC	0x18	0x0000
SPI_I2SCTRL	0x1C	0x0000
SPI_I2SCLKP	0x20	0x0002

13.4.1 SPI control register1 (SPI_CTRL1) (Not used in I²S mode)

Bit	Register	Reset value	Type	Description
Bit 15	SLBEN	0x0	rw	Single line bidirectional half-duplex enable 0: Disabled 1: Enabled
Bit 14	SLBTD	0x0	rw	Single line bidirectional half-duplex transmission direction This bit and the SLBEN bit together determine the data output direction in “Single line bidirectional half-duplex” mode. 0: Receive-only mode 1: Transmit-only mode
Bit 13	CCEN	0x0	rw	RC calculation enable 0: Disabled 1: Enabled
Bit 12	NTC	0x0	rw	Transmit CRC next When this bit is set, it indicates that the next data transferred is CRC value. 0: Next transmitted data is the normal value 1: Next transmitted data is CRC value
Bit 11	FBN	0x0	rw	Frame bit num This bit is used to configure the number of data frame bit for transmission/reception. 0: 8-bit data frame 1: 16-bit data frame
Bit 10	ORA	0x0	rw	Receive-only active In two-wire unidirectional mode, when this bit is set, it indicates that Receive-only is active, but the transmit is not allowed. 0: Transmission and reception 1: Receive-only mode
Bit 9	SWCSEN	0x0	rw	Software CS enable When this bit is set, the CS pin level is determined by the SWCSIL bit. The status of I/O level on the CK pin is invalid. 0: Disabled 1: Enabled
Bit 8	SWCSIL	0x0	rw	Software CS internal level This bit is valid only when the SWCSEN is set. It determines the level on the CS pin. In master mode, this bit must be set. 0: Low level 1: High level

Bit 7	LTF	0x0	rw	LSB transmit first This bit is used to select for MST transfer first or LSB transfer first. 0: MSB 1: LSB
Bit 6	SPIEN	0x0	rw	SPI enable 0: Disabled 1: Enabled
Bit 5: 3	MDIV	0x0	rw	Master clock frequency division In master mode, the peripheral clock divided by the prescaler is used as SPI clock. The MDIV[3] bit is in the SPI_CTRL2 register, MDIV[3: 0]: 0000: Divided by 2 0001: Divided by 4 0010: Divided by 8 0011: Divided by 16 0100: Divided by 32 0101: Divided by 64 0110: Divided by 128 0111: Divided by 256 1000: Divided by 512 1001: Divided by 1024
Bit 2	MSTEN	0x0	rw	Master enable 0: Disabled (Slave) 1: Enabled (Master)
Bit 1	CLKPOL	0x0	rw	Clock polarity Indicates the polarity of clock output in idle state. 0: Low level 1: High level
Bit 0	CLKPHA	0x0	rw	Clock phase 0: Data capture starts from the first clock edge 1: Data capture starts from the second clock edge

Note: The SPI_CTRL1 register must be 0 in I^2S mode.

13.4.2 SPI control register2 (SPI_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Forced 0 by hardware.
Bit 9	MDIV3EN	0x0	rw	Master clock frequency divided by 3 enable 0: Disabled 1: Enabled Note: When this bit is set, the MDIV[3: 0] becomes invalid, and the SPI clock is forced to be PCLK/3.
Bit 8	MDIV[3]	0x0	rw	Master clock frequency division Refer to the MDIV[2: 0] of the SPI_CTRL1 register.
Bit 7	TDBEIE	0x0	rw	Transmit data buffer empty interrupt enable 0: Disabled 1: Enabled
Bit 6	RDBFIE	0x0	rw	Receive data buffer full interrupt enable 0: Disabled 1: Enabled
Bit 5	ERRIE	0x0	rw	Error interrupt enable This bit controls interrupt generation when errors occur (CCERR, MMERR, ROERR, TUERR and CSPAS) 0: Disabled 1: Enabled
Bit 4	TIEN	0x0	rw	TI mode enable 0: TI mode disabled (Motorola mode) 1: TI mode enabled (TI mode) Note: This mode is not used in I2S mode. It must be 0 in

				I2S mode.
Bit 3	Reserved	0x0	resd	Kept at its default value
Bit 2	HWCSOE	0x0	rw	Hardware CS output enable This bit is valid only in master mode. When this bit is set, the I/O output on the CS pin is low; when this bit is 0, the I/O input on the CS pin must be set high. 0: Disabled 1: Enabled
Bit 1	DMATEN	0x0	rw	DMA transmit enable 0: Disabled 1: Enabled
Bit 0	DMAREN	0x0	rw	DMA receive enable 0: Disabled 1: Enabled

13.4.3 SPI status register (SPI_STS)

Bit	Register	Reset value	Type	Description
Bit 15: 9	Reserved	0x00	resd	Forced 0 by hardware
Bit 8	CSPAS	0x0	ro	CS pulse abnormal setting flag 0: CS pulse flag normal 1: CS pulse flag is set abnormally Note: This bit is used for TI slave mode. It is cleared by reading the STS register.
Bit 7	BF	0x0	ro	Busy flag 0: SPI is not busy. 1: SPI is busy.
Bit 6	ROERR	0x0	ro	Receiver overflow error 0: No overflow error 1: Overflow error occurs.
Bit 5	MMERR	0x0	ro	Master mode error This bit is set by hardware and cleared by software (read/write access to the SPI_STS register, followed by write operation to the SPI_CTRL1 register) 0: No mode error 1: Mode error occurs.
Bit 4	CCERR	0x0	rw0c	CRC error Set by hardware, and cleared by software. 0: No CRC error 1: CRC error occurs.
Bit 3	TUERR	0x0	ro	Transmitter underload error Set by hardware, and cleared by software (read the SPI_STS register). 0: No underload error 1: Underload error occurs. Note: This bit is only used in I^2S mode.
Bit 2	ACS	0x0	ro	Audio channel state This bit indicates the status of the current audio channel. 0: Left channel 1: Right channel Note: This bit is only used in I^2S mode.
Bit 1	TDBE	0x1	ro	Transmit data buffer empty 0: Transmit data buffer is not empty. 1: Transmit data buffer is not empty.
Bit 0	RDBF	0x0	ro	Receive data buffer full

				0: Transmit data buffer is not full. 1: Transmit data buffer is full.

13.4.4 SPI data register (SPI_DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DT	0x0000	rw	Data value This register controls read and write operations. When the data bit is set as 8 bit, only the 8-bit LSB [7: 0] is valid.

13.4.5 SPICRC register (SPI_CPOLY) (Not used in I²S mode)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CPOLY	0x0007	rw	CRC polynomial This register contains the polynomial used for CRC calculation. Note: This register is valid only in SPI mode.

13.4.6 SPIRxCRC register (SPI_RCRC) (Not used in I²S mode)

Bit	Register	Reset value	Type	Description
Bit 15: 0	RCRC	0x0000	ro	Receive CRC When CRC calculation is enabled, this register contains the CRC value computed based on the received data. This register is reset when the CCEN bit in the SPI_CTRL1 register is cleared. When the data frame format is set to 8-bit data, only the 8-bit LSB ([7: 0]) are calculated based on CRC8 standard; when 16-bit data bit is selected, follow CRC16 standard. Note: This register is only used in SPI mode.

13.4.7 SPITxCRC register (SPI_TCRC)

Bit	Register	Reset value	Type	Description
Bit 15: 0	TCRC	0x0000	ro	Transmit CRC When CRC calculation is enabled, this register contains the CRC value computed based on the transmitted data. This register is reset when the CCEN bit in the SPI_CTRL1 register is cleared. When the data frame format is set to 8-bit data, only the 8-bit LSB ([7: 0]) are calculated based on CRC8 standard; when 16-bit data bit is selected, follow CRC16 standard. Note: This register is only used in SPI mode.

13.4.8 SPI_I2S register (SPI_I2SCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 12	Reserved	0x0	resd	Forced 0 by hardware.
Bit 11	I2SMSEL	0x0	rw	I²S mode select 0: SPI mode 1: I²S mode
Bit 10	I2SEN	0x0	rw	I²S enable 0: Disabled 1: Enabled
Bit 9: 8	OPERSEL	0x0	rw	I²S operation mode select 00: Slave transmission 01: Slave reception 10: Master transmission 11: Master reception
Bit 7	PCMFSSEL	0x0	rw	PCM frame synchronization This bit is valid only when the PCM standard is used. 0: Short frame synchronization

				1: Long frame synchronization
Bit 6	Reserved	0x0	resd	Kept at its default value
Bit 5: 4	STDSEL	0x0	rw	I²S standard select 00: Philips standard 01: MSB-aligned standard (left-aligned) 10: LSB-aligned standard (right-aligned) 11: PCM standard
Bit 3	I2SCLKPOL	0x0	rw	I²S clock polarity This bit indicates the clock polarity on the clock pin in idle state. 0: Low 1: High
Bit 2: 1	I2SDBN	0x0	rw	I²S data bit num 00: 16-bit data length 01: 24-bit data length 10: 32-bit data length 11: Not allowed.
Bit 0	I2SCBN	0x0	rw	I²S channel bit num This bit can be configured only when the I²S is set to 16-bit data; otherwise, it is fixed to 32-bit by hardware. 0: 16-bit wide 1: 32-bit wide

13.4.9 SPI_I2S prescaler register (SPI_I2SCLKP)

Bit	Register	Reset value	Type	Description
Bit 15: 12	Reserved	0x0	resd	Forced 0 by hardware.
Bit 9	I2SMCLKOE	0x0	rw	I²S Master clock output enable 0: Disabled 1: Enabled
Bit 8	I2SODD	0x0	rw	Odd factor for I²S division 0: Actual divider factor =I2SDIV*2 1: Actual divider factor =(I2SDIV*2)+1
Bit 11: 10 Bit 7: 0	I2SDIV	0x02	rw	I²S division It is not allowed to configure I2SDIV[9: 0]=0 or I2SDIV[9: 0]=1

14 Timer

AT32F435 timers include basic timers, general-purpose timers, and advanced timers.

Please refer to Section 14.1 ~ Section 14.4 for the detailed function modes. All functions of different timers are shown in the following tables.

Table 14-1 TMR functional comparison

Timer type	Timer	Counter bit	Count mode	Repetition	Prescaler	DMA requests	Capture/compare channel	PWM input mode	EXT input	Break input
Advanced-control timer	TMR1 TMR8 TMR20	16	Up Down Up/Down	16-bit	1~65535	O	4	O	O	O
General-purpose timer	TMR2 TMR5	16/32	Up Down Up/Down	X	1~65535	O	4	O	O	X
	TMR3 TMR4	16	Up Down Up/Down	X	1~65535	O	4	O	O	X
	TMR9 TMR12	16	Up	X	1~65535	X	2	O	X	X
	TMR10 TMR11 TMR13 TMR14	16	Up	X	1~65535	X	1	X	X	X
Basic timer	TMR6 TMR7	16	Up	X	1~65535	O	X	X	X	X

Timer type	Timer	Counter bit	Count mode	PWM output	Single pulse output	Complementary output	Dead-time	Encoder interface connection	Interfacing with hall sensors	Linkage peripheral
Advanced-control timer	TMR1 TMR8 TMR20	16	Up Down Up/Down	O	O	O	O	O	O	Timer synchronization ADC/DAC
General-purpose timer	TMR2 TMR5	16/32	Up Down Up/Down	O	O	X	X	O	O	Timer synchronization ADC/DAC
	TMR3 TMR4	16	Up Down Up/Down	O	O	X	X	O	O	Timer synchronization
	TMR9 TMR12	16	Up	O	O	X	X	X	X	Timer synchronization
	TMR10 TMR11 TMR13 TMR14	16	Up	O	O	X	X	X	X	NA
Basic timer	TMR6 TMR7	16	Up	X	X	X	X	X	X	DAC

14.1 Basic timer (TMR6 and TMR7)

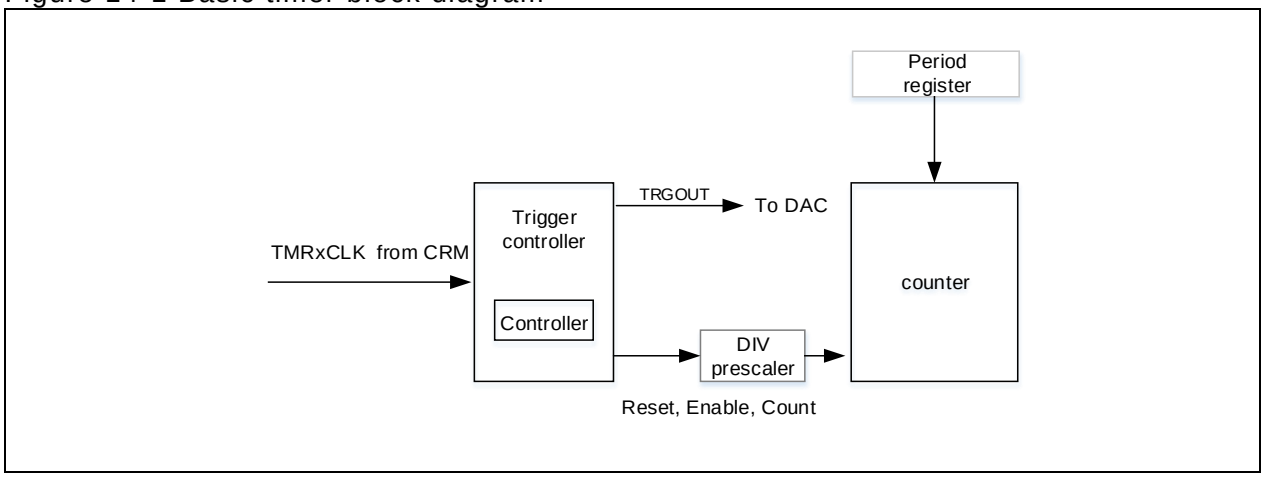
14.1.1 TMR6 and TMR7 introduction

Each of the basic timers (TMR6 and TMR7) includes a 16-bit up counter and the corresponding control logic. without being connected to external I/Os, they can be used for a basic timing and providing clocks for the digital-to-analog converter (DAC).

14.1.2 TMR6 and TMR7 main features

- 16-bit up counter, auto reload
- 16-bit prescaler used to divide the TMR_CLK frequency by any factor between 1 and 65536
- Synchronization circuit to trigger DAC (Unique characteristics for TMR6 and TMR7)

Figure 14-1 Basic timer block diagram

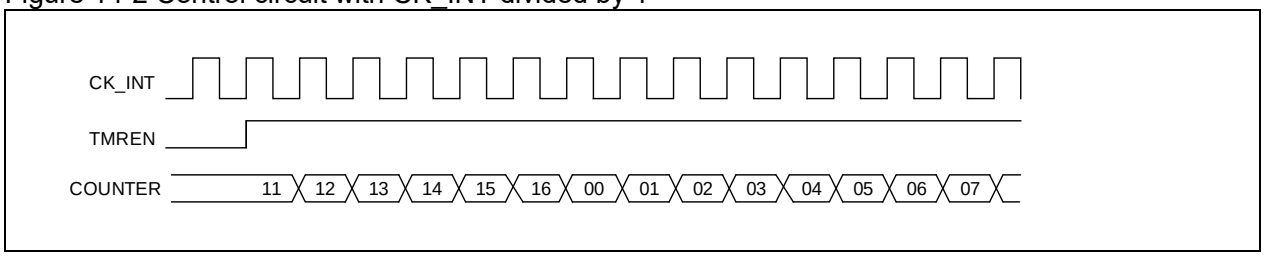


14.1.3 TMR6 and TMR7 function overview

14.1.3.1 Count clock

The counter clock of TMR6 and TMR7 is provided by the internal clock source (CK_INT) divided by prescaler.

Figure 14-2 Control circuit with CK_INT divided by 1



14.1.3.2 Counting mode

The basic timer only supports upcounting mode, and it has an internal 16-bit counter.

The TMRx_PR register is used to set the counting period. The value in the TMRx_PR is immediately moved to the shadow register by default. When the periodic buffer is enabled (PRBEN=1), the value in the TMRx_PR register is transferred to the shadow register only at an overflow event.

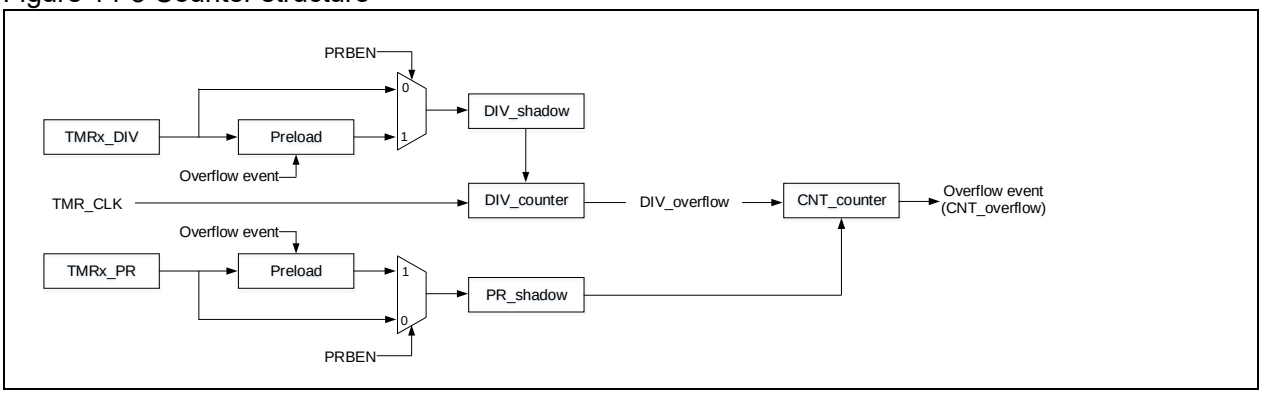
The TMRx_DIV register is used to set the counting frequency. The counter counts once every count clock period (DIV[15:0]+1). Similar to the TMRx_PR register, when the periodic buffer is enabled, the value in the TMRx_DIV register is updated to the shadow register at an overflow event.

Reading the TMRx_CNT register returns to the current counter value, and writing to the TMRx_CNT register updates the current counter value to the value being written.

An overflow event is generated by default. Set OVFEN=1 in the TMRx_CTRL1 to disable generation of update events. The OVFS bit in the TMRx_CTRL1 register is used to select overflow event source. By default, counter overflow/underflow, setting OVFSWTR bit and the reset signal generated by the slave timer controller in reset mode trigger the generation of an overflow event. When the OVFS bit is set, only

counter overflow/underflow triggers an overflow event.
Setting the TMREN bit (TMREN=1) enables the timer to start counting. Base on synchronization logic, however, the actual enable signal TMR_EN is set 1 clock cycle after the TMREN is set.

Figure 14-3 Counter structure



Upcounting mode

Set CMSEL[1:0]=2'b00 and OWCDIR=1'b0 in the TMRx_CTRL1 register to enable upcounting mode. In this mode, the counter counts from 0 to the value programmed in the TMRx_PR register, restarts from 0, and generates a counter overflow event, with the OVFIF bit being set to 1. If the overflow event is disabled, the counter is no longer reloaded with the preload value and period value at a counter overflow event; otherwise, the counter is updated with the preload value and period value at an overflow event.

Figure 14-4 Overflow event when PRBEN=0

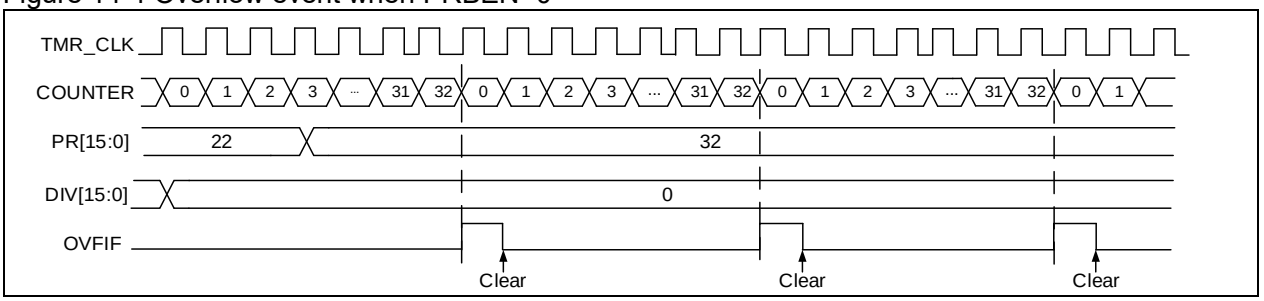


Figure 14-5 Overflow event when PRBEN=1

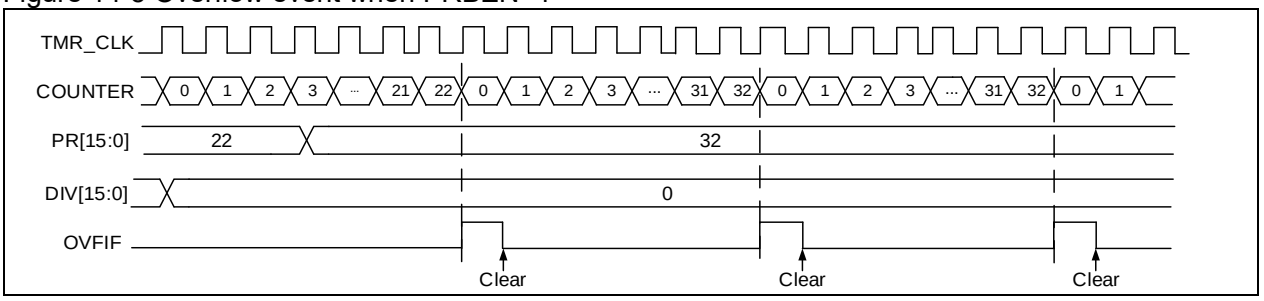
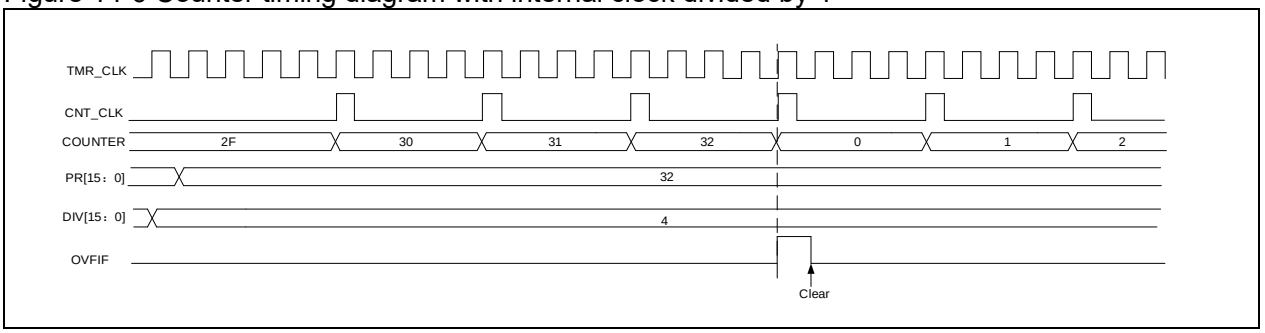


Figure 14-6 Counter timing diagram with internal clock divided by 4



14.1.3.3 Debug mode

When the microcontroller enters debug mode (Cortex ®-M4F core halted), the TMRx counter stops counting when the TMRx_PAUSE bit is set.

14.1.4 TMR6 and TMR7 registers

These peripheral registers must be accessed by word (32 bits).

In Table 14-2, all the TMRx registers are mapped to a 16-bit addressable space.

Table 14-2 TMR6 and TMR7— register table and reset value

Register	Offset	Reset value
TMRx_CTRL1	0x00	0x0000
TMRx_CTRL2	0x04	0x0000
TMRx_IDEN	0x0C	0x0000
TMRx_ISTS	0x10	0x0000
TMRx_SWEVT	0x14	0x0000
TMRx_CVAL	0x24	0x0000
TMRx_DIV	0x28	0x0000
TMRx_PR	0x2C	0x0000

14.1.4.1 TMR6 and TMR7 control register1 (TMRx_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7	PRBEN	0x0	rw	Period buffer enable 0: Period buffer is disabled. 1: Period buffer is enabled.
Bit 6: 4	Reserved	0x0	resd	Kept at its default value.
Bit 3	OCMEN	0x0	rw	One cycle mode enable This bit is used to select whether to stop the counter at overflow event. 0: Disabled 1: Enabled
Bit 2	OVFS	0x0	rw	Overflow event source This bit is used to configure overflow event or DMA request sources. 0: Counter overflow, setting the OVFSWTR bit or overflow event generated from the slave controller 1: Only counter overflow generates an overflow event.
Bit 1	OVFEN	0x0	rw	Overflow event enable This bit is used to enable or disable OEV event generation. 0: OEV event is enabled. An overflow event is generated by any of the following events: - Counter overflow - Setting the OVFSWTR bit - Overflow event generated from the slave controller 1: OEV event is disabled. If the OVFSWTR bit is set, or a hardware reset is generated from the slave controller, the counter and the prescaler are reinitialized. Note: This bit is set and cleared by software.
Bit 0	TMREN	0x0	rw	TMR enable 0: Disabled 1: Enabled

14.1.4.2 TMR6 and TMR7 control register2 (TMRx_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 15: 7	Reserved	0x000	resd	Kept at its default value.
Bit 6: 4	PTOS	0x0	rw	Master TMR output selection This field is used to select the signals in master mode to be sent to slave timers. 000: Reset 001: Enable 010: Update
Bit 3: 0	Reserved	0x0	resd	Kept at its default value.

14.1.4.3 TMR6 and TMR7 DMA/interrupt enable register (TMRx_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15: 9	Reserved	0x00	resd	Kept at its default value.
Bit 8	OVFDEN	0x0	rw	Overflow event DMA request enable 0: Disabled 1: Enabled
Bit 7: 1	Reserved	0x00	resd	Kept at its default value.
Bit 0	OVFIEN	0x0	rw	Overflow interrupt enable 0: Disabled 1: Enabled

14.1.4.4 TMR6 and TMR7 interrupt status register (TMRx_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 1	Reserved	0x0000	resd	Kept at its default value.
Bit 0	OVFIF	0x0	rw0c	Overflow interrupt flag This bit is set by hardware at an overflow event. It is cleared by software. 0: No overflow event occurs. 1: Overflow event occurs, and OVFEN=0, and OVFS=0 in the TMRx_CTRL1 register: − An overflow event occurs when OVFG=1 in the TMRx_SWEVE register − An overflow event occurs when the counter value (CVAL) is reinitialized by a trigger event.

14.1.4.5 TMR6 and TMR7 software event register (TMRx_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 1	Reserved	0x0000	resd	Kept at its default value.
Bit 0	OVFSWTR	0x0	rw0c	Overflow event triggered by software An overflow event is trigged by software. 0: No effect 1: Generate an overflow event by software write operation.

14.1.4.6 TMR6 and TMR7 counter value (TMRx_CVAL)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0000	rw	Counter value

14.1.4.7 TMR6 and TMR7 division (TMRx_DIV)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DIV	0x0000	rw	Divider value The counter clock frequency $f_{CK_CNT} = f_{TMR_CLK}$ / (DIV[15: 0]+1). At each overflow event, DIV value is sent to the DIV register.

14.1.4.8 TMR6 and TMR7 period register (TMRx_PR)

Bit	Register	Reset value	Type	Description
Bit 15: 0	PR	0x0000	rw	Period value This indicates the period value of the TMRx counter. The timer stops working when the period value is 0.

14.2 General-purpose timer (TMR2 to TMR5)

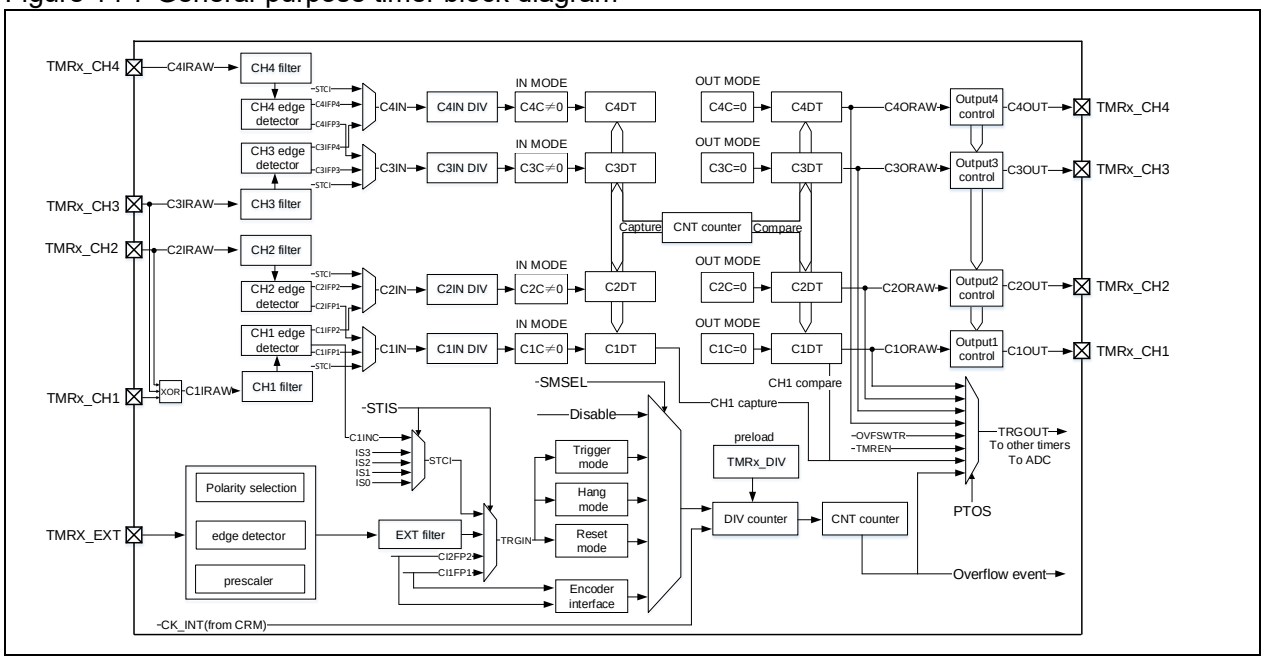
14.2.1 TMR2 to TMR5 introduction

The general-purpose timer (TMR2 to TMR5) consists of a 16-bit counter supporting up, down, up/down (bidirectional) counting modes, four capture/compare registers, and four independent channels to achieve input capture and programmable PWM output.

14.2.2 TMR2 to TMR5 main features

- Source of count clock is selectable : internal clock, external clock and internal trigger
- 16-bit up, down, up/down and encoder mode counter (TMR2/5 can be extended to 32-bit)
- 4 independent channels for input capture, output compare, PWM generation and one-pulse mode output
- Synchronization control between master and slave timers
- Interrupt/DMA is generated at overflow event, trigger event and channel event
- Support TMR burst DMA transfer

Figure 14-7 General-purpose timer block diagram

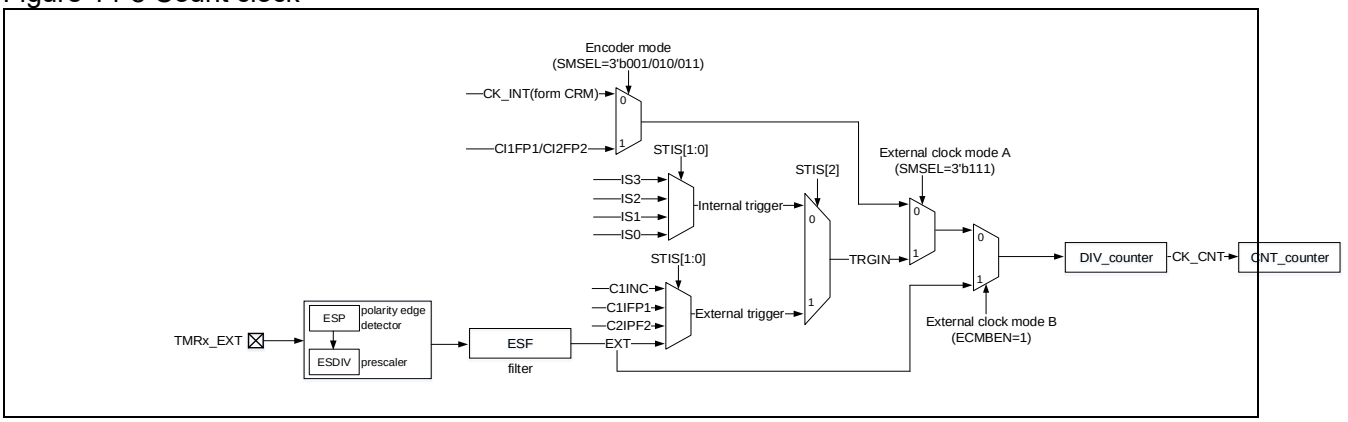


14.2.3 TMR2 to TMR5 functional overview

14.2.3.1 Count clock

The count clock of TMR2~TMR5 can be provided by the internal clock (CK_INT), external clock (external clock mode A and B) and internal trigger input (ISx).

Figure 14-8 Count clock

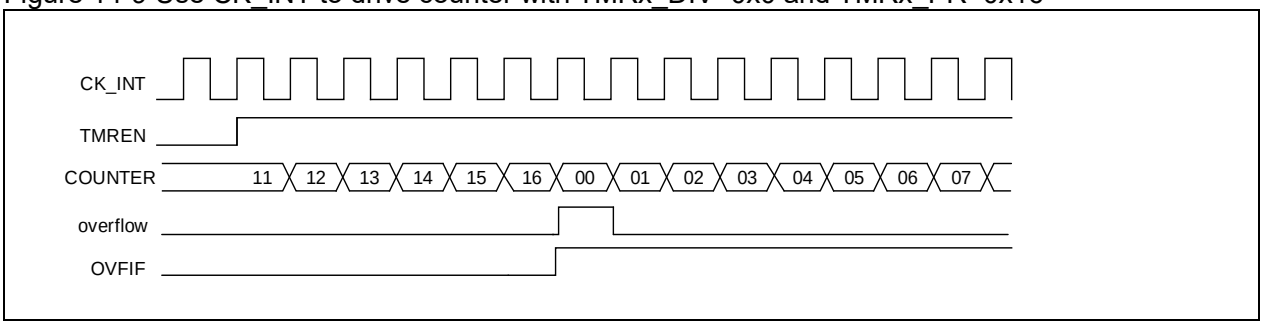


Internal clock (CK_INT)

By default, the CK_INT divided by a prescaler is used to drive the counter to start counting. The configuration process is as follows:

- Set the CLKDIV[1:0] bit in the TMRx_CTRL1 register to set the CK_INT frequency;
- Set the TWCMSEL[1:0] bit in the TMRx_CTRL1 register to select count mode. If the one-way count direction is set, configure OWCDIR bit in the TMRx_CTRL1 register to select the specific direction;
- Set the TMRx_DIV register to set the counting frequency;
- Set the TMRx_PR register to set the counting period;
- Set the TMREN bit in the TMRx_CTRL1 register to enable the counter.

Figure 14-9 Use CK_INT to drive counter with TMRx_DIV=0x0 and TMRx_PR=0x16



External clock (TRGIN/EXT)

The counter clock can be provided by two external clock sources, namely, TRGIN and EXT signals.

SMSEL=3'b111: External clock mode A is selected. Select an external clock source TRGIN signal by setting the STIS[2:0] bit to drive the counter to start counting. The external clock sources include: C1INC (STIS=3'b100, channel 1 rising edge and falling edge), C1IFP1 (STIS=3'b101, channel 1 signal with filtering and polarity selection), C2IFP2 (STIS=3'b110, channel 2 signal with filtering and polarity selection) and EXT (STIS=3'b111, external input signal with polarity selection, frequency division and filtering).

ECMBEN=1: External clock mode B is selected. The counter is driven by external input that has gone through polarity selection, frequency division and filtering. The external clock mode B is equivalent to the external clock mode A which selects EXT signal as an external force TRGIN.

To use external clock mode A, follow the steps below:

- Set external source TRGIN parameters

  If the TMRx_CH1 is used as a source of TRGIN, it is necessary to configure channel 1 input filter

(C1DF[3:0] in TMRx_CM1 register) and channel 1 input polarity (C1P/C1CP in TMRx_CCTRL register);

If the TMRx_CH2 is used as source of TRGIN, it is necessary to configure channel 1 input filter (C2DF[3:0] in TMRx_CM1 register) and channel 2 input polarity (C2P/C2CP in TMRx_CCTR register);

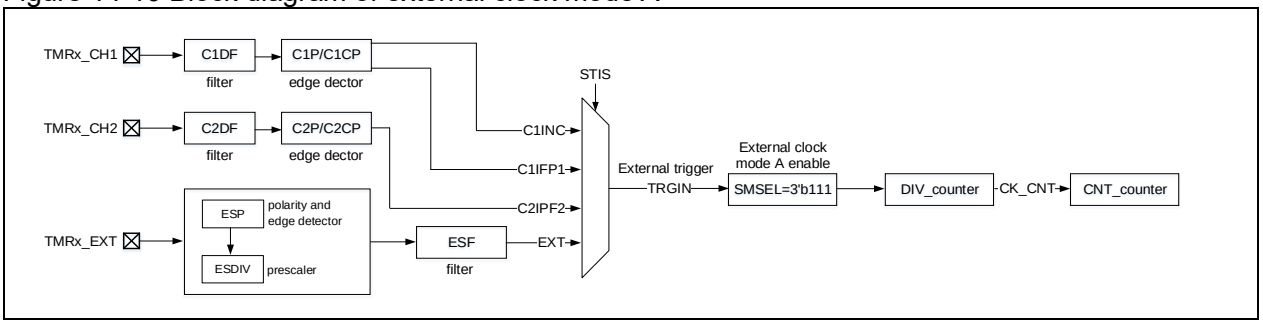
If the TMRx_EXT is used as a source of TRGIN, it is necessary to configure the external signal polarity (ESP in TMRx_STCTRL register), external signal frequency division (ESDIV[1:0] in TMRx_STCTRL) and external signal filter (ESF[3:0] in TMRx_STCTRL register).

- Set TRGIN signal source using the STIS[1:0] bit in TMRx_STCTRL register
- Enable external clock mode A by setting SMSEL=3'b111 in TMRx_STCTR register
- Set counting frequency through the DIV[15:0] in TMRx_DIV register
- Set counting period through the PR[15:0] in TMRx_PR register
- Enable counter through the TMREN bit in TMRx_CTRL1 register

To use external clock mode B, follow the steps below:

- Set external signal polarity through the ESP bit in TMRx_STCTRL register
- Set external signal frequency division through the ESDIV[1:0] bit in TMRx_STCTRL register
- Set external signal filter through the ESF[3:0] bit in TMRx_STCTRL register
- Enable external clock mode B through the ECMBEN bit in TMRx_STCTR register
- Set counting frequency through the DIV[15:0] bit in TMRx_DIV register
- Set counting period through the PR[15:0] bit in TMRx_PR register
- Enable counter through the TMREN in TMRx_CTRL1 register

Figure 14-10 Block diagram of external clock mode A



Note: The delay between the signal on the input side and the actual clock of the counter is due to the synchronization circuit.

Figure 14-11 Counting in external clock mode A, with PR=0x32 and DIV=0x0

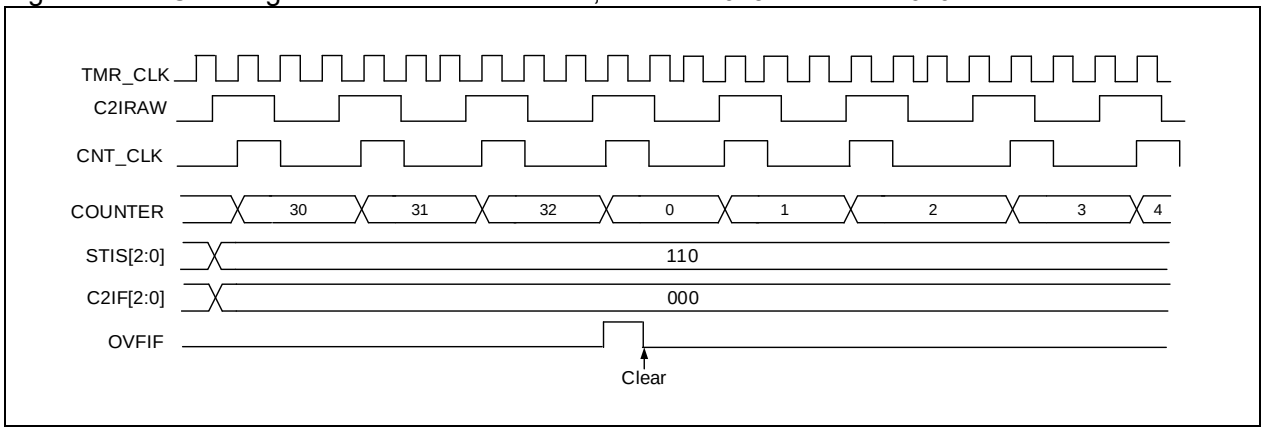
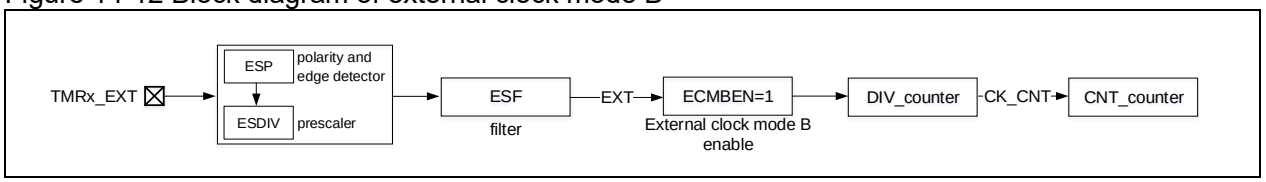
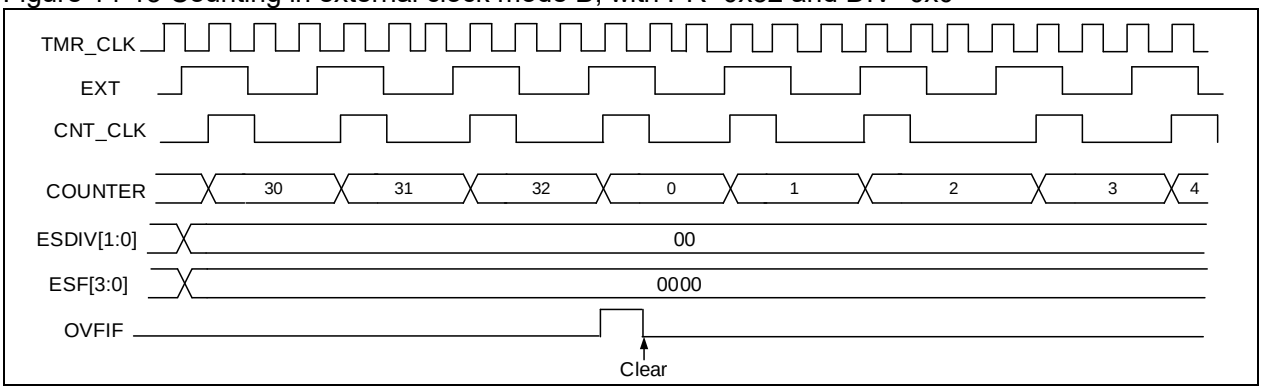


Figure 14-12 Block diagram of external clock mode B



Note: The delay between the EXT signal on the input side and the actual clock of the counter is due to the synchronization circuit.

Figure 14-13 Counting in external clock mode B, with PR=0x32 and DIV=0x0



Internal trigger input (ISx)

Timer synchronization allows interconnection between several timers. The TMR_CLK of one timer can be provided by the TRGOUT signal output by another timer. Set the STIS[2: 0] bit to select internal trigger signal to enable counting.

Each timer (TMR2 to TMR5) consists of a 16-bit prescaler, which is used to generate the CK_CNT that enables the counter to count. The frequency division relationship between the CK_CNT and TMR_CLK can be adjusted by setting the value of the TMRx_DIV register. The prescaler value can be modified at any time, but it takes effect only when the next overflow event occurs.

The internal trigger input is configured as follows:

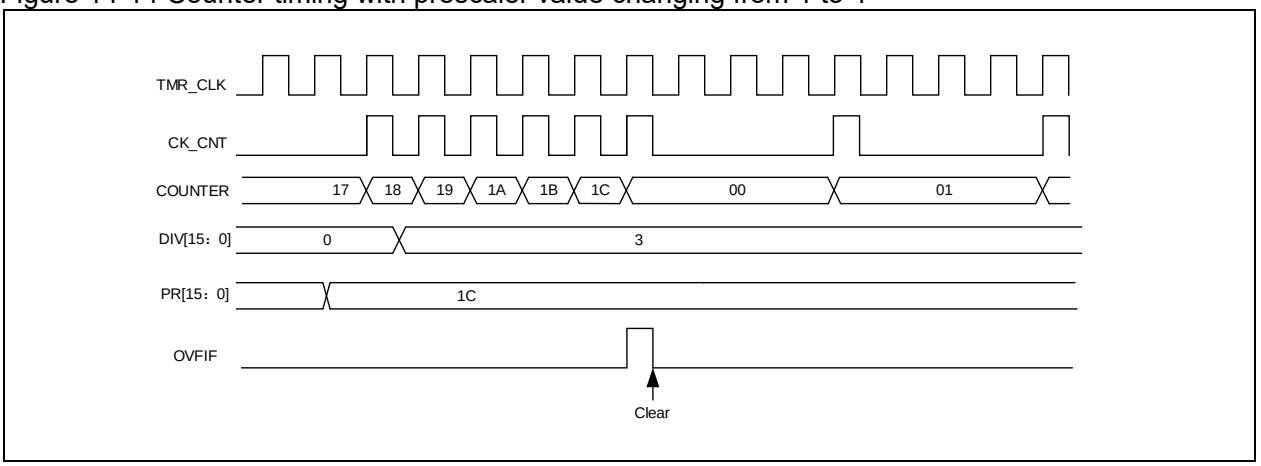
- Set the TMRx_PR register to set the counting period;
- Set the TMRx_DIV register to set the counting frequency;
- Set the TWCMSEL[1:0] bit in the TMRx_CTRL1 register to set the count mode;
- Set the STIS[2:0] bit (range: 3'b000~3'b011) in the TMRx_STCTRL register and select internal trigger;
- Set SMSEL[2:0]=3'b111 in the TMRx_STCTRL register and select external clock mode A;
- Set the TMREN bit in the TMRx_CTRL1 register to enable TMRx counter.

Table 14-3 TMRx internal trigger connection

Slave controller	IS0 (STIS = 000)	IS1 (STIS = 001)	IS2 (STIS = 010)	IS3 (STIS = 011)
TMR2	TMR1	TMR8/USB_SOF(2)	TMR3	TMR4
TMR3	TMR1	TMR2	TMR5	TMR4
TMR4	TMR1	TMR2	TMR3	TMR8
TMR5	TMR2	TMR3	TMR4	TMR8

Note 1: If there is no corresponding timer in a device, the corresponding trigger signal ISx is not present.

Figure 14-14 Counter timing with prescaler value changing from 1 to 4



14.2.3.2 Counting mode

The timer (TMR2 to TMR5) supports several counting modes to meet different application scenarios. Each timer has an internal 16-bit upcounter, downcounter, upcounter/downcounter. TMR2/5 can be extended to 32-bit by setting the PMEN bit to 1.

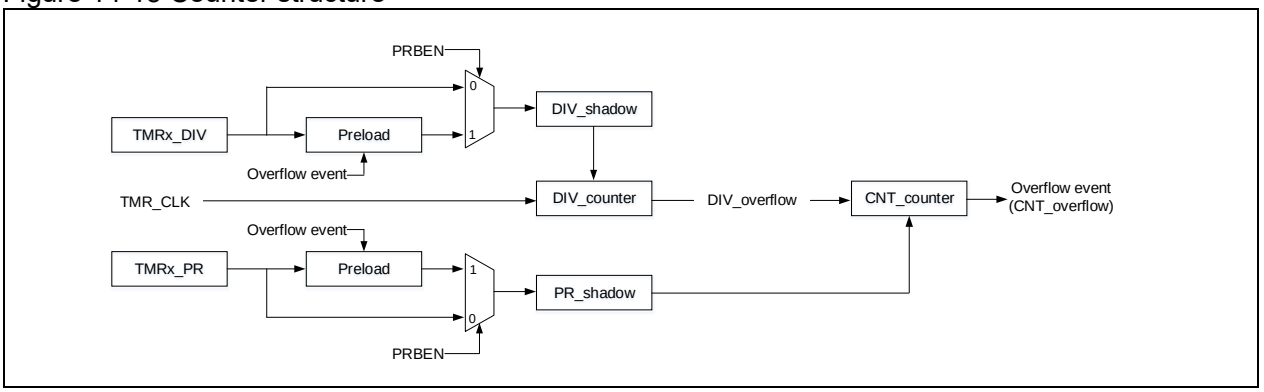
The TMRx_PR register is used to set the counting period. The value in the TMRx_PR is immediately moved to the shadow register by default. When the periodic buffer is enabled (PRBEN=1), the value in the TMRx_PR register is transferred to the shadow register only at an overflow event. The OVFEN and OVFS bits are used to configure the overflow event.

The TMRx_DIV register is used to configure the counting frequency. The counter counts once every count clock period (DIV[15:0]+1). Similar to TMRx_PR register, when the periodic buffer is enabled, the value in the TMRx_DIV register is updated to the shadow register at an overflow event.

An overflow event is generated by default. Set OVFEN=1 in the TMRx_CTRL1 to disable generation of update events. The OVFS bit in the TMRx_CTRL1 register is used to select overflow event source. By default, counter overflow/underflow, setting OVFSWTR bit and the reset signal generated by the slave timer controller in reset mode trigger the generation of an overflow event. When the OVFS bit is set, only counter overflow/underflow triggers an overflow event.

Setting the TMREN bit (TMREN=1) enables the timer to start counting. Base on synchronization logic, however, the actual enable signal TMR_EN is set 1 clock cycle after the TMREN is set.

Figure 14-15 Counter structure



Upcounting mode

Set CMSEL[1:0]=2'b00 and OWCDIR=1'b0 in the TMRx_CTRL1 register to enable upcounting mode. In this mode, the counter counts from 0 to the value programmed in the TMRx_PR register, then restarts from 0, and generates a counter overflow event, with the OVFIF bit being set to 1. If the overflow event is disabled, the counter is no longer reloaded with the preload value and period value at a counter overflow event; otherwise, the counter is updated with the preload value and period value on an overflow event.

Figure 14-16 Overflow event when PRBEN=0

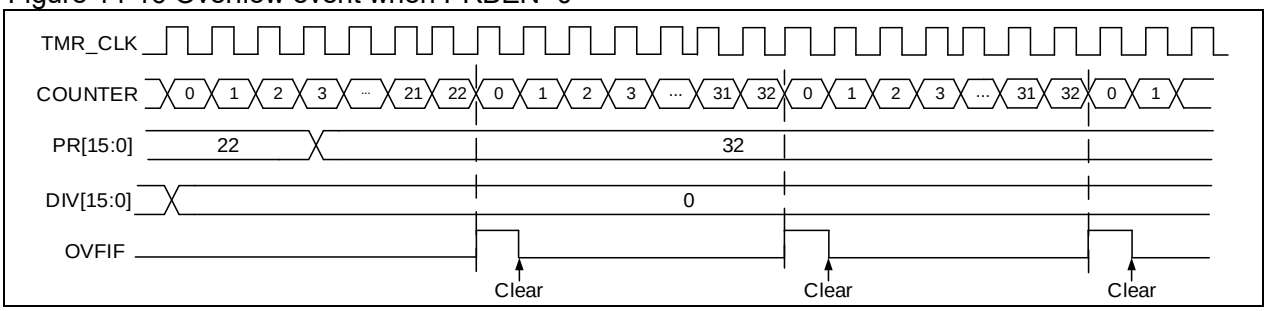
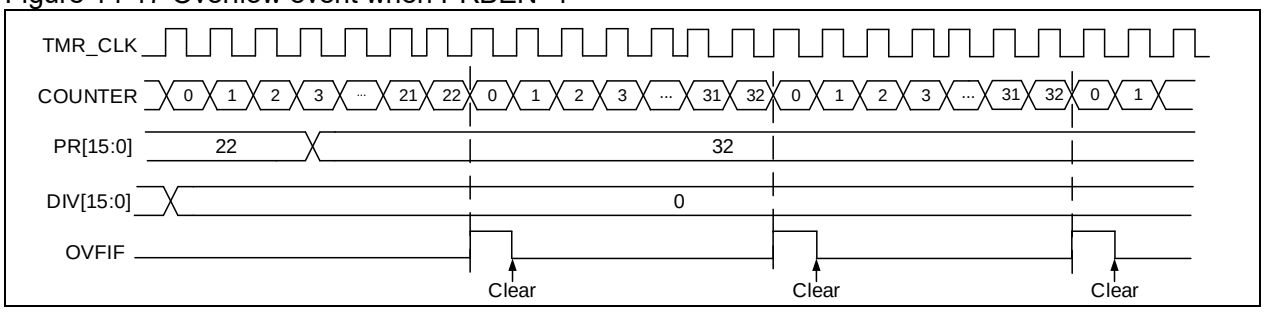


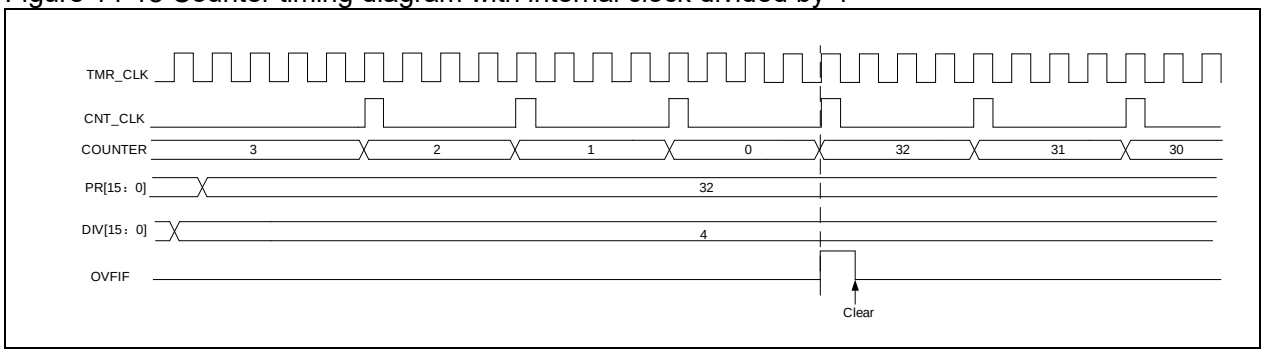
Figure 14-17 Overflow event when PRBEN=1



Downcounting mode

CMSEL[1:0]=2'b00 and OWCDIR=1'b1 in the TMRx_CTRL1register to enable downcounting mode. In this mode, the counter counts from the value programmed in the TMRx_PR register down to 0, and restarts from the value programmed, and generates a counter underflow event.

Figure 14-18 Counter timing diagram with internal clock divided by 4



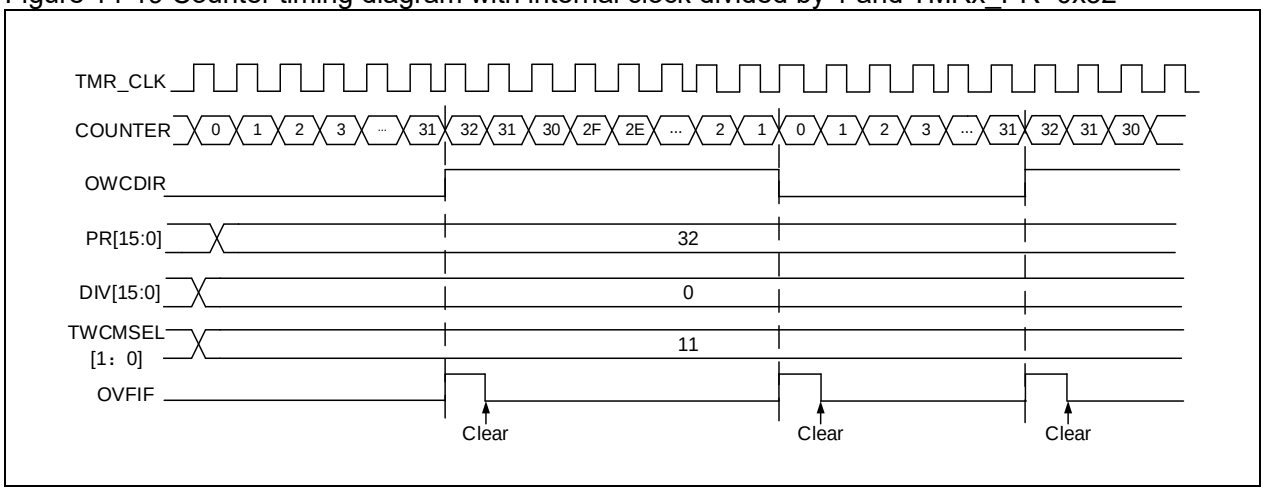
Up/down counting mode

Set CMSEL[1:0]≠2'b00 in the TMRx_CTRL1 register to enable up/down counting mode. In this mode, the counter counts up/down alternatively. When the counter counts from the value programmed in the TMRx_PR register down to 1, an underflow event is generated, and then restarts counting from 0; When the counter counts from 0 to the value of the TMRx_PR register -1, an overflow event is generated, and then restarts counting from the value of the TMRx_PR register. The OWCDIR bit indicates the current counting direction.

The TWCMSEL[1:0] bit in the TMRx_CTRL1 register is also used to select the CxIF flag setting method in up/down counting mode. In up/down counting mode 1 (TWCMSEL[1:0]=2'b01), CxIF flag can only be set when the counter counts down; in up/down counting mode 2 (TWCMSEL[1:0]=2'b10), CxIF flag can only be set when the counter counts up; in up/down counting mode 3 (TWCMSEL[1:0]=2'b11), CxIF flag can be set when the counter counts up/down.

Note: The OWCDIR is ready-only in up/down counting mode.

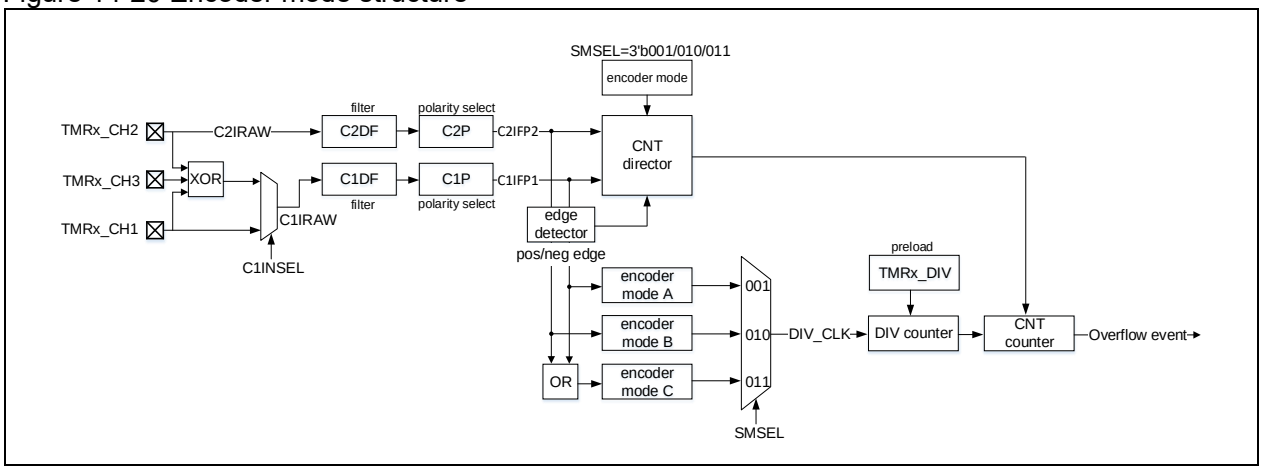
Figure 14-19 Counter timing diagram with internal clock divided by 1 and TMRx_PR=0x32



Encoder interface mode

To enable the encoder interface mode, write SMSEL[2: 0]= 3’b001/3’b010/3’b011. In this mode, the two inputs (C1IN/C2IN) are required. Depending on the level on one input, the counter counts up or down on the edge of the other input. The OWCDIR bit indicates the direction of the counter.

Figure 14-20 Encoder mode structure



Encoder mode A: SMSEL=3’b001. The counter counts on the selected C1IFP1 edge (rising and falling edges), and the counting direction is dependent on the edge direction of C1IFP1 and the level of C2IFP2.

Encoder mode B: SMSEL=3’b010. The counter counts on the selected C2IFP2 edge (rising and falling edges), and the counting direction is dependent on the edge direction of C2IFP2 and the level of C1IFP1.

Encoder mode C: SMSEL=3’b011. The counter counts on both C1IFP1 and C2IFP2 edges (rising and falling edges). The counting direction is dependent on the C1IFP1 edge direction and C2IFP2 level, and C2IFP2 edge direction and C1IFP1 level.

To use encoder mode, follow the procedures below:

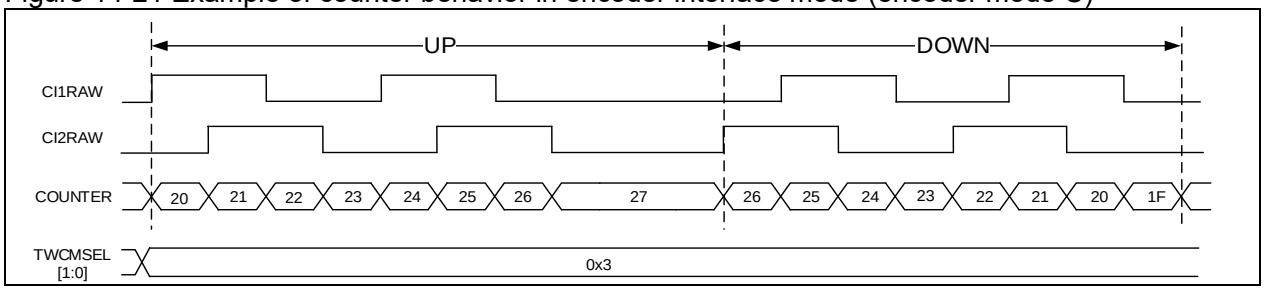
- Set channel 1 input signal filtering through the C1DF[3:0] bit in the TMRx_CM1 register;
  Set channel 1 input signal active level through the C1P bit in the TMRx_CCTRL register
- Set channel 2 input signal filtering through the C2DF[3:0] bit in the TMRx_CM1 register;
  Set channel 2 input signal active level through the C2P bit in the TMRx_CCTRL register
- Set channel 1 as input mode through the C1C[1:0] bit in the TMRx_CM1 register;
  Set channel 2 as input mode through the C2C[1:0] bit in the TMRx_CM1 register
- Select encoder mode A (SMSEL=3’b001), encoder mode B (SMSEL=3’b010), or encoder mode C (SMSEL=3’b011) by setting the SMSEL[2:0] bit in the TMRx_STCTRL register
- Set counting cycles through the PR[15:0] bit in the TMRx_PR register
- Set counting frequency through the DIV[15:0] bit in the TMRx_DIV register
- Configure the corresponding IOs of TMRx_CH1 and TMRx_CH2 as multiplexed mode

– Enable counter through the TMREN bit in the TMRx_CTRL1 register

Table 14-4 Counting direction versus encoder signals

Active edge	Level on opposite signal (C1INFP1 to C2IN, C2INFP2 to C1IN)	C1INFP1 signal		C2INFP2 signal	
		Rising	Falling	Rising	Falling
Count on C1IN only	High	Down	Up	No count	No count
	Low	Up	Down	No count	No count
Count on C2IN only	High	No count	No count	Up	Down
	Low	No count	No count	Down	Up
Count on both C1IN and C2IN	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

Figure 14-21 Example of counter behavior in encoder interface mode (encoder mode C)



14.2.3.3 TMR input function

Each of TMR2~TMR5 timers has four independent channels, with each channel being configured as input or output.

As input, each channel input signal is handled as follows:

– TMRx_CHx outputs the pre-processed CxIRAW. The C1INSEL bit is used to select the source of C1IRAW from TMRx_CH1 or the XOR-ed TMRx_CH1, TMRx_CH2 and TMRx_CH3. The sources of C2IRAW, C3IRAW and C4IRAW are TMRx_CH2, TMRx_CH3 and TMRx_CH4, respectively.

– CxIRAW inputs digital filter and outputs filtered CxIF signal. The digital filter uses the CxDF bit to program sampling frequency and sampling times.

– CxIF inputs edge detector, and outputs the CxIFPx signal after edge selection. The edge selection depends on both CxP and CxCP bits. It is possible to select input rising edge, falling edge or both edges.

– CxIFPx inputs capture signal selector, and outputs the CxIN signal after capture signal selection. The capture signal selection is defined by CxC bit. It is possible to select CxIFPx, CyIFPx or STCI as CxIN source. Of those, CyIFPx (x≠y) is the CyIFPy signal that is from Y channel and processed by channel -x edge detector (for example, C1IFP2 is the channel 1’s C1IFP1 signal that passed through channel 2 edge detection). The STCI comes from slave timer controller, and its source is selected by STIS bit.

– CxIN outputs the CxIPS signal that is divided by input channel divider. The divider factor can be defined as No division, /2, /4 or /8, by the CxIDIV bit. It can be used for filtering, selection, division and input capture of input signals.

Figure 14-22 Input/output channel 1 main circuit

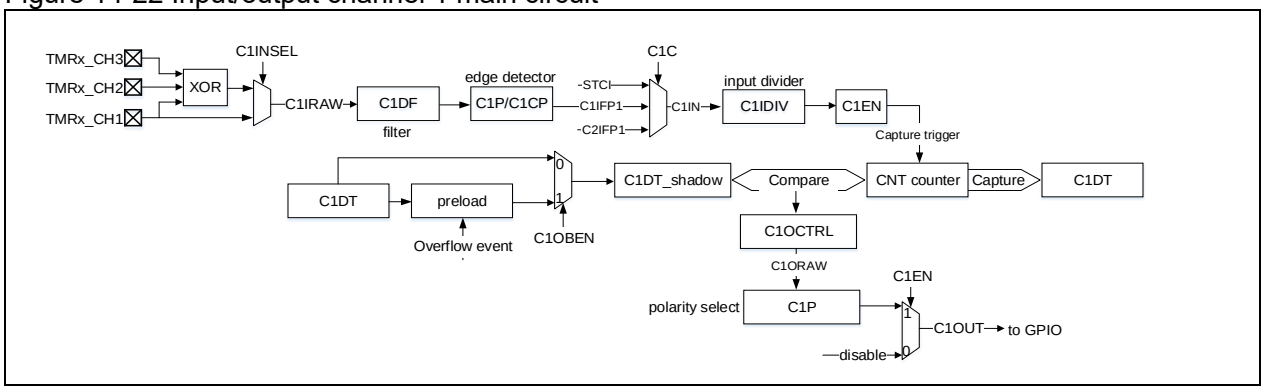
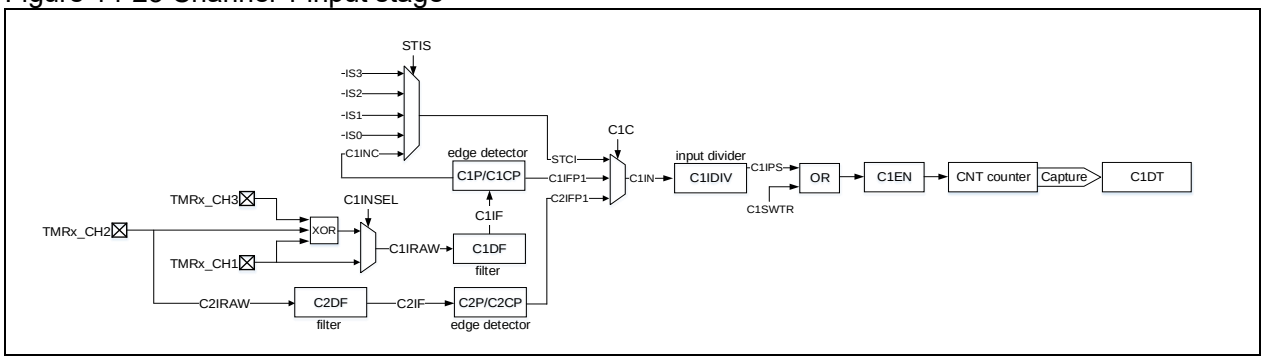


Figure 14-23 Channel 1 input stage



Input mode

In input mode, the TMRx_CxDT registers latch the current counter values after the selected trigger signal is detected, and the capture compare interrupt flag bit (CxIF) is set to 1. An interrupt or a DMA request will be generated if the CxIEN or CxDEN bit is enabled. If the selected trigger signal is detected when the CxIF is set, a capture overflow event occurs. The TMRx_CxDT register overwrites the recorded value with the current counter value, and the CxRF is set to 1.

To capture the rising edge of C1IN input, following the configuration procedure mentioned below:

- Set C1C=01 in the TMRx_CxDT register to select the C1IN as channel 1 input
- Set the filter bandwidth of C1IN signal (CxDF[3: 0])
- Set the active edge on the C1IN channel by writing C1P=0 (rising edge) in the TMRx_CCTR register
- Program the capture frequency division of C1IN signal (C1DIV[1: 0])
- Enable channel 1 input capture (C1EN=1)
- If needed, enable the relevant interrupt or DMA request by setting the C1IEN bit in the TMRx_IDEN register or the C1DEN bit in the TMRx_IDEN register

Timer Input XOR function

The 3 timer input pins (TMRx_CH1, TMRx_CH2 and TMRx_CH3) are connected to the channel 1 (selected by setting the C1INSE in the TMRx_CTRL2 register) through an XOR gate.

The XOR gate can be used to connect Hall sensors. For example, connect the three XOR inputs to the three Hall sensors respectively so as to calculate the position and speed of the rotation by analyzing three Hall sensor signals.

Input selection

The TMR2 IS1 (internal trigger input 1) and TMR5 channel 4 are mappable through the TMRx_RMP register. The TMR2 IS1 can be configured as TMR8_TRGO, Ethernet PTP output, OTG1_FS_SOF or OTG2_FS_SOF. TMR5 channel 4 input can be configured as GPIO, LICK, LEXT or ERTC.

PWM input

The PWM input mode applies to channel 1 and channel 2. To enable this mode, map the C1IN and C2IN to the same TMRx_CHx, and configure the CxIFPx of channel 1/2 to trigger slave timer controller reset.

The PWM input mode can be used to measure the period and duty cycle of input signal. The period and duty cycle of channel 1 can be measured as follows:

- Set C1C=2'b01 to set C1IN as C1IFP1;
- Set C1P=1'b0 to set C1IFP1 rising edge active;
- Set C2C=2'b10 to set C2IN as C1IFP2;
- Set C2P=1'b1 to set C1IFP2 falling edge active;
- Set STIS=3'b101 to set C1IFP1 as the slave timer trigger signal;
- Set SMSEL=3'b110 to set the slave timer in reset mode;
- Set C1EN=1'b1 and C2EN=1'b1 to enable channel 1 and input capture.

In these configurations, the rising edge of channel 1 input signal triggers capture and saves captured values to the C1DT register, and channel 1 input signal rising edge resets the counter. The falling edge of channel 1 input signal triggers capture and saves captured values to the C2DT register. The period and duty of channel 1 input signal can be calculated through C1DT and C2DT respectively.

Figure 14-24 Example of PWM input mode configuration

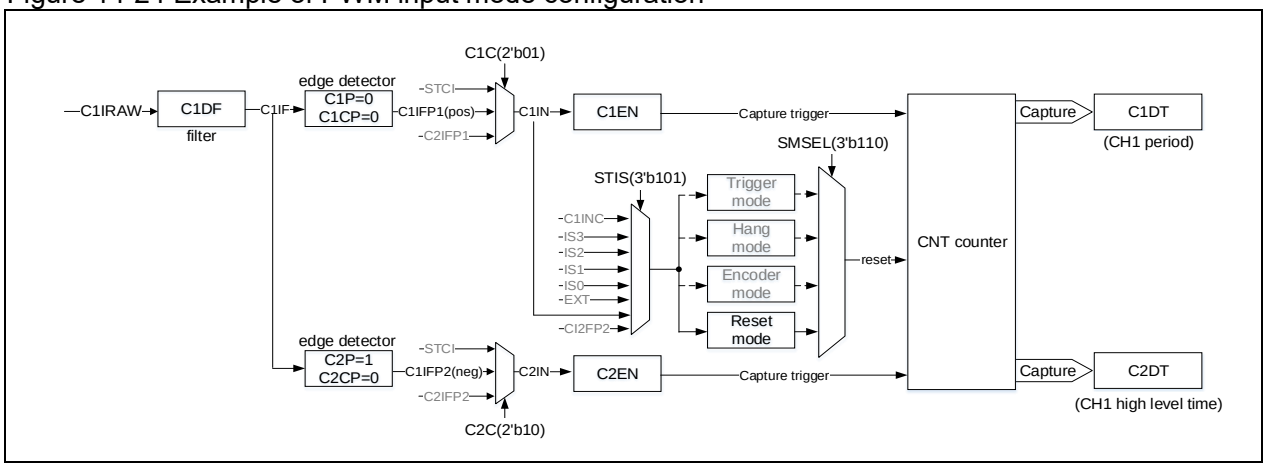
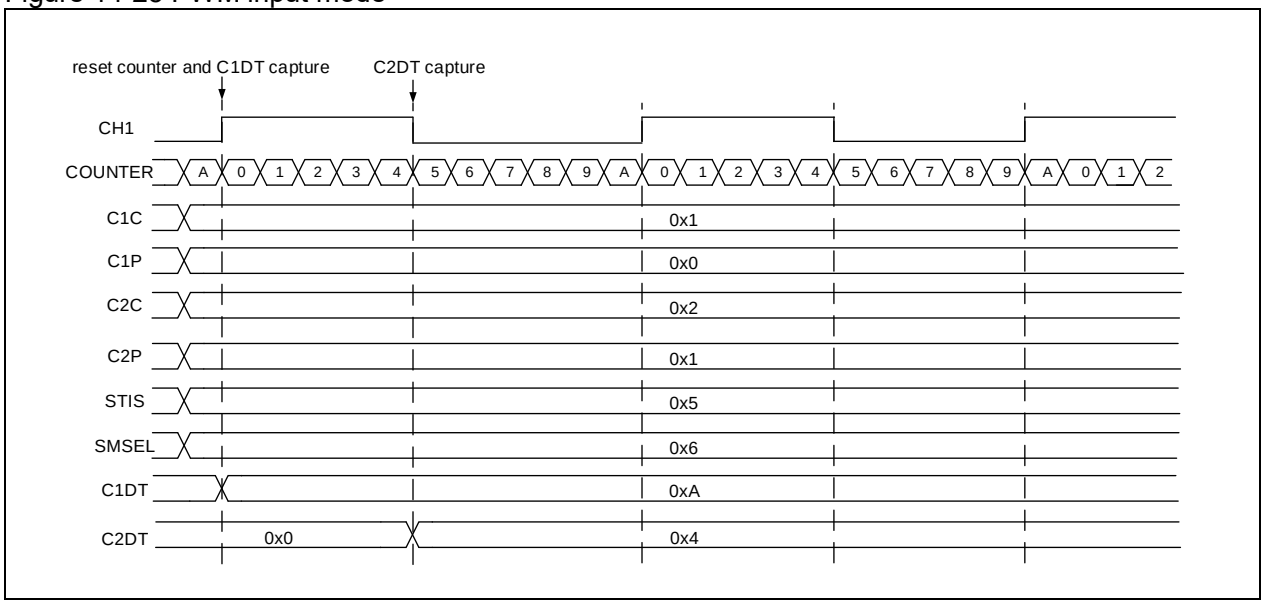


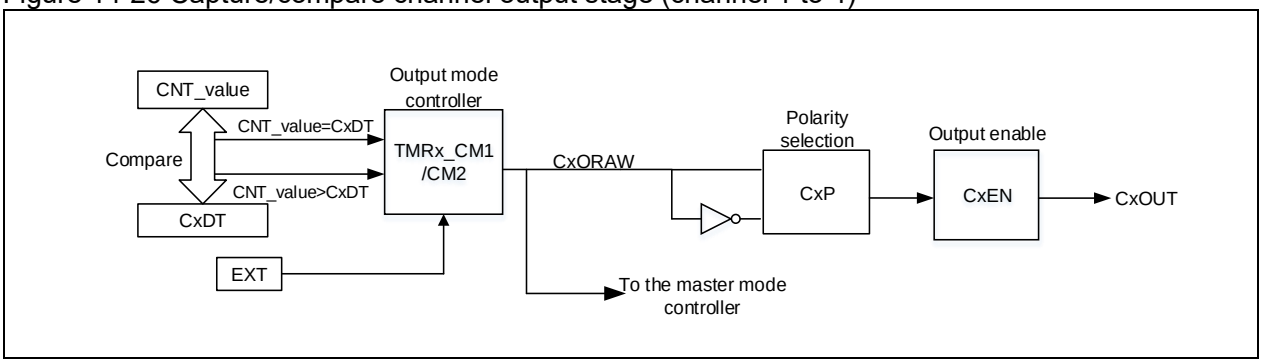
Figure 14-25 PWM input mode



14.2.3.4 TMR output function

The TMR output consists of a comparator and an output controller. It is used to program the period, duty cycle and polarity of the output signal.

Figure 14-26 Capture/compare channel output stage (channel 1 to 4)



Output mode

Write CxC[1 0]≠2'b00 to configure the channel as output to implement multiple output modes. In this case, the counter value is compared with the value in the TMRx_CxDT register, and the intermediate signal CxORAW is generated according to the output mode selected by CxOCTRL[2: 0], which is sent to IO after being processed by the output control circuit. The period of the output signal is configured by the TMRx_PR register, while the duty cycle by the TMRx_CxDT register.

Output compare modes include:

- **PWM mode A**: Set CxOCTRL=3'b110 to enable PWM mode A. In upcounting, when TMRx_C1DT>TMRx_CVAL, C1ORAW outputs high; otherwise, outputs low. In downcounting, when TMRx_C1DT<TMRx_CVAL, C1ORAW outputs low; otherwise, outputs high. To set PWM mode A, the following process is recommended:
  - Set the TMRx_PR register to set PWM period;
  - Set the TMRx_CxDT register to set PWM duty cycle;
  - Set CxOCTRL=3'b110 in TMRx_CM1/CM2 register and set output mode as PWM mode A;
  - Set the TMRx_DIV register to set the counting frequency;
  - Set the TWCMSEL[1:0] bit in the TMRx_CTRL1 to set the count mode;
  - Set the CxP and CxCP bits in the TMRx_CCTRL register to set the output polarity;
  - Set the CxEN and CxCEN bits in the TMRx_CCTRL register to enable channel output;
  - Set the OEN bit in the TMRx_BRK register to enable TMRx output;
  - Set the corresponding GPIO of TMR output channel as the multiplexed mode;
  - Set the TMREN bit in the TMRx_CTRL1 register to enable TMRx counter.
- **PWM mode B**: Set CxOCTRL=3'b111 to enable PWM mode B. In upcounting, when TMRx_C1DT>TMRx_CVAL, C1ORAW outputs low; otherwise, outputs high. In downcounting, when TMRx_C1DT<TMRx_CVAL, C1ORAW outputs high; otherwise, outputs low.
- **Forced output mode:** Set CxOCTRL=3'b100/101 to enable forced output mode. In this case, the CxORAW is forced to be the programmed level, irrespective of the counter value. Despite this, the channel flag bit and DMA request still depend on the compare result.
- **Output compare mode:** Set CxOCTRL=3'b001/010/011 to enable output compare mode. In this case, when the counter value matches the value of the CxDT register, the CxORAW is forced high (CxOCTRL=3'b001), low (CxOCTRL=3'b010) or toggling (CxOCTRL=3'b011).
- **One-pulse mode:** This is a particular case of PWM mode. Set OCMEN=1 to enable one-pulse mode. In this mode, the comparison match is performed in the current counting period. The TMREN bit is cleared as soon as the current counting is completed. Therefore, only one pulse is output. When configured as in upcounting mode, the configuration must follow the rule: CVAL<CxDT≤PR; in downcounting mode, CVAL>CxDT is required.
- **Fast output mode:** Set CxOIEN=1 to enable this mode. If enabled, the CxORAW signal will not change when the counter value matches the CxDT, but at the beginning of the current counting period. In other words, the comparison result is advanced, so the comparison result between the counter value and the TMRx_CxDT register will determine the level of CxORAW in advance.

Figure 27 gives an example of output compare mode (toggle) with C1DT=0x3. When the counter value is equal to 0x3, C1OUT toggles.

Figure 28 gives an example of the combination between upcounting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

Figure 29 gives an example of the combination between up/down counting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

Figure 30 gives an example of the combination between upcounting mode and one-pulse PWM mode B. The counter only counts only one cycle, and the output signal sends only one pulse.

Figure 14-27 C1ORAW toggles when counter value matches the C1DT value

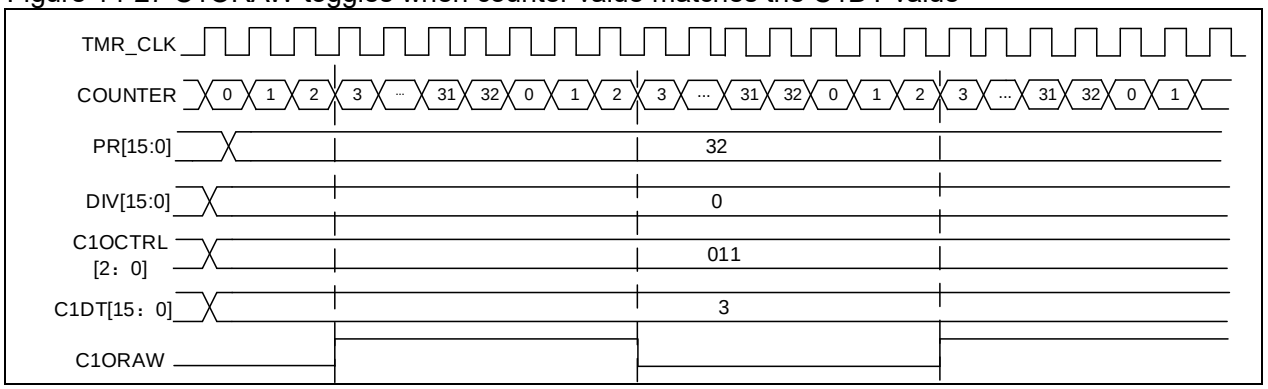


Figure 14-28 Upcounting mode and PWM mode A

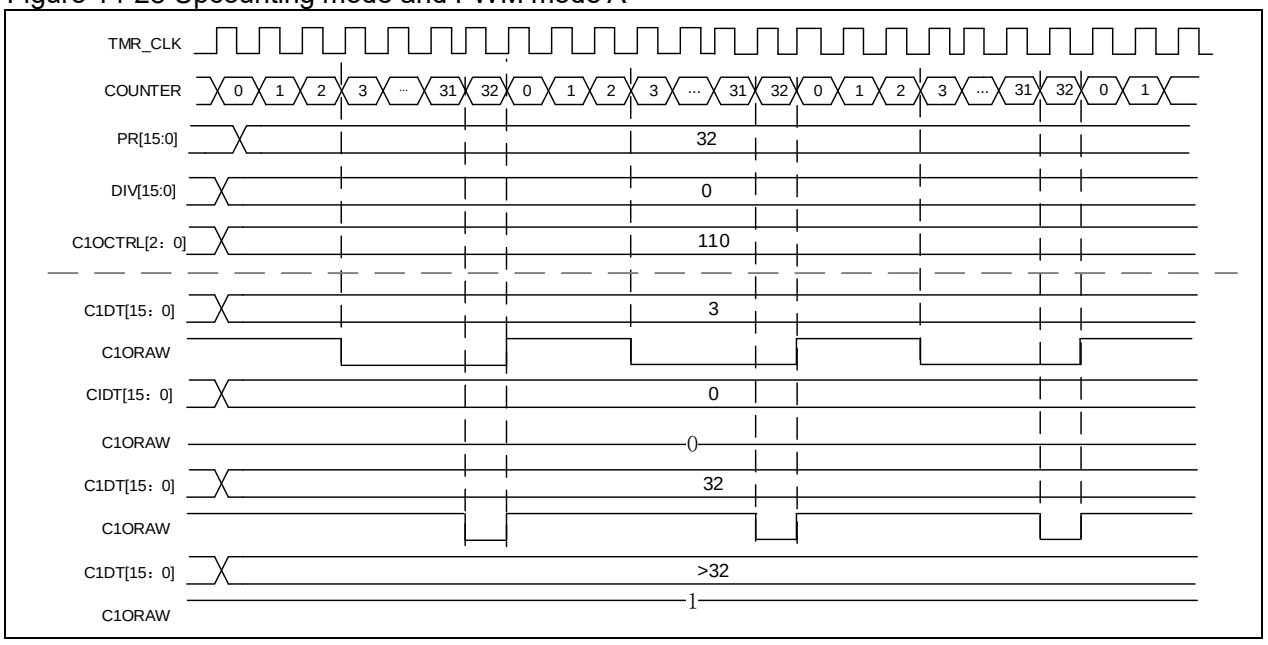


Figure 14-29 Up/down counting mode and PWM mode A

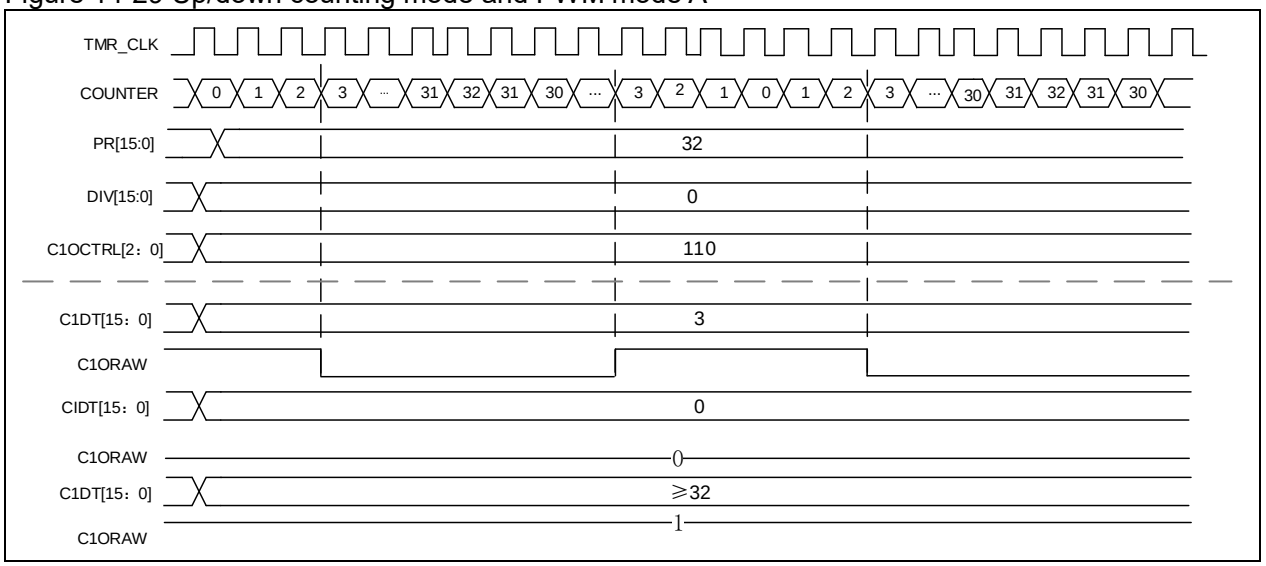
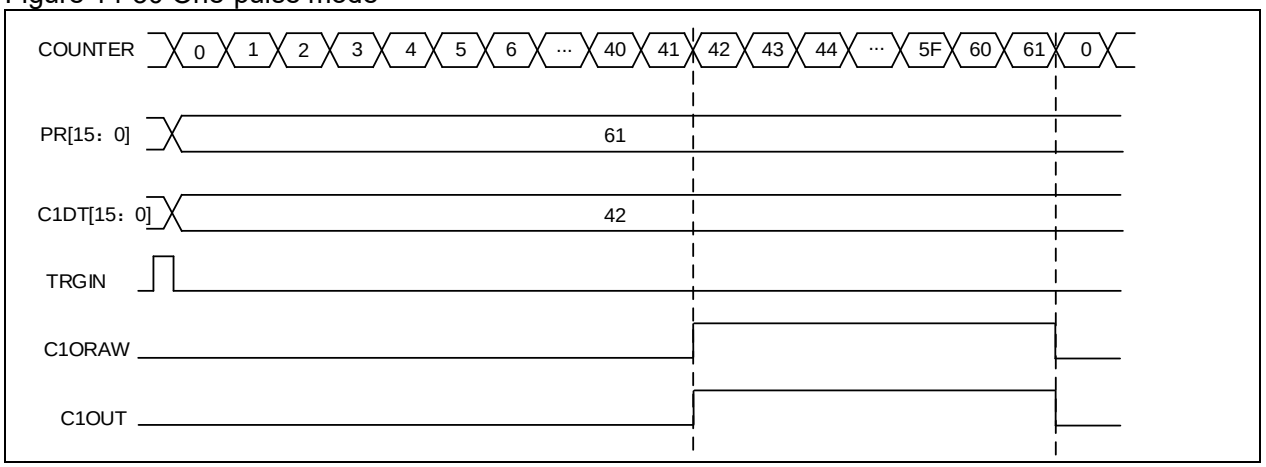


Figure 14-30 One-pulse mode



Master timer event output

When TMR is selected as the master timer, the following signal sources can be selected as TRGOUT signal to output to the slave timer, by setting the PTOS bit in the TMRxCTRL2 register.

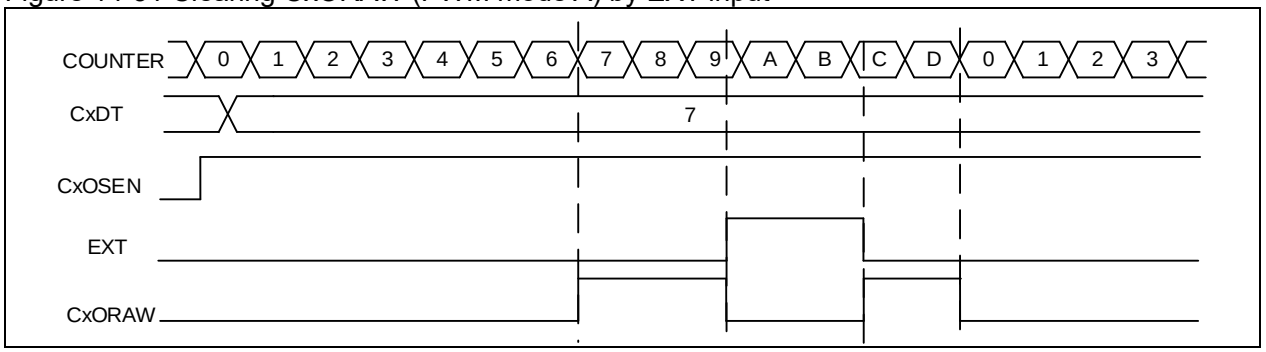
- PTOS=3'b000, TRGOUT outputs software overflow event (OVFSWTR bit in the TMRx_SWEVT register);
- PTOS=3'b001, TRGOUT outputs counter enable signal;
- PTOS=3'b010, TRGOUT outputs counter overflow event;
- PTOS=3'b011, TRGOUT outputs capture and compare event;
- PTOS=3'b100, TRGOUT outputs C1ORAW signal;
- PTOS=3'b101, TRGOUT outputs C2ORAW signal;
- PTOS=3'b110, TRGOUT outputs C3ORAW signal;
- PTOS=3'b111, TRGOUT outputs C4ORAW signal.

CxORAW clear

When the CxOSEN bit is set to 1, the CxORAW signal for a given channel is cleared by applying a high level to the EXT input. The CxORAW signal remains unchanged until the next overflow event.

This function can only be used in output capture or PWM modes, and does not work in forced mode. Figure 14-31 shows the example of clearing CxORAW signal. When the EXT input is high, the CxORAW signal, which was originally high, is driven low; when the EXT is low, the CxORAW signal outputs the corresponding level according to the comparison result between the counter value and CxDT value.

Figure 14-31 Clearing CxORAW (PWM mode A) by EXT input



14.2.3.5 TMR synchronization

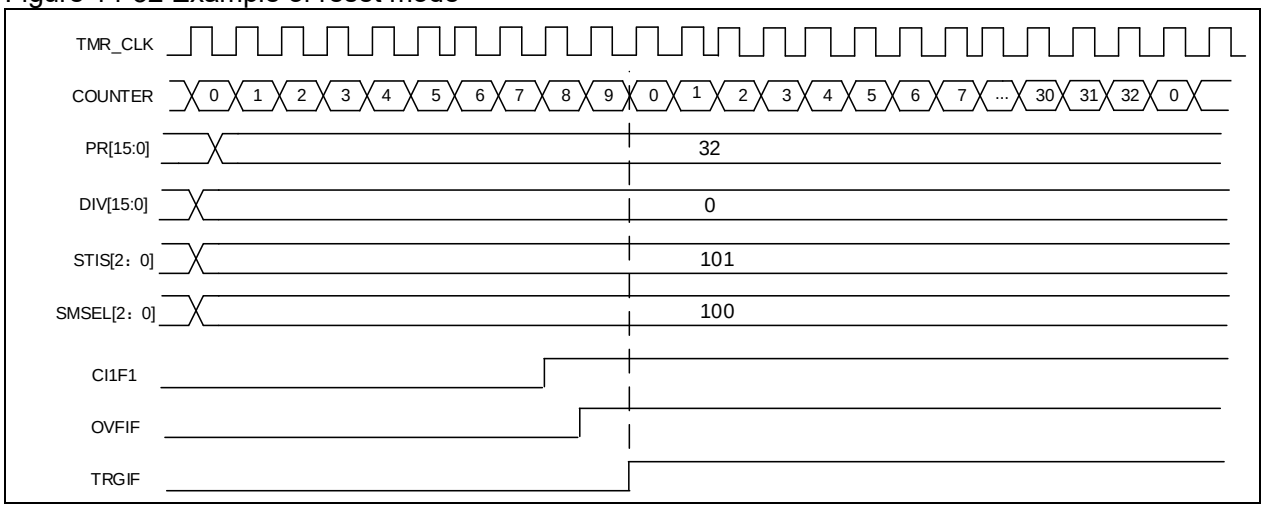
The timers are linked together internally for timer synchronization. Master timer is selected by setting the PTOS[2: 0] bit; Slave timer is selected by setting the SMSEL[2: 0] bit.

Slave mode include:

Slave mode: Reset mode

The counter and its prescaler can be reset by a selected trigger signal. An overflow event is generated when OVFS=0.

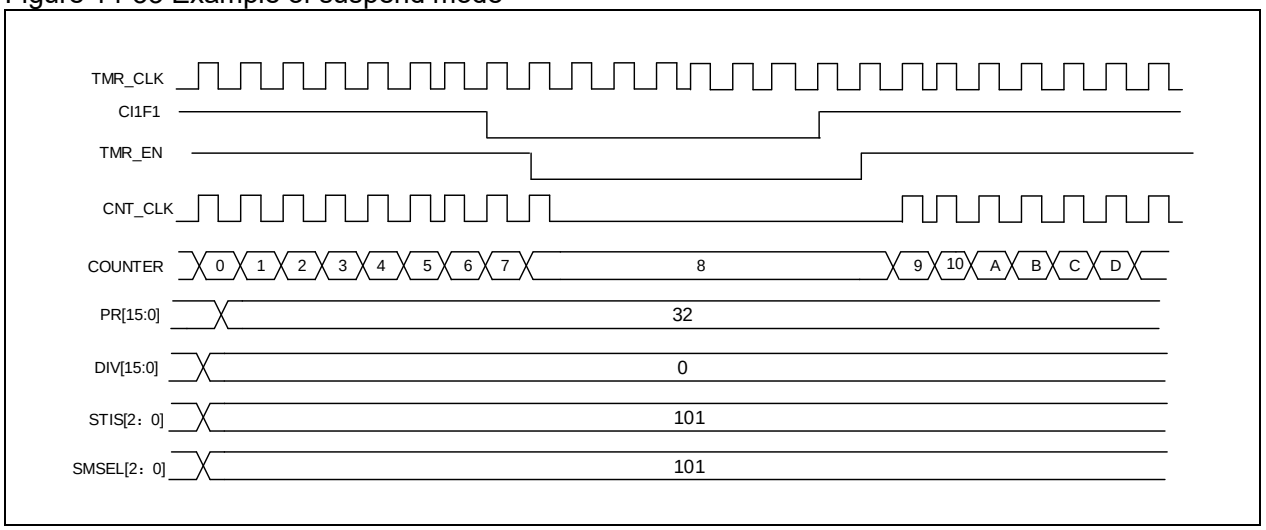
Figure 14-32 Example of reset mode



Slave mode: Suspend mode

In this mode, the counter is controlled by a selected trigger input. The counter starts counting when the trigger input is high and stops as soon as the trigger input is low.

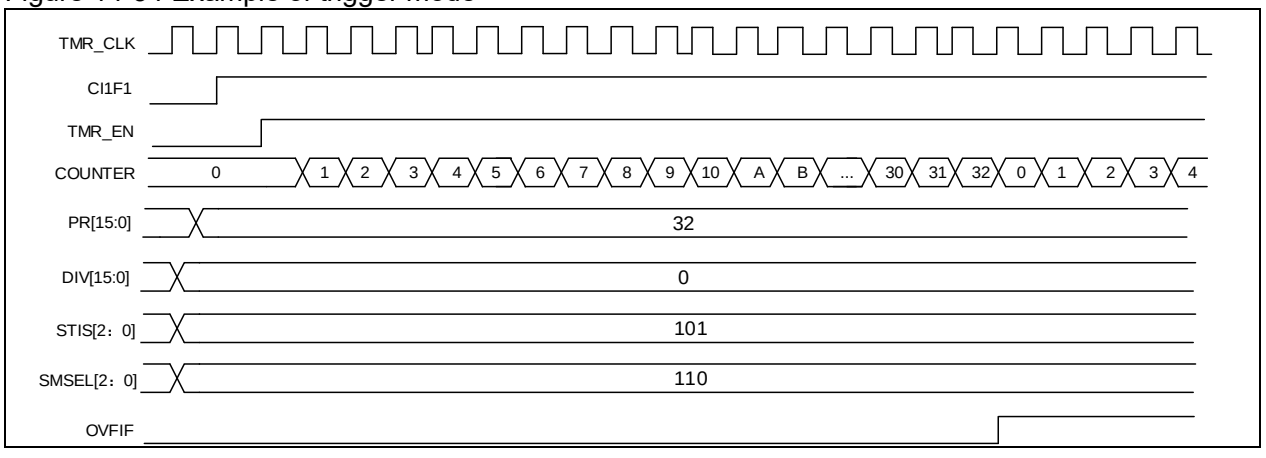
Figure 14-33 Example of suspend mode



Slave mode: Trigger mode

The counter can start counting on the rising edge of a selected trigger input (TMR_EN=1)

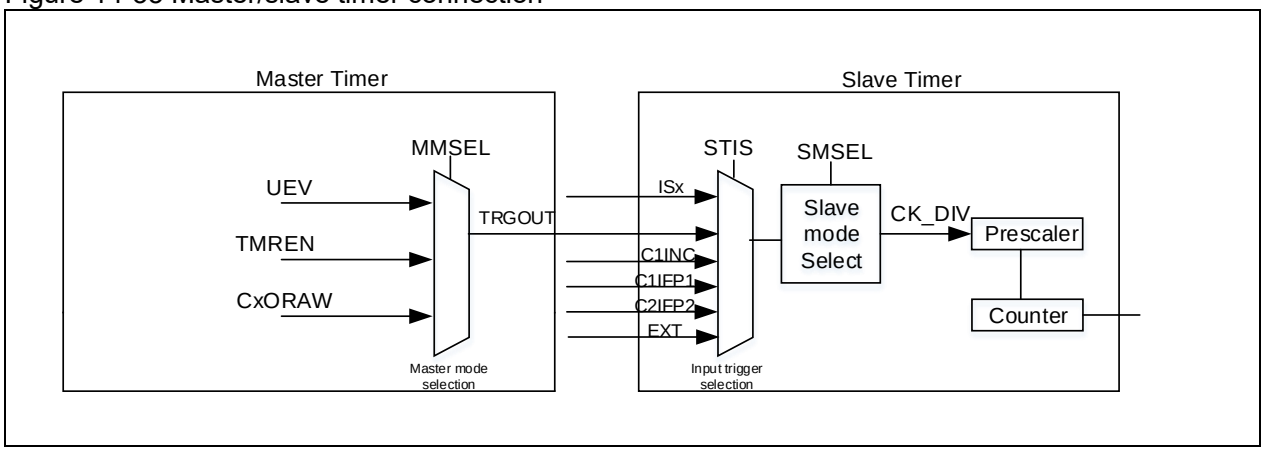
Figure 14-34 Example of trigger mode



Master/slave timer interconnection

Both Master and slave timer can be configured in different master and slave modes respectively. The combination of both them can be used for various purposes. The figure below provides an example of interconnection between master timer and slave timer.

Figure 14-35 Master/slave timer connection



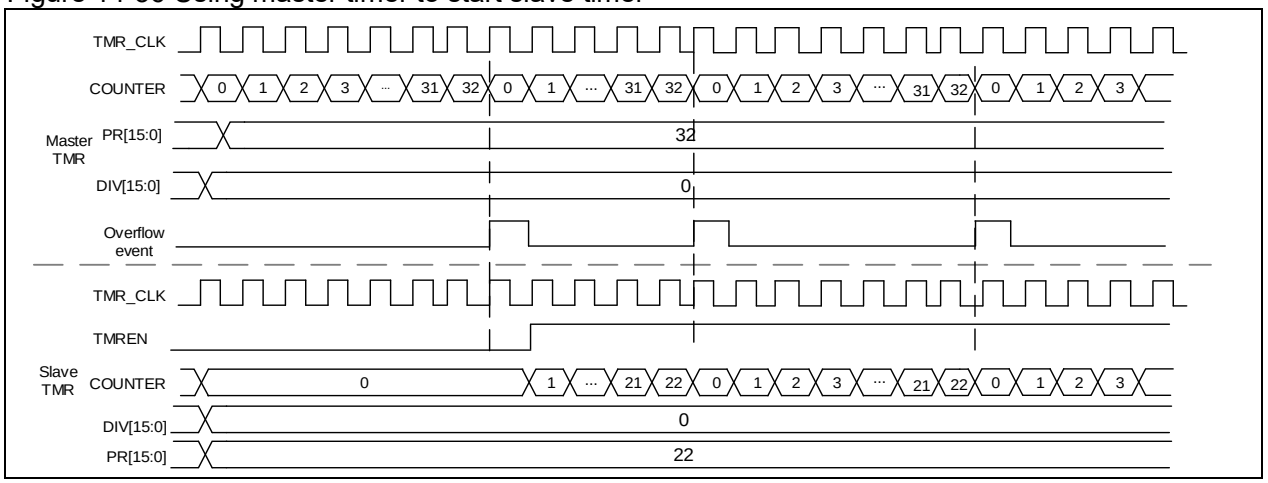
Using master timer to clock the slave timer:

- Configure master timer output signal TRGOUT as an overflow event (PTOS[2: 0]=3'b010). The master timer outputs a pulse signal at each counter overflow event, which is used as the counting clock of the slave timer.
- Configure the master timer counting period (TMRx_PR registers)
- Configure the slave timer trigger input signal TRGIN as master timer output (STIS[2: 0] in the TMRx_STCTRL register)
- Configure the slave timer to use external clock mode A (SMSEL[2: 0]=3'b111 in the TMRx_STCTRL register)
- Set TMREN =1 in both master timer and slave timer to enable them

Using master timer to start slave timer:

- Configure master timer output signal TRGOUT as an overflow event (PTOS[2: 0]=3'b010). The master timer outputs a pulse signal at each counter overflow event, which is used as the counting clock of the slave timer.
- Configure master timer counting period (TMRx_PR registers)
- Configure slave timer trigger input signal TRGIN as master timer input
- Configure slave timer as trigger mode (SMSEL=3'b110 in the TMR2_STCTRL register)
- Set TMREN=1 to enable master timer.

Figure 14-36 Using master timer to start slave timer

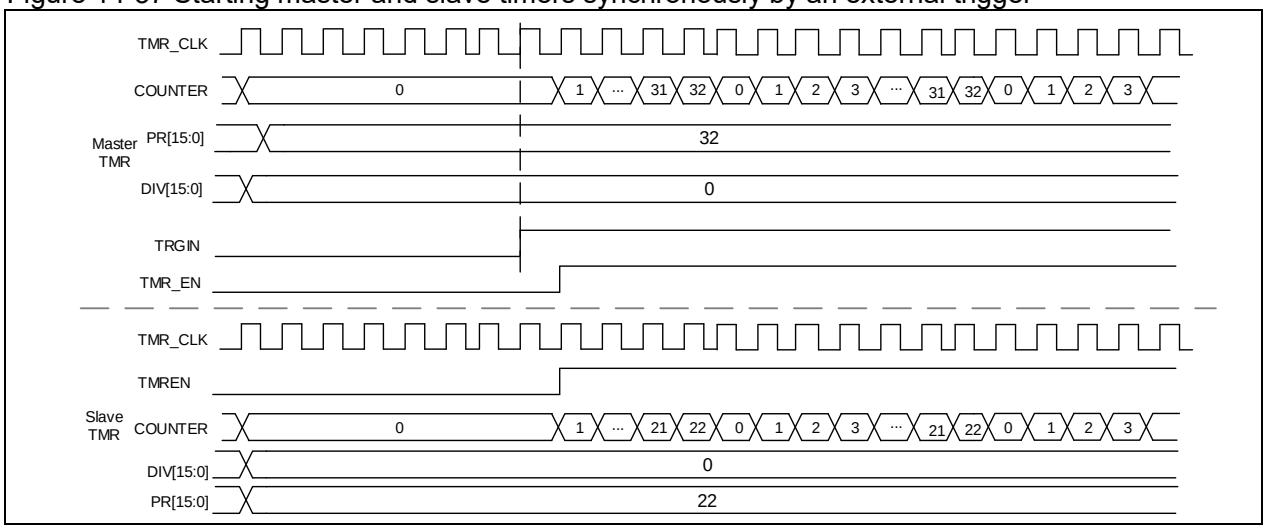


Starting master and slave timers synchronously by an external trigger:

In this example, configure the master timer as master/slave mode synchronously and enable its slave timer synchronization function. This mode is used for synchronization between master timer and slave timer.

- Set the STS bit of the master timer.
- Configure master timer output signal TRGOUT as an overflow event (PTOS[2: 0]=3'b010). The master timer outputs a pulse signal at each counter overflow event, which is used as the counting clock of the slave timer.
- Configure the slave timer mode of the master timer as trigger mode, and select C1IN as trigger source
- Configure slave timer trigger input signal TRGIN as master timer output
- Configure slave timer as trigger mode (SMSEL=3'b110 in the TMR2_STCTRL register)

Figure 14-37 Starting master and slave timers synchronously by an external trigger



14.2.3.6 Debug mode

When the microcontroller enters debug mode (Cortex™-M4F core halted), the TMRx counter stops counting by setting the TMRx_PAUSE in the DEBUG module. Refer to Chapter 30.2 for more information.

14.2.4 TMRx registers

These peripheral registers must be accessed by word (32 bits).

All TMRx register are mapped into a 16-bit addressable space.

Table 14-5 TMRx register map and reset value

Register	Offset	Reset value
TMRx_CTRL1	0x00	0x0000
TMRx_CTRL2	0x04	0x0000
TMRx_STCTRL	0x08	0x0000
TMRx_IDEN	0x0C	0x0000
TMRx_ISTS	0x10	0x0000
TMRx_SWEVT	0x14	0x0000
TMRx_CM1	0x18	0x0000
TMRx_CM2	0x1C	0x0000
TMRx_CCTRL	0x20	0x0000
TMRx_CVAL	0x24	0x0000
TMRx_DIV	0x28	0x0000

TMRx_PR	0x2C	0x0000
TMRx_C1DT	0x34	0x0000
TMRx_C2DT	0x38	0x0000
TMRx_C3DT	0x3C	0x0000
TMRx_C4DT	0x40	0x0000
TMRx_DMACTRL	0x48	0x0000
TMRx_DMADT	0x4C	0x0000
TMR2_RMP	0x50	0x0000
TMR5_RMP	0x50	0x0000

14.2.4.1 TMR2 to TMR5 control register1 (TMRx_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 11	Reserved	0x00	resd	Kept at its default value.
Bit 10	PMEN	0x0	rw	Plus Mode Enable This bit is used to enable TMRx plus mode. In this mode, TMRx_CVAL, TMRx_PR and TMRx_CxDT are extended from 16-bit to 32-bit. 0: Disabled 1: Enabled Note: This function is only valid for TMR2 and TMR5. It is not applicable to other TMRs.
Bit 9: 8	CLKDIV	0x0	rw	Clock division This field is used to set the ratio between digital filter sampling frequency (f_{DT}) and timer clock frequency (f_{CK_INT}). 00: Normal 01: Divided by 2 10: Divided by 4 11: Reserved
Bit 7	PRBEN	0x0	rw	Period buffer enable 0: Period buffer is disabled 1: Period buffer is enabled
Bit 6: 5	TWCMSEL	0x0	rw	Two-way counting mode selection 00: One-way counting mode, depending on the OWCDIR bit 01: Two-way counting mode1, count up and down alternately, the output flag bit is set only when the counter counts down 10: Two-way counting mode2, count up and down alternately, the output flag bit is set only when the counter counts up 11: Two-way counting mode3, count up and down alternately, the output flag bit is set when the counter counts up / down
Bit 4	OWCDIR	0x0	rw	One-way count direction 0: Up 1: Down
Bit 3	OCMEN	0x0	rw	One cycle mode enable This bit is use to select whether to stop counting at an overflow event 0: The counter does not stop at an overflow event 1: The counter stops at an overflow event
Bit 2	OVFS	0x0	rw	Overflow event source This bit is used to select overflow event or DMA request sources. 0: Counter overflow, setting the OVFSWTR bit or overflow event generated by slave timer controller 1: Only counter overflow generates an overflow event

Bit 1	OVFEN	0x0	rw	Overflow event enable 0: Enabled 1: Disabled
Bit 0	TMREN	0x0	rw	TMR enable 0: Disabled 1: Enabled

14.2.4.2 TMR2 to TMR5 control register2 (TMRx_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7	C1INSEL	0x0	rw	C1IN selection 0: CH1 pin is connected to C1IRAW input 1: The XOR result of CH1, CH2 and CH3 pins is connected to C1IRAW input
Bit 6: 4	PTOS	0x0	rw	Master TMR output selection This field is used to select the TMRx signal sent to the slave timer. 000: Reset 001: Enable 010: Update 011: Compare pulse 100: C1ORAW signal 101: C2ORAW signal 110: C3ORAW signal 111: C4ORAW signal
Bit 3	DRS	0x0	rw	DMA request source 0: Capture/compare event 1: Overflow event
Bit 2: 0	Reserved	0x0	resd	Kept at its default value.

14.2.4.3 TMR2 to TMR5 slave timer control register (TMRx_STCTRL)

Bit	Register	Reset value	Type	Description
Bit 15	ESP	0x0	rw	External signal polarity 0: High or rising edge 1: Low or falling edge
Bit 14	ECMBEN	0x0	rw	External clock mode B enable This bit is used to enable external clock mode B 0: Disabled 1: Enabled
Bit 13: 12	ESDIV	0x0	rw	External signal divide This field is used to select the frequency division of an external trigger 00: Normal 01: Divided by 2 10: Divided by 4 11: Divided by 8
Bit 11: 8	ESF	0x0	rw	External signal filter This field is used to filter an external signal. The external signal can be sampled only after it has been generated N times 0000: No filter, sampling by f_{DTS} 0001: $f_{SAMPLING}$ =f_{CK_INT}, N=2 0010: $f_{SAMPLING}$ =f_{CK_INT}, N=4 0011: $f_{SAMPLING}$ =f_{CK_INT}, N=8 0100: $f_{SAMPLING}$ =f_{DTS}/2, N=6 0101: $f_{SAMPLING}$ =f_{DTS}/2, N=8 0110: $f_{SAMPLING}$ =f_{DTS}/4, N=6 0111: $f_{SAMPLING}$=f_{DTS}/4, N=8 1000: $f_{SAMPLING}$=f_{DTS}/8, N=6 1001: $f_{SAMPLING}$=f_{DTS}/8, N=8 1010: $f_{SAMPLING}$=f_{DTS}/16, N=5 1011: $f_{SAMPLING}$=f_{DTS}/16, N=6 1100: $f_{SAMPLING}$=f_{DTS}/16, N=8

				1101: $f_{SAMPLING}$=f_{DTS}/32, N=5 1110: $f_{SAMPLING}$=f_{DTS}/32, N=6 1111: $f_{SAMPLING}$=f_{DTS}/32, N=8
Bit 7	STS	0x0	rw	Subordinate TMR synchronization If enabled, master and slave timer can be synchronized. 0: Disabled 1: Enabled
Bit 6: 4	STIS	0x0	rw	Subordinate TMR input selection This field is used to select the subordinate TMR input. 000: Internal selection 0 (IS0) 001: Internal selection 1 (IS1) 010: Internal selection 2 (IS2) 011: Internal selection 3 (IS3) 100: C1IRAW input detector (C1INC) 101: Filtered input 1 (C1IF1) 110: Filtered input 2 (C1IF2) 111: External input (EXT) Please refer to Table 14-3 and 14-5 for more information on ISx for each timer.
Bit 3	Reserved	0x0	resd	Kept at its default value
Bit 2: 0	SMSEL	0x0	rw	Subordinate TMR mode selection 000: Slave mode is disabled 001: Encoder mode A 010: Encoder mode B 011: Encoder mode C 100: Reset mode — Rising edge of the TRGIN input reinitializes the counter 101: Suspend mode — The counter starts counting when the TRGIN is high 110: Trigger mode — A trigger event is generated at the rising edge of the TRGIN input 111: External clock mode A — Rising edge of the TRGIN input clocks the counter Note: Please refer to count mode section for the details on encoder mode A/B/C.

14.2.4.4 TMR2 to TMR5 DMA/interrupt enable register (TMRx_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15	Reserved	0x0	resd	Kept at its default value
Bit 14	TDEN	0x0	rw	Trigger DMA request enable 0: Disabled 1: Enabled
Bit 13	Reserved	0x0	resd	Kept at its default value
Bit 12	C4DEN	0x0	rw	Channel 4 DMA request enable 0: Disabled 1: Enabled
Bit 11	C3DEN	0x0	rw	Channel 3 DMA request enable 0: Disabled 1: Enabled
Bit 10	C2DEN	0x0	rw	Channel 2 DMA request enable 0: Disabled 1: Enabled
Bit 9	C1DEN	0x0	rw	Channel 1 DMA request enable 0: Disabled 1: Enabled
Bit 8	OVFDEN	0x0	rw	Overflow event DMA request enable 0: Disabled 1: Enabled
Bit 7	Reserved	0x0	resd	Kept at its default value
Bit 6	TIEN	0x0	rw	Trigger interrupt enable 0: Disabled 1: Enabled
Bit 5	Reserved	0x0	resd	Kept at its default value

Bit 4	C4IEN	0x0	rw	Channel 4 interrupt enable 0: Disabled 1: Enabled
Bit 3	C3IEN	0x0	rw	Channel 3 interrupt enable 0: Disabled 1: Enabled
Bit 2	C2IEN	0x0	rw	Channel 2 interrupt enable 0: Disabled 1: Enabled
Bit 1	C1IEN	0x0	rw	Channel 1 interrupt enable 0: Disabled 1: Enabled
Bit 0	OVFIEN	0x0	rw	Overflow interrupt enable 0: Disabled 1: Enabled

14.2.4.5 TMR2 to TMR5 interrupt status register (TMRx_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 13	Reserved	0x0	resd	Kept at its default value
Bit 12	C4RF	0x0	rw0c	Channel 4 recapture flag Please refer to C1RF description.
Bit 11	C3RF	0x0	rw0c	Channel 3 recapture flag Please refer to C1RF description.
Bit 10	C2RF	0x0	rw0c	Channel 2 recapture flag Please refer to C1RF description.
Bit 9	C1RF	0x0	rw0c	Channel 1 recapture flag This bit indicates whether a recapture is detected when C1IF=1. This bit is set by hardware, and cleared by writing “0”. 0: No capture is detected 1: Capture is detected.
Bit 8: 7	Reserved	0x0	resd	Kept at its default value
Bit 6	TRGIF	0x0	rw0c	Trigger interrupt flag This bit is set by hardware on a trigger event. It is cleared by writing “0”. 0: No trigger event occurs 1: Trigger event is generated. Trigger event: an active edge is detected on TRGIN input, or any edge in suspend mode.
Bit 5	Reserved	0x0	resd	Kept at its default value
Bit 4	C4IF	0x0	rw0c	Channel 4 interrupt flag Please refer to C1IF description.
Bit 3	C3IF	0x0	rw0c	Channel 3 interrupt flag Please refer to C1IF description.
Bit 2	C2IF	0x0	rw0c	Channel 2 interrupt flag Please refer to C1IF description.
Bit 1	C1IF	0x0	rw0c	Channel 1 interrupt flag If the channel 1 is configured as input mode: This bit is set by hardware on a capture event. It is cleared by software or read access to the TMRx_C1DT 0: No capture event occurs 1: Capture event is generated If the channel 1 is configured as output mode: This bit is set by hardware on a compare event. It is cleared by software. 0: No compare event occurs 1: Compare event is generated
Bit 0	OVFIF	0x0	rw0c	Overflow interrupt flag This bit is set by hardware on an overflow event. It is cleared by software. 0: No overflow event occurs 1: Overflow event is generated. If OVFEN=0 and OVFS=0 in the TMRx_CTRL1 register:

	− An overflow event is generated when OVFG= 1 in the TMRx_SWEVE register; − An overflow event is generated when the counter CVAL is reinitialized by a trigger event.

14.2.4.6 TMR2 to TMR5 software event register (TMRx_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 7	Reserved	0x000	resd	Kept at its default value.
Bit 6	TRGSWTR	0x0	rw	Trigger event triggered by software This bit is set by software to generate a trigger event. 0: No effect 1: Generate a trigger event.
Bit 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	C4SWTR	0x0	wo	Channel 4 event triggered by software Please refer to C1M description.
Bit 3	C3SWTR	0x0	wo	Channel 3 event triggered by software Please refer to C1M description.
Bit 2	C2SWTR	0x0	wo	Channel 2 event triggered by software Please refer to C1M description
Bit 1	C1SWTR	0x0	wo	Channel 1 event triggered by software This bit is set by software to generate a channel 1 event. 0: No effect 1: Generate a channel 1 event.
Bit 0	OVFSWTR	0x0	wo	Overflow event triggered by software This bit is set by software to generate an overflow event. 0: No effect 1: Generate an overflow event.

14.2.4.7 TMR2 to TMR5 channel mode register1 (TMRx_CM1)

Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15	C2OSEN	0x0	rw	Channel 2 output switch enable
Bit 14: 12	C2OCTRL	0x0	rw	Channel 2 output control
Bit 11	C2OBEN	0x0	rw	Channel 2 output buffer enable
Bit 10	C2OIEN	0x0	rw	Channel 2 output enable immediately
Bit 9: 8	C2C	0x0	rw	Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0': 00: Output 01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS register.
Bit 7	C1OSEN	0x0	rw	Channel 1 output switch enable 0: C1ORAW is not affected by EXT 1: Once high level is detect on EXT input, clear C1ORAW.
Bit 6: 4	C1OCTRL	0x0	rw	Channel 1 output control This field defines the behavior of the original signal C1ORAW. 000: Disconnected. C1ORAW is disconnected from C1OUT; 001: C1ORAW is high when TMRx_CVAL=TMRx_C1DT 010: C1ORAW is low when TMRx_CVAL=TMRx_C1DT 011: Switch C1ORAW level when TMRx_CVAL=TMRx_C1DT 100: C1ORAW is forced low 101: C1ORAW is forced high. 110: PWM mode A – OWCDIR=0, C1ORAW is high once TMRx_C1DT>TMRx_CVAL, else low; – OWCDIR=1, C1ORAW is low once TMRx_ C1DT <TMRx_CVAL, else high; 111: PWM mode B

				– OWCDIR=0, C1ORAW is low once TMRx_ C1DT >TMRx_CVAL, else high; – OWCDIR=1, C1ORAW is high once TMRx_ C1DT <TMRx_CVAL, else low. *Note: In the configurations other than 000', the C1OUT is connected to C1ORAW. The C1OUT output level is not only subject to the changes of C1ORAW, but also the output polarity set by CCTRL.*
Bit 3	C1OBEN	0x0	rw	Channel 1 output buffer enable 0: Buffer function of TMRx_C1DT is disabled. The new value written to the TMRx_C1DT takes effect immediately. 1: Buffer function of TMRx_C1DT is enabled. The value to be written to the TMRx_C1DT is stored in the buffer register, and can be sent to the TMRx_C1DT register only on an overflow event.
Bit 2	C1OIEN	0x0	rw	Channel 1 output enable immediately In PWM mode A or B, this bit is used to accelerate the channel 1 output's response to the trigger event. 0: Need to compare the CVAL with C1DT before generating an output 1: No need to compare the CVAL and C1DT. An output is generated immediately when a trigger event occurs.
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

Input capture mode:

Bit	Register	Reset value	Type	Description
Bit 15: 12	C2DF	0x0	rw	Channel 2 digital filter
Bit 11: 10	C2IDIV	0x0	rw	Channel 2 input divider
Bit 9: 8	C2C	0x0	rw	Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0': 00: Output 01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7: 4	C1DF	0x0	rw	Channel 1 digital filter This field defines the digital filter of the channel 1. N stands for the number of filtering, indicating that the input edge can pass the filter only after N sampling events. 0000: No filter, sampling is done at f_{DTS} 1000: $f_{SAMPLING}=f_{DTS}/8$, N=6 0001: $f_{SAMPLING}=f_{CK_INT}$, N=2 1001: $f_{SAMPLING}=f_{DTS}/8$, N=8 0010: $f_{SAMPLING}=f_{CK_INT}$, N=4 1010: $f_{SAMPLING}=f_{DTS}/16$, N=5 0011: $f_{SAMPLING}=f_{CK_INT}$, N=8 1011: $f_{SAMPLING}=f_{DTS}/16$, N=6 0100: $f_{SAMPLING}=f_{DTS}/2$, N=6 1100: $f_{SAMPLING}=f_{DTS}/16$, N=8 0101: $f_{SAMPLING}=f_{DTS}/2$, N=8 1101: $f_{SAMPLING}=f_{DTS}/32$, N=5 0110: $f_{SMPLING}=f_{DTS}/4$, N=6 1110: $f_{SAMPLING}=f_{DTS}/32$, N=6

				0111: $f_{SAMPLING}=f_{DTS}/4$, N=8 1111: $f_{SAMPLING}=f_{DTS}/32$, N=8
Bit 3: 2	C1IDIV	0x0	rw	Channel 1 input divider This field defines Channel 1 input divider. 00: No divider. An input capture is generated at each active edge. 01: An input compare is generated every 2 active edges 10: An input compare is generated every 4 active edges 11: An input compare is generated every 8 active edges Note: the divider is reset once C1EN='0'
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

14.2.4.8 TMR2 to TMR5 channel mode register2 (TMRx_CM2)

Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15	C4OSEN	0x0	rw	Channel 4 output switch enable
Bit 14: 12	C4OCTRL	0x0	rw	Channel 4 output control
Bit 11	C4OBEN	0x0	rw	Channel 4 output buffer enable
Bit 10	C4OIEN	0x0	rw	Channel 4 output enable immediately
Bit 9: 8	C4C	0x0	rw	Channel 4 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C4EN='0': 00: Output 01: Input, C4IN is mapped on C4IFP4 10: Input, C4IN is mapped on C3IFP4 11: Input, C4IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7	C3OSEN	0x0	rw	Channel 3 output switch enable
Bit 6: 4	C3OCTRL	0x0	rw	Channel 3 output control
Bit 3	C3OBEN	0x0	rw	Channel 3 output buffer enable
Bit 2	C3OIEN	0x0	rw	Channel 3 output enable immediately
Bit 1: 0	C3C	0x0	rw	Channel 3 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C3EN='0': 00: Output 01: Input, C3IN is mapped on C3IFP3 10: Input, C3IN is mapped on C4IFP3 11: Input, C3IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

Input capture mode:

Bit	Register	Reset value	Type	Description
Bit 15: 12	C4DF	0x0	rw	Channel 4 digital filter
Bit 11: 10	C4IDIV	0x0	rw	Channel 4 input divider
Bit 9: 8	C4C	0x0	rw	Channel 4 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C4EN='0': 00: Output 01: Input, C4IN is mapped on C4IFP4 10: Input, C4IN is mapped on C3IFP4 11: Input, C4IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7: 4	C3DF	0x0	rw	Channel 3 digital filter

Bit 3: 2	C3IDIV	0x0	rw	Channel 3 input divider
Bit 1:0	C3C	0x0	rw	Channel 3 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C3EN='0': 00: Output 01: Input, C3IN is mapped on C3IFP3 10: Input, C3IN is mapped on C4IFP3 11: Input, C3IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

14.2.4.9 TMR2 to TMR5 channel control register (TMRx_CCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 14	Reserved	0x0	resd	Kept at its default value.
Bit 13	C4P	0x0	rw	Channel 4 polarity Please refer to C1P description.
Bit 12	C4EN	0x0	rw	Channel 4 enable Please refer to C1EN description.
Bit 11: 10	Reserved	0x0	resd	Default value
Bit 9	C3P	0x0	rw	Channel 3 polarity Please refer to C1P description.
Bit 8	C3EN	0x0	rw	Channel 3 enable Please refer to C1EN description.
Bit 7: 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	C2P	0x0	rw	Channel 2 polarity Please refer to C1P description.
Bit 4	C2EN	0x0	rw	Channel 2 enable Please refer to C1EN description.
Bit 3: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	C1P	0x0	rw	Channel 1 polarity When the channel 1 is configured as output mode: 0: C1OUT is active high 1: C1OUT is active low When the channel 1 is configured as input mode: 0: C1IN active edge is on its rising edge. When used as external trigger, C1IN is not inverted. 1: C1IN active edge is on its falling edge. When used as external trigger, C1IN is inverted.
Bit0	C1EN	0x0	rw	Channel 1 enable 0: Input or output is disabled 1: Input or output is enabled

Table 14-6 Standard CxOUT channel output control bit

CxEN bit	CxOUT output state
0	Output disabled (CxOUT=0, Cx_EN=0)
1	CxOUT = CxORAW + polarity, Cx_EN=1

Note: The state of the external I/O pins connected to the standard CxOUT channel depends on the CxOUT channel state and the GPIO and IOMUX registers.

14.2.4.10 TMR2 to TMR5 counter value (TMRx_CVAL)

Bit	Register	Reset value	Type	Description
Bit 31: 16	CVAL	0x0000	rw	Counter value When TMR2 or TMR5 enables plus mode (the PMEN bit in the TMR_CTRL1 register), the CVAL is expanded to 32 bits.
Bit 15: 0	CVAL	0x0000	rw	Counter value

14.2.4.11 TMR2 to TMR5 division value (TMRx_DIV)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DIV	0x0000	rw	Divider value The counter clock frequency $f_{CK_CNT} = f_{TMR_CLK}$ /(DIV[15: 0]+1). DIV contains the value written at an overflow event.

14.2.4.12 TMR2 to TMR5 period register (TMRx_PR)

Bit	Register	Reset value	Type	Description
Bit 31: 16	PR	0x0000	rw	Period value When TMR2 or TMR5 enables plus mode (the PMEN bit in the TMR_CTRL1 register), the PR is expanded to 32 bits.
Bit 15: 0	PR	0x0000	rw	Period value This defines the period value of the TMRx counter. The timer stops working when the period value is 0.

14.2.4.13 TMR2 to TMR5 channel 1 data register (TMRx_C1DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	C1DT	0x0000	rw	Channel 1 data register When TMR2 or TMR5 enables plus mode (the PMEN bit in the TMR_CTRL1 register), the C1DT is expanded to 32 bits.
Bit 15: 0	C1DT	0x0000	rw	Channel 1 data register When the channel 1 is configured as input mode: The C1DT is the CVAL value stored by the last channel 1 input event (C1IN) When the channel 1 is configured as output mode: C1DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C1OBEN bit, and the corresponding output is generated on C1OUT as configured.

14.2.4.14 TMR2 to TMR5 channel 2 data register (TMRx_C2DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	C2DT	0x0000	rw	Channel 2 data register When TMR2 or TMR5 enables plus mode (the PMEN bit in the TMR_CTRL1 register), the C2DT is expanded to 32 bits.
Bit 15: 0	C2DT	0x0000	rw	Channel 2 data register When the channel 2 is configured as input mode: The C2DT is the CVAL value stored by the last channel 2 input event (C1IN) When the channel 2 is configured as output mode: C2DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C2OBEN bit, and the corresponding output is generated on C2OUT as configured.

14.2.4.15 TMR2 to TMR5 channel 3 data register (TMRx_C3DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	C3DT	0x0000	rw	Channel 3 data register When TMR2 or TMR5 enables plus mode (the PMEN bit in the TMR_CTRL1 register), the C3DT is expanded to 32 bits.
Bit 15: 0	C3DT	0x0000	rw	Channel 3 data register When the channel 3 is configured as input mode: The C3DT is the CVAL value stored by the last channel 3 input event (C1IN) When the channel 3 is configured as output mode: C3DT is the value to be compared with the CVAL value.

				Whether the written value takes effective immediately depends on the C3OBEN bit, and the corresponding output is generated on C3OUT as configured.

14.2.4.16 TMR2 to TMR5 channel 4 data register (TMRx_C4DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	C4DT	0x0000	rw	Channel 4 data register When TMR2 or TMR5 enables plus mode (the PMEN bit in the TMR_CTRL1 register), the C4DT is expanded to 32 bits.
Bit 15: 0	C4DT	0x0000	rw	Channel 4 data register When the channel 4 is configured as input mode: The C4DT is the CVAL value stored by the last channel 4 input event (C1IN) When the channel 4 is configured as output mode: C4DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C4OBEN bit, and the corresponding output is generated on C4OUT as configured.

14.2.4.17 TMR2 to TMR5 DMA control register (TMRx_DMACTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 13	Reserved	0x0	resd	Kept at its default value.
Bit 12: 8	DTB	0x00	rw	DMA transfer bytes This field defines the number of DMA transfers: 00000: 1 byte 00001: 2 bytes 00010: 3 bytes 00011: 4 bytes .... 10000: 17 bytes 10001: 18 bytes
Bit 7: 5	Reserved	0x0	resd	Kept at its default value.
Bit 4: 0	ADDR	0x00	rw	DMA transfer address offset ADDR is defined as an offset starting from the address of the TMRx_CTRL1 register. 00000: TMRx_CTRL1 00001: TMRx_CTRL2 00010: TMRx_STCTRL .....

14.2.4.18 TMR2 to TMR5 DMA data register (TMRx_DMADT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DMADT	0x0000	rw	DMA data register A read or write operation to the DMADT register accesses the TMR registers at the following address: TMRx peripheral address + ADDR*4 to TMRx peripheral address + ADDR*4 + DTB*4.

14.2.4.19 TMR5 channel input remapping register (TMR2_RMP)

Bit	Register	Reset value	Type	Description
Bit 15: 12	Reserved	0x0	resd	Kept at its default value.
Bit 11: 10	TMR2_IS1_IRMP	0x0	rw	TMR2 IS1 input remap 00: TMR8_TRGO output 01: Ethernet PTP output 10: OTG1_FS_SOF 11: OTG2_FS_SOF
Bit 9:0	Reserved	0x000	resd	Kept at its default value.

14.2.4.20 TMR2 channel input remapping register (TMR5_RMP)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7: 6	TMR5_CH4_IRMP	0x0	rw	TMR5 channel 4 input remap 00: TMR5 channel 4 input connected to GPIO

				01: Internal clock LICK 10: Internal clock LEXT 11: ERTC wakeup interrupt
Bit 5: 0	Reserved	0x00	resd	Kept at its default value.

14.3 General-purpose timer (TMR9 to TMR14)

14.3.1 TMR9 to TMR14 introduction

The general-purpose timer (TMR9 to TMR14) consists of a 16-bit counter supporting upcounting mode. These timers can be synchronized.

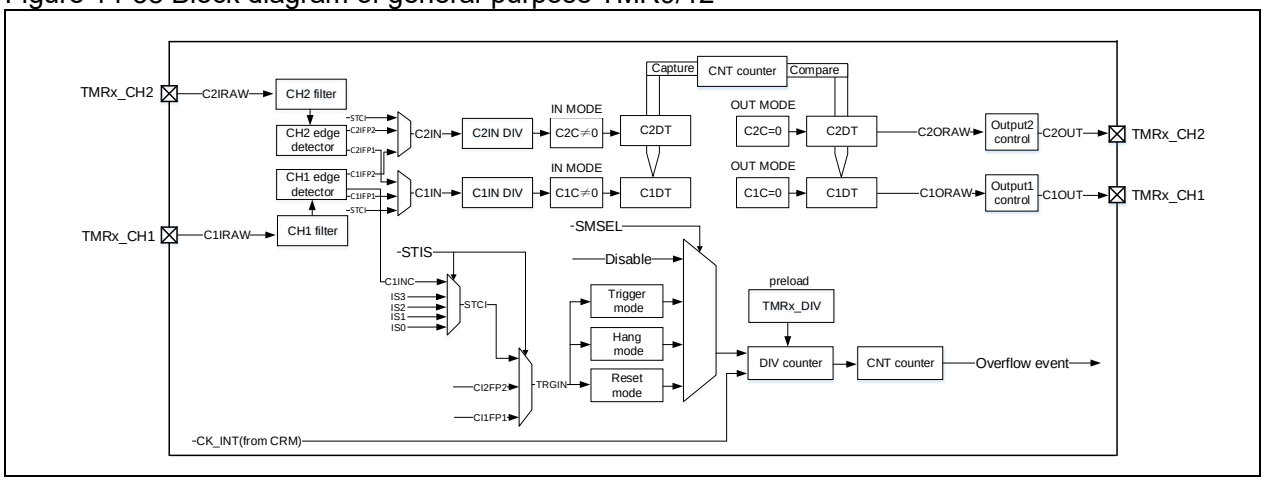
14.3.2 TMR9 to TMR14 main features

14.3.2.1 TMR9 and TMR12 main features

The main functions of general-purpose TMR9 and TMR12 include:

- Source of counter clock: internal clock and external clock
- 16-bit upcounter
- 2 independent channels for input capture, output compare, PWM generation and one-pulse mode output
- Synchronization control between master and slave timers
- Interrupt is generated at overflow event, trigger event and channel event

Figure 14-38 Block diagram of general-purpose TMR9/12

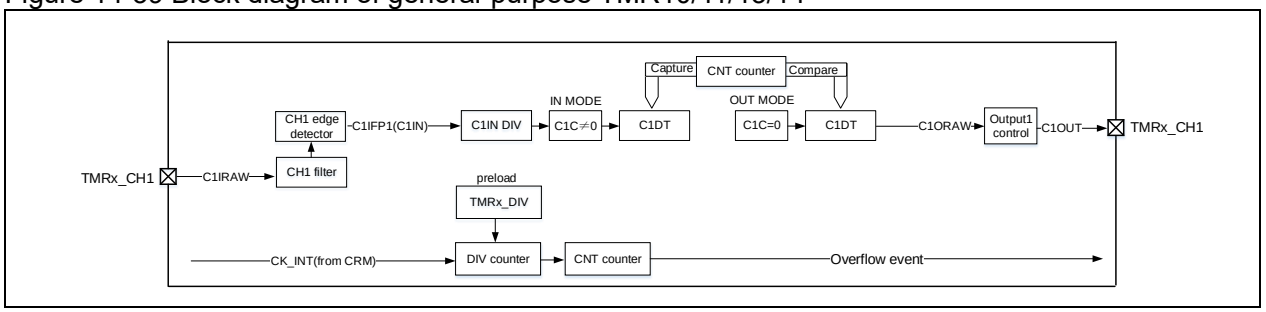


14.3.2.2 TMR10, TMR11, TMR13 and TMR14 main features

The main functions of general-purpose TMRx (TMR10, TMR11, TMR13 and TMR14) include:

- Source of counter clock: internal clock
- 16-bit upcounter
- 1 independent channel for input capture, output compare, PWM generation
- Synchronization control between master and slave timers
- Interrupt is generated at overflow event and channel event

Figure 14-39 Block diagram of general-purpose TMR10/11/13/14

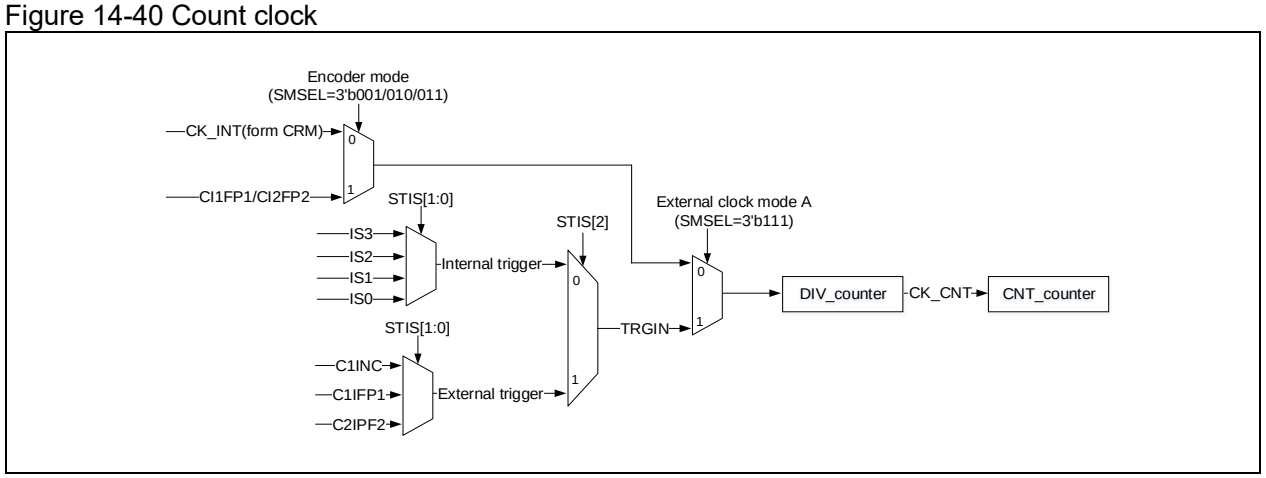


14.3.3 TMR9 to TMR14 functional overview

14.3.3.1Count clock

The count clock of general-purpose timers can be provided by the internal clock (CK_INT), external clock (external clock mode A) and internal trigger input (ISx).

Figure 14-40 Count clock

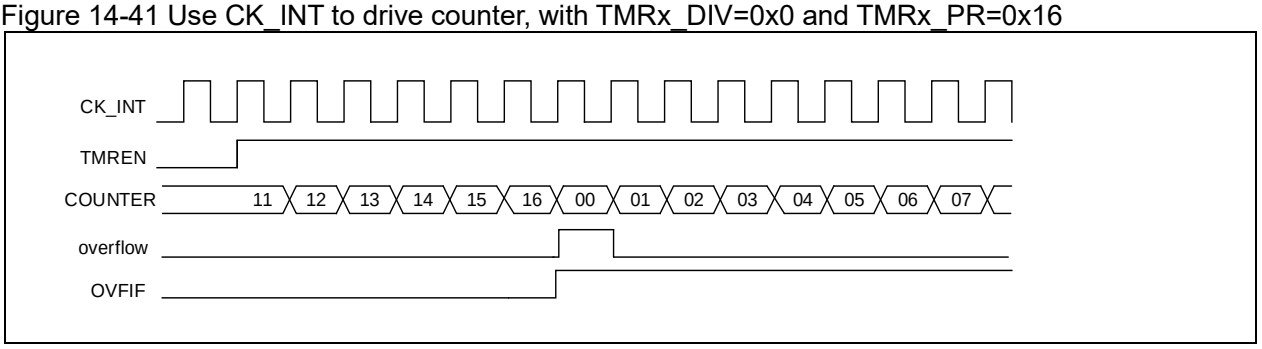


Internal clock (CK_INT)

By default, the CK_INT divided by the prescaler is used to drive the counter to start counting. The configuration process is as follows:

- Set the CLKDIV[1:0] bit in the TMRx_CTRL1 register to set the CK_INT frequency;
- Set the TWCMSEL[1:0] bit in the TMRx_CTRL1 register to select count mode. If the one-way count direction is set, configure OWCDIR bit in the TMRx_CTRL1 register to select the specific direction;
- Set the TMRx_DIV register to set the counting frequency;
- Set the TMRx_PR register to set the counting period;
- Set the TMREN it in the TMRx_CTRL1 register to enable the counter.

Figure 14-41 Use CK_INT to drive counter, with TMRx_DIV=0x0 and TMRx_PR=0x16



External clock (TMR9/12 only)

The counter clock can be provided by TRGIN signal.

SMSEL=3'b111: External clock mode A is selected. Select an external clock source TRGIN signal by

setting the STIS[2:0] bit to drive the counter to start counting. The external clock sources include: C1INC (STIS=3'b100, channel 1 rising edge and falling edge), C1IFP1 (STIS=3'b101, channel 1 signal with filtering and polarity selection) and C2IFP2 (STIS=3'b110, channel 2 signal with filtering and polarity selection).

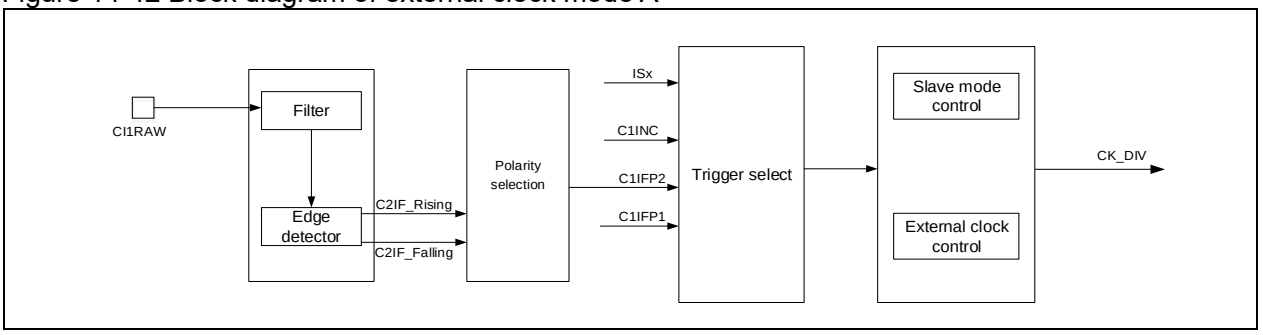
To use external clock mode A, follow the steps below:

- Set external source TRGIN parameters

  If the TMRx_CH1 is used as a source of TRGIN, it is necessary to configure channel 1 input filter (C1DF[3:0] in TMRx_CM1 register) and channel 1 input polarity (C1P/C1CP in TMRx_CCTRL register);

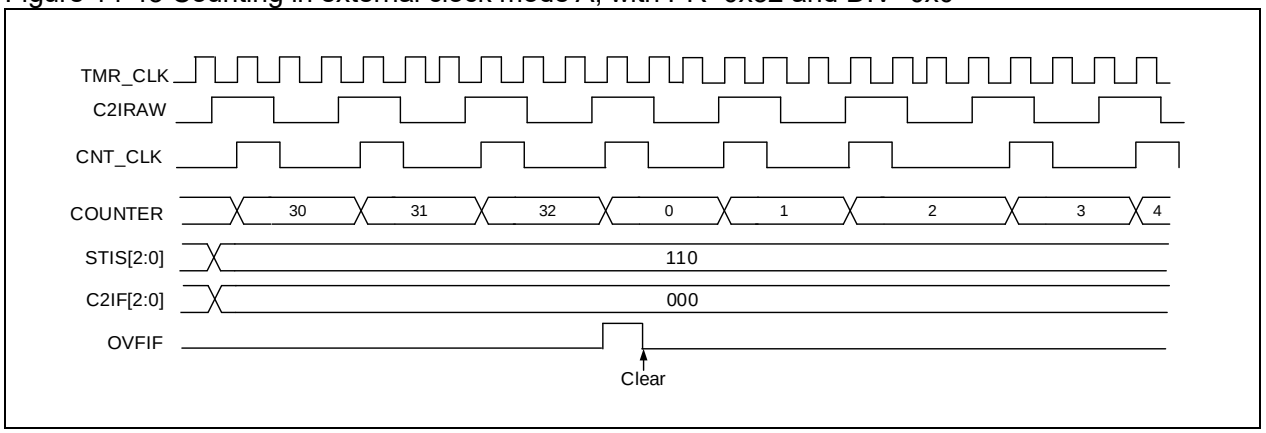
  If the TMRx_CH2 is used as source of TRGIN, it is necessary to configure channel 1 input filter (C2DF[3:0] in TMRx_CM1 register) and channel 2 input polarity (C2P/C2CP in TMRx_CCTR register);
- Set TRGIN signal source using the STIS[1:0] bit in TMRx_STCTRL register
- Enable external clock mode A by setting SMSEL=3'b111 in TMRx_STCTR register
- Set counting frequency through the DIV[15:0] in TMRx_DIV register
- Set counting period through the PR[15:0] in TMRx_PR register
- Enable counter through the TMREN bit in TMRx_CTRL1 register

Figure 14-42 Block diagram of external clock mode A



Note: The delay between the signal on the input side and the actual clock of the counter is due to the synchronization circuit.

Figure 14-43 Counting in external clock mode A, with PR=0x32 and DIV=0x0



Internal trigger input (ISx)

Timer synchronization allows interconnection between several timers. The TMR_CLK of one timer can be provided by the TRGOUT signal output by another timer. Set the STIS[2: 0] bit to select internal trigger signal to enable counting.

Each timer (TMR9 ~TMR14) consists of a 16-bit prescaler, which is used to generate the CK_CNT that enables the counter to count. The frequency division relationship between the CK_CNT and TMR_CLK can be adjusted by setting the value of the TMRx_DIV register. The prescaler value can be modified at any time, but it takes effect only when the next overflow event occurs.

The internal trigger input is configured as follows:

- Set the TMRx_PR register to set the counting period;
- Set the TMRx_DIV register to set the counting frequency;
- Set the STIS[2:0] bit (range: 3'b000~3'b011) in the TMRx_STCTRL register and select internal trigger;
- Set SMSEL[2:0]=3'b111 in the TMRx_STCTRL register and select external clock mode A;
- Set the TMREN bit in the TMRx_CTRL1 register to enable TMRx counter.

Figure 14-44 Counter timing with prescaler value changing from 1 to 4

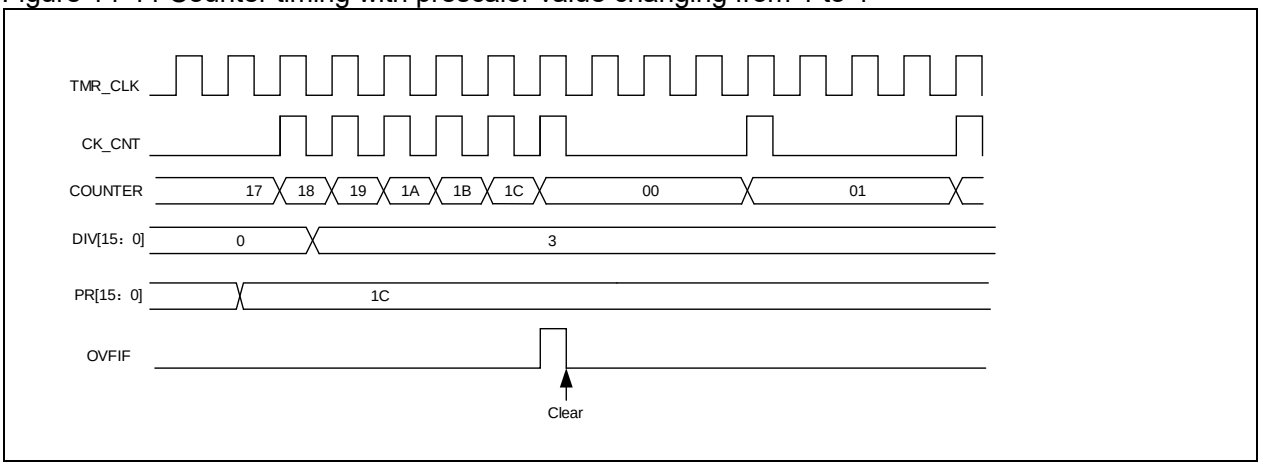


Table 14-7 TMRx internal trigger connection

Slave controller	IS0 (STIS=000)	IS1 (STIS = 001)	IS2 (STIS = 010)	IS3 (STIS = 011)
TMR9	TMR2_TRGOUT	TMR3_TRGOUT	TMR10_OC	TMR11_OC
TMR12	TMR4_TRGOUT	TMR5_TRGOUT	TMR13_OC	TMR14_OC

Note: If there is no corresponding timer in a device, the corresponding trigger signal ISx is not present.

14.3.3.2 Counting mode

The general-purpose timer only supports upcounting mode, and it consists of a 16-bit counter.

The TMRx_PR register issued to set the counting period. The value in the TMRx_PR is immediately moved to the shadow register by default. When the periodic buffer is enabled (PRBEN=1), the value in the TMRx_PR register is transferred to the shadow register only at an overflow event. The OVFEN and OVFS bits are used to configure the overflow event.

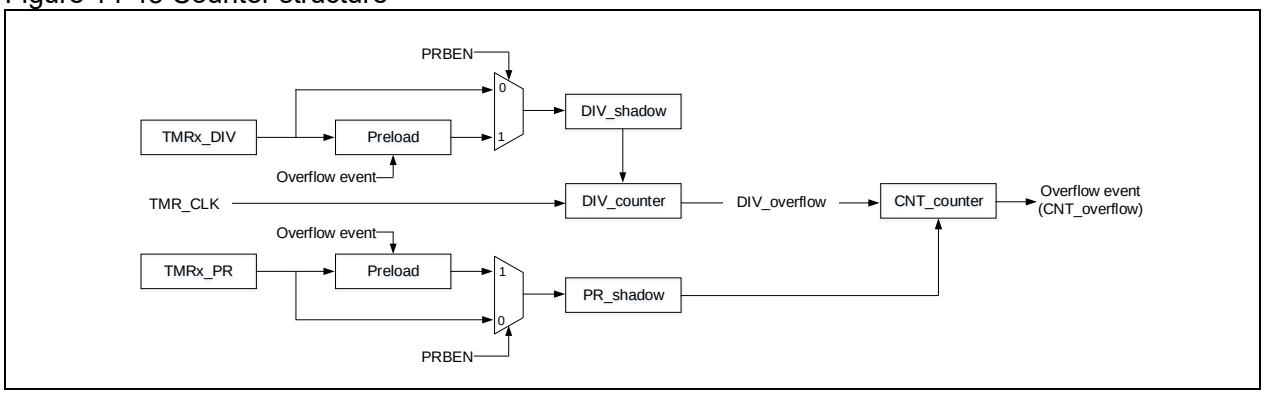
The TMRx_DIV register is used to configure the counting frequency. The counter counts once every count clock period (DIV[15:0]+1). Similar to the TMRx_PR register, when the periodic buffer is enabled, the value in the TMRx_DIV register is updated to the shadow register at an overflow event.

Reading the TMRx_CNT register returns to the current counter value, and writing to the TMRx_CNT register updates the current counter value to the value being written.

An overflow event is generated by default. Set OVFEN=1 in the TMRx_CTRL1 to disable generation of update events. The OVFS bit in the TMRx_CTRL1 register is used to select overflow event source. By default, counter overflow/underflow, setting OVFSWTR bit and the reset signal generated by the slave timer controller in reset mode trigger the generation of an overflow event. When the OVFS bit is set, only counter overflow/underflow triggers an overflow event.

Setting TMREN=1 to enable the timer to start counting. Base on synchronization logic, however, the actual enable signal TMR_EN is set 1 clock cycle after the TMREN is set.

Figure 14-45 Counter structure



Upcounting mode

Set CMSEL[1:0]=2'b00 and OWCDIR=1'b0 in the TMRx_CTRL1 register to enable upcounting mode. In this mode, the counter counts from 0 to the value programmed in the TMRx_PR register, then restarts from 0, and generates a counter overflow event, with the OVFIF bit being set to 1. If the overflow event is disabled, the counter is no longer reloaded with the preload value and period value at a counter overflow event, otherwise, the counter is updated with the preload value and period value on an overflow event.

Figure 14-46 Overflow event when PRBEN=0

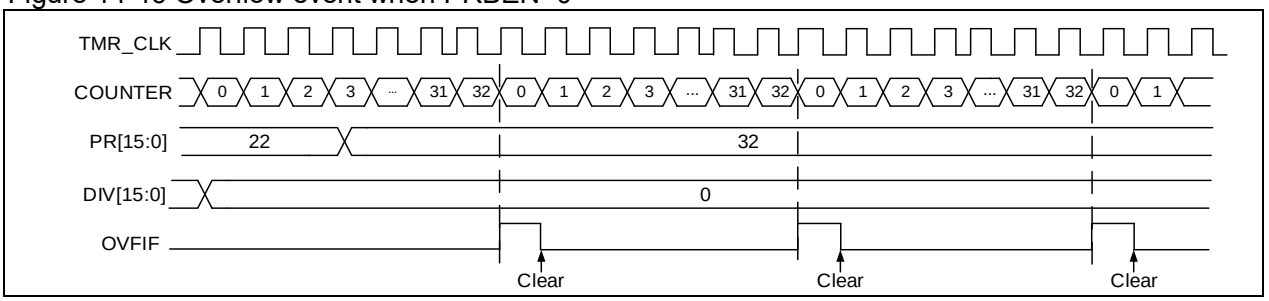
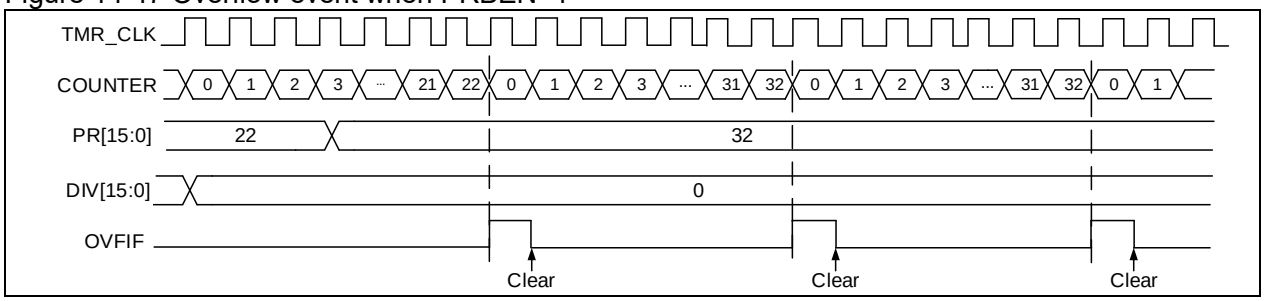


Figure 14-47 Overflow event when PRBEN=1



14.3.3.3 TMR input function

Each timer of TMR9 and TMR12 has two independent channels, while each of TMR10, TMR11, TMR13 and TMR14 has an independent channel. Each channel can be configured as input or output.

As input, each channel input signal is handled as follows:

- TMRx_CHx outputs the pre-processed CxIRAW. Set the C1INSEL bit to select TMRx_CH1 as the source of C1IRAW.
- CxIRAW inputs digital filter and outputs filtered CxIF signal. The digital filter uses the CxDF bit to program sampling frequency and sampling times.
- CxIF inputs edge detector, and outputs the CxIFPx signal after edge selection. The edge selection depends on both CxP and CxCP bits. It is possible to select input rising edge, falling edge or both edges.
- CxIFPx inputs capture signal selector, and outputs the CxIN signal after capture signal selection. The capture signal selection is defined by CxC bit. It is possible to select CxIFPx, CyIFPx or STCI as CxIN source. Of those, CyIFPx (x≠y) is the CyIFPy signal that is from Y channel and processed by channel -x edge detector (for example, C1IFP2 is the channel 1's C1IFP1 signal that passed through channel 2 edge detection). The STCI comes from slave timer controller, and its source is selected by STIS bit.
- CxIN outputs the CxIPS signal that is divided by input channel divider. The divider factor can be defined as No division, /2, /4 or /8, by the CxIDIV bit. It can be used for filtering, selection, division and input capture of input signals.

Figure 14-48 Input/output channel 1 main circuit

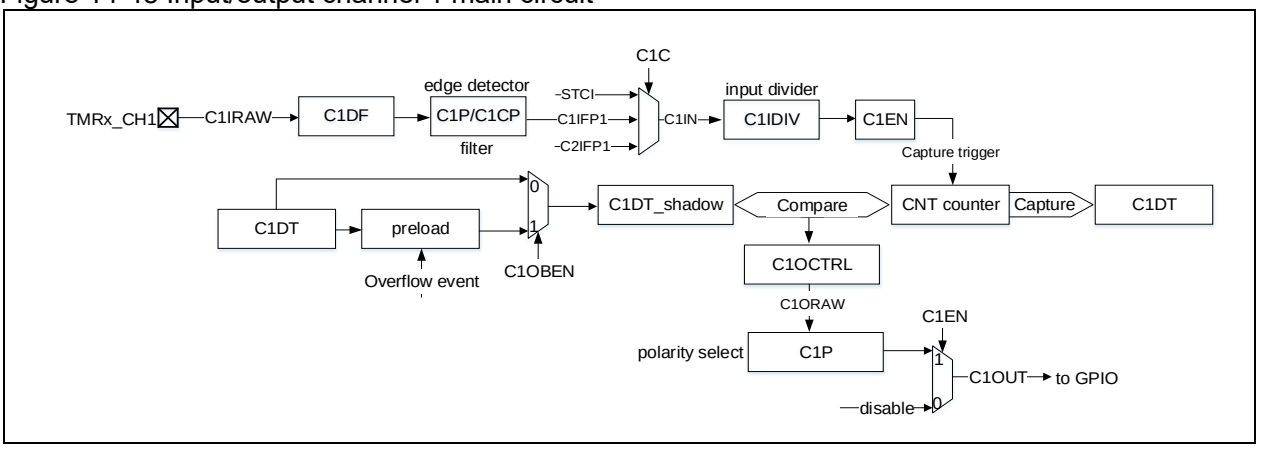
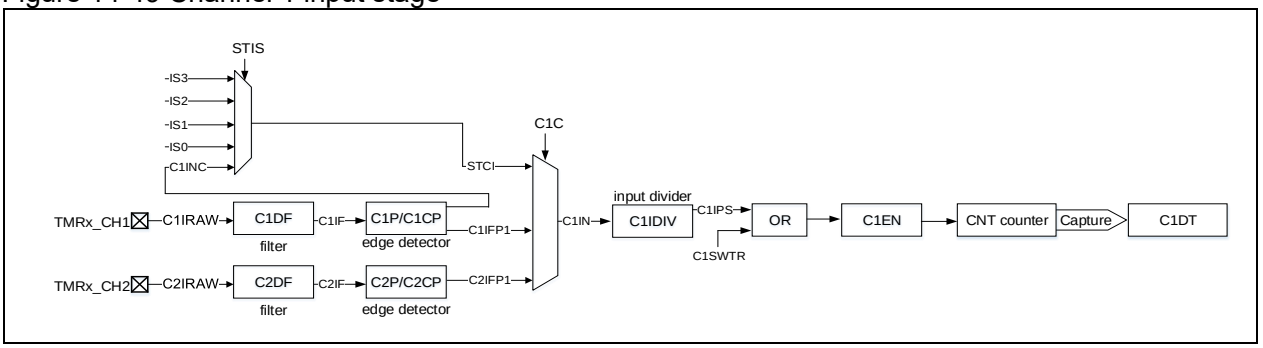


Figure 14-49 Channel 1 input stage



Input mode

In input mode, the TMRx_CxDT registers latch the current counter values after the selected trigger signal is detected, and the capture compare interrupt flag bit (CxIF) is set. An interrupt or a DMA request will be generated if the CxIEN or CxDEN bit is enabled. If the selected trigger signal is detected when the CxIF is set to 1, a capture overflow event occurs. The TMRx_CxDT register overwrites the recorded value with the current counter value, and the CxRF is set to 1.

To capture the rising edge of C1IN input, following the configuration procedure mentioned below:

- Set C1C=01 in the TMRx_CxDT register to select the C1IN as channel 1 input;
- Set the filter bandwidth of C1IN signal (CxDF[3: 0]);
- Set the active edge on the C1IN channel by writing C1P=0 (rising edge) in the TMRx_CCTR Register;
- Program the capture frequency division of C1IN signal (C1DIV[1: 0]);
- Enable channel 1 input capture (C1EN=1);
- If needed, enable the relevant interrupt by setting the C1IEN bit in the TMRx_IDEN register.

PWM input (TMR9/12)

The PWM input mode applies to channel 1 and channel 2. To enable this mode, map the C1IN and C2IN to the same TMRx_CHx, and configure the CxIFPx of channel 1/2 to trigger slave timer controller reset.

The PWM input mode can be used to measure the period and duty cycle of input signal. The period and duty cycle of channel 1 can be measured as follows:

- Set C1C=2‘b01 to set C1IN as C1IFP1;
- Set C1P=1’b0 to set C1IFP1 rising edge active;
- Set C2C=2‘b10 to set C2IN as C1IFP2;
- Set C2P=1’b1 to set C1IFP2 falling edge active;
- Set STIS=3’b101 to set C1IFP1 as the slave timer trigger signal;
- Set SMSEL=3‘b110 to set the slave timer in reset mode;
- Set C1EN=1’b1 and C2EN=1’b1 to enable channel 1 and input capture.

In these configurations, the rising edge of channel 1 input signal triggers capture and saves captured values to the C1DT register, and channel 1 input signal rising edge resets the counter. The falling

edge of channel 1 input signal triggers capture and saves captured values to the C2DT register. The period and duty of channel 1 input signal can be calculated through C1DT and C2DT respectively.

Figure 14-50 Example of PWM input mode configuration

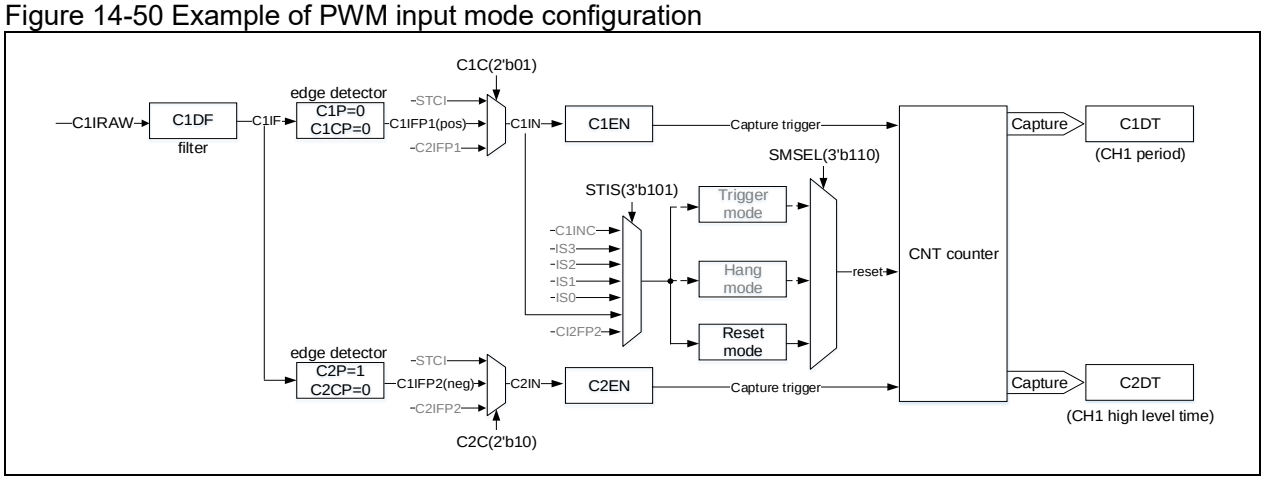
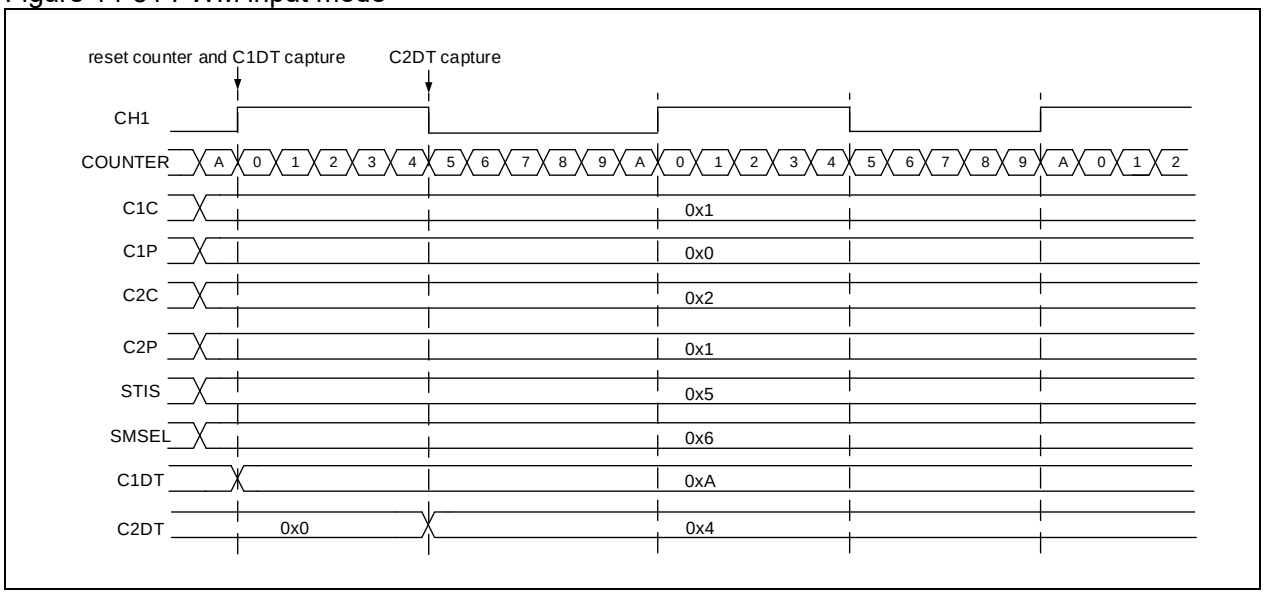


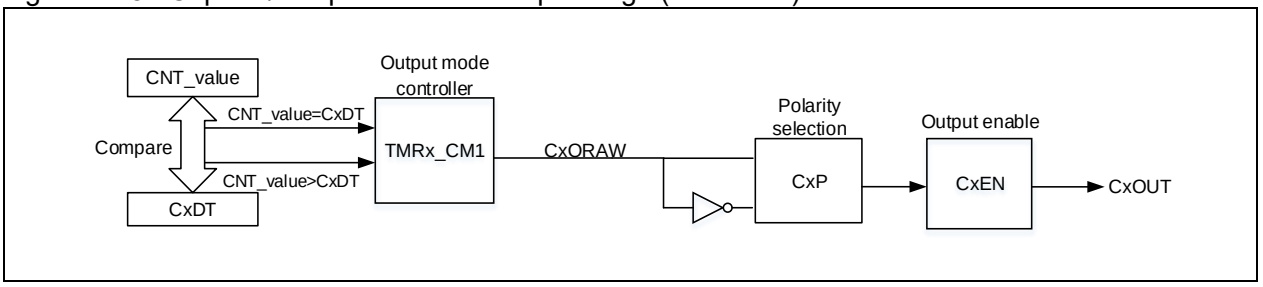
Figure 14-51 PWM input mode



14.3.3.4 TMR output function

The TMR output consists of a comparator and an output controller. It is used to program the period, duty cycle and polarity of the output signal.

Figure 14-52 Capture/compare channel output stage (channel 1)



Output mode

Write CxC[1: 0]≠2'b00 to configure the channel as output to implement multiple output modes. In this case, the counter value is compared with the value in the TMRx_CxDT register, and the intermediate signal CxORAW is generated according to the output mode selected by CxOCTRL[2: 0], which is sent to IO after being processed by the output control circuit. The period of the output signal is configured by the TMRx_PR register, while the duty cycle by the TMRx_CxDT register.

Output compare modes include:

- **PWM mode A**: Set CxOCTRL=3'b110 to enable PWM mode A. In upcounting, when TMRx_C1DT>TMRx_CVAL, C1ORAW outputs high; otherwise, outputs low. In downcounting, when TMRx_C1DT<TMRx_CVAL, C1ORAW outputs low; otherwise, outputs high. To set PWM mode A, the following process is recommended:
  - Set the TMRx_PR register to set PWM period;
  - Set the TMRx_CxDT register to set PWM duty cycle;
  - Set CxOCTRL=3'b110 in TMRx_CM1/CM2 register and set output mode as PWM mode A;
  - Set the TMRx_DIV register to set the counting frequency;
  - Set the TWCMSEL[1:0] bit in the TMRx_CTRL1 to set the count mode;
  - Set the CxP and CxCP bits in the TMRx_CCTRL register to set the output polarity;
  - Set the CxEN and CxCEN bits in the TMRx_CCTRL register to enable channel output;
  - Set the OEN bit in the TMRx_BRK register to enable TMRx output;
  - Set the corresponding GPIO of TMR output channel as the multiplexed mode;
  - Set the TMREN bit in the TMRx_CTRL1 register to enable TMRx counter.
- **PWM mode B**: Set CxOCTRL=3'b111 to enable PWM mode B. In upcounting, when TMRx_C1DT>TMRx_CVAL, C1ORAW outputs low; otherwise, outputs high. In downcounting, when TMRx_C1DT<TMRx_CVAL, C1ORAW outputs high; otherwise, outputs low.
- **Forced output mode:** Set CxOCTRL=3'b100/101 to enable forced output mode. In this case, the CxORAW is forced to be the programmed level, irrespective of the counter value. Despite this, the channel flag bit and DMA request still depend on the compare result.
- **Output compare mode:** Set CxOCTRL=2'b001/010/011 to enable output compare mode. In this case, when the counter value matches the value of the CxDT register, the CxORAW is forced high (CxOCTRL=3'b001), low (CxOCTRL=3'b010) or toggling (CxOCTRL=3'b011).
- **One-pulse mode (TMR9/12 only):** This is a particular case of PWM mode. Set OCMEN=1 to enable one-pulse mode. In this mode, the comparison match is performed in the current counting period. The TMREN bit is cleared as soon as the current counting is completed. Therefore, only one pulse is output. When configured as in upcounting mode, the configuration must follow the rule: CVAL<CxDT≤PR; in downcounting mode, CVAL>CxDT is required.
- **Fast output mode (TMR9/12 only):** Set CxOIEN=1 to enable this mode. If enabled, the CxORAW signal will not change when the counter value matches the CxDT, but at the beginning of the current counting period. In other words, the comparison result is advanced, so the comparison result between the counter value and the TMRx_CxDT register will determine the level of CxORAW in advance.

Figure 53 gives an example of output compare mode (toggle) with C1DT=0x3. When the counter value is equal to 0x3, C1OUT toggles.

Figure 54 gives an example of the combination between upcounting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

Figure 55 gives an example of the combination between upcounting mode and one-pulse PWM mode B. The counter only counts only one cycle, and the output signal sends only one pulse.

Figure 14-53 C1ORAW toggles when counter value matches the C1DT value

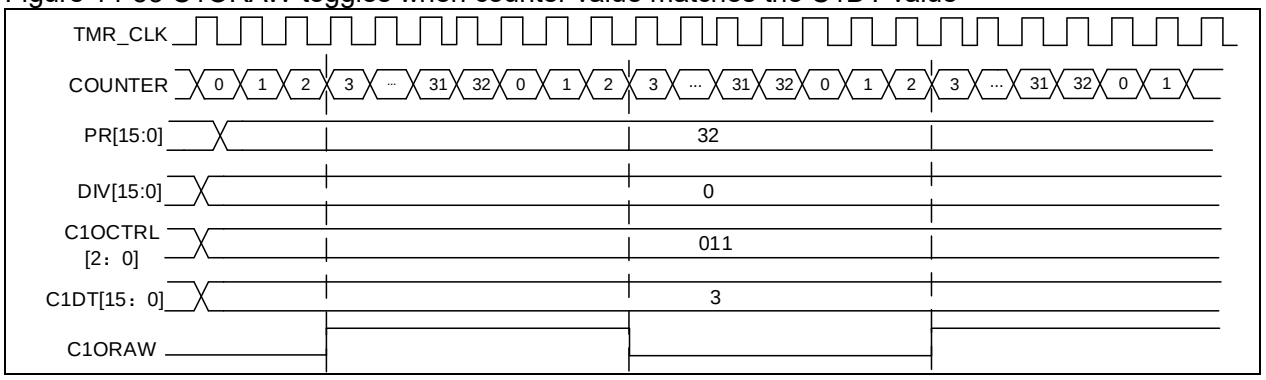


Figure 14-54 Upcounting mode and PWM mode A

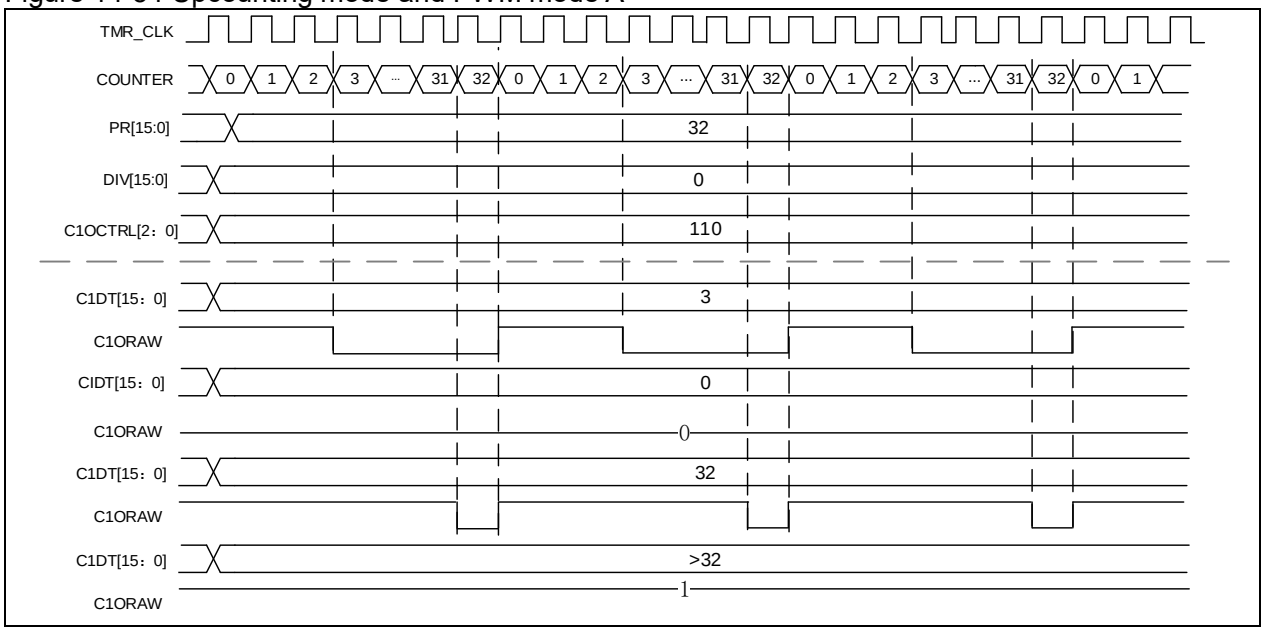
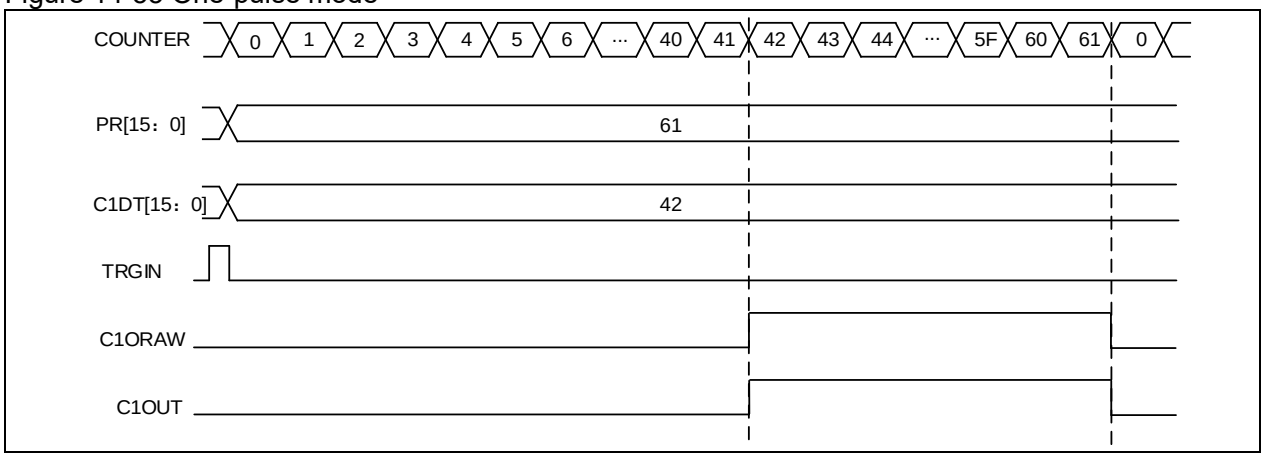


Figure 14-55 One-pulse mode



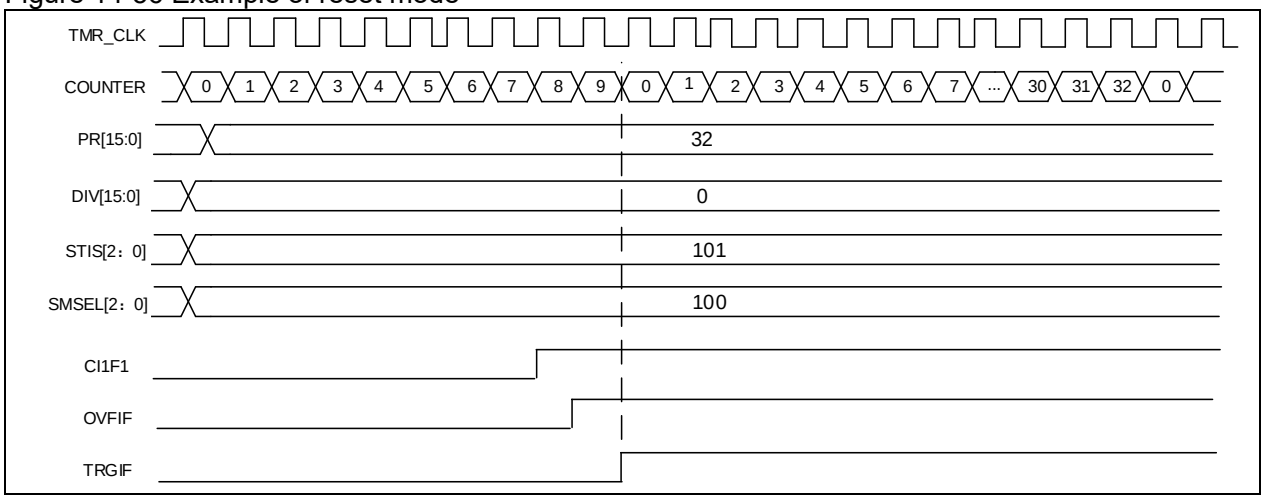
14.3.3.5 TMR synchronization

TMR9 and TMR12 are linked together internally for timer synchronization. Slave timer is selected by setting the SMSEL[2: 0] bit.

Slave mode: Reset mode

The counter and its prescaler can be reset by a selected trigger signal. An overflow event is generated when OVFS=0.

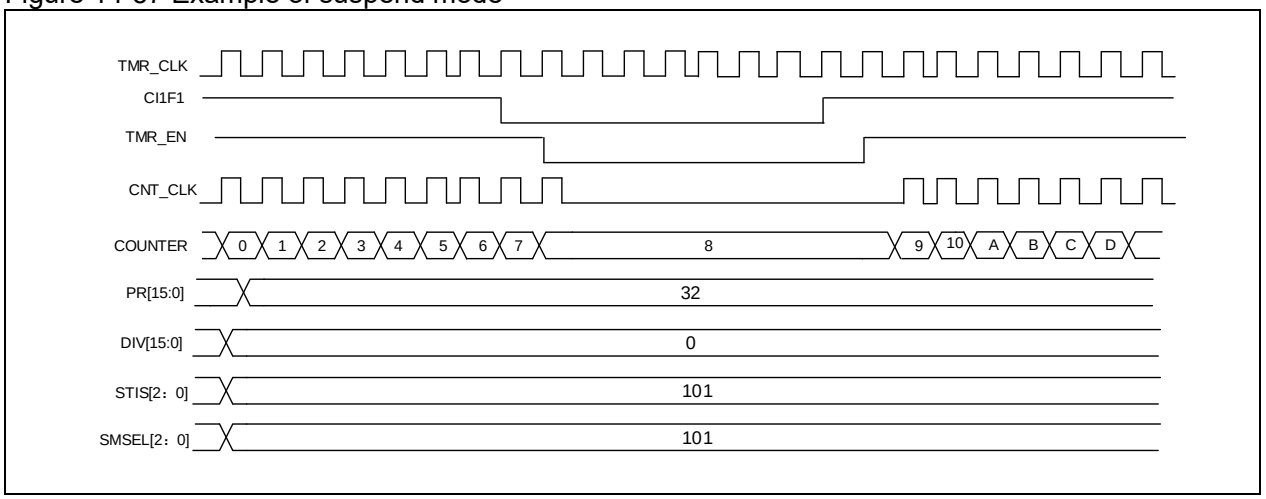
Figure 14-56 Example of reset mode



Slave mode: Suspend mode

In this mode, the counter is controlled by a selected trigger input. The counter starts counting when the trigger input is high and stops as soon as the trigger input is low.

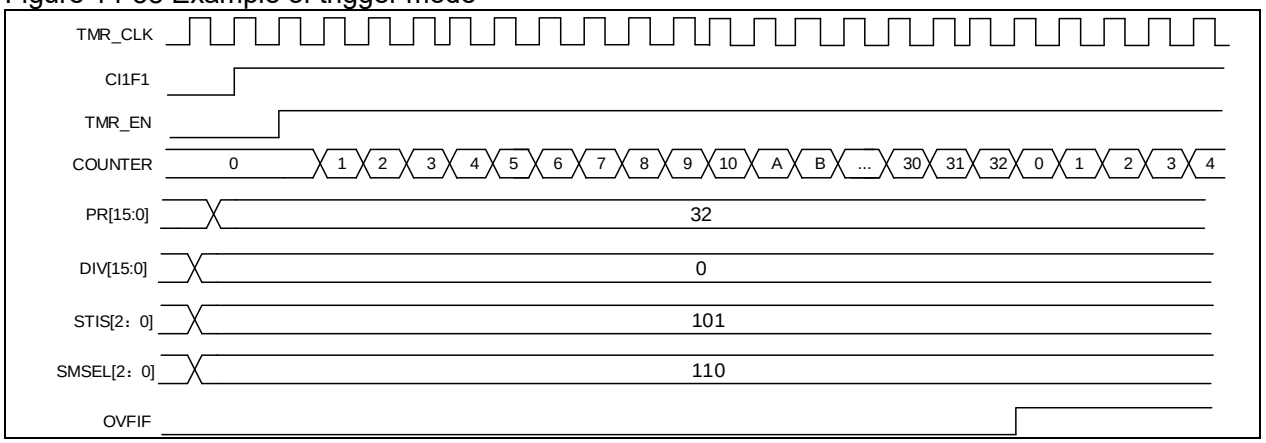
Figure 14-57 Example of suspend mode



Slave mode: Trigger mode

The counter can start counting on the rising edge of a selected trigger input (TMR_EN=1)

Figure 14-58 Example of trigger mode



14.3.3.6 Debug mode

When the microcontroller enters debug mode (Cortex™-M4F core halted), the TMRx counter stops counting by setting the TMRx_PAUSE in the DEBUG module. Refer to Chapter 30.2 for more information.

14.3.4 TMR9 and TMR12 registers

These peripheral registers must be accessed by word (32 bits).
All TMRx register are mapped into a 16-bit addressable space.

Table 14-8 TMRx register map and reset value

Register name	Register	Reset value
TMRx_CTRL1	0x00	0x0000
TMRx_STCTRL	0x08	0x0000
TMRx_IDEN	0x0C	0x0000
TMRx_ISTS	0x10	0x0000
TMRx_SWEVT	0x14	0x0000
TMRx_CM1	0x18	0x0000
TMRx_CCTRL	0x20	0x0000
TMRx_CVAL	0x24	0x0000
TMRx_DIV	0x28	0x0000
TMRx_PR	0x2C	0x0000
TMRx_C1DT	0x34	0x0000 0000
TMRx_C2DT	0x38	0x0000 0000

14.3.4.1 TMR9 and TMR12 control register1 (TMRx_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at its default value
Bit 9: 8	CLKDIV	0x0	rw	Clock divider This field is used to set the ratio between digital filter sampling frequency (f_{DT}) and timer clock frequency (f_{CK_INT}). 00: Normal 01: Divided by 2 10: Divided by 4 11: Reserved
Bit 7	PRBEN	0x0	rw	Period buffer enable 0: Period buffer is disabled 1: Period buffer is enabled
Bit 6: 4	Reserved	0x0	resd	Kept at its default value
Bit 3	OCMEN	0x0	rw	One cycle mode enable

				This bit is use to select whether to stop counting at an update event 0: The counter does not stop at an update event 1: The counter stops at an update event
Bit 2	OVFS	0x0	rw	Overflow event source This bit is used to select overflow event or DMA request sources. 0: Counter overflow, setting the OVFSWTR bit or overflow event generated by slave timer controller 1: Only counter overflow generates an overflow event
Bit 1	OVFEN	0x0	rw	Overflow event enable 0: Enabled 1: Disabled
Bit 0	TMREN	0x0	rw	TMR enable 0: Enabled 1: Disabled

14.3.4.2 TMR9 and TMR12 Slave timer control register (TMRx_STCTRL)

Bit	Register	Reset value	Type	Description
Bit 15:7	Reserved	0x000	resd	Kept at its default value
Bit 6: 4	STIS	0x0	rw	Subordinate TMR input selection This field is used to select the subordinate TMR input. 000: Internal selection 0 (IS0) 001: Internal selection 1 (IS1) 010: Internal selection 2 (IS2) 011: Internal selection 3 (IS3) 100: C1IRAW input detector (C1INC) 101: Filtered input 1 (C1IF1) 110: Filtered input 2 (C1IF2) 111: Reserved Please refer to Table 14-7 for more information on ISx for each timer.
Bit 3	Reserved	0x0	resd	Kept at its default value
Bit 2: 0	SMSEL	0x0	rw	Subordinate TMR mode selection 000: Slave mode is disabled 001: Encoder mode A 010: Encoder mode B 011: Encoder mode C 100: Reset mode —Rising edge of the TRGIN input reinitializes the counter 101: Suspend mode — The counter starts counting when the TRGIN is high 110: Trigger mode — A trigger event is generated at the rising edge of the TRGIN input 111: External clock mode A — Rising edge of the TRGIN input clocks the counter Note: Please refer to count mode section for details on encoder mode A/B/C.

14.3.4.3 TMR9 and TMR12 DMA/interrupt enable register (TMRx_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15:7	Reserved	0x0	resd	Kept at its default value.
Bit 6	TIEN	0x0	rw	Trigger interrupt enable 0: Disabled 1: Enabled
Bit 5:3	Reserved	0x0	resd	Kept at its default value.
Bit 2	C2IEN	0x0	rw	Channel 2 interrupt enable 0: Disabled 1: Enabled
Bit 1	C1IEN	0x0	rw	Channel 1 interrupt enable 0: Disabled

				1: Enabled
Bit 0	OVFIEN	0x0	rw	Overflow interrupt enable 0: Disabled 1: Enabled

14.3.4.4 TMR9 and TMR12 interrupt status register (TMRx_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 11	Reserved	0x0	resd	Kept at its default value.
Bit 10	C2RF	0x0	rw0c	Channel 2 recapture flag Please refer to C1RF description.
Bit 9	C1RF	0x0	rw0c	Channel 1 recapture flag This bit indicates whether a recapture is detected when C1IF=1. This bit is set by hardware, and cleared by writing “0”. 0: No capture is detected 1: Capture is detected.
Bit 8: 7	Reserved	0x0	resd	Kept at its default value.
Bit 6	TRGIF	0x0	rw0c	Trigger interrupt flag This bit is set by hardware on a trigger event. It is cleared by writing “0”. 0: No trigger event occurs 1: Trigger event is generated. Trigger event: an active edge is detected on TRGIN input, or any edge in suspend mode.
Bit 5:3	Reserved	0x0	resd	Kept at its default value.
Bit 2	C2IF	0x0	rw0c	Channel 2 interrupt flag Please refer to C1IF description.
Bit 1	C1IF	0x0	rw0c	Channel 1 interrupt flag If the channel 1 is configured as input mode: This bit is set by hardware on a capture event. It is cleared by software or read access to the TMRx_C1DT 0: No capture event occurs 1: Capture event is generated If the channel 1 is configured as output mode: This bit is set by hardware on a compare event. It is cleared by software. 0: No compare event occurs 1: Compare event is generated
Bit 0	OVFIF	0x0	rw0c	Overflow interrupt flag This bit is set by hardware on an overflow event. It is cleared by software. 0: No overflow event occurs 1: Overflow event is generated.

14.3.4.5 TMR9 and TMR12 software event register (TMRx_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 7	Reserved	0x000	resd	Kept at its default value.
Bit 6	TRGSWTR	0x0	rw	Trigger event triggered by software This bit is set by software to generate a trigger event. 0: No effect 1: Generate a trigger event.
Bit 5:3	Reserved	0x0	resd	Kept at its default value.
Bit 2	C2SWTR	0x0	wo	Channel 2 event triggered by software Please refer to C1M description
Bit 1	C1SWTR	0x0	wo	Channel 1 event triggered by software This bit is set by software to generate a channel 1 event. 0: No effect 1: Generate a channel 1 event.
Bit 0	OVFSWTR	0x0	wo	Overflow event triggered by software This bit is set by software to generate an overflow event. 0: No effect 1: Generate an overflow event.

14.3.4.6 TMR9 and TMR12 channel mode register1 (TMRx_CM1)

The channel can be used in input (capture mode) or output (compare mode). The direction of a channel is defined by the corresponding CxC bits. All the other bits of this register have different functions in input and output modes. The CxOx describes its function in output mode when the channel is in output mode, while the CxIx describes its function in output mode when the channel is in input mode. Attention must be given to the fact that the same bit can have different functions in input mode and output mode.

Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14: 12	C2OCTRL	0x0	rw	Channel 2 output control
Bit 11	C2OBEN	0x0	rw	Channel 2 output buffer enable
Bit 10	C2OIEN	0x0	rw	Channel 2 output enable immediately
Bit 9: 8	C2C	0x0	rw	Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0': 00: Output 01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS register.
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6: 4	C1OCTRL	0x0	rw	Channel 1 output control This field defines the behavior of the original signal C1ORAW. 000: Disconnected. C1ORAW is disconnected from C1OUT; 001: C1ORAW is high when TMRx_CVAL=TMRx_C1DT 010: C1ORAW is low when TMRx_CVAL=TMRx_C1DT 011: Switch C1ORAW level when TMRx_CVAL=TMRx_C1DT 100: C1ORAW is forced low 101: C1ORAW is forced high. 110: PWM mode A – OWCDIR=0, C1ORAW is high once TMRx_C1DT>TMRx_CVAL, else low; – OWCDIR=1, C1ORAW is low once TMRx_ C1DT <TMRx_CVAL, else high; 111: PWM mode B – OWCDIR=0, C1ORAW is low once TMRx_ C1DT >TMRx_CVAL, else high; – OWCDIR=1, C1ORAW is high once TMRx_ C1DT <TMRx_CVAL, else low. *Note: In the configurations other than 000', the C1OUT is connected to C1ORAW. The C1OUT output level is not only subject to the changes of C1ORAW, but also the output polarity set by CCTRL.*
Bit 3	C1OBEN	0x0	rw	Channel 1 output buffer enable 0: Buffer function of TMRx_C1DT is disabled. The new value written to the TMRx_C1DT takes effect immediately. 1: Buffer function of TMRx_C1DT is enabled. The value to be written to the TMRx_C1DT is stored in the buffer register, and can be sent to the TMRx_C1DT register only on an overflow event.
Bit 2	C1OIEN	0x0	rw	Channel 1 output enable immediately In PWM mode A or B, this bit is used to accelerate the channel 1 output's response to the trigger event. 0: Need to compare the CVAL with C1DT before generating an output

				1: No need to compare the CVAL and C1DT. An output is generated immediately when a trigger event occurs.
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

Input capture mode:

Bit	Register	Reset value	Type	Description
Bit 15: 12	C2DF	0x0	rw	Channel 2 digital filter
Bit 11: 10	C2IDIV	0x0	rw	Channel 2 input divider
Bit 9: 8	C2C	0x0	rw	Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0': 00: Output 01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7: 4	C1DF	0x0	rw	Channel 1 digital filter This field defines the digital filter of the channel 1. N stands for the number of filtering, indicating that the input edge can pass the filter only after N sampling events. 0000: No filter, sampling is done at f_{DTS} 1000: $f_{SAMPLING}=f_{DTS}/8$, N=6 0001: $f_{SAMPLING}=f_{CK_INT}$, N=2 1001: $f_{SAMPLING}=f_{DTS}/8$, N=8 0010: $f_{SAMPLING}=f_{CK_INT}$, N=4 1010: $f_{SAMPLING}=f_{DTS}/16$, N=5 0011: $f_{SAMPLING}=f_{CK_INT}$, N=8 1011: $f_{SAMPLING}=f_{DTS}/16$, N=6 0100: $f_{SAMPLING}=f_{DTS}/2$, N=6 1100: $f_{SAMPLING}=f_{DTS}/16$, N=8 0101: $f_{SAMPLING}=f_{DTS}/2$, N=8 1101: $f_{SAMPLING}=f_{DTS}/32$, N=5 0110: $f_{SMPLING}=f_{DTS}/4$, N=6 1110: $f_{SAMPLING}=f_{DTS}/32$, N=6 0111: $f_{SAMPLING}=f_{DTS}/4$, N=8 1111: $f_{SAMPLING}=f_{DTS}/32$, N=8
Bit 3: 2	C1IDIV	0x0	rw	Channel 1 input divider This field defines Channel 1 input divider. 00: No divider. An input capture is generated at each active edge. 01: An input compare is generated every 2 active edges 10: An input compare is generated every 4 active edges 11: An input compare is generated every 8 active edges Note: the divider is reset once C1EN='0'
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

14.3.4.7 TMR9 and TMR12 channel control register (TMRx_CCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	C2P	0x0	rw	Channel 2 polarity Please refer to C1P description.
Bit 4	C2EN	0x0	rw	Channel 2 enable Please refer to C1EN description.
Bit 3: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	C1P	0x0	rw	Channel 1 polarity When the channel 1 is configured as output mode: 0: C1OUT is active high 1: C1OUT is active low When the channel 1 is configured as input mode: 0: C1IN active edge is on its rising edge. When used as external trigger, C1IN is not inverted. 1: C1IN active edge is on its falling edge. When used as external trigger, C1IN is inverted.
Bit0	C1EN	0x0	rw	Channel 1 enable 0: Input or output is disabled 1: Input or output is enabled

Table 14-9 Standard CxOUT channel output control bit

CxEN bit	CxOUT output state
0	Output disabled (CxOUT=0)
1	CxOUT = CxORAW + polarity

Note: The state of the external I/O pins connected to the standard CxOUT channel depends on the CxOUT channel state and the GPIO and IOMUX registers.

14.3.4.8 TMR9 and TMR12 counter value (TMRx_CVAL)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0000	rw	Counter value

14.3.4.9 TMR9 and TMR12 division value (TMRx_DIV)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DIV	0x0000	rw	Divider value The counter clock frequency $f_{CK_CNT} = f_{TMR_CLK}$ /(DIV[15: 0]+1). DIV contains the value written at an overflow event.

14.3.4.10 TMR9 and TMR12 period register (TMRx_PR)

Bit	Register	Reset value	Type	Description
Bit 15: 0	PR	0x0000	rw	Period value This defines the period value of the TMRx counter. The timer stops working when the period value is 0.

14.3.4.11 TMR9 and TMR12 channel 1 data register (TMRx_C1DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	C1DT	0x0000	rw	Channel 1 data register When the channel 1 is configured as input mode: The C1DT is the CVAL value stored by the last channel 1 input event (C1IN) When the channel 1 is configured as output mode: C1DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C1OBEN bit, and the corresponding output is generated on C1OUT as configured.

14.3.4.12 TMR9 and TMR12 channel 2 data register (TMRx_C2DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	C2DT	0x0000	resd	Kept at its default value.
Bit 15: 0	C2DT	0x0000	rw	Channel 2 data register When the channel 2 is configured as input mode: The C2DT is the CVAL value stored by the last channel 2 input event (C1IN) When the channel 2 is configured as output mode: C2DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C2OBEN bit, and the corresponding output is generated on C2OUT as configured.

14.3.5 TMR10, TMR11, TMR13 and TMR14 registers

These peripheral registers must be accessed by word (32 bits).

All TMRx register are mapped into a 1-bit addressable space.

Table 14-10 TMRx register map and reset value

Register	Offset	Reset value
TMRx_CTRL1	0x00	0x0000
TMRx_IDEN	0x0C	0x0000
TMRx_ISTS	0x10	0x0000
TMRx_SWEVT	0x14	0x0000
TMRx_CM1	0x18	0x0000
TMRx_CCTRL	0x20	0x0000
TMRx_CVAL	0x24	0x0000
TMRx_DIV	0x28	0x0000
TMRx_PR	0x2C	0x0000
TMRx_C1DT	0x34	0x0000

14.3.5.1 TMR10, TMR11, TMR13 and TMR14 control register1 (TMRx_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at its default value
Bit 9: 8	CLKDIV	0x0	rw	Clock divider This field is used to set the ratio between digital filter sampling frequency (f_{DT}) and timer clock frequency (f_{CK_INT}). 00: Normal 01: Divided by 2 10: Divided by 4 11: Reserved
Bit 7	PRBEN	0x0	rw	Period buffer enable 0: Period buffer is disabled 1: Period buffer is enabled
Bit 6: 4	Reserved	0x0	resd	Default value
Bit 3	OCMEN	0x0	rw	One cycle mode enable This bit is use to select whether to stop counting at an overflow event 0: The counter does not stop at an overflow event 1: The counter stops at an overflow event
Bit 2	OVFS	0x0	rw	Overflow event source This bit is used to select overflow event or DMA request sources. 0: Counter overflow, setting the OVFSWTR bit or overflow

				event generated by slave timer controller 1: Only counter overflow generates an overflow event
Bit 1	OVFEN	0x0	rw	Overflow event enable 0: Enabled 1: Disabled
Bit 0	TMREN	0x0	rw	TMR enable 0: Enabled 1: Disabled

14.3.5.2 TMR10, TMR11, TMR13 and TMR14 DMA/interrupt enable register (TMRx_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15:2	Reserved	0x0	resd	Kept at its default value
Bit 1	C1IEN	0x0	rw	Channel 1 interrupt enable 0: Disabled 1: Enabled
Bit 0	OVFIEN	0x0	rw	Overflow interrupt enable 0: Disabled 1: Enabled

14.3.5.3 TMR10, TMR11, TMR13 and TMR14 interrupt status register (TMRx_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x0	resd	Kept at its default value.
Bit 9	C1RF	0x0	rw0c	Channel 1 recapture flag This bit indicates whether a recapture is detected when C1IF=1. This bit is set by hardware, and cleared by writing “0”. 0: No capture is detected 1: Capture is detected.
Bit 8: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	C1IF	0x0	rw0c	Channel 1 interrupt flag If the channel 1 is configured as input mode: This bit is set by hardware on a capture event. It is cleared by software or read access to the TMRx_C1DT 0: No capture event occurs 1: Capture event is generated If the channel 1 is configured as output mode: This bit is set by hardware on a compare event. It is cleared by software. 0: No compare event occurs 1: Compare event is generated
Bit 0	OVFIF	0x0	rw0c	Overflow interrupt flag This bit is set by hardware on an overflow event. It is cleared by software. 0: No overflow event occurs 1: Overflow event is generated.

14.3.5.4 TMR10, TMR11, TMR13 and TMR14 software event register (TMRx_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 2	Reserved	0x000	resd	Kept at its default value.
Bit 1	C1SWTR	0x0	wo	Channel 1 event triggered by software This bit is set by software to generate a channel 1 event. 0: No effect 1: Generate a channel 1 event.
Bit 0	OVFSWTR	0x0	wo	Overflow event triggered by software This bit is set by software to generate an overflow event. 0: No effect 1: Generate an overflow event.

14.3.5.5 TMR10, TMR11, TMR13 and TMR14 channel mode register1 (TMRx_CM1)

The channel can be used in input (capture mode) or output (compare mode). The direction of a channel is defined by the corresponding CxC bits. All the other bits of this register have different functions in input and output modes. The CxOx describes its function in output mode when the channel is in output mode, while the CxIx describes its function in output mode when the channel is in input mode. Attention must be given to the fact that the same bit can have different functions in input mode and output mode.

Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15:7	Reserved	0x0	resd	Kept at its default value.
Bit 6: 4	C1OCTRL	0x0	rw	Channel 1 output control This field defines the behavior of the original signal C1ORAW. 000: Disconnected. C1ORAW is disconnected from C1OUT; 001: C1ORAW is high when TMRx_CVAL=TMRx_C1DT 010: C1ORAW is low when TMRx_CVAL=TMRx_C1DT 011: Switch C1ORAW level when TMRx_CVAL=TMRx_C1DT 100: C1ORAW is forced low 101: C1ORAW is forced high. 110: PWM mode A – OWCDIR=0, C1ORAW is high once TMRx_C1DT>TMRx_CVAL, else low; – OWCDIR=1, C1ORAW is low once TMRx_ C1DT <TMRx_CVAL, else high; 111: PWM mode B – OWCDIR=0, C1ORAW is low once TMRx_ C1DT >TMRx_CVAL, else high; – OWCDIR=1, C1ORAW is high once TMRx_ C1DT <TMRx_CVAL, else low. *Note: In the configurations other than 000', the C1OUT is connected to C1ORAW. The C1OUT output level is not only subject to the changes of C1ORAW, but also the output polarity set by CCTRL.*
Bit 3	C1OBEN	0x0	rw	Channel 1 output buffer enable 0: Buffer function of TMRx_C1DT is disabled. The new value written to the TMRx_C1DT takes effect immediately. 1: Buffer function of TMRx_C1DT is enabled. The value to be written to the TMRx_C1DT is stored in the buffer register, and can be sent to the TMRx_C1DT register only on an overflow event.
Bit 2	C1OIEN	0x0	rw	Channel 1 output enable immediately In PWM mode A or B, this bit is used to accelerate the channel 1 output's response to the trigger event. 0: Need to compare the CVAL with C1DT before generating an output 1: No need to compare the CVAL and C1DT. An output is generated immediately when a trigger event occurs.
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Reserved 11: Reserved

Input capture mode:

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7: 4	C1DF	0x0	rw	Channel 1 digital filter This field defines the digital filter of the channel 1. N stands for the number of filtering, indicating that the input edge can pass the filter only after N sampling events. 0000: No filter, sampling is done at f_{DTS} 1000: $f_{SAMPLING}=f_{DTS}/8$, N=6 0001: $f_{SAMPLING}=f_{CK_INT}$, N=2 1001: $f_{SAMPLING}=f_{DTS}/8$, N=8 0010: $f_{SAMPLING}=f_{CK_INT}$, N=4 1010: $f_{SAMPLING}=f_{DTS}/16$, N=5 0011: $f_{SAMPLING}=f_{CK_INT}$, N=8 1011: $f_{SAMPLING}=f_{DTS}/16$, N=6 0100: $f_{SAMPLING}=f_{DTS}/2$, N=6 1100: $f_{SAMPLING}=f_{DTS}/16$, N=8 0101: $f_{SAMPLING}=f_{DTS}/2$, N=8 1101: $f_{SAMPLING}=f_{DTS}/32$, N=5 0110: $f_{SMPLING}=f_{DTS}/4$, N=6 1110: $f_{SAMPLING}=f_{DTS}/32$, N=6 0111: $f_{SAMPLING}=f_{DTS}/4$, N=8 1111: $f_{SAMPLING}=f_{DTS}/32$, N=8
Bit 3: 2	C1IDIV	0x0	rw	Channel 1 input divider This field defines Channel 1 input divider. 00: No divider. An input capture is generated at each active edge. 01: An input compare is generated every 2 active edges 10: An input compare is generated every 4 active edges 11: An input compare is generated every 8 active edges Note: the divider is reset once C1EN='0'
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Reserved 11: Reserved

14.3.5.6 TMR10, TMR11, TMR13 and TMR14 channel control register (TMRx_CCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	C1P	0x0	rw	Channel 1 polarity When the channel 1 is configured as output mode: 0: C1OUT is active high 1: C1OUT is active low When the channel 1 is configured as input mode: 0: C1IN active edge is on its rising edge. When used as external trigger, C1IN is not inverted. 1: C1IN active edge is on its falling edge. When used as external trigger, C1IN is inverted.
Bit0	C1EN	0x0	rw	Channel 1 enable 0: Input or output is disabled 1: Input or output is enabled

Table 14-11 Standard CxOUT channel output control bit

CxEN bit	CxOUT output state
0	Output disabled (CxOUT=0)
1	CxOUT = CxORAW + polarity

Note: The state of the external I/O pins connected to the standard CxOUT channel depends on the CxOUT channel state and the GPIO and IOMUX registers.

14.3.5.7 TMR10, TMR11, TMR13 and TMR14 counter value (TMRx_CVAL)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0000	rw	Counter value

14.3.5.8 TMR10, TMR11, TMR13 and TMR14 division value (TMRx_DIV)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DIV	0x0000	rw	Divider value The counter clock frequency $f_{CK_CNT} = f_{TMR_CLK}$ /(DIV[15: 0]+1). DIV contains the value written at an overflow event.

14.3.5.9 TMR10, TMR11, TMR13 and TMR14 period register (TMRx_PR)

Bit	Register	Reset value	Type	Description
Bit 15: 0	PR	0x0000	rw	Period value This defines the period value of the TMRx counter. The timer stops working when the period value is 0.

14.3.5.10 TMR10, TMR11, TMR13 and TMR14 channel 1 data register (TMRx_C1DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C1DT	0x0000	rw	Channel 1 data register When the channel 1 is configured as input mode: The C1DT is the CVAL value stored by the last channel 1 input event (C1IN) When the channel 1 is configured as output mode: C1DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C1OBEN bit, and the corresponding output is generated on C1OUT as configured.

14.4 Advanced-control timers (TMR1, TMR8 and TMR20)

14.4.1 TMR1, TMR8 and TMR20 introduction

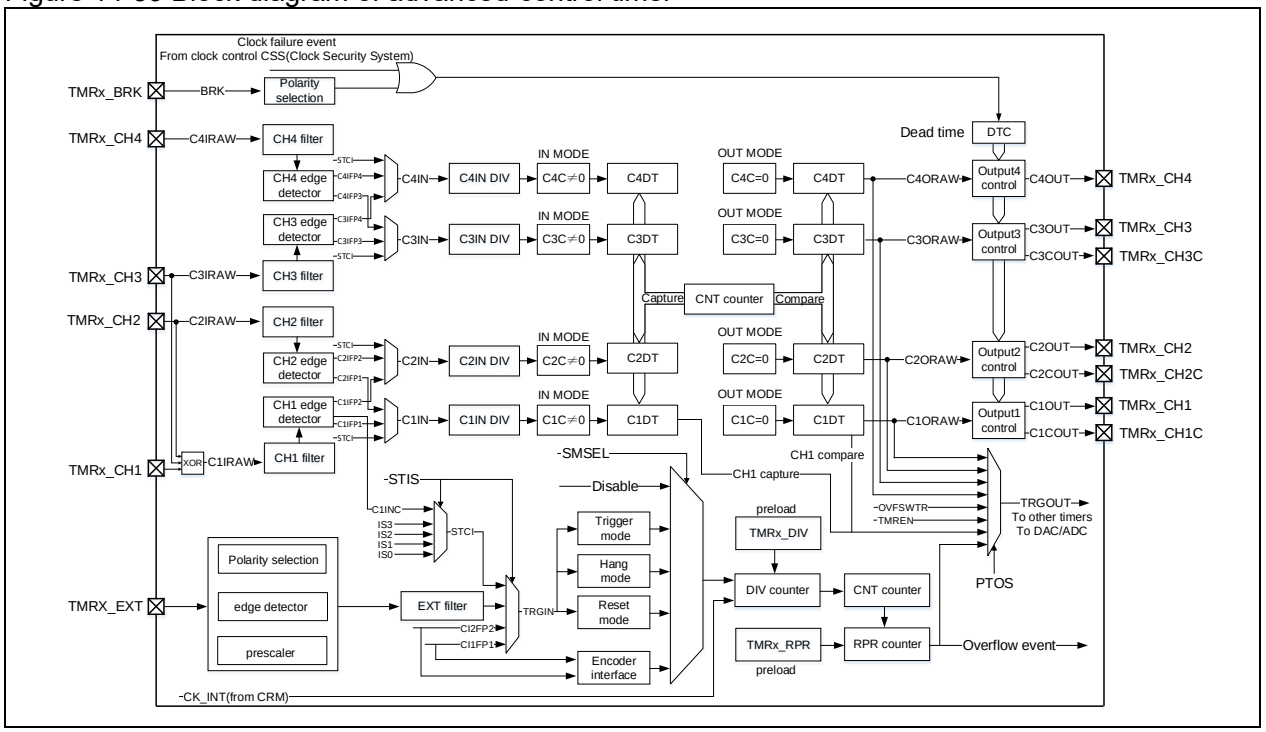
Each of the advanced-control timer (TMR1, TMR8 and TMR20) consists of a 16-bit counter supporting up and down counting modes, four capture/compare registers, and four independent channels to achieve embedded dead-time, input capture and programmable PWM output.

14.4.2 TMR1, TMR8 and TMR20 main features

The main functions of general-purpose TMR1, TMR8 and TMR20 include:

- Source of counter clock: internal clock, external clock an internal trigger input
- 16-bit up, down, up/down, repetition and encoder mode counter
- Five independent channels for input capture, output compare, PWM generation, one-pulse mode output and embedded dead-time
- Three independent channels for complementary output
- TMR break function
- Synchronization control between master and slave timers
- Interrupt/DMA is generated at overflow event, trigger event, break signal input and channel event
- Support TMR burst DMA transfer

Figure 14-59 Block diagram of advanced-control timer

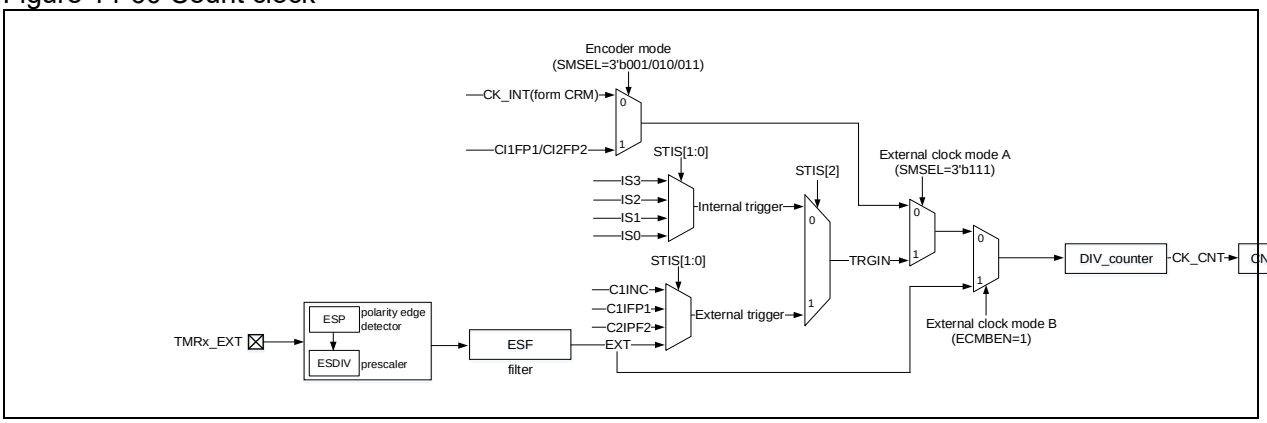


14.4.3 TMR1, TMR8 and TMR20 functional overview

14.4.3.1 Count clock

The count clock of TMR1, TMR8 and TMR20 can be provided by the internal clock (CK_INT), external clock (external clock mode A and B) and internal trigger input (ISx).

Figure 14-60 Count clock

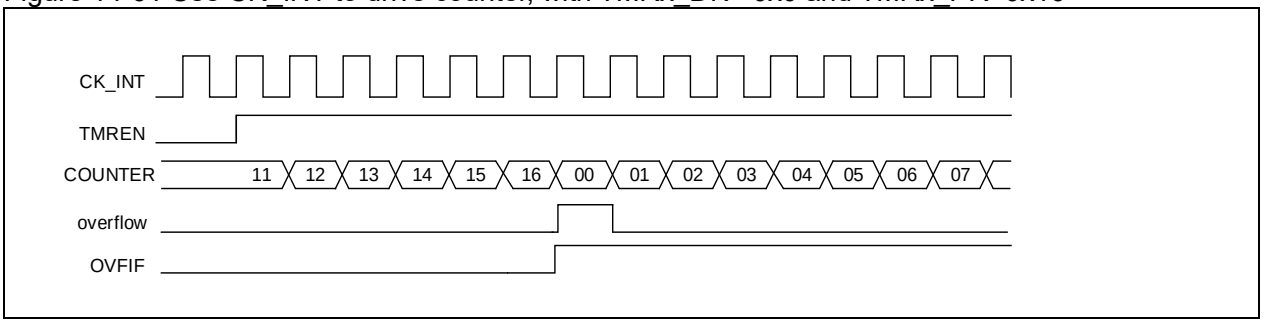


Internal clock (CK_INT)

By default, the CK_INT divided by the prescaler is used to drive the counter to start counting. The configuration process is as follows:

- Configure the CLKDIV[1:0] bit in the TMRx_CTRL1 register and set CK_INT frequency.
- Configure the TWCMSEL[1:0] bit in the TMRx_CTRL1 register and select count mode. If the one-way count direction is set, configure OWCDIR bit in the TMRx_CTRL1 register to select the specific direction.
- Configure the TMRx_DIV register and set counting frequency.
- Configure the TMRx_PR register and set counting period.
- Configure the TMREN bit in the TMRx_CTRL1 register and enable the counter.

Figure 14-61 Use CK_INT to drive counter, with TMRx_DIV=0x0 and TMRx_PR=0x16



External clock (TRGIN/EXT)

The counter clock can be provided by two external clock sources, namely, TRGIN and EXT signals.

SMSEL=3'b111: External clock mode A is selected. Select an external clock source TRGIN signal by setting the STIS[2:0] bit to drive the counter to start counting. The external clock sources include: C1INC (STIS=3'b100, channel 1 rising edge and falling edge), C1IFP1 (STIS=3'b101, channel 1 signal with filtering and polarity selection), C2IFP2 (STIS=3'b110, channel 2 signal with filtering and polarity selection) and EXT (STIS=3'b111, external input signal with polarity selection, frequency division and filtering).

ECMBEN=1: External clock mode B is selected. The counter is driven by external input that has gone through polarity selection, frequency division and filtering. The external clock mode B is equivalent to the external clock mode A which selects EXT signal as an external force TRGIN.

To use external clock mode A, follow the steps below:

- Set external source TRGIN parameters

  If the TMRx_CH1 is used as a source of TRGIN, it is necessary to configure channel 1 input filter (C1DF[3:0] in TMRx_CM1 register) and channel 1 input polarity (C1P/C1CP in TMRx_CCTRL register);

  If the TMRx_CH2 is used as source of TRGIN, it is necessary to configure channel 1 input filter (C2DF[3:0] in TMRx_CM1 register) and channel 2 input polarity (C2P/C2CP in TMRx_CCTR register);

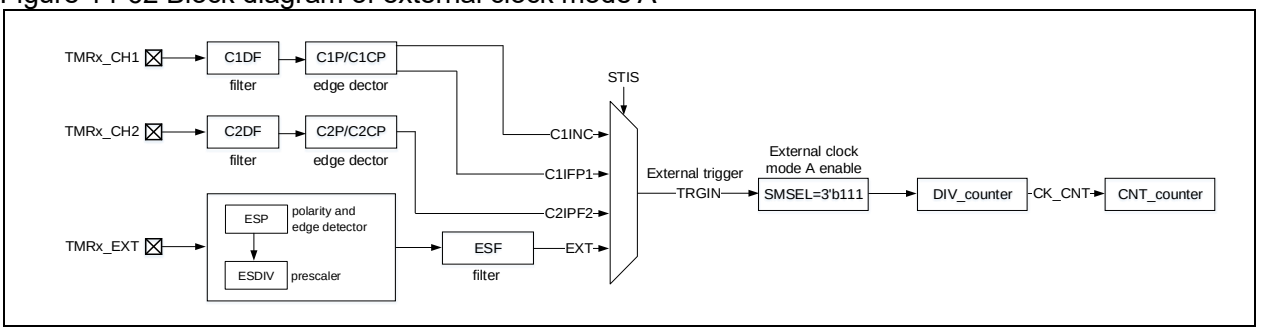
If the TMRx_EXT is used as a source of TRGIN, it is necessary to configure the external signal polarity (ESP in TMRx_STCTRL register), external signal frequency division (ESDIV[1:0] in TMRx_STCTRL) and external signal filter (ESF[3:0] in TMRx_STCTRL register).

- Set TRGIN signal source using the STIS[1:0] bit in TMRx_STCTRL register
- Enable external clock mode A by setting SMSEL=3'b111 in TMRx_STCTR register
- Set counting frequency through the DIV[15:0] in TMRx_DIV register
- Set counting period through the PR[15:0] in TMRx_PR register
- Enable counter through the TMREN bit in TMRx_CTRL1 register

To use external clock mode B, follow the steps below:

- Set external signal polarity through the ESP bit in TMRx_STCTRL register
- Set external signal frequency division through the ESDIV[1:0] bit in TMRx_STCTRL register
- Set external signal filter through the ESF[3:0] bit in TMRx_STCTRL register
- Enable external clock mode B through the ECMBEN bit in TMRx_STCTR register
- Set counting frequency through the DIV[15:0] bit in TMRx_DIV register
- Set counting period through the PR[15:0] bit in TMRx_PR register
- Enable counter through the TMREN in TMRx_CTRL1 register

Figure 14-62 Block diagram of external clock mode A



Note: The delay between the signal on the input side and the actual clock of the counter is due to the synchronization circuit.

Figure 14-63 Counting in external clock mode A, with PR=0x32 and DIV=0x0

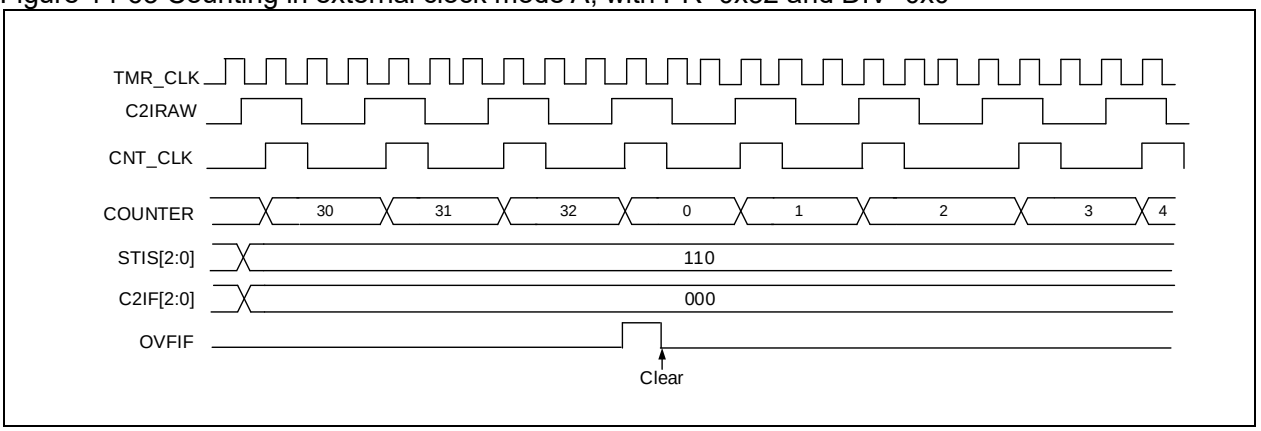
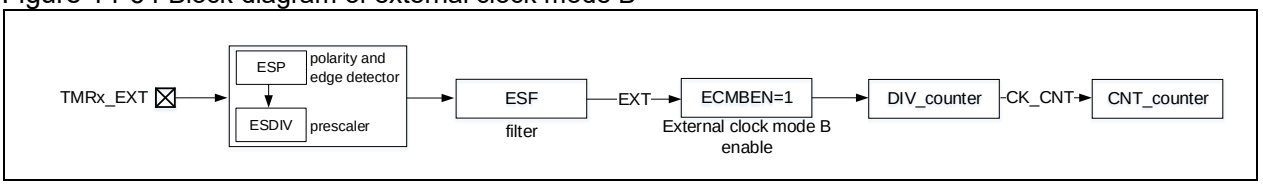
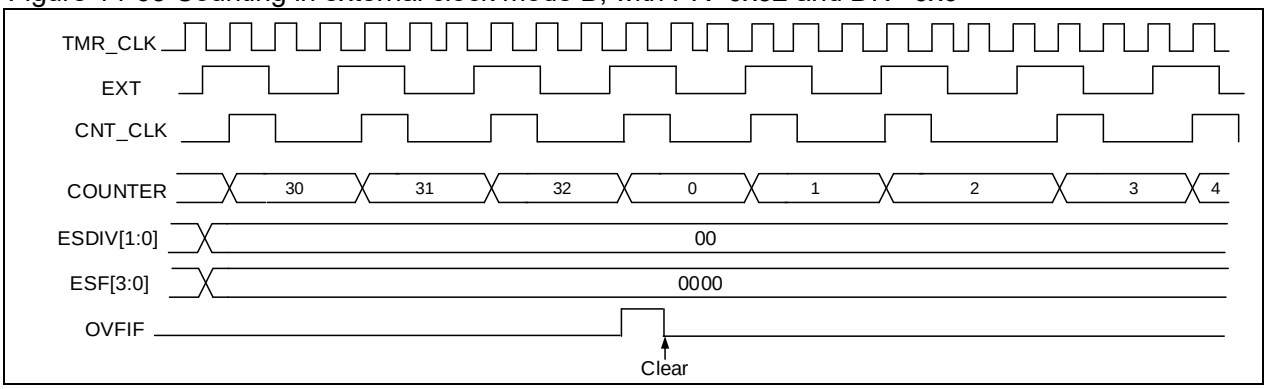


Figure 14-64 Block diagram of external clock mode B



Note: The delay between the signal on the input side and the actual clock of the counter is due to the synchronization circuit.

Figure 14-65 Counting in external clock mode B, with PR=0x32 and DIV=0x0



Internal trigger input (ISx)

Timer synchronization allows interconnection between several timers. The TMR_CLK of one timer can be provided by the TRGOUT signal output by another timer. Set the STIS[2: 0] bit to select internal trigger signal to enable counting.

Each timer consists of a 16-bit prescaler, which is used to generate the CK_CNT that enables the counter to count. The frequency division relationship between the CK_CNT and TMR_CLK can be adjusted by setting the value of the TMRx_DIV register. The prescaler value can be modified at any time, but it takes effect only when the next overflow event occurs.

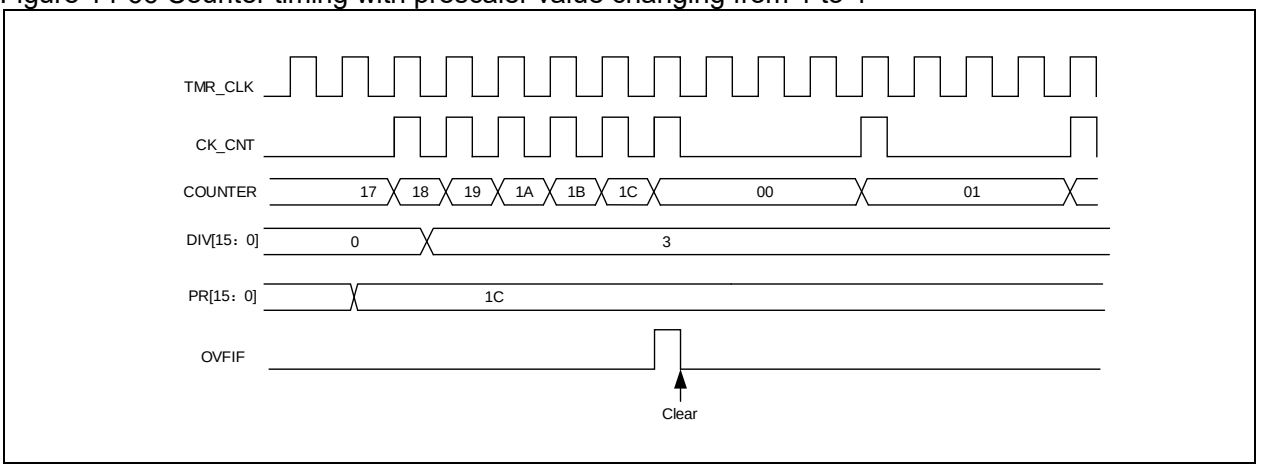
The internal trigger input is configured as follows:

- Set the TMRx_PR register to set counting period.
- Set the TMRx_DIV register to set counting frequency.
- Set the TWCMSEL[1:0] bit in the TMRx_CTRL1 register to set count mode.
- Set the STIS[2:0] bit (range: 3'b000~3'b011) in the TMRx_STCTRL register and select internal trigger.
- Set SMSEL[2:0]=3'b111 in the TMRx_STCTRL register and select external clock mode A.
- Set the TMREN bit in the TMRx_CTRL1 register to enable TMRx counter.

Table 14-12 TMRx internal trigger connection

Slave timer	IS0 (STIS=000)	IS1 (STIS=001)	IS2 (STIS=010)	IS3 (STIS=011)
TMR1	TMR5	TMR2	TMR3	TMR4
TMR8	TMR1	TMR2	TMR4	TMR5
TMR20	TMR8	TMR2	TMR4	TMR5

Figure 14-66 Counter timing with prescaler value changing from 1 to 4



14.4.3.2 Counting mode

The advanced-control timer consists of an internal 16-bit counter supporting up, down, up/down counting modes to meet different application scenarios.

The TMRx_PR register is used to set the counting period. The value in the TMRx_PR is immediately moved to the shadow register by default. When the periodic buffer is enabled (PRBEN=1), the value in the TMRx_PR register is transferred to the shadow register only at an overflow event. The OVFEN and OVFS bits are used to configure the overflow event.

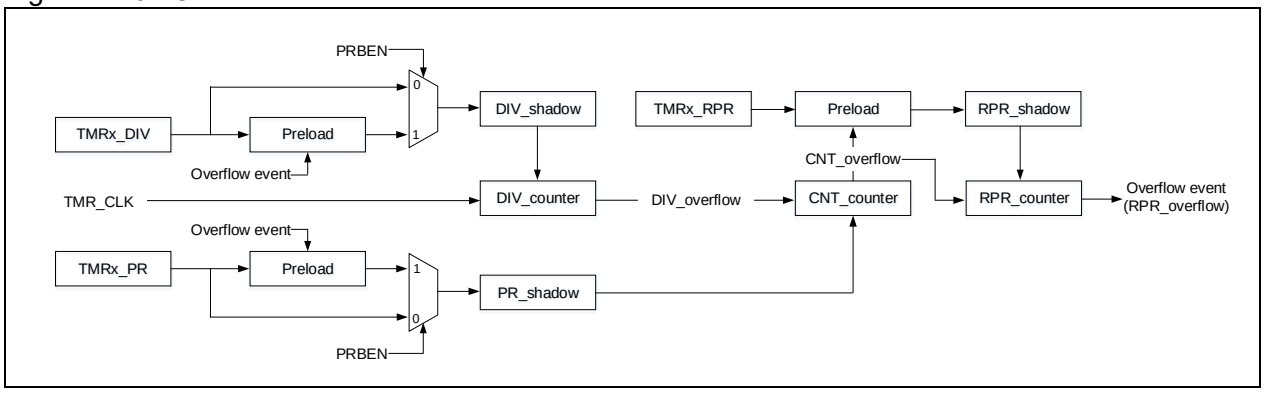
The TMRx_DIV register is used to configure the counting frequency. The counter counts once every count clock period (DIV[15:0]+1). Similar to the TMRx_PR register, when the periodic buffer is enabled, the value in the TMRx_DIV register is updated to the shadow register at an overflow event.

Reading the TMRx_CNT register returns to the current counter value, and writing to the TMRx_CNT register updates the current counter value to the value being written.

An overflow event is generated by default. Set OVFEN=1 in the TMRx_CTRL1 to disable generation of update events. The OVFS bit in the TMRx_CTRL1 register is used to select overflow event source. By default, counter overflow/underflow, setting OVFSWTR bit and the reset signal generated by the slave timer controller in reset mode trigger the generation of an overflow event. When the OVFS bit is set, only counter overflow/underflow triggers an overflow event.

Setting TMREN=1 to enable the timer to start counting. Base on synchronization logic, however, the actual enable signal TMR_EN is set 1 clock cycle after the TMREN is set.

Figure 14-67 Counter structure



Upcounting mode

Set CMSEL[1:0]=2'b00 and OWCDIR=1'b0 in the TMRx_CTRL1 register to enable upcounting mode. In this mode, the counter counts from 0 to the value programmed in the TMRx_PR register, then restarts from 0, and generates a counter overflow event, with the OVFIF bit being set to 1. If the overflow event is disabled, the counter is no longer reloaded with the preload value and period value at a counter overflow event, otherwise, the counter is updated with the preload value and period value on an overflow event.

Figure 14-68 Overflow event when PRBEN=0

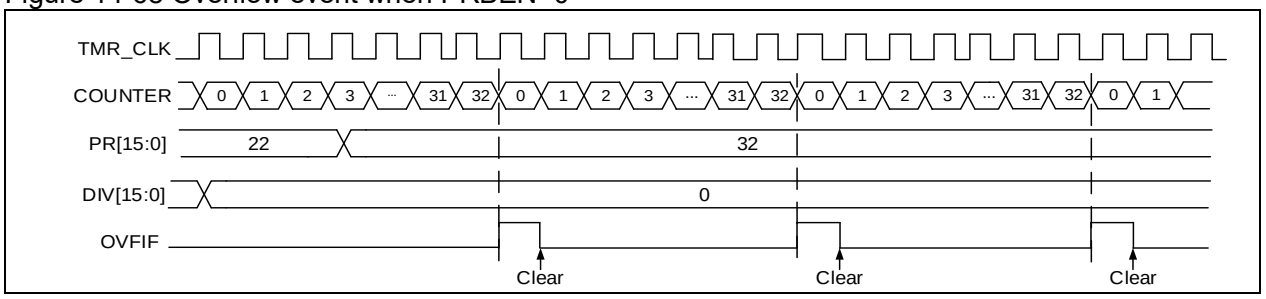
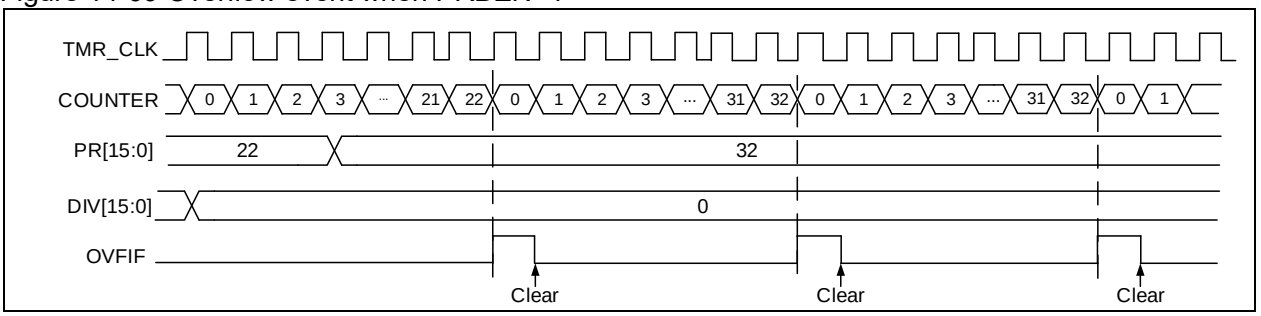


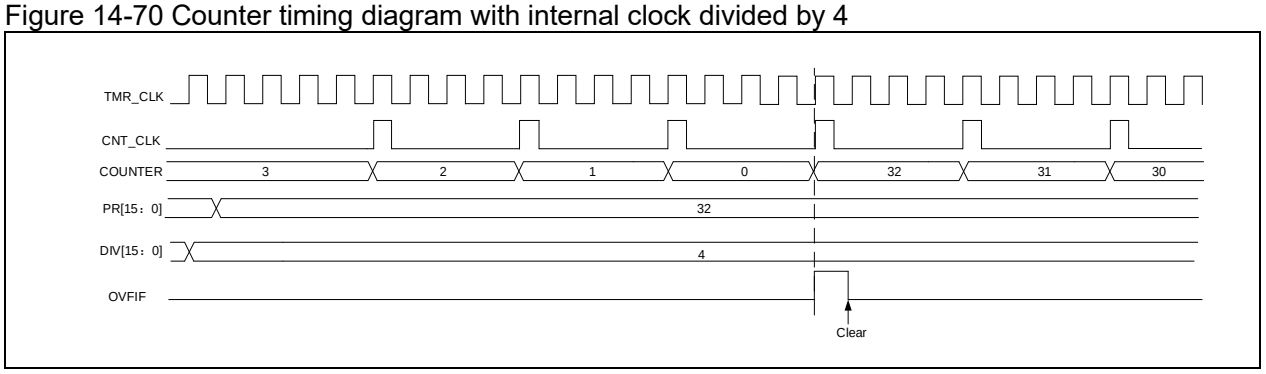
Figure 14-69 Overflow event when PRBEN=1



Downcounting mode

Set CMSEL[1:0]=2'b00 and OWCDIR=1'b1 in the TMRx_CTRL1 register to enable downcounting mode. In this mode, the counter counts from the value programmed in the TMRx_PR register down to 0, and restarts from the value programmed in the TMRx_PR register, and generates a counter underflow event.

Figure 14-70 Counter timing diagram with internal clock divided by 4



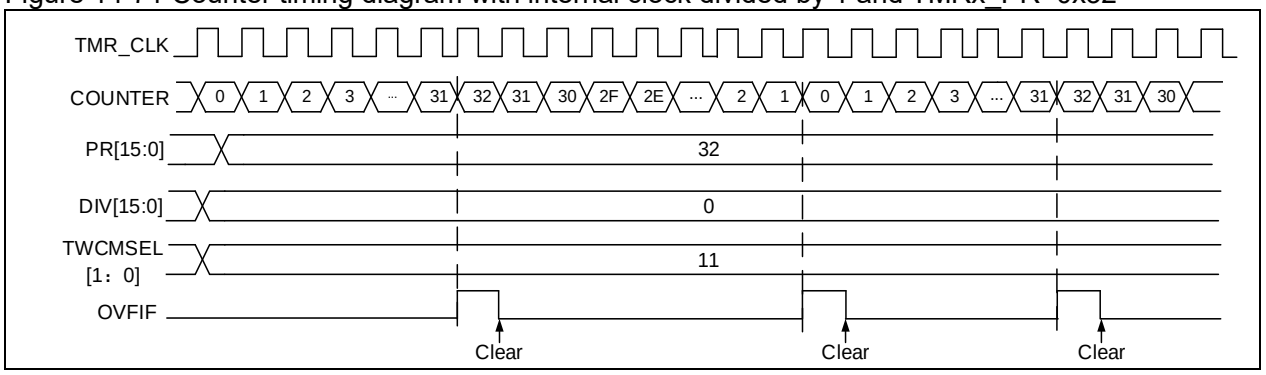
Up/down counting mode

Set CMSEL[1:0]≠2'b00 in the TMRx_CTRL1 register to enable up/down counting mode. In this mode, the counter counts up/down alternatively. When the counter counts from the value programmed in the TMRx_PR register down to 1, an underflow event is generated, and then restarts counting from 0; when the counter counts from 0 to the value of the TMRx_PR register -1, an overflow event is generated, and then restarts counting from the value of the TMRx_PR register. The OWCDIR bit indicates the current counting direction.

The TWCMSEL[1:0] bit in the TMRx_CTRL1 register is also used to select the CxIF flag setting method in up/down counting mode. In up/down counting mode 1 (TWCMSEL[1:0]=2'b01), CxIF flag can only be set when the counter counts down; in up/down counting mode 2 (TWCMSEL[1:0]=2'b10), CxIF flag can only be set when the counter counts up; in up/down counting mode 3 (TWCMSEL[1:0]=2'b11), CxIF flag can be set when the counter counts up/down.

Note: The OWCDIR is ready-only in up/down counting mode.

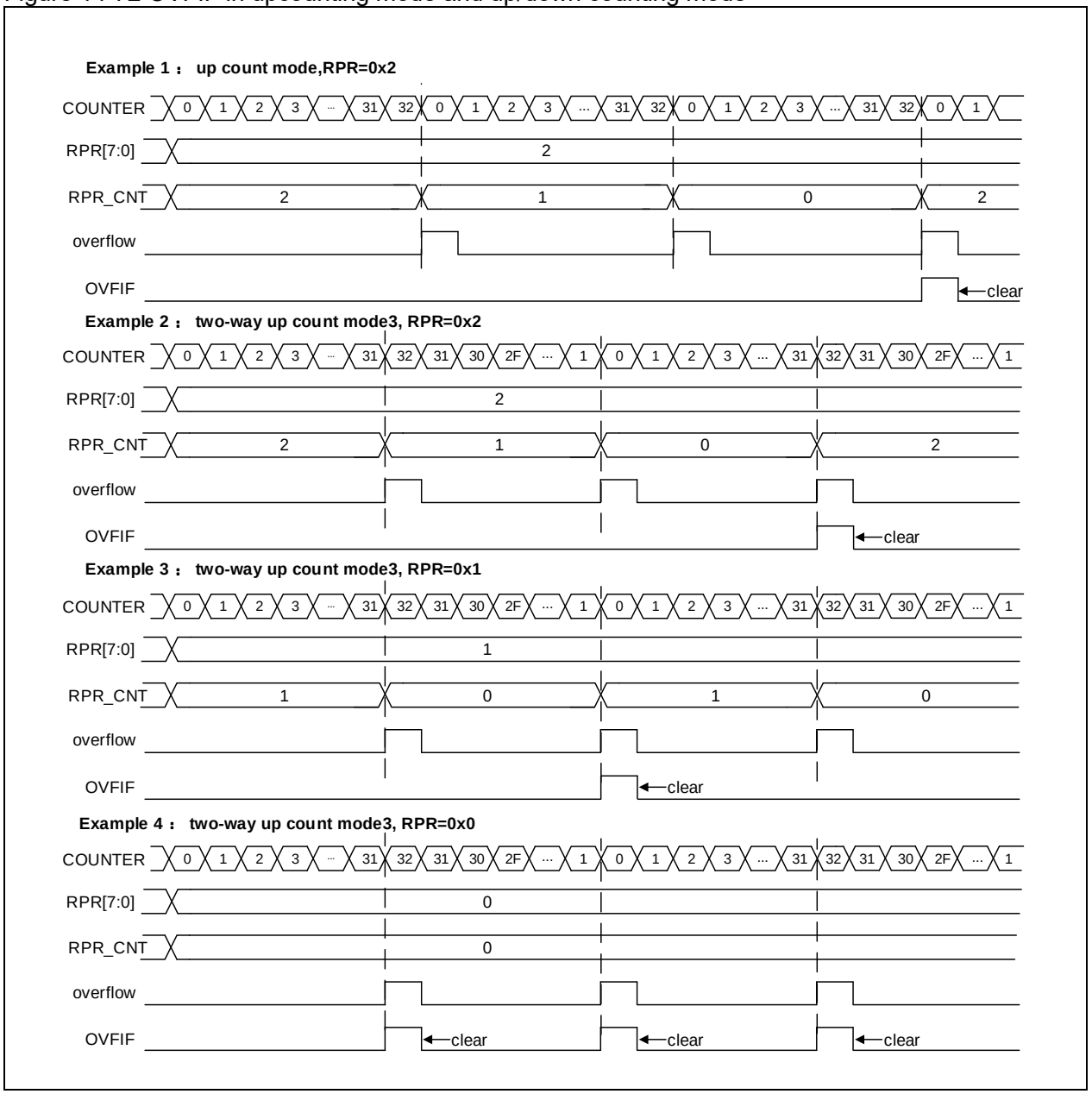
Figure 14-71 Counter timing diagram with internal clock divided by 1 and TMRx_PR=0x32



Repetition counter mode

The TMRx_RPR register is used to set the counting period of repetition counter. The repetition counter mode is enabled when the repetition counter value is not equal to 0. In this mode, the repetition counter is decremented at each counter overflow (RPR[15:0]+1). An overflow event is generated when the repetition counter reaches 0. The frequency of the overflow event can be adjusted by setting the repetition counter value.

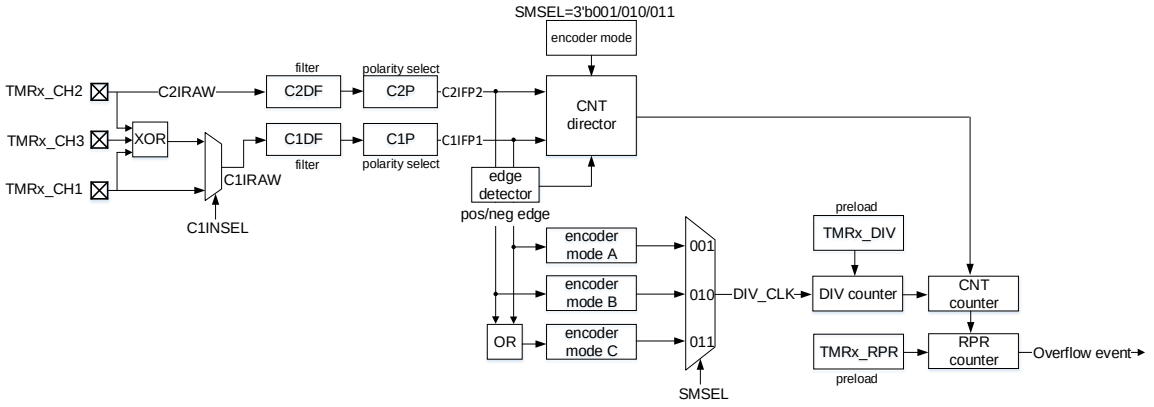
Figure 14-72 OVFIF in upcounting mode and up/down counting mode



Encoder interface mode

To enable the encoder interface mode, write SMSEL[2: 0]= 3'b001/3'b010/3'b011. In this mode, the two inputs (C1IN/C2IN) are required. Depending on the level on one input, the counter counts up or down on the edge of the other input. The OWCDIR bit indicates the direction of the counter.

Figure 14-73 Encoder mode structure



Encoder mode A: SMSEL=3'b001. The counter counts on the selected C1IFP1 edge (rising and falling edges), and the counting direction is dependent on the edge direction of C1IFP1 and the level of C2IFP2.

Encoder mode B: SMSEL=3'b010. The counter counts on the selected C2IFP2 edge (rising and falling edges), and the counting direction is dependent on the edge direction of C2IFP2 and the level of C1IFP1.

Encoder mode C: SMSEL=3'b011. The counter counts on both C1IFP1 and C2IFP2 edges (rising and falling edges). The counting direction is dependent on the C1IFP1 edge direction and C2IFP2 level, and C2IFP2 edge direction and C1IFP1 level.

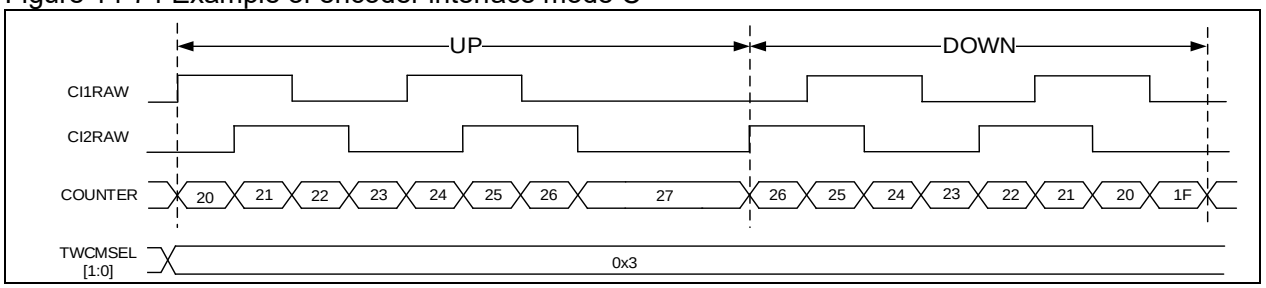
To use encoder mode, follow the procedures below:

- Set channel 1 input signal filtering through the C1DF[3:0] bit in the TMRx_CM1 register;
  Set channel 1 input signal active level through the C1P bit in the TMRx_CCTRL register
- Set channel 2 input signal filtering through the C2DF[3:0] bit in the TMRx_CM1 register;
  Set channel 2 input signal active level through the C2P bit in the TMRx_CCTRL register
- Set channel 1 as input mode through the C1C[1:0] bit in the TMRx_CM1 register;
  Set channel 2 as input mode through the C2C[1:0] bit in the TMRx_CM1 register
- Select encoder mode A (SMSEL=3'b001), encoder mode B (SMSEL=3'b010), or the encoder mode C (SMSEL=3'b011) by setting the SMSEL[2:0] bit in the TMRx_STCTRL register
- Set counting cycles through the PR[15:0] bit in the TMRx_PR register
- Set counting frequency through the DIV[15:0] bit in the TMRx_DIV register
- Configure the corresponding IOs of TMRx_CH1 and TMRx_CH2 as multiplexed mode
- Enable counter through the TMREN bit in the TMRx_CTRL1 register

Table 14-13 Counting direction versus encoder signals

Active edge	Level on opposite signal (C1INFP1 to C2IN, C2INFP2 to C1IN)	C1INFP1 signal		C2INFP2 signal	
		Rising	Falling	Rising	Falling
Count on C1IN only	High	Down	Up	No count	No count
	Low	Up	Down	No count	No count
Count on C2IN only	High	No count	No count	Up	Down
	Low	No count	No count	Down	Up
Count on both C1IN and C2IN	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

Figure 14-74 Example of encoder interface mode C



14.4.3.3 TMR input function

Each timer of TMR1, TMR8 and TMR20 has four independent channels. Each channel can be configured as input or output.

As input, each channel input signal is handled as follows:

- TMRx_CHx outputs the pre-processed CxIRAW. The C1INSEL bit is used to select the source of C1IRAW from TMRx_CH1 or the XOR-ed TMRx_CH1, TMRx_CH2 and TMRx_CH3. The sources of C2IRAW, C3IRAW and C4IRAW are TMRx_CH2, TMRx_CH3 and TMRx_CH4, respectively.
- CxIRAW inputs digital filter and outputs filtered CxIF signal. The digital filter uses the CxDF bit to program sampling frequency and sampling times.
- CxIF inputs edge detector, and outputs the CxIFPx signal after edge selection. The edge selection depends on both CxP and CxCP bits. It is possible to select input rising edge, falling edge or both edges.
- CxIFPx inputs capture signal selector, and outputs the CxIN signal after capture signal selection. The capture signal selection is defined by CxC bit. It is possible to select CxIFPx, CyIFPx or STCI as CxIN source. Of those, CyIFPx (x≠y) is the CyIFPy signal that is from Y channel and processed by channel -x edge detector (for example, C1IFP2 is the channel 1’s C1IFP1 signal that passed through channel 2 edge detection). The STCI comes from slave timer controller, and its source is selected by STIS bit.
- CxIN outputs the CxIPS signal that is divided by input channel divider. The divider factor can be defined as No division, /2, /4 or /8, by the CxIDIV bit. It can be used for filtering, selection, division and input capture of input signals.

Figure 14-75 Input/output channel 1 main circuit

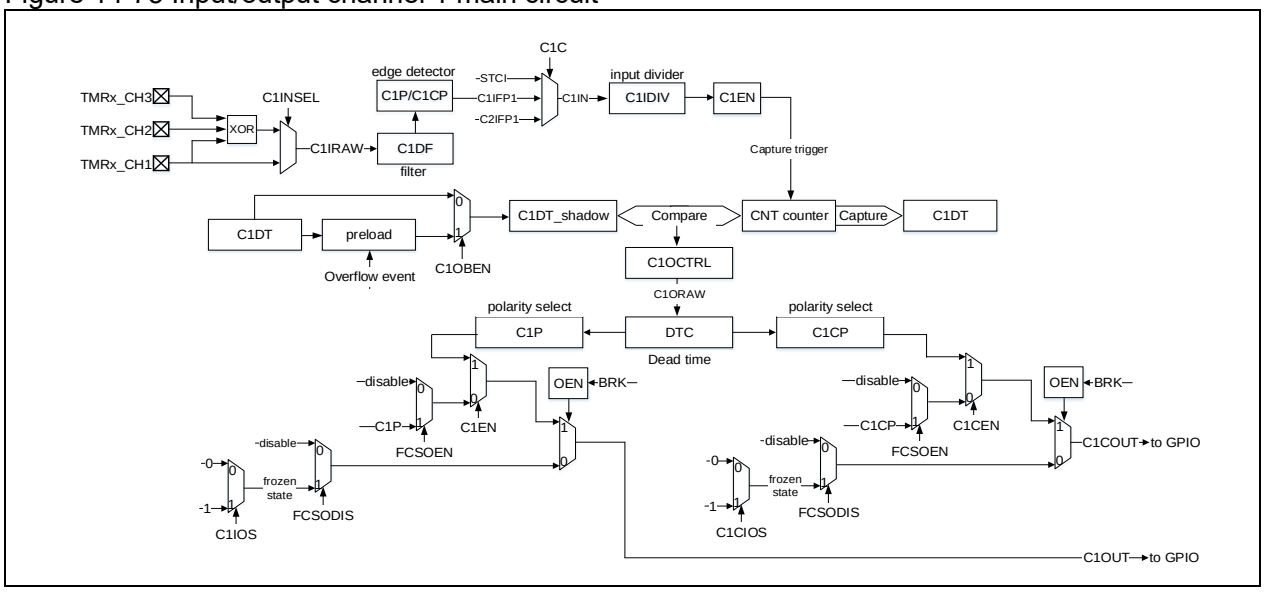
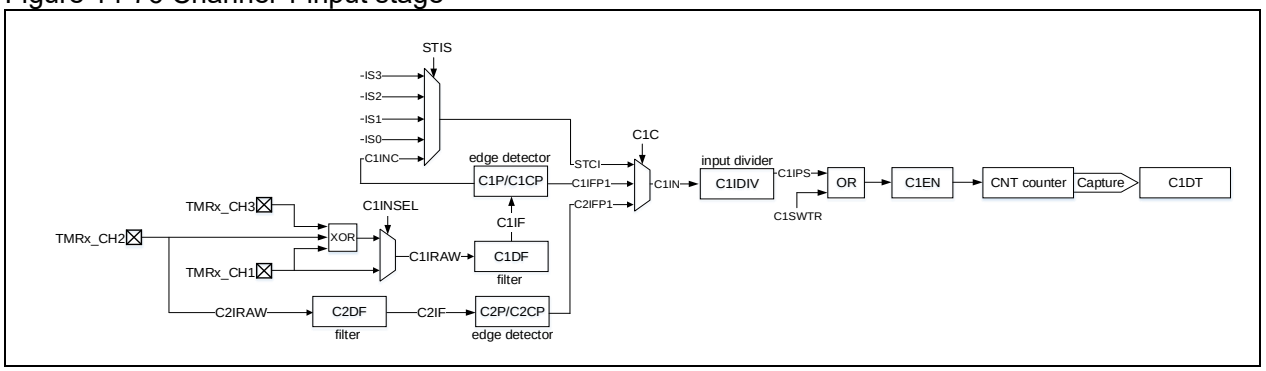


Figure 14-76 Channel 1 input stage



Input mode

In input mode, the TMRx_CxDT registers latch the current counter values after the selected trigger signal is detected, and the capture compare interrupt flag bit (CxIF) is set to 1. An interrupt/DMA request will be generated if the CxIEN or CxDEN bit are enabled. If the selected trigger signal is detected when the CxIF is set, a capture overflow event occurs. The TMRx_CxDT register overwrites the recorded value with the current counter value, and the CxRF is set to 1.

To capture the rising edge of C1IN input, following the configuration procedure mentioned below:

- Set C1C=01 in the TMRx_CxDT register to select the C1IN as channel 1 input
- Set the filter bandwidth of C1IN signal (CxDF[3: 0])
- Set the active edge on the C1IN channel by writing C1P=0 (rising edge) in the TMRx_CCTR register
- Program the capture frequency division of C1IN signal (C1DIV[1: 0])
- Enable channel 1 input capture (C1EN=1)
- If needed, enable the relevant interrupt or DMA request by setting the C1IEN bit in the TMRx_IDEN register or the C1DEN bit in the TMRx_IDEN register

Timer Input XOR function

The timer input pins (TMRx_CH1, TMRx_CH2 and TMRx_CH3) are connected to the channel 1 (selected by setting the C1INSE in the TMRx_CTRL2 register) through an XOR gate.

The XOR gate can be used to connect Hall sensors. For example, connect the three XOR inputs to the three Hall sensors respectively so as to calculate the position and speed of the rotation by analyzing three Hall sensor signals.

PWM input

The PWM input mode applies to channel 1 and channel 2. To enable this mode, map the C1IN and C2IN to the same TMRx_CHx, and configure the CxIFPx of channel 1/2 to trigger slave timer controller reset.

The PWM input mode can be used to measure the period and duty cycle of input signal. The period and duty cycle of channel 1 can be measured as follows:

- Set C1C=2‘b01 to set C1IN as C1IFP1;
- Set C1P=1’b0 to set C1IFP1 rising edge active;
- Set C2C=2‘b10 to set C2IN as C1IFP2;
- Set C2P=1’b1 to set C1IFP2 falling edge active;
- Set STIS=3’b101 to set C1IFP1 as the slave timer trigger signal;
- Set SMSEL=3‘b110 to set the slave timer in reset mode;
- Set C1EN=1’b1 and C2EN=1’b1 to enable channel 1 and input capture.

In these configurations, the rising edge of channel 1 input signal triggers capture and saves captured values to the C1DT register, and channel 1 input signal rising edge resets the counter. The falling edge of channel 1 input signal triggers capture and saves captured values to the C2DT register. The period and duty of channel 1 input signal can be calculated through C1DT and C2DT respectively.

Figure 14-77 Example of PWM input mode configuration

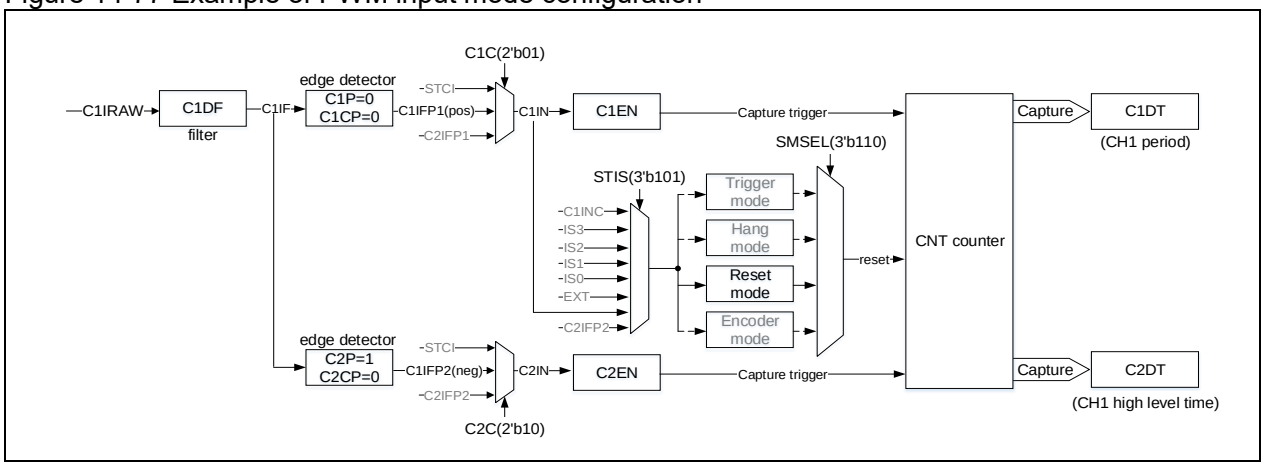
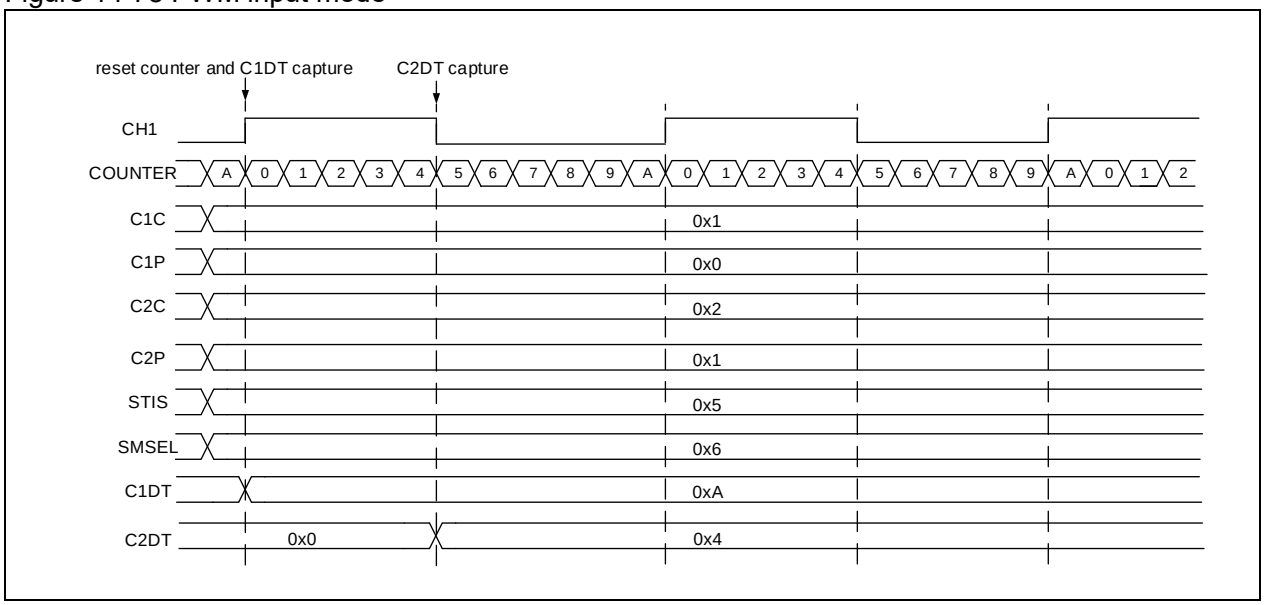


Figure 14-78 PWM input mode



14.4.3.4 TMR output function

The TMR output consists of a comparator and an output controller. It is used to program the period, duty cycle and polarity of the output signal. The advanced-control timer output function varies from one channel to one channel.

Figure 14-79 Channel output stage (channel 1 to 3)

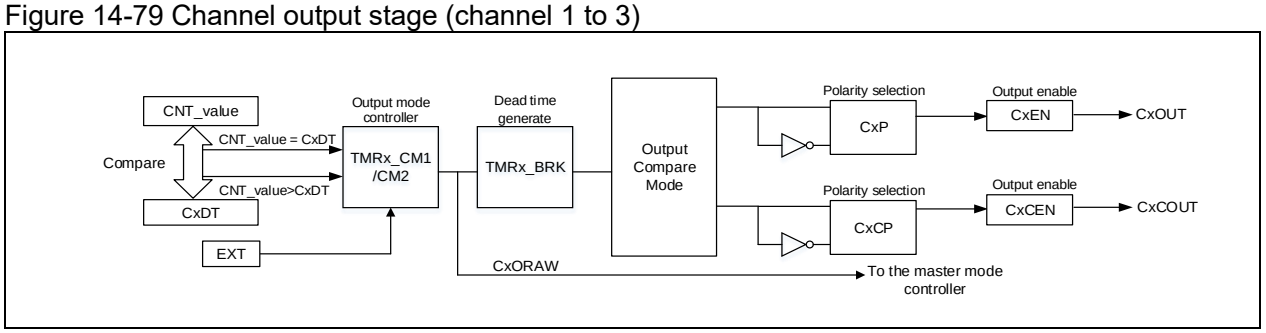
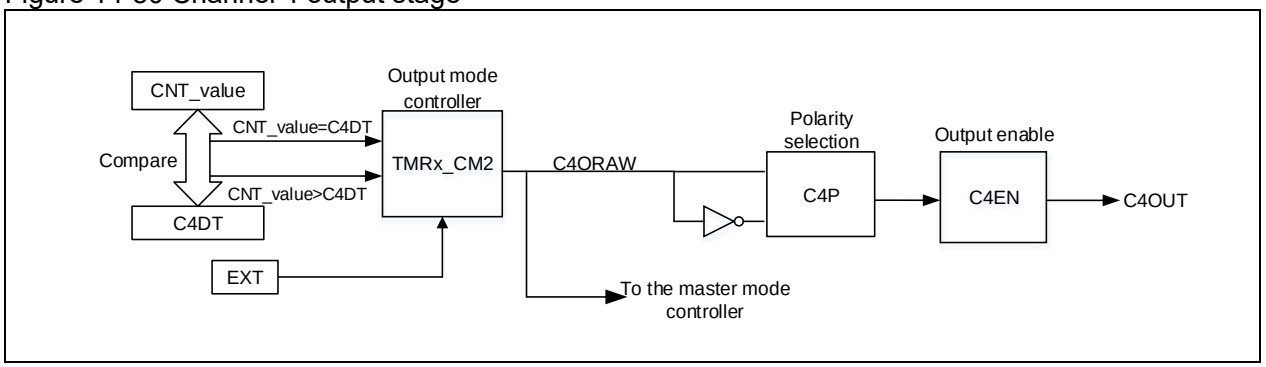


Figure 14-80 Channel 4 output stage



Output mode

Write CxC[1: 0]≠2'b00 to configure the channel as output to implement multiple output modes. In this case, the counter value is compared with the value in the TMRx_CxDT register, and the intermediate signal CxORAW is generated according to the output mode selected by CxOCTRL[2: 0], which is sent to IO after being processed by the output control circuit. The period of the output signal is configured by the TMRx_PR register, while the duty cycle by the TMRx_CxDT register.

Output compare modes include:

- **PWM mode A:** Set CxOCTRL=3'b110 to enable PWM mode A. In upcounting, when TMRx_C1DT>TMRx_CVAL, C1ORAW outputs high; otherwise, outputs low. In downcounting, when TMRx_C1DT<TMRx_CVAL, C1ORAW outputs low; otherwise, outputs high. To set PWM mode A, the following process is recommended:
  - Set the TMRx_PR register to set PWM period;
  - Set the TMRx_CxDT register to set PWM duty cycle;
  - Set CxOCTRL=3'b110 in the TMRx_CM1/CM2 register to set output mode as PWM mode A;
  - Set the TMRx_DIV register and set the counting frequency;
  - Set the TWCMSEL[1:0] bit in the TMRx_CTRL1 register to set the count mode;
  - Set CxP bit and CxCP bit in the TMRx_CCTRL register to set output polarity;
  - Set CxEN bit and CxCEN bit in the TMRx_CCTRL register to enable channel output;
  - Set the OEN bit in the TMRx_BRK register to enable TMRx output;
  - Set the corresponding GPIO of TMR output channel as the multiplexed mode;
  - Set the TMREN bit in the TMRx_CTRL1 register to enable TMRx counter.
- **PWM mode B:** Set CxOCTRL=3'b111 to enable PWM mode B. In upcounting, when TMRx_C1DT>TMRx_CVAL, C1ORAW outputs low; otherwise, outputs high. In downcounting, when TMRx_C1DT<TMRx_CVAL, C1ORAW outputs high; otherwise, outputs low.
- **Forced output mode:** Set CxOCTRL=3'b100/101 to enable forced output mode. In this case, the CxORAW is forced to be the programmed level, irrespective of the counter value. Despite this, the channel flag bit and DMA request still depend on the compare result.
- **Output compare mode:** Set CxOCTRL=3'b001/010/011 to enable output compare mode. In this case, when the counter value matches the value of the CxDT register, the CxORAW is forced high (CxOCTRL=3'b001), low (CxOCTRL=3'b010) or toggling (CxOCTRL=3'b011).
- **One-pulse mode:** This is a particular case of PWM mode. Set OCMEN=1 to enable one-pulse mode. In this mode, the comparison match is performed in the current counting period. The TMREN bit is cleared as soon as the current counting is completed. Therefore, only one pulse is output. When configured as in upcounting mode, the configuration must follow the rule: CVAL<CxDT≤PR; in downcounting mode, CVAL>CxDT is required.
- **Fast output mode:** Set CxOIEN=1 to enable this mode. If enabled, the CxORAW signal will not change when the counter value matches the CxDT, but at the beginning of the current counting period. In other words, the comparison result is advanced, so the comparison result between the counter value and the TMRx_CxDT register will determine the level of CxORAW in advance.

Figure 14-81 gives an example of output compare mode (toggle) with C1DT=0x3. When the counter value is equal to 0x3, C1OUT toggles.

Figure 14-82 gives an example of the combination between upcounting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

Figure 14-83 gives an example of the combination between up/down counting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

Figure 14-84 gives an example of the combination between upcounting mode and one-pulse PWM mode B. The counter only counts only one cycle, and the output signal sends only one pulse.

Figure 14-81 C1ORAW toggles when counter value matches the C1DT value

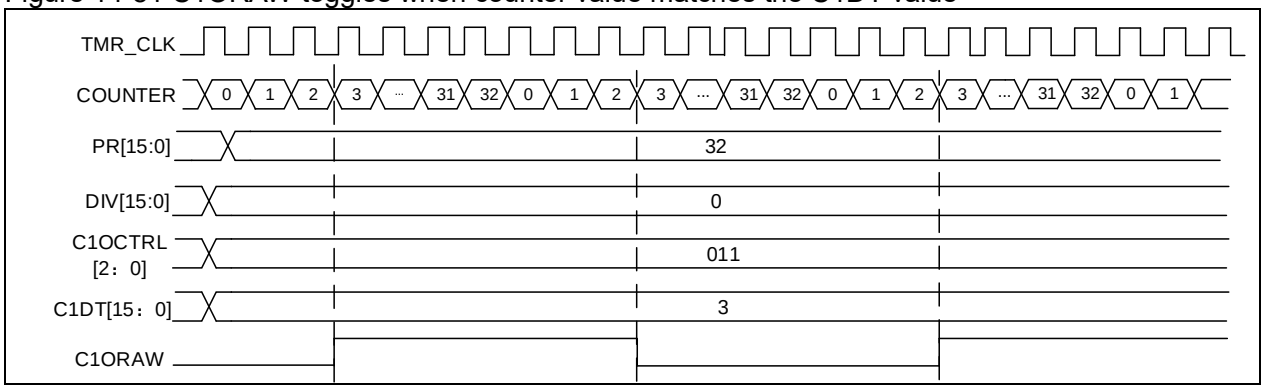


Figure 14-82 Upcounting mode and PWM mode A

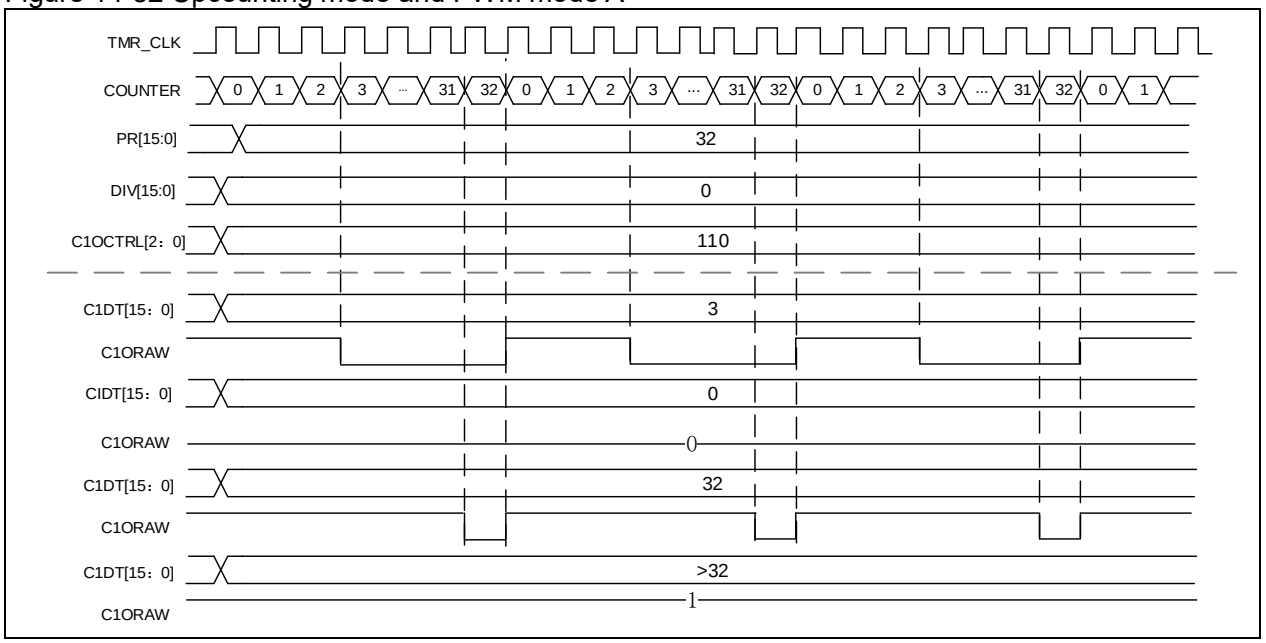


Figure 14-83 Up/down counting mode and PWM mode A

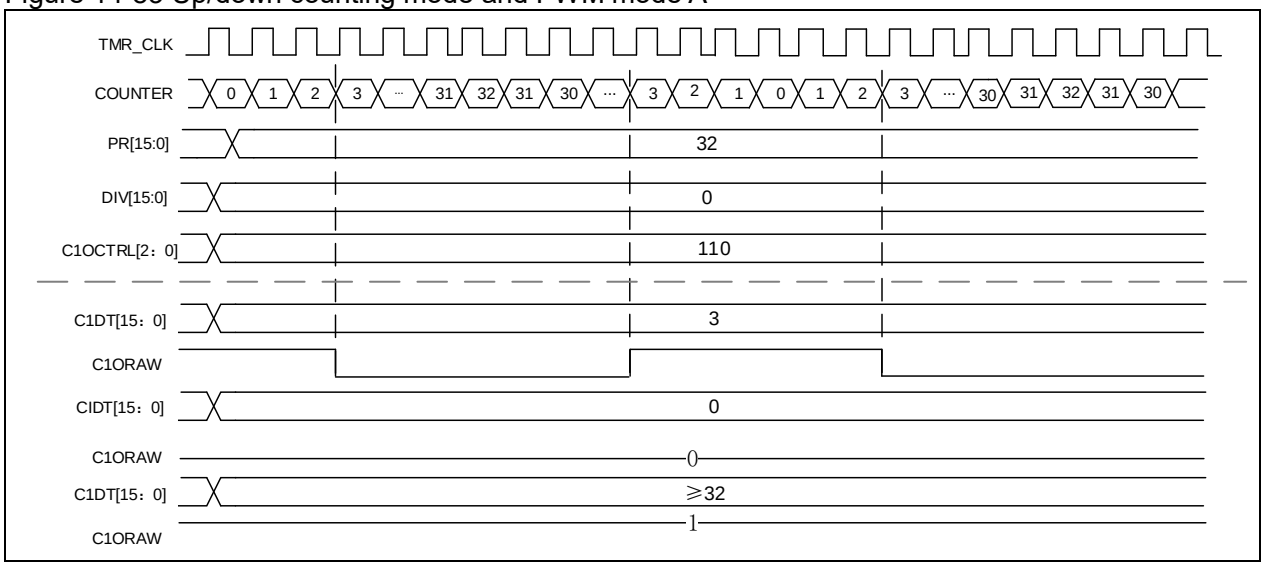
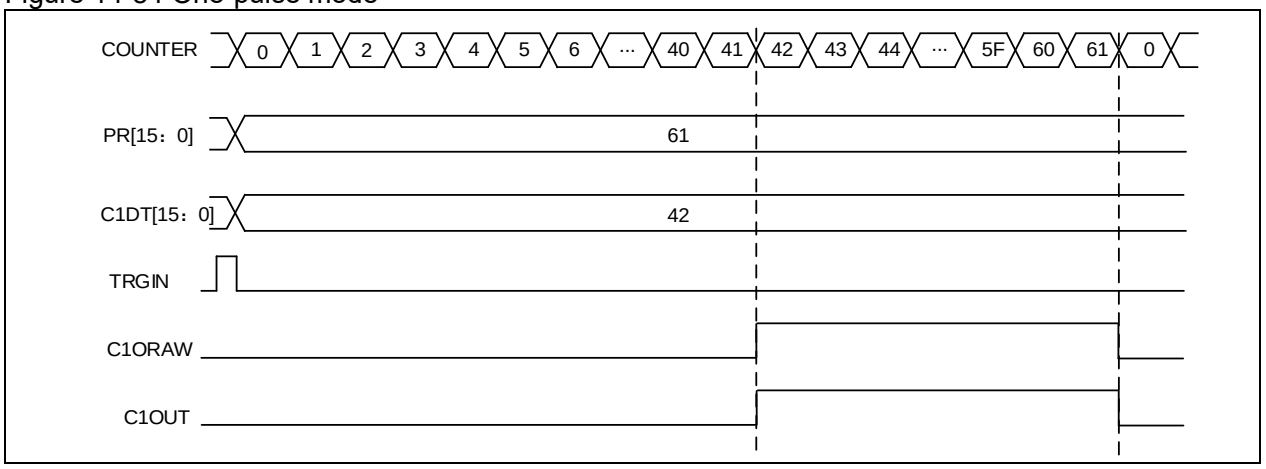


Figure 14-84 One-pulse mode



Master timer event output

When TMR is selected as the master timer, the following signal sources can be selected as TRGOUT signal to output to the slave timer, by setting the PTOS bit in the TMRxCTRL2 register.

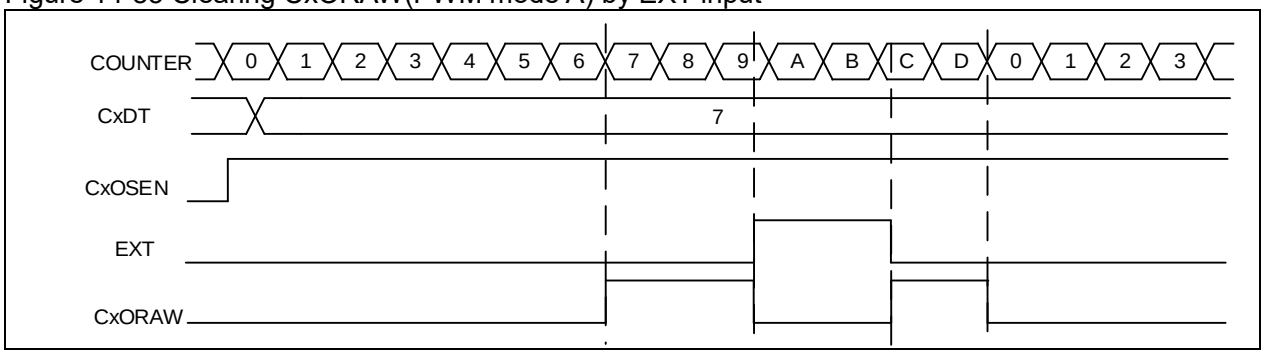
- PTOS=3'b000, TRGOUT outputs software overflow event (OVFSWTR bit in the TMRx_SWEVT register).
- PTOS=3'b001, TRGOUT outputs counter enable signal.
- PTOS=3'b010, TRGOUT outputs counter overflow event.
- PTOS=3'b011, TRGOUT outputs capture and compare event.
- PTOS=3'b100, TRGOUT outputs C1ORAW signal.
- PTOS=3'b101, TRGOUT outputs C2ORAW signal.
- PTOS=3'b110, TRGOUT outputs C3ORAW signal.
- PTOS=3'b111, TRGOUT outputs C4ORAW signal.

CxORAW clear

When the CxOSEN bit is set to 1, the CxORAW signal for a given channel is cleared by applying a high level to the EXT input. The CxORAW signal remains unchanged until the next overflow event.

This function can only be used in output capture or PWM modes, and does not work in forced mode. *Figure 14-85* shows the example of clearing CxORAW. When the EXT input is high, the CxORAW signal, which was originally high, is driven low; when the EXT is low, the CxORAW signal outputs the corresponding level according to the comparison result between the counter value and CxDT value.

Figure 14-85 Clearing CxORAW(PWM mode A) by EXT input



Dead-time insertion

The channel 1 to 3 of the advanced-control timers contains a set of reverse channel output. This function is enabled by the CxCEN bit and its polarity is defined by CxCP. Refer to Table 14-15 for more information about the output state of CxOUT and CxCOUT.

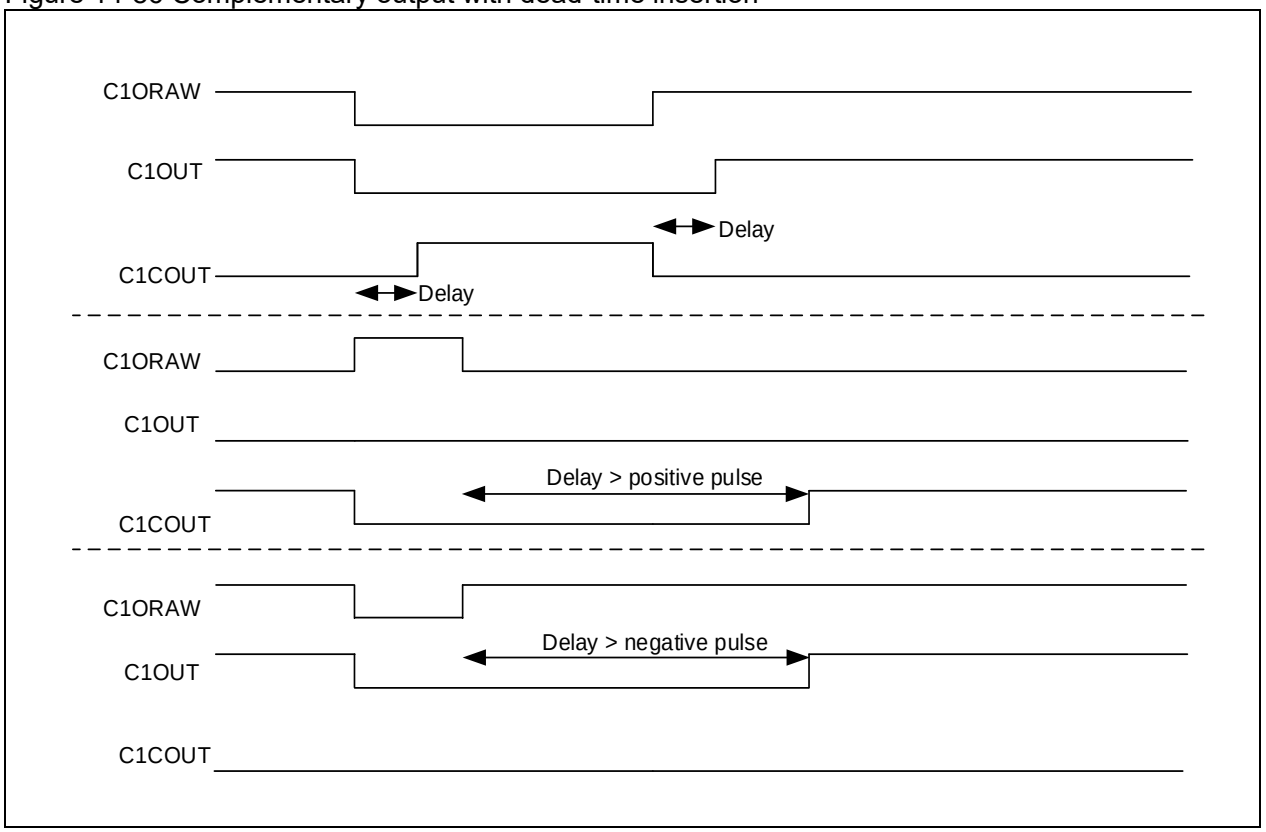
The dead-time is activated when switching to IDLEF state (OEN falling down to 0).

Setting both CxEN and CxCEN bits, and using DTC[7:0] bit to insert dead-time of different durations. After the dead-time insertion, the rising edge of the CxOUT is delayed compared to the rising edge of the reference signal; the rising edge of the CxCOU is delayed compared to the falling edge of the reference signal.

If the delay is greater than the width of the active output, and if C1OUT and C1COUT do not generate corresponding pulses, the dead-time should be less than the width of the active output.

Figure 14-86 gives an example of dead-time insertion when CxP=0, CxCP=0, OEN=1, CxEN=1 and CxCEN=1.

Figure 14-86 Complementary output with dead-time insertion



14.4.3.5 TMR break function

When the break function is enabled (BRKEN=1), the CxOUT and CxCOUT are jointly controlled by OEN, FCSODIS, FCSOEN, CxIOS and CxCIOS. But, CxOUT and CxCOUT cannot be set both to active level at the same time. Please refer to 14-15 for more details.

The break source can be the break input pin or a clock failure event. The polarity is controlled by the

BRKV bit.

When a break event occurs, there are the following actions:

- The OEN bit is cleared asynchronously, and the channel output state is selected by setting the FCSODIS bit. This function works even if the MCU oscillator is off.
- Once OEN=0, the channel output level is defined by the CxIOS bit. If FCSODIS=0, the timer output is disabled, otherwise, the output enable remains high.
- When complementary outputs are used:
  - The outputs are first put in reset state, that is, inactive state (depending on the polarity). This is done asynchronously so that it works even if no clock is provided to the timer.
  - If the timer clock is still active, then the dead-time generator is activated. The CxIOS and CxCIOS bits are used to program the level after dead-time. Even in this case, the CxIOS and CxCIOS cannot be driven to their active level at the same time. It should be note that because of synchronization on OEN, the dead-time duration is usually longer than usual (around 2 clk_tmr clock cycles)
  - If FCSODIS=0, the timer releases the enable output, otherwise, it keeps the enable output; the enable output becomes high as soon as one of the CxEN and CxCEN bits becomes high.
- If the break interrupt or DMA request is enabled, the break statue flag is set, and a break interrupt or DMA request can be generated.
- If AOEN=1, the OEN bit is automatically set again at the next overflow event.

Note: When the break input is active, the OEN cannot be set, nor the status flag, BRKIF can be cleared.

Figure 14-87 TMR output control

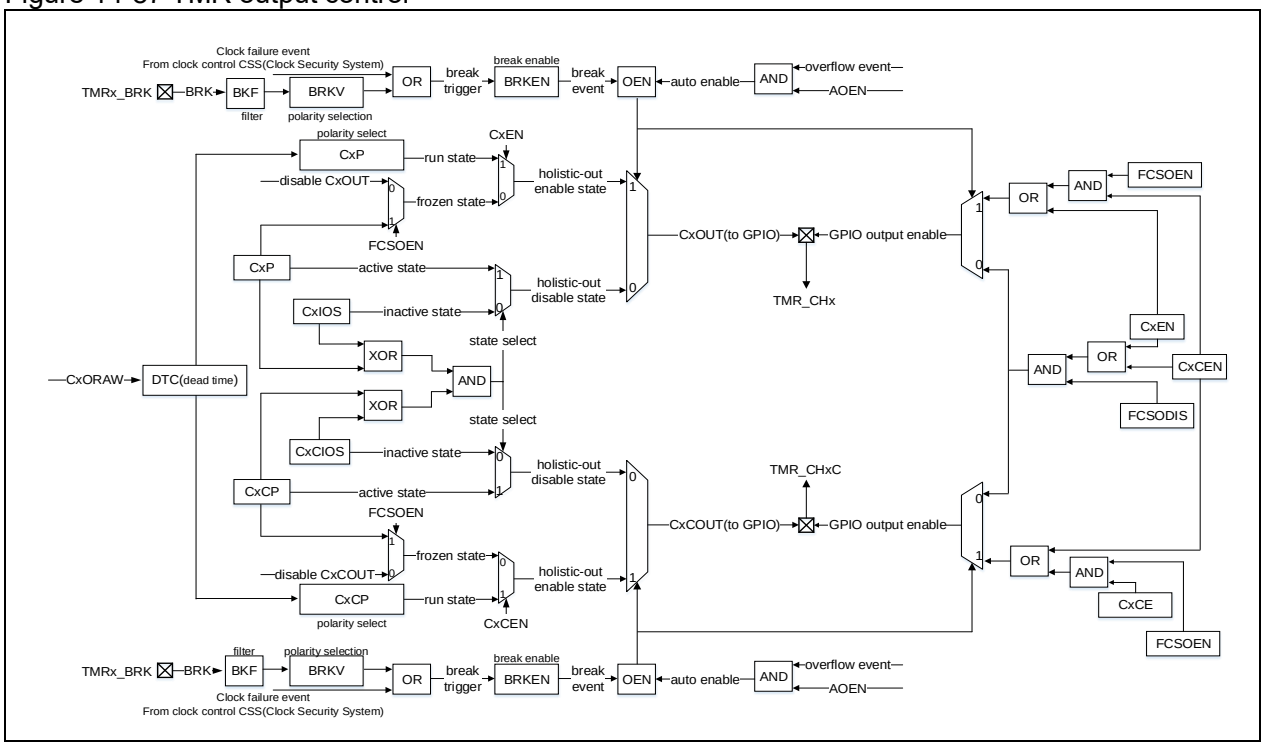
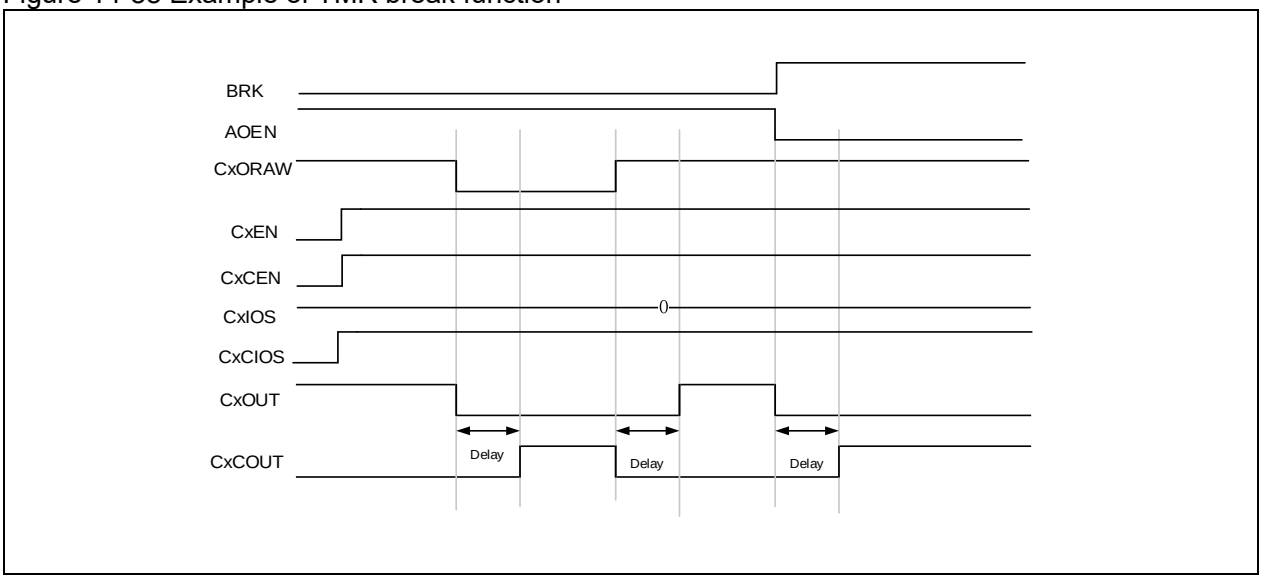


Figure 14-88 Example of TMR break function



14.4.3.6 TMR synchronization

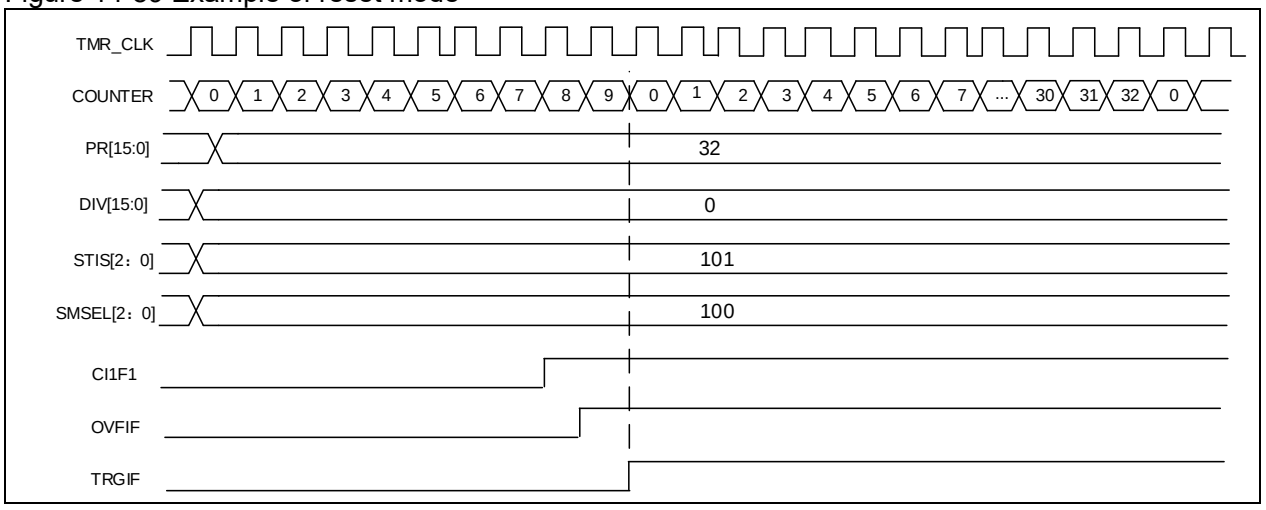
The timers are linked together internally for timer synchronization. Master timer is selected by setting the PTOS[2: 0] bit; Slave timer is selected by setting the SMSEL[2: 0] bit.

Slave modes include:

Slave mode: Reset mode

The counter and its prescaler can be reset by a selected trigger signal. An overflow event can be generated when OVFS=0.

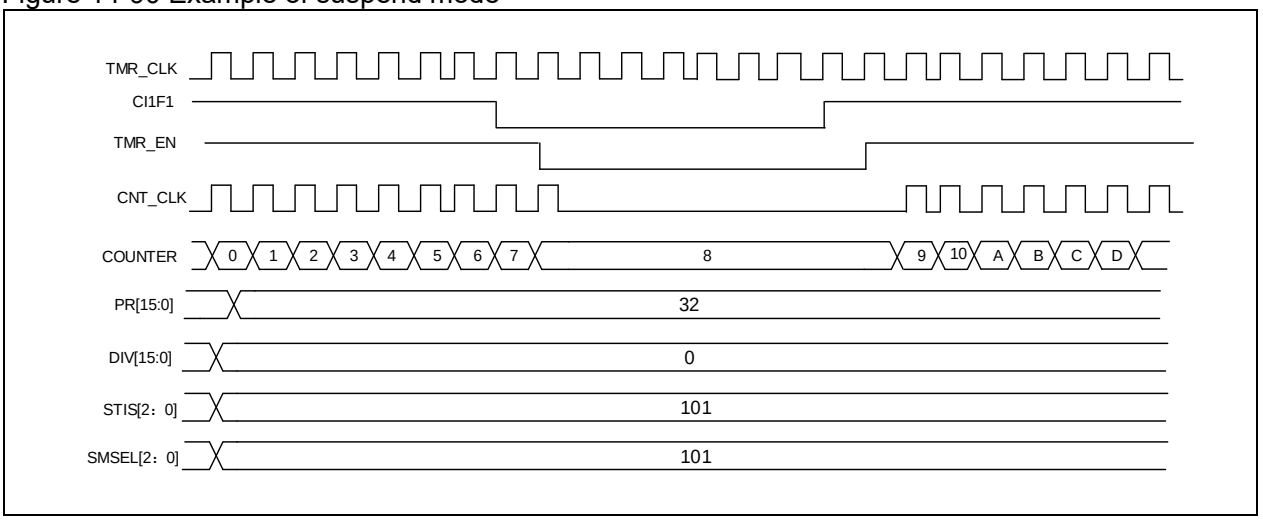
Figure 14-89 Example of reset mode



Slave mode: Suspend mode

In this mode, the counter is controlled by a selected trigger input. The counter starts counting when the trigger input is high and stops as soon as the trigger input is low.

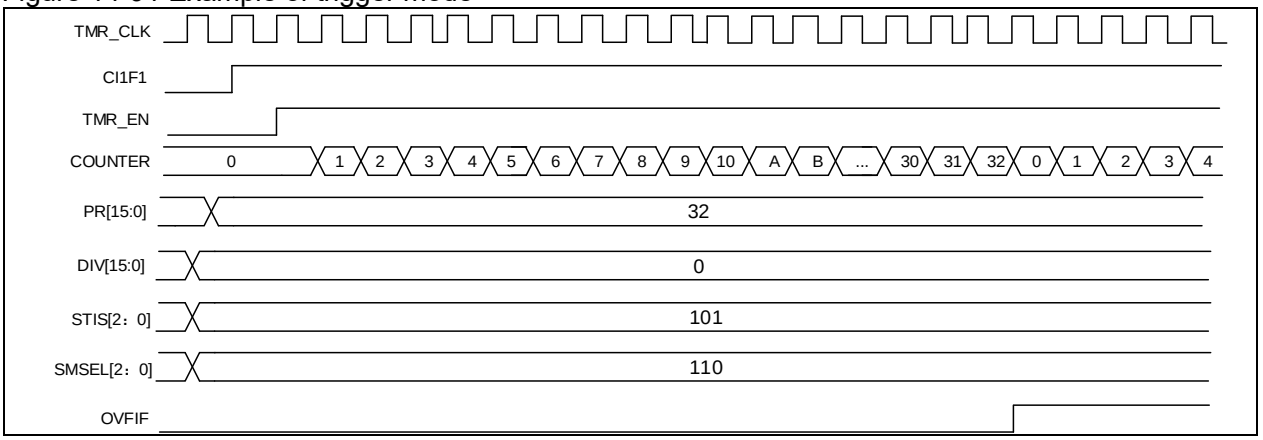
Figure 14-90 Example of suspend mode



Slave mode: Trigger mode

The counter can start counting on the rising edge of a selected trigger input (TMR_EN=1)

Figure 14-91 Example of trigger mode



14.4.3.7 Debug mode

When the microcontroller enters debug mode (Cortex™-M4F core halted), the TMRx counter stops counting by setting the TMRx_PAUSE in the DEBUG module. Refer to Chapter 30.2 for more information.

14.4.4 TMR1, TMR8 and TM20 registers

These peripheral registers must be accessed by word (32 bits).

TMR1 and TMR8 register are mapped into a 16-bit addressable space.

Table 14-14 TMR1 and TMR8 register map and reset value

Register	Offset	Reset value
TMRx_CTRL1	0x00	0x0000
TMRx_CTRL2	0x04	0x0000 0000
TMRx_STCTRL	0x08	0x0000
TMRx_IDEN	0x0C	0x0000
TMRx_ISTS	0x10	0x0000
TMRx_SWEVT	0x14	0x0000
TMRx_CM1	0x18	0x0000
TMRx_CM2	0x1C	0x0000
TMRx_CCTRL	0x20	0x0000
TMRx_CVAL	0x24	0x0000
TMRx_DIV	0x28	0x0000
TMRx_PR	0x2C	0x0000
TMRx_RPR	0x30	0x0000
TMRx_C1DT	0x34	0x0000
TMRx_C2DT	0x38	0x0000
TMRx_C3DT	0x3C	0x0000
TMRx_C4DT	0x40	0x0000
TMRx_BRK	0x44	0x0000
TMRx_DMACTRL	0x48	0x0000
TMRx_DMADT	0x4C	0x0000
TMRx_CM3	0x70	0x0000
TMRx_C5DT	0x74	0x0000

14.4.4.1 TMR1, TMR8 and TMR20 control register1 (TMRx_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at its default value.
Bit 9: 8	CLKDIV	0x0	rw	Clock division This field is used to set the ratio between digital filter sampling frequency (f_{DT}) and timer clock frequency (f_{CK_INT}). It is also used to adjust the ratio between dead time base (T_{DTS}) and timer clock cycle (T_{CK_INT}). 00: Normal 01: Divided by 2 10: Divided by 4 11: Reserved
Bit 7	PRBEN	0x0	rw	Period buffer enable 0: Period buffer is disabled 1: Period buffer is enabled
Bit 6: 5	TWCMSEL	0x0	rw	Two-way counting mode selection 00: One-way counting mode, depending on the OWCDIR bit 01: Two-way counting mode1, count up and down alternately, the output flag bit is set only when the counter

				counts down 10: Two-way counting mode2, count up and down alternately, the output flag bit is set only when the counter counts up 11: Two-way counting mode3, count up and down alternately, the output flag bit is set when the counter counts up / down
Bit 4	OWCDIR	0x0	rw	One-way count direction 0: Up 1: Down
Bit 3	OCMEN	0x0	rw	One cycle mode enable This bit is use to select whether to stop counting at an update event 0: The counter does not stop at an update event 1: The counter stops at an update event
Bit 2	OVFS	0x0	rw	Overflow event source This bit is used to select overflow event or DMA request sources. 0: Counter overflow, setting the OVFSWTR bit or overflow event generated by slave timer controller 1: Only counter overflow generates an overflow event
Bit 1	OVFEN	0x0	rw	Overflow event enable 0: Enabled 1: Disabled
Bit 0	TMREN	0x0	rw	TMR enable 0: Disabled 1: Enabled

14.4.4.2 TMR1, TMR8 and TMR20 control register2 (TMRx_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 31	TRGOUT2EN	0x0	rw	TRGOUT2 enable 0: TRGOUT2 disabled 1: TRGOUT2 enabled
Bit 30: 15	Reserved	0x0000	resd	Kept at its default value.
Bit 14	C4IOS	0x0	rw	Channel 4 idle output state
Bit 13	C3CIOS	0x0	rw	Channel 3 complementary idle output state
Bit 12	C3IOS	0x0	rw	Channel 3 idle output state
Bit 11	C2CIOS	0x0	rw	Channel 2 complementary idle output state
Bit 10	C2IOS	0x0	rw	Channel 2 idle output state
Bit 9	C1CIOS	0x0	rw	Channel 1 complementary idle output state OEN = 0 after dead-time: 0: C1OUTL=0 1: C1OUTL=1
Bit 8	C1IOS	0x0	rw	Channel 1 idle output state OEN = 0 after dead-time: 0: C1OUT=0 1: C1OUT=1
Bit 7	C1INSEL	0x0	rw	C1IN selection 0: CH1 pin is connected to C1IRAW input 1: The XOR result of CH1, CH2 and CH3 pins is connected to C1IRAW input
Bit 6: 4	PTOS	0x0	rw	Master TMR output selection This field is used to select the TMRx signal sent to the slave timer. 000: Reset 001: Enable 010: Update 011: Compare pulse 100: C1ORAW signal 101: C2ORAW signal 110: C3ORAW signal 111: C4ORAW signal
Bit 3	DRS	0x0	rw	DMA request source

				0: Capture/compare event 1: Overflow event
Bit 2	CCFS	0x0	rw	Channel control bit flash selection This bit only acts on channels that have complementary output. If the channel control bits are buffered: 0: Control bits are updated by setting the HALL bit 1: Control bits are updated by setting the HALL bit or a rising edge on TRGIN.
Bit 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	CBCTRL	0x0	rw	Channel buffer control This bit acts on channels that have complementary output. 0: CxEN, CxCEN and CxOCTRL bits are not buffered. 1: CxEN, CxCEN and CxOCTRL bits are not buffered.

14.4.4.3 TMR1, TMR8 and TMR20 slave timer control register (TMRx_STCTRL)

Bit	Register	Reset value	Type	Description
Bit 15	ESP	0x0	rw	External signal polarity 0: High or rising edge 1: Low or falling edge
Bit 14	ECMBEN	0x0	rw	External clock mode B enable This bit is used to enable external clock mode B 0: Disabled 1: Enabled
Bit 13: 12	ESDIV	0x0	rw	External signal divide This field is used to select the frequency division of an external trigger 00: Normal 01: Divided by 2 10: Divided by 4 11: Divided by 8
Bit 11: 8	ESF	0x0	rw	External signal filter This field is used to filter an external signal. The external signal can be sampled only after it has been generated N times 0000: No filter, sampling by f_{DTS} 0001: $f_{SAMPLING}$ =f_{CK_INT}, N=2 0010: $f_{SAMPLING}$ =f_{CK_INT}, N=4 0011: $f_{SAMPLING}$ =f_{CK_INT}, N=8 0100: $f_{SAMPLING}$ =f_{DTS}/2, N=6 0101: $f_{SAMPLING}$ =f_{DTS}/2, N=8 0110: $f_{SAMPLING}$ =f_{DTS}/4, N=6 0111: $f_{SAMPLING}$=f_{DTS}/4, N=8 1000: $f_{SAMPLING}$=f_{DTS}/8, N=6 1001: $f_{SAMPLING}$=f_{DTS}/8, N=8 1010: $f_{SAMPLING}$=f_{DTS}/16, N=5 1011: $f_{SAMPLING}$=f_{DTS}/16, N=6 1100: $f_{SAMPLING}$=f_{DTS}/16, N=8 1101: $f_{SAMPLING}$=f_{DTS}/32, N=5 1110: $f_{SAMPLING}$=f_{DTS}/32, N=6 1111: $f_{SAMPLING}$=f_{DTS}/32, N=8
Bit 7	STS	0x0	rw	Subordinate TMR synchronization If enabled, master and slave timer can be synchronized. 0: Disabled 1: Enabled
Bit 6: 4	STIS	0x0	rw	Subordinate TMR input selection This field is used to select the subordinate TMR input. 000: Internal selection 0 (IS0) 001: Internal selection 1 (IS1) 010: Internal selection 2 (IS2) 011: Internal selection 3 (IS3) 100: C1IRAW input detector (C1INC)

				101: Filtered input 1 (C1IF1) 110: Filtered input 2 (C1IF2) 111: External input (EXT) Please refer to Table 14-3 and 14-5 for more information on ISx for each timer.
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2: 0	SMSEL	0x0	rw	Subordinate TMR mode selection 000: Slave mode is disabled 001: Encoder mode A 010: Encoder mode B 011: Encoder mode C 100: Reset mode — Rising edge of the TRGIN input reinitializes the counter 101: Suspend mode — The counter starts counting when the TRGIN is high 110: Trigger mode — A trigger event is generated at the rising edge of the TRGIN input 111: External clock mode A — Rising edge of the TRGIN input clocks the counter Note: Please refer to count mode section for the details on encoder mode A/B/C.

14.4.4.4 TMR1, TMR8 and TMR20 DMA/interrupt enable register (TMRx_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	TDEN	0x0	rw	Trigger DMA request enable 0: Disabled 1: Enabled
Bit 13	HALLDE	0x0	rw	HALL DMA request enable 0: Disabled 1: Enabled
Bit 12	C4DEN	0x0	rw	Channel 4 DMA request enable 0: Disabled 1: Enabled
Bit 11	C3DEN	0x0	rw	Channel 3 DMA request enable 0: Disabled 1: Enabled
Bit 10	C2DEN	0x0	rw	Channel 2 DMA request enable 0: Disabled 1: Enabled
Bit 9	C1DEN	0x0	rw	Channel 1 DMA request enable 0: Disabled 1: Enabled
Bit 8	OVFDEN	0x0	rw	Overflow event DMA request enable 0: Disabled 1: Enabled
Bit 7	BRKIE	0x0	rw	Break interrupt enable 0: Disabled 1: Enabled
Bit 6	TIEN	0x0	rw	Trigger interrupt enable 0: Disabled 1: Enabled
Bit 5	HALLIEN	0x0	rw	HALL interrupt enable 0: Disabled 1: Enabled
Bit 4	C4IEN	0x0	rw	Channel 4 interrupt enable 0: Disabled 1: Enabled
Bit 3	C3IEN	0x0	rw	Channel 3 interrupt enable 0: Disabled 1: Enabled

Bit 2	C2IEN	0x0	rw	Channel 2 interrupt enable 0: Disabled 1: Enabled
Bit 1	C1IEN	0x0	rw	Channel 1 interrupt enable 0: Disabled 1: Enabled
Bit 0	OVFIEN	0x0	rw	Overflow interrupt enable 0: Disabled 1: Enabled

14.4.4.5 TMR1, TMR8 and TMR20 interrupt status register (TMRx_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	C4RF	0x0	rw0c	Channel 4 recapture flag Please refer to C1RF description.
Bit 11	C3RF	0x0	rw0c	Channel 3 recapture flag Please refer to C1RF description.
Bit 10	C2RF	0x0	rw0c	Channel 2 recapture flag Please refer to C1RF description.
Bit 9	C1RF	0x0	rw0c	Channel 1 recapture flag This bit indicates whether a recapture is detected when C1IF=1. This bit is set by hardware, and cleared by writing “0”. 0: No capture is detected 1: Capture is detected.
Bit 8	Reserved	0x0	resd	Default value
Bit 7	BRKIF	0x0	rw0c	Break interrupt flag This bit indicates whether the break input is active or not. It is set by hardware and cleared by writing “0” 0: Inactive level 1: Active level
Bit 6	TRGIF	0x0	rw0c	Trigger interrupt flag This bit is set by hardware on a trigger event. It is cleared by writing “0”. 0: No trigger event occurs 1: Trigger event is generated. Trigger event: an active edge is detected on TRGIN input, or any edge in suspend mode.
Bit 5	HALLIF	0x0	rw0c	HALL interrupt flag This bit is set by hardware on HALL event. It is cleared by writing “0”. 0: No Hall event occurs. 1: Hall event is detected. HALL even: CxEN, CxCEN and CxOCTRL are updated.
Bit 4	C4IF	0x0	rw0c	Channel 4 interrupt flag Please refer to C1IF description.
Bit 3	C3IF	0x0	rw0c	Channel 3 interrupt flag Please refer to C1IF description.
Bit 2	C2IF	0x0	rw0c	Channel 2 interrupt flag Please refer to C1IF description.
Bit 1	C1IF	0x0	rw0c	Channel 1 interrupt flag If the channel 1 is configured as input mode: This bit is set by hardware on a capture event. It is cleared by software or read access to the TMRx_C1DT 0: No capture event occurs 1: Capture event is generated If the channel 1 is configured as output mode: This bit is set by hardware on a compare event. It is cleared by software. 0: No compare event occurs 1: Compare event is generated

Bit 0	OVFIF	0x0	rw0c	Overflow interrupt flag This bit is set by hardware on an overflow event. It is cleared by software. 0: No overflow event occurs 1: Overflow event is generated. If OVFEN=0 and OVFS=0 in the TMRx_CTRL1 register: − An overflow event is generated when OVFG= 1 in the TMRx_SWEVE register; − An overflow event is generated when the counter CVAL is reinitialized by a trigger event.

14.4.4.6 TMR1, TMR8 and TMR20 software event register (TMRx_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x000	resd	Kept at its default value.
Bit 7	BRKSWTR	0x0	wo	Break event triggered by software This bit is set by software to generate a break event. 0: No effect 1: Generate a break event.
Bit 6	TRGSWTR	0x0	rw	Trigger event triggered by software This bit is set by software to generate a trigger event. 0: No effect 1: Generate a trigger event.
Bit 5	HALLSWTR	0x0	wo	HALL event triggered by software This bit is set by software to generate a HALL event. 0: No effect 1: Generate a HALL event. Note: This bit acts only on channels that have complementary output.
Bit 4	C4SWTR	0x0	wo	Channel 4 event triggered by software Please refer to C1M description.
Bit 3	C3SWTR	0x0	wo	Channel 3 event triggered by software Please refer to C1M description.
Bit 2	C2SWTR	0x0	wo	Channel 2 event triggered by software Please refer to C1M description
Bit 1	C1SWTR	0x0	wo	Channel 1 event triggered by software This bit is set by software to generate a channel 1 event. 0: No effect 1: Generate a channel 1 event.
Bit 0	OVFSWTR	0x0	wo	Overflow event triggered by software This bit is set by software to generate an overflow event. 0: No effect 1: Generate an overflow event.

14.4.4.7 TMR1, TMR8 and TMR20 channel mode register1 (TMRx_CM1)

The channel can be used in input (capture mode) or output (compare mode). The direction of a channel is defined by the corresponding CxC bits. All the other bits of this register have different functions in input and output modes. The CxOx describes its function in output mode when the channel is in output mode, while the CxIx describes its function in output mode when the channel is in input mode. Attention must be given to the fact that the same bit can have different functions in input mode and output mode.

Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15	C2OSEN	0x0	rw	Channel 2 output switch enable
Bit 14: 12	C2OCTRL	0x0	rw	Channel 2 output control
Bit 11	C2OBEN	0x0	rw	Channel 2 output buffer enable
Bit 10	C2OIEN	0x0	rw	Channel 2 output enable immediately
Bit 9: 8	C2C	0x0	rw	Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0': 00: Output

				01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS register.
Bit 7	C1OSEN	0x0	rw	Channel 1 output switch enable 0: C1ORAW is not affected by EXT input. 1: Once a high level is detect on EXT input, clear C1ORAW.
Bit 6: 4	C1OCTRL	0x0	rw	Channel 1 output control This field defines the behavior of the original signal C1ORAW. 000: Disconnected. C1ORAW is disconnected from C1OUT; 001: C1ORAW is high when TMRx_CVAL=TMRx_C1DT 010: C1ORAW is low when TMRx_CVAL=TMRx_C1DT 011: Switch C1ORAW level when TMRx_CVAL=TMRx_C1DT 100: C1ORAW is forced low 101: C1ORAW is forced high. 110: PWM mode A – OWCDIR=0, C1ORAW is high once TMRx_C1DT>TMRx_CVAL, else low; – OWCDIR=1, C1ORAW is low once TMRx_ C1DT <TMRx_CVAL, else high; 111: PWM mode B – OWCDIR=0, C1ORAW is low once TMRx_ C1DT >TMRx_CVAL, else high; – OWCDIR=1, C1ORAW is high once TMRx_ C1DT <TMRx_CVAL, else low. *Note: In the configurations other than 000', the C1OUT is connected to C1ORAW. The C1OUT output level is not only subject to the changes of C1ORAW, but also the output polarity set by CCTRL.*
Bit 3	C1OBEN	0x0	rw	Channel 1 output buffer enable 0: Buffer function of TMRx_C1DT is disabled. The new value written to the TMRx_C1DT takes effect immediately. 1: Buffer function of TMRx_C1DT is enabled. The value to be written to the TMRx_C1DT is stored in the buffer register, and can be sent to the TMRx_C1DT register only on an overflow event.
Bit 2	C1OIEN	0x0	rw	Channel 1 output enable immediately In PWM mode A or B, this bit is used to accelerate the channel 1 output's response to the trigger event. 0: Need to compare the CVAL with C1DT before generating an output 1: No need to compare the CVAL and C1DT. An output is generated immediately when a trigger event occurs.
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

Input capture mode:

Bit	Register	Reset value	Type	Description
Bit 15: 12	C2DF	0x0	rw	Channel 2 digital filter
Bit 11: 10	C2IDIV	0x0	rw	Channel 2 input divider
Bit 9: 8	C2C	0x0	rw	Channel 2 configuration

				This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0': 00: Output 01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7: 4	C1DF	0x0	rw	Channel 1 digital filter This field defines the digital filter of the channel 1. N stands for the number of filtering, indicating that the input edge can pass the filter only after N sampling events. 0000: No filter, sampling is done at f_{DTS} 1000: $f_{SAMPLING}=f_{DTS}/8$, N=6 0001: $f_{SAMPLING}=f_{CK_INT}$, N=2 1001: $f_{SAMPLING}=f_{DTS}/8$, N=8 0010: $f_{SAMPLING}=f_{CK_INT}$, N=4 1010: $f_{SAMPLING}=f_{DTS}/16$, N=5 0011: $f_{SAMPLING}=f_{CK_INT}$, N=8 1011: $f_{SAMPLING}=f_{DTS}/16$, N=6 0100: $f_{SAMPLING}=f_{DTS}/2$, N=6 1100: $f_{SAMPLING}=f_{DTS}/16$, N=8 0101: $f_{SAMPLING}=f_{DTS}/2$, N=8 1101: $f_{SAMPLING}=f_{DTS}/32$, N=5 0110: $f_{SMPLING}=f_{DTS}/4$, N=6 1110: $f_{SAMPLING}=f_{DTS}/32$, N=6 0111: $f_{SAMPLING}=f_{DTS}/4$, N=8 1111: $f_{SAMPLING}=f_{DTS}/32$, N=8
Bit 3: 2	C1IDIV	0x0	rw	Channel 1 input divider This field defines Channel 1 input divider. 00: No divider. An input capture is generated at each active edge. 01: An input compare is generated every 2 active edges 10: An input compare is generated every 4 active edges 11: An input compare is generated every 8 active edges Note: the divider is reset once C1EN='0'
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

14.4.4.8 TMR1, TMR8 and TMR20 channel mode register2 (TMRx_CM2)

The channel can be used in input (capture mode) or output (compare mode). The direction of a channel is defined by the corresponding CxC bits. All the other bits of this register have different functions in input and output modes. The CxOx describes its function in output mode when the channel is in output mode, while the CxIx describes its function in output mode when the channel is in input mode. Attention must be given to the fact that the same bit can have different functions in input mode and output mode.

Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15	C4OSEN	0x0	rw	Channel 4 output switch enable
Bit 14: 12	C4OCTRL	0x0	rw	Channel 4 output control
Bit 11	C4OBEN	0x0	rw	Channel 4 output buffer enable
Bit 10	C4OIEN	0x0	rw	Channel 4 output enable immediately
Bit 9: 8	C4C	0x0	rw	Channel 4 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C4EN='0':

				00: Output 01: Input, C4IN is mapped on C4IFP4 10: Input, C4IN is mapped on C3IFP4 11: Input, C4IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7	C3OSEN	0x0	rw	Channel 3 output switch enable
Bit 6: 4	C3OCTRL	0x0	rw	Channel 3 output control
Bit 3	C3OBEN	0x0	rw	Channel 3 output buffer enable
Bit 2	C3OIEN	0x0	rw	Channel 3 output enable immediately
Bit 1: 0	C3C	0x0	rw	Channel 3 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C3EN='0': 00: Output 01: Input, C3IN is mapped on C3IFP3 10: Input, C3IN is mapped on C4IFP3 11: Input, C3IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

Input capture mode:

Bit	Register	Reset value	Type	Description
Bit 15: 12	C4DF	0x0	rw	Channel 4 digital filter
Bit 11: 10	C4IDIV	0x0	rw	Channel 4 input divider
Bit 9: 8	C4C	0x0	rw	Channel 4 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C4EN='0': 00: Output 01: Input, C4IN is mapped on C4IFP4 10: Input, C4IN is mapped on C3IFP4 11: Input, C4IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7: 4	C3DF	0x0	rw	Channel 3 digital filter
Bit 3: 2	C3IDIV	0x0	rw	Channel 3 input divider
Bit 1:0	C3C	0x0	rw	Channel 3 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C3EN='0': 00: Output 01: Input, C3IN is mapped on C3IFP3 10: Input, C3IN is mapped on C4IFP3 11: Input, C3IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

14.4.4.9 TMR1, TMR8 and TMR20 Channel control register (TMRx_CCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 14	Reserved	0x0	resd	Kept its default value.
Bit 13	C4P	0x0	rw	Channel 4 polarity Please refer to C1P description.
Bit 12	C4EN	0x0	rw	Channel 4 enable Please refer to C1EN description.
Bit 11	C3CP	0x0	rw	Channel 3 complementary polarity Please refer to C1P description.
Bit 10	C3CEN	0x0	rw	Channel 3 complementary enable Please refer to C1EN description.
Bit 9	C3P	0x0	rw	Channel 3 polarity Please refer to C1P description.
Bit 8	C3EN	0x0	rw	Channel 3 enable Please refer to C1EN description.
Bit 7	C2CP	0x0	rw	Channel 2 complementary polarity Please refer to C1P description.
Bit 6	C2CEN	0x0	rw	Channel 2 complementary enable

				Please refer to C1EN description.
Bit 5	C2P	0x0	rw	Channel 2 polarity Please refer to C1P description.
Bit 4	C2EN	0x0	rw	Channel 2 enable Please refer to C1EN description.
Bit 3	C1CP	0x0	rw	Channel 1 complementary polarity 0: C1COUT is active high. 1: C1COUT is active low.
Bit 2	C1CEN	0x0	rw	Channel 1 complementary enable 0: Output is disabled. 1: Output is enabled.
Bit 1	C1P	0x0	rw	Channel 1 polarity When the channel 1 is configured as output mode: 0: C1OUT is active high 1: C1OUT is active low When the channel 1 is configured as input mode: 0: C1IN active edge is on its rising edge. When used as external trigger, C1IN is not inverted. 1: C1IN active edge is on its falling edge. When used as external trigger, C1IN is inverted.
Bit0	C1EN	0x0	rw	Channel 1 enable 0: Input or output is disabled 1: Input or output is enabled

Table 14-15 Complementary output channel CxOUT and CxCOUT control bits with break function

Control bit					Output state (1)	
OEN bit	FCSODIS bit	FCSOEN bit	CxEN bit	CxCEN bit	CxOUT output state	CxCOUT output state
1	X	0	0	0	Output disabled (no driven by the timer) CxOUT=0, Cx_EN=0	Output disabled (no driven by the timer) CxCOUT=0, CxCEN=0
		0	0	1	Output disabled (no driven by the timer) CxOUT=0, Cx_EN=0	CxORAW + polarity, CxCOUT= CxORAW xor CxCP, CxCEN=1
		0	1	0	CxORAW+ polarity CxOUT= CxORAW xor CxP, Cx_EN=1	Output disabled (no driven by the timer) CxCOUT=0, CxCEN=0
		0	1	1	CxORAW+polarity+dead-time, Cx_EN=1	CxORAW inverted+polarity+dead-time, CxCEN=1
		1	0	0	Output disabled (no driven by the timer) CxOUT=CxP, Cx_EN=0	Output disabled (no driven by the timer) CxCOUT=CxCP, CxCEN=0
		1	0	1	Off-state (Output enabled with inactive level) CxOUT=CxP, Cx_EN=1	CxORAW + polarity, CxCOUT= CxORAW xor CxCP, CxCEN=1
		1	1	0	CxORAW + polarity, CxOUT= CxORAW xor CxP, Cx_EN=1	Off-state (Output enabled with inactive level) CxCOUT=CxCP, CxCEN=1
		1	1	1	CxORAW+ polarity+dead-time, Cx_EN=1	CxORAW inverted+polarity+dead-time, CxCEN=1
0	0	X	0	0	Output disabled (the corresponding IO disconnected from timer, IO floating)	
	0		0	1		

0		1	0	Asynchronously: CxOUT=CxP, Cx_EN=0, CxCOUT=CxCP, CxCEN=0;
0		1	1	If the clock is present: after a dead-time, CxOUT=CxIOS, CxCOUT=CxCIOS, assuming that CxIOS and CxCIOS do not correspond to CxOUT and CxCOUT active level.
1		0	0	CxEN=CxCEN=0: output disabled (the corresponding IO disconnected from timer, IO floating)
1		0	1	Other: Off-state (the corresponding channel outputs inactive level)
1		1	0	Asynchronously: CxOUT =CxP, Cx_EN=1, CxCOUT=CxCP, CxCEN=1;
1		1	1	If the clock is present: after a dead-time, CxOUT=CxIOS, CxCOUT=CxCIOS, assuming that CxIOS and CxCIOS do not correspond to CxOUT and CxCOUT active level.

Note: If the two outputs of a channel are not used (CxEN = CxCEN = 0), CxIOS, CxCIOS, CxP and CxCP must be cleared.

Note: The state of the external I/O pins connected to the complementary CxOUT and CxCOUT channels depends on the CxOUT and CxCOUT channel state and the GPIO and the IOMUX registers.

14.4.4.10 TMR1, TMR8 and TMR20 counter value (TMRx_CVAL)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0000	rw	Counter value

14.4.4.11 TMR1, TMR8 and TMR20 division value (TMRx_DIV)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DIV	0x0000	rw	Divider value The counter clock frequency $f_{CK_CNT} = f_{TMR_CLK}$ / (DIV[15: 0]+1). The value of this register is transferred to the actual prescaler register when an overflow event occurs.

14.4.4.12 TMR1, TMR8 and TMR20 period register (TMRx_PR)

Bit	Register	Reset value	Type	Description
Bit 15: 0	PR	0x0000	rw	Period value This defines the period value of the TMRx counter. The timer stops working when the period value is 0.

14.4.4.13 TMR1, TMR8 and TMR20 repetition period register (TMRx_RPR)

Bit	Register	Reset value	Type	Description
Bit 15: 0	RPR	0x00	rw	Repetition of period value This field is used to reduce the generation rate of overflow events. An overflow event is generated when the repetition counter reaches 0.

14.4.4.14 TMR1, TMR8 and TMR20 channel 1 data register (TMRx_C1DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C1DT	0x0000	rw	Channel 1 data register When the channel 1 is configured as input mode: The C1DT is the CVAL value stored by the last channel 1 input event (C1IN) When the channel 1 is configured as output mode: C1DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C1OBEN bit, and the corresponding output is generated on C1OUT as configured.

14.4.4.15 TMR1, TMR8 and TMR20 channel 2 data register (TMRx_C2DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C2DT	0x0000	rw	Channel 2 data register When the channel 2 is configured as input mode: The C2DT is the CVAL value stored by the last channel 2 input event (C1IN) When the channel 2 is configured as output mode: C2DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C2OBEN bit, and the corresponding output is generated on C2OUT as configured.

14.4.4.16 TMR1, TMR8 and TMR20 channel 3 data register (TMRx_C3DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C3DT	0x0000	rw	Channel 3 data register When the channel 3 is configured as input mode: The C3DT is the CVAL value stored by the last channel 3 input event (C1IN) When the channel 3 is configured as output mode: C3DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C3OBEN bit, and the corresponding output is generated on C3OUT as configured.

14.4.4.17 TMR1, TMR8 and TMR20 channel 4 data register (TMRx_C4DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C3DT	0x0000	rw	Channel 4 data register When the channel 4 is configured as input mode: The C4DT is the CVAL value stored by the last channel 4 input event (C1IN) When the channel 3 is configured as output mode: C4DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C4OBEN bit, and the corresponding output is generated on C4OUT as configured.

14.4.4.18 TMR1, TMR8 and TMR20 break register (TMRx_BRK)

Bit	Register	Reset value	Type	Description
Bit 15	OEN	0x0	rw	Output enable This bit acts on the channels as output. It is used to enable CxOUT and CxCOUT outputs. 0: Disabled 1: Enabled
Bit 14	AOEN	0x0	rw	Automatic output enable OEN is set automatically at an overflow event. 0: Disabled 1: Enabled
Bit 13	BRKV	0x0	rw	Break input validity This bit is used to select the active level of a break input. 0: Break input is active low. 1 Break input is active high.
Bit 12	BRKEN	0x0	rw	Break enable This bit is used to enable break input. 0: Break input is disabled. 1: Break input is enabled.
Bit 11	FCSOEN	0x0	rw	Frozen channel status when holistic output enable

				This bit acts on the channels that have complementary output. It is used to set the channel state when the timer is inactive and MOEN=1. 0: CxOUT/CxCOUT outputs are disabled. 1: CxOUT/CxCOUT outputs are enabled. Output inactive level.
Bit 10	FCSODIS	0x0	rw	Frozen channel status when holistic output disable This bit acts on the channels that have complementary output. It is used to set the channel state when the timer is inactive and MOEN=0. 0: CxOUT/CxCOUT outputs are disabled. 1: CxOUT/CxCOUT outputs are enabled. Output idle level.
Bit 9: 8	WPC	0x0	rw	Write protection configuration This field is used to enable write protection. 00: Write protection is OFF. 01: Write protection level 3, and the following bits are write protected: TMRx_BRK: DTC, BRKEN, BRKV and AOEN TMRx_CTRL2: CxIOS and CxCIOS 10: Write protection level 2. The following bits and all bits in level 3 are write protected: TMRx_CCTRL: CxP and CxCP TMRx_BRK: FCSODIS and FCSOEN 11: Write protection level 1. The following bits and all bits in level 2 are write protected: TMRx_CMx: C2OCTRL and C2OBEN Note: Once WPC>0, its content remains frozen until the next system reset.
Bit 7: 0	DTC	0x00	rw	Dead-time configuration This field defines the duration of the dead-time insertion. The 3-bit MSB of DTC[7: 0] is used for function selection: 0xx: DT = DTC [7: 0] * TDTS 10x: DT = (64+ DTC [5: 0]) * TDTS * 2 110: DT = (32+ DTC [4: 0]) * TDTS * 8 111: DT = (32+ DTC [4: 0]) * TDTS * 16

Note: Based on lock configuration, AOEN, BRKV, BRKEN, FCSODIS, FCSOEN and DTC[7:0] can all be write protected. Thus it is necessary to configure write protection when writing to the TMRx_BRK register for the first time.

14.4.4.19 TMR1, TMR8 and TMR20 DMA control register (TMRx_DMACTRL)

Bit	Register	Reset value	Type	Description
Bit 15:13	Reserved	0x0	resd	Kept at its default value.
Bit 12:8	DTB	0x00	rw	DMA transfer bytes This field defines the number of DMA transfers: 00000: 1 byte 00001: 2 bytes 00010: 3 bytes 00011: 4 bytes .... 10000: 17 bytes 10001: 18 bytes
Bit 7:5	Reserved	0x0	resd	Kept at its default value.
Bit 4: 0	ADDR	0x00	rw	DMA transfer address offset ADDR is defined as an offset starting from the address of the TMRx_CTRL1 register: 00000: TMRx_CTRL1 00001: TMRx_CTRL2 00010: TMRx_STCTRL .....

14.4.4.20 TMR1, TMR8 and TMR20 DMA data register (TMRx_DMADT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DMADT	0x0000	rw	DMA data register A write/read operation to the DMADT register accesses any TMR register located at the following address: TMRx peripheral address + ADDR*4 to TMRx peripheral address + ADDR*4 + DTB*4

14.4.4.21 TMR1, TMR8 and TMR20 channel mode register3 (TMRx_CM3)

Bit	Register	Reset value	Type	Description
Bit 15: 6	Reserved	0x000	resd	Kept at its default value.
Bit 7	C5OSEN	0x0	rw	Channel 5 output switch enable
Bit 6: 4	C5OCTRL	0x0	rw	Channel 5 output control
Bit 3	C5OBEN	0x0	rw	Channel 5 output buffer enable
Bit 2	C5OIEN	0x0	rw	Channel 5 output immediately enable
Bit 1: 0	Reserved	0x0	resd	Kept at its default value.

14.4.4.22 TMR1, TMR8 and TMR20 channel 5 data register (TMRx_C5DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C5DT	0x0000	rw	Channel 5 data register C5DT holds the value that is to be compared with the CVAL. Whether the written data will takes effect immediately depends on the C5OBEN bit, and the corresponding output generates on the C5OUT bit.

15 Window watchdog timer (WWDT)

15.1 WWDT introduction

The window watchdog downcounter must be reloaded in a limited time window to prevent the watchdog circuits from generating a system reset. The window watch dog is used to detect the occurrence of system malfunctions.

The window watchdog timer is clocked by a divided APB1_CLK. The precision of the APB1_CLK enables the window watchdog to take accurate control of the limited window.

15.2 WWDT main features

- 7-bit downcounter
- If the watchdog is enabled, a system reset is generated when the value of the downcounter is less than 0x40 or when the downcounter is reloaded outside the window.
- The downcounter can be reloaded by enabling the counter interrupt.

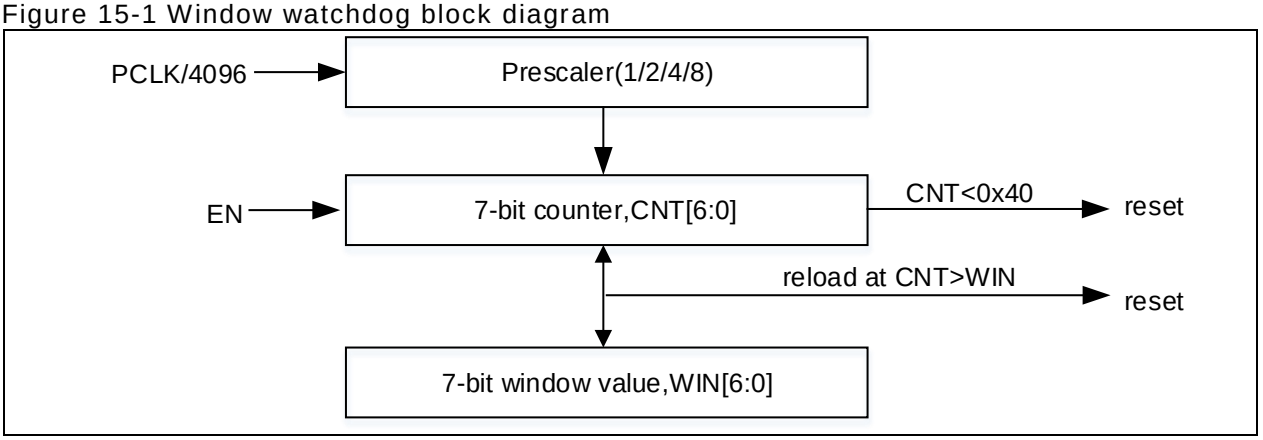
15.3 WWDT functional overview

If the watchdog is enabled, a system reset is generated at the following conditions:

When the 7-bit downcounter scrolls from 0x40 to 0x3F;

When the counter is reloaded while the 7-bit downcounter is greater than the value programmed in the window register.

Figure 15-1 Window watchdog block diagram



To prevent system reset, the counter must be reloaded only when its value is less than the value stored in the window register and greater than 0x40.

The WWDT counter is clocked by a divided APB1_CLK, with the division factor being defined by the DIV[1: 0] bit in the WWDT_CFG register. The counter value determines the maximum counter period before the watchdog generates a reset. The WIN[6: 0] bit can be used to configure the window value.

WWDT offers reload counter interrupt feature. If enabled, the WWDT will set the RLDF flag when the counter value reaches 0x40h, and an interrupt is generated accordingly. The interrupt service routine (ISTS) can be used to reload the counter to prevent a system reset. Note that if CNT[6]=0, setting the WWDTEN bit will generate a system reset, so the CNT[6] bit must be always set (CNT[6]=1) while writing to the WWDT_CTRL register to prevent the occurrence of an immediate reset once the window watchdog is enabled.

The formula to calculate the window watchdog time out:

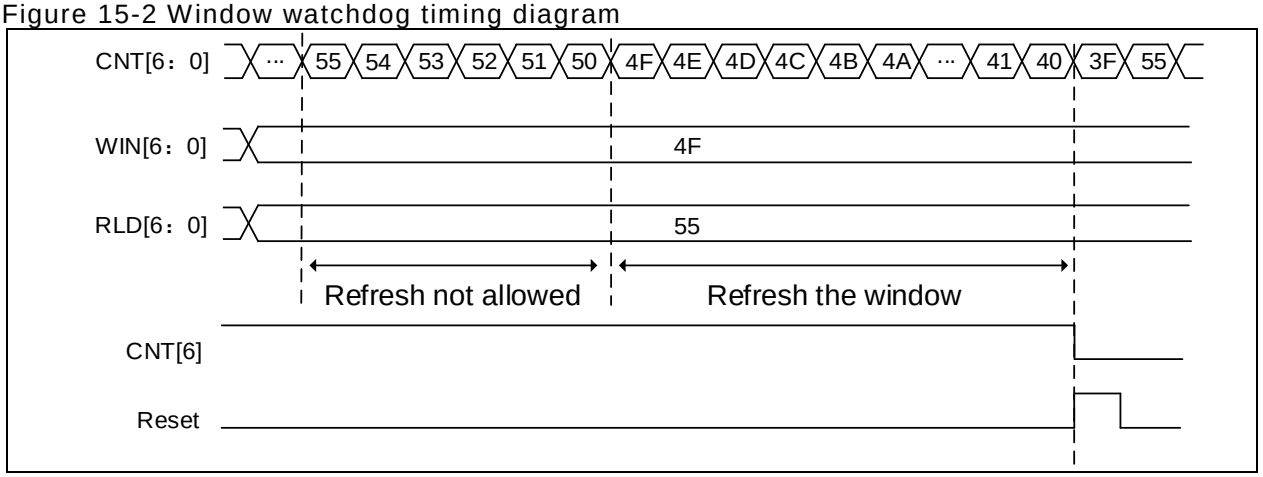
$$T_{WWDT} = T_{PCLK1} \times 4096 \times 2^{DIV[1:0]} \times (CNT[5:0] + 1); (ms)$$

Where: T_{PCLK1} refers to APB1 clock period, in ms.

Table 15-1 Minimum and maximum timeout value when PCLK1=72 MHz

Prescaler	Min. Timeout value	Max. Timeout value
0	56.5μs	3.64ms
1	113.5μs	7.28ms
2	227.5μs	14.56ms
3	455μs	29.12ms

Figure 15-2 Window watchdog timing diagram



15.4 Debug mode

When the microcontroller enters debug mode (Cortex™-M4F core halted), the WWDT counter stops counting by setting the WWDT_PAUSE in the DEBUG module. Refer to Chapter 30.2 for more information.

15.5 WWDT registers

These peripheral registers must be accessed by word (32 bits).

Table 15-2 WWDT register map and reset value

Register name	Offset	Reset value
WWDT_CTRL	0x00	0x7F
WWDT_CFG	0x04	0x7F
WWDT_STS	0x08	0x00

15.5.1 Control register (WWDT_CTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7	WWDTEN	0x0	rw1s	Window watchdog enable 0: Disabled 1: Enabled This bit is set by software, but can be cleared only after reset.
Bit 6: 0	CNT	0x7F	rw	Downcounter When the counter counts down to 0x3F, a reset is generated.

15.5.2 Configuration register (WWDT_CFG)

Bit	Register	Reset value	Type	Description
Bit 31: 10	Reserved	0x000000	resd	Kept at its default value.
Bit 9	RLDIEN	0x0	rw	Reload counter interrupt 0: Disabled 1: Enabled
Bit 8: 7	DIV	0x0	rw	Clock division value 00: PCLK1 divided by 4096 01: PCLK1 divided by 8192 10: PCLK1 divided by 16384

				11: PCLK1 divided by 32768
Bit 6: 0	WIN	0x7F	rw	Window value If the counter is reloaded while its value is greater than the window register value, a reset is generated. The counter must be reloaded between 0x40 and WIN[6: 0].

15.5.3 Status register (WWDT_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 1	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 0	RLDF	0x0	rw0c	Reload counter interrupt flag This flag is set when the downcounter reaches 0x40. 'This bit is set by hardware and cleared by software.

16 Watchdog timer (WDT)

16.1 WDT introduction

The WDT is driven by a dedicated low-speed clock (LICK). Due to the lower clock accuracy of LICK, the WDT is best suited to the applications that have lower timing accuracy and can run independently outside the main application.

16.2 WDT main features

- 12-bit downcounter
- The counter is clocked by LICK (can work in Stop and Standby modes)
- The counter can be configured to stop counting either in Deepsleep or Standby mode
- A system reset is generated under the following circumstances:
  - When the counter value is decremented to 0
  - When the counter is reloaded outside the window

16.3 WDT functional overview

WDT enable:

Both software and hardware operations can be used to enable WDT. In other words, the WDT can be enabled by writing 0xCCCC to the WDT_CMD register; or when the user enables the hardware watchdog through user system data area, the WDT will be automatically enabled after power-on reset.

WDT reset:

When the counter value of the WDT counts down to 0, a WDT reset be generated. Thus the WDT_CMD register must be written with the value 0xAAAA at regular intervals to reload the counter value to avoid the WDT reset.

WDT write-protected:

The WDT_DIV and WDT_RLD registers are write-protected. Writing the value 0x5555 to the WDT_CMD register will unlock write protection. The update status of these two registers are indicated by the DIVF and RLDF bits in the WDT_STS register. If a different value is written to the WDT_CMD register, these two registers will be re-protected. Writing the value 0xAAAA to the WDT_CMD register also enables write protection.

WDT clock:

The WDT counter is clocked by the LICK. The LICK is an internal RC clock with a typical value of 40kHz, with its range falling between 30kHz and 60kHz. The timeout period is also within a certain range, so a margin should be taken into account when configuring timeout period. The LICK can be calibrated to obtain the WDT timeout with a relatively accuracy. For more details, please refer to section 4.1.1.

WDT low power counting mode:

WDT can work in Sleep, Deepsleep and Standby modes. It is possible to stop counting in Deepsleep and Standby modes by setting the nWDT_DEPSLP and nWDT_STDBY bits in the User System Data area.

If the counter is disabled, it will stop decrementing as soon as the Deepsleep and Standby modes are entered. This means that the WDT would not perform a system reset in both low power modes. After waking up from these two modes, it continues downcounting from the value at the time of the entry of these modes.

Figure 16-1 WDT block diagram

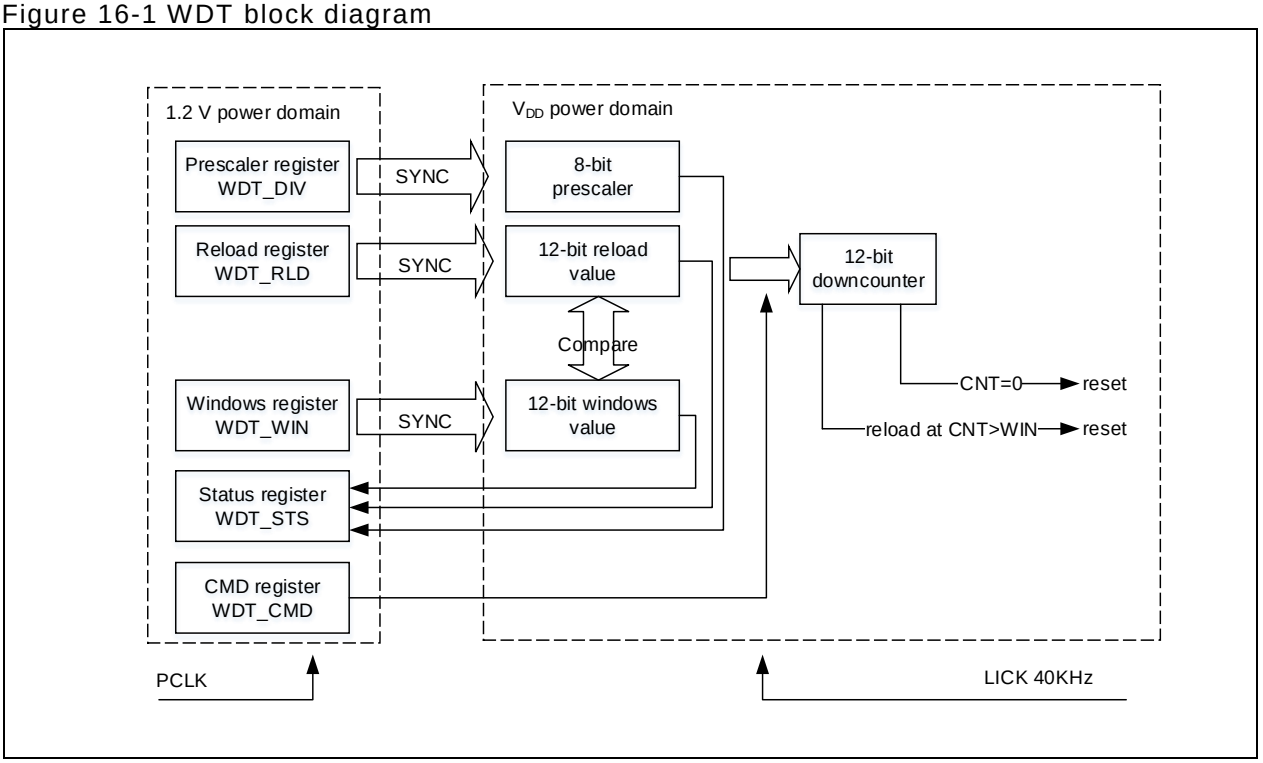


Table 16-1 WDT timeout period (LICK=40kHz)

Prescaler divider	DIV[2: 0] bits	Min.timeout (ms) RLD[11: 0] = 0x000	Max. timeout (ms) RLD[11: 0] = 0xFFF
/4	0	0.1	409.6
/8	1	0.2	819.2
/16	2	0.4	1638.4
/32	3	0.8	3276.8
/64	4	1.6	6553.6
/128	5	3.2	13107.2
/256	(6 or 7)	6.4	26214.4

16.4 Debug mode

When the microcontroller enters debug mode (Cortex™-M4F core halted), the WDT counter stops counting by setting the WDT_PAUSE in the DEBUG module. Refer to Chapter 30.2 for more information.

16.5 WDT registers

These peripheral registers must be accessed by words (32 bits).

Table 16-2 WDT register and reset value

Register name	Offset	Reset value
WDT_CMD	0x00	0x0000 0000
WDT_DIV	0x04	0x0000 0000
WDT_RLD	0x08	0x0000 0FFF
WDT_STS	0x0C	0x0000 0000
WDT_WIN	0x10	0x0000 0FFF

16.5.1 Command register (WDT_CMD)

(Reset in Standby mode)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	CMD	0x0000	wo	Command register 0xAAAA: Reload counter 0x5555: Unlock write-protected WDT_DIV and WDT_RLD 0xCCCC: Enable WDT. If the hardware watchdog has been enabled, ignore this operation.

16.5.2 Divider register (WDT_DIV)

Bit	Register	Reset value	Type	Description
Bit 31: 3	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 2: 0	DIV	0x0	rw	Clock division value 000: LICK divided by 4 001: LICK divided by 8 010: LICK divided by 16 011: LICK divided by 32 100: LICK divided by 64 101: LICK divided by 128 110: LICK divided by 256 111: LICK divided by 256 The write protection must be unlocked in order to enable write access to the register. The register can be read only when DIVF=0.

16.5.3 Reload register (WDT_RLD)

(Reset in Standby mode)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value.
Bit 11: 0	RLD	0xFFF	rw	Reload value The write protection must be unlocked in order to enable write access to the register. The register can be read only when RLDF=0.

16.5.4 Status register (WDT_STS)

(Reset in Standby mode)

Bit	Register	Reset value	Type	Description
Bit 31: 2	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 1	RLDF	0x0	ro	Reload value update complete flag 0: Reload value update complete 1: Reload value update is in process. The reload register WDT_RLD can be written only when RLDF=0
Bit 0	DIVF	0x0	ro	Division value update complete flag 0: Division value update complete 1: Division value update is in process. The divider register WDT_DIV can be written only when DIVF=0

16.5.5 Window register (WDT_WIN)

(Reset in Standby mode)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x000000	resd	Kept at its default value.
Bit 11 : 0	WIN	0xFFF	ro	Window value When the counter value is greater than the window value, the reload counter will perform a reset. The reload counter value falls between 0 and the window value.

17 Enhanced real-time clock (ERTC)

17.1 ERTC introduction

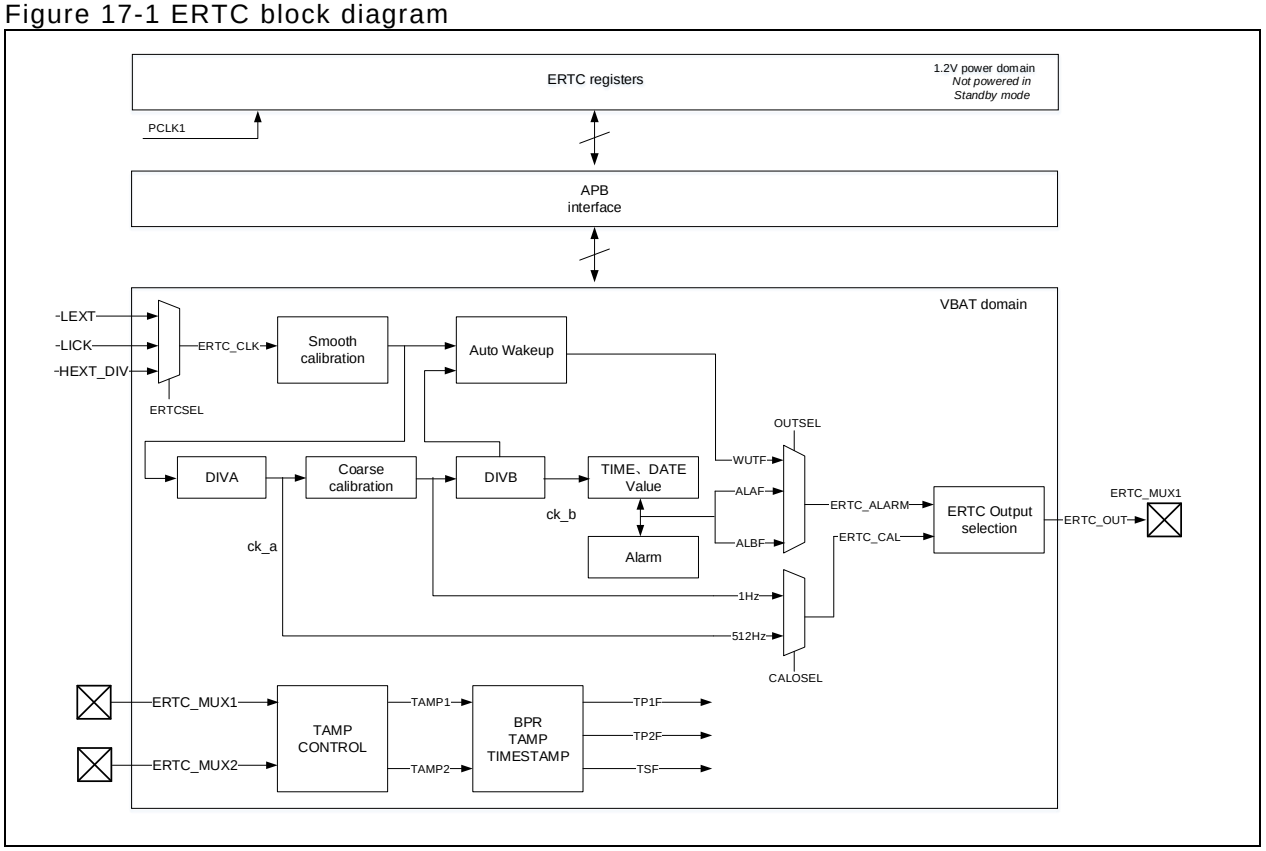
The real-time clock provides a calendar clock function. The time and date can be modified by modifying the ERTC_TIME and ERTC_DATE register.

The RTC module is in battery powered domain, which means that it keeps running and free from the influence of system reset and VDD power off as long as VBAT is powered.

17.2 ERTC main features

- Real-time calendar (automatic processing of month days, including 28 (February in a common year), 29 (February in a leap year), 30 (a lunar month of 30 days) and 31 (a solar month of 31 days), where the current register being a multiple of 4 indicates a leap year), two programmable alarms
- Periodic auto-wakeup
- Reference clock detection
- Two programmable tamper detection, supporting time stamp feature
- Supports fine calibration and coarse calibration
- 20 x battery powered registers
- 5 x interrupts: alarm A, alarm B, periodic auto-wakeup, tamper detection and time stamp
- Multiplexed function output, calibration clock output, alarm event or wakeup event
- Multiplexed function input, reference clock input, two-way tamper detection and time stamp

Figure 17-1 ERTC block diagram



17.3 ERTC function overview

17.3.1 ERTC clock

ERTC clock source (ERTC_CLK) is selected via clock controller from a LEXT, LICK, and a divided HEXT clock (By setting the ERTCSEL[1: 0] in the CRM_BPDC register). The HEXT divider value is configured through the ERTC_DIV[4: 0] bit in the CRM_CFG register.

The ERTC embeds two dividers: A and B, programmed by the DIVA[6: 0] and DIVB[14: 0] respectively. It is recommended that the DIVA is configured to a higher value in order to minimum power consumption. After being divided by prescaler A and B, the ERTC_CLK generates ck_a and ck_b clocks, respectively. The ck_a is used for subsecond update, while the ck_b is used for calendar update and periodic auto wakeup. The clock frequency of ck_a and ck_b can be obtained from the following equation:

$$F_{ck_a} = \frac{f_{ERTC_CLK}}{DIVA + 1}$$

$$F_{ck_b} = \frac{f_{ERTC_CLK}}{(DIVB + 1) \times (DIVA + 1)}$$

To obtain ck_b with frequency of 1 Hz, DIVA=127, DIVB=255, and 32.768 kHz LEXT should be used. This ck_b is then used for calendar update.

Note: To use a divided HEXT by the ERTC_CLK, it is necessary to configure a HEXT divider value before switching a clock source to the HEXT.

17.3.2 ERTC initialization

ERTC register write protection

After a power-on reset, all ERTC registers are write protected. Such protection mechanism is not affected by the system reset. Write access to the ERTC registers (except the ERTC_STS[14: 8], ERTC_TAMP and ERTC_BPRx registers) can be enabled by unlocking it.

To unlick the write protection of ERTC registers, the steps below should be respected:

1. Enable power interface clock by setting PWCEN=1 in the CRM_APB1EN register
2. Unlock write protection of the battery powered domain by setting BPWEN=1 in the PWC_CTRL register
3. Write 0xCA and 0x53 to the ERTC_WP register in sequence. Writing an incorrect key will activate the write protection again.

Table 17-1 lists the ERTC registers that can be configured only after the write protection is unlocked and when the initialization mode is entered.

Table 17-1 RTC register map and reset values

Register	ERTC_WP enabled	Whether to enter initialization mode	Others
ERTC_TIME	Y	Y	-
ERTC_DATE	Y	Y	-
ERTC_CTRL	Y	Bit 7, bit 6 and 4 only	-
ERTC_STS	Y, except [14: 8]	-	-
ERTC_DIV	Y	Y	-
ERTC_WAT	Y	N	Configurable when WATWF=1
ERTC_CCAL	Y	Y	-
ERTC_ALA	Y	N	Configurable when ALAWF =1
ERTC_ALB	Y	N	Configurable when ALAWF =1

ERTC_WP	-	-	-
ERTC_SBS	-	-	-
ERTC_TADJ	Y	N	Configurable when TADJF=0
ERTC_TSTM	-	-	-
ERTC_TSDT	-	-	-
ERTC_TSSBS	-	-	-
ERTC_SCAL	Y	N	Configurable when CALUPDF=0
ERTC_TAMP	N	N	-
ERTC_ALASBS	Y	N	Configurable when ALAWF =1
ERTC_ALBSBS	Y	N	Configurable when ALBWF=1
ERTC_BPRx	N	N	-

Clock and calendar initialization

After the register write protection is unlocked, follow the procedure below for clock and calendar initialization:

1. Set the IMEN bit to enter initialization mode
2. Wait until the initialization flag INITF bit is set
3. Configure DIVB and DIVA.
4. Configure the clock and calendar values.
5. Leave the initialization mode by clearing the IMEN bit. Wait until the UPDF bit is set, indicating the completion of the calendar update. The calendar starts counting.

The ERTC also allows the fine-tuning for daylight saving time and clock.

Daylight saving time feature: It is used to increase (ADD1H=1) or decrease (DEC1H=1) one hour in the calendar, without completing the whole initialization process.

Clock calibration: It is used for the fine calibration of the current clock. If only DECSBS[14: 0] is configured, the value will be added to the DIVB counter and a clock latency will be generated. If only ADD1S bit is set, the current clock will increase by one second. If both DECSBS[14: 0] and ADDIS bit are configured, the clock will increase by a fraction of a second.

Time latency (ADD1S=0): DECSBS/(DIVB+1)

Time advance (ADD1S=1): 1-(DECSBS/(DIVB+1)

Note: To avoid subsecond overflow, SBS[15]=0 must be asserted before setting the ERTC_TADJ register. Reference clock detection and coarse digital calibration cannot be used at the same time. Thus when RCDEN=1, coarse digital calibration is not supported.

Reading the calendar

The ERTC offers two different ways to read the calendar, that is, synchronous read (DREN=0) and asynchronous read (DREN=1).

In the case of DREN=0, the clock and calendar values can be obtained by reading a synchronous shadow register via the PCLK1. The UPDF bit is set each time the shadow register is synchronized with the ERTC calendar value located in the battery powered domain. The synchronization is performed every two ERTC_CLK. The shadow register is reset by a system reset. To ensure consistency between the 3 values (ERTC_SBS, ERTC_TIME and ERTC_DATE registers), reading lower-order registers will lock the values in the higher-order registers until the ERTC_DATE register is read. For example, reading the ERTC_SBS register will lock the values in the ERTC_TIME and ERTC_DATE registers.

In the case of DREN=1, the ERTC will perform direct read access to the ERTC clock and calendar located in the battery powered domain with the PCLK1, avoiding the occurrence of errors caused by

time synchronization. In this mode, the UPDF flag is cleared by hardware. To ensure the data is correct when reading clock and calendar, the software must read the clock and calendar registers twice, and compare the results of two read operations. If the result is not aligned, read again until that the results of two read accesses are consistent. Besides, it is also possible to compare the least significant bits of the two read operations to determine their consistency.

Note: In Standby and Deepsleep modes, the current calendar values are not copied into the shadow registers. When waking up from these two modes, UPDF=0 must be asserted, and then wait until UPDF=1, to ensure that the latest calendar value can be read. In synchronous read (DREN=0) mode, the frequency of the PCLK1 must be at least seven times the ERTC_CLK frequency. In asynchronous read (DREN=1), an additional APB cycle is required to complete the read operations of the calendar register.

Alarm clock initialization

The ERTC contains two programmable alarm clocks: alarm clock A and alarm clock B, and their respective interrupts.

The alarm clock value is programmed with the ERTC_ALASBS/ERTC_ALA (ERTC_ALBSBS/ERTC_ALB). When the programmed alarm value matches the calendar value, an alarm event is generated if an alarm clock is enabled. The MASKx bit can be used to selectively mask calendar fields. The calendar fields, which are masked, are not allocated with an alarm clock.

To configure the alarm clocks, the following steps should be respected:

1. Disable alarm clock A or alarm clock B (by setting ALAEN=0 or ALBEN=0)
2. Wait until the ALAWF or ALBWF bit is set to enable write access to the alarm clock A or B
3. Configure alarm clock A or B registers (ERTC_ALA/ERTC_ALASBS and ERTC_ALB/ERTC_ALBSBS)
4. Enable alarm clock A or B by setting ALAEN=1 or ALBEN=1

Note: If MASK1=0 in the ERTC_ALA or ERTC_ALB, the alarm clock can work normally only when the DIVB value is at least equal to 3.

17.3.3 Periodic automatic wakeup

Periodic automatic wakeup unit is used to wake up ERTC from low power consumption modes automatically. The period is programmed with the VAL[15: 0] bi (When WATCLK[2]=1, it is extended to 17 bits, and the wakeup counter value is VAL+216). When the wakeup counter value drops from the VAL to zero, the WATF bit is set, and a wakeup event is generated, with the wakeup counter being reloaded with the VAL value. An interrupt is also generated if a periodic wakeup interrupt is enabled.

The WATCLK[2: 0] bit can be used to select a wakeup timer clock, including ERTC_CLK/16, ERTC_CLK/8, ERTC_CLK/4, ERTC_CLK/2 and ck_b (usually 1Hz). The cooperation between wakeup timer clocks and wakeup counter values enable users to adjust the wakeup period freely.

To enable a periodic automatic wakeup, the following steps should be respected:

1. Disable a periodic automatic wakeup by setting WATEN=0
2. Wait until WATWF=1 to enable write access to the wakeup reload timer and WATCLK[2: 0]
3. Configure the wakeup timer counter value and wakeup timer through VAL[15: 0] and WATCLK[2: 0] bits
4. Enable a timer by setting WATEN=1

Note: A wakeup timer is not affected by a system reset and low power consumption modes (Sleep, Deepsleep and Standby modes).

Note: In DEBUG mode, if ERTC_CLK is selected as the wakeup timer, the counter for periodic wakeup works normally..

17.3.4 ERTC calibration

Two calibration methods are available: coarse and fine calibration. But the two calibration methods cannot be used together.

Coarse digital calibration:

Coarse digital calibration can be used to advance or delay the calendar updates by increasing or decreasing ck_a cycles.

When positive calibration is enabled (CALDIR=0), 2 ck_a cycles are added every minute (around 15360 ck_a cycles) for the first 2xCALVAL minutes of the 64-minute cycle. This causes the calendar to be updated sooner.

When negative calibration is enabled (CALDIR=1), 1 ck_a cycle is ignored every minute (around ck_a cycles) for the first 2xCALVAL minutes of the 64-minute cycle. This causes the calendar to be updated later.

Note: Coarse digital calibration can work correctly only when the DIVA is 6 or above.

Smooth digital calibration:

Smooth digital calibration has a higher and well-distributed performance than the coarse digital calibration. The calibration is performed by increasing or decreasing ERTC_CLK in an evenly manner.

The smooth digital calibration period is around 220 ERTC_CLK (32 seconds) when the ERTC_CLK is 32.768 kHz. The DEC[8: 0] bit specifies the number of pulses to be masked during the 220 ERTC_CLK cycles. A maximum of 511 pulses can be removed. When the ADD is set, 512 pulses can be inserted during the 220 ERTC_CLK cycles. When DEC[8: 0] and ADD are sued together, a deviation ranging from -511 to +512 ERTC_CLK cycles can be added during the 220 ERTC_CLK cycles.

The effective calibrated frequency (F_{SCAL}):

$$F_{SCAL} = F_{ERTC_CLK} \times [\, 1 + \frac{ADD \times 512 - DEC}{2^{20} + DEC - ADD \times 512} \,]$$

When the divider A is less than 3, the calibration operates as if ADD was equal to 0. The divider B value should be reduced so that each second is accelerated by 8 ERTC_CLK cycles, which means that 256 ERTC_CLK cycles are added every 32 seconds. When DEC[8: 0] and ADD are sued together, a deviation ranging from -255 to +256 ERTC_CLK cycles can be added during the 220 ERTC_CLK cycles.

At this point, the effective calibrated frequency (F_{SCAL})

$$F_{SCAL} = F_{ERTC_CLK} \times [\, 1 + \frac{256 - DEC}{2^{20} + DEC - 256} \,]$$

It is also possible to select 8 or 16-second digital calibration period through the CAL8 and CAL16 bits. The 8-second period takes priority over 16-second. In other words, when both 8-second and 16-second are enabled, 8-second calibration period prevails.

The CALUPDF flag in the ERTC indicates the calibration status. During the configuration of ERTC_SCAL registers, the CALUPDF bit is set, indicating that the calibration value is being updated; Once the calibration value is successfully applied, this bit is cleared automatically, indicating the completion of the calibration value update.

17.3.5 Reference clock detection

The calendar update can be synchronized (not used in low-power modes) to a reference clock (usually the mains 50 or 60 Hz) with a higher precision. This reference clock is used to calibrate the precision of the calendar update frequency (1 Hz)

When it is enabled, the reference clock edge detection is performed during the first 7 ck_a periods around each of the calendar updates. When detected, the edge is used to update calendar values, and 3 ck_a periods are used for subsequent calendar updates. Each time the reference clock edge is detected, the divider A value is forced to reload, making the reference clock and the 1 Hz clock are aligned. If the 1 Hz clock has a slight shift, a more accurate reference clock can be used to fine-tune the 1 Hz clock so that it is aligned with the reference clock. If no reference clock edge is detected, the calendar is updated based on ERTC's original clock source.

Note: Once the reference clock detection is enabled, the DIVA and DIVB must be kept at its respective reset value (0x7F and 0xFF respectively). The clock synchronization cannot be used in conjunction with the coarse digital calibration.

17.3.6 Time stamp function

When time stamp event is detected on the tamper pin (valid edge is detected), the current calendar value will be stored to the time stamp register.

When a time stamp event occurs, the time stamp flag bit (TSF) in the ERTC_STS register will be set. If a new time stamp event is detected when time stamp flag (TSF) is already set, then the time stamp overflow flag (TSOF) will be set, but the time stamp registers will remain the result of the last event. By setting the TSIEN bit, an interrupt can be generated when a time stamp event occurs.

Usage of time stamp:

1. How to enable time stamp when a valid edge is detected on a tamper pin

- Select a time stamp in by setting the TSPIN bit
- Select a rising edge or falling edge to trigger time stamp by setting the TSEDG bit
- Enable time stamp by setting TSEN=1

2. How to save time stamp on a tamper event

- Configure tamper detection registers
- Enable tamper detection time stamp by setting TPTSEN=1

Note: The TSF bit will be set after two ck_a cycles following a time stamp event. It is suggested that users poll TSOF bit when the TSF is set.

17.3.7 Tamper detection

The ERTC has two tamper detection modes: TAMP1 and TAMP2. They can be configured as a level detection with filter or edge detection respectively. TAMP1 can select either ERTC_MUX or ERTC_MUX2 through the TSPIN bit, while the TAMP2 can only select ERTC_MUX2.

The TP1F or TP2F will be set when a valid tamper event is detected. An interrupt will also be generated if a tamper detection interrupt is enabled. If the TPTSEN bit is already set, a time stamp event will be generated accordingly. Once a tamper event occurs, the battery powered registers will be reset so as to ensure data security in the battery powered domain.

How to configure edge detection

1. Select edge detection by setting TPFLT=00, and select a valid edge (either TP1EDG or TP2EDG)
2. According to your needs, configure whether to activate a time stamp on a tamer event (TPTSEN=1)
3. According to your needs, enable a tamper detection interrupt (TPIEN=1)
4. To use TAMP1 mode, select ERTC_MUX1 or ERTC_MUX2 (through the TP1PIN bit), and enable TAMP1 (setting TP1EN=1); To use TAMP2 mode, just need enable TAMP2 (TP2EN=1)

How to configure level detection with filter

1. Select level detection with filter, and valid level sampling times (TPFLT≠00)
2. Select tamper detection valid level (through TP1EDG or TP2EDG)
3. Select tamper detection sampling frequency (through the TPFREQ bit)

4. According to your needs, enable tamper detection pull-up (setting TPPU=1). When TPPU=1 is asserted, tamper detection pre-charge time must be configured through the TPPR bit
5. According to your needs, configure whether to activate a time stamp on a tamper event (TPTSEN=1)
6. According to your needs, enable a tamper interrupt (TPIEN=1)
7. To use TAMP1 mode, select ERTC_MUX1 or ERTC_MUX2 (through the TP1PIN bit), and enable TAMP1 (setting TP1EN=1); To use TAMP2 mode, just need enable TAMP2 (TP2EN=1)

In the case of edge detection mode, the following two points deserve our attention:

1. If a rising edge is configured to enable tamper detection, and the tamper detection pin turns to high level before tamper detection is enabled, then a tamper event will be detected right after the tamper detection is enabled;
2. If a falling edge is configured to enable tamper detection, and the tamper detection pin turns to low level before tamper detection is enabled, then a tamper event will be detected right after the tamper detection is enabled;

Note: Tamper detection is still active even when the VDD power is OFF.

Note: In Standby mode, TAMP2 (PA0 pin) can not use pre-charge function.

17.3.8 Multiplexed function output

ERTC provides a set of multiplexed function output for the following events:

1. Clocks calibrated (OUTSEL=0 and CALOEN=1)
- Output 512Hz (CALOSEL=0)
- Output 1Hz (CALOSEL=1)

2. Alarm clock A (OUTSEL=1)
3. Alarm clock B (OUTSEL=2)
4. Wakeup events (OUTSEL=3)

When alarm clock or wakeup events are selected (OUTSEL≠0), it is possible to select output type (open-drain or push-pull) with the OUTTYPE bit, and output polarity with the OUTP bit.

17.3.9 ERTC wakeup

ERTC can be woken up by alarm clocks, periodic auto wakeup feature, time stamps or tamper events. To enable an ERTC interrupt, configure as follows:

1. Configure the EXINT line corresponding to ERTC interrupts as an interrupt mode and enable it, and select a rising edge
2. Enable a NVIC channel corresponding to ERTC interrupts
3. Enable an ERTC interrupt

Table 17-2 lists the ERTC clock sources, events and interrupts that are able to wakeup low-power modes.

Table 17-2 ERTC low-power mode wakeup

Clock sources	Events	Wake up Sleep	Wake up Deepsleep	Wakeup Standby
HEXT	Alarm clock A	√	×	×
	Alarm clock B	√	×	×
	Periodic automatic wakeup	√	×	×
	Time stamp	√	×	×
	Tamper event	√	×	×
LICK	Alarm clock A	√	√	√
	Alarm clock B	√	√	√
	Periodic automatic wakeup	√	√	√
	Time stamp	√	√	√
	Tamper event	√	√	√
LEXT	Alarm clock A	√	√	√
	Alarm clock B	√	√	√
	Periodic automatic wakeup	√	√	√
	Time stamp	√	√	√
	Tamper event	√	√	√

Table 17-3 Interrupt control bits

Interrupt events	Event flag	Interrupt enable bit	EXINT line
Alarm clock A	ALAF	ALAIEN	17
Alarm clock B	ALBF	ALBIEN	17
Periodic automatic wakeup	WATF	WATIEN	22
Time stamp	TSF	TSIEN	21
Tamper event	TP1F/TP2F	TPIEN	21

17.4 ERTC registers

These peripheral registers must be accessed by words (32 bits).
ERTC registers are 16-bit addressable registers.

Table 17-4 ERTC register map and reset values

Register name	Offset	Reset value
ERTC_TIME	0x00	0x0000 0000
ERTC_DATE	0x04	0x0000 2101
ERTC_CTRL	0x08	0x0000 0000
ERTC_STS	0x0C	0x0000 0007
ERTC_DIV	0x10	0x007F 00FF
ERTC_WAT	0x14	0x0000 FFFF
ERTC_CCAL	0x18	0x0000 0000
ERTC_ALA	0x1C	0x0000 0000
ERTC_ALB	0x20	0x0000 0000
ERTC_WP	0x24	0x0000 0000
ERTC_SBS	0x28	0x0000 0000
ERTC_TADJ	0x2C	0x0000 0000
ERTC_TSTM	0x30	0x0000 0000
ERTC_TSDT	0x34	0x0000 000D

ERTC_TSSBS	0x38	0x0000 0000
ERTC_SCAL	0x3C	0x0000 0000
ERTC_TAMP	0x40	0x0000 0000
ERTC_ALASBS	0x44	0x0000 0000
ERTC_ALBSBS	0x48	0x0000 0000
ERTC_BPRx	0x50-0x9C	0x0000 0000

17.4.1 ERTC time register (ERTC_TIME)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value.
Bit 22	AMPM	0x0	rw	AM/PM 0: AM 1: PM Note: This bit is applicable for 12-hr format only. It is 0 for 24-hr format instead.
Bit 21: 20	HT	0x0	rw	Hour tens
Bit 19: 16	HU	0x0	rw	Hour units
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14: 12	MT	0x0	rw	Minute tens
Bit 11: 8	MU	0x0	rw	Minute units
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6: 4	ST	0x0	rw	Second tens
Bit 3: 0	SU	0x0	rw	Second units

17.4.2 ERTC date register (ERTC_DATE)

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23: 20	YT	0x0	rw	Year tens
Bit 19: 16	YU	0x0	rw	Year units
Bit 15: 13	WK	0x1	rw	Week day 0: Forbidden 1: Monday 2: Tuesday 3: Wednesday 4: Thursday 5: Friday 6: Saturday 7: Sunday
Bit 12	MT	0x0	rw	Month tens
Bit 11: 8	MU	0x1	rw	Month units
Bit 7: 6	Reserved	0x0	resd	Kept at its default value.
Bit 5: 4	DT	0x0	rw	Date tens
Bit 3: 0	DU	0x1	rw	Date units

17.4.3 ERTC control register (ERTC_CTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23	CALOEN	0x0	rw	Calibration output enable 0: Calibration output disabled 1: Calibration output enabled
Bit 22: 21	OUTSEL	0x0	rw	Output source selection 00: Output source disabled

				01: Alarm clock A 10: Alarm clock B 11: Wakeup events
Bit 20	OUTP	0x0	rw	Output polarity 0: High 1: Low
Bit 19	CALOSEL	0x0	rw	Calibration output selection 0: 512Hz 1: 1Hz
Bit 18	BPR	0x0	rw	Battery powered domain data register This bit in the battery powered domain is not affected by a system reset. It is used to store the daylight saving time change or others that need to be saved permanently.
Bit 17	DEC1H	0x0	wo	Decrease 1 hour 0: No effect 1: Subtract 1 hour Note: This bit is applicable only when the current hour is not 0. The next second takes effect when this bit is set (don't set this bit when the hour is being incremented)
Bit 16	ADD1H	0x0	wo	Add 1 hour 0: No effect 1: Add 1 hour Note: The next second takes effect when this bit is set (don't set this bit when the hour is being incremented)
Bit 15	TSIEN	0x0	rw	Timestamp interrupt enable 0: Timestamp interrupt disabled 1: Timestamp interrupt enabled
Bit 14	WATIEN	0x0	rw	Wakeup timer interrupt enable 0: Wakeup timer interrupt disable 1: Wakeup timer interrupt enabled
Bit 13	ALBIEN	0x0	rw	Alarm B interrupt enable 0: Alarm B interrupt disabled 1: Alarm B interrupt enabled
Bit 12	ALAIEN	0x0	rw	Alarm A interrupt enable 0: Alarm A interrupt disabled 1: Alarm A interrupt enabled
Bit 11	TSEN	0x0	rw	Timestamp enable 0: Timestamp disabled 1: Timestamp enabled
Bit 10	WATEN	0x0	rw	Wakeup timer enable 0: Wakeup timer disabled 1: Wakeup timer enabled
Bit 9	ALBEN	0x0	rw	Alarm B enable 0: Alarm B disabled 1: Alarm B enabled
Bit 8	ALAEN	0x0	rw	Alarm A enable 0: Alarm A disabled 1: Alarm A enabled
Bit 7	CCALEN	0x0	rw	Coarse calibration enable 0: Coarse calibration disabled 1: Coarse calibration enabled
Bit 6	HM	0x0	rw	Hour mode 0: 24-hour format 1: 12-hour format
Bit 5	DREN	0x0	rw	Date/time register direct read enable 0: Date/time register direct read disabled. ERTC_TIME, ERTC_DATE and ERTC_SBS values are taken from the synchronized registers, which are updated once every two ERTC_CLK cycles 1: Date/time register direct read enabled. ERTC_TIME, ERTC_DATE and ERTC_SBS values are taken from the battery powered domain.
Bit 4	RCDEN	0x0	rw	Reference clock detection enable 0: Reference clock detection disabled

				1: Reference clock detection enabled
Bit 3	TSEDG	0x0	rw	Timestamp trigger edge 0: Rising edge 1: Falling edge
Bit 2: 0	WATCLK	0x0	rw	Wakeup timer clock selection 000: ERTC_CLK/16 001: ERTC_CLK/8 010: ERTC_CLK/4 011: ERTC_CLK/2 10x: ck_b 11x: ck_b is selected. 2^{16} is added to the wakeup counter value, and wakeup time =ERTC_WAT+2^{16}. *Note: The write access to this field is supported when WATEN=0 and WATWF=1.*

17.4.4 ERTC initialization and status register (ERTC_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	CALUPDF	0x0	ro	Calibration value update complete flag 0: Calibration value update is complete 1: Calibration value update is in progress This bit is automatically set when software writes to the ERTC_SCAL register. It is automatically cleared when a new calibration value is taking into account. When this bit is set, the write access to the ERTC_SCAL register is not allowed.
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	TP2F	0x0	rw0c	Tamper detection 2 flag 0: No tamper event 1: Tamper event occurs
Bit 13	TP1F	0x0	rw0c	Tamper detection 1 flag 0: No tamper event 1: Tamper event occurs
Bit 12	TSOF	0x0	rw0c	Timestamp overflow flag 0: No timestamp overflow 1: Timestamp overflow occurs If a new time stamp event is detected when time stamp flag (TSF) is already set, this bit will be set by hardware.
Bit 11	TSF	0x0	rw0c	Timestamp flag 0: No timestamp event 1: Timestamp event occurs It is recommended to double check the TSOF flag after reading a timestamp and clearing the TSF. Otherwise, a new timestamp event may be detected while clearing the TSF. *Note: The clearing operation of this bit takes effect after two APB_CLK cycles.*
Bit 10	WATF	0x0	rw0c	Wakeup timer flag 0: No wakeup timer event 1: Wakeup timer event occurs *Note: The clearing operation of this bit takes effect after two APB_CLK cycles.*
Bit 9	ALBF	0x0	rw0c	Alarm clock B flag 0: No alarm clock event 1: Alarm clock event occurs *Note: The clearing operation of this bit takes effect after two APB_CLK cycles.*
Bit 8	ALAF	0x0	rw0c	Alarm clock A flag 0: No alarm clock event 1: Alarm clock event occurs *Note: The clearing operation of this bit takes effect after two APB_CLK cycles.*
Bit 7	IMEN	0x0	rw	Initialization mode enable 0: Initialization mode disabled

				1: Initialization mode enabled When an initialization mode is entered, the calendar stops running.
Bit 6	IMF	0x0	ro	Enter initialization mode flag 0: Initialization mode is not entered 1: Initialization mode is entered The ERTC_TIME, ERTC_DATE and ERTC_DIV registers can be modified only when an initialization mode is enabled (INITEN=1) and entered (INITEF=1).
Bit 5	UPDF	0x0	rw0c	Calendar update flag 0: Calendar update is in progress 1: Calendar update is complete The UPDF bit is set each time the shadow register is synchronized with the ERTC calendar value located in the battery powered domain. The synchronization is performed every two ERTC_CLK.
Bit 4	INITF	0x0	ro	Calendar initialization flag 0: Calendar has not been initialized 1: Calendar has been initialized This bit is set when the calendar year filed (ERTC_DATE) is different from 0. It is cleared when the year is 0.
Bit 3	TADJF	0x0	ro	Time adjustment flag 0: No time adjustment 1: Time adjustment is in progress This bit is automatically set when a write access to the ERTC_TADJ register is performed. It is automatically cleared at the end of time adjustment.
Bit 2	WATWF	0x1	ro	Wakeup timer register allows write flag 0: Wakeup timer register configuration not allowed 1: Wakeup timer register configuration allowed
Bit 1	ALBWF	0x1	ro	Alarm B register allows write flag 0: Alarm B register write operation not allowed 1: Alarm B register write operation allowed
Bit 0	ALAWF	0x1	ro	Alarm A register allows write flag 0: Alarm A register write operation not allowed 1: Alarm A register write operation allowed

17.4.5 ERTC divider register (ERTC_DIV)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value.
Bit 22: 16	DIVA	0x7F	rw	Divider A
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14: 0	DIVB	0x00FF	rw	Divider B Calendar clock = ERTC_CLK/((DIVA+1)x(DIVB+1))

17.4.6 ERTC wakeup timer register (ERTC_WAT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	VAL	0xFFFF	rw	Wakeup timer reload value

17.4.7 ERTC coarse calibration register (ERTC_CCAL)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7	CALDIR	0x0	rw	Calibration direction 0: Positive calibration 1: Negative calibration
Bit 6: 5	Reserved	0x0	resd	Kept at its default value.
Bit 4: 0	CALVAL	0x00	rw	Calibration value Positive calibration

				00000: +0 ppm 00001: +4 ppm 00010: +8 ppm 11111: +126 ppm Negative calibration 00000: -0 ppm 00001: -2 ppm 00010: -4 ppm … 11111: - 63 ppm

17.4.8 ERTC alarm clock A register (ERTC_ALA)

Bit	Register	Reset value	Type	Description
Bit 31	MASK4	0x0	rw	Date/week day mask 0: Date/week day is not masked 1: Alarm clock doesn't care about date/week day
Bit 30	WKSEL	0x0	rw	Date/week day select 0: Date 1: Week day (DT[1: 0] is not used)
Bit 29: 28	DT	0x0	rw	Date tens
Bit 27: 24	DU	0x0	rw	Date/week day units
Bit 23	MASK3	0x0	rw	Hour mask 0: No hour mask 1: Alarm clock doesn't care about hours
Bit 22	AMPM	0x0	rw	AM/PM 0: AM 1: PM *Note: This bit is applicable for 12-hour format only. It is 0 for 24-hour format.*
Bit 21: 20	HT	0x0	rw	Hour tens
Bit 19: 16	HU	0x0	rw	Hour units
Bit 15	MASK2	0x0	rw	Minute mask 0: No minute mask 1: Alarm clock doesn't care about minutes
Bit 14: 12	MT	0x0	rw	Minute tens
Bit 11: 8	MU	0x0	rw	Minute units
Bit 7	MASK1	0x0	rw	Second mask 0: No second mask 1: Alarm clock doesn't care about seconds
Bit 6: 4	ST	0x0	rw	Second tens
Bit 3: 0	SU	0x0	rw	Second units

17.4.9 ERTC alarm clock B register (ERTC_ALB)

Bit	Register	Reset value	Type	Description
Bit 31	MASK4	0x0	rw	Date/week day mask 0: Date/week day is not masked 1: Alarm clock doesn't care about date/week day
Bit 30	WKSEL	0x0	rw	Date/week day select 0: Date 1: Week day (DT[1: 0] is not used)
Bit 29: 28	DT	0x0	rw	Date tens
Bit 27: 24	DU	0x0	rw	Date/week day units
Bit 23	MASK3	0x0	rw	Hour mask 0: No hour mask 1: Alarm clock doesn't care about hours
Bit 22	AMPM	0x0	rw	AM/PM 0: AM 1: PM *Note: This bit is applicable for 12-hour format only. It is 0 for 24-hour format.*
Bit 21: 20	HT	0x0	rw	Hour tens

Bit 19: 16	HU	0x0	rw	Hour units
Bit 15	MASK2	0x0	rw	Minute mask 0: No minute mask 1: Alarm clock doesn't care about minutes
Bit 14: 12	MT	0x0	rw	Minute tens
Bit 11: 8	MU	0x0	rw	Minute units
Bit 7	MASK1	0x0	rw	Second mask 0: No second mask 1: Alarm clock doesn't care about seconds
Bit 6: 4	ST	0x0	rw	Second tens
Bit 3: 0	SU	0x0	rw	Second units

17.4.10 ERTC write protection register (ERTC_WP)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value
Bit 7: 0	CMD	0x00	wo	Command register All ERTC register write protection is unlocked by writing 0xCA and 0x53. Writing any other value will re-activate write protection.

17.4.11 ERTC subsecond register (ERTC_SBS)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 0	SBS	0x0000	ro	Sub-second value Subsecond is the value in the DIVB counter. Clock frequency = ERTC_CLK/(DIVA+1)

17.4.12 ERTC time adjustment register (ERTC_TADJ)

Bit	Register	Reset value	Type	Description
Bit 31	ADD1S	0x0	wo	Add 1 second 0: No effect 1: Add one second This bit can be written only when TADJF=0. It is intended to be used with DECSBS in order to fine-tune the time.
Bit 30: 15	Reserved	0x0000	resd	Kept at its default value
Bit 14: 0	DECSBS	0x0000	wo	DECSBS[14: 0]: Decrease sub-second value Delay (ADD1S=0): Delay = DECSBS/(DIVB+1) Advance (ADD1S=1) : Advance =1-(DECSBS/(DIVB+1))

17.4.13 ERTC time stamp time register (ERTC_TSTM)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value
Bit 22	AMPM	0x0	ro	AM/PM 0: AM 1: PM *Note: This bit is applicable for 12-hour format only. It is 0 for 24-hour format.*
Bit 21: 20	HT	0x0	ro	Hour tens
Bit 19: 16	HU	0x0	ro	Hour units
Bit 15	Reserved	0x0	resd	Kept at its default value
Bit 14: 12	MT	0x0	ro	Minute tens
Bit 11: 8	MU	0x0	ro	Minute units
Bit 7	Reserved	0x0	resd	Kept at its default value
Bit 6: 4	ST	0x0	ro	Second tens
Bit 3: 0	SU	0x0	ro	Second units

Note: The content of this register is valid only when the TSF is set in the ERTC_STS register. It is cleared when TSF bit is reset.

17.4.14 ERTC time stamp date register (ERTC_TSDT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 13	WK	0x0	ro	Week day
Bit 12	MT	0x0	ro	Month tens
Bit 11: 8	MU	0x0	ro	Month units
Bit 7: 6	Reserved	0x0	resd	Kept at its default value
Bit 5: 4	DT	0x0	ro	Date tens
Bit 3: 0	DU	0x0	ro	Date units

Note: The content of this register is valid only when the TSF is set in the ERTC_STS register. It is cleared when TSF bit is reset.

17.4.15 ERTC time stamp subsecond register (ERTC_TSSBS)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 0	SBS	0x0000	ro	Sub-second value

Note: The content of this register is valid only when the TSF is set in the ERTC_STS register. It is cleared when TSF bit is reset.

17.4.16 ERTC smooth calibration register (ERTC_SCAL)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15	ADD	0x0	rw	Add ERTC clock 0: No ERTC clock added 1: One ERTC_CLK is inserted every 2^{11} ERTC_CLK cycles
Bit 14	CAL8	0x0	rw	8-second calibration period 0: No effect 1: 8-second calibration
Bit 13	CAL16	0x0	rw	16 second calibration period 0: No effect 1: 16-second calibration
Bit 12: 9	Reserved	0x0	resd	Kept at its default value
Bit 8: 0	DEC	0x000	rw	Decrease ERTC clock DEC out of ERTC_CLK cycles are masked during the 2^{20} ERTC_CLK periods. This bit is usually used with ADD. When the ADD is set, the actual number of ERTC_CLK is equal to 2^{20}+512-DEC during the 2^{20} ERTC_CLK periods.

17.4.17 ERTC tamper configuration register (ERTC_TAMP)

Bit	Register	Reset value	Type	Description
Bit 31: 19	Reserved	0x0000	resd	Kept at its default value
Bit 18	OUTTYPE	0x0	rw	Output type 0: Open-drain output 1: Push-pull output
Bit 17	TSPIN	0x0	rw	Time stamp detection pin selection 0: ERTC_MUX1 1: ERTC_MUX2
Bit 16	TP1PIN	0x0	rw	Tamper detection pin selection 0: ERTC_MUX1 1: ERTC_MUX2
Bit 15	TPPU	0x0	rw	Tamper detection pull-up 0: Tamper detection pull-up enabled

				1: Tamper detection pull-up disabled
Bit 14: 13	TPPR	0x0	rw	Tamper detection pre-charge time 0: 1 ERTC_CLK cycle 1: 2 ERTC_CLK cycles 2: 4 ERTC_CLK cycles 3: 8 ERTC_CLK cycles
Bit 12: 11	TPFLT	0x0	rw	Tamper detection filter time 0: No filter 1: Tamper is detected after 2 consecutive samples 2: Tamper is detected after 4 consecutive samples 3: Tamper is detected after 8 consecutive samples
Bit 10: 8	TPFREQ	0x0	rw	Tamper detection frequency 0: ERTC_CLK/32768 1: ERTC_CLK/16384 2: ERTC_CLK/8192 3: ERTC_CLK/4096 4: ERTC_CLK/2048 5: ERTC_CLK/1024 6: ERTC_CLK/512 7: ERTC_CLK/256
Bit 7	TPTSEN	0x0	rw	Tamper detection timestamp enable 0: Tamper detection timestamp disabled 1: Tamper detection timestamp enabled. Save timestamp on a tamper event.
Bit 6: 5	Reserved	0x0	resd	Kept at its default value
Bit 4	TP2EDG	0x0	rw	Tamper detection 2 valid edge If TPFLT=0: 0: Rising edge 1: Falling edge If TPFLT>0: 0: Low 1: High
Bit 3	TP2EN	0x0	rw	Tamper detection 2 enable 0: Tamper detection 2 disabled 1: Tamper detection 2 enabled
Bit 2	TPIEN	0x0	rw	Tamper detection interrupt enable 0: Tamper detection interrupt disabled 1: Tamper detection interrupt enabled
Bit 1	TP1EDG	0x0	rw	Tamper detection 1 valid edge If TPFLT=0: 0: Rising edge 1: Falling edge If TPFLT>0: 0: Low 1: High
Bit 0	TP1EN	0x0	rw	Tamper detection 1 enable 0: Tamper detection 1 disabled 1: Tamper detection 1 enabled

17.4.18 ERTC alarm clock A subsecond register (ERTC_ALASBS)

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0	resd	Kept at its default value
Bit 27: 24	SBSMSK	0x0	rw	Sub-second mask 0: No comparison. Alarm A doesn't care about subseconds. 1: SBS[0] is compared 2: SBS[1: 0] are compared 3: SBS[2: 0] are compared ... 14: SBS[13: 0] are compared 15: SBS[14: 0] are compared
Bit 23: 15	Reserved	0x000	rw	Kept at its default value
Bit 14: 0	SBS	0x0000	rw	Sub-second value

17.4.19 ERTC alarm clock B subsecond register (ERTC_ALBSBS)

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0	resd	Kept at its default value
Bit 27: 24	SBSMSK	0x0	rw	Sub-second mask 0: No comparison. Alarm B doesn't care about subseconds. 1: SBS[0] is compared 2: SBS[1: 0] are compared 3: SBS[2: 0] are compared ... 14: SBS[13: 0] are compared 15: SBS[14: 0] are compared
Bit 23: 15	Reserved	0x000	rw	Kept at its default value
Bit 14: 0	SBS	0x0000	rw	Sub-second value

17.4.20 ERTC battery powered domain data register (ERTC_BPRx)

Bit	Register	Reset value	Type	Description
Bit 31: 0	DT	0x0000 0000	rw	Battery powered domain data BPR_DTx registers are powered on by V_{BAT} so that they are not reset by a system reset. They are reset on a tamper event or when a battery powered domain is reset.

18 Analog-to-digital converter (ADC)

18.1 ADC introduction

The ADC is a peripheral that converts an analog input signal into a 12-bit/10-bit/8-bit/6-bit digital signal. Its sampling rate is as high as 5.33 MSPS. It has up to 19 channels for sampling and conversion.

18.2 ADC main features

In terms of analog:

- 12-bit, 10-bit, 8-bit or 6-bit configurable resolution
- Self-calibration time: 205 ADC clock cycles
- ADC conversion time
  - Fast channel: ADC conversion time is 0.1875 µs at 80 MHz in 12-bit resolution (5.33MSps)
  - Slow channel: ADC conversion time is 0.2375 µs at 80 MHz in 12-bit resolution (4.21MSps)
  - Fast channel: ADC conversion time is 0.1126 µs at 80 MHz in 6-bit resolution (8.88MSps)
  - Slow channel: ADC conversion time is 0.1626 µs at 80 MHz in 6-bit resolution (6.15MSps)
- ADC supply requirement: Refer to AT32F435/437 datasheet for more information
- ADC input range: $V_{REF-} \leq V_{IN} \leq V_{REF+}$

In terms of digital control:

- Regular channels and preempted channels with different priority
- Regular channels and preempted channels both have their own trigger detection circuit
- Each channel can independently define its own sampling time
- Conversion sequence supports various conversion modes
- Oversampling
- Optional data alignment mode
- Programmable voltage monitor threshold
- Regular channels with DMA transfers
- Interrupt generation at one of the following events:
  - Conversion overflow of regular channels
  - End of the conversion of preempted channels
  - End of the conversion of regular channels
  - Voltage outside the threshold programmed
- ADC master/slave modes
- Master/slave shift mode with programmable timing shift length between ADCs
  - Master-slave mode with DMA is suited to high-performance requirements

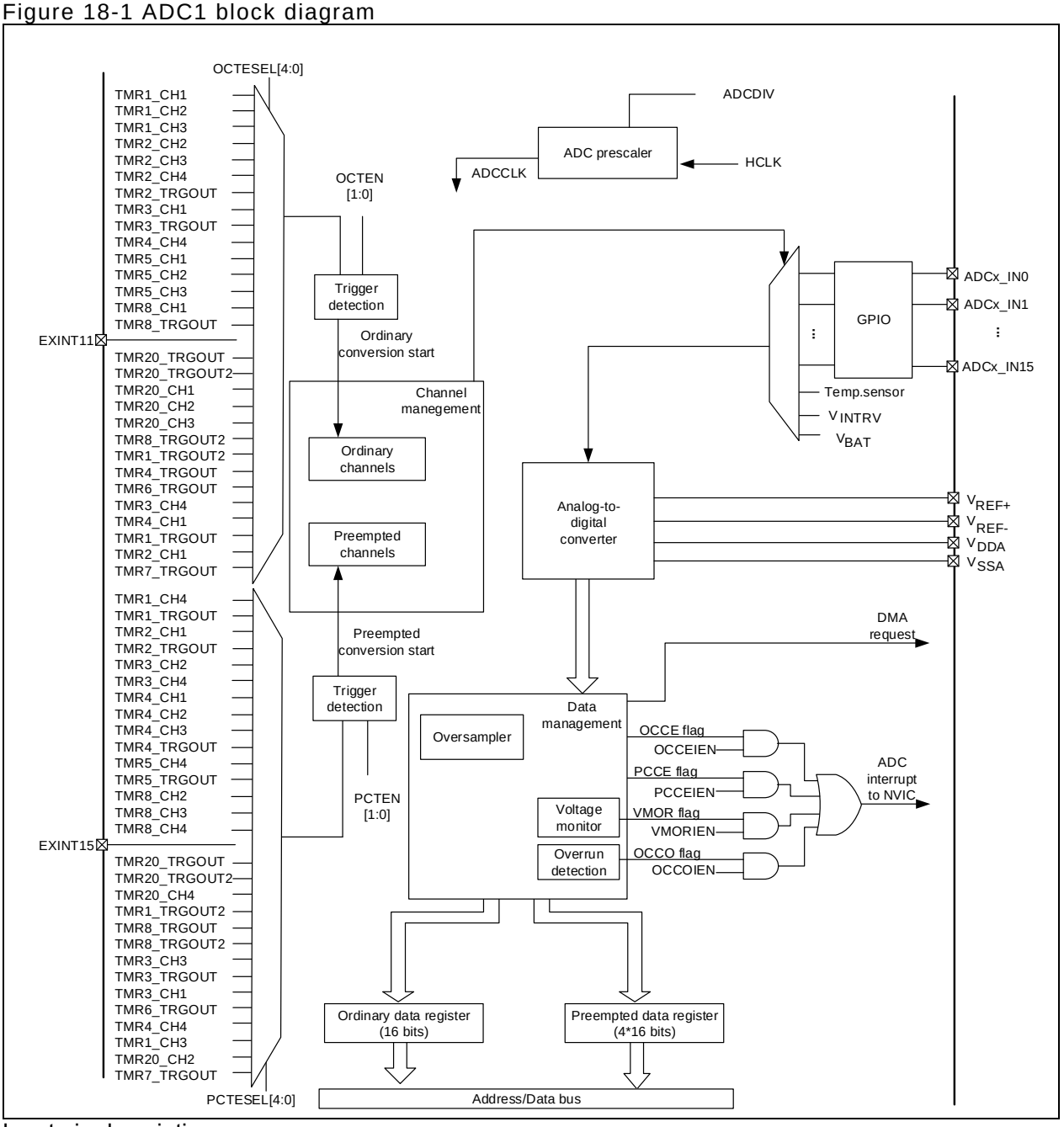
18.3 ADC structure

Figure 18-1 shows the block diagram of ADC1.

Differences between ADC2/ADC3 and ADC1:

1. ADC2/ADC3 are not connected to the internal temperature sensor (Temp. sensor) and internal reference voltage (V_{INTRV}) and battery voltage (V_{BAT}).

Figure 18-1 ADC1 block diagram



Input pin description:

- V_{DDA}: Analog supply, ADC analog supply
- V_{SSA}: Analog supply ground, ADC analog supply ground
- V_{REF+}: Analog reference positive, high/positive reference voltage for the ADC
- V_{REF-}: Analog reference negative, low/negative reference voltage for the ADC
- ADCx_IN: Analog input signal channels

Refer to the Datasheet for more information about the input pin connections and voltage ranges.

18.4 ADC functional overview

18.4.1 Channel management

Analog signal channel input:

There are 19 analog signal channel inputs for each of the ADCs, expressed by ADC_Inx (x=0 to 18).

- ADC1_IN0 to ADC1_IN15 are referred to as the external analog input, ADC1_IN16 as an internal temperature sensor, ADC1_IN17 as an internal reference voltage, and ADC1_IN18 as a battery voltage.
- ADC2_IN0 to ADC2_IN15 are referred to as the external analog input, and ADC2_IN16 and ADC2_IN17 as Vss, and ADC1_IN18 as V_{REF-}.
- ADC3_IN0 to ADC3_IN3, and ADC3_IN10 to ADC3_IN13 are referred to as the external analog input, and ADC3_IN16 and ADC3_IN17 as Vss, and ADC3_IN18 as V_{REF-}.

Channel conversion

The conversions are divided into two groups: ordinary and preempted channels. The preempted group has priority over the ordinary group.

If the preempted channel trigger occurs during the ordinary channel conversion, then the ordinary channel conversion is interrupted, giving the priority to the preempted channel, and the ordinary channel continues its conversion at the end of the preempted channel conversion. If the ordinary channel trigger occurs during the preempted channel conversion, the ordinary channel conversion won't start until the end of the preempted channel conversion.

Program the ADC_Inx into the ordinary channel sequence (ADC_OSQx) and the preempted channel sequence (ADC_PSQ), and the same channel can be repeated, the total number of sequences is determined by OCLEN and PCLEN, then it is ready to enable the ordinary channel or preempted channel conversion.

18.4.1.1 Internal temperature sensor

The temperature sensor is connected to ADC1_IN16. Before the temperature sensor channel conversion, it is required to enable the ITSRVEN bit in the ADC_CCTRL register and wait after power-on time.

Obtain the temperature using the following formula:

Temperature (in °C) = {(V 25 - V_{SENSE}) / Avg_Slope} + 25.

Where,

V_{25} = V_{SENSE} value for 25°C and

Avg_Slope = Average Slope for curve between Temperature vs. V_{SENSE} (given in mV/°C).

18.4.1.2 Internal reference voltage

The internal reference voltage of the typical value 1.2 V is connected to ADC1_IN17. It is required to enable the ITSRVEN bit in the ADC_CCTRL register before the internal reference channel conversion. The converted data of such channel can be used to calculate the external reference voltage.

18.4.1.3 Battery voltage

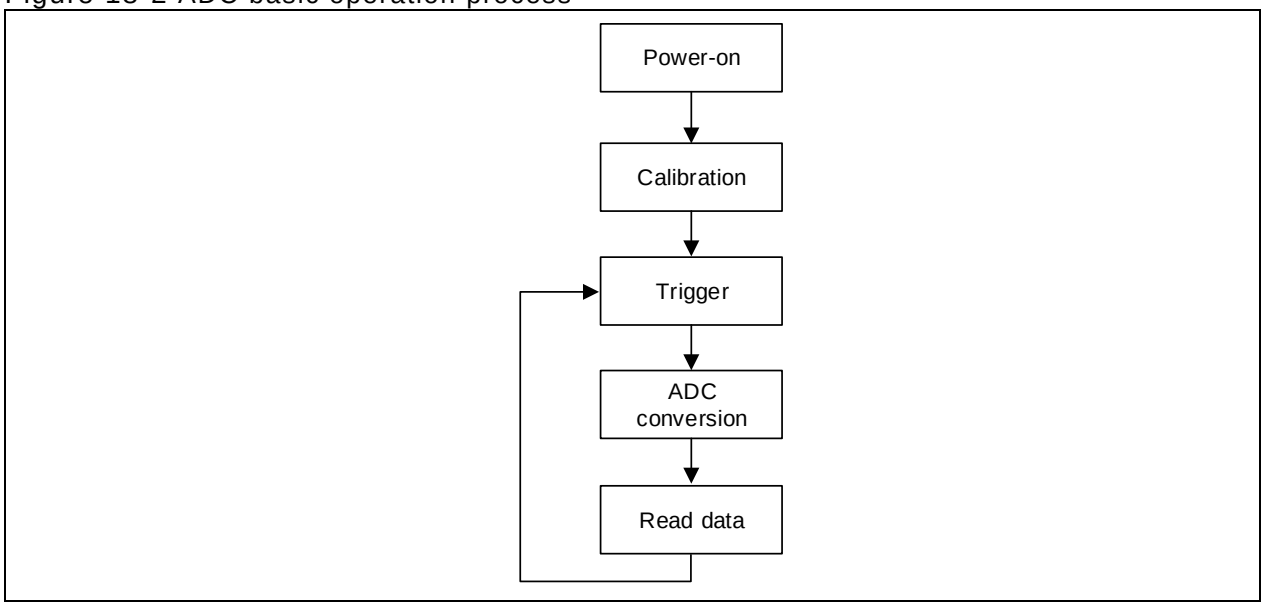
V_{BAT} is higher than V_{DDA}. Thus the V_{BAT} must be divided by 4 before being connected to the ADC1_IN18. The conversion of battery voltage channels starts only when the VBATEN bit is set in the ADC_CCTRL register in order to power the divided-by-4 circuit.

Note: When the ADC is used for sampling VBAT voltage, the VBAT keeps consuming as along as the VBATEN is set. Thus it is recommended to enable VBAT only during the sampling period, and disable it at the end of the sampling in order to lower power consumption.

18.4.2 ADC operation process

Figure 18-2 shows the basic operation process of the ADC. It is recommended to do the calibration after the initial power-on in order to improve the accuracy of sampling and conversion. After the calibration, trigger is used to enable ADC sampling and conversion. Read data at the end of the conversion.

Figure 18-2 ADC basic operation process



18.4.2.1 Power-on and calibration

Power-on

Set the ADCxEN bit in the CRM_APB2EN register to enable ADC clocks: PCLK2 and ADCCLK.

Program the desired ADCCLK frequency by setting the ADCDIV bit in the ADC_CCTRL register. The ADCCLK is derived from HCLK frequency division.

Note: ADCCLK must be less than 80 MHz, while the ADCCLK frequency must be lower than PCLK2.

Then set the ADCEN bit in the ADC_CTRL2 register to supply the ADC, and wait until the RDY flag is set before starting ADC conversion. Clear the ADCEN bit will stop the ADC conversion and result in a reset. In the meantime, the ADC is switched off to save power.

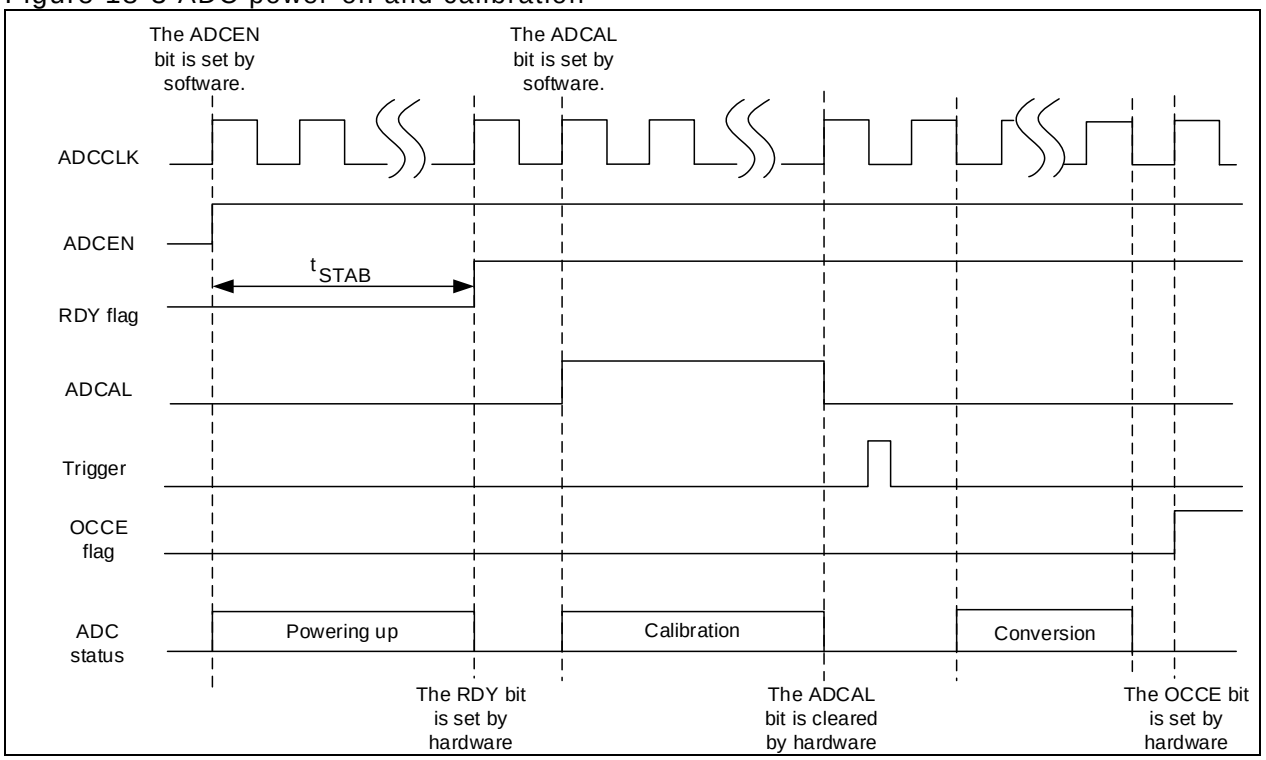
Calibration

After power-on, the calibration is enabled by setting the ADCAL bit in the ADC_CTRL2 register. When the calibration is complete, the ADCAL bit is cleared by hardware and the conversion is performed by software trigger.

After each calibration, the calibration value is stored in the least significant 7 bits of the ADC_CALVAL register, and then value is automatically sent back to the ADC so as to eliminate capacitance errors. The storage of the calibration value will not set the OCCE flag, or generate interrupts or DMA requests.

Note:Calibration is permitted only in 12-bit resolution mode. Switching resolution can be made only after the completion of calibration. Then, it is necessary to wait until the RDY flag is set before trigger operation.

Figure 18-3 ADC power-on and calibration



18.4.2.2 Trigger

The ADC triggers contain ordinary channel trigger and preempted channel trigger. The ordinary channel conversion is triggered by ordinary channel triggers while the preempted channel conversion is triggered by preempted ones. The valid polarity for external trigger sources can be selected by the OCETE and PCETE bits in the ADC_CTRL2 register. The ADC starts conversion after a trigger source is detected.

The conversion can be triggered by software write operation to the OCSWTRG and PCSWTRG bits in the ADC_CTRL2 register, or by an external event. The external events include timer and pin triggers. The OCTESEL and PCTESEL bits in the ADC_CTRL2 register are used to select specific trigger sources, as shown in Table 18-1 and Table 18-2.

Table 18-1 Trigger sources for ordinary channels

OCTESEL	Trigger source	OCTESEL	Trigger source
00000	TMR1_CH1 event	10000	TMR20_TRGOUT event
00001	TMR1_CH2 event	10001	TMR20_TRGOUT2 event
00010	TMR1_CH3 event	10010	TMR20_CH1 event
00011	TMR2_CH2 event	10011	TMR20_CH2 event
00100	TMR2_CH3 event	10100	TMR20_CH3 event
00101	TMR2_CH4 event	10101	TMR8_TRGOUT2 event
00110	TMR2_TRGOUT event	10110	TMR1_TRGOUT2 event
00111	TMR3_CH1 event	10111	TMR4_TRGOUT event
01000	TMR3_TRGOUT event	11000	TMR6_TRGOUT event
01001	TMR4_CH4 event	11001	TMR3_CH4 event
01010	TMR5_CH1 event	11010	TMR4_CH1 event
01011	TMR5_CH2 event	11011	TMR1_TRGOUT event
01100	TMR5_CH3 event	11100	TMR2_CH1 event
01101	TMR8_CH1 event	11101	Reserved
01110	TMR8_TRGOUT event	11110	TMR7_TRGOUT event
01111	EXINT line11 External pin	11111	Reserved

Table 18-2 Trigger sources for preempted channels

PCTESEL	Trigger source	PCTESEL	Trigger source
00000	TMR1_CH4 event	10000	TMR20_TRGOUT event
00001	TMR1_TRGOUT event	10001	TMR20_TRGOUT2 event
00010	TMR2_CH1 event	10010	TMR20_CH4 event
00011	TMR2_TRGOUT event	10011	TMR1_TRGOUT2 event
00100	TMR3_CH2 event	10100	TMR8_TRGOUT event
00101	TMR3_CH4 event	10101	TMR8_TRGOUT2 event
00110	TMR4_CH1 event	10110	TMR3_CH3 event
00111	TMR4_CH2 event	10111	TMR3_TRGOUT event
01000	TMR4_CH3 event	11000	TMR3_CH1 event
01001	TMR4_TRGOUT event	11001	TMR6_TRGOUT event
01010	TMR5_CH4 event	11010	TMR4_CH4 event
01011	TMR5_TRGOUT event	11011	TMR1_CH3 event
01100	TMR8_CH2 event	11100	TMR20_CH2 event
01101	TMR8_CH3 event	11101	Reserved
01110	TMR8_CH4 event	11110	TMR7_TRGOUT event
01111	EXINT line15 External pin	11111	Reserved

18.4.2.3 Sampling and conversion sequence

The sampling period can be configured by setting the CSPTx bit in the ADC_SPT1 and ADC_SPT2 registers. Channels 0/1/10/11/12/13 are fast channels with a minimum of 2.5 sampling cycles, and others represent slow ones supporting a minimum of 6.5 sampling cycles.

The resolution can be programmed through the CRSEL bit in the ADC_CTRL1 register. A lower resolution has shorter conversion time. A single one conversion time is calculated with the following formula:

$$\text{A single one conversion time (ADCCLK period)} = \text{sampling time} + \text{resolution bits} + 0.5$$

Example:

If the CSPTx selects 2.5 period, and CRSEL select 12-bit resolution, then one conversion needs 2.5+12.5=15 ADCCLK periods

If the CSPTx selects 6.5 period, and CRSEL select 10-bit resolution, then one conversion needs 6.5+10.5=17 ADCCLK periods.

18.4.3 Conversion sequence management

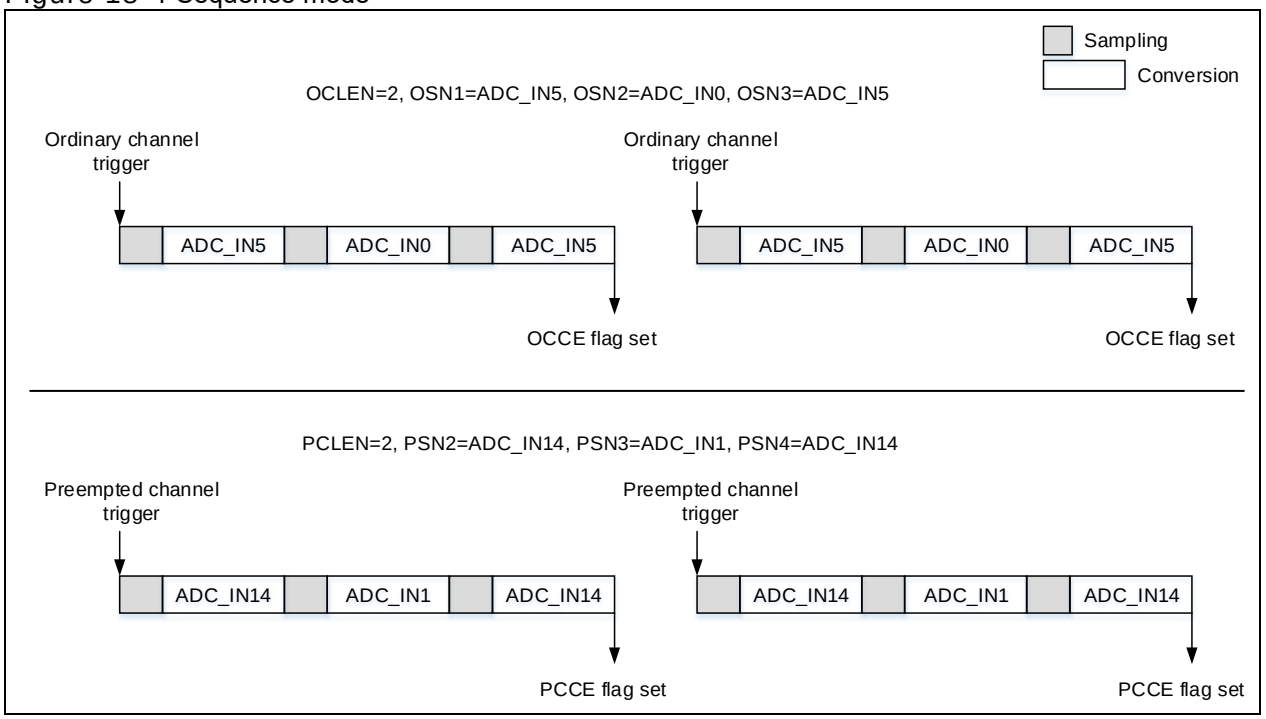
Only one channel is converted at each trigger event by default, that is, OSN1-defined channel or PSN4-defined channel.

The detailed conversion sequence modes are described in the following sections. With this, the channels can be converted in a specific order.

18.4.3.1 Sequence mode

The sequence mode is enabled by setting the SQEN bit in the ADC_CTRL1 register. The ADC_OSQx registers are used to configure the sequence and total number of the ordinary channels while the ADC_PSQ register is used to define the sequence and total number of the preempted channels. When the sequence mode is enabled, a single trigger event enables the conversion of a group of channels in order. The ordinary channels start converting from the QSN1 while the preempted channels starts from the PSNx, where x=4-PCLEN. Figure 18-4 shows an example of the behavior in sequence mode.

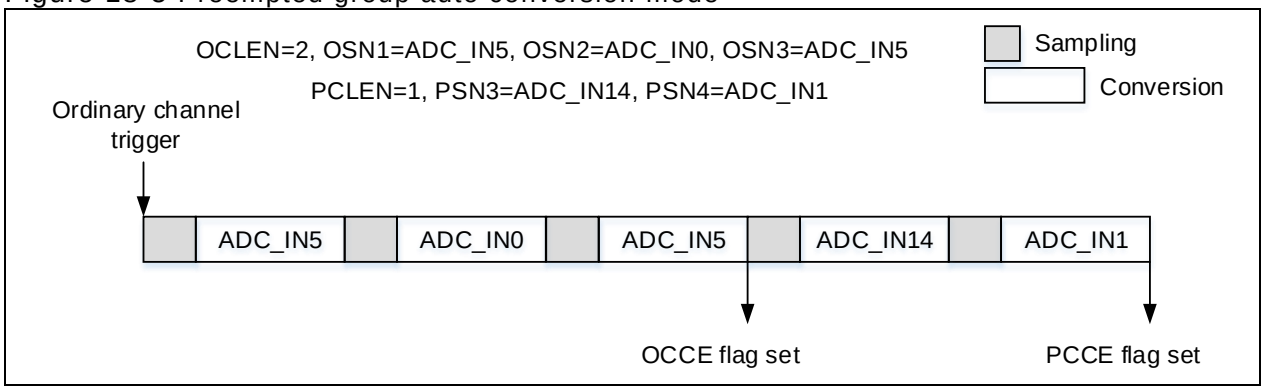
Figure 18-4 Sequence mode



18.4.3.2 Automatic preempted group conversion mode

The automatic preempted group conversion mode is enabled by setting the PCAUTOEN bit in the ADC_CTRL1 register. Once the ordinary channel conversion is over, the preempted group will automatically continue its conversion. This mode can work with the sequence mode. The preempted group conversion starts automatically at the end of the conversion of the ordinary group. Figure 18-5 shows an example of the behavior when the automatic preempted group conversion mode works with the ordinary group.

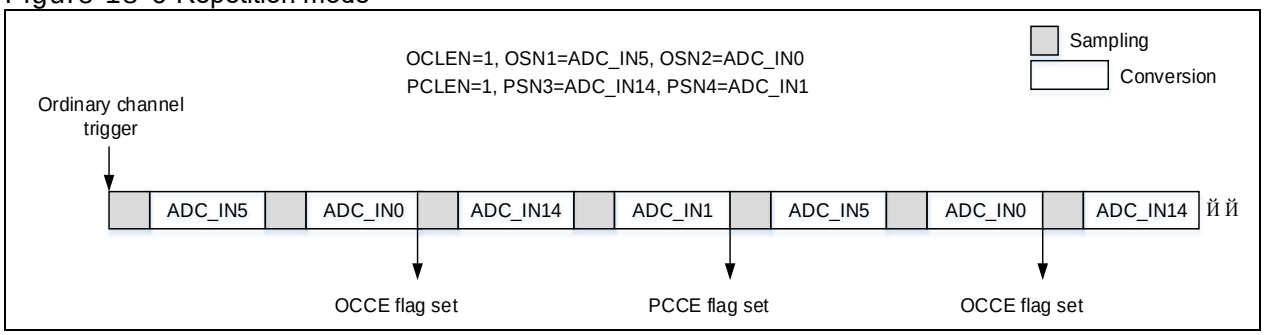
Figure 18-5 Preempted group auto conversion mode



18.4.3.3 Repetition mode

The repetition mode is enabled by setting the RPEN bit in the ADC_CTRL2 register. When a trigger signal is detected, the ordinary channels will be converted repeatedly. This mode can work with the ordinary channel conversion in the sequence mode to enable the repeated conversion of the ordinary group. Such mode can also work with the preempted group auto conversion mode to repeatedly convert the ordinary group and preempted group in sequence. Figure 18-6 shows an example of the behavior when the repetition mode works with the sequence mode and preempted group auto conversion mode.

Figure 18-6 Repetition mode



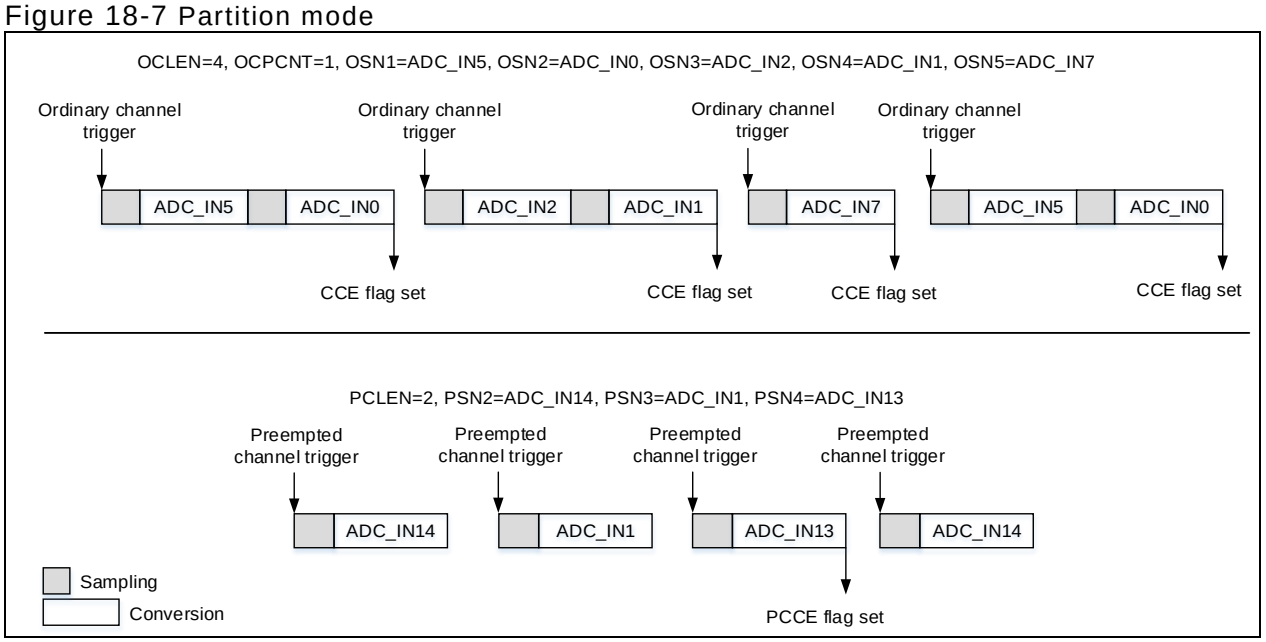
18.4.3.4 Partition mode

The partition mode of the ordinary group can be enabled by setting the OCPEN bit in the ADC_CTRL1 register. In this mode, the ordinary group conversion sequence length (OCLEN bit in the ADC_OSQ1 register) is divided into a smaller sub-group, in which the number of the channels is programmed with the OCPCNT bit in the ADC_CTRL1 register. A single trigger event will enable the conversion of all the channels in the sub-group. Each trigger event selects different sub-group in order.

Set the PCPEN bit in the ADC_CTRL1 register will enable the partition mode of the preempted group. In this mode, the preempted group conversion sequence length (PCLEN bit in the ADC_OSQ1 register) is divided into a sub-group with only one channel. A single one trigger event will enable the conversion of all the channels in the sub-group. Each trigger event selects different sub-group in order.

The partition mode cannot be used with the repetition mode at the same time. Figure 18-7 shows an example of the behavior in partition mode for ordinary group and preempted group.

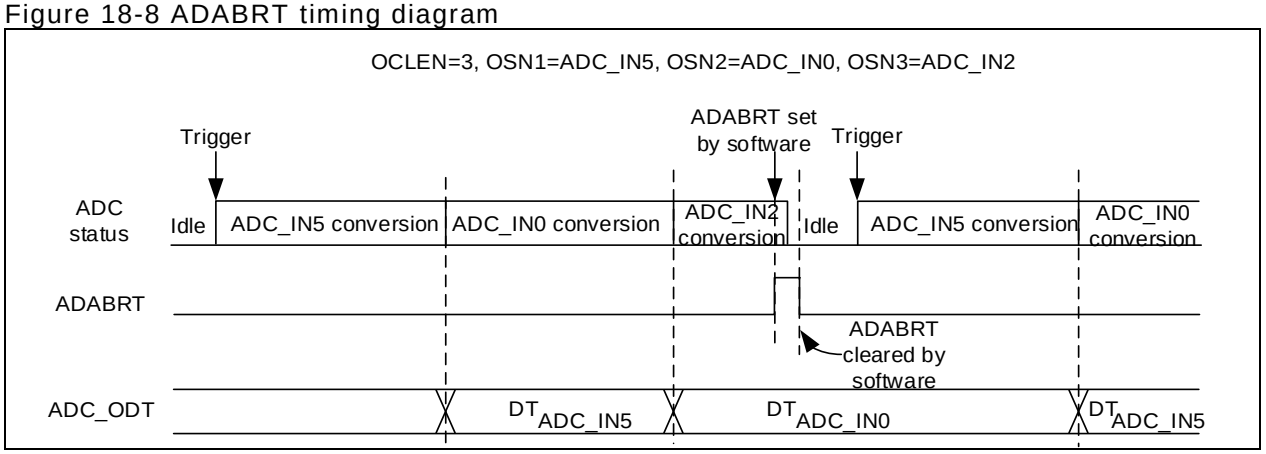
Figure 18-7 Partition mode



18.4.4 End of conversion

The ADABRT bit in the ADC_CTRL2 register can be used to stop ADC conversions. At the end of the conversions, the conversion sequence returns to the first channel. This allows the user to configure a new channel sequence, and the ADC starts conversions from the beginning based on the new order when a trigger event occurs. Figure 18-8 shows the timing diagram when the ADABRT bit is set.

Figure 18-8 ADABRT timing diagram



18.4.5 Oversampling

A single oversampling converted data can be done through multiple conversions of the same channel in which the cumulative converted data is averaged.

- Oversampling ratio is selected through the OSRSEL bit in the ADC_OVSP register. This bit is used to specify the oversampling multiple, which is performed by converting the same channel several times
- Oversampling shift is selected through the OSSSEL bit in the ADC_OVSP register, which is performed by right shift

If the averaged data is greater than 16 bits, then only pick up the right-aligned 16-bit data and put them into a 16-bit data register, shown in Table 18-3.

Example:

If 4x oversampling is selected through the OSRSEL bit, then the same channel is converted by four times in a single oversampling conversion, and the converted data derived from 4 conversions is put together. If 6-bit resolution is selected through the OSSSEL bit, then the cumulative data is divided by 2^6 and rounded up.

Table 18-3 Correlation between maximum cumulative data, oversampling multiple and shift digits

Oversampling multiple	**2x**	**4x**	**8x**	**16x**	**32x**	**64x**	**128x**	**256x**
Max cumulative data	0x1FFE	0x3FFC	0x7FF8	0xFFF0	0x1FFE0	0x3FFC0	0x7FF80	0xFFF00
No shift	0x1FFE	0x3FFC	0x7FF8	0xFFF0	0xFFE0	0xFFC0	0xFF80	0xFF00
Shift 1 digit	0x0FFF	0x1FFE	0x3FFC	0x7FF8	0xFFF0	0xFFE0	0xFFC0	0xFF80
Shift 2 digits	0x0800	0x0FFF	0x1FFE	0x3FFC	0x7FF8	0xFFF0	0xFFE0	0xFFC0
Shift 3 digits	0x0400	0x0800	0x0FFF	0x1FFE	0x3FFC	0x7FF8	0xFFF0	0xFFE0
Shift 4 digits	0x0200	0x0400	0x0800	0x0FFF	0x1FFE	0x3FFC	0x7FF8	0xFFF0
Shift 5 digits	0x0100	0x0200	0x0400	0x0800	0x0FFF	0x1FFE	0x3FFC	0x7FF8
Shift 6 digits	0x0080	0x0100	0x0200	0x0400	0x0800	0x0FFF	0x1FFE	0x3FFC
Shift 7 digits	0x0040	0x0080	0x0100	0x0200	0x0400	0x0800	0x0FFF	0x1FFE
Shift 8 digits	0x020	0x0040	0x0080	0x0100	0x0200	0x0400	0x0800	0x0FFF

When using oversampling conversion mode, the DTALIGN and PCDTOx are ignored, and data must be right aligned. The CRSEL bit is used to select the desired oversampling resolution only, without changing the data operation mode.

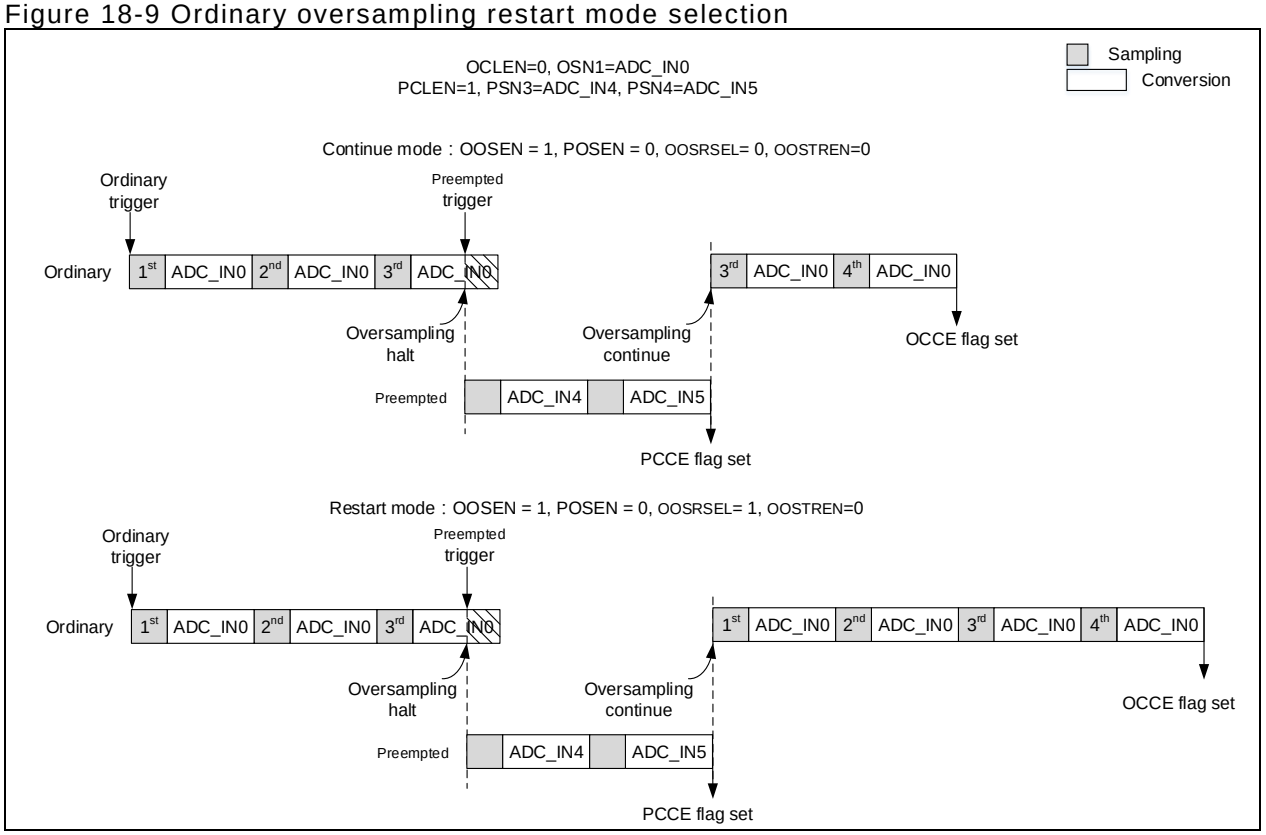
18.4.5.1 Oversampling of ordinary group of channels

The OOSRSEL bit in the ADC_OVSP register can be used to resume ordinary oversampling mode.

- OOSRSEL=0: continuous conversion mode. Ordinary group of channels, after being interrupted by preempted group of channels during oversampling, will retain the converted data and resume from the last interrupted ordinary conversion.
- OOSRSEL=1: restart mode. Ordinary group of channels, after being interrupted by preempted group of channels during oversampling, will be reset and restart the ordinary conversion.

Figure 18-9 shows the differences between ordinary continuous modes and restart mode in 4x oversampling rate and sequential mode.

Figure 18-9 Ordinary oversampling restart mode selection

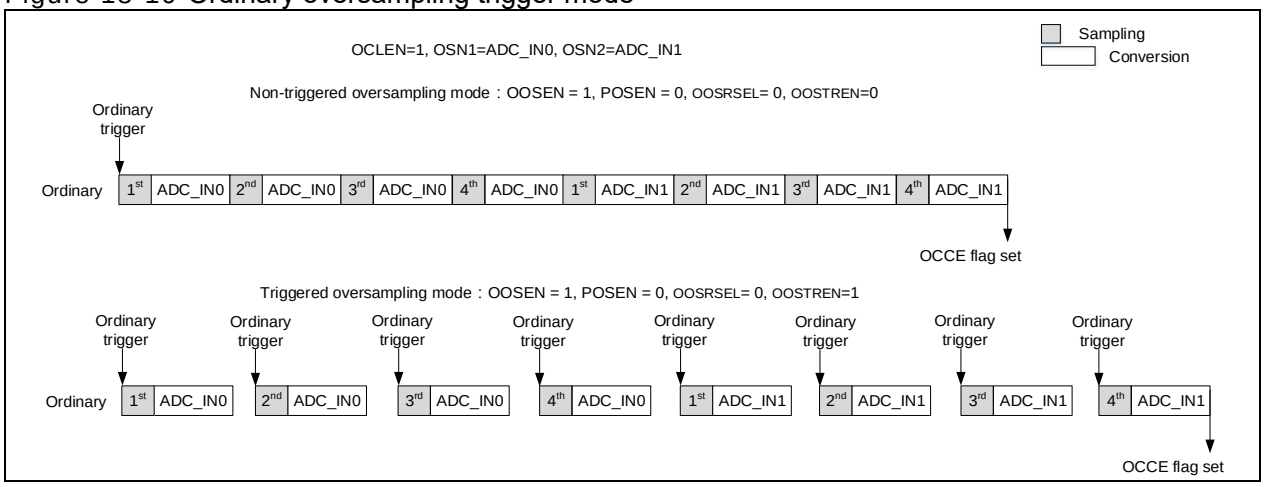


Trigger mode can be enabled by setting the OOSTREN bit in the ADC_OVSP register. The user must trigger each of the ordinary conversions. In this mode, once the ordinary conversion is interrupted by preempted group of channels, it is necessary to re-trigger ordinary group of channels before resuming the ordinary channels.

When the trigger mode works together with conversion sequence management mode, trigger mode is applied, and the conversion complete flag follows the conversion sequence management mode. Figure 18-10 shows the behavior when the ordinary trigger mode works together with resume mode in 4x oversampling rate and sequential mode.

Note: It is not possible to use both the trigger mode and repetition mode simultaneously.

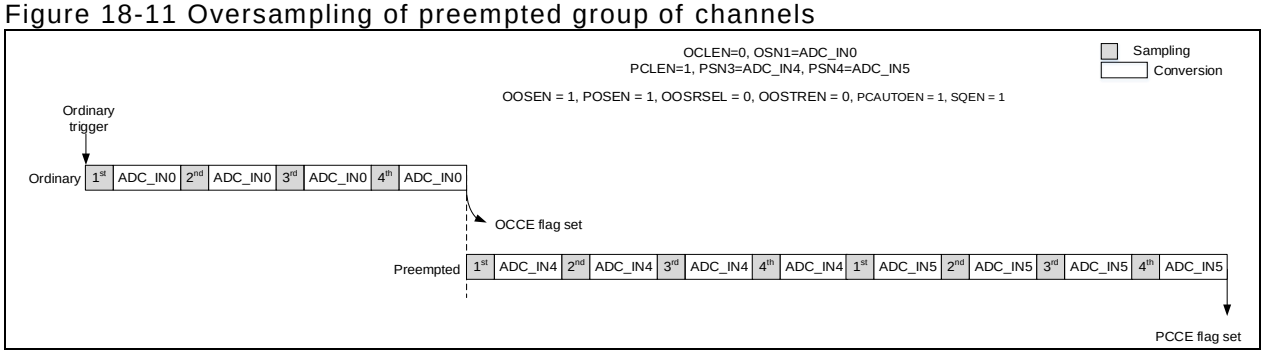
Figure 18-10 Ordinary oversampling trigger mode



18.4.5.2 Oversampling of preempted group of channels

It is possible to use both the preempted oversampling and ordinary oversampling simultaneously or individually. The oversampling of the preempted group of channels does not affect the ordinary oversampling modes. Figure 18-11 shows the behavior when the preempted oversampling and ordinary oversampling trigger mode are used simultaneously in 4x oversampling rate and auto-conversion of preempted group.

Figure 18-11 Oversampling of preempted group of channels



18.4.6 Data management

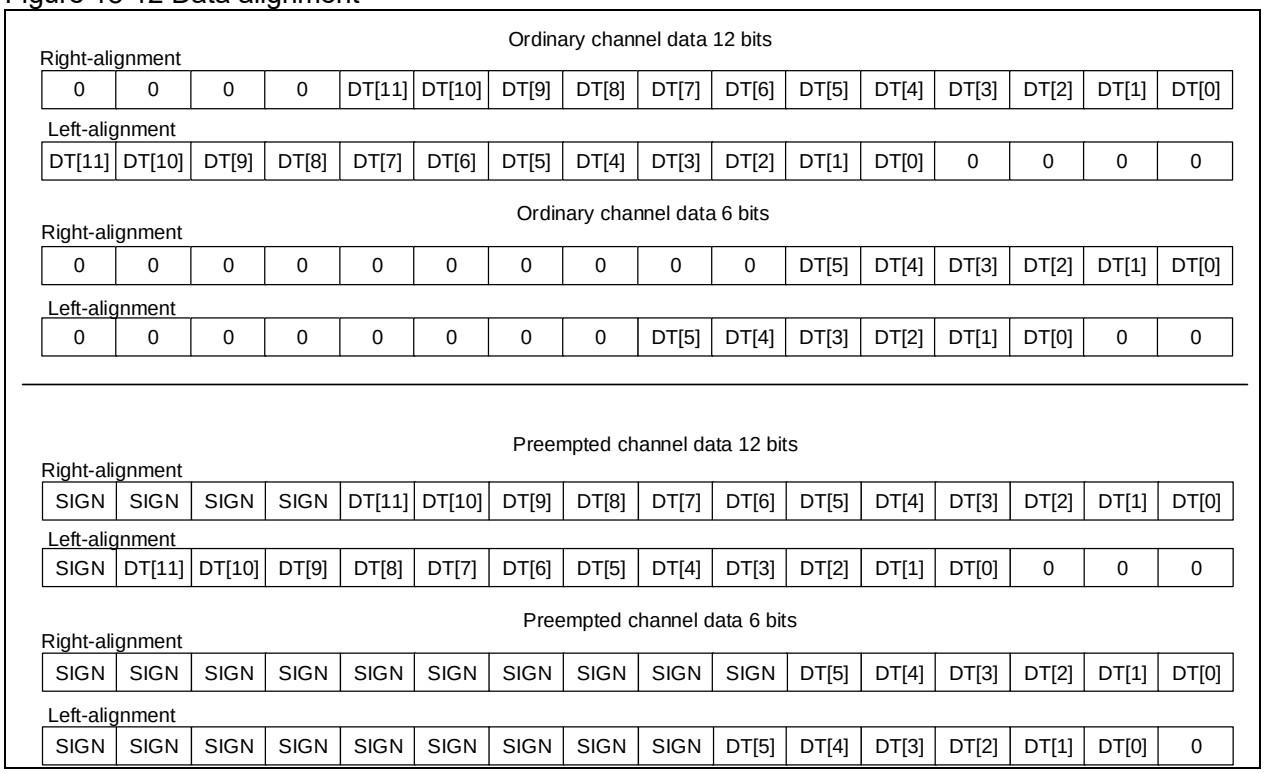
At the end of the conversion of the ordinary group, the converted value is stored in the ADC_ODT register. Once the preempted group conversion ends, the converted data of the preempted group is stored in the ADC_PDTx register.

18.4.6.1 Data alignment

DTALIGN bit in the ADC_CTRL2 register selects the alignment of data (right-aligned or left-aligned). Apart from this, the converted data of the preempted group is decreased by the offset written in the ADC_PCDTOx register. Thus the result may be a negative value, marked by SIGN.

The data are aligned on a half-word basis expect when the resolution is set to 6-bit. In this case, the data are aligned on a byte basis, as shown in Figure 18-12.

Figure 18-12 Data alignment



18.4.6.2 Data read

Read access to the ADC_ODT register using CPU or DMA gets the converted data of the ordinary group. Read access to the ADC_PDTx register using CPU gets the converted data of the preempted group.

The EOCSFEN bit in the ADC_CTRL2 register can be used to select when to set the ordinary group conversion complete flag, that is, at the end of sequence mode or each time the ordinary data register is updated.

When the OCDMAEN bit is set in the ADC_CTRL2 register, the ADC will issue DMA requests each time the ADC_OTD register is updated.

When the EOCSFEN or OCDMAEN is set, the ADC will automatically start overflow detection. If an overflow event occurs, the OCCO flag will be set, the ADC stops conversion, and the last valid data is stored in the data register. If the DMA is used, the DMA request remains set so that the DMA can read the last valid data. The OCCO flag is cleared by software, and the ADC is triggered again so that it starts conversion from the next channel of the valid data. In this case, even if an overflow event occurs halfway, all the data read are valid and in order.

The OCDRCEN bit in the ADC_CTRL2 register can be used to select whether to continually send DMA requests after the DMA transfer register is reset.

18.4.7 Voltage monitoring

The OCVMEN bit or PCVMEN bit in the ADC_CTRL1 register is used to enable voltage monitoring based on the converted data.

The VMOR bit will be set if the converted result is outside the high threshold (ADC_VMHB[11:0] register) or is less than the low threshold (ADC_VMLB[11:0]register).

In 10-bit resolution, ADC_VMHB[11:10] and ADC_VMLB[11:10] must be set to 2’b00;

In 8-bit resolution, ADC_VMHB[11:8] and ADC_VMLB[11:8] must be set to 4’b0000;

In 6-bit resolution, ADC_VMHB[11:6] and ADC_VMLB[11:6] must be set to 6’b000000;

The VMSGEN bit in the ADC_CTRL1 register is used to enable voltage monitor on either a single channel or all the channels. The VMCSEL bit is used to select a specific channel that requires voltage monitoring.

Voltage monitoring is based on the comparison result between the original converted data and the 12-bit voltage monitor boundary register, irrespective of the CRSEL, PCDTOx and DTALIGN bits.

When using an oversampler, voltage monitoring is based on the comparison result between the 16-bit registers (ADC_VMHB[15:0] and ADC_VMLB[15:0]) and the oversampled data.

18.4.7.1 Status flag and interrupts

Each of the ADCs has its dedicated ADCx_STS registers, that is, RDY flag, OCCO flag, OCCS (ordinary channel conversion start flag), PCCS (preempted channel conversion start flag), PCCE (preempted channel conversion end flag), OCCE (ordinary channel conversion end flag) and VMOR (voltage monitor out of range).

Three ADCx_STS registers are mapped onto the ADC_CSTS register, so it is possible to obtain the status of these tree registers by simply reading the ADC_CSTS register.

OCCO, PCCE, CCE and VMOR have their respective interrupt enable bits. Once the interrupt bits are enabled, the corresponding flag is set and an interrupt is sent to CPU. ADC1 shares an interrupt vector with ADC2 and ADC3.

18.5 Master/Slave mode

If Master/Slave mode is enabled, the master is triggered to work with the slave to do the channel conversion. The ADC_ODT register is used as a single interface obtaining the ordinary channel converted data of master/salve ADC.

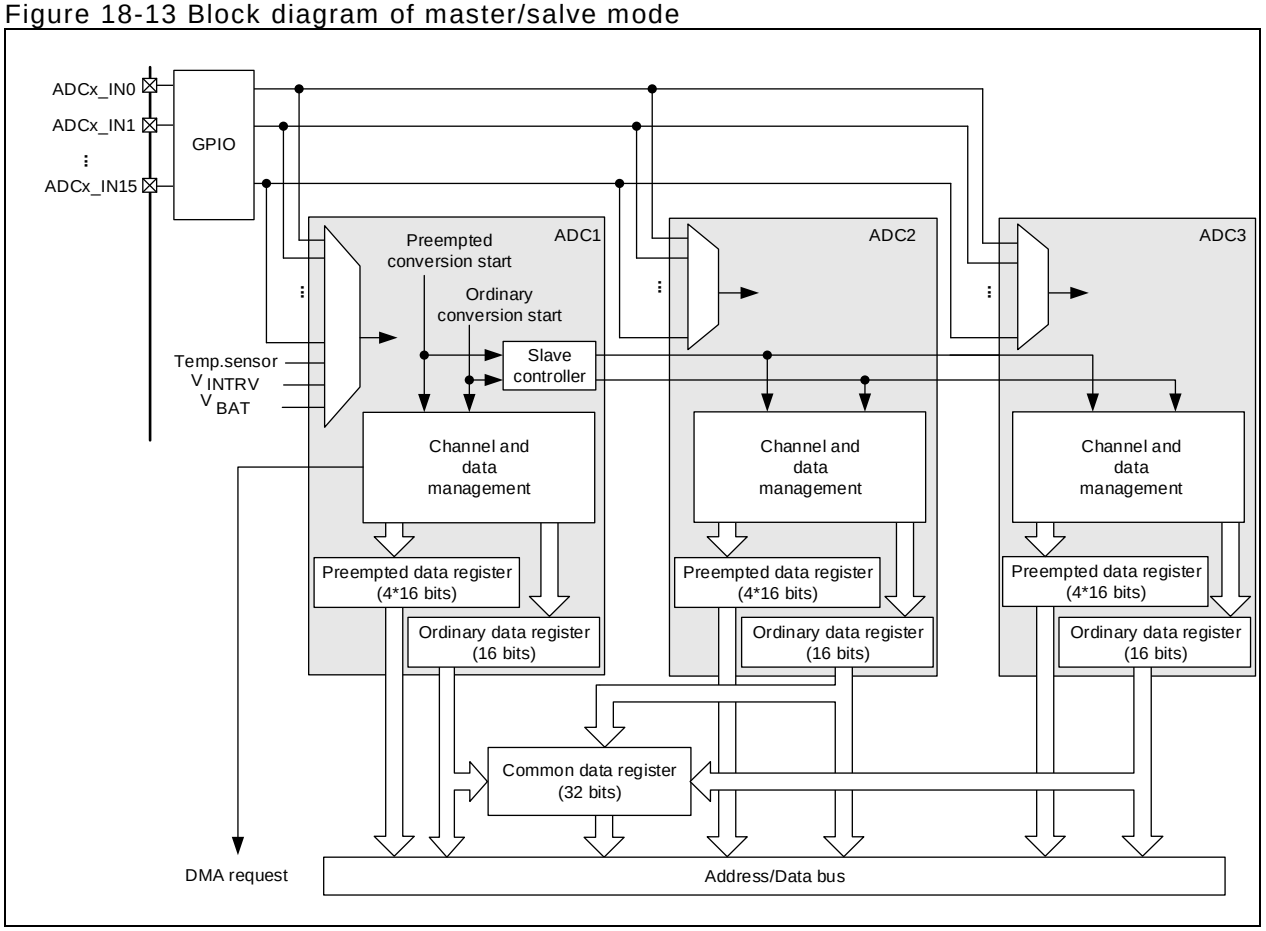
In a single master/slave mode, ADC1 acts as a master while ADC2 as a slave, and ADC3 behaves independently. In dual master/slave mode, ADC1 acts as a master, while both ADC2 and ADC3 act as slaves. In master/salve mode, it is necessary to enable both the trigger mode of master and slave ADC at the same time.

Note: ADC conversion abort (ADABRT) cannot be used in master/slave mode. In this mode, each of the ADCEN bit of the ADCs must be cleared in order to stop ADC conversions.

Note: Both the master and slave must have the same resolution in order to avoid losing synchronization between master and slave.

Note: To enable several ADC conversions with low resolution simultaneously, it is recommended to use the master/slave mode together with DMA1 or DMA2.

Figure 18-13 Block diagram of master/salve mode



18.5.1 Data management

In Master/Slave mode, the converted data of ordinary channels is also stored in the ADC_CODT register. The MSDMASEL bit in the ADC_CCTRL register can be used to select from five DMA transfer modes, as shown in Table 18-4. As long as the MSDMASEL is set, the ADC1 DMA channel is used to generate a DMA request each time the data is ready, and overflow detection on master/salve is also enabled. Once an overflow event occurs, the ADC will stop conversion, the DMA request is halted, and loss of synchronization may happen between the master and slave, so it is recommended to re-initialize ADC before re-triggering on an overflow event.

Table 18-4 Master/slave DMA mode

MSDMASEL	Master/slave mode	DMA requests	ADC_CODT[31:0]
001	Single slave	1st	16 bit 0, ADC1_ODT[15:0]
		2nd	16 bit 0, ADC2_ODT [15:0]
		3rd	16 bit 0, ADC1_ODT [15:0]
	Dual slave	1st	16 bit 0, ADC1_ODT[15:0]
		2nd	16 bit 0, ADC2_ODT [15:0]
		3rd	16 bit 0, ADC3_ODT[15:0]
		4th	16 bit 0, ADC1_ODT [15:0]
010	Single slave	1st	ADC2_ODT[15:0], ADC1_ODT[15:0]
		2nd	ADC2_ODT[15:0], ADC1_ODT[15:0]
	Dual slave	1st	ADC2_ODT[15:0], ADC1_ODT[15:0]
		2nd	ADC1_ODT[15:0], ADC3_ODT[15:0]
		3rd	ADC3_ODT[15:0], ADC2_ODT[15:0]
		4th	ADC2_ODT[15:0], ADC1_ODT[15:0]
011	Single slave	1st	16 bit 0, ADC2_ODT[7:0], ADC1_ODT[7:0]
		2nd	16 bit 0, ADC2_ODT[7:0], ADC1_ODT[7:0]
	Dual slave	1st	16 bit 0, ADC2_ODT[7:0], ADC1_ODT[7:0]
		2nd	16 bit 0, ADC1_ODT[7:0], ADC3_ODT[7:0]
		3rd	16 bit 0, ADC3_ODT[7:0], ADC2_ODT[7:0]
		4th	16 bit 0, ADC2_ODT[7:0], ADC1_ODT[7:0]
100	Dual slave	1st	8 bit 0, ADC3_ODT[7:0], ADC2_ODT[7:0], ADC1_ODT[7:0]
		2nd	8 bit 0, ADC3_ODT[7:0], ADC2_ODT[7:0], ADC1_ODT[7:0]
101	Dual slave	1st	ADC2_ODT[15:0], ADC1_ODT[15:0]
		2nd	16 bit 0, ADC3_ODT[15:0]
		3rd	ADC2_ODT[15:0], ADC1_ODT[15:0]

MSDRCEN bit in the ADC_CCTRL register can be used to select when to stop DMA request, that is, when the DMA request remains set until the end of data transfer, or when the DMA transfer register is reset.

18.5.2 Simultaneous mode

Regular simultaneous mode

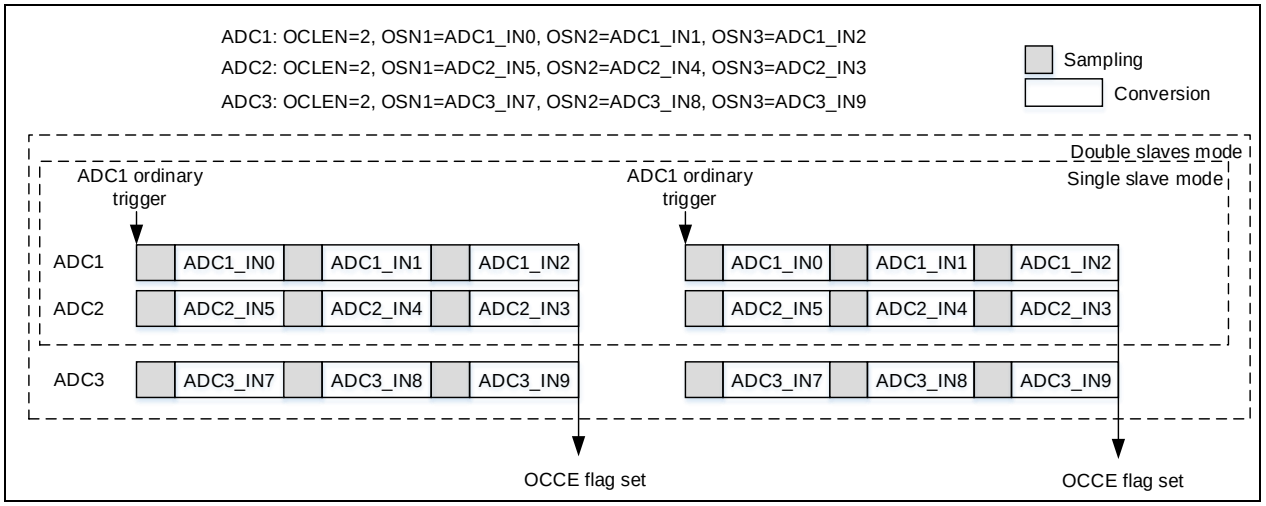
MSSEL bit in the ADC_CCTRL register is used to select regular simultaneous mode. If this mode is enabled, the regular channels of the master are triggered so that both the master and the slave convert the regular channels simultaneously. In this mode, it is required to configure the same sampling time and the same sequence length for the master and slave to avoid the loss of data due to the lack of synchronization.

Figure 18-14 shows an example of the regular simultaneous mode

The single slave mode can work with the mode 1/2/3 of the transfer mode (MSDMASEL), while the dual slave mode can work with the mode 1/4/5.

Note: The same channel is not allowed to be sampled by several ADCs simultaneously. Do not put the same channel in the same sequence location of different ADCs.

Figure 18-14 Regular simultaneous mode

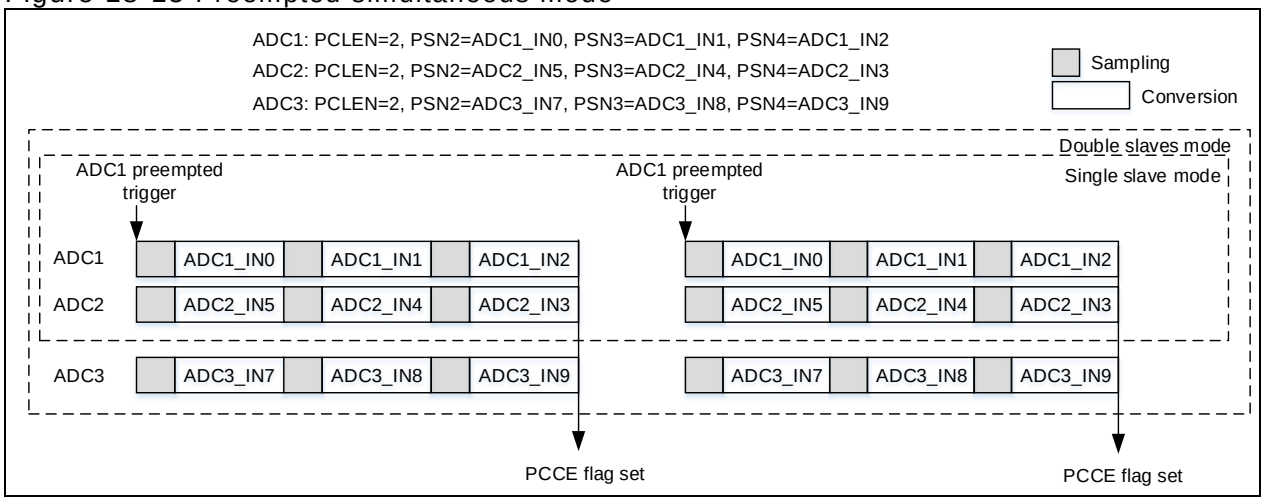


Preempted simultaneous mode

MSSEL bit in the ADC_CCTRL register is used to select preempted simultaneous mode. If this mode is enabled, the preempted channels of the master is triggered so that both the master and the slave convert the preempted channels simultaneously. Figure 18-15 shows an example of the preempted simultaneous mode

Note: The same channel is not allowed to be sampled by several ADCs simultaneously. Do not put the same channel in the same sequence location of different ADCs.

Figure 18-15 Preempted simultaneous mode



Combined regular/preempted simultaneous mode

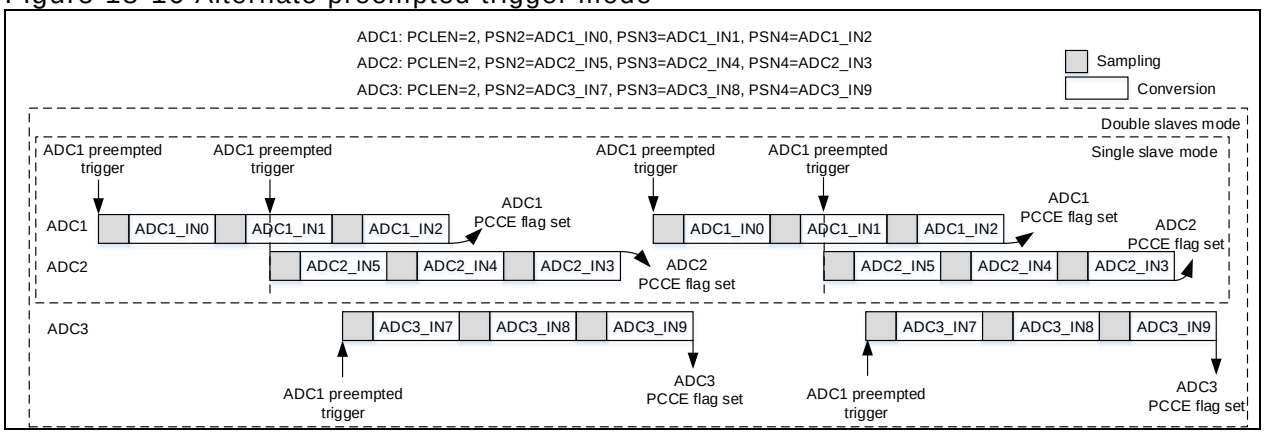
MSSEL bit in the ADC_CCTRL register is used to select combined regular/preempted simultaneous mode. If this mode is enabled, the regular channels of the master is triggered so that the master works with the slave to convert the regular channels simultaneously, or the preempted channels of the master is triggered to enable the master and slave to convert the preempted channels simultaneously.

18.5.3 Alternate preempted trigger mode

Alternate preempted trigger mode

MSSEL bit in the ADC_CCTRL register selects the alternate preempted trigger mode. If this mode is enabled, the preempted channels of the master are triggered continuously so that the master/slave ADCs convert the preempted channels alternately. Figure 18-16 shows an example of the alternate preempted trigger mode.

Figure 18-16 Alternate preempted trigger mode



Combined regular simultaneous + alternate preempted trigger mode

MSSEL bit in the ADC_CCTRL register is used to select combined regular simultaneous + alternate preempted trigger mode. In this mode, trigger the regular group of the master to start regular simultaneous conversion of master/slave, or trigger the preempted group of the master continuously to allow the master/slave ADCs to convert the preempted group alternately.

If the regular conversion is interrupted by the preempted trigger, the regular conversion of all ADCs is stopped, and one of the ADCs starts the preempted conversion. At this point, the master will ignore the preempted trigger until the regular conversion is resumed.

18.5.4 Regular shift mode

MSSEL bit in the ADC_CCTRL register is used to select regular shift mode on regular group. After a master trigger occurs, the conversion interval between ADCs is based on the programmed shift length (through the ASISEL bit in the ADC_CCTRL register), as shown in Figure 18-17.

In this mode, the sampling interval between ADCs is configured to be at least 2.5 ADCCLK cycles by hardware. Thus the ASISEL bit becomes invalid when it cannot meet such sampling interval. Because of this feature, it is possible to put the same channel in the same ADC location without causing the overlapped sampling time.

The single slave mode can work with the transfer mode 1/2/3 (MSDMASEL), while the dual slave mode can work with all transfer modes. Figure 18-18 shows how to store data when the dual slave regular shift mode works with DMA mode 2.

Note: The preempted trigger or slave regular trigger is not allowed in this mode.

Figure 18-17 Regular shift mode

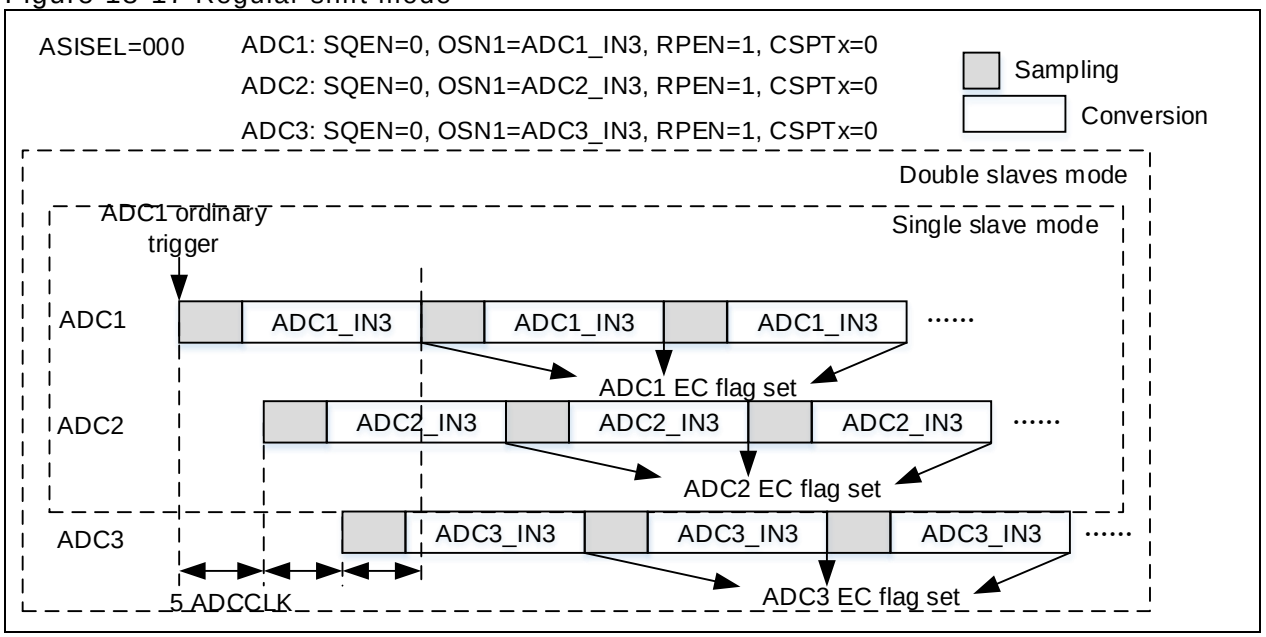
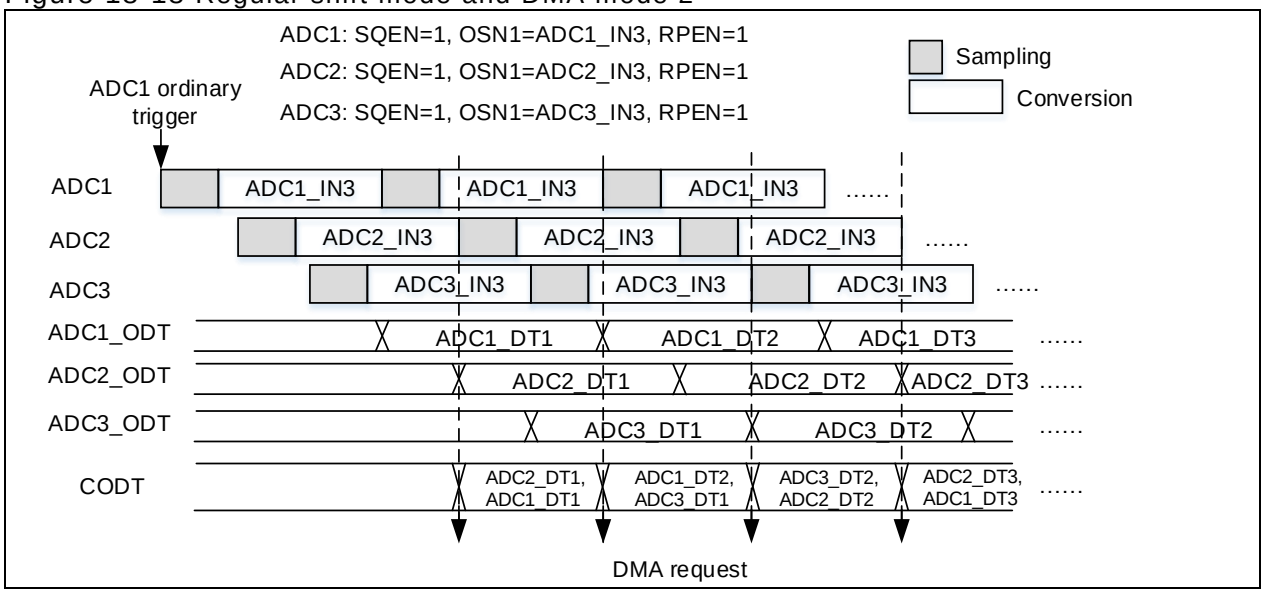


Figure 18-18 Regular shift mode and DMA mode 2



18.6 ADC registers

Table 18-5 lists ADC register map and their reset values.

These peripheral registers must be accessed by word (32 bits).

Table 18-5 ADC register map and reset values

Register name	Offset	Reset value
ADC1_STS	0x000	0x0000 0000
ADC1_CTRL1	0x004	0x0000 0000
ADC1_CTRL2	0x008	0x0000 0000
ADC1_SPT1	0x00C	0x0000 0000
ADC1_SPT2	0x010	0x0000 0000
ADC1_PCDTO1	0x014	0x0000 0000
ADC1_PCDTO2	0x018	0x0000 0000
ADC1_PCDTO3	0x01C	0x0000 0000
ADC1_PCDTO4	0x020	0x0000 0000
ADC1_VMHB	0x024	0x0000 FFFF
ADC1_VMLB	0x028	0x0000 0000
ADC1_OSQ1	0x02C	0x0000 0000
ADC1_OSQ2	0x030	0x0000 0000
ADC1_OSQ3	0x034	0x0000 0000
ADC1_PSQ	0x038	0x0000 0000
ADC1_PDT1	0x03C	0x0000 0000
ADC1_PDT2	0x040	0x0000 0000
ADC1_PDT3	0x044	0x0000 0000
ADC1_PDT4	0x048	0x0000 0000
ADC1_ODT	0x04C	0x0000 0000
ADC1_OVSP	0x080	0x0000 0000
ADC1_CALVAL	0x0B4	0x0000 0000

ADC2_STS	0x100	0x0000 0000
ADC2_CTRL1	0x104	0x0000 0000
ADC2_CTRL2	0x108	0x0000 0000
ADC2_SPT1	0x10C	0x0000 0000
ADC2_SPT2	0x110	0x0000 0000
ADC2_PCDTO1	0x114	0x0000 0000
ADC2_PCDTO2	0x118	0x0000 0000
ADC2_PCDTO3	0x11C	0x0000 0000
ADC2_PCDTO4	0x120	0x0000 0000
ADC2_VMHB	0x124	0x0000 FFFF
ADC2_VMLB	0x128	0x0000 0000
ADC2_OSQ1	0x12C	0x0000 0000
ADC2_OSQ2	0x130	0x0000 0000
ADC2_OSQ3	0x134	0x0000 0000
ADC2_PSQ	0x138	0x0000 0000
ADC2_PDT1	0x13C	0x0000 0000
ADC2_PDT2	0x140	0x0000 0000
ADC2_PDT3	0x144	0x0000 0000
ADC2_PDT4	0x148	0x0000 0000
ADC2_ODT	0x14C	0x0000 0000
ADC2_OVSP	0x180	0x0000 0000
ADC2_CALVAL	0x1B4	0x0000 0000
ADC3_STS	0x200	0x0000 0000
ADC3_CTRL1	0x204	0x0000 0000
ADC3_CTRL2	0x208	0x0000 0000
ADC3_SPT1	0x20C	0x0000 0000
ADC3_SPT2	0x210	0x0000 0000
ADC3_PCDTO1	0x214	0x0000 0000
ADC3_PCDTO2	0x218	0x0000 0000
ADC3_PCDTO3	0x21C	0x0000 0000
ADC3_PCDTO4	0x220	0x0000 0000
ADC3_VMHB	0x224	0x0000 FFFF
ADC3_VMLB	0x228	0x0000 0000
ADC3_OSQ1	0x22C	0x0000 0000
ADC3_OSQ2	0x230	0x0000 0000
ADC3_OSQ3	0x234	0x0000 0000
ADC3_PSQ	0x238	0x0000 0000
ADC3_PDT1	0x23C	0x0000 0000
ADC3_PDT2	0x240	0x0000 0000
ADC3_PDT3	0x244	0x0000 0000
ADC3_PDT4	0x248	0x0000 0000

ADC3_ODT	0x24C	0x0000 0000
ADC3_OVSP	0x280	0x0000 0000
ADC3_CALVAL	0x2B4	0x0000 0000
ADC_CSTS	0x300	0x0000 0000
ADC_CCTRL	0x304	0x0000 0000
ADC_CODT	0x308	0x0000 0000

18.6.1 ADC status register (ADC_STS)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 7	Reserved	0x0000000	resd	Kept at its default value.
Bit 6	RDY	0x0	rw0c	ADC conversion ready flag This bit is read only. It is set by hardware when the ADC is powered. 0: Not ready for ADC conversion 1: Ready for ADC conversion
Bit 5	OCCO	0x0	rw0c	Ordinary channel conversion overflow flag) This bit is set by hardware and cleared by software (writing 0). 0: No overflow occurred 1: Overflow occurred Overflow detection is applicable to the case of DMA transfer enable or EOCSFEN =1
Bit 4	OCCS	0x0	rw0c	Ordinary channel conversion start flag This bit is set by hardware and cleared by software (writing 0). 0: No ordinary channel conversion started 1: Ordinary channel conversion has started
Bit 3	PCCS	0x0	rw0c	Preempted channel conversion start flag This bit is set by hardware and cleared by software (writing 0). 0: No preempted channel conversion started 1: Preempted channel conversion has started
Bit 2	PCCE	0x0	rw0c	Preempted channel end of conversion flag This bit is set by hardware and cleared by software (writing 0). 0: Conversion is not complete 1: Conversion is complete
Bit 1	OCCE	0x0	rw0c	End of conversion flag This bit is set by hardware. It is cleared by software (writing 0) or by reading the ADC_ODT register. 0: Conversion is not complete 1: Conversion is complete
Bit 0	VMOR	0x0	rw0c	Voltage monitoring out of range flag This bit is set by hardware and cleared by software (writing 0). 0: Voltage is within the value programmed 1: Voltage is outside the value programmed

18.6.2 ADC control register1 (ADC_CTRL1)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 27	Reserved	0x00	resd	Kept at its default value.
Bit 26	OCCOIE	0x0	rw	Ordinary channel conversion overflow interrupt enable 0: Ordinary channel conversion overflow interrupt disabled 1: Ordinary channel conversion overflow interrupt enabled
Bit 25: 24	CRSEL	0x0	rw	Conversion resolution select 00: 12-bit 01: 10-bit 10: 8-bit 11: 6-bit
Bit 23	OCVMEN	0x0	rw	Voltage monitoring enable on ordinary channels 0: Voltage monitoring disabled on ordinary channels 1: Voltage monitoring enabled on ordinary channels
Bit 22	PCVMEN	0x0	rw	Voltage monitoring enable on preempted channels 0: Voltage monitoring disabled on preempted channels 1: Voltage monitoring enabled on preempted channels
Bit 21: 16	Reserved	0x0	resd	Kept at its default value.
Bit 15: 13	OCPCNT	0x0	rw	Partitioned mode conversion count of ordinary channels 000: 1 channel 001: 2 channels …… 111: 8 channels Note: In this mode, the preempted group converts only one channel at each trigger.
Bit 12	PCPEN	0x0	rw	Partitioned mode enable on preempted channels 0: Partitioned mode disabled on preempted channels 1: Partitioned mode enabled on preempted channels
Bit 11	OCPEN	0x0	rw	Partitioned mode enable on ordinary channels This is set and cleared by software to enable or disable partitioned mode on ordinary channels. 0: Partitioned mode disabled on ordinary channels 1: Partitioned mode enabled on ordinary channels
Bit 10	PCAUTOEN	0x0	rw	Preempted group automatic conversion enable after ordinary group 0: Preempted group automatic conversion disabled 1: Preempted group automatic conversion enabled
Bit 9	VMSGEN	0x0	rw	Voltage monitoring enable on a single channel 0: Disabled (Voltage monitoring enabled on all channels) 1: Enabled (Voltage monitoring enabled a single channel)
Bit 8	SQEN	0x0	rw	Sequence mode enable 0: Sequence mode disabled (a single channel is converted) 1: Sequence mode enabled (the selected multiple channels are converted) Note: If this mode is enabled and the CCEIEN/PCCEIEN is set, an OCCE or PCCE interrupt is generated only at the end of conversion of the last channel.

Bit 7	PCCEIEN	0x0	rw	Conversion end interrupt enable on Preempted channels 0: Conversion end interrupt disabled on Preempted channels 1: Conversion end interrupt enabled on Preempted channels
Bit 6	VMORIEN	0x0	rw	Voltage monitoring out of range interrupt enable 0: Voltage monitoring out of range interrupt disabled 1: Voltage monitoring out of range interrupt enabled
Bit 5	CCEIEN	0x0	rw	Channel conversion end interrupt enable 0: Channel conversion end interrupt disabled 1: Channel conversion end interrupt enabled
Bit 4: 0	VMCSEL	0x00	rw	Voltage monitoring channel select This filed is valid only when the VMSGEN is enabled. 00000: ADC_IN0 channel 00001: ADC_IN1 channel …… 01111: ADC_IN15 channel 10000: ADC_IN16 channel 10001: ADC_IN17 channel 10010~11111: Unused, configuration is not allowed.

18.6.3 ADC control register2 (ADC_CTRL2)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 30: 26	Reserved	0x00	resd	Kept at its default value
Bit 30	OCSWTRG	0x0	rw	Conversion of ordinary channels triggered by software 0: Conversion of ordinary channels not triggered 1: Conversion of ordinary channels triggered (This bit is cleared by software or by hardware as soon as the conversion starts)
Bit 29: 28	OCETE	0x0	rw	Ordinary channel external trigger edge select 00: Edge trigger forbidden 01: Rising edge 01: Falling edge 11: Any edge
Bit 31 Bit 27: 24	OCTESEL	0x0	rw	Ordinary channel conversion trigger event select Note: Refer to section 18.4.1.1 for details on bits.
Bit 22	PCSWTRG	0x0	rw	Conversion of preempted channels triggered by software 0: Conversion of preempted channels not triggered 1: Conversion of preempted channels triggered (This bit is cleared by software or by hardware as soon as the conversion starts)
Bit 21: 20	PCETE	0x0	rw	Preempted channel external trigger edge select 00: Edge trigger forbidden 01: Rising edge 01: Falling edge 11: Any edge
Bit 23 Bit 19: 16	PCTESEL	0x0	rw	Preempted channel conversion trigger event select Note: Refer to section 18.4.1.1 for details on bits.
Bit 15: 12	Reserved	0x0	resd	Kept at its default value.

Bit 11	DTALIGN	0x0	rw	Data alignment 0: Right alignment 1: Left alignment
Bit 10	EOCSFEN	0x0	rw	Each ordinary channel conversion OCCE flag enable) 0: Disabled 1: Enabled Note: Overflow detection is enabled automatically when this bit is set.
Bit 9	OCDRCEN	0x0	rw	Ordinary channel DMA request continue enable for independent mode) 0: Disabled (After the completion of the programmed number of DMA transfers, no DMA request generated at the end of ordinary conversion) 1: Enabled (Don't care about the programmed number of DMA transfers, Each ordinary channel sends DMA request at the end of ordinary conversion) Note: This bit is set only in non-master/slave mode with OCDMAEN = 1.
Bit 8	OCDMAEN	0x0	rw	DMA transfer enable of ordinary channels 0: Disabled 1: Enabled
Bit 7: 5	Reserved	0x0	resd	Kept at its default value.
Bit 3	ADCALINIT	0x0	rw	Initialize A/D calibration This bit is set by software and cleared by hardware. It is cleared after the calibration registers are initialized. 0: No initialization occurred or initialization completed 1: Enable initialization or initializations is ongoing
Bit 2	ADCAL	0x0	rw	A/D Calibration 0: No calibration occurred or calibration completed 1: Enable calibration or calibration is in process
Bit 1	RPEN	0x0	rw	Repetition mode enable 0: Repetition mode disabled When SQEN=0, a single conversion is done each time when a trigger event arrives; when SQEN=1, a group of repetition is done each timer when a trigger event arrives. 1: Repetition mode enabled When SQEN =0, continuous conversion mode on a single channel is enabled at each trigger event; when SQEN =1, continuous conversion mode on a group of channels is enabled at each trigger event.
Bit 0	ADCEN	0x0	rw	A/D converter enable 0: A/D converter disabled (ADC goes to power-down mode) 1: A/D converter enabled Note: When this bit is in OFF state, write an ON command can wake up The ADC from power-down mode. When this bit in ON state, write another ON command can start a regular group conversion. The application should pay attention to the fact that there is a delay of t_{STAB} between power on and start of conversion.

18.6.4 ADC sampling time register 1 (ADC_SPT1)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 27	Reserved	0x00	resd	Kept at its default value.
Bit 26: 24	CSPT18	0x0	rw	Sample time selection of channel ADC_IN18 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 23: 21	CSPT17	0x0	rw	Sample time selection of channel ADC_IN17 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles 111: 239.5 cycles
Bit 20: 18	CSPT16	0x0	rw	Sample time selection of channel ADC_IN16 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 17: 15	CSPT15	0x0	rw	Sample time selection of channel ADC_IN15 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 14: 12	CSPT14	0x0	rw	Sample time selection of channel ADC_IN14 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles

Bit 11: 9	CSPT13	0x0	rw	Sample time selection of channel ADC_IN13 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 8: 6	CSPT12	0x0	rw	Sample time selection of channel ADC_IN12 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 5: 3	CSPT11	0x0	rw	Sample time selection of channel ADC_IN11 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 2: 0	CSPT10	0x0	rw	Sample time selection of channel ADC_IN10 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles

18.6.5 ADC sampling time register 2 (ADC_SPT2)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value
Bit 29: 27	CSPT9	0x0	rw	Sample time selection of channel ADC_IN9 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles

Bit 26: 24	CSPT8	0x0	rw	Sample time selection of channel ADC_IN8 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 23: 21	CSPT7	0x0	rw	Sample time selection of channel ADC_IN7 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 20: 18	CSPT6	0x0	rw	Sample time selection of channel ADC_IN6 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 17: 15	CSPT5	0x0	rw	Sample time selection of channel ADC_IN5 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 14: 12	CSPT4	0x0	rw	Sample time selection of channel ADC_IN4 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles

Bit 11: 9	CSPT3	0x0	rw	Sample time selection of channel ADC_IN3 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 8: 6	CSPT2	0x0	rw	Sample time selection of channel ADC_IN2 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 5: 3	CSPT1	0x0	rw	Sample time selection of channel ADC_IN1 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 2: 0	CSPT0	0x0	rw	Sample time selection of channel ADC_IN0 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles

18.6.6 ADC preempted channel data offset register x (ADC_PCDTOx) (x=1..4)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value
Bit 11: 0	PCDTOx	0x000	rw	Data offset for Preempted channel x Converted data stored in the ADC_PDTx = Raw converted data – ADC_PCDTOx

18.6.7 ADC voltage monitor high threshold register (ADC_VWHB)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x00000	resd	Kept at its default value
Bit 15: 0	VMHB	0xFFF	rw	Voltage monitoring high boundary

18.6.8 ADC voltage monitor low threshold register (ADC_ VWLB)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x00000	resd	Kept at its default value
Bit 15: 0	VMLB	0xFFF	rw	Voltage monitoring low boundary

18.6.9 ADC ordinary sequence register 1 (ADC_ OSQ1)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value
Bit 23: 20	OCLEN	0x0	rw	Ordinary conversion sequence length 0000: 1 conversion 0001: 2 conversions …… 1111: 16 conversions
Bit 19: 15	OSN16	0x00	rw	Number of 16th conversion in ordinary sequence
Bit 14: 10	OSN15	0x00	rw	Number of 15th conversion in ordinary sequence
Bit 9: 5	OSN14	0x00	rw	Number of 14th conversion in ordinary sequence
Bit 4: 0	OSN13	0x00	rw	Number of 13th conversion in ordinary sequence Note: The number can be from 0 to 17. For example, if the number is set to 3, it means that the 13th conversion is ADC_IN3 channel.

18.6.10 ADC ordinary sequence register 2 (ADC_ OSQ2)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value
Bit 29: 25	OSN12	0x00	rw	Number of 12th conversion in ordinary sequence
Bit 24: 20	OSN11	0x00	rw	Number of 11th conversion in ordinary sequence
Bit 19: 15	OSN10	0x00	rw	Number of 10th conversion in ordinary sequence
Bit 14: 10	OSN9	0x00	rw	Number of 9th conversion in ordinary sequence
Bit 9: 5	OSN8	0x00	rw	Number of 8th conversion in ordinary sequence
Bit 4: 0	OSN7	0x00	rw	Number of 7th conversion in ordinary sequence Note: The number can be from 0 to 17. For example, if the number is set to 8, it means that the 7th conversion is ADC_IN8 channel.

18.6.11 ADC ordinary sequence register 3 (ADC_ OSQ3)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value
Bit 29: 25	OSN6	0x00	rw	Number of 6th conversion in ordinary sequence
Bit 24: 20	OSN5	0x00	rw	Number of 5th conversion in ordinary sequence
Bit 19: 15	OSN4	0x00	rw	Number of 4th conversion in ordinary sequence
Bit 14: 10	OSN3	0x00	rw	Number of 3rd conversion in ordinary sequence
Bit 9: 5	OSN2	0x00	rw	Number of 2nd conversion in ordinary sequence

Bit 4: 0	OSN1	0x00	rw	Number of 1st conversion in ordinary sequence Note: The number can be from 0 to 17. For example, if the number is set to 8, it means that the 1st conversion is ADC_IN17 channel.

18.6.12 ADC preempted sequence register (ADC_ PSQ)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value
Bit 21: 20	PCLEN	0x0	rw	Preempted conversion sequence length 00: 1 conversion 01: 2 conversions 10: 3 conversions 11: 4 conversions
Bit 19: 15	PSN4	0x00	rw	Number of 4th conversion in preempted sequence
Bit 14: 10	PSN3	0x00	rw	Number of 3rd conversion in preempted sequence
Bit 9: 5	PSN2	0x00	rw	Number of 2nd conversion in preempted sequence
Bit 4: 0	PSN1	0x00	rw	Number of 1st conversion in preempted sequence Note: The number can be from 0 to 17. For example, if the number is set to 3, it refers to the ADC_IN3 channel. If PCLEN is less than 4, the conversion sequence starts from 4-PCLEN. For example, when ADC_PSQ ([21: 0]) =10 00110 00101 00100 00011, it indicates that the scan conversion follows the sequence: 4, 5, 6, not 3, 4,5.

18.6.13 ADC preempted data register x (ADC_ PDTx) (x=1..4)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 0	PDTx	0x0000	rw	Conversion data from preempted channel

18.6.14 ADC ordinary data register (ADC_ ODT)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	ADC2ODT	0x0000	ro	ADC2 conversion data of ordinary channel Note: These bits are reserved in ADC2 and ADC3. In ADC1, these bits are valid only in master/slave mode, and they contain the conversion result from the ADC2 ordinary channels.
Bit 15: 0	ODT	0x0000	ro	Conversion data of ordinary channel

18.6.15 ADC oversampling register (ADC_ OVSP)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 11	Reserved	0x0000	resd	Kept at its default value.

Bit 10	OOSRSEL	0x0	rw	Ordinary oversampling restart mode select When the ordinary oversampling is interrupted by preempted conversions, this bit can be used to select where to resume ordinary conversions. 0: Continuous mode (ordinary oversampling buffer will be reserved) 1: Restart mode (ordinary oversampling buffer will be cleared, that is, the previously oversampled times are reset)
Bit 9	OOSTREN	0x0	rw	Ordinary oversampling trigger mode enable 0: Disabled (only one trigger is needed for all oversampling conversions) 1: Enabled (Each oversampling conversion needs a trigger)
Bit 8: 5	OSSSEL	0x0	rw	Oversampling shift select This field is used to define the number of right-shift used in the oversampling results. 0000: No shift 0001: 1 bit 0010: 2 bits 0011: 3 bits 0100: 4 bits 0101: 5 bits 0110: 6 bits 0111: 7 bits 1000: 8 bits 1001~1111: Unused. Do not configure.
Bit 4: 2	OSRSEL	0x0	rw	Oversampling ratio select 000: 2x 001: 4x 010: 8x 011: 16x 100: 32x 101: 64x 110: 128x 111: 256x
Bit 1	POSEN	0x0	rw	Preempted oversampling enable 0: Preempted oversampling disabled 1: Preempted oversampling enabled
Bit 0	OOSEN	0x0	rw	Ordinary oversampling enable 0 : Ordinary oversampling disabled 1: Ordinary oversampling enabled

18.6.16 ADC calibration value register (ADC_CALVAL)

Accessed by words.

Bit	Register	Reset value	Type	Description
	Reserved	0x0		Kept at its default value.
	CALVAL	0x0		A/D Calibration value

18.6.17 ADC common status register (ADC_CSTS)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x0000	resd	Kept at its default value.
Bit 22	RDY3	0x0	ro	ADC3 conversion ready flag This bit is the mapping bit of the RDY bit in the ADC3_STS register.

Bit 21	OCCO3	0x0	ro	ADC3 ordinary channel conversion overflow flag This bit is the mapping bit of the OCCO bit in the ADC3_STS register.
Bit 20	OCCS3	0x0	ro	ADC3 ordinary channel conversion start flag This bit is the mapping bit of the OCCS bit in the ADC3_STS register.
Bit 19	PCCS3	0x0	ro	ADC3 Preempted channel conversion start flag This bit is the mapping bit of the PCCS bit in the ADC3_STS register.
Bit 18	PCCE3	0x0	ro	ADC3 preempted channels conversion end flag This bit is the mapping bit of the PCCE bit in the ADC3_STS register.
Bit 17	OCCE3	0x0	ro	ADC3 ordinary channels conversion end flag This bit is the mapping bit of the OCCE bit in the ADC3_STS register.
Bit 16	VMOR3	0x0	ro	ADC3 voltage monitoring out of range flag This bit is the mapping bit of the VMOR bit in the ADC3_STS register.
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	RDY2	0x0	ro	ADC2 conversion ready flag This bit is the mapping bit of the RDY bit in the ADC2_STS register.
Bit 13	OCCO2	0x0	ro	ADC2 ordinary channel conversion overflow flag This bit is the mapping bit of the OCCO bit in the ADC2_STS register.
Bit 12	OCCS2	0x0	ro	ADC2 ordinary channel conversion start flag This bit is the mapping bit of the OCCS bit in the ADC2_STS register.
Bit 11	PCCS2	0x0	ro	ADC2 preempted channel conversion start flag This bit is the mapping bit of the PCCS bit in the ADC2_STS register.
Bit 10	PCCE2	0x0	ro	ADC2 Preempted channels conversion end flag This bit is the mapping bit of the PCCE bit in the ADC2_STS register.
Bit 9	OCCE2	0x0	ro	ADC2 ordinary channels conversion end flag This bit is the mapping bit of the OCCE bit in the ADC2_STS register.
Bit 8	VMOR2	0x0	ro	ADC2 voltage monitoring out of range flag This bit is the mapping bit of the VMOR bit in the ADC2_STS register.
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6	RDY1	0x0	ro	ADC1 conversion ready flag This bit is the mapping bit of the RDY bit in the ADC1_STS register.
Bit 5	OCCO1	0x0	ro	ADC1 ordinary channel conversion overflow flag This bit is the mapping bit of the OCCO bit in the ADC1_STS register.

Bit 4	OCCS1	0x0	ro	ADC1 ordinary channel conversion start flag This bit is the mapping bit of the OCCS bit in the ADC1_STS register.
Bit 3	PCCS1	0x0	ro	ADC1 Preempted channel conversion start flag This bit is the mapping bit of the PCCS bit in the ADC1_STS register.
Bit 2	PCCE1	0x0	ro	ADC1 preempted channels conversion end flag This bit is the mapping bit of the PCCE bit in the ADC1_STS register.
Bit 1	OCCE1	0x0	ro	ADC1 ordinary channels conversion end flag This bit is the mapping bit of the OCCE bit in the ADC1_STS register.
Bit 0	VMOR1	0x0	ro	ADC1 voltage monitoring out of range flag This bit is the mapping bit of the VMOR bit in the ADC1_STS register.

18.6.18 ADC common control register (ADC_CSTS)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x0000	resd	Kept at its default value.
Bit 23	ITSRVEN	0x0	rw	Internal temperature sensor and V_{INTRV} enable 0: Disabled 1: Enabled
Bit 22	VBATEN	0x0	rw	V_{BAT} enable 0: Disabled 1: Enabled
Bit 21: 20	Reserved	0x0	resd	Kept at its default value.
Bit 19: 16	ADCDIV	0x0	rw	ADC division 0000: HCLK/2 0001: HCLK/3 … 1111: HCLK/17 Note: The clock divided by this field are used by all ADCs. The maximum value of the ADCCLK is 80 MHz. After this division, the ADCCLK cannot be higher than PCLK2.
Bit 28 Bit 15: 14	MSDMASEL	0x0	rw	Ordinary channel DMA transfer mode select in master/ slave mode MSDMASEL[2] is the 28th bit in the ADC_CCTRL register. MSDMASEL[2: 0] is defined as follows: 000: No DMA transfer 001: DMA mode 1 010: DMA mode 2 011: DMA mode 3 100: DMA mode 4 101: DMA mode 5 110~111: Unused. Do not configure. Note: This field is applicable in master/slave mode. Refer to Section 18.5.1 for details on DMA mode x.

Bit 13	MSDRCEN	0x0	rw	Ordinary channel DMA request continuation enable in master/slave mode 0: Disabled (After the completion of the programmed number of DMA transfers, no DMA request generated at the end of ordinary conversion) 1: Enabled (Don't care about the programmed number of DMA transfers, Each ordinary channel sends DMA request at the end of ordinary conversion) Note: This bit is applicable in master/slave mode and when DMA mode x is selected through the MSDMASEL bit.
Bit 12	Reserved	0x0	resd	Kept at its default value.
Bit 11: 8	ASISEL	0x0	rw	Adjacent ADC sampling interval select in ordinary shift mode 0000: 5 * TADCCLK 0001: 6 * TADCCLK 0010: 7 * TADCCLK ... 1111: 20 * TADCCLK Note: This field is used to configure the conversion interval between adjacent ADCs in ordinary shift mode. Refer to Section 18.5.4 for details.
Bit 7: 5	Reserved	0x0	resd	Kept at its default value.
Bit 4: 0	MSSEL	0x00	rw	Combined master/slave mode select 00000: Non-combined mode 00001: Combined ordinary simultaneous+preempted simultaneous modes (single slave) 00010: Combined ordinary simultaneous+alternate preempted trigger modes (single slave) 00011~00100: Unused. Do not configure 00101: Preempted simultaneous mode (single slave) 00110: Ordinary simultaneous mode (single slave) 00111: Ordinary shift mode (single slave) 01000: Unused. Do not configure 01001: Alternate preempted trigger mode (single slave) 01010~10000: Unused. Do not configure 10001: Combined ordinary simultaneous+preempted simultaneous modes (dual slave) 10010: Combined ordinary simultaneous+alternate preempted trigger modes (dual slave) 10011~10100: Unused. Do not configure 10101: Preempted simultaneous mode (dual slave) 10110: Ordinary simultaneous mode (dual slave) 10111: Ordinary shift mode (dual slave) 11000: Unused. Do not configure 11001: Alternate preempted trigger mode (dual slave) 11010~11111: Unused. Do not configure Note: In combined master/slave mode, the change of configuration will cause loss of synchronization between master and slave. Thus it is recommended to disable combined master/slave mode before changing.

18.6.19 ADC common data register (ADC_CODT)

Accessed by words.

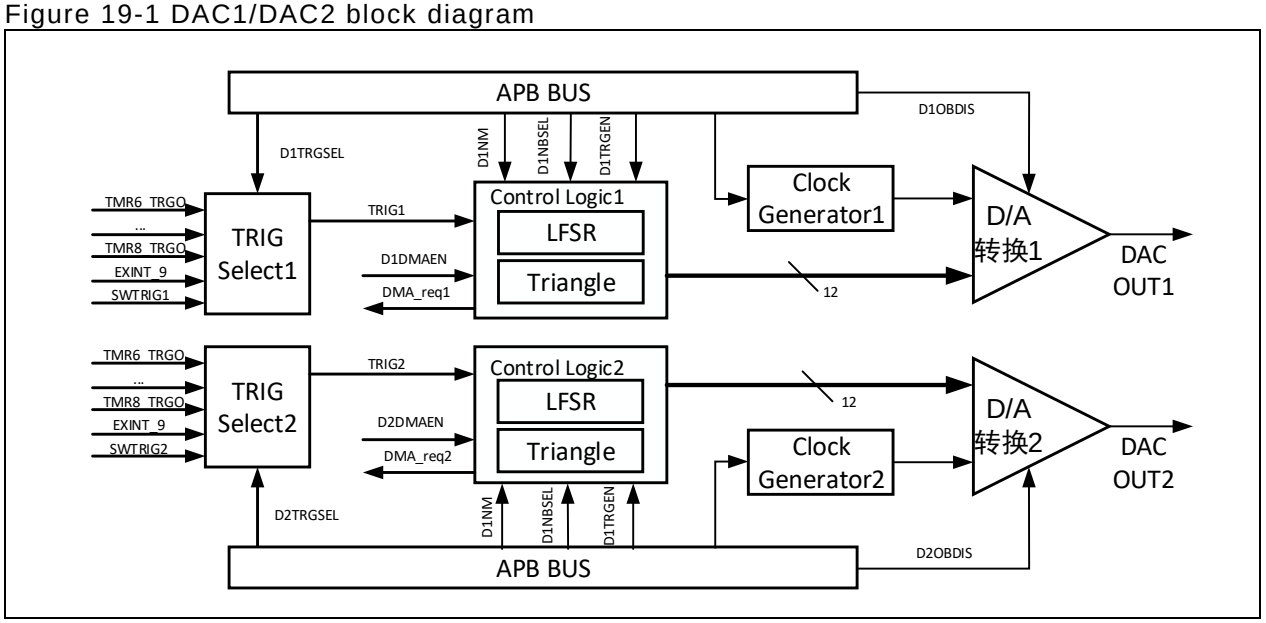
Bit	Register	Reset value	Type	Description
Bit 31: 16	CODTH	0x0000	rw	Ordinary conversion high halfword data in master slave mode Note: The meanings of data in this field vary from DMA mode to DMA mode. Refer to Section 18.5.1 for details.
Bit 15: 0	CODTL	0x0000	rw	Ordinary conversion low halfword data in master slave mode Note: The meanings of data in this field vary from DMA mode to DMA mode. Refer to Section 18.5.1 for details.

19 Digital-to-analog converter (DAC)

19.1 DAC introduction

The DAC uses a 12-bit digital input to generate an analog output between 0 and reference voltage. The digital part of the DAC can be configured in 8-bit or 12-bit mode and can be used in conjunction with the DMA. It supports left or right alignment in a single /dual DAC modes. It has two output channels, DAC1 and DCA2, with its own converter each. Each DAC1/DAC2 can be converted independently or simultaneously in dual DAC mode. The input reference voltage V_{REF+} makes conversion more accuracy.

Figure 19-1 DAC1/DAC2 block diagram



19.2 DAC main features

- A single/dual DAC 8-bit or 12-bit digital input
- Left or right data alignment
- Noise-wave/Triangular-wave generation
- Dual DAC or single DAC1/DAC2 independent conversions
- DMA mode for DAC1/DAC2
- Software or external triggers for conversion
- Input reference voltage V_{REF+}

19.3 Design tips

The following information can be used as DAC design reference:

- Analog module configuration
  The analog part of the DAC1/DAC2 can be enabled by setting the ENx bit in the DAC_CTRL register, but its digital part is not subject to this bit. The DAC integrates two output gains that can be used to reduce the output impedance, and to drive external loads directly without the need of an external operational amplifier. The DAC1/DAC2 output gain can be enabled and disabled through the DxOBDIS bit in the DAC_CTRL register.
- DMA capability

  The DAC1/DAC2 both have a DMA capability that can be enabled by setting the DxDMAEN bit in the DAC_CTRL register. One DMA request is generated when a trigger signal is active while the DxTRGEN bit is set. The DAC DMA request is not added up, meaning the new DAM request will be ignored and no error is reported.

  In dual DAC mode, the application can handle two channels (DAC1/DAC2) by using only one DMA request and a DMA channel.

- DMA underflow

  When the DAC DMA request is enabled, an overflow occurs if a second external trigger arrives before the acknowledgement for the first external trigger is received. In this case, no new external trigger is handled, or no new DMA request is issued, and the DxDMAUDRF bit in the DAC_SR register is set, reporting the error condition. An interrupt is generated if its corresponding DxDMAUDRIEN bit in the DAC_CTRL register is set.

  The software clears the DxDMAUDRF bit by writing 1. The DxDMAEN bit should be cleared in the DAC_CTRL register before re-starting a DMA transfer and re-initializing DMA and DAC.

- Input/output configuration

  The digital inputs are linearly converted to analog voltage outputs by the DAC, and it is between 0 and V_{REF+}. The analog DAC module is supplied by VDDA. The positive analog reference voltage input falls between 2.0 V and VDDA. To avoid parasitic interruption and excessive consumption, the PA4 or PA5 should be configured to analog input.

  DAC output = V_{REF+} x (DxODT[11: 0]/4095)

19.4 Functional overview

19.4.1 Trigger events

If the DxTRGEN bit in the DAC_CTRL register is set, the DAC conversion can then be triggered by an external event (timer counter, external interrupt line) or by software. The DxTRGSEL[2: 0] is used to select trigger sources.

Table 19-1 Trigger source selection

Source	DxTRGSEL [2:0]	Description
TMR6_TRGOUT	000	On-chip signals
TMR8_TRGOUT	001	
TMR7_TRGOUT	010	
TMR5_TRGOUT	011	
TMR2_TRGOUT	100	
TMR4_TRGOUT	101	
EXINT_9	110	External signals
DxSWTRG	111	Software trigger

When the DxTRGEN bit is set, the data stored into the HDRx register is transferred into the DAC_DxODT register each time a DAC detects an active trigger event,. If the software trigger is selected, the DxSWTRG flag is cleared by hardware after writing 1. The DAC output becomes active after a period of time once the data is loaded into the DAC_DxODT register.

When the DxTRGEN bit is cleared, each data written to the data register is immediately transferred into the DAC_DxODT register without the need of a trigger event.

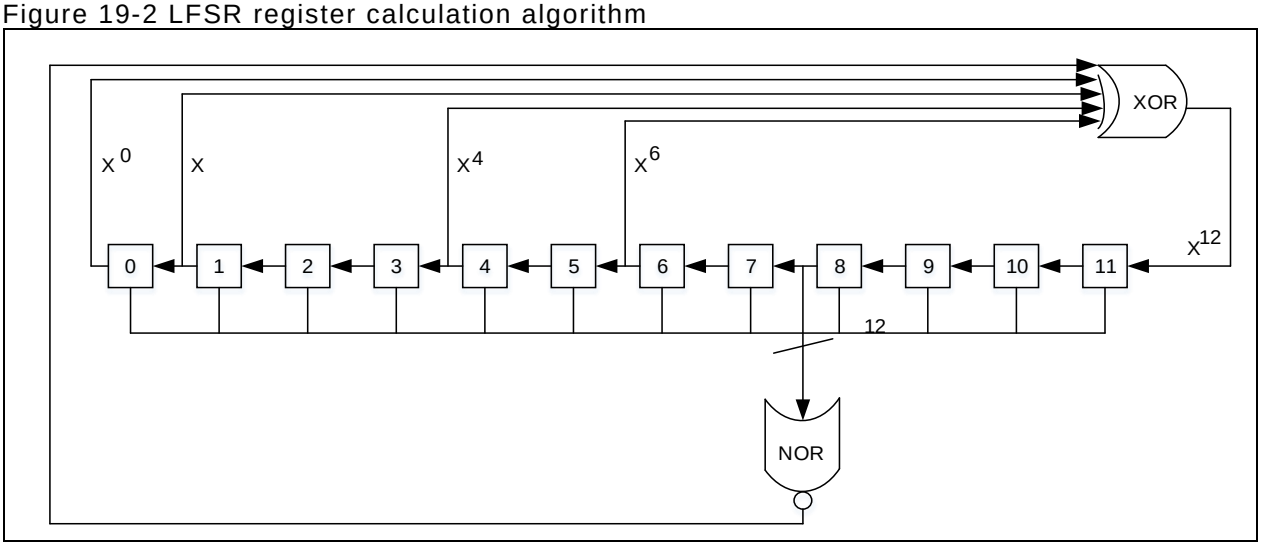
19.4.2 Noise/Triangular-wave generation

The DAC can output a variable-amplitude pseudo noise and a triangular wave, which is done by the LENinear Feedback Shift Register and triangle wave generator respectively. The DAC variable-amplitude pseudo noise generation is selected by setting DxNM[1:0]=01 in the LFSR, while the DAC triangular-wave generation is selected by setting the DxNM[1:0]=1x.

LFSR logic

The preloaded value in the LFSR is 0xAAA. This register is updated after each trigger event based on a specific calculation algorithm.

Figure 19-2 LFSR register calculation algorithm



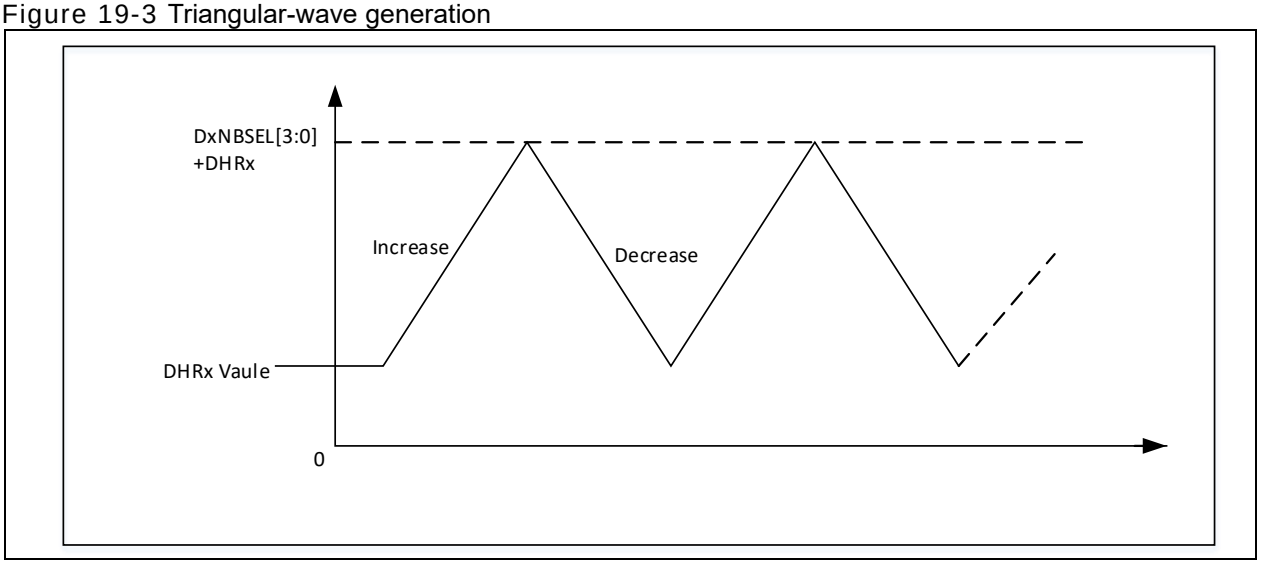
The DxNBSEL [3: 0] bit in the DAC_CTRL register is set to mark partially or totally the LFSR data. The resulting value is then added up to the DHRx value without overflow and this value is loaded into the DAC_DxODT register. It is possible to disable LFSR function and reset LFSR wave generation algorithm by setting DxNM[1: 0]=00.

Triangular wave logic

The DAC triangular-wave generation is selected by setting DxNM[1: 0]=1x. The amplitude is configured through the DxNBSEL [3: 0] bit in the DAC_CTRL register. An internal triangular-wave counter is incremented at each trigger event. Once the maximum amplitude programmed in the DxNBSEL [3: 0] is reached, the value of this counter is decremented down to 0, then incremented again, and so on.

Meanwhile, the value of this counter is then added up to the DHRx register without overflow and the resulting value is loaded into the DAC_DxODT register. It is possible to disable/reset the triangular-wave generation by setting DxNM[1: 0]=00.

Figure 19-3 Triangular-wave generation



19.4.3 DAC data alignment

The DAC supports a single DAC and dual DA mode. The data format is dependent on the selected configuration mode.

Single DAC data format:

8-bit right alignment: load data into the DAC_DxDTH8R [7:0]

12-bit left alignment: load data into the DAC_DxDTH12L [15: 4]

12-bit right alignment: load data in the DAC_DxDTH12R [11: 0]

Dual DAC data format:

8-bit right alignment: load data into the DAC_DDTH8R [7: 0] and DAC_DDTH8R [15: 8]

12-bit left alignment: load data into the DAC_DDTH12L [15: 4] and DAC_DDTH12L [31: 20]

12-bit right alignment: load data into the DAC_DDTH12R [11: 0] and DAC_DDTH12R [27:16]

The loaded 8-bit data corresponds to the DHRx[11:4] and the loaded 12-bit data corresponds to the DHRx[11: 0]

19.5 DAC registers

These peripheral registers must be accessed by word (32 bits).

Table 19-2 DAC register map and reset values

Register name	Offset	Reset value
DAC_CTRL	000h	0x0000 0000
DAC_SWTRG	004h	0x0000 0000
DAC_D1DTH12R	008h	0x0000 0000
DAC_D1DTH12L	00Ch	0x0000 0000
DAC_D1DTH8R	010h	0x0000 0000
DAC_D2DTH12R	014h	0x0000 0000
DAC_D2DTH12L	018h	0x0000 0000
DAC_D2DTH8R	01Ch	0x0000 0000
DAC_DDTH12R	020h	0x0000 0000
DAC_DDTH12L	024h	0x0000 0000
DAC_DDTH8R	028h	0x0000 0000
DAC_D1ODT	02Ch	0x0000 0000
DAC_D2ODT	030h	0x0000 0000
DAC_STS	034h	0x0000 0000

19.5.1 DAC control register (DAC_CTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value
Bit 29	D2DMAUDRIEN	0x0	rw	DAC2 DMA transfer underrun interrupt enable This bit is set and cleared by software. 0: DAC2 DMA transfer underrun interrupt disabled 1: DAC2 DMA transfer underrun interrupt enabled
Bit 28	D2DMAEN	0x0	rw	DAC2 DMA transfer enable This bit is set and cleared by software. 0: DAC2 DMA mode disabled 1: DAC2 DMA mode enabled
Bit 27: 24	D2NBSEL	0x0	rw	DAC2 noise bit select These bits are used to select the mark bit in noise generation mode or amplitude in triangular-wave generation mode. 0000: Unmask LSFR bit0 /Triangle amplitude is equal to 1 0001: Unmask LSFR bit[1: 0] /Triangle amplitude is equal to 3 0010: Unmask LSFR bit[2: 0] /Triangle amplitude is equal to 7 0011: Unmask LSFR bit[3: 0] /Triangle amplitude is equal to 15 0100: Unmask LSFR bit[4: 0] /Triangle amplitude is equal to 31 0101: Unmask LSFR bit[5: 0] /Triangle amplitude is equal to 63 0110: Unmask LSFR bit[6: 0] /Triangle amplitude is equal to 127

				0111: Unmask LSFR bit[7: 0] /Triangle amplitude is equal to 255 1000: Unmask LSFR bit[8: 0] /Triangle amplitude is equal to 511 1001: Unmask LSFR bit[9: 0] /Triangle amplitude is equal to 1023 1010: Unmask LSFR bit[10:0] /Triangle amplitude is equal to 2047 ≥1011: Unmask LSFR bit[11: 0] /Triangle amplitude is equal to 4095
Bit 23: 22	D2NM	0x0	rw	DAC2 noise mode 00: Wave generation disabled 01: Noise wave generation enabled 1x: Triangular wave generation enabled
Bit 21: 19	D2TRGSEL	0x0	rw	DAC2 trigger select 000: TMR6 TRGOUT event 001: TMR8 TRGOUT event 010: TMR7 TRGOUT event 011: TMR5 TRGOUT event 100: TMR2 TRGOUT event 101: TMR4 TRGOUT event 110: External interrupt line 9 111: Software trigger Note: These bits can be valid only when D2TRGEN = 1.
Bit 18	D2TRGEN	0x0	rw	DAC2 trigger enable 0: DAC2 trigger disabled 1: DAC2 trigger enabled Note: When the DAC2 trigger is disabled, the data written into the DAC_D2DTHx register is transferred into the DAC_D2ODT register after one APB1 clock cycle. When the DAC2 trigger is enabled, the data written into the DAC_D2DTHx register is transferred into the DAC_D2ODT register after three APB1 clock cycles. If the software trigger is selected, it takes one APB1 clock cycle to have the data written into the DAC_D2DTHx register transferred into the DAC_D2ODT register.
Bit 17	D2OBDIS	0x0	rw	DAC2 output buffer disable 0: DAC2 output buffer enabled 1: DAC2 output buffer disabled
Bit 16	D2EN	0x0	rw	DAC2 enable 0: DAC2 disabled 1: DAC2 enabled
Bit 15: 14	Reserved	0x0	resd	Kept at its default value
Bit 13	D1DMAUDRIEN	0x0	rw	DAC1 DMA transfer underrun interrupt enable This bit is set and cleared by software. 0: DAC1 DMA transfer underrun interrupt disabled 1: DAC1 DMA transfer underrun interrupt enabled
Bit 12	D1DMAEN	0x0	rw	DAC1 DMA transfer enable 0: DAC1 DMA transfer disabled 1: DAC1 DMA transfer enabled
Bit 11: 8	D1NBSEL	0x0	rw	DAC1 noise bit select These bits are used to select the mark bit in noise generation mode or amplitude in triangular-wave generation mode. 0000: Unmask LSFR bit0/Triangle amplitude is equal to 1 0001: Unmask LSFR bit[1:0]/Triangle amplitude is equal to 3 0010: Unmask LSFR bit[2: 0]/Triangle amplitude is equal to 7 0011: Unmask LSFR bit[3: 0]/Triangle amplitude is equal to 15 0100: Unmask LSFR bit[4: 0]/Triangle amplitude is equal to 31

				0101: Unmask LSFR bit[5: 0]/Triangle amplitude is equal to 63 0110: Unmask LSFR bit[6: 0]/Triangle amplitude is equal to 127 0111: Unmask LSFR bit[7: 0]/Triangle amplitude is equal to 255 1000: Unmask LSFR bit[8: 0]/Triangle amplitude is equal to 511 1001: Unmask LSFR bit[9: 0]/Triangle amplitude is equal to 1023 1010: Unmask LSFR bit[10: 0]/Triangle amplitude is equal to 2047 ≥1011: Unmask LSFR bit[11:0]/Triangle amplitude is equal to 4095
Bit 7: 6	D1NM	0x0	rw	DAC1 noise mode 00: Wave generation disabled 01: Noise wave generation enabled 1x: Triangular wave generation enabled
Bit 5: 3	D1TRGSEL	0x0	rw	DAC1 trigger select 000: TMR6 TRGOUT event 001: TMR8 TRGOUT event 010: TMR7 TRGOUT event 011: TMR5 TRGOUT event 100: TMR2 TRGOUT event 101: TMR4 TRGOUT event 110: External interrupt line 9 111: Software trigger Note: These bits can be valid only when D1TRGEN = 1.
Bit 2	D1TRGEN	0x0	rw	DAC1 trigger enable 0: DAC1 trigger disabled 1: DAC1 trigger enabled Note: When the DAC1 trigger is disabled, the data written into the DAC_D1DTHx register is transferred into the DAC_D1ODT register after one APB1 clock cycle. When the DAC1 trigger is enabled, the data written into the DAC_D1DTHx register is transferred into the DAC_ D1ODT register after three APB1 clock cycles If the software trigger is selected, it takes one APB1 clock cycle to have the data written into the DAC_D1DTHx register transferred into the DAC_ D1ODT register.
Bit 1	D1OBDIS	0x0	rw	DAC1 output buffer disable 0: DAC1 output buffer enabled 1: DAC1 output buffer disabled
Bit 0	D1EN	0x0	rw	DAC1 enable 0: DAC1 disabled 1: DAC1 enabled

19.5.2 DAC software trigger register (DAC_SWTRG)

Bit	Register	Reset value	Type	Description
Bit 31: 2	Reserved	0x0000 0000	resd	Kept at its default value
Bit 1	D2SWTRG	0x0	rw	DAC2 software trigger 0: DAC2 software trigger disabled 1: DAC2 software trigger enabled Note: This bit is cleared by hardware (one APB1 clock cycle later) once the DAC_D2DTH data is loaded into the DAC_D2ODT register.
Bit 0	D1SWTRG	0x0	rw	DAC1 software trigger 0: DAC1 software trigger disabled 1: DAC1 software trigger enabled Note: This bit is cleared by hardware (one APB1 clock cycle later) once the DAC_D1DTH data is loaded into the DAC_D1ODT register.

19.5.3 DAC1 12-bit right-aligned data holding register (DAC_D1DTH12R)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value
Bit 11: 0	D1DT12R	0x000	rw	DAC1 12-bit right-aligned data

19.5.4 DAC1 12-bit left-aligned data holding register (DAC_D1DTH12L)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 4	D1DT12L	0x000	rw	DAC1 12-bit left-aligned data
Bit 3: 0	Reserved	0x0	resd	Kept at its default value

19.5.5 DAC1 8-bit right-aligned data holding register (DAC_D1DTH8R)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value
Bit 7: 0	D1DT8R	0x00	rw	DAC1 8-bit right-aligned data

19.5.6 DAC2 12-bit right-aligned data holding register (DAC_D2DTH12R)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value
Bit 11: 0	D2DT12R	0x000	rw	DAC2 12-bit right-aligned data

19.5.7 DAC2 12-bit left-aligned data holding register (DAC_D2DTH12L)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 4	D2DT12L	0x000	rw	DAC2 12-bit left-aligned data
Bit 3: 0	Reserved	0x0	resd	Kept at its default value

19.5.8 DAC2 8-bit right-aligned data holding register (DAC_D2DTH8R)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value
Bit 7: 0	D2DT8R	0x00	rw	DAC2 8-bit right-aligned data

19.5.9 Dual DAC 12-bit right-aligned data holding register (DAC_DDTH12R)

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0	resd	Kept at its default value
Bit 27: 16	DD2DT12R	0x000	rw	DAC2 12-bit right-aligned data
Bit 15: 12	Reserved	0x0	resd	Kept at its default value
Bit 11: 0	DD1DT12R	0x000	rw	DAC1 12-bit right-aligned data

19.5.10 Dual DAC 12-bit left-aligned data holding register (DAC_DDTH12L)

Bit	Register	Reset value	Type	Description
Bit 31: 20	DD2DT12L	0x000	rw	DAC2 12-bit left-aligned data
Bit 19: 16	Reserved	0x0	resd	Kept at its default value
Bit 15: 4	DD1DT12L	0x000	rw	DAC1 12-bit left-aligned data
Bit 3: 0	Reserved	0x0	resd	Kept at its default value

19.5.11 Dual DAC 8-bit right-aligned data holding register (DAC_DDTH8R)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 8	DD2DT8R	0x00	rw	DAC2 8-bit right-aligned data
Bit 7: 0	DD1DT8R	0x00	rw	DAC1 8-bit right-aligned data

19.5.12 DAC1 data output register (DAC_ D1ODT)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value
Bit 11: 0	D1ODT	0x000	rw	DAC1 output data

19.5.13 DAC2 data output register (DAC_ D2ODT)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value
Bit 11: 0	D2ODT	0x000	rw	DAC2 output data

19.5.14 DAC status register (DAC_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value
Bit 29	D2DMAUDRF	0x0	w1c	DAC2 DMA transfer underrun flag 0: No DAC2 DMA transfer underrun 1: DAC2 DMA transfer underrun occurs Note: This bit is cleared by writing 1.
Bit 28: 14	Reserved	0x0000	resd	Kept at its default value
Bit 13	D1DMAUDRF	0x0	w1c	DAC1 DMA transfer underrun flag 0: No DAC1 DMA transfer underrun 1: DAC1 DMA transfer underrun occurs Note: This bit is cleared by writing 1.
Bit 12: 0	Reserved	0x0000	resd	Kept at its default value

20 CAN

20.1 CAN introduction

CAN (Controller Area Network) is a distributed serial communication protocol for real-time and reliable data communication among various nodes. It supports the CAN protocol version 2.0A and 2.0B.

20.2 CAN main features

- Baud rates up to 1M bit/s
- Supports the time triggered communication
- Interrupt enable and mask
- Configurable automatic retransmission mode

Transmission

- Three transmit mailboxes
- Configurable transmit priority
- Supports the time stamp on transmission

Reception

- Two FIFOs with three-level depth
- 28 filter banks
- Supports the identifier list mode
- Supports the identifier mask mode
- FIFO overrun management

Time triggered communication mode

- 16-bit timers
- Time stamp on transmission

20.3 Baud rate

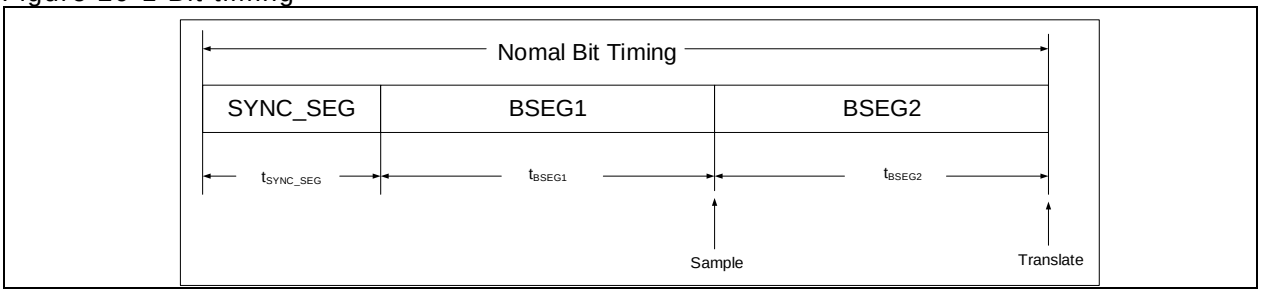
The nominal bit time of the CAN bus consists of three parts as follows:

Synchronization segment (SYNC_SEG): This segment has one time unit, and its time duration is defined by the BRDIV[11: 0] bit in the CAN_BTMG register.

Bit segment 1 (BIT SEGMENT 1): It is referred to as BSEG1 including the PROP_SEG and PHASE_SEG1 of the CAN standard. Its duration is between 1 and 16 time units, defined by the BTS1[3: 0] bit.

Big segment 2 (BIT SEGMENT 2): It is referred to as BSEG2 including the PHASE_SEG2 of the CAN standard. Its duration is between 1 and 8 time units, defined by the BTS2[2: 0] bit.

Figure 20-1 Bit timing



Baud rate formula:

$$BaudRate = \frac{1}{\text{Nomal Bit Timimg}}$$

$$Nomal\ \ Bit\ \ Timing = t_{SYNC_SEG} + t_{BSEG1} + t_{BSEG2}$$

where

$$t_{SYNC_SEG} = 1\ \text{x}\ t_q$$

$$t_{BSEG1} = (1 + \text{BTS1}[3:0])\ \text{x}\ t_q$$

$$t_{BSEG2} = (1 + \text{BTS2}[2:0])\ \text{x}\ t_q$$

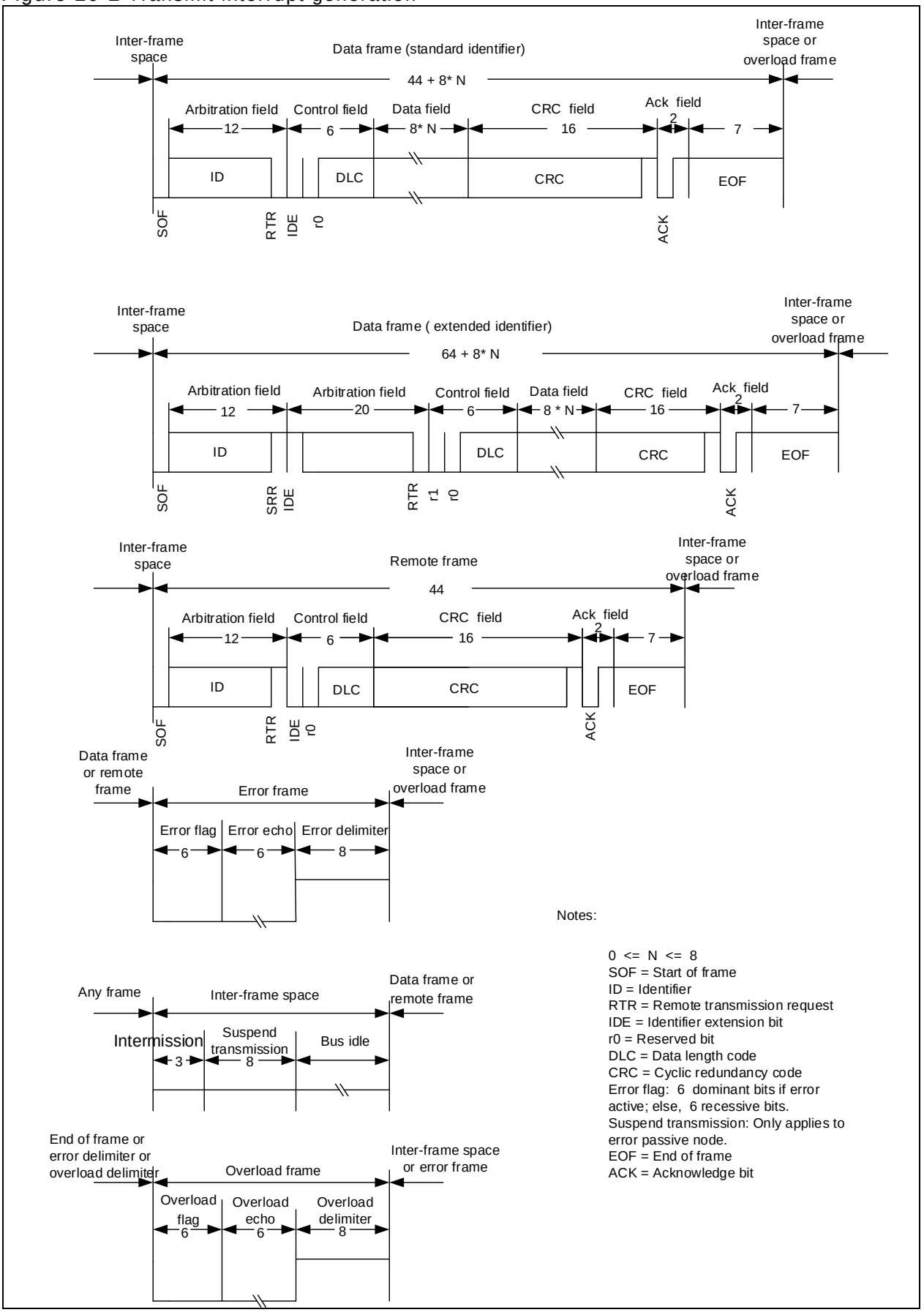
$$t_q = (1 + \text{BRDIV}[11:0])\ \text{x}\ t_{pclk}$$

Hard synchronization and resynchronization

The start location of each bit in CAN nodes is always in synchronization segment by default, and the sampling is performed at the edge location of bit segment 1 and big segment 2 simultaneously.

During the actual transmission, each bit of the CAN nodes has certain phase error due to the oscillator drift, transmission delay among the network nodes and noise interference. To avoid the impact on the communication, the start-bit edge and its subsequent falling edge can be synchronized or resynchronized. The time length of the synchronization compensation can not be greater than the resynchronization width (1 to 4 time units, defined by the RSAW[1: 0] bit).

Figure 20-2 Transmit interrupt generation



Notes:

0 <= N <= 8
SOF = Start of frame
ID = Identifier
RTR = Remote transmission request
IDE = Identifier extension bit
r0 = Reserved bit
DLC = Data length code
CRC = Cyclic redundancy code
Error flag: 6 dominant bits if error active; else, 6 recessive bits.
Suspend transmission: Only applies to error passive node.
EOF = End of frame
ACK = Acknowledge bit

20.4 Interrupt management

The CAN controller contains four interrupt vectors that can be used to enable or disable interrupts by setting the CAN_INTEN register.

Figure 20-3 Transmit interrupt generation

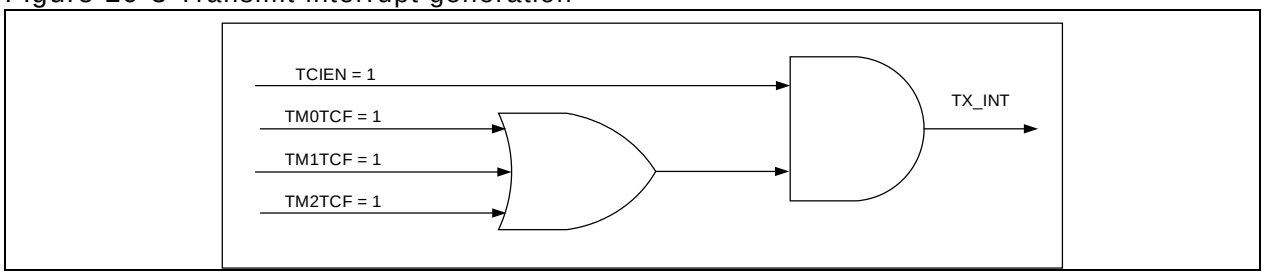


Figure 20-4 Receive interrupt 0 generation

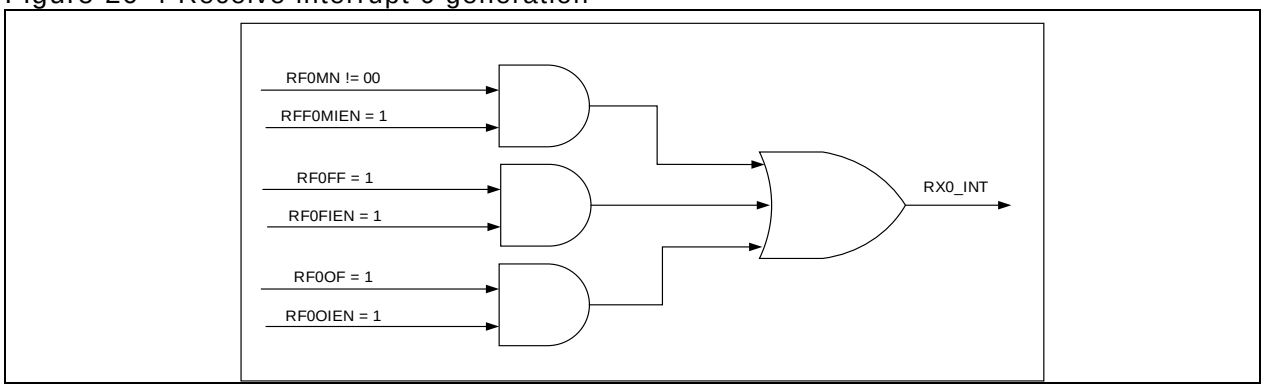


Figure 20-5 Receive interrupt 1 generation

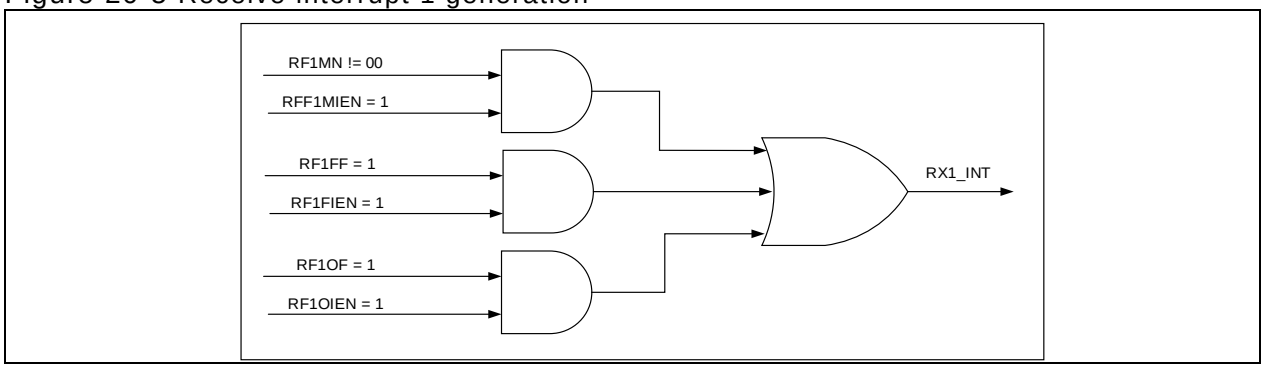
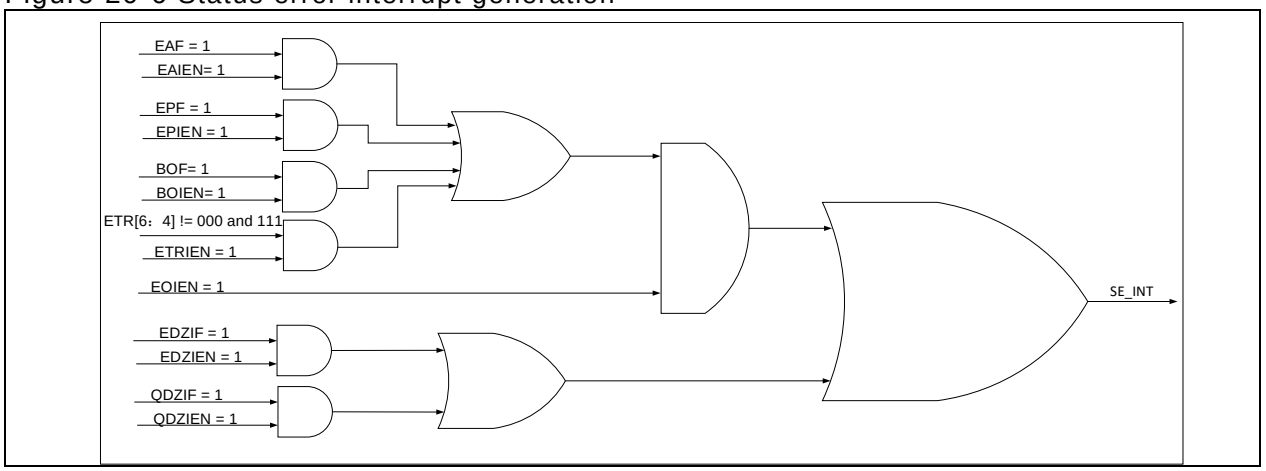


Figure 20-6 Status error interrupt generation



20.5 Design tips

The following information can be used as reference for CAN application development:

- Debug control
  When the system enters the debug mode, the CAN controller stops or continues to work normally, depending on the CANx_PAUSE bit in the DEBUG_CTRL register or the PTD bit in the CAN_MCTRL register.
- Time triggered communication
  The timer triggered communication is used to improve the real-time performance so as to avoid bus competition. It is activated by setting TTCEN=1 in the CAN_MCTRL register. The internal 16-bit timer is incremented each CAN bit time, and is sampled on the Start Of Frame bit to generate the time stamp value, which is stored in the CAN_RFCx and CAN_TMCx register.
- Register access protection
  The CAN_BTMG register can be modified only when the CAN is in frozen mode.

Although the transmission of incorrect data will not cause problems at the network level, it can have severe impact on the application. Thus a transmit mailbox can be modified only when it is in empty state.

The filter configuration in the CAN_FMCFG, CAN_FBWCFG and CAN_FRF registers can be modified only when FCS=1. The CAN_FiFBx register can be modified only when FCS=1 or FAENx=0.

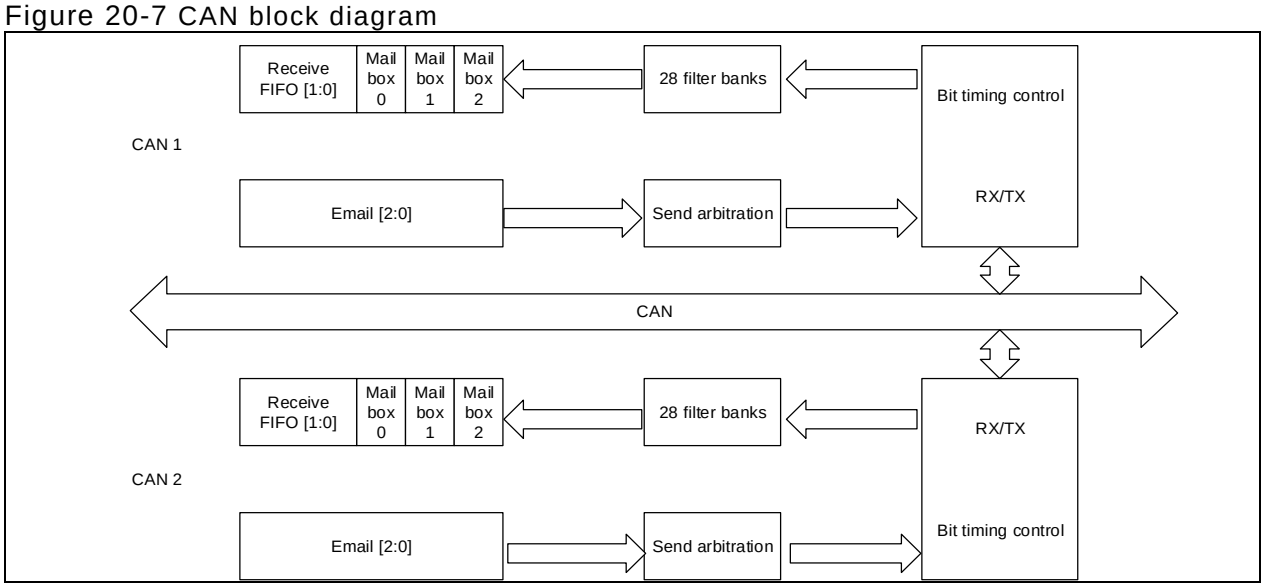
20.6 Functional overview

20.6.1 General description

As the number of nodes in the CAN network and the number of messages grows, an enhanced filtering mechanism is required to handle all types of messages in order to reduce the processing time of message reception. One FIFO scheme is used to ensure that the CPU can concentrate on application tasks for a long period of time without the loss of messages. In the meantime, the priority order of the messages to be transmitted is configured by hardware. Standard identifiers (11-bit) and extended identifiers (29-bit) are fully supported by hardware.

Based on the above mentioned conditions, the CAN controller provides 28 scalable/configurable identifier filter banks, 2 receive FIFOs with storing 3 complete messages each and being totally managed by hardware, and 3 transmit mailboxes with their transmit priority order defined by the transmit scheduler.

Figure 20-7 CAN block diagram



20.6.2 Operating modes

The CAN controller has three operating modes:

- Sleep mode

  After a system reset, the CAN controller is in Sleep mode. In this mode, the CAN clock is stopped to reduce power consumption and an internal pull-up resistance is disabled. However, the software can still access to the mailbox registers.

  The software request the CAN controller to enter Sleep mode by setting the DZEN bit in the CAN_MCTRL register. The hardware confirms the request by setting the DZC bit in the CAN_MSTS register.

  Exit Sleep mode in two ways: The CAN controller can be woke up by hardware clearing the DZEN bit when the AEDEN bit in the CAN_MCTRL register and the CAN bus activity is detected. It can also be woke up by software clearing the DZEN bit.

  Switch to Frozen mode: The CAN controller switches from Sleep mode to Frozen mode when the FZEN bit is set in the CAN_MCTRL register and the DZEN bit is cleared. Such switch operation is confirmed by hardware setting the FZC bit in the CAN_MSTS register.

  Switch to Communication mode: The CAN controller enters Communication mode when the FZEN and DZEN bits are both cleared and the CAN controller has synchronized with the bus. In other words, it must wait for 11 consecutive recessive bits to be detected on the CANRX pin.

- Frozen mode

  The software initialization can be done only in Frozen mode, including the CAN_BTMG and CAN_MCTRL registers. But the initialization of the 14 CAN filter banks (mode, scale, FIFO association, activation and filter values) can be done in non-Frozen mode. When the CAN controller is in Frozen mode, message reception and transmission are both disabled.

  Switch to Communication mode: The CAN controller leaves Frozen mode when the FZEN bit is cleared in the CAN_MCTRL register. This switch operation is confirmed by hardware clearing the FZC bit in the CAN_MSTS register. The CAN controller must be synchronized with the bus.

  Switch to Sleep mode: The CAN controller enters Sleep mode if DZEN=1 and FZEN=0 in the CAN_MCTRL register. This switch operation is confirmed by hardware setting the DZC bit in the CAN_MSTS register.

- Communication mode

  After the CAN_BTMG and CAN_MCTRL registers are configured in Frozen mode, the CAN controller enters Communication mode and is ready for message reception and transmission.

  Switch to Sleep mode: The CAN controller switches to Sleep mode when the DZEN bit is set in the CAN_MCTRL register and the current CAN bus transmission is complete.

  Switch to Frozen mode: The CAN controller enters Frozen mode when the FZEN bit is set in the CAN_MCTRL register and the current CAN bus transmission is complete.

20.6.3 Test modes

The CAN controller defines three test modes, including Listen-only mode, Loop back mode and combined Listen-only and Loop back mode. Test mode can be selected by setting the LOEN and LBEN bits in the CAN_BTMG register.

- Listen-only mode is selected when the LOEN bit is set in the CAN_BTMG register. In this mode, the CAN is able to receive data, but only recessive bits are output on the CANTX pin. In the meantime, the dominant bits output on the CANTX can be monitored by the receive side but without affecting the CAN bus.
- Loop back mode is selected by setting the LBEN bit in the CAN_BTMG register. In this mode, The CAN only receives the level signal on its CANTX pin. Meanwhile, the CAN can also send data to the external bus. The Loop back mode is mainly used for self-test functions.
- It is possible to combine the Listen-only and Loop back mode by setting the LOEN and LBEN bits in the CAN_BTMG register. In this case, the CAN is disconnected from the bus network, the CANTX pin remains in recessive state, and the transmit side is connected to the receive side.

20.6.4 Message filtering

The received message has to go through filtering by its identifier. If passed, the message will be stored in the corresponding FIFOs. If not, the message will be discarded. The whole operation is done by hardware without using CPU resources.

Filter bit width

The CAN controller provides 28 configurable and scalable filter banks (0~27). Each filter bank has two 32-bit registers, CAN_FiFB1 and CAN_FiFB2. The filter bit width can be configured as two 16 bits or one 32 bits, depending on the corresponding bits in the CAN_FBWCFG register.

32-bit filter register CAN_FiFBx includes the SID[10: 0], EID[17: 0], IDT and RTR bits.

CAN_FiFB1[31: 21]	CAN_FiFB1[20: 3]	CAN_FiFB1[2: 0]		
CAN_FiFB2[31: 21]	CAN_FiFB2[20: 3]	CAN_FiFB2[2: 0]		
SID[10: 0]/EID[28: 18]	EID[17: 0]	IDT	RTR	0

Two 16-bit filter register CAN_FiFBx includes SID[10: 0], IDT, RTR and EID[17: 15] bits

CAN_FiFB1[31: 21]	CAN_FiFB1 [20: 19]	CAN_FiFB1 [18: 16]	CAN_FiFB1[15: 5]	CAN_FiFB1 [4: 3]	CAN_FiFB1 [2: 0]	
CAN_FiFB2[31: 21]	CAN_FiFB2 [20: 19]	CAN_FiFB2 [18: 16]	CAN_FiFB2[15: 5]	AN_FiFB2 [4: 3]	CAN_FiFB2 [2: 0]	
SID[10: 0]	IDT	RTR	EID[17: 15]	SID[10: 0]	IDT	RTR	EID[17: 15]

Filtering mode

The filter can be configured in identifier mask mode or in identifier list mode by setting the FMSELx bit in the CAN_FMCFG register. The mask mode is used to specify which bits must match the pre-programmed identifiers, and which bits do not need. In identifier list mode, the identifier must match the pre-programmed identifier. The two modes can be used in conjunction with filter width to deliver four filtering modes below:

Figure 20-8 32-bit identifier mask mode

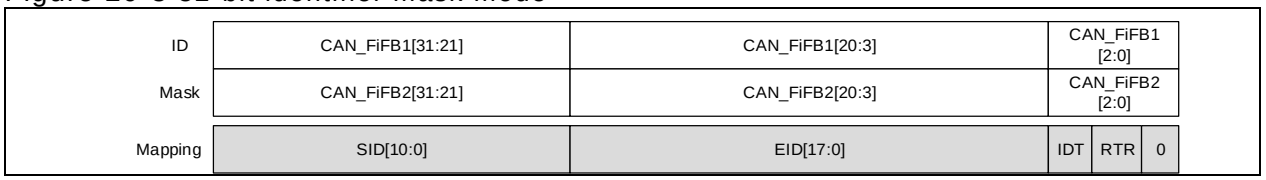


Figure 20-9 32-bit identifier list mode

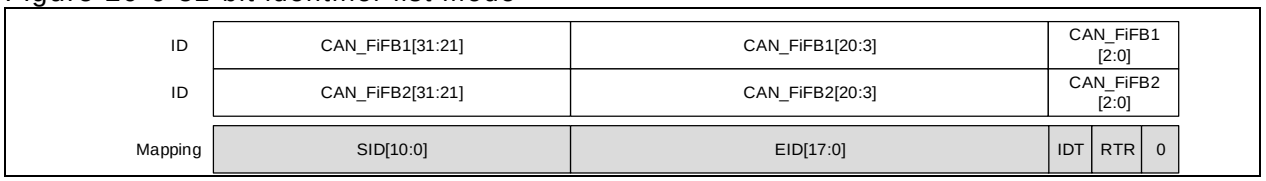


Figure 20-10 16-bit identifier mask mode

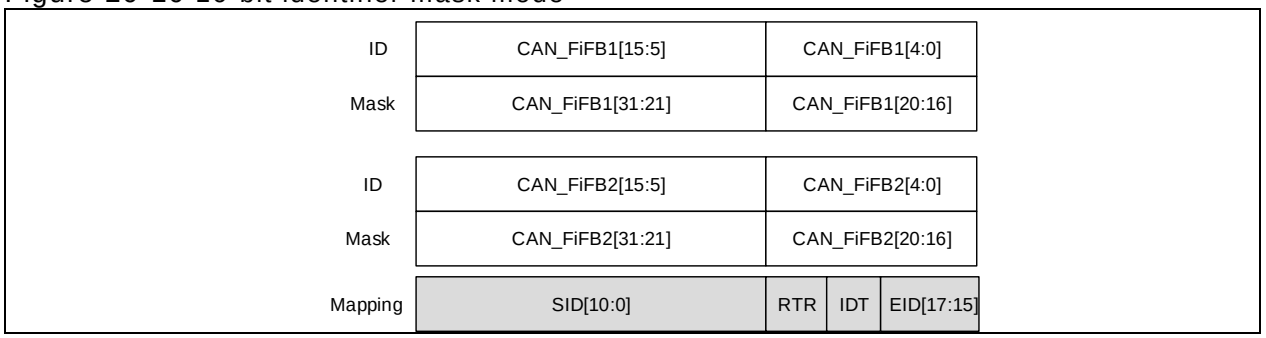
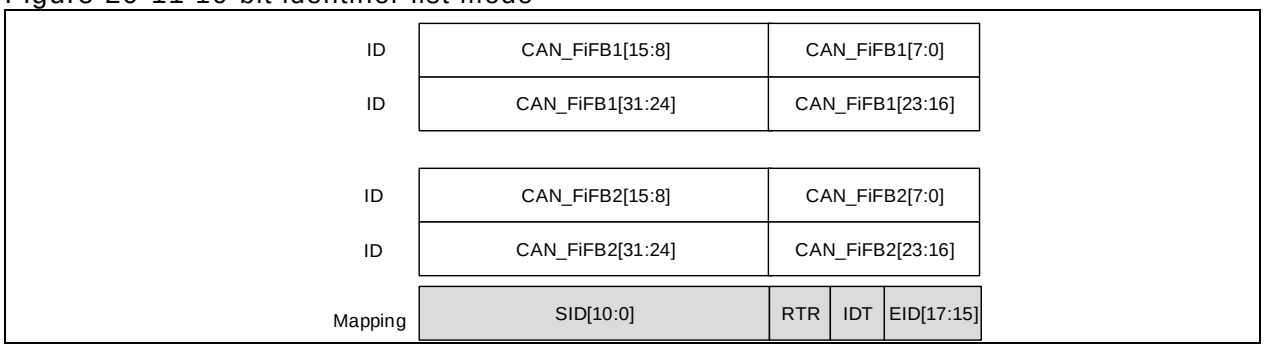


Figure 20-11 16-bit identifier list mode



Filter match number

28 filter banks have different filtering effects dependent on the bit width mode. For example, 32-bit identifier mask mode contains the filters numbered n while 16-bit identifier list mode contains the filters numbered n, n+1, n+2 and n+3. When a frame of message passes through the filter number N, the number N is stored in the RFFMN[7: 0] bit in the CAN_RFCx register. The distribution of the filter number does not take into account the activation state of the filter banks.

Priority rules

When the CAN controller receives a frame of message, the message may pass through several filters. In this case, the filter match number stored in the receive mailbox is determined according to the following priority rules:

- A 32-bit filter has priority over a 16-bit filter
- For filters with identical bit width, the identifier list mode has priority over the identifier mask mode
- For filter with identical bit width and identifier mode, the lower number has priority over the higher number.

Filter configuration

- The CAN filters are configured by setting the FCS bit in the CAN_FCTRL register.
- Identifier mask mode or identifier list mode can be selected by setting the FMSELx bit in the CAN_FMCFG register.
- The filter bit width can be configured as two 16 bits or one 32 bits by setting the FBWSELx bit in the CAN_FBWCFG register.
- The filter x is associated with FIFO0 or FIFO1 by setting the FRFSELx bit in the CAN_FRF register.
- The filter banks x are activated by setting FAENx=1 in the CAN_FACFG register.
- Configure 0~27 filter banks by writing to the CAN_FiFBx register (i=0…27; x=1,2).
- Complete the CAN filter configuration by setting FCS=0 in the CAN_FCTRL register.

20.6.5 Message transmission

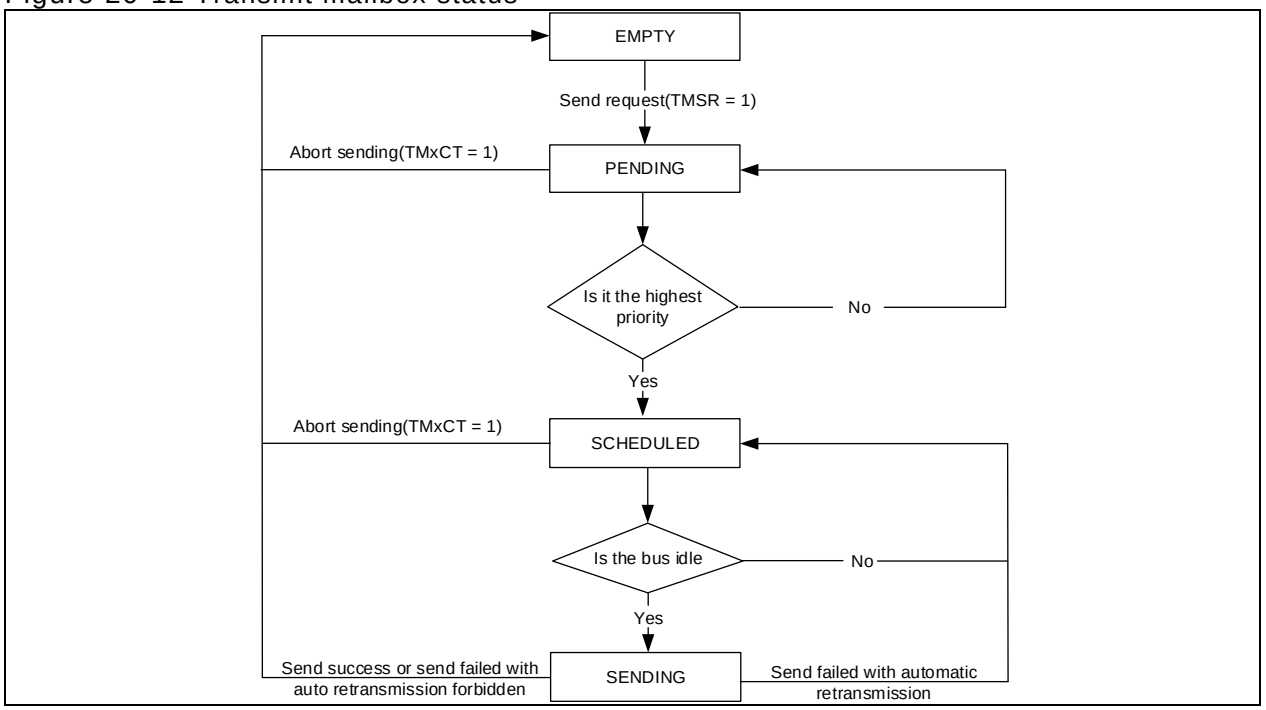
Register configuration

To transmit a message, it is necessary to select one transmit mailbox and configure it through the CAN_TMIx, CAN_TMCx, CAN_TMDTLx and CAN_TMDTHx registers. Once the mailbox configuration is complete, setting the TMSR bit in the CAN_TMIx register can initiate CAN transmission.

Message transmission

The mailbox enters pending state immediately after the mailbox is configured and the CAN controller receives the transmit request. At this point, the CAN controller will confirm whether the mailbox is given the highest priority or not. If yes, it will enter SCHEDULED STATE, otherwise, it will wait to get the highest priority. The mailbox in SCHEDULED state will monitor the CAN bus state so that the messages in SCHEDULED mailbox can be transmitted as soon as the CAN bus becomes idle. The mailbox will enter EMPTY state at the end of the message transmission.

Figure 20-12 Transmit mailbox status



Transmit priority configuration

When two or more transmit boxes are in PENDING state, their transmit priority must be given.

By identifier: When MMSSR=0 in the CAN_MCTRL register, the transmit order is defined by the identifier of the message in the mailbox. The message with lower identifier value has the highest priority. If the identifier values are the same, the message with lower mailbox number will be transmitted first.

By transmit request order: When MMSSR=1 in the CAN_MCTRL register, the transmit priority is given by the transmit request order of mailboxes.

Transmit status and error status

The TMxTCF, TMxTSF, TMxALF, TMxTEF and TMxEF bits in the CAN_TSTS register are used to indicate transmit status and error status.

TMxTCF bit: Transmission complete flag, indicating that the data transmission is complete when TMxTCF=1.

TMxTSF bit: Transmission success flag, indicating that the data has been transmitted successfully when TMxTSF =1.

TMxALF bit: Transmission arbitration lost flag, indicating that the data transmission arbitration is lost when TMxALF=1.

TMxTEF bit: Transmission error flag, indicating that the data transmission failed due to bus error, and an error frame is sent when TMxTEF=1.

TMxEF bit: Mailbox empty flag, indicating that the data transmission is complete and the mailbox becomes empty when TMxEF=1.

Transmit abort

The TMxCT bit is set in the CAN_TSTS register to abort the transmission of the current mailbox, detailed as follows:

When the current transmission fails or arbitration is lost, if the automatic retransmission mode is disabled, the transmit mailbox become EMPTY; if the automatic retransmission mode is enabled, the transmit mailbox becomes SCHEDULED, the mailbox transmission then is aborted and becomes EMPTY.

When the current transmission is complete successfully, the mailbox becomes EMPTY.

20.6.6 Message reception

Register configuration

The CAN_RFIx, CAN_RFCx, CAN_RFDTLx and CAN_RFDTHx registers can be used by user applications to obtain valid messages.

Message reception

The CAN controller boasts two FIFO with three levels to receive messages. FIFO rule is adopted. When the message is received correctly and has passed the identifier filtering, it is regarded as a valid message and is stored in the corresponding FIFO. The number of the received messages RFxMN[1: 0] will be incremented by one whenever the receive FIFO receives a valid message. If a valid message is received when RFxMN[1: 0]=3, the controller will select either to overwrite the previous messages or discard the new incoming message through the MDRSEL bit in the CAN_MCTRL register.

In the meantime, when the user reads a frame of message and the RFxR is set in the CAN_RFx register, one FIFO mailbox is released, and RFxMN[1: 0] bit is decremented by one in the CAN_RFx register.

Receive FIFO status

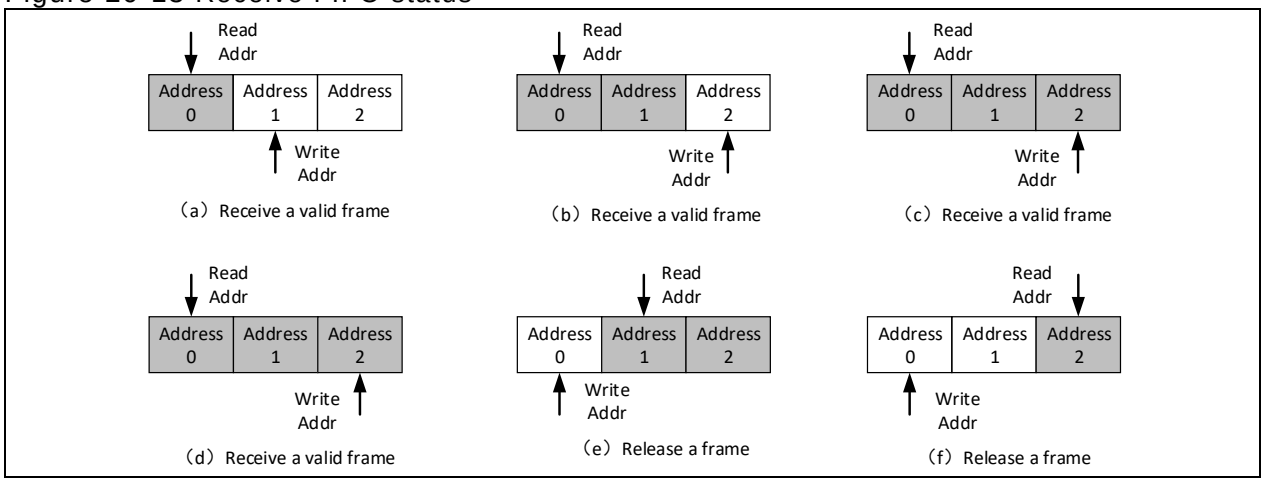
RFxMN[1: 0], RFxFF and RFxOF bits in the RFx register are used to indicate receive FIFO status.

RFxMN[1: 0]: indicates the number of valid messages stored in the FIFOx.

RFxFF: indicates that three valid messages are stored in the FIFOx (i.e. the three mailboxes are full), as shown in (c) of Figure 20-13.

RFxOF: indicates that a new valid message has been received while the FIFOx is full, as shown in (d) of Figure 20-13.

Figure 20-13 Receive FIFO status



20.6.7 Error management

The status of CAN nodes is indicated by the receive error counter (TEC) and transmit error counter (REC) bits in the CAN_ESTS register. In the meantime, the ETR[2: 0] bit in the CAN_ESTS register is used to record the last error source, and the corresponding interrupts will be generated when the CAN_INTEN register is enabled.

- Error active flag: When both TEC and REC are lower than 128, the system is in the error active state. An error active flag is set when an error is detected.
- Error passive flag: When either TEC or REC is greater than 127, the system is in the error passive state. An error passive flag is set when an error is detected.

Bus-off state: The bus-off state is entered when TEC is greater than 255. In this state, it is impossible to transmit and receive messages. The CAN resumes from bus-off state in two ways:
Option 1: When AEBOEN=0 in the CAN_MCTRL register, in communication mode, the software requests to enter Frozen mode and exit Frozen mode, and CAN will then resume from bus-off state after 128 occurrences of 11 consecutive recessive bits have been detected on the CAN RX pin.

Option 2: When AEBOEN=1 in the CAN_MCTRL register, the CAN will resume from bus-off state automatically after 128 occurrences of 11 consecutive recessive bits have been detected on the CAN RX pin

20.7 CAN registers

These peripheral registers must be accessed by words (32 bits).

Table 20-1 CAN register map and reset values

Register name	Offset	Reset value
MCTRL	000h	0x0001 0002
MSTS	004h	0x0000 0C02
TSTS	008h	0x1C00 0000
RF0	00Ch	0x0000 0000
FR1	010h	0x0000 0000
INTEN	014h	0x0000 0000
ESTS	018h	0x0000 0000
BTMG	01Ch	0x0123 0000
Reserved	020h~17Fh	xx
TMI0	180h	0xXXXX XXXX
TMC0	184h	0xXXXX XXXX
TMDTL0	188h	0xXXXX XXXX
TMDTH0	18Ch	0xXXXX XXXX
TMI1	190h	0xXXXX XXXX
TMC1	194h	0xXXXX XXXX
TMDTL1	198h	0xXXXX XXXX
TMDTH1	19Ch	0xXXXX XXXX
TMI2	1A0h	0xXXXX XXXX
TMC2	1A4h	0xXXXX XXXX
TMDTL2	1A8h	0xXXXX XXXX
TMDTH2	1ACh	0xXXXX XXXX
RFI0	1B0h	0xXXXX XXXX
RFC0	1B4h	0xXXXX XXXX
RFDTL0	1B8h	0xXXXX XXXX
RFDTH0	1BCh	0xXXXX XXXX
RFI1	1C0h	0xXXXX XXXX
RFC1	1C4h	0xXXXX XXXX
RFDTL1	1C8h	0xXXXX XXXX
RFDTH1	1CCh	0xXXXX XXXX
Reserved	1D0h~1FFh	xx
FCTRL	200h	0x2A1C 0E01
FMCFG	204h	0x0000 0000
Reserved	208h	xx
FSCFG	20Ch	0x0000 0000
Reserved	210h	xx
FRF	214h	0x0000 0000
Reserved	218h	xx

FACFG	21Ch	0x0000 0000
Reserved	220h~23Fh	xx
FB0F1	240h	0xXXXX XXXX
FB0F2	244h	0xXXXX XXXX
FB1F1	248h	0xXXXX XXXX
FB1F2	24Ch	0xXXXX XXXX
…	…	…
FB27F1	318h	0xXXXX XXXX
FB27F2	31Ch	0xXXXX XXXX

20.7.1 CAN control and status registers

20.7.1.1CAN master control register (CAN_MCTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	PTD	0x1	rw	Prohibit trans when debug 0: Transmission works during debug 1: Transmission is prohibited during debug. Receive FIFO can be still accessible normally. Note: Transmission can be disabled only when PTD and CANx_PAUSE bits in the DEBUG_CTRL register are set simultaneously.
Bit 15	SPRST	0x0	rw1s	Software partial reset 0: Normal 1: Software partial reset Note: SPRST only reset receive FIFO and MCTRL register. The CAN enters Sleep mode after reset. Then this bit is automatically cleared by hardware.
Bit 14: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7	TTCEN	0x0	rw	Time triggered communication mode enable 0: Time triggered communication mode disabled 1: Time triggered communication mode enabled
Bit 6	AEBOEN	0x0	rw	Automatic exit bus-off enable 0: Automatic exit bus-off disabled 1: Automatic exit bus-off enabled Note: When Automatic exit bus-off mode is enabled, the hardware will automatically leave bus-off mode as soon as an exit timing is detected on the CAN bus. When Automatic exit bus-off mode is disabled, the software must enter/leave the freeze mode once more, and then the bus-off state is left only when an exit timing is detected on the CAN bus.
Bit 5	AEDEN	0x0	rw	Automatic exit doze mode enable 0: Automatic exit sleep mode disabled 1: Automatic exit sleep mode enabled Note: When Automatic exit sleep mode is disabled, the sleep mode is left by software clearing the sleep request command. When Automatic exit sleep mode is enabled, the sleep mode is left without the need of software intervention as soon as a message is monitored on the CAN bus.
Bit 4	PRSFEN	0x0	rw	Prohibit retransmission enable when sending fails enable 0: Retransmission is enabled. 1: Retransmission is disabled.
Bit 3	MDRSEL	0x0	rw	Message discard rule select when overflow 0: The previous message is discarded.

				1: The new incoming message is discarded.
Bit 2	MMSSR	0x0	rw	Multiple message transmit sequence rule 0: The message with the smallest identifier is first transmitted. 1: The message with the first request order is first transmitted.
Bit 1	DZEN	0x1	rw	Doze mode enable 0: Sleep mode is disabled. 1: Sleep mode is enabled. Note: The hardware will automatically leave sleep mode when the AEDEN is set and a message is monitored on the CAN bus. After CAN reset or partial software reset, this bit is forced to be set by hardware, that is, the CAN will keep in sleep mode, by default.
Bit 0	FZEN	0x0	rw	Freeze mode enable 0: Freeze mode disabled 1: Freeze mode enabled Note: The CAN leaves Freeze mode once 11 consecutive recessive bits have been detected on the RX pin. For this reason, the software acknowledges the entry of Freeze mode after the FZC bit is cleared by hardware. The Freeze mode is entered only when the current CAN activity (transmission or reception) is completed. Thus the software acknowledges the exit of Freeze mode after the FZC bit is cleared by hardware.

20.7.1.2 CAN master status register (CAN_MSTS)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value.
Bit 11	REALRX	0x1	ro	Real time level on RX pin 0: Low 1: High
Bit 10	LSAMPRX	0x1	ro	Last sample level on RX pin 0: Low 1: High Note: This value keeps updating with the REALRX.
Bit 9	CURS	0x0	ro	Current receive status 0: No reception occurs 1: Reception is in progress Note: This bit is set by hardware when the CAN reception starts, and it is cleared by hardware at the end of reception.
Bit 8	CUSS	0x0	ro	Current transmit status 0: No transmit occurs 1: Transmit is in progress Note: This bit is set by hardware when the CAN transmission starts, and it is cleared by hardware at the end of transmission.
Bit 7: 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	EDZIF	0x0	rw1c	Enter doze mode interrupt flag 0: Sleep mode is not entered or no condition for flag set. 1: Sleep mode is entered. Note: This bit is set by hardware only when EDZIEN=1 and the CAN enters Sleep mode. When set, this bit will generate a status change interrupt. This bit is cleared by software (writing 1 to itself) or by hardware when DZC is cleared.
Bit 3	QDZIF	0x0	rw1c	Exit doze mode interrupt flag 0: Sleep mode is not left or no condition for exit. 1: Sleep mode has been left or exit condition has generated.

				Note: This bit is cleared by software (writing 1 to itself) Sleep mode is left when a SOF is detected on the bus. When QDZIEN=1, this bit will generate a status change interrupt.
Bit 2	EOIF	0x0	rw1c	Error occur interrupt flag 0: No error interrupt or no condition for error interrupt flag 1: Error interrupt is generated. Note: This bit is cleared by software (writing 1 to itself). This bit is set by hardware only when the corresponding bit is set in the CAN_ESTS register and the corresponding interrupt enable bit in the CAN_INTEN register is enabled. When set, this bit will generate a status change interrupt.
Bit 1	DZC	0x1	ro	Doze mode acknowledge 0: The CAN is not in Sleep mode. 1: CAN is in Sleep mode. Note: This bit is used to decide whether the CAN is in Sleep mode or not. This bit acknowledges the Sleep mode request generated by software. The Sleep mode can be entered only when the current CAN activity (transmission or reception) is completed. For this reason, the software acknowledges the entry of Sleep mode after this bit is set by hardware. The Sleep mode is left only once 11 consecutive recessive bits have been detect on the CAN RX pin. For this reason, the software acknowledges the exit of Sleep mode after this bit is cleared by hardware.
Bit 0	FZC	0x0	ro	Freeze mode confirm 0: The CAN is not in Freeze mode. 1: The CAN is in Freeze mode. Note: This bit is used to decide whether the CAN is in Freeze mode or not. This bit acknowledges the Freeze mode request generated by software. The Freeze mode can be entered only when the current CAN activity (transmission or reception) is completed. For this reason, the software acknowledges the entry of Freeze mode after this bit is set by hardware. The Freeze mode is left only once 11 consecutive recessive bits have been detect on the CAN RX pin. For this reason, the software acknowledges the exit of Freeze mode after this bit is cleared by hardware.

20.7.1.3 CAN transmit status register (CAN_TSTS)

Bit	Register	Reset value	Type	Description
Bit 31	TM2LPF	0x0	ro	Transmit mailbox 2 lowest priority flag 0: Mailbox 2 is not given the lowest priority. 1: Lowest priority (This indicates that more than one mailboxes are pending for transmission, the mailbox 2 has the lowest priority.)
Bit 30	TM1LPF	0x0	ro	Transmit mailbox 1 lowest priority flag 0: Mailbox 1 is not given the lowest priority. 1: Lowest priority (This indicates that more than one mailboxes are pending for transmission, the mailbox 1 has the lowest priority.)
Bit 29	TM0LPF	0x0	ro	Transmit mailbox 0 lowest priority flag 0: Mailbox 0 is not given the lowest priority. 1: Lowest priority (This indicates that more than one mailboxes are pending for transmission, the mailbox 0 has the lowest priority.)
Bit 28	TM2EF	0x1	ro	Transmit mailbox 2 empty flag This bit is set by hardware when no transmission is

				pending in the mailbox 2.
Bit 27	TM1EF	0x1	ro	Transmit mailbox 1 empty flag This bit is set by hardware when no transmission is pending in the mailbox 1.
Bit 26	TM0EF	0x1	ro	Transmit mailbox 0 empty flag This bit is set by hardware when no transmission is pending in the mailbox 0.
Bit 25: 24	TMNR	0x0	ro	Transmit Mailbox number record Note: If the transmit mailbox is free, these two bits refer to the number of the next transmit mailbox free. For example, in case of free CAN, the value of these two bit becomes 01 after a message transmit request is written. If the transmit box is full, these two bits refer to the number of the transmit mailbox with the lowest priority. For example, when there are three messages are pending for transmission, the identifiers of mailbox 0, mailbox 1 and mailbox 2 are 0x400, 0x433 and 0x411 respectively, and the value of these two bits becomes 01.
Bit 23	TM2CT	0x0	rw1c	Transmit mailbox 2 cancel transmit 0: No effect 1: Transmission is cancelled. Note: Software sets this bit to abort the transmission of mailbox 2. This bit is cleared by hardware when the transmit message in the mailbox 2 is cleared. Setting this bit has no effect if the mailbox 2 is free.
Bit 22: 20	Reserved	0x0	resd	Kept at its default value.
Bit 19	TM2TEF	0x0	rw1c	Transmit mailbox 2 transmission error flag 0: No error 1: Mailbox 2 transmission error Note: This bit is set when the mailbox 2 transmission error occurred. It is cleared by software writing 1 or by hardware at the start of the next transmission
Bit 18	TM2ALF	0x0	rw1c	Transmit mailbox 2 arbitration lost flag 0: No arbitration lost 1: Transmit mailbox 2 arbitration lost Note: This bit is set when the mailbox 2 transmission failed due to an arbitration lost. It is cleared by software writing 1 or by hardware at the start of the next transmission
Bit 17	TM2TSF	0x0	rw1c	Transmit mailbox 2 transmission success flag 0: Transmission failed 1: Transmission was successful. Note: This bit indicates whether the mailbox 2 transmission is successful or not. It is cleared by software writing 1.
Bit 16	TM2TCF	0x0	rw1c	Transmit mailbox 2 transmission completed flag 0: Transmission is in progress 1: Transmission is completed Note: This bit is set by hardware when the transmission/abort request on mailbox 2 has been completed. It is cleared by software writing 1 or by hardware when a new transmission request is received. Clearing this bit will clear the TM2TSF, TM2ALF and TM2TEF bits of mailbox 2.
Bit 15	TM1CT	0x0	rw1s	Transmit mailbox 1 cancel transmit 0: No effect 1: Mailbox 1 cancel transmit Note: This bit is set by software to abort the transmission

				request on mailbox 1. Clearing the message transmission on mailbox 1 will clear this bit. Setting by this software has no effect when the mailbox 1 is free.
Bit 14: 12	Reserved	0x0	resd	Kept at its default value.
Bit 11	TM1TEF	0x0	rw1c	Transmit mailbox 1 transmission error flag 0: No error 1: Mailbox 1 transmission error Note: This bit is set when the mailbox 1 transmission error occurred. It is cleared by software writing 1 or by hardware at the start of the next transmission
Bit 10	TM1ALF	0x0	rw1c	Transmit mailbox 1 arbitration lost flag 0: No arbitration lost 1: Transmit mailbox 1 arbitration lost Note: This bit is set when the mailbox 1 transmission failed due to an arbitration lost. It is cleared by software writing 1 or by hardware at the start of the next transmission
Bit 9	TM1TSF	0x0	rw1c	Transmit mailbox 1 transmission success flag 0: Transmission failed 1: Transmission was successful. Note: This bit indicates whether the mailbox 1 transmission is successful or not. It is cleared by software writing 1.
Bit 8	TM1TCF	0x0	rw1c	Transmit mailbox 1 transmission completed flag 0: Transmission is in progress 1: Transmission is completed Note: This bit is set by hardware when the transmission/abort request on mailbox 1 has been completed. It is cleared by software writing 1 or by hardware when a new transmission request is received. Clearing this bit will clear the TM1TSF, TM1ALF and TM1TEF bits of mailbox 1.
Bit 7	TM0CT	0x0	rw1s	Transmit mailbox 0 cancel transmit 0: No effect 1: Mailbox 0 cancel transmit Note: This bit is set by software to abort the transmission request on mailbox 0. Clearing the message transmission on mailbox 0 will clear this bit. Setting by this software has no effect when the mailbox 0 is free.
Bit 6: 4	Reserved	0x0	resd	Kept at its default value.
Bit 3	TM0TEF	0x0	rw1c	Transmit mailbox 0 transmission error flag 0: No error 1: Mailbox 0 transmission error Note: This bit is set when the mailbox 0 transmission error occurred. It is cleared by software writing 0 or by hardware at the start of the next transmission
Bit 2	TM0ALF	0x0	rw1c	Transmit mailbox 0 arbitration lost flag 0: No arbitration lost 1: Transmit mailbox 0 arbitration lost Note: This bit is set when the mailbox 0 transmission failed due to an arbitration lost. It is cleared by software writing 1 or by hardware at the start of the next transmission
Bit 1	TM0TSF	0x0	rw1c	Transmit mailbox 0 transmission success flag 0: Transmission failed 1: Transmission was successful. Note:

				This bit indicates whether the mailbox 0 transmission is successful or not. It is cleared by software writing 1.
Bit 0	TM0TCF	0x0	rw1c	Transmit mailbox 0 transmission completed flag 0: Transmission is in progress 1: Transmission is completed Note: This bit is set by hardware when the transmission/abort request on mailbox 0 has been completed. It is cleared by software writing 1 or by hardware when a new transmission request is received. Clearing this bit will clear the TM0TSF, TM0ALF and TM0TEF bits of mailbox 0.

20.7.1.4CAN receive FIFO 0 register (CAN_RF0)

Bit	Register	Reset value	Type	Description
Bit 31: 6	Reserved	0x0000000	resd	Kept at its default value.
Bit 5	RF0R	0x0	rw1s	Receive FIFO 0 release 0: No effect 1: Release FIFO Note: This bit is set by software to release FIFO 0. It is cleared by hardware when the FIFO 0 is released. Setting this bit by software has no effect when the FIFO 0 is empty. If there are more than two messages pending in the FIFO 0, the software has to release the FIFO 0 to access the second message.
Bit 4	RF0OF	0x0	rw1c	Receive FIFO 0 overflow flag 0: No overflow 1: Receive FIFO 0 overflow Note: This bit is set by hardware when a new message has been received and passed the filter while the FIFO 0 is full. It is cleared by software by writing 1.
Bit 3	RF0FF	0x0	rw1c	Receive FIFO 0 full flag 0: Receive FIFO 0 is not full 1: Receive FIFO 0 is full Note: This bit is set by hardware when three messages are pending in the FIFO 0. It is cleared by software by writing 1.
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1: 0	RF0MN	0x0	ro	Receive FIFO 0 message num Note: These two bits indicate how many messages are pending in the FIFO 0. RF0ML bit is incremented by one each time a new message has been received and passed the filter while the FIFO 0 is not full. RF0ML bit is decremented by one each time the software releases the receive FIFO 0 by writing 1 to the RF0R bit.

20.7.1.5CAN receive FIFO 1 register (CAN_RF1)

Bit	Register	Reset value	Type	Description
Bit 31: 6	Reserved	0x0000000	resd	Kept at its default value.
Bit 5	RF1R	0x0	rw1s	Receive FIFO 1 release 0: No effect 1: Release FIFO Note: This bit is set by software to release FIFO 1. It is cleared by hardware when the FIFO 1 is released. Setting this bit by software has no effect when the FIFO 1 is empty.

				If there are more than two messages pending in the FIFO 0, the software has to release the FIFO 1 to access the second message.
Bit 4	RF1OF	0x0	rw1c	Receive FIFO 1 overflow flag 0: No overflow 1: Receive FIFO 1 overflow Note: This bit is set by hardware when a new message has been received and passed the filter while the FIFO 1 is full. It is cleared by software by writing 1.
Bit 3	RF1FF	0x0	rw1c	Receive FIFO 1 full flag 0: Receive FIFO 1 is not full 1: Receive FIFO 1 is full Note: This bit is set by hardware when three messages are pending in the FIFO 1. It is cleared by software by writing 1.
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1: 0	RF1MN	0x0	ro	Receive FIFO 1 message num Note: These two bits indicate how many messages are pending in the FIFO 1. RF1ML bit is incremented by one each time a new message has been received and passed the filter while the FIFO 1 is not full. RF1ML bit is decremented by one each time the software releases the receive FIFO 1 by writing 1 to the RF1R bit.

20.7.1.6 CAN interrupt enable register (CAN_INTEN)

Bit	Register	Reset value	Type	Description
Bit 31: 18	Reserved	0x0000	resd	Kept at its default value.
Bit 17	EDZIEN	0x0	rw	Enter doze mode interrupt enable 0: Enter sleep mode interrupt disabled 1: Enter sleep mode interrupt enabled Note: EDZIF flag bit corresponds to this interrupt. An interrupt is generated when both this bit and EDZIF bit are set.
Bit 16	QDZIEN	0x0	rw	Quit doze mode interrupt enable 0: Quit sleep mode interrupt disabled 1: Quit sleep mode interrupt enabled Note: The flag bit of this interrupt is the QDZIF bit. An interrupt is generated when both this bit and QDZIF bit are set.
Bit 15	EOIEN	0x0	rw	Error occur interrupt enable 0: Error interrupt disabled 1: Error interrupt enabled Note: The flag bit of this interrupt is the EOIF bit. An interrupt is generated when both this bit and EOIF bit are set.
Bit 14: 12	Reserved	0x0	resd	Kept at its default value.
Bit 11	ETRIEN	0x0	rw	Error type record interrupt enable 0: Error type record interrupt disabled 1: Error type record interrupt enabled Note: EOIF is set only when this interrupt is enabled and the ETR[2: 0] is set by hardware.
Bit 10	BOIEN	0x0	rw	Bus-off interrupt enable 0: Bus-off interrupt disabled 1: Bus-off interrupt enabled Note: EOIF is set only when this interrupt is enabled and the BOF is set by hardware.
Bit 9	EPIEN	0x0	rw	Error passive interrupt enable 0: Error passive interrupt disabled 1: Error passive interrupt enabled Note: EOIF is set only when this interrupt is enabled and

				the EPF is set by hardware.
Bit 8	EAIEN	0x0	rw	Error active interrupt enable 0: Error warning interrupt disabled 1: Error warning interrupt enabled Note: EOIF is set only when this interrupt is enabled and the EAF is set by hardware.
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6	RF1OIEN	0x0	rw	Receive FIFO 1 overflow interrupt enable 0: Receive FIFO 1 overflow interrupt disabled 1: Receive FIFO 1 overflow interrupt enabled Note: The flag bit of this interrupt is the RF1OF bit. An interrupt is generated when this bit and RF1OF bit are set.
Bit 5	RF1FIEN	0x0	rw	Receive FIFO 1 full interrupt enable 0: Receive FIFO 1 full interrupt disabled 1: Receive FIFO 1 full interrupt enabled Note: The flag bit of this interrupt is the RF1FF bit. An interrupt is generated when this bit and RF1FF bit are set.
Bit 4	RF1MIEN	0x0	rw	FIFO 1 receive message interrupt enable 0: FIFO 1 receive message interrupt disabled 1: FIFO 1 receive message interrupt enabled Note: The flag bit of this interrupt is RF1MN bit, so an interrupt is generated when this bit and RF1MN bit are set.
Bit 3	RF0OIEN	0x0	rw	Receive FIFO 0 overflow interrupt enable 0: Receive FIFO 0 overflow interrupt disabled 1: Receive FIFO 0 overflow interrupt enabled Note: The flag bit of this interrupt is RF0OF bit, so an interrupt is generated when this bit and RF0OF bit are set.
Bit 2	RF0FIEN	0x0	rw	Receive FIFO 0 full interrupt enable 0: Receive FIFO 0 full interrupt disabled 1: Receive FIFO 0 full interrupt enabled Note: The flag bit of this interrupt is the RF0FF bit. An interrupt is generated when this bit and RF0FF bit are set
Bit 1	RF0MIEN	0x0	rw	FIFO 0 receive message interrupt enable 0: FIFO 0 receive message interrupt disabled 1: FIFO 0 receive message interrupt enabled Note: The flag bit of this interrupt is the RF0MN bit. An interrupt is generated when this bit and RF0MN bit are set
Bit 0	TCIEN	0x0	rw	Transmit mailbox empty interrupt enable 0: Transmit mailbox empty interrupt disabled 1: Transmit mailbox empty interrupt enabled Note: The flag bit of this interrupt is the TMxTCF bit. An interrupt is generated when this bit and TMxTCF bit are set

20.7.1.7 CAN error status register (CAN_ESTS)

Bit	Register	Reset value	Type	Description
Bit 31: 24	REC	0x00	ro	Receive error counter This counter is implemented in accordance with the receive part of the fault confinement mechanism of the CAN protocol.
Bit 23: 16	TEC	0x00	ro	Transmit error counter This counter is implemented in accordance with the transmit part of the fault confinement mechanism of the CAN protocol.
Bit 15: 7	Reserved	0x00	resd	Kept at its default value.
Bit 6: 4	ETR	0x0	rw	Error type record 000: No error 001: Bit stuffing error 010: Format error 011: Acknowledgement error 100: Recessive bit error 101: Dominant bit error 110: CRC error 111: Set by software

				Note: This field is used to indicate the current error type. It is set by hardware according to the error condition detected on the CAN bus. It is cleared by hardware when a message has been transmitted or received successfully. If the error code 7 is not used by hardware, this field can be set by software to monitor the code update.
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2	BOF	0x0	ro	Bus-off flag 0: Bus-off state is not entered. 1: Bus-off state is entered. Note: When the TEC is greater than 255, the bus-off state is entered, and this bit is set by hardware.
Bit 1	EPF	0x0	ro	Error passive flag 0: Error passive state is not entered 1: Error passive state is entered Note: This bit is set by hardware when the current error times has reached the Error passive state limit (Receive Error Counter or Transmit Error Counter >127)
Bit 0	EAF	0x0	ro	Error active flag 0: Error active state is not entered 1: Error active state is entered Note: This bit is set by hardware when the current error times has reached the Error active state limit (Receive Error Counter or Transmit Error Counter ≥96)

20.7.1.8 CAN bit timing register (CAN_BTMG)

Bit	Register	Reset value	Type	Description
Bit 31	LOEN	0x0	rw	Listen-Only mode 0: Listen-Only mode disabled 1: Listen-Only mode enabled
Bit 30	LBEN	0x0	rw	Loop back mode 0: Loop back mode disabled 1: Loop back mode enabled
Bit 29: 26	Reserved	0x0	resd	Kept at its default value.
Bit 25: 24	RSAW	0x1	rw	Resynchronization width tRSAW = tCAN x (RSAW[1: 0] + 1) Note: This field defines the maximum of time unit that the CAN hardware is allowed to lengthen or shorten in a bit.
Bit 23	Reserved	0x0	resd	Kept at its default value.
Bit 22: 20	BTS2	0x2	rw	Bit time segment 2 tBTS2 = tCAN x (BTS2[2: 0] + 1) Note: This field defines the number of time unit in Bit time segment 2.
Bit 19: 16	BTS1	0x3	rw	Bit time segment 1 tBTS1 = tCAN x (BTS1[3: 0] + 1) Note: This field defines the number of time unit in Bit time segment 1.
Bit 15: 12	Reserved	0x0	resd	Kept at its default value.
Bit 11: 0	BRDIV	0x000	rw	Baud rate division tq = (BRDIV[11: 0]+1) x tPCLK Note: This field defines the length of a time unit (tq).

20.7.2 CAN mailbox registers

This section describes the registers of the transmit and receive mailboxes. Refer to 20.6.5 for more information on register map.

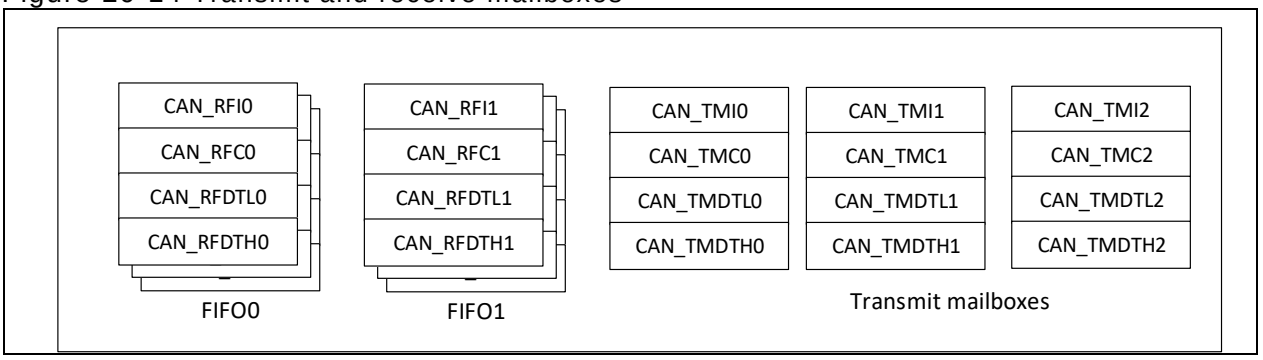
Transmit and receive mailboxes are the same except:

- RFFMN field in the CAN_RFCx register
- A receive mailbox is read only
- A transmit mailbox can be written only when empty. TMxEF=1 in the CAN_TSTS register indicates that the mailbox is empty.

There are three transmit mailboxes and two receive mailboxes. Each receive mailbox has 3-level depth of FIFO, and can only access to the first received message in the FIFO.

Each mailbox contains four registers.

Figure 20-14 Transmit and receive mailboxes



20.7.2.1 Transmit mailbox identifier register (CAN_TMIx) (x=0..2)

Note: 1. This register is write protected when its mailboxes are pending for transmission.

2.This register implements the Transmit Request control (bit 0) — reset value 0.

Bit	Register	Reset value	Type	Description
Bit 31: 21	TMSID/ TMEID	0xXXX	rw	Transmit mailbox standard identifier or extended identifier high bytes Note: This field defines the 11-bit high bytes of the standard identifier or extended identifier.
Bit 20: 3	TMEID	0xXXXXX	rw	Transmit mailbox extended identifier Note: This field defines the 18-bit low bytes of the extended identifier.
Bit 2	TMIDSEL	0xX	rw	Transmit mailbox identifier type select 0: Standard identifier 1: Extended identifier
Bit 1	TMFRSEL	0xX	rw	Transmit mailbox frame type select 0: Data frame 1: Remote frame
Bit 0	TMSR	0x0	rw	Transmit mailbox send request 0: No effect 1: Transmit request Note: This bit is cleared by hardware when the transmission has been completed (The mailbox becomes empty)

20.7.2.2 Transmit mailbox data length and time stamp register (CAN_TMCx) (x=0..2)

All the bits in the register are write protected when the mailbox is not in empty state.

Bit	Register	Reset value	Type	Description
Bit 31: 16	TMTS	0xXXXX	rw	Transmit mailbox time stamp Note: This field contains the value of the CAN timer sampled at the SOF transmission.
Bit 15: 9	Reserved	0xXX	resd	Kept at its default value
Bit 8	TMTSTEN	0xX	rw	Transmit mailbox time stamp transmit enable 0: Time stamp is not sent 1: Time stamp is sent Note: This bit is valid only when the time-triggered communication mode is enabled. In the time stamp MTS[15: 0], the MTS[7: 0] is stored in the TMDT7, and MTS[15: 8] in the TMDT6. The data length must be programmed as 8 to send time stamp.
Bit 7: 4	Reserved	0xX	resd	Kept at its default value
Bit 3: 0	TMDTBL	0xX	rw	Transmit mailbox data byte length Note: This field defines the data length of a transmit

				message. A transmit message can contain from 0 to 8 data bytes.

20.7.2.3 Transmit mailbox data low register (CAN_TMDTLx) (x=0..2)

All the bits in the register are write protected when the mailbox is not in empty state.

Bit	Register	Reset value	Type	Description
Bit 31: 24	TMDT3	0xXX	rw	Transmit mailbox data byte 3
Bit 23: 16	TMDT2	0xXX	rw	Transmit mailbox data byte 2
Bit 15: 8	TMDT1	0xXX	rw	Transmit mailbox data byte 1
Bit 7: 0	TMDT0	0xXX	rw	Transmit mailbox data byte 0

20.7.2.4 Transmit mailbox data high register (CAN_TMDTHx) (x=0..2)

All the bits in the register are write protected when the mailbox is not in empty state.

Bit	Register	Reset value	Type	Description
Bit 31: 24	TMDT7	0xXX	rw	Transmit mailbox data byte 7
Bit 23: 16	TMDT6	0xXX	rw	Transmit mailbox data byte 6 Note: This field will be replaced with MTS[15: 8] when the time-triggered communication mode is enabled and the corresponding time stamp transmit is enabled.
Bit 15: 8	TMDT5	0xXX	rw	Transmit mailbox data byte 5
Bit 7: 0	TMDT4	0xXX	rw	Transmit mailbox data byte 4

20.7.2.5 Receive FIFO mailbox identifier register (CAN_RFIx) (x=0..1)

Note: All the receive mailbox registers are read only.

Bit	Register	Reset value	Type	Description
Bit 31: 21	RFSID/RFEID	0xXXX	ro	Receive FIFO standard identifier or receive FIFO extended identifier Note: This field defines the 11-bit high bytes of the standard identifier or extended identifier.
Bit 20: 3	RFEID	0xXXXXX	ro	Receive FIFO extended identifier Note: This field defines the 18-bit low bytes of the extended identifier.
Bit 2	RFIDI	0xX	ro	Receive FIFO identifier type indication 0: Standard identifier 1: Extended identifier
Bit 1	RFFRI	0xX	Ro	Receive FIFO frame type indication 0: Data frame 1: Remote frame
Bit 0	Reserved	0x0	resd	Kept at its default value

20.7.2.6 Receive FIFO mailbox data length and time stamp register (CAN_RFCx) (x=0..1)

Note: All the receive mailbox registers are read only.

Bit	Register	Reset value	Type	Description
Bit 31: 16	RFTS	0xXXXX	ro	Receive FIFO time stamp Note: This field contains the value of the CAN timer sampled at the start of a receive frame.
Bit 15: 8	RFFMN	0xXX	ro	Receive FIFO filter match number Note: This field contains the filter number that a message has passed through.
Bit 7: 4	Reserved	0xX	resd	Kept at its default value
Bit 3: 0	RFDTL	0xX	ro	Receive FIFO data length Note: This field defines the data length of a receive message. A transmit message can contain from 0 to 8 data bytes. For a remote frame, its data length RFDTl is fixed 0.

20.7.2.7 Receive FIFO mailbox data low register (CAN_RFDTLx) (x=0..1)

Note: All the receive mailbox registers are read only.

Bit	Register	Reset value	Type	Description
Bit 31: 24	RFDT3	0xXX	ro	Receive FIFO data byte 3
Bit 23: 16	RFDT2	0xXX	ro	Receive FIFO data byte 2
Bit 15: 8	RFDT1	0xXX	ro	Receive FIFO data byte 1
Bit 7: 0	RFDT0	0xXX	ro	Receive FIFO data byte 0

20.7.2.8 Receive FIFO mailbox data high register (CAN_RFDTHx) (x=0..1)

Note: All the receive mailbox registers are read only.

Bit	Register	Reset value	Type	Description
Bit 31: 24	RFDT7	0xXX	ro	Receive FIFO data byte 7
Bit 23: 16	RFDT6	0xXX	ro	Receive FIFO data byte 6
Bit 15: 8	RFDT5	0xXX	ro	Receive FIFO data byte 5
Bit 7: 0	RFDT4	0xXX	ro	Receive FIFO data byte 4

20.7.3 CAN filter registers

20.7.3.1 CAN filter control register (CAN_FCTRL)

Note: All the non-reserved bits of this register are controlled by software completely.

Bit	Register	Reset value	Type	Description
Bit 31: 1	Reserved	0x160E0700	resd	Kept at its default value
Bit 0	FCS	0x1	rw	Filter configuration switch 0: Disabled (Filter bank is active) 1: Enabled (Filter bank is in configuration mode) Note: The initialization of the filter bank can be configured only when it is in configuration mode.

20.7.3.2 CAN filter mode configuration register (CAN_FMCFG)

Note: This register can be written only when FCS=1 in the CAN_FCTRL register (The filter is in configuration mode)

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x00000	resd	Kept at its default value
Bit 27: 0	FMSELx	0x0000	rw	Filter mode select Each bit corresponds to a filter bank. 0: Identifier mask mode 1: Identifier list mode

20.7.3.3 CAN filter bit width configuration register (CAN_ FBWCFG)

Note: This register can be written only when FCS=1 in the CAN_FCTRL register (The filter is in configuration mode)

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x00000	resd	Kept at its default value
Bit 27: 0	FBWSELx	0x0000	rw	Filter bit width select Each bit corresponds to a filter bank. 0: Dual 16-bit 1: Single 32-bit

20.7.3.4 CAN filter FIFO association register (CAN_ FRF)

Note: This register can be written only when FCS=1 in the CAN_FCTRL register (The filter is in configuration mode)

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x00000	resd	Kept at its default value
Bit 27: 0	FRFSELx	0x0000	rw	Filter relation FIFO select Each bit corresponds to a filter bank. 0: Associated with FIFO0 1: Associated with FIFO1

20.7.3.5 CAN filter activation control register (CAN_ FACFG)

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x00000	resd	Kept at its default value
Bit 27: 0	FAENx	0x0000	rw	Filter active enable Each bit corresponds to a filter bank. 0: Disabled 1: Enabled

20.7.3.6 CAN filter bank i filter bit register (CAN_ FiFBx) (i=0..13; x=1..2)

Note: There are 14 filter banks (i=0..13). Each filter bank consists of two 32-bit registers, CAN_FiFB[2:1]. This register can be modified only when the FAENx bit of the CAN_FACFG register is cleared or the FCS bit of the CAN_FCTRL register is set.

Bit	Register	Reset value	Type	Description
Bit 31: 0	FFDB	0xXXXX XXXX	rw	Filters filter data bit Identifier list mode: The configuration value of the register matches with the level of the corresponding bit of the data received on the bus (If it is a standard frame, the value of the corresponding bit of the extended frame is neglected.) Identifier mark mode: Only the bit with its register configuration value 1 can match with the level of the corresponding bit of the data received on the bus. It don't care when the register value is 0.

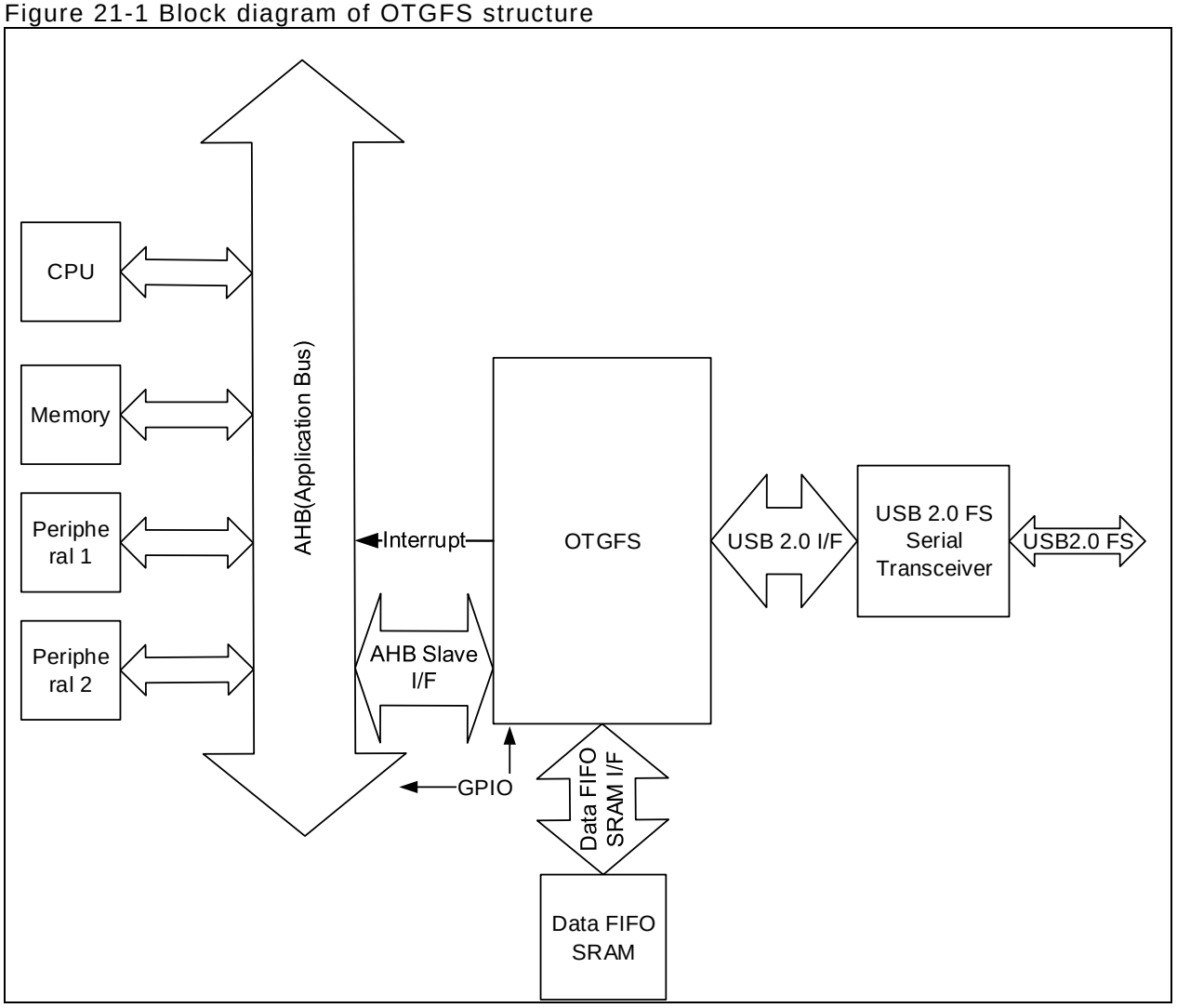
21 Universal serial bus full-seed device interface (OTGFS)

As a full-speed dual-role device, the OTGFS is fully compliant with the Universal Serial Bus Specification Revision2.0.

21.1 USBFS structure

Figure 21-1 shows the block diagram of the OTGFS structure. The OTGFS module is connected to the AHB and has a dedicated SRAM of 1280 bytes.

Figure 21-1 Block diagram of OTGFS structure



21.2 OTGFS functional description

Two independent OTGFS modules are embedded in the device. The OTGFS module consists of an OTGFS controller, PHY and 1280-byte SRAM.

The OTGFS supports control transfer, bulk transfer, interrupt transfer and synchronous transfer.

The OTGFS is a USB full-speed dual role device controller. The status of the ID line determines whether the OTGFS acts as a host or device. When the ID line is floating, the OTGFS is used as a device. It is used as a host while the ID line is grounded. The internal 1.5KΩ pull-up resistor and 1.5KΩ pull-down resistor are available in the OTG PHY for the sake of dual role device.

In device mode, the OTGFS supports one bidirectional control endpoint, 7 IN endpoints, and 7 OUT endpoints; in hose mode, the OTGFS supports 16 host channels.

The OTGFS supports SOF and OE pulse features: a SOF pulse generates at a SOF packet, the pulse can output to pins and the timer 2; an OE pulse generates when the OTGFS outputs data, the pulse can output to pins.

Suspend mode is supported. The OTGFS goes into power-saving mode after Suspend mode is entered.

As a device, a unified FIFO buffer is allocated for all OUT endpoints, and a separate FIFO buffer is provided to each of IN endpoints. As a host, a unified receive FIFO is allocated for all receive channels, a unified transmit FIFO for all non-periodic transmit channels, and a unified transmit FIFO for all periodic transmit channels.

If the STOPPCLK is set in the OTGFS_PCGCCTL register, the OTGFS enters Suspend mode when the bus signal is not received for 3ms. The LP_MODE bit in the OTGFS_GCCFG register can be used to disable PHY receive in order to reduce power consumption.

21.3 OTGFS clock and pin configuration

21.3.1 OTGFS clock configuration

The OTGFS interface has two clocks: USB control clock and APB bus clock. The USB full-speed device bus speed standard is 12Mb/s±0.25%, so it is necessary to supply 48MHz±0.25% for the OTGFS to perform USB bus sampling.

USBFS 48M clock has two sources:

- HICK 48M

  When the HICK 48M clock is used as a USB control clock, it is recommended to enable ACC feature.
- Divided by PLL

  The PLL output frequency must ensure that the USBDIV (see CRM_CFG register) can be divided to 48MHz.

Note: The APB clock frequency must be greater than 30MHz when OTGFS is enabled.

21.3.2 OTGFS pin configuration

When the USB module is enabled in the CRM, PA11 and PA12 can be multiplexed as DP/DM; PA8 can be multiplexed as SOF output and configured as a push-pull multiplexed output feature.

The OTGFS input/output pins are multiplexed with GPIOs. The GPIOs are used as OTGFS in one of the following conditions:

Table 21-1 OTGFS input/output pins

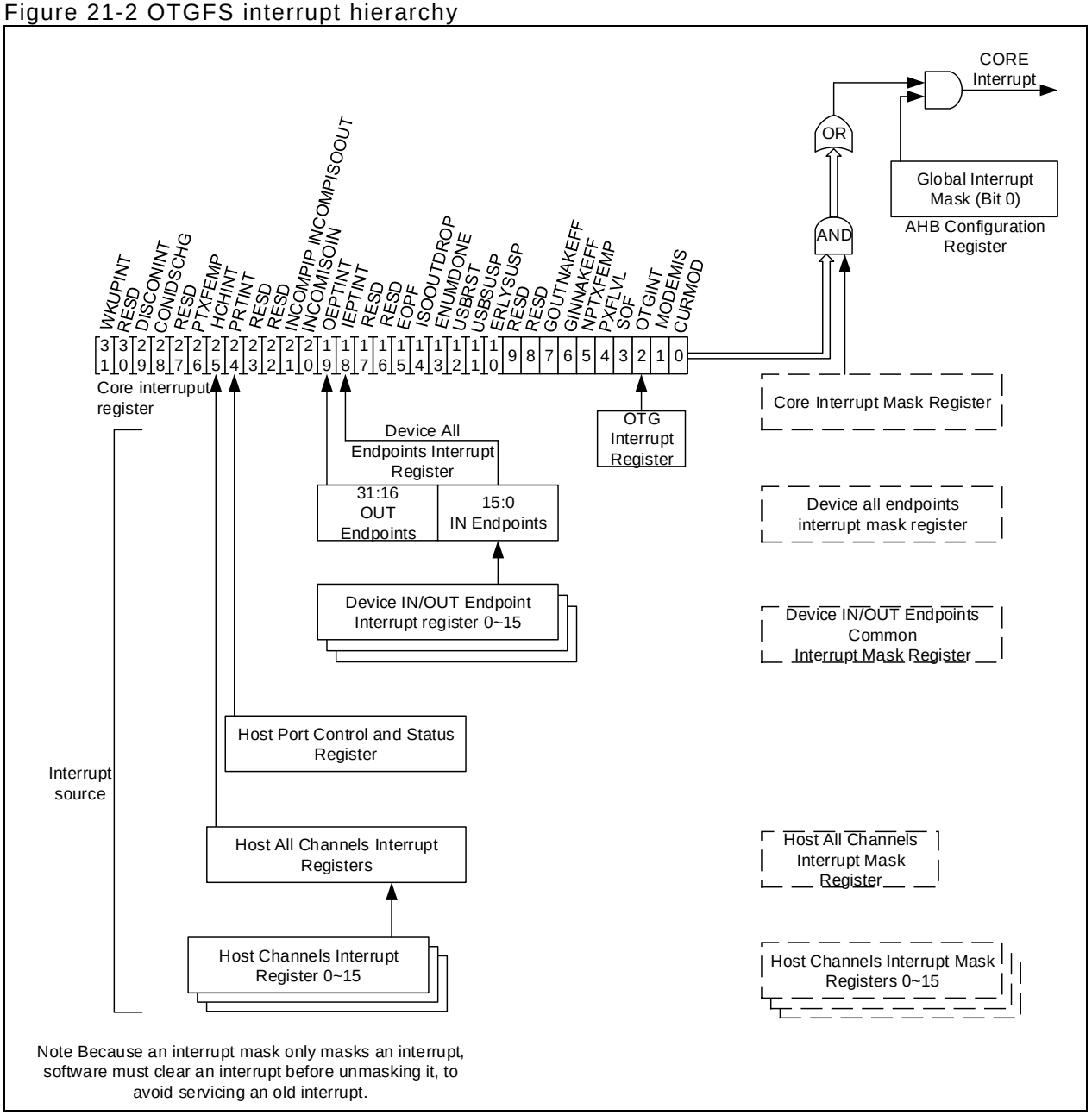
Pin	GPIO	Description
OTGFS1_SOF	PA8	Enable OTG1 in CRM, and configure PA8 multiplexed function register as 0xA
OTGFS1_VBUS	PA9	Configure PA9 as multiplexed function mode and PA9 multiplexed function register as 0xA
OTGFS1_ID	PA10	Enable OTG1 in CRM, configure PA10 as multiplexed function mode and PA10 multiplexed function register as 0xA configure
OTGFS1_D-	PA11	Enable OTG1 in CRM, configure PA11 multiplexed function register as 0xA, and set PWRDOWN=1
OTGFS1_D+	PA12	Enable OTG1 in CRM, configure PA12 multiplexed function register as 0xA, and set PWRDOWN=1
OTGFS1_OE	PA13	Enable OTG1 in CRM, and configure PA13 multiplexed function register as 0xA
OTGFS2_SOF	PA4	Enable OTG2 in CRM, and configure PA4 multiplexed function register as 0xC
OTGFS2_VBUS	PB13	Configure PB13 as multiplexed function mode and PB13 multiplexed function register as 0xC
OTGFS2_ID	PB12	Enable OTG2 in CRM, configure PB12 as multiplexed function mode and PB12 multiplexed function register as 0xC
OTGFS2_D-	PB14	Enable OTG2 in CRM, configure PB14 multiplexed function register as 0xC, and set PWRDOWN=1
OTGFS2_D+	PB15	Enable OTG2 in CRM, configure PB15 multiplexed function register

		as 0xC, and set PWRDOWN=1
OTGFS2_OE	PC9	Enable OTG2 in CRM, and configure PC9 multiplexed function register as 0xB

21.4 OTGFS interrupts

Figure 21-2 shows the OTGFS interrupt hierarchy. Refer to the OTGFS interrupt register (OTGFS_GINTSTS) and OTGFS interrupt mask register (OTGFS_GINTMSK).

Figure 21-2 OTGFS interrupt hierarchy



21.5 OTGFS functional description

21.5.1 OTGFS initialization

If the cable is connected during power-on, the current operation mode bit (CURMOD bit) in the controller interrupt register indicates the mode. The OTGFS controller enters host mode when A-type plug is connected or device mode when a B-type plug is connected.

This section explains the initialization of the OTGFS controller after power-on. The application must follow the initialization sequence, however in host or device mode. All controller global registers are initialized according to the controller configuration.

1. Program the following fields in the global AHB configuration register:
- Global interrupt mask bit = 0x1
- Non-periodic transmit FIFO empty level
- Periodic transmit FIFO empty level

2. Program the following fields in the global AHB configuration register:
- OTGFS_GINTMSK.RXFLVLMSK = 0x0

3. Program the following fields in the OTGFS_GUSBCFG register:
- Full-speed timeout standard bit
- USB turnaround time bit

4. The software must unmask the following bits in the OTGFS_GINTMSK register:
- OTG interrupt mask
- Mode mismatch interrupt mask

5. The software can read the CURMOD bit in the OTGFS_GINTSTS register to determine whether the OTGFS controller is operating in host or device mode.

21.5.2 OTGFS FIFO configuration

21.5.2.1 Device mode

A dynamic FIFO allocation is required during power-on or USB reset. In device mode, the application must meet the following conditions before modifying FIFO SRAM allocation.
- OTGFS_DIEPCTLx/ OTGFS_DOEPCTLx.EPENA = 0x0
- OTGFS_DIEPCTLx/ OTGFS_DOEPCTLx.NAKSTS = 0x1

The TXFNUM bit in the OTGFS_GRSTCTL register is used to refresh the controller transmit FIFO. Refer to Section Refresh controller transmit FIFO for more information.

Attention should be paid to the following information during FIFO SRAM allocation:

(1) Receive FIFO SRAM allocation
- SRAM for SETUP Packets: 13 WORDs must be reserved in the receive FIFO to receive one SETUP Packet on control endpoint. The controller does not use these locations, which are reserved for SETUP packets.
- One WORD is to be reserved for global OUT NAK
- Status information is written to the FIFO along with each received packet. Therefore, a minimum space of (largest packet size/4) + 1 must be allocated to receive data packets. In most cases, two (largest packet size/4) + 1 spaces are recommended so that the USB can receive the subsequent packet while the previous packet is being transferred to the AHB. If there is a longer latency on AHB, sufficient spaces must be reserved to receive multiple packets in order to prevent synchronous data packet loss.
- Transfer complete status information, along with the last packet for each endpoint, is also pushed to the FIFO
- One location must be reserved for the disable status bit of each endpoint
- Typically, two WORDs for each OUT endpoint are recommended.

(2) Transmit FIFO SRAM allocation

The minimum SRAM space required for each IN endpoint transmit FIFO is the maximum data packet size for that particular IN endpoint. The more the space allocated to the transmit IN endpoint FIFO, the better the USB performance, and this helps to avoid latency on the AHB line.

Table 21-2 OTGFS transmit FIFO SRAM allocation

FIFO name	SRAM size
Receive FIFO	rx_fifo_size, including setup packets, OUT endpoint control information and OUT data packets.
Transmit FIFO 0	tx_fifo_size[0]
Transmit FIFO 1	tx_fifo_size[1]
Transmit FIFO 2	tx_fifo_size[2]
……	……
Transmit FIFO i	tx_fifo_size[i]

Configure the following registers according to the above mentioned:

1. OTGFS receive FIFO size register (OTGFS_GRXFSIZ)

- OTGFS_GRXFSIZ.RXFDEP = rx_fifo_size

2. Endpoint 0 TX FIFO size register (OTGFS_DIEPTXF0)

- OTGFS_DIEPTXF0.INEPT0TXDEP = tx_fifo_size[0]
- OTGFS_DIEPTXF0.INEPT0TXSTADDR = rx_fifo_size

3. Device IN endpoint transmit FIFO#1 size register (OTGFS_DIEPTXF1)

- OTGFS_DIEPTXF1.INEPTXFSTADDR = OTGFS_DIEPTXF0.INEPT0TXSTADDR + tx_fifo_size[0]

4. Device IN endpoint transmit FIFO#2 size register (OTGFS_DIEPTXF2)

- OTGFS_DIEPTXF2.INEPTXFSTADDR = OTGFS_DIEPTXF1.INEPTXFSTADDR + tx_fifo_size[1]

5. Device IN endpoint transmit FIFO#i size register (OTGFS_DIEPTXFi)

- OTGFS_DIEPTXFi.INEPTXFSTADDR = OTGFS_DIEPTXFi-1.INEPTXFSTADDR +tx_fifo_size[i-1]

6. After SRAM allocation, refresh transmit FIFO and receive FIFO to ensure normal FIFO running.

- OTGFS_GRSTCTL.TXFNUM = 0x10
- OTGFS_GRSTCTL.TXFFLSH = 0x1
- OTGFS_GRSTCTL.RXFFLSH = 0x1

The application cannot perform other operations on the controller until the TXFFLSH and RXFFLSH bits are cleared.

21.5.2.2 Host mode

In host mode, the application must confirm the following status before changing FIFO SRAM allocation:

- All channels have been disabled
- All FIFOs are empty

After FIFO SRAM allocation is complete, the application must refresh all FIFOs in the controller through the TXFNUM bit in the OTGFS_GRSTCTL register.

After allocation, the FIFO pointers must be reset by refreshing operation to ensure normal FIFO running. Refer to Section Refresh controller transmit FIFO for more information.

(1) Receive FIFO SRAM allocation

Status information is written to the FIFO along with each received packet. Therefore, a minimum space of (largest packet size/4) + 2 must be allocated to receive data packets. If more synchronous endpoints are enabled, then at least two (largest packet size/4) + 2 spaces must be allocated to receive back-to-back packets. In most cases, two (largest packet size/4) + 2 spaces are recommended so that the USB can receive the subsequent packet while the previous packet is being transferred to the AHB. If there is a longer latency on AHB, sufficient spaces must be reserved to receive multiple packets in order to prevent synchronous data packet loss.

Transfer complete status information and channel abort information, along with the last packet in the host channel is also pushed to the FIFO. Thus, two WORDs must be allocated for this.

(2) Transmit FIFO SRAM allocation

The minimum SRAM space required for the host non-periodic transmit FIFO is the largest packet size of all non-periodic OUT channels. The more the space allocated to the non-periodic FIFO, the better the USB performance, and this helps to avoid latency on the AHB line. Typically, two largest packet sizes of space is recommended so that the AHB can get the next data packet while the current packet is being transferred to the USB. If there is a longer latency on AHB, sufficient spaces must be reserved to receive multiple packets in order to prevent synchronous data packet loss.

(3) Internal storage space allocation

Table 21-3 OTGFS internal storage space allocation

FIFO Name	Data SRAM Size
Receive FIFO	rx_fifo_size
Non-periodic transmit FIFO	tx_fifo_size[0]
Periodic transmit FIFO	tx_fifo_size[1]

Configure the following registers according to the above mentioned:

1. OTGFS receive FIFO size register (OTGFS_GRXFSIZ)

- OTGFS_GRXFSIZ.RXFDEP = rx_fifo_size

2. OTGFS Non-periodic TX FIFO size register (OTGFS_GNPTXFSIZ)

- OTGFS_GNPTXFSIZ.NPTXFDEP = tx_fifo_size[0]
- OTGFS_GNPTXFSIZ. NPTXFSTADDR = rx_fifo_size

3. OTGFS host periodic transmit FIFO size register (OTGFS_HPTXFSIZ)

- OTGFS_HPTXFSIZ.PTXFSIZE = tx_fifo_size[1]
- OTGFS_HPTXFSIZ.PTXFSTADDR = OTGFS_GNPTXFSIZ.NPTXFSTADDR + tx_fifo_size[0]

4. After SRAM allocation, refresh transmit FIFO and receive FIFO to ensure normal FIFO running.

- OTGFS_GRSTCTL.TXFNUM = 0x10
- OTGFS_GRSTCTL.TXFFLSH = 0x1
- OTGFS_GRSTCTL.RXFFLSH = 0x1
- The application cannot perform other operations on the controller until the TXFFLSH and RXFFLSH bits are cleared.

21.5.2.3 Refresh controller transmit FIFO

The application refreshes all transmit FIFOs through the TXFFLSH bit in the OTGFS_GRSTCTL register:

- Check whether GINNAKEFF=0 or not in the OTGFS_GINTSTS register. If this bit has been cleared, write 0x1 to the OTGFS_DCTL.SGNPINNAK register. When the NACK valid interrupt is set, it means that the controller does not read FIFO.
- Wait until GINNAKEFF = 0x1 in the OTGFS_GINTSTS register, indicating that the NAK configuration has taken effect for all IN endpoints.
- Poll the OTGFS_GRSTCTL register and wait until AHBIDLE=1. AHBIDLE = H indicates that the controller does not write the FIFO.
- Confirm whether TXFFLSH = 0x0 or not in the OTGFS_GRSTCTL register. If TXFFLSH is cleared, write the transmit FIFO number to be refreshed into the OTGFS_GRSTCTL.TXFNUM register.
- Set TXFFLSH = 0x1 in the OTGFS_GRSTCTL register, and wait until it is cleared.
- Set the CGNPINNAK bit in the OTGFS_DCTL register.

21.5.3 OTGFS host mode

21.5.3.1 Host initialization

The following steps must be respected to initialize the controller:

1. Unmask interrupt through the PRTINTMSK bit in the OTGFS_GINTMSK register
2. Program the OTGFS_HCFG register
3. Set PRTPWR = 0x1 in the OTGFS_HPRT register to drive VBUS supply on the USB
4. Wait until that the PRTCONDETbit is set in the OTGFS_HPRT0 register, indicating that the device is connected to the port
5. Set PRTRST = 0x1 in the OTGFS_HPRT register to issue a reset operation
6. Wait for at least 10 ms to ensure the completion of the reset
7. Set PRTRST = 0x0 in the OTGFS_HPRT register
8. Wait for the interrupt (PRTENCHNG bit in the OTGFS_HPRT register)
9. Read the PRTSPD bit in the OTGFS_HPRT register to get the enumeration speed

10. Configure the HFIR register according to the selected PHY clock value
11. Select the size of the receive FIFO by setting the OTGFS_GRXFSIZ register
12. Select the start address and size of the non-periodic transmit FIFO by setting the OTGFS_GNPTXFSIZ register
13. Select the start address and size of the periodic transmit FIFO by setting the OTGFS_HPTXFSIZ register

To communicate with the device, the application must enable and initialize at least one channel according to OTGFS channel initialization requirements.

21.5.3.2 OTGFS channel initialization

To communicate with the device, the application must enable and initialize at least one channel according to the following steps:

1. Unmask the following interrupts by setting the OTGFS_GINTMSK register:

- Non-periodic transmit FIFO empty for OUT transfers
- Non-periodic transmit FIFO half empty for OUT transfers

2. Unmask the interrupts of the selected channels by setting the OTGFS_HAINTMSK register
3. Unmask the transfer-related interrupts in the host channel interrupt register by setting the OTGFS_HCINTMSKx register
4. Configure the total transfer size (in bytes), and the expected number of the packets (including short packets) for the OTGFS_HCTSIZx register of the selected channel. The application must configure the PID bit according to the initial data PID (it is the PID on the first OUT transfer, or to be received from the first IN transfer)
5. Configure the transfer size to ensure that the transfer size of the channel is a multiple of the largest packet size
6. Configure the OTGFS_HCCHARx register of the selected channel according to the device endpoint characteristics such as type, speed and direction (the channel cannot be enabled by setting the enable bit until the application is ready for packet transfer or reception)

21.5.3.3 Halting a channel

The application can disable a channel by writing 0x1 to the CHDIS and CHENA bits in the OTGFS_HCCHARx register. This enables the host to refresh the summited requests (if any) and generates a channel halted interrupt. The application cannot re-allocate channels for other transactions until an interrupt is generated in the OTGFS_HCINTx register (CHHLTD bit). Those transactions that have already been started on the USB line are not interrupted by the host.

Before disabling a channel, the application must ensure that there is at least one free space available in the non-periodic request queue (when disabling a non-period channel) or the periodic request queue (when disabling a periodic channel). The application can refresh the submitted requests when the request queue is full (before disabling the channel) by setting CHDIS=0x1, and CHENA=0 in the OTGFS_HCCHARx register.

When there is a transaction input in the request queue, the controller will trigger a RXFLVL interrupt. The application must generate a channel halted interrupt through the OTGFS_GRXSTSP register.

The application is expected to abort a channel on any of the following conditions:

- When an interrupt (XFERC bit) is received in the OTGFS_HCINTx register during a non-periodic IN transfer
- When an STALL , XACTERR , BBLERR or DTGLERR interrupt in the OTGFS_HCINTx register is received for an IN or OUT channel
- When a DISCONINT (device disconnected) interrupt event is received in the OTGFS_GINTSTS register, the application must check the PRTCONSTS bit in the OTGFS_HPRT register. This is because when the device is disconnected with the host, the PRTCONSTS bit will be reset in the OTGFS_HPRT register. The application must initiate a software reset to ensure that all channels have been cleared. Once the device is reconnected, the host must start a USB reset.
- When the application needs to abort a transfer before normal completion

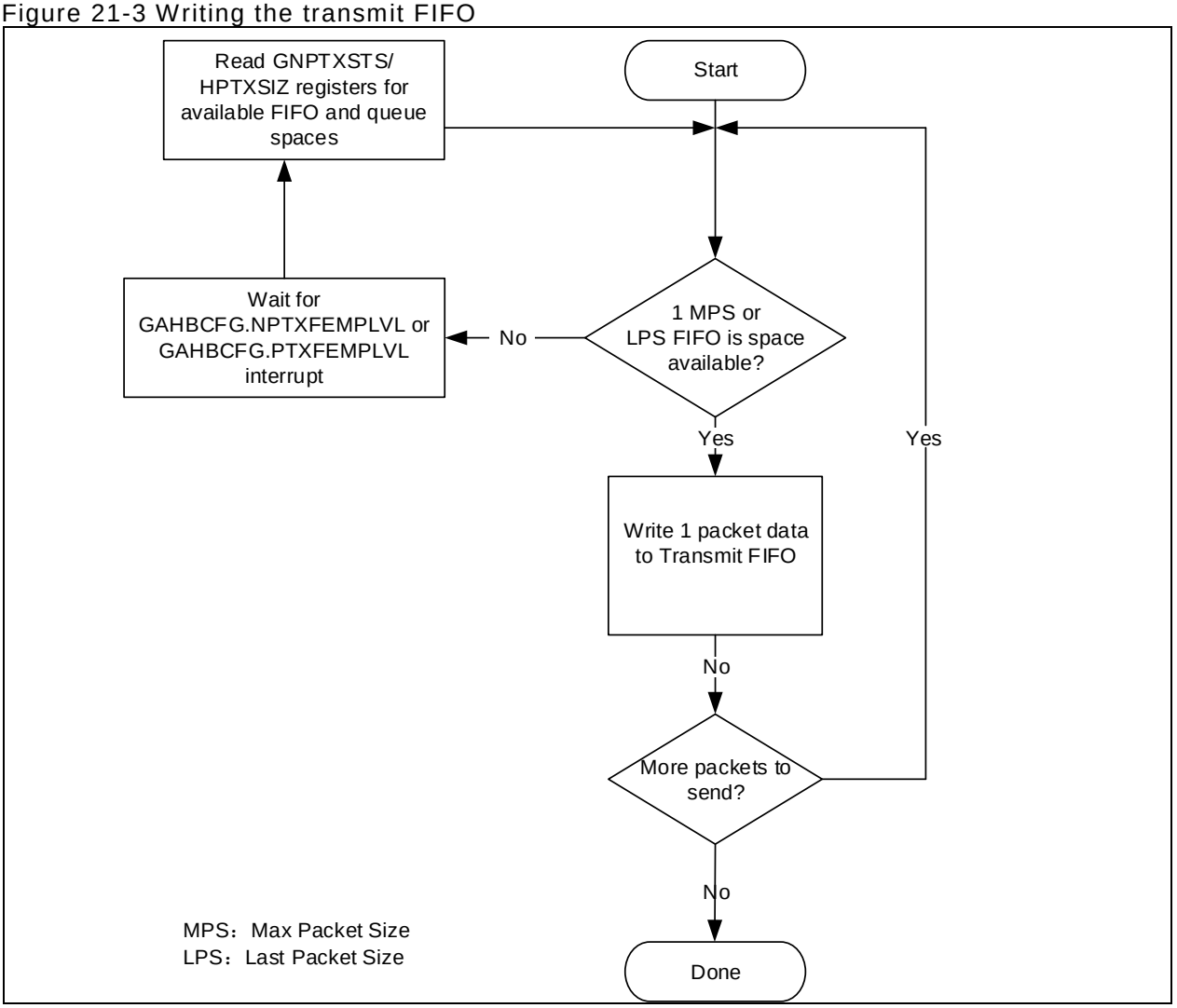
21.5.3.4 Queue depth

Up to 8 interrupt and synchronous transfer requests are supported in the periodic hardware transfer request queue; while up to 8 control and bulk transfer requests are allowed in the non-periodic hardware transfer request queue.

- Writing the transmit FIFO

Figure 21-3 shows the flow chart of writing the transmit FIFO. The OTGFS host automatically writes a request (OUT request) to the periodic/non-periodic request queue when writing the last one WORD packet. The application must ensure that at least one free space is available in the periodic/non-periodic request queue before starting to write to the transmit FIFO. The application must always write to the transmit FIFO in WORDs. If the packet size is not aligned with WORD, the application must use padding. The OTGFS host determines the actual packet size according to the programmed maximum packet size and transfer size.

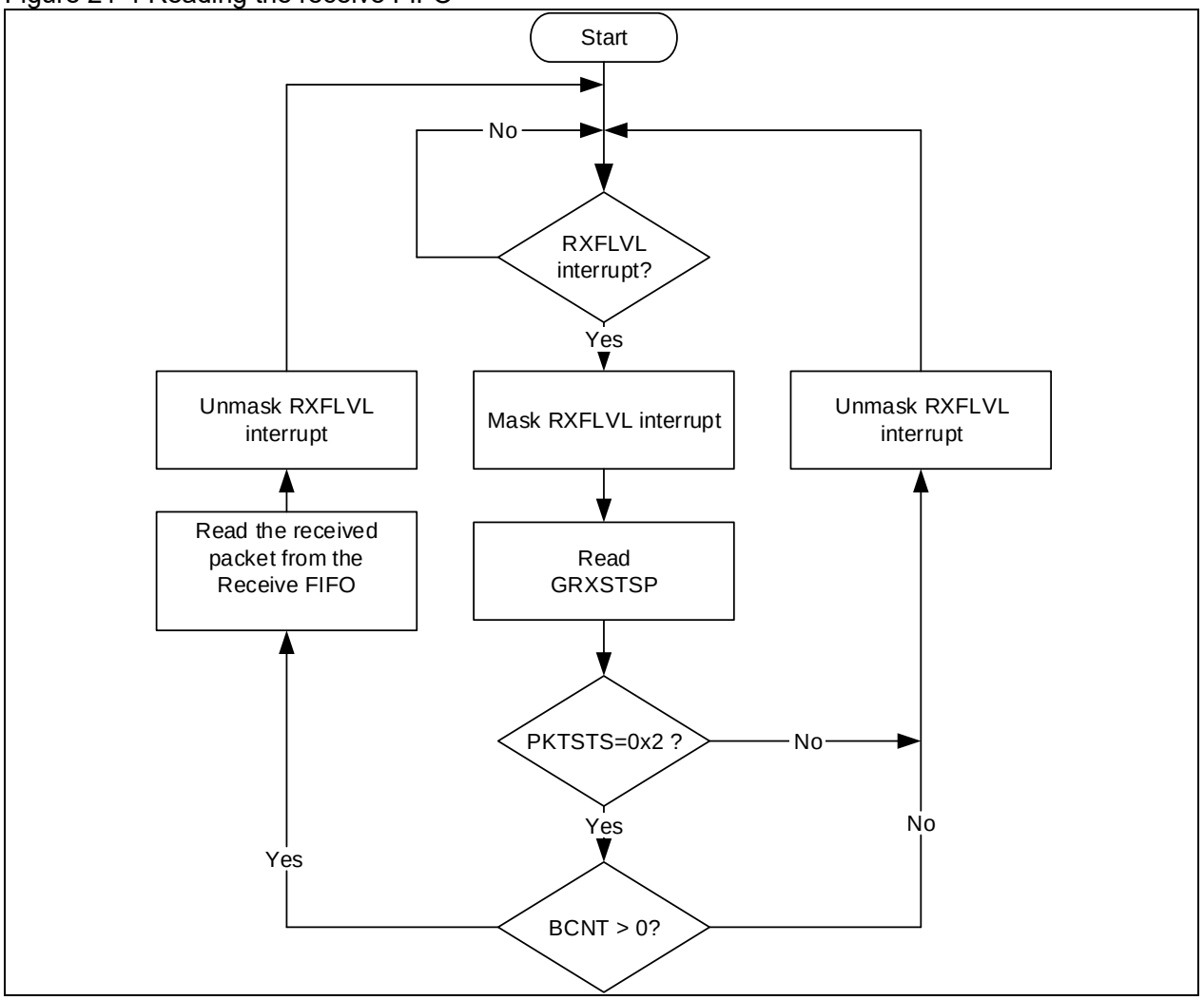
Figure 21-3 Writing the transmit FIFO



- Reading the receive FIFO

Figure 21-4 shows the flow chart of reading the receive FIFO. The application must ignore all packet statuses other than IN data packet (0x0010)

Figure 21-4 Reading the receive FIFO



21.5.3.5 Special cases

(1) Handling babble conditions

The OTGFS controller handles two cases of babble: packet babble and port babble. Packet babble occurs if the device sends more than the largest packet size for the channel. Port babble occurs if the controller continues to receive data from the device at EOF2 (the end of frame 2, which is very close to SOF)

When the OTGFS controller detects a packet babble, it stops writing data to the receiver buffer and waits for the completion of packet. When it detects the end of packet, the OTGFS flushes the data already written in the receiver buffer and generates a babble interrupt.

When the OTGFS controller detects a port babble, it flushes the receive FIFO and disables the port. Then the controller generates a Port disable interrupt. Once receiving the interrupt, the application must determine that this is not caused by an overcurrent condition (another cause of the port disable interrupt) by checking the PRTOVRCACT bit in the OTGFS_HPRT register, then perform a software reset. The controller does not send any more tokens if a port babble signal is detected.

(2) Handling device disconnected conditions

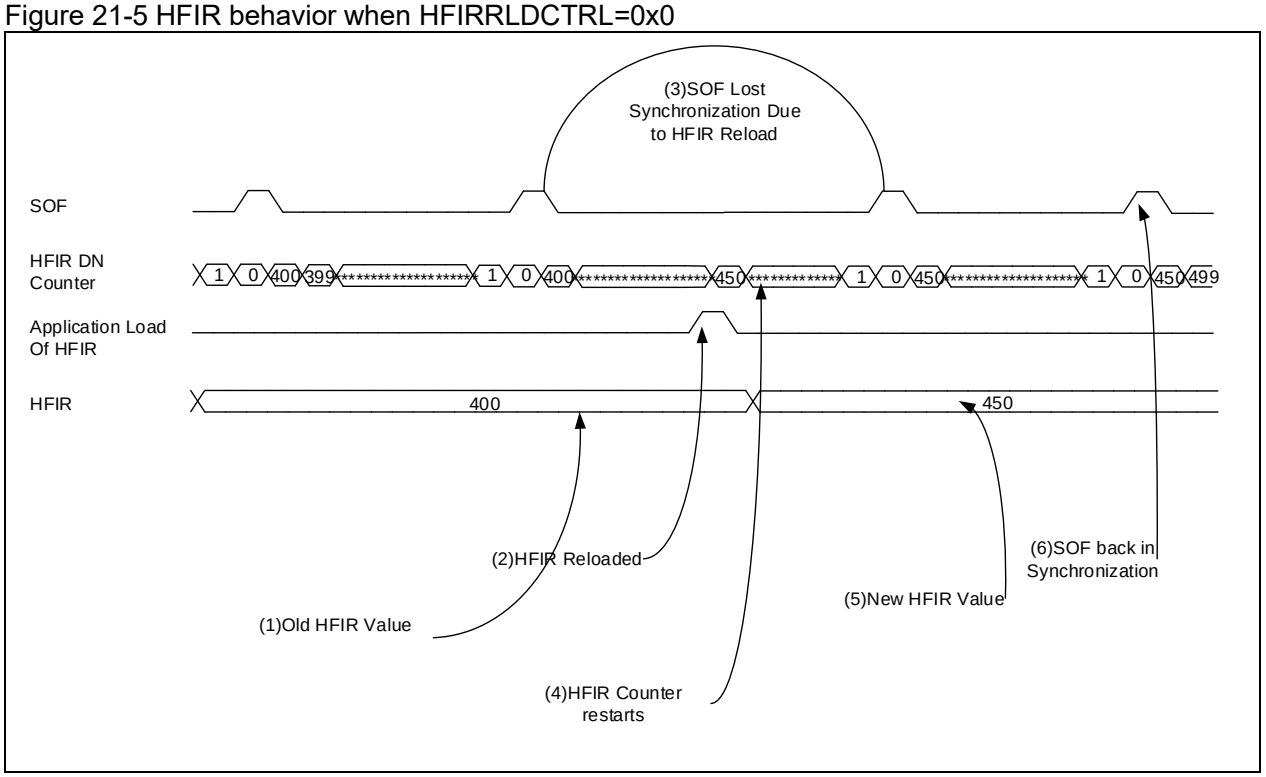
If the device is suddenly disconnected, an interrupt is generated on a disconnect event (DISCONINT bit in the OTGFS_GINTSTS register). Upon receiving this interrupt, the application must start a software reset through the CSFTRST in the OTGFS_GRSTCTL register.

21.5.3.6 Host HFIR feature

The host frame interval register (HFIR) defines the interval between two consecutive SOFs (full-speed) or Keep-Alive tokens. This field contains the number of PHY clock for the required frame interval. This is mainly used to adjust the SOF duration based on PHY clock frequencies.

Figure 21-5 shows the HFIR behavior when the HFIRRLDCTRL is set to 0x0 in the OTGFS_HFIR register.

Figure 21-5 HFIR behavior when HFIRRLDCTRL=0x0

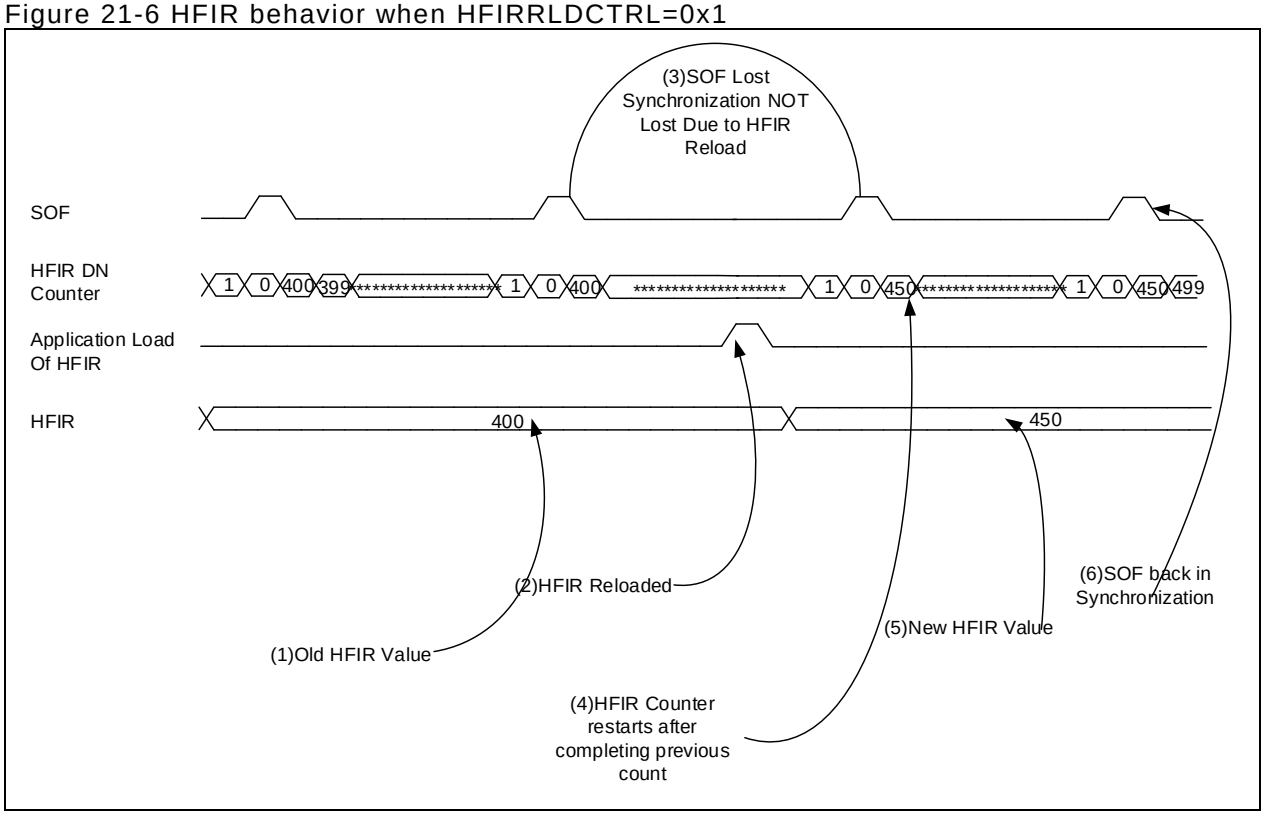


The sequence of operation is as follows:

1. After power-on reset, the current HFIR value set by the application is shown
2. The application loads a new value into the HFIR register
3. The HFIR downcounter is reloaded, so it will immediately restart counting to cause SOF synchronization loss
4. Restart HFIR counter
5. The HFIR register receives a new programmed value
6. Obtain SOF synchronization again after the first SOF is generated using the HFIR new feature

Figure 21-6 shows the HFIR behavior when HFIRRLDCTRL=0x1 in the OTGFS_HFIR register.

Figure 21-6 HFIR behavior when HFIRRLDCTRL=0x1



The sequence of operation is as follows:

1. After power-on reset, the current HFIR value set by the application is shown
2. The application loads a new HFIR value; the HFIR counter does not apply this new value, but continues counting until it reaches 0
3. The counter generates a SOF when it reaches 0 using the old HRIF value
4. The HFIR counter applies a new value
5. New HFIR value takes effect

The SOF synchronization resumes after going through above-mentioned steps.

21.5.3.7 Initialize bulk and control IN transfers

Figure 21-7 shows a typical bulk or control IN transfer operation. Refer to channel 2 (ch_2) for more information. The assumptions are as follows:

- The application is attempting to receive two largest-packet-size packets (transfer size is 64 bytes)
- The receive FIFO contains at least one largest-packet-size packet and two status WORDs per each packet (72 bytes for full-speed transfer)
- The non-periodic request queue depth is 4

(1) Operation process for common bulk and control IN transfers

The sequence of operations shown in Figure 21-7 is as follows:

1. Initialize channel 2 (according to OTGFS channel initialization requirements)
2. Set the CHENA bit in the OTGFS_HCCHAR2 register to write an IN request to the non-periodic request queue
3. The controller issues an IN token after completing the current OUT transfer
4. The controller generates a RXFLVL interrupt as soon as the receive packet is written into the receive FIFO
5. To handle the RXFLVL interrupt, mask the RXFLVL interrupt and read the received packet status to determine the number of bytes received, and then read the receive FIFO. Following this step to unmask the RXFLVL interrupt
6. The controller generates the RXFLVL interrupt when the transfer complete status is written into the

receive FIFO

7. The application must read the receive packet status, and ignore it when the receive packet status is not an IN data packet
8. The controller generates the XFERC interrupt as soon as the receive packet is read
9. To handle the XFERC interrupt, disable the channel (see Halting a channel) and stop writing the OTGFS_HCCHAR2 register. The controller writes a channel halted request to the non-periodic request queue once the OTGFS_HCCHAR2 register is written
10. The controller generates the RXFLVL interrupt as soon as the halt status is written to the receive FIFO
11. Read and ignore the receive packet status
12. The controller generates a CHHLTD interrupt as soon as the halt status is read from the receive FIFO
13. In response to the CHHLTD interrupt, the processor does not allocate the channel for other transfers.

(2) Handling interrupts

The following code describes the interrupt service routine related to the channel during bulk and control IN transfers

```
Unmask (XACTERR/XFERC/BBLERR/STALL/DATATGLERR)
if (XFERC)
    {
    Reset Error Count
    Unmask CHHLTD
    Disable Channel
    Reset Error Count
    Mask ACK
    }
else if (XACTERR or BBLERR or STALL)
    {
    Unmask CHHLTD
    Disable Channel
    if (XACTERR)
        {
        Increment Error Count
        Unmask ACK
        }
    }
else if (ChHltd)
    {
    Mask CHHLTD
    if (Transfer Done or (Error_count == 3))
        {
        De-allocate Channel
        }
    else
        {
        Re-initialize Channel
        }
    }
else if (ACK)
    {
    Reset Error Count
```

```
    Mask ACK
    }
else if (DATATGLERR)
    {
    Reset Error Count
}
```

21.5.3.8 Initialize bulk and control OUT/SETUP transfers

Figure 21-7 shows a typical bulk or control transfer OUT/SETUP transfer operation. Refer to channel 1 (ch_1) for more information. It is necessary to send two bulk transfer OUT packets. The control transfer SETUP operation is the same, just the fact that it has only one packet. The assumptions are as follows:

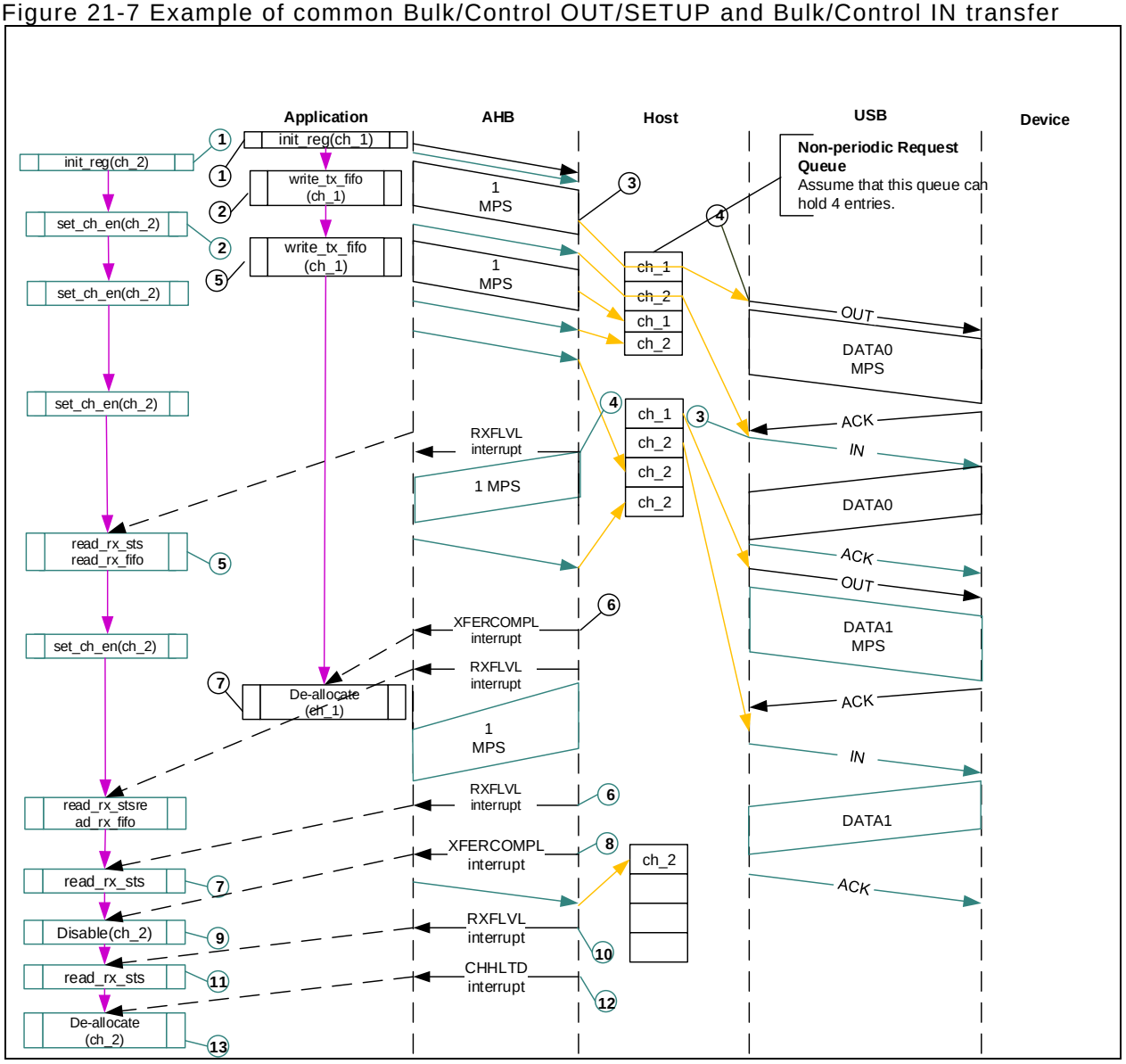
- The application is attempting to send two largest-packet-size packets (transfer size is 64 bytes)
- The non-periodic transmit FIFO can store two packets (128 bytes for full-speed transfer)
- The non-periodic request queue depth is 4

(1) OUT/SETUP operation process for common bulk and control transfer

The sequence of operations shown in Figure 21-7 is as follows:

1. Initialize channel 1 (according to OTGFS channel initialization requirements)
2. Write the first packet for channel 1
3. Along with the last WORD write, the controller writes a request to the non-periodic request queue
4. The controller sends an OUT token in the current frame as soon as the non-periodic queue becomes empty
5. Write the second packet (the last one) to the channel 1
6. The controller generate an XFERC interrupt as soon as the previous transfer is completed successfully
7. In response to the XFERC interrupt, the processor does not allocate the channel for other transfers.

Figure 21-7 Example of common Bulk/Control OUT/SETUP and Bulk/Control IN transfer



(2) Handling interrupts

The following code describes the interrupt service routine related to the channel during bulk and control transfer OUT/SETUP operation

```
Unmask (NAK/XACTERR/NYET/STALL/XFERC)
if (XFERC)
    {
    Reset Error Count
    Mask ACK
    De-allocate Channel
    }
else if (STALL)
    {
    Transfer Done = 1
    Unmask CHHLTD
    Disable Channel
    }
else if (NAK or XACTERR or NYET)
    {
    Rewind Buffer Pointers
```

```
    Unmask CHHLTD
    Disable Channel
    if (XactErr)
        {
        Increment Error Count
        Unmask ACK
        }
    else
        {
        Reset Error Count
        }
    }
else if (CHHLTD)
    {
    Mask CHHLTD
    if (Transfer Done or (Error_count == 3))
        {
        De-allocate Channel
        }
    else
        {
        Re-initialize Channel (Do ping protocol for HS)
        }
    }
else if (ACK)
    {
    Reset Error Count
    Mask ACK
}
```

Notes:

- The application can only write the transmit FIFO when the transmit FIFO and request queue has free spaces. The application must check whether there is a free space in the transmit FIFO through the NPTXFEMP bit in the OTGFS_GINTSTS register
- The application can only write a request when the request queue has free spaces and wait until an XFERC interrupt is received

21.5.3.9 Initialize interrupt IN transfers

Figure 21-8 shows the operation process of a typical interrupt IN transfer. Refer to channel 2 (ch_2). The assumptions are as follows:

- The application is attempting to receive one largest-packet-size packet (transfer size is 64 bytes) from an odd frame
- The receive FIFO can store at least one largest-packet-size packet and two status WORDs per packet (1031 bytes for full-speed transfer)
- The periodic request queue depth is 4

(1) Common interrupt IN operation process

The sequence of operations shown in Figure 21-8 (channel 2) is as follows:

1. Initialize channel 2 (according to OTGFS channel initialization requirements). The application must set the ODDFRM bit in the OTGFS_HCCHAR2 register
2. Set the CHENA bit in the OTGFS_HCCHAR2 register to write an IN request to the periodic request queue
3. The OTGFS host writes an IN request to the periodic request queue each time the CHENA is set in the OTGFS_HCCHAR2 register

4. The OTGFS host attempts to send an IN token in the next frame (odd)
5. The OTGFS host generates a RXFLVL interrupt as soon as an IN packet is received and written to the receive FIFO
6. To handle the RXFLVL interrupt, read the received packet status to determine the number of bytes received, then read the receive FIFO. The application must mask the RXFLVL interrupt before reading the receive FIFO, and unmask the interrupt after reading the entire packet
7. The controller generates the RXFLVL interrupt when the transfer complete status is written to the receive FIFO. The application must read and ignore the receive packet when the receive packet is not an IN packet
8. The controller generates an XFERC interrupt as soon as the receive packet is read
9. To handle the XFERC interrupt, read the PKTCN bit in the OTGFS_HCTSIZ2 register. If the PKTCNT bit in the OTGFS_HCTSIZ2 is not equal to 0, disable the channel before re-initializing the channel for the next transfer. If PKTCNT == 0 in the OTGFS_HCTSIZ2 register, re-initialize the channel for the next transfer. In this case, the application must reset the ODDFRM bit in the OTGFS_HCCHAR2 register.

(2) Handling interrupts

The following code describes the interrupt service routine related to the channel during interrupt IN transfer

```
Unmask (NAK/XACTERR/XFERC/BBLERR/STALL/FRMOVRUN/DATATGLERR)
if (XFERC)
  {
  Reset Error Count
  Mask ACK
  if (HCTSIZx.PKTCNT == 0)
    {
    De-allocate Channel
    }
  else
    {
    Transfer Done = 1
    Unmask CHHLTD
    Disable Channel
    }
  }
else if (STALL or FRMOVRUN or NAK or DATATGLERR or BBLERR)
  {
  Mask ACK
  Unmask CHHLTD
  Disable Channel
  if (STALL or BBLERR)
    {
    Reset Error Count
    Transfer Done = 1
    }
  else if (!FRMOVRUN)
    {
    Reset Error Count
    }
  }
else if (XACTERR)
  {
```

```
    Increment Error Count
    Unmask ACK
    Unmask CHHLTD
    Disable Channel
    }
 else if (CHHLTD)
    {
    Mask CHHLTD
    if (Transfer Done or (Error_count == 3))
       {
       De-allocate Channel
       }
    else Re-initialize Channel (in next b_interval - 1 uF/F)
       }
    }
 else if (ACK)
    {
    Reset Error Count
    Mask ACK
 }
```

The application can only write a request to the same channel when the remaining space in the request queue reaches the number defined in the MC field, before switching to other channels (if any).

21.5.3.10 Initialize interrupt OUT transfers

Figure shows a typical interrupt OUT transfer operation. Refer to channel 1 (ch_1). The assumptions are as follows:

- The application is attempting to send one largest-packet-size packet (transfer size is 64 bytes) to every frame
- The periodic transmit FIFO can store one packet (1KB bytes for full-speed transfer)
- The periodic request queue depth is 4

(1) Common interrupt IN operation process

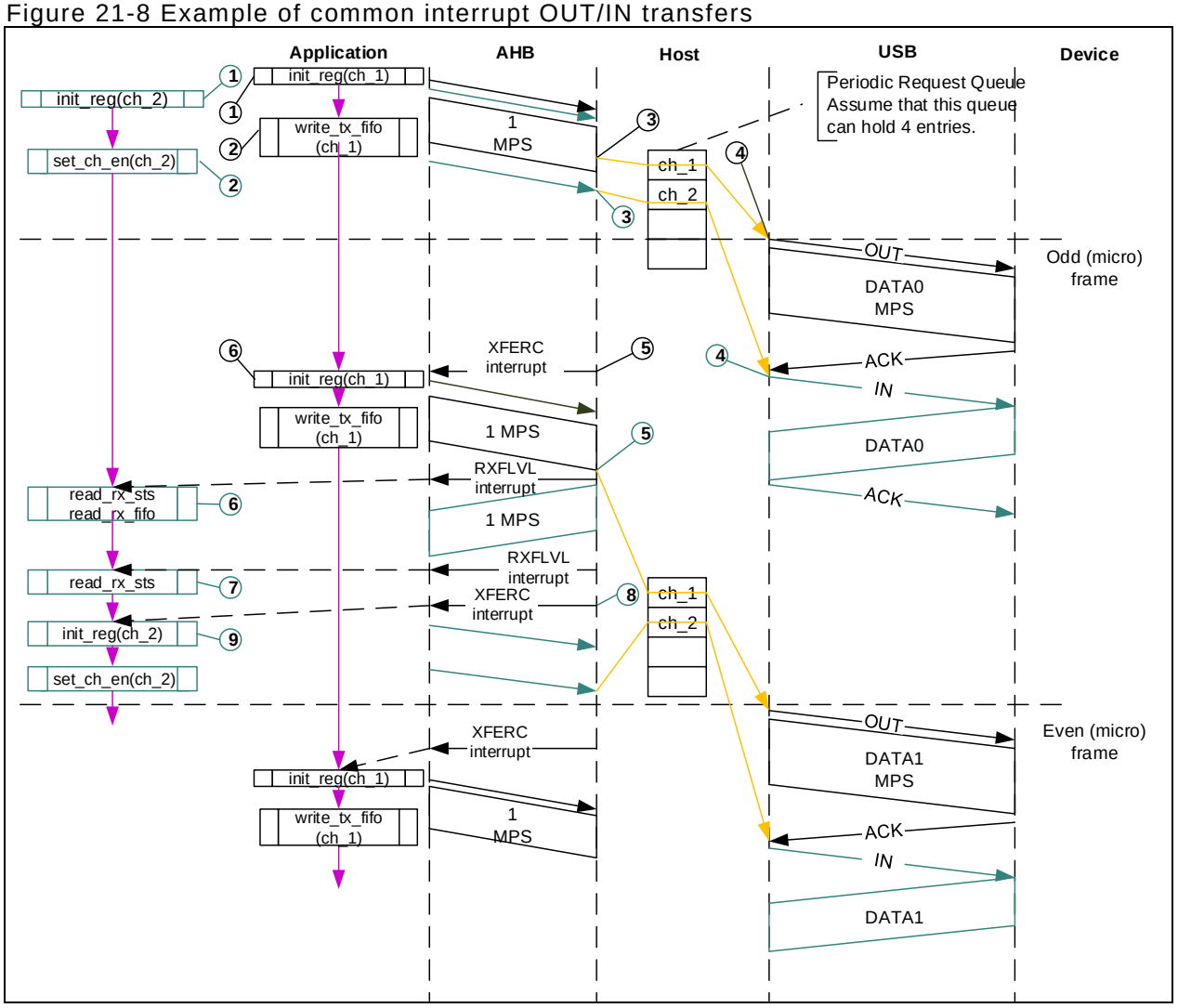
The sequence of operations shown in Figure 21-8 (channel 1) is as follows:

1. Initialize channel 1 (according to OTGFS channel initialization requirements). The application must set the ODDFRM bit in the OTGFS_HCCHAR2 register
2. Write the first packet to the channel 1
3. Along with the last WORD write of each packet, the host writes a request to the periodic request queue
4. The host sends an OUT token in the next frame (odd)
5. The host generates an XFERC interrupt after the last packet is transmitted successfully
6. In response to the XFERC interrupt, re-initialize the channel for the next transfer.

(2) Handling interrupts

Figure 21-8 shows an example of common interrupt OUT/IN transfers

Figure 21-8 Example of common interrupt OUT/IN transfers



The following code describes the interrupt service routine related to the channel during interrupt OUT transfers

```
Unmask (NAK/XACTERR/STALL/XFERC/FRMOVRUN)
if (XFERC)
{
Reset Error Count
Mask ACK
De-allocate Channel
}
else if (STALL or FRMOVRUN)
{
Mask ACK
Unmask CHHLTD
Disable Channel
if (STALL)
{
Transfer Done = 1
}
}
else if (NAK or XACTERR)
{
Rewind Buffer Pointers
```

```
Reset Error Count
Mask ACK
Unmask CHHLTD
Disable Channel
}
else if (CHHLTD)
{
Mask CHHLTD
if (Transfer Done or (Error_count == 3))
{
De-allocate Channel
}
else
{
Re-initialize Channel (in next b_interval - 1 uF/F)
}
}
else if (ACK)
{
Reset Error Count
Mask ACK
}
```

Before switching to other channels (if any), the application can only write packets based on the number defined in the MC filed to the transmit FIFO and request queue when the transmit FIFO has free spaces. The application can determine whether the transmit FIFO has free spaces through the NPTXFEMP bit in the OTGFS_GINTSTS register.

21.5.3.11 Initialize synchronous IN transfers

Figure 21-9 shows the operation process of a typical synchronous IN transfer. Refer to channel 2 (ch_2). The assumptions are as follows:

- The application is attempting to receive one largest-packet-size packet (transfer size is 1023 bytes) from the next odd frame
- The receive FIFO can store at least one largest-packet-size packet and two status WORDs per packet (1031 bytes for full-speed transfer)
- The periodic request queue depth is 4

(1) Common interrupt IN operation process

The sequence of operations shown in Figure 21-9 (channel 2) is as follows:

1. Initialize channel 2 (according to OTGFS channel initialization requirements). The application must set the ODDFRM bit in the OTGFS_HCCHAR2 register
2. Set the CHENA bit in the OTGFS_HCCHAR2 register to write an IN request to the periodic request queue
3. The OTGFS host writes an IN request to the periodic request queue each time the CHENA is set in the OTGFS_HCCHAR2 register
4. The OTGFS host attempts to send an IN token in the next frame (odd)
5. The OTGFS host generates a RXFLVL interrupt as soon as an IN packet is received and written to the receive FIFO
6. To handle the RXFLVL interrupt, read the received packet status to determine the number of bytes received, then read the receive FIFO. The application must mask the RXFLVL interrupt before reading the receive FIFO, and unmask the interrupt after reading the entire packet
7. The controller generates the RXFLVL interrupt when the transfer complete status is written to the receive FIFO. The application must read and ignore the receive packet when the receive packet is

not an IN packet (GRXSTSR.PKTSTS!= 0x0010)

8. The controller generates an XFERC interrupt as soon as the receive packet is read
9. To handle the XFERC interrupt, read the PKTCN bit in the OTGFS_HCTSIZ2 register. If the PKTCNT bit in the OTGFS_HCTSIZ2 is not equal to 0, disable the channel before re-initializing the channel for the next transfer. If PKTCNT == 0 in the OTGFS_HCTSIZ2 register, re-initialize the channel for the next transfer. In this case, the application must reset the ODDFRM bit in the OTGFS_HCCHAR2 register.

(2) Handling interrupts

The following code describes the interrupt service routine related to the channel during synchronous IN transfers

```
Unmask (XACTERR/XFERC/FRMOVRUN/BBLERR)
if (XFERC or FRMOVRUN)
  {
  if (XFERC and (HCTSIZx.PKTCNT == 0))
    {
    Reset Error Count
    De-allocate Channel
    }
  else
    {
    Unmask CHHLTD
    Disable Channel
    }
  }
else if (XACTERR or BBLERR)
  {
  Increment Error Count
  Unmask CHHLTD
  Disable Channel
  }
else if (CHHLTD)
  {
  Mask CHHLTD
  if (Transfer Done or (Error_count == 3))
    {
    De-allocate Channel
    }
  else
    {
    Re-initialize Channel
    }
}
```

21.5.3.12 Initialize synchronous OUT transfers

Figure 21-9 shows a typical synchronous OUT transfer operation. Refer to channel 1 (ch_1). The assumptions are as follows:

- The application is attempting to send one largest-packet-size packet (transfer size is 1023 bytes) to every frame from the next odd frame
- The periodic transmit FIFO can store one packet (1KB bytes for full-speed transfer)
- The periodic request queue depth is 4

(1) Common interrupt IN operation process

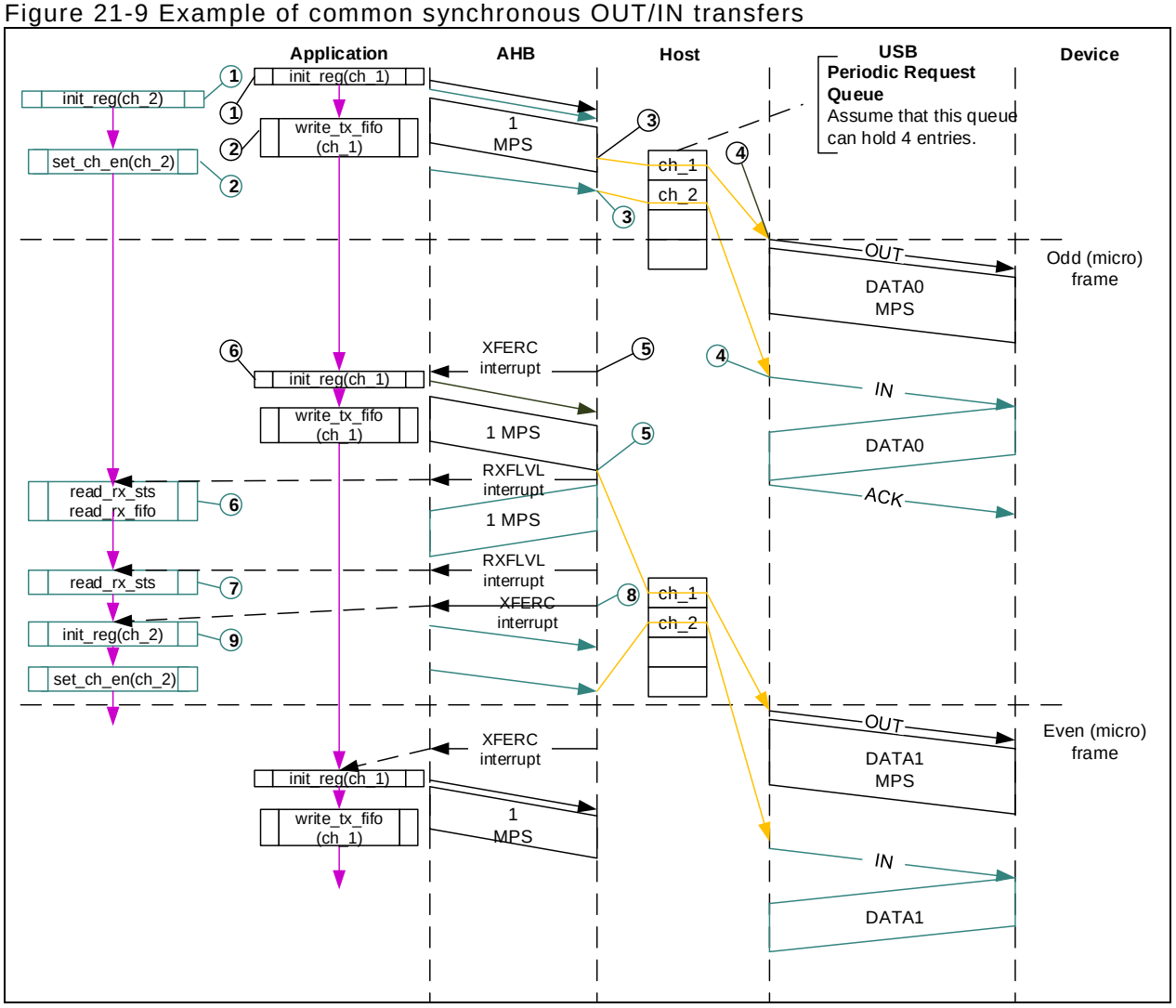
The sequence of operations shown in Figure 21-9 (channel 2) is as follows:

1. Initialize channel 1 (according to OTGFS channel initialization requirements). The application must set the ODDFRM bit in the OTGFS_HCCHAR2 register
2. Write the first packet to the channel 1
3. Along with the last WORD write of each packet, the host writes a request to the periodic request queue
4. The OTGFS host sends an OUT token in the next frame (odd)
5. The host generates an XFERC interrupt after the last packet is transmitted successfully
6. In response to the XFERC interrupt, re-initialize the channel for the next transfer.

(2) Handling interrupts

Figure 21-9 shows an example of common synchronous OUT transfers

Figure 21-9 Example of common synchronous OUT/IN transfers



The following code describes the interrupt service routine related to the channel during synchronous OUT transfers

```
Unmask (FRMOVRUN/XFERC)
if (XFERC)
  {
  De-allocate Channel
  }
else if (FRMOVRUN)
  {
  Unmask CHHLTD
  Disable Channel
```

```
  }
else if (CHHLTD)
  {
  Mask CHHLTD
  De-allocate Channel
}
```

21.5.4 OTGFS device mode

21.5.4.1 Device initialization

The application must perform the following steps to initialize the controller during power-on or after switching a mode from host to device:

1. Program the following fields in the OTGFS_DCFG register
- Device speed
- Non-zero-length status OUT handshake
- Periodic frame interval

2. Clear the SFTDISCON bit in the OTGFS_DCTL register. The controller will start connection after clearing this bit

3. Program the OTGFS_GINTMSK register to unmask the following interrupts:
- USB reset
- Enumeration done
- Early suspend
- USB suspend
- SOF

4. Wait for the USBRESET interrupt in the OTGFS_GINTSTS register. It indicates that a reset signal has been detected on the USB (lasting for about 10ms). Upon receiving this interrupt, the application must follow the steps defined in USB initialization on USB reset.

5. Wait for the ENUMDONE interrupt in the OTGFS_GINTSTS register. It indicates the end of USB reset. Upon receiving this interrupt, the application must read the OTGFS_DSTS register to determine the enumeration speed and perform the steps defined in Endpoint initialization on enumeration completion. At this time, the device is ready to accept SOF packets and perform control transfers on control endpoint 0.

21.5.4.2 Endpoint initialization on USB reset

This section describes the operations required for the application to perform when a USB reset signal is detected:

1. Set the NAB bit for all OUT endpoints
- OTGFS_DOEPCTLx.SNAK = 0x1(for all OUT endpoints)

2. Unmask the following interrupt bits
- OTGFS_DAINTMSK.INEP0 = 0x1(control IN endpoint 0)
- OTGFS_DAINTMSK.OUTEP0 = 0x1(control OUT endpoint 0)
- OTGFS_DOEPMSK.SETUP = 0x1
- OTGFS_DOEPMSK.XFERC = 0x1
- OTGFS_DIEPMSK.XFERC = 0x1
- OTGFS_DIEPMSK.TIMEOUT = 0x1

3. To receive/transmit data, the device must perform Device initialization steps to initialize registers

4. Allocate SRAM for each endpoint
- Program the OTGFS_GRXFSIZ register to be able to receive control OUT data and SETUP data. If the allocated SRAM is equal to at least 1 largest-packet-size of control endpoint 0 + 2 WORDs (for the status of the control OUT data packet) +10 WORDs (for setup packets)
- Program the OTGFS_DIEPTXF0 register to be able to transmit control IN data. The allocated SRAM is equal to at least 1 largest-packet-size of control endpoint 0

5. Reset the device address in the device configuration register

6. Program the following fields in the endpoint-specific registers to ensure that control OUT endpoint 0 is able to receive a SETUP packet

- OTGFS_DOEPTSIZ0.SUPCNT = 0x3(to receive up to 3 consecutive SETUP packets)

At this point, all initialization required to receive SETUP packets is done.

21.5.4.3 Endpoint initialization on enumeration completion

This section describes the operations required for the application to perform when an enumeration completion interrupt signal is detected:

- Upon detecting the enumeration completion interrupt signal, read the OTGFS_DSTS register to get the enumeration speed
- Program the MPS bit in the OTGFS_DIEPCTL0 register to set the maximum packet size. This operation is used to configure control endpoint 0. The maximum packet size for a control endpoint depends on the enumeration speed
- Unmask SOF interrupts.

At this point, the device is ready to receive SOF packets and has been configured to perform control transfers on control endpoint 0.

21.5.4.4 Endpoint initialization on SetAddress command

This section describes the operations required for the application to perform when the application receives a SetAddress command in a SETUP packet

- Program the OTGFS_DCFG register with the device address received in the SetAddress command
- Program the controller to send an IN packet

21.5.4.5 Endpoint initialization on SetConfiguration/SetInterface command

This section describes the operations required for the application to perform when the application receives a SetConfiguration / SetInterface command in a SETUP packet

- When a SetConfiguration command is received, the application must program the endpoint registers according to the characteristics of the valid endpoints defined in the new configuration
- When a SetInterface command is received, the application must program the endpoint registers of the endpoints affected by this command
- Some endpoints that were valid in the previous configuration are not valid in the new configuration. These invalid endpoints must be disabled
- Refer to Endpoint activation and USB endpoint deactivation for more information on how to activate or disable a certain endpoint
- Unmask the interrupt for each valid endpoint and mask the interrupts for all invalid endpoints in the DAINTMSK register
- Refer to OTGFS FIFO configuration for more information on how to program SRAM for each FIFO
- After all required endpoints are configured, the application must program the controller to send a status IN packet

At this point, the device controller has been ready to receive and transmit any type of data packet.

21.5.4.6 Endpoint activation

This section describes how to activate a device endpoint or configure an existing device endpoint to a new type.

1.Program the following bits in the OTGFS_DIEPCTLx register (for IN or bidirectional endpoints) or the OTGFS_DOEPCTLx register (for OUT or bidirectional endpoints)

- Largest packet size
- USB valid endpoint = 0x1
- Endpoint start data toggle (for interrupt and bulk endpoints
- Endpoint type
- Transmit FIFO number

2. Once the endpoint is activated, the controller starts decoding the tokens issued to this endpoint and sends out a valid handshake for each valid token received for the endpoint

21.5.4.7 USB endpoint deactivation

This section describes how to deactivate an existing endpoint. Disable the suspended transfer before performing endpoint deactivation.

- Clear the USB valid endpoint bit in the OTGFS_DIEPCTLx register (for IN or bidirectional endpoints) or the OTGFS_DOEPCTLx register (for OUT or bidirectional endpoints)
- Once the endpoint is deactivated, the controller will ignore the tokens issued to this endpoint, which causes a USB timeout.

21.5.4.8 Control write transfers (SETUP/Data OUT/Status IN)

This section describes the steps required for control write transfers.

The application programming process is as follows:

1. When the SETUP bit is set in the OTGFS_DOEPINTx register, it indicates that a valid SETUP packet has been sent to the application, and data stage is initiated, see OUT data transfers. At the end of the SETUP stage, the application must rewrite 3 to the SUPCNT bit in the OTGFS_DOEPTSIZx register to receive the subsequent SETUP packet

2. If the last SETUP packet received before the generation of the SETUP interrupt indicates data OUT stage, program the controller to perform OUT transfers based on Asynchronous OUT data transfer operation

3. The application can receive up to 64-byte data for a single OUT data transfer of control endpoint 0. If the application expects to receive more than 64-byte data during data OUT stage, it must re-enable the endpoint to receive another 64-byte data, and it must continue this operation until the completion of all data reception in data stage

4. When the XFERC interrupt is set in the OTGFS_DOEPINTx register during the last OUT transfer, it indicates the end of data OUT stage of control transfer

5. Once the completion of data OUT stage, the application must perform the following steps:

- If the application needs to transfer a new SETUP packet, it must re-enable control OUT endpoints (refer to OUT data transfers)

  OTGFS_DOEPCTLx.EPENA = 0x1

- To execute the received SETUP commands, the application must configure the corresponding registers in the controller. This is optional, depending on the received SETUP command type

6. During status IN stage, the application must follow the requirements of Non-periodic (for bulk and control) IN data transfers to program registers to perform data IN transfers

7. When the XFERC interrupt is set in the OTGFS_DOEPINTx register is set, it indicates that the status stage of control transfers is started. As soon as Data transfer complete mode and Status stage start bit are set in the receive FIFO packet status register, the controller generates an interrupt. The Transfer complete interrupt can be cleared through the XFERC bit in the OTGFS_DOEPINTx register

Repeat above-mentioned steps until an interrupt (XFERC bit in the OTGFS_DIEPINTx register) is generated on the endpoint, which indicates the end of control write transfers.

21.5.4.9 Control read transfers (SETUP/Data IN/Status OUT)

This section describes the steps required for control read transfers.

The application programming process is as follows:

- When the SETUP bit is set in the OTGFS_DOEPINTx register, it indicates that a valid SETUP packet has been sent to the application, and data stage is initiated, see OUT data transfers. At the end of the SETUP stage, the application must rewrite 3 to the SUPCNT bit in the OTGFS_DOEPTSIZx register to receive the subsequent SETUP packet
- If the last SETUP packet received before the generation of the SETUP interrupt indicates data IN stage, program the controller to perform IN transfers based on Non-periodic IN data transfer operation

- The application can receive up to 64-byte data for a single IN data transfer of control endpoint 0. If the application expects to receive more than 64-byte data during data IN stage, it must re-enable the endpoint to receive another 64-byte data, and it must continue this operation until the completion of all data transfers in data stage
- Repeat above-mentioned steps until the XFERC interrupt is generated in the OTGFS_DIEPINTx register for each IN transfer on the endpoint
- When the XFERC interrupt is set in the OTGFS_DOEPINTx register during the last IN transfer, it indicates the end of data OUT stage of control transfer
- To execute data OUT transfer at status OUT stage, the application must configure the controller. This is optional.

The application must program the NZSTSOUTHSHK bit in the OTGFS_DCFG register, and then send data OUT transfer at status stage

The XFERC interrupt bit is set in the OTGFS_DOEPINTx register to indicate the end of status OUT stage of control transfer, marking the completion of control read transfers.

21.5.4.10 Control transfers (SETUP/Status IN)

This section describes the two-phase control transfer operation.

The application programming process is as follows:

1. When the SETUP bit is set in the OTGFS_DOEPINTx register, it indicates that a valid SETUP packet has been sent to the application, and data stage is initiated, see OUT data transfers. At the end of the SETUP stage, the application must rewrite 3 to the SUPCNT bit in the OTGFS_DOEPTSIZx register to receive the subsequent SETUP packet
2. The application decodes the last SETUP packet received before the generation of the SETUP interrupt. If the SETUP packet indicates two-level control commands, the application must perform the following steps:

- Set OTGFS_DOEPCTLx.EPENA = 0x1
- The application must program the registers in the controller to perform the received SETUP commands

3. For status IN stage, the application must program the registers based on Non-periodic (bulk and control) IN data transfers to perform data IN transfers

4. The XFERC interrupt bit is set in the OTGFS_DIEPINTx register to indicate the end of status IN stage of control transfers.

21.5.4.11 Read FIFO packets

This section describes how to read FIFO packets (OUT data and SETUP packets)

1. The application must read the OTGFS_GRXSTSP register as soon as the RXFLVL interrupt bit is detected in the OTGFS_GINTSTS register
2. The application can mask the RXFLVL interrupt bit in the OTGFS_GINTSTS register by setting RXFLVL = 0x0 in the OTGFS_GINTMSK register, until it has read the data packets from the receive FIFO
3. If the received packet byte is not 0, the byte count amount of data is popped from the receive data FIFO and stored in memory. If the received packet byte count is 0, no data is read from the receive data FIFO

4. The receive FIFO packet status reading indicates one of the following conditions:

5. Global OUT NAK mode: PKTSTS = Global OUT NAK, BCNT = 0x000, EPNUM = Don't Care (0x0) and DPID = Don't Care (0x00), indicating that the global OUT NAK bit has taken effect

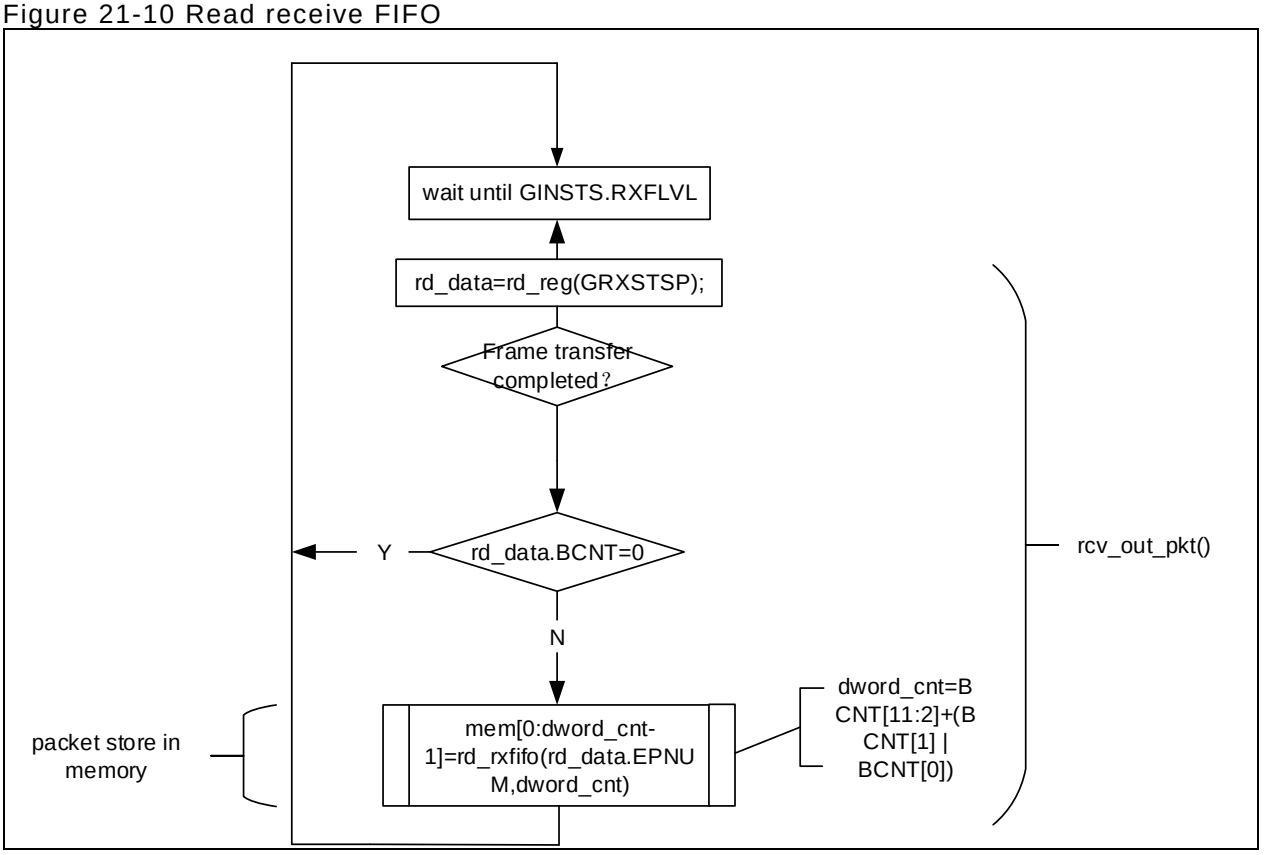
- SETUP packet mode: PKTSTS = SETUP, BCnt = 0x008, EPNUM = Control EP Num and DPID = D0, indicating that a SETUP packet for the specified endpoint is now available for reading from the receive FIFO
- Setup stage done mode: PKTSTS = Setup Stage Done, BCNT = 0x0, EPNUM = Control EP Num and DPID = Don't Care (0x00), indicating the completion of the Setup stage for the specified

endpoint, and the start of the data stage. After this request is popped from the receive FIFO, the controller triggers a Setup interrupt on the specified control OUT endpoint

- Data OUT packet mode: PKTSTS = DataOUT, BCnt =size of the received data OUT packet (0 ≤ BCNT ≤ 1024), EPNUM =Endpoint number on which the data packet was received, DPID =Actual data PID
- Data transfer complete mode: PKTSTS = Data OUT transfer done, BCNT = 0x0, EPNUM =OUT endpoint number on which the data transfer is complete, DPID = Don't Care (0x00). These data indicate that an OUT data transfer for the specified OUT endpoint has been complete. After this request is popped from the receive FIFO, the controller triggers a Transfer Completed interrupt on the specified OUT endpoint. PKTSTS code can be found in the OTGFS_GRXSTSR / OTGFS_GRXSTSP register

7. After the valid data is popped from the receive FIFO, the RXFLVL interrupt bit in the OTGFS_GINTSTS register must be unmasked
8. Step 1-5 must be repeated each time the application detects the interrupt line due to the RXFLVL bit in the OTGFS_GINTSTS register. Reading an empty receive FIFO will result in unexpected behavior. Figure 21-10 shows a flowchart.

Figure 21-10 Read receive FIFO



21.5.4.12 OUT data transfers

This section describes the internal data flow during data OUT and SETUP transfers, and how the application handles SETUP transfers.

(1) Setup transfers

This section describes how to handle SETUP data packets and the application's operating sequence of handling SETUP transfers. After power-on reset, the application must follow the OTGFS Initialization process to initialize the controller. Before communicating with the host, the application must initialize the endpoints based on Device Initialization, and refer to Read FIFO packets for more information.

[Application requirements]

1. To receive a SETUP packet, the SUPCNT bit (OTGFS_DOEPTSIZx) on a control OUT endpoint must be programmed to be a non-zero value. When the application sets the SUPCNT bit to a non-zero value, the controller receives SETUP packets and writes them to the receive FIFO, irrespective

of the NAK status bit and EPENA bit in the OTGFS_DOEPCTLx register. The SUPCNT bit is decremented each time the control endpoint receives a SETUP packet. If the SUPCNT bit is not programmed to a proper value before receiving a SETUP packet, the controller still receives the SETUP packet and decrements the SUPCNT bit, but the application may not be able to determine the exact number of SETUP packets received in the SETUP stage of a control transfer.

- OTGFS_DOEPTSIZx.SUPCNT = 0x3

2. The application must allocate some extra space for the receive data FIFO to ensure that up to three SETUP packets can be received on a control endpoint

- The space to be reserved is 13 WORDs. Four WORDs are required for one SETUP packet, one WORD is required for the Setup stage and 8 WORDs are required to store two extra SETUP packets among all control endpoints
- Four WORDs per SETUP packet are required to store 8-byte SETUP data and 4-byte Transfer completed status and 4-byte SETUP status (SETUP packet mode). The controller must reserve this space to receive data
- FIFO is used to write SETUP data only, and never for data packets

3. The application must read 2-WORDs SETUP packet from the receive data

4. The application must read and discard the Transfer Completed status WORD from the receive FIFO, and ignore the Transfer Completed interrupt due to this read operation.

[Internal data flow]

1. When a SETUP packet is received, the controller writes the received data to the receive FIFO, without checking whether there is available space in the receive FIFO, irrespective of the NAK and Stall bits on the control endpoints.

- The controller sets the IN NAK and OUT NAK bits for the control IN/OUT endpoints on which the SETUP packet was received.

2. For every SETUP packet received on the USB line, 3 WORDs of data are written to the receive FIFO, and the SUPCNT bit is decremented by 1 automatically.

- The first WORD contains control information used internally by the controller
- The second WORD contains the first 4 bytes of the SETUP command
- The third WORD contains the last 4 bytes of the SETUP command

3. When the SETUP stage switches to data IN/OUT stage, the controller writes a SETUP status done WORD to the receive FIFO, indicating the end of the SETUP stage.

4. The application reads the SETUP packages through the AHB bus.

5. When the application pops the Setup stage done WORD from the receive FIFO, the controller interrupts the application through the SETUP interrupt bit in the OTGFS_DOEPINTx register, indicating that the application can start processing the SETUP packet received.

6. The controller clears the endpoint enable bit for control OUT endpoints.

[Application programming process]

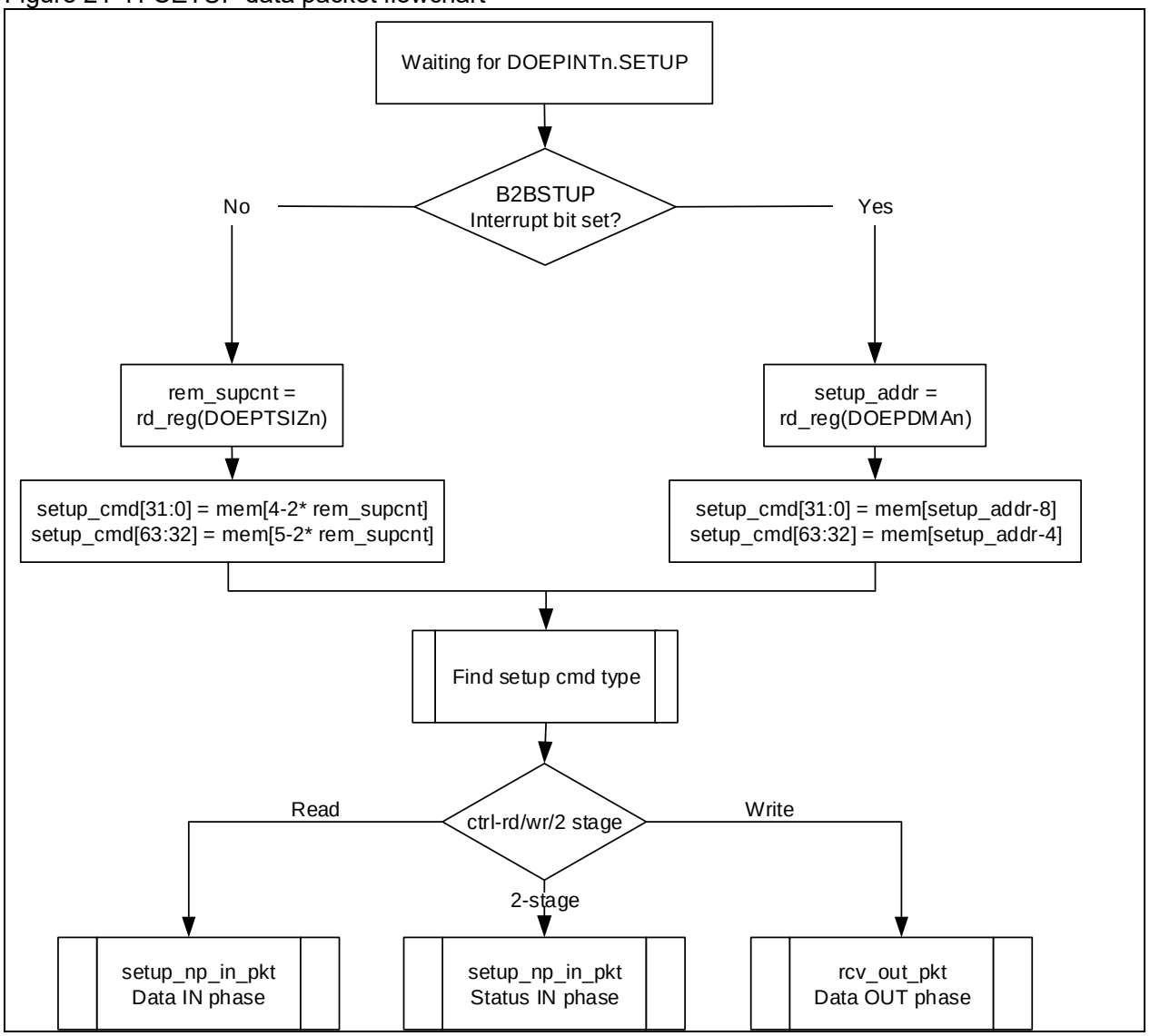
1. Program the OTGFS_DOEPTSIZx register

- OTGFS_DOEPTSIZx.SUPCNT = 0x3

2. Wait for the RXFLVL interrupt bit in the OTGFS_GINTSTS register and read and empty the data packets from the receive FIFO (Refer to Read FIFO packets for details). This operation can be repeated several times.

3. When the SETUP interrupt bit is set in the OTGFS_DOEPINTx register, it indicates that the SETUP data transfer has been completed successfully. Upon this interrupt, the application must read the OTGFS_DOEPTSIZx register to determine the number of SETUP packets received, and process the last received SETUP packet.

Figure 21-11 SETUP data packet flowchart



(2) Handling more than three consecutive SETUP packets

Per the USB 2.0 specification, typically, a host does not send more than three consecutive SETUP packets to the same endpoint during a SETUP packet error. However, the USB2.0 specification does not limit the number of consecutive SETUP packets a host can send to the same endpoint. If this condition occurs, the OTGFS controller generates an interrupt (B2BSTUP bit in the OTGFS_DOEPINTx register).

21.5.4.13 IN data transfers

This section describes the internal data flow during IN data transfers and how the application handles IN data transfers.

1. The application can either select a polling or an interrupt mode.

- In polling mode, the application monitors the status of the endpoint transmit data FIFO by reading the OTGFS_DTXFSTSx register to determine whether there is enough space in the data FIFO.
- In interrupt mode, the application must wait for the TXFEMP interrupt bit in the OTGFS_DIEPINTx register, and then read the OTGFS_DTXFSTSx register to determine whether there is enough space in the data FIFO.
- To write a single non-zero-length data packet, there must be enough space to write the entire data packet in the data FIFO.
- To write zero-length data packet, the application does not need to check the FIFO space.

2. Either way, when the application determines that there is enough space to write a transmit packet, the

application can first write into the endpoint control register before writing the data into the data FIFO. Normally, except for setting the endpoint enable bit, the application must do a read modify write on the OTGFS_DIEPCTLx register to avoid modifying the contents of the register. If the space is enough, the application can write multiple data packets for the same endpoint into the transmit FIFO. For the periodic IN endpoints, the application must write packets for only one frame. It can write packets for the next periodic transfer only after the previous transfer has been completed.

21.5.4.14 Non-periodic (bulk and control) IN data transfers

To initialize the controller after power-on reset, the application must perform the steps list in OTGFS Initialization. Before communicating with a host, the controller must follow the steps defined in Device Initialization to initialize endpoints.

[Application requirements]

1. For IN transfers, the Transfer Size bit in the Endpoint Transfer Size register indicates a payload that contains multiple largest-packet-size packets and a short packet. This short packet is transmitted at the end of the transfer.

- To transmit several largest-packet-size packets and a short packet:

  Transfer size [epnum] = n * mps[epnum] + sp (n is an integer ≥ 0 and 0 ≤ sp < mps[epnum])

  If (sp > 0), then packet count [epnum] = n + 1. Otherwise, packet count [epnum] = n
- To transmit a single zero-length data packet:

  Transfer size [epnum] = 0x0

  Packet count [epnum] = 0x1
- To transmit several largest-packet-size packets and a zero-length data packet (at the end of the transfer), the application must split the transfer into two parts. First send the largest-packet-size packets and then the zero-length data packet alone.

  First transfer: Transfer size [epnum] = n * mps[epnum]; Packet count = n;

  Second transfer: Transfer size [epnum] = 0x0; Packet count = 0x1;

2. If an endpoint is enabled for data transfers, the controller updates the Transfer size register. At the end of the IN transfer (indicated by endpoint disable interrupt bit), the application must read the Transfer size register to determine how much data in the transmit FIFO have already been sent on the USB line.

3. Data fetched in the transmit FIFO = Application-programmed initial transfer size – Controller-updated final transfer size

- Data transmitted on USB = (Application-programmed initial packet count – Controller-updated final packet count) * mps[epnum]
- Data to be transmitted on USB = Application-programmed initial transfer size – Data transmitted on USB

[Internal data flow]

1. The application must set the transfer size and packet count bits in the endpoint control registers and enable the endpoint to transmit the data.

2. The application must also write the required data to the transmit FIFO of the endpoint.

3. Each time a data packet is sent to the transmit FIFO by the application the transfer size for this endpoint is decremented with the packet size. The application must continue to write data until the transfer size of the endpoint becomes 0. After writing data to the FIFO, the “packet count in the FIFO” is incremented (this is a 3-bit count for each IN endpoint transmit FIFO data packet, which is internally maintained by the controller. For an IN endpoint FIFO, the maximum number of packets maintained by the controller at any time is 8). For non-zero-length packets, a separate flag is set for each FIFO, without any data in the FIFO.

4. After the data is written to the transmit FIFO, the controller reads them upon receiving an IN token. For each non-synchronous IN data packet transmitted with an ACK handshake signal, the number of packets for the endpoint is decremented by 1, until the packet count becomes 0. The packet count is not decremented due to a timeout.

5. For zero-length data packets (indicated by an internal zero-length flag), the controller sends zero-length packets according to the IN token, and the packet count is decremented automatically.

6. If there are no data in the FIFO on a received IN token and the packet count for the endpoint is 0, the controller generates an “IN token received when FIFO is empty” interrupt, and the NAK bit for the endpoint is not set. The controller responds with a NAK handshake signal to the non-synchronous endpoints on the USB.

7. The controller rewinds the FIFO pointers internally and no timeout interrupt is generated except for the control IN endpoints.

8. When the transfer size is 0 and the packet count is also 0, the Transfer completed interrupt is generated and the endpoint enable bit is cleared.

[Application programming sequence]

1. Program the OTGFS_DIEPTSIZx register according to the transfer size and the corresponding packet count.

2. Program the OTGFS_DIEPCTLx register according to the endpoint characteristics and set the CNAK and endpoint enable bits.

3. While sending non-zero-length data packets, the application must poll the OTGFS_DTXFSTSx register (where n is the FIFO number related to that endpoint) to determine whether there is enough space in the data FIFO. The application can also use the TXFEMP bit in the OTGFS_DIEPINTx register before writing data.

21.5.4.15 Non-synchronous OUT data transfers

To initialize the controller after power-on reset, the application must perform the steps list in “OTGFS Initialization”. Before communicating with a host, the application must initialize endpoints based on the process described in “Endpoint Initialization” and by referring to “Read FIFO packets”. This section describes a regular non-synchronous OUT transfers (control, bulk or interrupt transfers).

[Application requirements]

1. For OUT data transfers, the transfer size of the endpoint transfer register must be set to a multiple of the largest packet size for the endpoint, and adjusted to the WORD boundary.

```
if (mps[epnum] mod 4) == 0
transfer size[epnum] = n * (mps[epnum]) //WORD Aligned
else
transfer size[epnum] = n * (mps[epnum] + 4 - (mps[epnum] mod 4)) //Non WORD
Aligned
packet count[epnum] = n
n > 0
```

2. When an OUT endpoint interrupt occurs, the application must read the endpoint’s transfer size register to calculate the size of the data in the memory. The received payload size must be less than the programmed transfer size.

- Payload size in memory = Application-programmed initial transfer size – Controller-updated final transfer size
- Number of USB packets the payload was received = Application-programmed initial packet count – Controller-updated final packet count

[Internal data flow]

1. The application must set the transfer size and packet count bits in the endpoint control registers, clear the NAK bit, and enable the endpoint to receive the data.

2. Once the NAK bit is cleared, the controller starts receiving data and writes it to the receive FIFO as long as there is available space in the receive FIFO. For each data packet received on the USB line, the data packet and its status are written to the receive FIFO. The packet count is decremented by 1 each time a packet (largest packet size or a short packet) is written to the receive FIFO.

- OUT data packets received with Bad Data CRC are emptied from the receive FIFO
- After sending an ACK to the data packet on the USB, the controller discards non-synchronous OUT data packets that the host (which cannot detect the ACK) re-transmits. The application does not detect multiple consecutive OUT data packets on the same endpoint with the same data PID. In this case, the packet count is not decremented.

- If there is no space in the receive FIFO, synchronous or non-synchronous data packets are ignored and not written to the receive FIFO. Besides, the non-synchronous OUT tokens receive a NAK handshake response.
- In all the above-mentioned cases, the packet count is not decremented because no data is written to the receive FIFO.

3. When the packet count becomes 0 or when a short packet is received on the endpoint, the NAK bit for the endpoint is set. Once the NAK bit is set, the synchronous or non-synchronous data packets are ignored and not written to the receive FIFO, and non-synchronous OUT tokens receive a NAK handshake response.

4. After the data is written to the receive FIFO, the application reads the data from the receive FIFO and writes it to the external memory, once packet at a time per endpoint.

5. At the end of data packet write to the external memory, the transfer size for the endpoint is decremented with the size of the written packet.

6. The OUT data transfer completed mode for an OUT endpoint is written to the receive FIFO one of the following conditions:

- The transfer size and packet count are both 0
- The last OUT data packet written to the receive FIFO is a short packet (0 ≤ data packet size < largest packet size)

7. When the application pops this entry (OUT data transfer complete), a transfer completed interrupt is generated and the endpoint enable bit is cleared.

[Application programming sequence]

1. Program the OTGFS_DOEPTSIZx register with the transfer size and the corresponding packet count.

2. Program the OTGFS_DOEPCTLx register with the endpoint characteristics, and set the endpoint enable and ClearNAK bits.

- OTGFS_DOEPCTLx.EPENA = 0x1
- OTG-DOEPCTLx.CNAK = 0x1

3. Wait for the RXFLVL interrupt in the OTGFS_GINTSTS register, and read out all data packets from the receive FIFO.

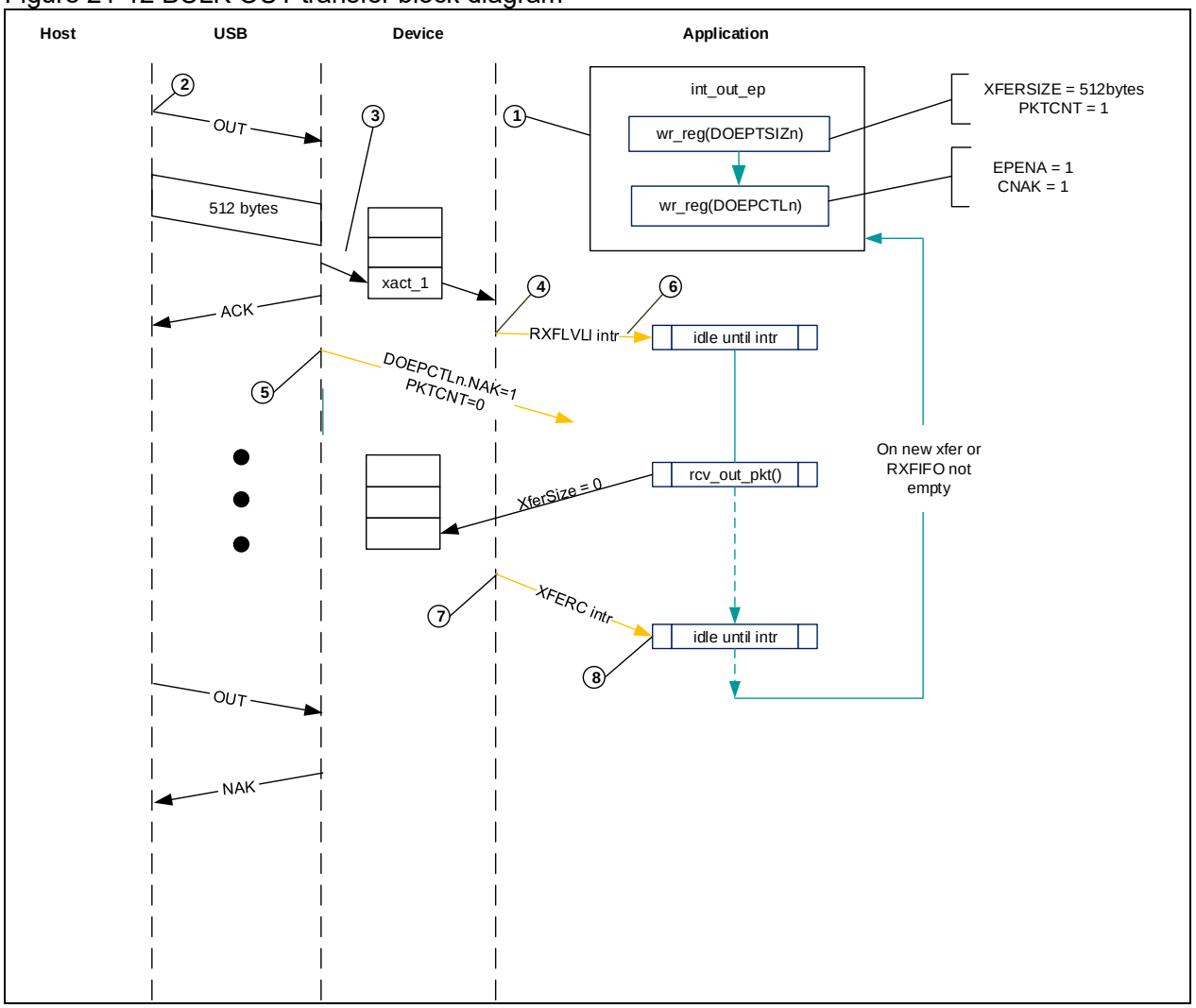
- This step can be repeated, depending on the transfer size

4. When the XFERC interrupt is set in the OTGFS_DOEPINTx register, it indicates a successful completion of the non-synchronous OUT data transfer. Read the OTGFS_DOEPTSIZx register to determine how much data has been received.

[Bulk OUT transfer]

Figure 21-12 describes the reception of a single bulk OUT data packet from the USB to the AHB and shows the events involved in the process.

Figure 21-12 BULK OUT transfer block diagram



After a SetConfiguration/SetInterface command is received, the application initializes all OUT endpoints by setting CNAK = 0x1 and EPENA = 0x1 in the OYG_DOEPCTLx register, and setting the XFERSIZE and PKTCNT bits in the OTGFS_DOEPTSIZx register.

1. The host attempts to send data (OUT token) to the endpoint

2. When the controller receives the OUT token on the USB, it stores data in the receive FIFO because the FIFO has free space.

3. Upon writing the complete data in the receive FIFO, the controller then triggers the RXFLVL interrupt bit in the OTGFS_GINTSTS register.

4. Upon receiving the packet count of USB packets, the controller internally sets the NAK bit for the endpoint to prevent it from receiving any more packets.

5. The application processes the interrupt and reads the data from the receive FIFO.

6. When the application reads all the data (equivalent to XFERSIZE), the controller generates an XFERC interrupt in the OTGFS_DOEPINTx register.

7. The application processes the interrupt and uses the XFERC bit in the OTGFS_DOEPINTx register to judge that the expected transfer is already complete.

21.5.4.16 Synchronous OUT data transfers

To initialize the controller after power-on reset, the application must perform the steps list in “OTGFS Initialization”. Before communicating with a host, the application must initialize endpoints based on the process described in “Endpoint Initialization” and by referring to “Read FIFO packets”. This section describes a regular synchronous OUT transfers.

[Application requirements]

1. All the application requirements are the same as that of non-synchronous OUT data transfers.

2. For synchronous OUT data transfers, the transfer size and packet count must be set to the number of the largest-packet-size packets that can be received in a single frame and not exceed this size. Synchronous OUT data transfer cannot span more than one frame.

- 1 ≤ packet count [epnum] ≤ 3

3. If the device supports the synchronous OUT endpoints, the application must read all synchronous OUT data packets from the receive FIFO before the end of the periodic frame (EOPF interrupt in the OTGFS_GINTSTS register)

4. To receive data in the subsequent frame, a synchronous OUT endpoint must be enabled before the generation of the EOPF and SOF interrupt in the OTGFS_GINTSTS register.

[Internal data flow]

1. The internal data flow for the synchronous OUT endpoints is the same as that for the non-synchronous OUT endpoints, just for a few differences.

2. When the synchronous OUT endpoint is enabled by setting the endpoint enable bit and by clearing the NAK bit, the even/odd frame bits are also set properly. The controller can receive data on a synchronous OUT endpoint in a particular frame only when the following condition is met:

- Even/Odd microframe (OTGFS_DOEPCTLx) =SOFFN[0] (OTGFS_DSTS)

3. When the application completely reads the synchronous OUT data packet (data and status) from the receive FIFO, the controller updates the RXDPID bit in the OTGFS_DOEPTSIZx register based on the data PID of the last synchronous OUT data packet read from the receive FIFO.

[Application programming sequence]

1. Program the transfer size and the corresponding packet count of the OTGFS_DOEPTSIZx register

2. Program the OTGFS_DOEPCTLx register with the endpoint enable, ClearNAK and Even/Odd frame bits

- Endpoint enable = 0x1
- CNAK = 0x1
- Even/Odd frame = (0x0: Even; 0x1: Odd)

3. Wait for the RXFLVL interrupt in the OTGFS_GINTSTS register, and read all the data packets from the receive FIFO. See “Read FIFO” for more information

- This step can be repeated several times, depending on the transfer size

4. When the XFERC interrupt is set in the OTGFS_DOEPINTx register, it indicates the completion of the synchronous OUT data transfers. But this interrupt does not necessarily mean that the data in memory are good.

5. This interrupt signal cannot always be detected by the synchronous OUT data transfers. However, the application can detect the INCOMPISOOUT interrupt in the OTGFS_GINTSTS register. See “Incomplete synchronous OUT data transfers” for more information.

6. Read the OTGFS_DOEPTSIZx register to determine the received transfer size and to determine whether the data received in the frame are valid or not. The application must treat the data received in memory as valid only when one of the following conditions is met:

- OTGFS_DOEPTSIZx.RxDPID = 0xD0 and the USB packet count in which the payload was received =0x1
- OTGFS_DOEPTSIZx.RxDPID = 0xD1 and the USB packet count in which the payload was received =0x2
- OTGFS_DOEPTSIZx.RxDPID = 0xD2 and the USB packet count in which the payload was received =0x3

The number of USB packets in which the payload was received= Application-programed initial packet count – Controller-updated final packet count

The application discards invalid data packets.

21.5.4.17 Enable synchronous endpoints

After sending a Set interface control command to the device, a host enables the synchronous endpoints. Then the host can send the initial synchronous IN token in any frame before transmission in the sequence of BInterval.

Instead, synchronous support in the OTGFS controller is based on a single-transfer level. The application must re-configure the controller on every frame. The OTGFS controller enables the synchronous

endpoint of the frame before the frame to be transmitted.

For example, to send data on the frame n, enable the endpoint of the frame n-1. Additionally, the OTGFS controller schedules the synchronous transfers by setting Even/Odd frame bits.

[Synchronous IN transfer interrupt]

The following interrupts must be processed to ensure successful scheduling of the synchronous transfers.

- XFERC interrupt in the OTGFS_DIEPINTx register (for endpoints)
- OTG INCOMPISOIN interrupt in the OTGFS_GINTSTS register (for global interrupts)

[Handling synchronous IN transfers]

The following steps must be performed to handle a synchronous IN transfer:

1. Unmask the incompISOOUT interrupt in the OTGFS_GINTSTS register by setting the INCOMISOINMSK interrupt bit in the OTGFS_GINTMSK register

2. Unmask the XFERC interrupt in the OTGFS_DIEPINTx register by setting the XFERCMSK bit in the OTGFS_DIEPMSK register

3. Enable synchronous endpoints with the following steps:

- Program the OTGFS_DIEPTSIZx register

  OTGFS_DIEPTSIZx.XFERSIZE= n * OTGFS_DIEPCTLx.MPS + sp, where 0 <= n <= 3 and 0 <= sp <OTGFS_DIEPCTLx.MPS. When the transfer size in a frame is less than that of the MPS bit in the OTGFS_DIEPCTLx register, n=0; When the transfer size in a frame is a multiple of that of the MPS bit in the OTGFS_DIEPCTLx register, sp=0.

  OTGFS_DIEPTSIZx.PKTCNT = 0x1

  The MC bit in the OTGFS_DIEPTSIZx register is set the same value as that of the PKTCNT bit in the OTGFS_DIEPTSIZx register.

- Program the OTGFS_DIEPCTLx register

  Read the OTGFS_DSTS register to determine the current frame number

  Program the OTGFS_DIEPCTLx with the maximum packet size (MPS bit)

  Set USBACTEP = 0x1 in the OTGFS_DIEPCTLx register

  Set EPTYPE = 0x1 in the OTGFS_DIEPCTLx register, marking synchronization

  Set the FIFO number of the endpoint through the TXFNUM bit in the OTGFS_DIEPCTLx register

  Set CNAK = 0x1 in the OTGFS_DIEPCTLx register

  If.SOFFN[0] = 0x0 in OTGFS_DSTS, then SETEVENFR = 0x1 in OTGFS_DIEPCTLx (otherwise, SETEVENFR = 0x1 in OTGFS_DIEPCTLx)

  If SOFFN[0] = 0x1 in OTGFS_DSTS, then SETODDFR = 0x1 in OTGFS_DIEPCTLx (otherwise, SETODDFR = 0x0 in OTGFS_DIEPCTLx)

  Set EPENA = 0x1 in OTGFS_DIEPCTLx

4. Write endpoint data to the corresponding transmit FIFO

For example, write address ranges are as follows:

- EP1 corresponding to 0x2000 - 0x2FFC
- EP2 corresponding to 0x3000 - 0x3FFC
- EP3 corresponding to 0x3000 - 0x3FFC
- ...

5. Wait for interrupts

- When an interrupt is generated (XFERC bit in OTGFS_DIEPINTx register), clear the XFERC interrupt; For the following transaction, repeat step 3-5 until the completion of data transfers.
- When an interrupt is generated (INCOMPISOIN bit in OTGFS_GINTSTS register), clear the INCOMPISOIN interrupt; For any synchronous IN endpoint, when Odd/Even bits match the current frame number bit 0, and when the endpoint remains enabled, the controller generates an interrupt at the end of the frame. This interrupt is generated on one of the following conditions:

  (1) There is no token in a frame

  (2) Late data write to the receive FIFO. An IN token has arrived before the completion of data write

  (3) IN token error

The INCOMPISOIN interrupt in the OTGFS_GINTSTS register is a global interrupt. Therefore, when more than one synchronous endpoints are in active state, the application must determine which one of the synchronous IN endpoints has not yet completed data transfers.

To achieve this, read the DSTS and DIEPCTLx bits of all synchronous endpoints. If the current endpoint has been enabled, and the read value of the SOFFN bit in the OTGFS_DSTS register is equal to the target frame number of the endpoint, it indicates that this endpoint has not finished data transfers. The application must keep track of and update the target frame number of the synchronous endpoint.

If data transfer is not yet complete on an endpoint, then Odd/Even bits have to be toggled.

Next:

(1) When the DPID is set to 1 (an odd frame) in the OTGFS_DIEPCTLx register, write 1 to the SETD0PID bit in the OTGFS_DIEPCTLx register makes it an even frame, then data transmission starts when there is an IN token input in the next frame.

(2) When the DPID is set to 0 in the OTGFS_DIEPCTLx register, write 1 to the SETD1PID bit in the OTGFS_DIEPCTLx register makes it an odd frame, then data transmission starts when there is an IN token input in the next frame.

21.5.4.18 Incomplete synchronous OUT data transfers

To initialize the controller after power-on reset, the application must perform the steps list in OTGFS Initialization. Before communicating with a host, the controller must follow the steps defined in Endpoint Initialization to initialize endpoints. This section describes the application programming sequence when the controller drops synchronous OUT data packets.

[Internal data flow]

1. For synchronous OUT endpoints, the XFERC interrupt (in the OTGFS_DOEPINTx register) may not always be generated. If the controller drops synchronous OUT data packets, the application may fail to detect the XFERC interrupt in the OTGFS_DOEPINTx register.

- When the receive FIFO cannot accommodate the complete ISO OUT data packet, the controller drops the received ISO OUT data.
- When the synchronous OUT data packet is received with CRC errors.
- When the synchronous OUT token received by the controller is corrupted.
- When the application is very slow in reading the receive FIFO

2. When the controller detects the end of periodic frames before transfer complete to all synchronous OUT endpoints, an interrupt of incomplete synchronous OUT data is generated, indicating that an XFERC interrupt in the OTGFS_DOEPINTx register is not set on at least one of the synchronous OUT endpoints. At this point, the endpoint with the incomplete data transfer remains enabled, but no valid transfers are in progress on this endpoint.

[Application programming sequence]

1. The assertion of the incomplete synchronous OUT data interrupt indicates that at least one synchronous OUT endpoint has an incomplete data transfer in the current frame.

2. If this occurs because the synchronous OUT data is not completely read out from the endpoint, the application must empty all synchronous OUT data (data and status) in the receive FIFO before proceeding.

- When all data are read from the receive FIFO, the application can detect the XFERC interrupt in the OTGFS_DOEPINTx register. In this case, the application must re-enable the endpoint to receive the synchronous OUT data in the next frame by following the steps listed in “SETUP/Data IN/Status OUT”

3. When it receives an incomplete synchronous OUT data interrupt, the application must read the control registers of all synchronous OUT endpoints to determine which one of the endpoints has an incomplete data transfer in the current frame. An endpoint transfer is regarded as incomplete if both of the following conditions are met:

- OTGFS_DOEPCTLx. Even/Odd frame bit= OTGFS_DSTS.SOFFN[0]
- OTGFS_DOEPCTLx. Endpoint enable = 0x1

4. The pervious step must be performed before the SOF interrupt of the GINTSTS register is detected to ensure that the current frame number is not changed.

5. For synchronous OUT endpoints with incomplete transfers, the application must drop the data in memory, and disable the endpoint through the endpoint disable bit in the OTGFS_DOEPCTLx register.

6. Wait for the endpoint disable interrupt in the OTGFS_DOEPINTx register, and enable the endpoint to receive new data in the next frame by following the steps listed in “SETUP/Data IN/Status OUT”. Because the controller can take some time to disable the endpoint, the application may not be able to receive the data in the next frame after receiving wrong synchronous data.

21.5.4.19 Incomplete synchronous IN data transfers

This section describes how the application behaves on incomplete synchronous IN transfers.

[Internal data flow]

1. Synchronous IN transfers are incomplete on one of the following conditions:

- The controller receives corrupted synchronous IN tokens from more than one synchronous IN endpoints. In this case, the application can detect the incomplete synchronous IN transfer interrupt in the GINTSTS register.
- The application is slow in writing complete data to the transmit FIFO, and an IN token is received before the completion of data write. In this case, the application can detect the INTKNTXFEMP interrupt in the OTGFS_DIEPINTx register. The application ignores this interrupt, which will result in the generation of the incomplete synchronous IN transfer interrupt (in OTGFS_GINTSTS register). The controller responds to the received IN token by sending a zero-length data packet to the USB.

2. Either way, the application must stop writing the transmit FIFO as soon as possible.

3. The application must set the NAK and disable bits of the endpoints.

4. The controller disables the endpoint, clears the disable bit, and triggers the endpoint disable interrupt.

[Application programming sequence]

1. When the transmit FIFO becomes empty, the application ignores the INTKNTXFEMP interrupt (in the OTGFS_DIEPINTx register) from any synchronous IN endpoint because this can trigger the incomplete synchronous IN interrupt.

2. The incomplete synchronous IN transfer interrupt (in the OTGFS_GINTSTS register) indicates that at least one synchronous IN endpoint is with incomplete synchronous IN transfers.

3. The application must read the endpoint control registers of all synchronous IN endpoints to determine which one is with incomplete synchronous IN transfers.

4. The application must write data to the periodic transmit FIFO of the endpoint.

5. Disable theses endpoints by setting the following bits in the OTGFS_DIEPCTLx register

- OTGFS_DIEPCTLx.SETNAK = 0x1
- OTGFS_DIEPCTLx. endpoint enable = 0x1

6. The endpoint disable interrupt in the DIEPINTx register indicates that the controller has disabled the endpoint.

7. At this point, the application must empty the data in the associated transmit FIFO or overwrite the existing data in the FIFO by enabling the endpoint for a new transfer in the next frame. The application must refresh the data through the OTGFS_GRSTCTL register.

21.5.4.20 Periodic IN (interrupt and synchronous) data transfers

This section describes a typical periodic IN data transfer.

To initialize the controller after power-on reset, the application must perform the steps list in OTGFS Initialization. Before communicating with a host, the controller must follow the steps defined in Endpoint Initialization to initialize endpoints.

[Application requirements]

1. Application requirements in “Non-periodic (bulk and control) IN data transfers” also apply to periodic IN data transfers, except for a slight difference of requirement 2.

- The application can only transmit multiples of largest-packet-size data packets, and a short packet. To transmit several largest-packet-size data packets and a short packet, the following conditions must be met:

Transfer size [epnum] = n * mps[epnum] + sp (where n and i are integers ≥ 0, and 0 ≤ sp < mps[epnum])

If (sp > 0), packet count [epnum] = n + 1. Otherwise, packet count [epnum] = n, mc[epnum] = packet count [epnum]

- The application cannot transmit a zero-length data packet at the end of a transfer. But it can transmit a single zero-length data packet in itself, provided packet count [epnum] = 1, mc[epnum] = packet count [epnum]

2. The application can only schedule data transfers of one frame at a time
- (OTGFS_DIEPTSIZx.MC – 1) * OTGFS_DIEPCTLx.MPS ≤OTGFS_DIEPTSIZx.XFERSIZ ≤OTGFS_DIEPTSIZx.MC * OTGFS_DIEPCTLx.MPS
- OTGFS_DIEPTSIZx.PKTCNT = OTGFS_DIEPTSIZx.MC
- If OTGFS_DIEPTSIZx.XFERSIZ < OTGFS_DIEPTSIZx.MC * OTGFS_DIEPCTLx.MPS, the last data packet of the transfer is a short packet.

3. For periodic IN endpoints, one-frame data must be prefetched before the data transfer in the next frame. This can be done by enabling periodic IN endpoint 1 frame before the scheduling of the frame to be transmitted.

4. The complete data to be transmitted in a frame must be written to the transmit FIFO by the application before the periodic IN token is received. Even when one-WORD data to be transmitted per frame is missing in the transmit FIFO while the periodic IN token is received, the controller behaves as when the FIFO is empty. When the transmit FIFO is empty, a zero-length data packet would be transmitted on the USB, and An NAK handshake signal would be transmitted for INTR IN endpoints.

[Internal data flow]

1. The application must set the transfer size and packet count bits of the endpoint registers, and enable the endpoint to transmit the data.

2. The application must also write the required data to the associated transmit FIFO.

3. Each time the application writes a packet to the transmit FIFO, the transfer size for the endpoint is decremented by the packet size. Continue to write data until the transfer size for the endpoint becomes 0

4. When an IN token for a periodic endpoint is received, the application writes the data to the FIFO (If any). If the complete data for the frame is not present in the FIFO, the controller generates an INTKNTXFEMP interrupt.
- A zero-length data packet is transmitted on the USB for synchronous IN endpoints
- An NAK handshake signal is transmitted on the USB for interrupt IN endpoints.

5. The packet count for the endpoints is decremented by one under the following conditions:
- For synchronous endpoints, when a zero-or non-zero-length data packet is transmitted
- For interrupt endpoints, when an ACK handshake is transmitted
- When the transfer size and packet count are both 0, the transfer complete interrupt for the endpoint is generated and the endpoint enable bit is cleared.

6. In the “Periodic frame interval” (by the PERFRINT bit in the OTGFS_DCFG register), when the controller finds non-empty any one of the IN endpoint FIFOs scheduled for the current frame non-empty, the controller generates an INCOMPISOIN interrupt in the OTGFS_GINTSTS register.

[Application programming sequence (frame transfers)]

1. Program the OTGFS_DIEPTSIZx register

2. Program the OTGFS_DIEPCTLx register based on endpoint characteristics, and set the CNAK and endpoint enable bits

3. Write the data to be transmitted into the transmit FIFO.

4. The assertion of the INTKNTXFEMP interrupt indicates that the application has not yet written all data to be transferred into the transmit FIFO.

5. If the interrupt endpoint is already enabled while this interrupt is detected, ignore the interrupt. If it is not enabled, enable the endpoint to transmit data on the next IN token. If it is enabled while the interrupt is detected, refer to “Incomplete synchronous IN data transfers”.

6. When the interrupt IN endpoint is set as a periodic endpoint, the controller internally can process the timeout on the interrupt IN endpoint, without the need of the application intervention. Therefore, the application can never detect the TIMEOUT interrupt (in the OTGFS_DIEPINTx register) on the periodic interrupt IN endpoints.

7. The assertion of the XFERC interrupt in the OTGFS_DIEPINTx register but without the INTKNTXFEMP interrupt indicates the successful completion of a synchronous IN transfer. When reading the OTGFS_DIEPTSIZx register, only transfer size =0 and packet count =0 indicate that all data

are transmitted on the USB line.

8. The assertion of the XFERC interrupt in the OTGFS_DIEPINTx register, with or without the INTKNTXFEMP interrupt, indicates the successful completion of an interrupt IN transfer. When reading the OTGFS_DIEPTSIZx register, only transfer size =0 and packet count =0 indicate that all data are transmitted on the USB line.

9. The assertion of the INCOMPISOIN interrupt but without the above-mentioned interrupts indicates that the controller did not receive at least one periodic IN token in the current frame. Refer to “Incomplete synchronous IN data transfers” for more information on synchronous IN endpoints.

21.6 OTGFS control and status registers

The application controls the OTGFS controller by reading from and writing to the control and status registers (CSRx) through the AHB slave interface. These registers are accessible by 32 bits, and the addresses are 32-bit aligned.

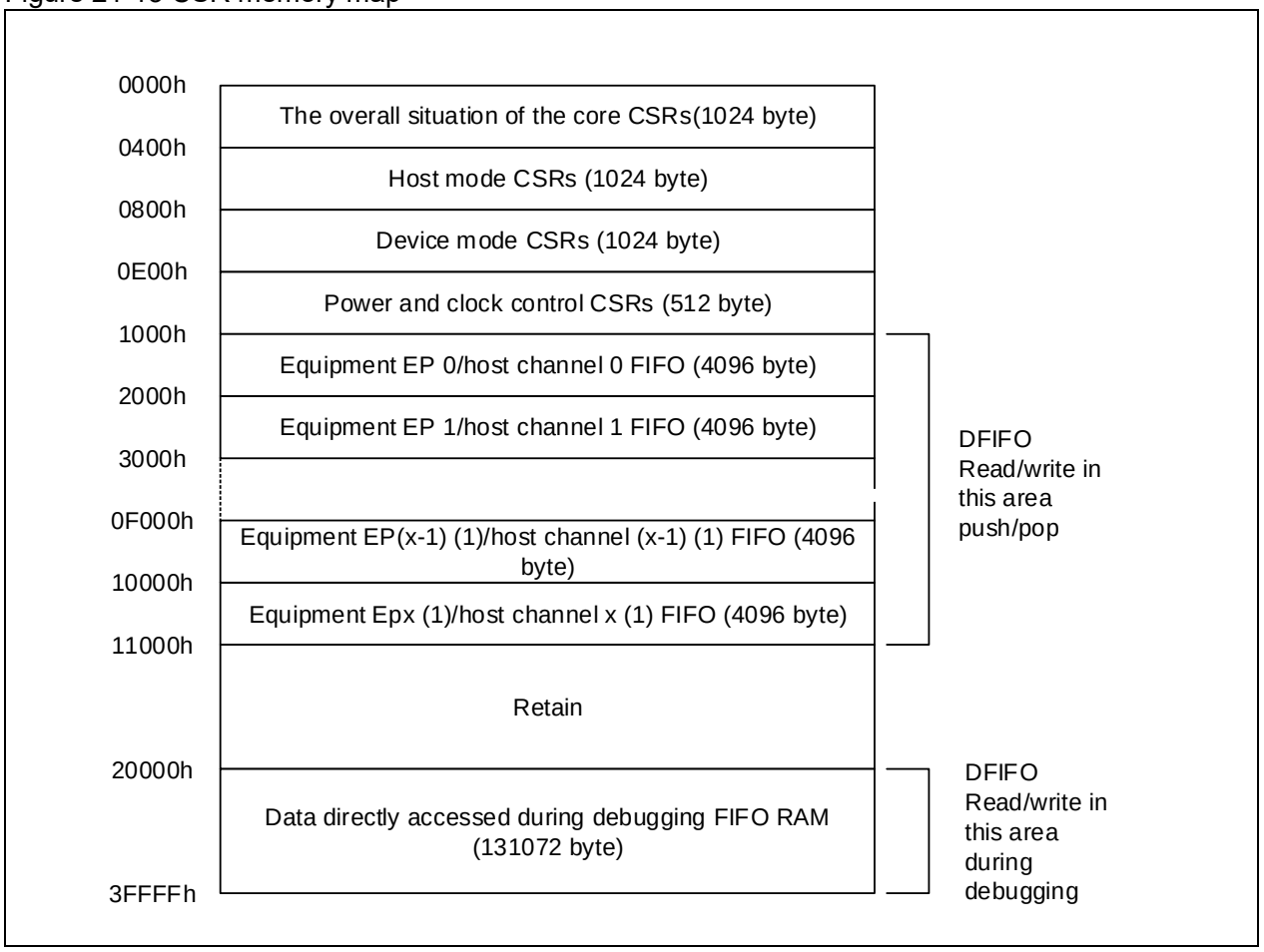
Only the controller global, power and clock control, data FIFO access and host port control and status registers are active in both host and device modes. When the OTGFS controller operates in either host or device mode, the application must not access the register group from the other mode. If an illegal access occurs, a mode mismatch interrupt is generated and the MODMIS bit (in the OTGFS_GINTSTS register) is affected.

When the controller switches from one mode to the other, the registers in the new mode must be re-initialized as they are after a power-on reset. These peripheral registers must be accessed by words (32-bit)

21.6.1 CSR register map

The host and device mode registers occupy different addresses. All registers are located in the AHB clock domain

Figure 21-13 CSR memory map



x = 7 in device mode, x =15 in host mode.

The OTGFS control and status registers contain OTGFS global register, host mode register, device mode register, data FIFO register, power and clock control register.

1. OTGFS global registers: They are active in both host and device modes. The register acronym is G.
2. Host-mode registers: They must be programmed every time the controller changes to host mode, The register acronym is H.
3. Device-mode registers: They must be programmed every time the controller changes to device mode, The register acronym is D.
4. Data FIFO access registers: These registers are valid in both in host and device modes, and are used to read or write the FIFO for a specific endpoint or channel in a given direction. If a host channel is of type IN, the FIFO can only be read. Similarly, if a host channel is of type OUT, the FIFO can only be written.
5. Power and clock control register: There is only one register for power and clock control. It is valid in both host and device modes.

21.6.2 OTGFS register address map

Table 21-4 shows the USB OTG register map and their reset values.

These peripheral registers must be accessed by words (32-bit)

Table 21-4 OTGFS register map and reset values

Register name	Offset	Reset value
OTGFS_GOTGCTL	0x000	0x0001 0000
OTGFS_GOTGINT	0x004	0x0000 0000
OTGFS_GAHBCFG	0x008	0x0000 0000
OTGFS_GUSBCFG	0x00C	0x0000 1400
OTGFS_GRSTCTL	0x010	0x2000 0000
OTGFS_GINTSTS	0x014	0x0400 0020
OTGFS_GINTMSK	0x018	0x0000 0000
OTGFS_GRXSTSR	0x01C	0x0000 0000
OTGFS_GRXSTSP	0x020	0x0000 0000
OTGFS_GRXFSIZ	0x024	0x0000 0200
OTGFS_GNPTXFSIZ/ OTGFS_DIEPTXF0	0x028	0x0200 0200
OTGFS_GNPTXSTS	0x02C	0x0008 0200
OTGFS_GCCFG	0x038	0x0000 0000
OTGFS_GUID	0x03C	0x0000 1000
OTGFS_HPTXFSIZ	0x100	0x0200 0400
OTGFS_DIEPTXF1	0x104	0x0000 0000
OTGFS_DIEPTXF2	0x108	0x0000 0000
OTGFS_DIEPTXF3	0x10C	0x0000 0000
OTGFS_DIEPTXF4	0x110	0x0000 0000
OTGFS_DIEPTXF5	0x114	0x0000 0000
OTGFS_DIEPTXF6	0x118	0x0000 0000
OTGFS_DIEPTXF7	0x11C	0x0000 0000
OTGFS_HCFG	0x400	0x0000 0000
OTGFS_HFIR	0x404	0x0000 EA60
OTGFS_HFNUM	0x408	0x0000 3FFF

OTGFS_HPTXSTS	0x410	0x0008 0100
OTGFS_HAINT	0x414	0x0000 0000
OTGFS_HAINTMSK	0x418	0x0000 0000
OTGFS_HPRT	0x440	0x0000 0000
OTGFS_HCCHAR0	0x500	0x0000 0000
OTGFS_HCINT0	0x508	0x0000 0000
OTGFS_HCINTMSK0	0x50C	0x0000 0000
OTGFS_HCTSIZ0	0x510	0x0000 0000
OTGFS_HCCHAR1	0x520	0x0000 0000
OTGFS_HCINT1	0x528	0x0000 0000
OTGFS_HCINTMSK1	0x52C	0x0000 0000
OTGFS_HCTSIZ1	0x530	0x0000 0000
OTGFS_HCCHAR2	0x540	0x0000 0000
OTGFS_HCINT2	0x548	0x0000 0000
OTGFS_HCINTMSK2	0x54C	0x0000 0000
OTGFS_HCTSIZ2	0x550	0x0000 0000
OTGFS_HCCHAR3	0x560	0x0000 0000
OTGFS_HCINT3	0x568	0x0000 0000
OTGFS_HCINTMSK3	0x56C	0x0000 0000
OTGFS_HCTSIZ3	0x570	0x0000 0000
OTGFS_HCCHAR4	0x580	0x0000 0000
OTGFS_HCINT4	0x588	0x0000 0000
OTGFS_HCINTMSK4	0x58C	0x0000 0000
OTGFS_HCTSIZ4	0x590	0x0000 0000
OTGFS_HCCHAR5	0x5A0	0x0000 0000
OTGFS_HCINT5	0x5A8	0x0000 0000
OTGFS_HCINTMSK5	0x5AC	0x0000 0000
OTGFS_HCTSIZ5	0x5B0	0x0000 0000
OTGFS_HCCHAR6	0x5C0	0x0000 0000
OTGFS_HCINT6	0x5C8	0x0000 0000
OTGFS_HCINTMSK6	0x5CC	0x0000 0000
OTGFS_HCTSIZ6	0x5D0	0x0000 0000
OTGFS_HCCHAR7	0x5E0	0x0000 0000
OTGFS_HCINT7	0x5E8	0x0000 0000
OTGFS_HCINTMSK7	0x5EC	0x0000 0000
OTGFS_HCTSIZ7	0x5F0	0x0000 0000
OTGFS_HCCHAR8	0x600	0x0000 0000
OTGFS_HCINT8	0x608	0x0000 0000
OTGFS_HCINTMSK8	0x60C	0x0000 0000
OTGFS_HCTSIZ8	0x610	0x0000 0000
OTGFS_HCCHAR9	0x620	0x0000 0000

OTGFS_HCINT9	0x628	0x0000 0000
OTGFS_HCINTMSK9	0x62C	0x0000 0000
OTGFS_HCTSIZ9	0x630	0x0000 0000
OTGFS_HCCHAR10	0x640	0x0000 0000
OTGFS_HCINT10	0x648	0x0000 0000
OTGFS_HCINTMSK10	0x64C	0x0000 0000
OTGFS_HCTSIZ10	0x650	0x0000 0000
OTGFS_HCCHAR11	0x660	0x0000 0000
OTGFS_HCINT11	0x668	0x0000 0000
OTGFS_HCINTMSK11	0x66C	0x0000 0000
OTGFS_HCTSIZ11	0x670	0x0000 0000
OTGFS_HCCHAR12	0x680	0x0000 0000
OTGFS_HCINT12	0x688	0x0000 0000
OTGFS_HCINTMSK12	0x68C	0x0000 0000
OTGFS_HCTSIZ12	0x690	0x0000 0000
OTGFS_HCCHAR13	0x6A0	0x0000 0000
OTGFS_HCINT13	0x6A8	0x0000 0000
OTGFS_HCINTMSK13	0x6AC	0x0000 0000
OTGFS_HCTSIZ13	0x6B0	0x0000 0000
OTGFS_HCCHAR14	0x6C0	0x0000 0000
OTGFS_HCINT14	0x6C8	0x0000 0000
OTGFS_HCINTMSK14	0x6CC	0x0000 0000
OTGFS_HCTSIZ14	0x6D0	0x0000 0000
OTGFS_HCCHAR15	0x6E0	0x0000 0000
OTGFS_HCINT15	0x6E8	0x0000 0000
OTGFS_HCINTMSK15	0x6EC	0x0000 0000
OTGFS_HCTSIZ15	0x6F0	0x0000 0000
OTGFS_DCFG	0x800	0x0220 0000
OTGFS_DCTL	0x804	0x0000 0002
OTGFS_DSTS	0x808	0x0000 0010
OTGFS_DIEPMSK	0x810	0x0000 0000
OTGFS_DOEPMSK	0x814	0x0000 0000
OTGFS_DAINT	0x818	0x0000 0000
OTGFS_DAINTMSK	0x81C	0x0000 0000
OTGFS_DIEPEMPMSK	0x834	0x0000 0000
OTGFS_DIEPCTL0	0x900	0x0000 0000
OTGFS_DIEPINT0	0x908	0x0000 0080
OTGFS_DIEPTSIZ0	0x910	0x0000 0000
OTGFS_DTXFSTS0	0x918	0x0000 0200
OTGFS_DIEPCTL1	0x920	0x0000 0000
OTGFS_DIEPINT1	0x928	0x0000 0080

OTGFS_DIEPTSIZ1	0x930	0x0000 0000
OTGFS_DTXFSTS1	0x938	0x0000 0200
OTGFS_DIEPCTL2	0x940	0x0000 0000
OTGFS_DIEPINT2	0x948	0x0000 0080
OTGFS_DIEPTSIZ2	0x950	0x0000 0000
OTGFS_DTXFSTS2	0x958	0x0000 0200
OTGFS_DIEPCTL3	0x960	0x0000 0000
OTGFS_DIEPINT3	0x968	0x0000 0080
OTGFS_DIEPTSIZ3	0x970	0x0000 0000
OTGFS_DTXFSTS3	0x978	0x0000 0200
OTGFS_DIEPCTL4	0x980	0x0000 0000
OTGFS_DIEPINT4	0x988	0x0000 0080
OTGFS_DIEPTSIZ4	0x990	0x0000 0000
OTGFS_DTXFSTS4	0x998	0x0000 0200
OTGFS_DIEPCTL5	0x9A0	0x0000 0000
OTGFS_DIEPINT5	0x9A8	0x0000 0080
OTGFS_DIEPTSIZ5	0x9B0	0x0000 0000
OTGFS_DTXFSTS5	0x9B8	0x0000 0200
OTGFS_DIEPCTL6	0x9C0	0x0000 0000
OTGFS_DIEPINT6	0x9C8	0x0000 0080
OTGFS_DIEPTSIZ6	0x9D0	0x0000 0000
OTGFS_DTXFSTS6	0x9D8	0x0000 0200
OTGFS_DIEPCTL7	0x9E0	0x0000 0000
OTGFS_DIEPINT7	0x9E8	0x0000 0080
OTGFS_DIEPTSIZ7	0x9F0	0x0000 0000
OTGFS_DTXFSTS7	0x9F8	0x0000 0200
OTGFS_DOEPCTL0	0xB00	0x0000 8000
OTGFS_DOEPINT0	0xB08	0x0000 0080
OTGFS_DOEPTSIZ0	0xB10	0x0000 0000
OTGFS_DOEPCTL1	0xB20	0x0000 0000
OTGFS_DOEPINT1	0xB28	0x0000 0080
OTGFS_DOEPTSIZ1	0xB30	0x0000 0000
OTGFS_DOEPCTL2	0xB40	0x0000 0000
OTGFS_DOEPINT2	0xB48	0x0000 0080
OTGFS_DOEPTSIZ2	0xB50	0x0000 0000
OTGFS_DOEPCTL3	0xB60	0x0000 0000
OTGFS_DOEPINT3	0xB68	0x0000 0080
OTGFS_DOEPTSIZ3	0xB70	0x0000 0000
OTGFS_DOEPCTL4	0xB80	0x0000 0000
OTGFS_DOEPINT4	0xB88	0x0000 0080
OTGFS_DOEPTSIZ4	0xB90	0x0000 0000

OTGFS_DOEPCTL5	0xBA0	0x0000 0000
OTGFS_DOEPINT5	0xBA8	0x0000 0080
OTGFS_DOEPTSIZ5	0xBB0	0x0000 0000
OTGFS_DOEPCTL6	0xBC0	0x0000 0000
OTGFS_DOEPINT6	0xBC8	0x0000 0080
OTGFS_DOEPTSIZ6	0xBD0	0x0000 0000
OTGFS_DOEPCTL7	0xBE0	0x0000 0000
OTGFS_DOEPINT7	0xBE8	0x0000 0080
OTGFS_DOEPTSIZ7	0xBF0	0x0000 0000
OTGFS_PCGCCTL	0xE00	0x0000 0000

21.6.3 OTGFS global registers

These registers are available in both host and device modes, and do not need to be reprogrammed when switching between two modes.

21.6.3.1 OTGFS status and control register (OTGFS_GOTGCTL)

This register controls the OTG function and reflects its status.

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x0000	resd	Kept at its default value.
Bit 21	CURMOD	0x0	ro	Current Mode of Operation Accessible in both host and device modes This bit indicates the current operation mode. 0: Device mode 1: Host mode
Bit 20: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	CONIDSTS	0x1	ro	Accessible in both host and device modes Connector ID status This bit indicates the connecter ID status. 0: OTGFS controller is in A-device mode 1: OTGFS controller is in B-device mode
Bit 15: 0	Reserved	0x0000	resd	Kept at its default value.

21.6.3.2 OTGFS interrupt status control register (OTGFS_GOTGINT)

The application reads this register to know about which kind of OTG interrupt is generated, and writes this register to clear the OTG interrupt.

Bit	Register	Reset value	Type	Description
Bit 31: 3	Reserved	0x0000	resd	Kept at its default value.
Bit 2	SESENDDET	0x0	rw1c	Available in both host and device modes Session end detected The controller sets this bit when a Bvalid (Vbus) signal is disconnected. This register can only be set by hardware. Writing 1 by software clears this bit.
Bit 1: 0	Reserved	0x0000	resd	Kept at its default value.

21.6.3.3 OTGFS AHB configuration register (OTGFS_GAHBCFG)

This register is used to configure the controller after power-on or mode change. This register mainly contains AHB-related parameters. Do not change this register after the initial configuration. The application must configure this register before starting transmission on either the AHB or USB.

Bit	Register	Reset value	Type	Description
Bit 31: 9	Reserved	0x000000	resd	Kept at its default value.
Bit 8	PTXFEMPLVL	0x0	rw	Accessible in host mode only Periodic TxFIFO empty level It indicates when the periodic TxFIFO empty interrupt bit in the GINTSTS register is triggered. 0: PTXFEMP (GINTSTS) interrupt indicates that the periodic TxFIFO is half empty

				1: PTXFEMP (GINTSTS) interrupt indicates that the periodic TxFIFO is fully empty
Bit 7	NPTXFEMPLVL	0x0	rw	Accessible in both host mode and device modes Non-Periodic TxFIFO empty level In host mode, this bit indicates when the non-periodic TxFIFO empty interrupt (NPTXFEMP in GINTSTS) is triggered. In device mode, this bit indicates when the IN endpoint TxFIFO empty interrupt (TXFEMP bit in DIEPINTn) is triggered. 0: The TxFEMP (in DIEPINTn) interrupt indicates that the IN endpoint TxFIFO is half empty 1: The TxFEMP (in DIEPINTn) interrupt indicates that the IN endpoint TxFIFO is fully empty
Bit 6: 1	Reserved	0x00	resd	Kept at its default value.
Bit 0	GLBINTMSK	0x0	rw	Accessible in both host mode and device modes Global interrupt mask The application uses this bit to mask or unmask the interrupts sent by the interrupt line to itself. 0: Mask the interrupts sent to the application 1: Unmask the interrupts sent to the application

21.6.3.4 OTGFS USB configuration register (OTGFS_GUSBCFG)

This register is used to configure the controller after power-on or a change between host mode and device mode. This register contains USB and USB-PHY related parameters. The application must program the register before handling any transaction on either the AHB or USB. Do not change this register after the initial configuration.

Bit	Register	Reset value	Type	Description
Bit 31	COTXPKT	0x0	rw	Accessible in both host mode and device modes Corrupt Tx packet This bit is for debug purpose only. Do not set this bit to 1.
Bit 30	FDEVMODE	0x0	rw	Accessible in both host mode and device modes Force device mode Writing 1 to this bit forces the controller to go into device mode, irrespective of the status of the ID input point. 0: Normal mode 1: Force device mode After setting this bit, the application must wait at least 25ms before the configuration takes effect.
Bit 29	FHSTMODE	0x0	rw	Accessible in both host mode and device modes Force host mode Writing 1 to this bit forces the controller to go into host mode, irrespective of the status of the ID input point. 0: Normal mode 1: Force host mode After setting this bit, the application must wait at least 25ms before the configuration takes effect.
Bit 28: 15	Reserved	0x0000	resd	Kept at its default value.
Bit 14	Reserved	0x0	resd	Kept at its default value.
Bit 13: 10	USBTRDTIM	0x5	rw	Accessible in device mode USB Turnaround Time This field sets the turnaround time in PHY clocks. It defines the response time when the MAC sends a request to the packet FIFO controller (PFC) to fetch data from the DFIFO (SPRAM). These bits must be configured as follows: 0101: When the MAC interface is 16-bit UTMI+ 1001: When the MAC interface is 8-bit UTMI+ Note: The aforementioned values are calculated based on a minimum of 30MHz AHB frequency. The USB turnaround time is critical for certifications with long cables and 5-Hub. If you want the AHB to run below 30 MHz, and don't care about the USB turnaround time, you can set larger values for these bits.

Bit 9: 3	Reserved	0x00	resd	Kept at its default value.
Bit 2: 0	TOUTCAL	0x0	rw	Accessible in both host mode and device modes FS Timeout calibration The number of PHY clocks that the application programs in these bits is added to the full-speed interpacket timeout duration in order to compensate for any additional latency introduced by the PHY. This action can be required, because the delay triggered by the PHY while generating the line state condition can vary from one PHY to another. In full-speed mode, the USB standard timeout value is 16~18 (inclusive) bit times. The application must program these bits based on the enumeration speed. The number of bit times added per PHY clock is 0.25 bit times.

21.6.3.5 OTGFS reset register (OTGFS_GRSTCTL)

The application resets various hardware modules in the controller through this register.

Bit	Register	Reset value	Type	Description
Bit 31	AHBIDLE	0x1	ro	Accessible in both host mode and device modes AHB master Idle This bit indicates that the AHB master state machine is in idle condition.
Bit 30: 11	Reserved	0x000	resd	Kept at its default value.
Bit 10: 6	TXFNUM	0x00	rw	Accessible in both host mode and device modes TxFIFO number This field indicates the FIFO number that must be refreshed through the TxFIFO Flush bit. Do not make changes to this field until the controller clears the TxFIFO Flush bit. 00000: - Non-periodic TxFIFO in host mode - Tx FIFO 0 in device mode 00001: - Periodic TxFIFO in host mode - TXFIFO 1 in device mode 00010: - TXFIFO 2 in device mode ... 01111: - TXFIFO 15 in device mode 10000: - Refresh all the transmit FIFOs in device or host mode
Bit 5	TXFFLSH	0x0	rw1s	Accessible in both host mode and device modes TxFIFO Flush This bit selectively refreshes a single or all transmit FIFOs, but can do so when the controller is not in the process of a transaction. The application must write this bit only after checking that the controller is neither writing to nor reading from the TxFIFO. Verify using these registers: Read: NAK effective interrupt (NAK Effective Interrupt) ensures that the controller is not reading from the FIFO Write: AHBIDLE bit in GRSTCTL ensures that the controller is not writing to the FIFO. For FIFO reprogramming, it is usually recommended to carry out flushing operation. In device endpoint disable state, it is also advised to use FIFO flushing operation. The application must wait until the controller clears this bit, before performing other operations. It takes 8 clocks to clear this bit (slowest of phy_clk or hclk)
Bit 4	RXFFLSH	0x0	rw1s	Accessible in both host mode and device modes RxFIFO flush The application can refresh the entire RxFIFO using this

				bit, but must first ensure that the controller is not in the process of a transaction. The application must only write to this bit after checking that the controller is neither reading from nor writing to the RxFIFO. The application must wait until the controller clears this bit, before performing other operations. It takes 8 clocks to clear this bit (slowest of PHY or AHB)
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2	FRMCNTRST	0x0	rw1s	Accessible in both host mode and device modes Host frame counter reset The application uses this bit to reset the frame number counter inside the controller. After the frame counter is reset, the subsequent SOS sent out by the controller has a frame number of 0. If the application writes 1 to this bit, it may not be able to read the value, because this bit is cleared after a few clock cycles by the controller
Bit 1	PIUSFTRST	0x0	rw1s	Accessible in both host mode and device modes PIU FS dedicated controller soft reset This bit is used to reset PIU full-speed dedicated controller All state machines in the PIU full-speed dedicated controller are reset to the idle state. When the PHY remains in the receive state for more than one-frame time due to PHY errors (such as operation interrupted or babble), this bit can be used to reset the PIU full-speed dedicated controller. This is can be cleared automatically, the controller this clear this bit after all the necessary logic is reset in the controller.
Bit 0	CSFTRST	0x0	rw1s	Accessible in both host mode and device modes Controller soft reset Resets the hclk and phy_clock domain as follows: Clears all interrupts and CSR registers except for the following bits: - HCFG.FSLSPCS - DCFG.DECSPD - DCTL.SFTDIS Resets all state machines (except AHB slave) to the idle state, and clears all the transmit and receive FIFOs. All transactions on the AHB master are terminated as soon as possible after completing the last phase of an AHB data transfer. All transactions on the USB are terminated immediately. The application can write to this bit at any time to reset the controller. This is can be cleared automatically, the controller this clear this bit after all the necessary logic is reset in the controller. The controller could take several clocks to clear this bit, depending on the current state of the controller. Once this bit is cleared, the application must wait at least 3 PHY clocks before accessing the PHY domain (synchronization delay). Additionally, the application must ensure that the bit 31 in this register is set (AHB master is in idle state) before performing other operations. Typically, the software set is used during software development and also when the user dynamically changes the PHY selection bits in the above-listed USB configuration registers. To change the PHY, the corresponding PHY clock is selected and used in the PHY domain. After a new clock is selected, the PHY domain has to be reset for normal operation.

21.6.3.6 OTGFS interrupt register (OTGFS_GINTSTS)

This register interrupts the application due to system-level events in the current mode (device or host mode), as shown in Figure 21-2.

Some of the bits in this register are valid only in host mode, while others are valid in device mode only. Besides, this register indicates the current mode.

The FIFO status interrupts are read-only. The FIFO interrupt conditions are cleared automatically as soon as the software reads from or writes to the FIFO while processing these interrupts.

The application must clear the GINTSTS register at initialization before enabling an interrupt bit to avoid any interrupt generation prior to initialization.

Bit	Register	Reset value	Type	Description
Bit 31	WKUPINT	0x0	rw1c	Accessible in both host mode and device modes Resume/Remote wakeup detected interrupt) In device mode, this interrupt is generated only when a resume signal (triggered by host) is detected on the USB bus. In host mode, this interrupt is generated only when a remote wakeup signal (triggered by device) is detected on the USB bus.
Bit 30	Reserved	0x0	resd	Kept at its default value.
Bit 29	DISCONINT	0x0	rw1c	Accessible in host mode only Disconnect detected interrupt The interrupt is generated when a device disconnect is detected.
Bit 28	CONIDSCHG	0x0	rw1c	Accessible in both host mode and device modes Connector ID status change This bit is set by the controller when there is a change in connector ID status.
Bit 27	Reserved	0x0	resd	Kept at its default value.
Bit 26	PTXFEMP	0x1	ro	Accessible in host mode only Periodic TxFIFO Empty The interrupt is generated when the Periodic Transmit FIFO is either half or completely empty and there is space for a request to be written in the periodic request queue. The half or completely empty status depends on the periodic transmit FIFO empty level bit in the AHB configuration register.
Bit 25	HCHINT	0x0	ro	Host channel interrupt The controller sets this bit to indicate that an interrupt is pending on one of the channels in the controller (in host mode). The application must read the Host All Channels Interrupt register to determine the exact number of the channel on which the interrupt occurred, and then read the Host Channel-n Interrupt register to determine the interrupt event source. The application must clear the corresponding status bit in the HCINTn (Host All Channels Interrupt) register to clear this bit.
Bit 24	PRTINT	0x0	ro	Host port interrupt The controller sets this bit to indicate a change in port status one of the ports. The application must read the Host Port Control and Status register to determine the exact event source. The application must clear the Host Port Control and Status register to clear this bit.
Bit 23: 22	Reserved	0x0	resd	Kept at its default value.
Bit 21	INCOMPIP INCOMPISOOUT	0x0	rw1c	Incomplete periodic transfer Accessible in host mode only In host mode, the controller sets this interrupt bit when there are incomplete periodic transfers still pending in the current frame. Incomplete Isochronous OUT Transfer Accessible in device mode only In device mode, the controller sets this interrupt bit to

				indicate that there is at least one synchronous OUT endpoint with incomplete transfers in the current frame. This interrupt is generated along with the End of Periodic Frame Interrupt bit in this register.
Bit 20	INCOMPISOIN	0x0	rw1c	Accessible in device mode only Incomplete Isochronous IN Transfer The controller sets this interrupt to indicate that there is at least one synchronous IN endpoint with incomplete transfers in the current frame. This interrupt is generated along with the End of Periodic Frame Interrupt bit in this register.
Bit 19	OEPTINT	0x0	ro	Accessible in device mode only OUT endpoints interrupt The controller sets this bit to indicate that an interrupt is pending on one of the OUT endpoints in the controller. The application must read the Device All Endpoints Interrupt register to determine the exact number of the OUT endpoint on which the interrupt occurred, and then read the corresponding Device OUT Endpoint-n Interrupt register to determine the exact source of the interrupt. The application must clear the corresponding status bit in the corresponding Device OUT Endpoint-n Interrupt register to clear this bit.
Bit 18	IEPTINT	0x0	ro	Accessible in device mode only IN Endpoints interrupt The controller sets this bit to indicate that an interrupt is pending one of the IN endpoints in the controller (in device mode). The application must read the Device All Endpoints Interrupt register to determine the exact number of the IN endpoint on which the interrupt occurred, and then read the corresponding Device IN Endpoint-n Interrupt register to determine the exact source of the interrupt. The application must clear the corresponding status bit in the corresponding Device IN Endpoint-n Interrupt register to clear this bit.
Bit 17: 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	EOPF	0x0	rw1c	Accessible in device mode only End of periodic frame interrupt This bit indicates that the period programmed in the periodic frame interval bit of the Device Configuration register has been reached in the current frame.
Bit 14	ISOOUTDROP	0x0	rw1c	Accessible in device mode only Isochronous OUT packet dropped interrupt) The controller sets this bit on the following condition: the controller fails to write a synchronous OUT packet into the receive FIFO because the receive FIFO does not have enough space to accommodate a maximum size packet for the synchronous OUT endpoint.
Bit 13	ENUMDONE	0x0	rw1c	Accessible in device mode only Enumeration done The controller sets this bit to indicate that speed enumeration is done. The application must read the Device Status register to obtain the enumeration speed.
Bit 12	USBRST	0x0	rw1c	Accessible in device mode only USB Reset The controller sets this bit to indicate that a reset is detected on the USB bus.
Bit 11	USBSUSP	0x0	rw1c	Accessible in device mode only USB Suspend The controller sets this bit to indicate that a suspend is detected on the USB bus. The controller enters the Suspend state when there is no activity on the bus for a long period of time.

Bit 10	ERLYSUSP	0x0	rw1c	Accessible in device mode only Early suspend The controller sets this bit to indicate that the idle state has been detected on the USB bus for 3 ms.
Bit 9: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	GOUTNAKEFF	0x0	ro	Accessible in device mode only Global OUT NAK effective This bit indicates that the Set Global OUT NAK bit in the Device Control register (set by the application) has taken effect. This bit can be cleared by writing the Clear Global OUT NAK bit in the Device Control register.
Bit 6	GINNAKEFF	0x0	ro	Accessible in device mode only Global IN Non-periodic NAK effective This bit indicates that the Set Global Non-periodic IN NA bit in the Device Control register (set by the application) has taken effect. That is, the controller has sampled the Global IN NAK bit set by the application. This bit can be cleared by writing the Clear Global Non-periodic IN NA bit in the Device Control register. This interrupt does not necessarily mean that a NAK handshake signal is sent out on the USB bus. The STALL bit has priority over the NAK bit.
Bit 5	NPTXFEMP	0x1	ro	Accessible in both host and device modes Non-periodic TxFIFO empty This interrupt is generated when the Non-periodic TxFIFO is either half or completely empty and there is enough space for at least one request to be written to the Non-periodic Transmit Request Queue. The half or completely empty depends on the Non-periodic TxFIFO Empty Level bit in the Core AHB Configuration register.
Bit 4	RXFLVL	0x0	ro	Accessible in both host and device modes RxFIFO Non-Empty Indicates that there is at least one packet to be read from the receive FIFO.
Bit 3	SOF	0x0	rw1c	Accessible in both host and device modes Start of Frame In host mode, the controller sets this bit to indicate that an SOF (full-speed) or Keep-Alive (low-speed) is transmitted on the USB bus. The application must set this bit to 1 to clear this interrupt. In device mode, the controller sets this bit to indicate that an SOF token has been received on the USB bus. The application must read the Device Status register to get the current frame number. This interrupt can be generated only when the controller is running in FS mode. This bit is set by the controller. The application must write 1 to clear this bit. Note: Reading this register immediately after power-on reset may return the value 0x1. If this register is read as 0x1 immediately after power-on reset, it does not mean that an SOF has been transmitted (in host mode) or received (in device mode). The reading of this register is valid only when an effective connection has been established between the host and the device. If this bit is set after power-on reset, the application can clear this bit.
Bit 2	OTGINT	0x0	ro	Accessible in both host and device modes OTG interrupt The controller sets this bit to indicate that an OTG protocol event is generated. The application must read the OTGFS_GOTGINT register to determine the exact source that caused this interrupt. The application must clear the corresponding status bit in the OTGFS_GOTGINT register to clear this bit.
Bit 1	MODEMIS	0x0	rw1c	Accessible in both host and device modes

				Mode mismatch interrupt The controller sets this bit when the application is attempting to access: A host-mode register, when the controller is running in device mode A device-mode register, when the controller is running in host mode An OKAY response occurs when the register access is completed on the AHB, but it is ignored by the controller internally, and does not affect the operation of the controller. This bit can be set by the controller only. The application must write 1 to clear this bit.
Bit 0	CURMOD	0x0	ro	Accessible in both host and device modes Current mode of operation This bit indicates the current mode. 0: Device mode 1: Host mode

21.6.3.7 OTGFS interrupt mask register (OTGFS_GINTMSK)

This register works with the Interrupt Register to interrupt the application. When an interrupt bit is masked, the interrupt related to this interrupt bit is not generated. However, the Interrupt Register bit corresponding to this interrupt is still set.

Interrupt mask: 0

Interrupt unmask: 1

Bit	Register	Reset value	Type	Description
Bit 31	WKUPINTMSK	0x0	rw	Accessible in both host and device modes Resume/Remote wakeup detected interrupt mask
Bit 30	Reserved	0x0	resd	Kept at its default value.
Bit 29	DISCONINTMSK	0x0	rw	Accessible in both host and device modes Disconnect detected interrupt mask
Bit 28	CONIDSCHGMSK	0x0	rw	Accessible in both host and device modes Connector ID status change mask
Bit 27	Reserved	0x0	resd	Kept at its default value.
Bit 26	PTXFEMPMSK	0x0	rw	Accessible in host mode only Periodic TxFIFO empty mask
Bit 25	HCHINTMSK	0x0	rw	Accessible in host mode only Host channels interrupt mask
Bit 24	PRTINTMSK	0x0	ro	Accessible in host mode only Host port interrupt mask
Bit 23: 22	Reserved	0x0	resd	Kept at its default value.
Bit 21	INCOMPIPMSK INCOMPISOOUTMSK	0x0	rw	Incomplete periodic transfer mask Accessible in host mode only Incomplete isochronous OUT transfer mask Accessible in device mode only
Bit 20	INCOMISOINMSK	0x0	rw	Accessible in device mode only Incomplete isochronous IN transfer mask
Bit 19	OEPTINTMSK	0x0	rw	Accessible in device mode only OUT endpoints interrupt mask
Bit 18	IEPTINTMSK	0x0	rw	Accessible in device mode only IN endpoints interrupt mask
Bit 17	Reserved	0x0	rw	Kept at its default value.
Bit 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	EOPFMSK	0x0	rw	Accessible in device mode only End of periodic frame interrupt mask
Bit 14	ISOOUTDROPMSK	0x0	rw	Device only isochronous OUT packet dropped interrupt mask
Bit 13	ENUMDONEMSK	0x0	rw	Accessible in device mode only Enumeration done mask
Bit 12	USBRSTMSK	0x0	rw	Accessible in device mode only USB Reset mask
Bit 11	USBSUSPMSK	0x0	rw	Accessible in device mode only

				USB suspend interrupt mask
Bit 10	ERLYSUSPMSK	0x0	rw	Accessible in device mode only Early suspend interrupt mask
Bit 9: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	GOUTNAKEFFMSK	0x0	rw	Accessible in device mode only Global OUT NAK effective mask
Bit 6	GINNAKEFFMSK	0x0	rw	Accessible in device mode only Global Non-periodic IN NAK effective mask
Bit 5	NPTXFEMPMSK	0x0	rw	Accessible in both host and device modes Non-periodic TxFIFO empty mask
Bit 4	RXFLVLMSK	0x0	rw	Accessible in both host and device modes Receive FIFO Non-empty mask
Bit 3	SOFMSK	0x0	rw	Accessible in both host and device modes Start of Frame mask
Bit 2	OTGINTMSK	0x0	rw	Accessible in both host and device modes OTG interrupt mask
Bit 1	MODEMISMSK	0x0	rw	Accessible in both host and device modes Mode mismatch interrupt mask
Bit 0	Reserved	0x0	resd	Kept at its default value.

21.6.3.8 OTGFS receive status debug read/OTG status read and POP registers (OTGFS_GRXSTSR / OTGFS_GRXSTSP)

A read to the Receive Status Debug Read register returns the data of the top of the Receive FIFO. A read to the Receive Status Read and Pop register pops the data of the top of the Receive FIFO.

The receive status contents are interpreted differently in host and device modes. Then controller ignores the receive status pop/read when the receive FIFO is empty and returns the value of 0x0000 0000. The application can only pop the receive status FIFO when the receive FIFO non-empty bit of the Core Interrupt register is set.

Host mode:

Bit	Register	Reset value	Type	Description
Bit 31: 21	Reserved	0x000	resd	Kept at its default value.
Bit 20: 17	PKTSTS	0x0	ro	Packet status Indicates the status of the received data packet. 0010: IN data packet received 0011: IN transfer completed (triggers an interrupt) 0101: Data toggle error (triggers an interrupt) 0111: Channel halted (triggers an interrupt) Others: Reserved Reset value: 0
Bit 16: 15	DPID	0x0	ro	Data PID Indicates the data PID of the received data packet. 00: DATA0 10: DATA1 01: DATA2 11: MDATA Reset value: 0
Bit 14: 4	BCNT	0x000	ro	Byte count Indicates the byte count of the received IN data packet.
Bit 3: 0	CHNUM	0x0	ro	Channel number Indicates the channel number to which the currently received data packet belongs.

Device mode:

Bit	Register	Reset value	Type	Description
Bit 31: 25	Reserved	0x00	resd	Kept at its default value.
Bit 24: 21	FN	0x0	ro	Frame number Indicates the least significant 4 bits of the frame number of the data packet received on the USB bus. This field is applicable only when the synchronous OUT endpoints are supported.

Bit 20: 17	PKTSTS	0x0	ro	Packet status Indicates the status of the received data packet. 0001: Global OUT NAK (triggers an interrupt) 0010: OUT data packet received 0011: OUT transfer completed (triggers an interrupt) 0100: SETUP transaction completed (triggers an interrupt) 0110: SETUP data packet received Others: Reserved
Bit 16: 15	DPID	0x0	ro	Data PID Indicates the data PID of the received OUT data packet. 00: DATA0 10: DATA1 01: DATA2 11: MDATA
Bit 14: 4	BCNT	0x000	ro	Byte count Indicates the byte count of the received data packet.
Bit 3: 0	EPTNUM	0x0	ro	Endpoint number Indicates the endpoint number to which the currently received data packet belongs.

21.6.3.9 OTGFS receive FIFO size register (OTGFS_GRXFSIZ)

The application can program the SRAM size that must be allocated to the receive FIFO.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	RXFDEP	0x0200	ro/rw	RxFIFO Depth This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 512 The power-on reset value of this register is defined as the largest receive data FIFO depth during the configuration.

21.6.3.10 OTGFS non-periodic Tx FIFO size (OTGFS_GNPTXFSIZ)/Endpoint 0 Tx FIFO size registers (OTGFS_DIEPTXF0)

The application can program the SRAM size and start address of the non-periodic transmit FIFO. The fields of this register varies with host mode or device mode.

Host:

Bit	Register	Reset value	Type	Description
Bit 31: 16	NPTXFDEP	0x0000	ro/rw	Non-periodic TxFIFO depth This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 256
Bit 15: 0	NPTXFSTADDR	0x0200	ro/rw	Non-periodic transmit SRAM start address This field contains the memory start address of the Non-periodic Transmit FIFO SRAM.

Device:

Bit	Register	Reset value	Type	Description
Bit 31: 16	INEPT0TXDEP	0x0000	ro/rw	N Endpoint TxFIFO 0 depth This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 256
Bit 15: 0	INEPT0TXSTADDR	0x0200	ro/rw	IN Endpoint FIFO0 transmit SRAM start address This field contains the memory start address of the IN Endpoint FIFO0 transmit SRAM.

21.6.3.11 OTGFS non-periodic Tx FIFO size/request queue status register (OTGFS_GNPTXSTS)

This register is valid in host mode only. It is a read-only register that contains the available space information for the Non-periodic TxFIFO and the Non-periodic Transmit Request Queue.

Bit	Register	Reset value	Type	Description
Bit 31	Reserved	0x0	resd	Kept at its default value.
Bit 30: 24	NPTXQTOP	0x00	ro	Top of the Non-periodic transmit request queue Indicates that the MAC is processing the request from the non-periodic transmit request queue. Bit [30: 27]: Channel/Endpoint number Bit [26: 25]: 00: IN/OUT token 01: Zero-length transmit packet (device IN/host OUT) 10: PING/CSPLIT token 11: Channel halted command Bit [24]: Terminate (last request for the selected channel/endpoint)
Bit 23: 16	NPTXQSPCAVAIL	0x08	ro	Non-periodic transmit request queue space available Indicates the amount of space available in the non-periodic transmit request queue. This queue supports both IN and OUT requests in host mode. 00: Non-periodic transmit request queue is full 01: 1 location available 02: 2 locations available N: n locations available (0 ≤ n ≤ 8) Others: Reserved Reset value: Configurable
Bit 15: 0	NPTXFSPCAVAIL	0x0200	ro	Non-periodic TxFIFO space available Indicates the amount of space available in the non-periodic TxFIFO. Values are in terms of 32-bit words. 00: Non-periodic transmit FIFO is full 01: 1 location available 02: 2 locations available N: n locations available (0 ≤ n ≤ 256) Others: Reserved Reset value: Configurable

21.6.3.12 OTGFS general controller configuration register (OTGFS_GCCFG)

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x000	resd	Kept at its default value.
Bit 21	VBUSIG	0x0	rw	VBUS ignored When this bit is set, the OTGFS controller does not monitor the Vbus pin voltage, and assumes that the Vbus is always active in both host and device modes, and leaves the Vbus pin for other purposes. 0: Vbus is not ignored 1: Vbus is ignored, and is deemed as always active
Bit 20	SOFOUTEN	0x0	rw	SOF output enable 0: No SOF pulse output 1: SOF pulse output on PIN
Bit 19: 18	Reserved	0x0	resd	Kept at its default value.
Bit 17	LP_MODE	0x0	rw	Low-power mode This bit is used to control the OTG PHY consumption. When this bit is set to 1 by software, the OTG PHY enters low-power mode; when this bit is cleared by software, the OTG PHY operates in normal mode. 0: Non-low-power mode 1: Low-power mode
Bit 16	PWRDOWN	0x0	rw	Power down This bit is used to activate the transceiver in transmission/reception. It must be pre-configured to allow

				USB communication. 0: Power down enable 1: Power down disable (Transceiver active)
Bit 15: 0	Reserved	0x0000	resd	Kept at its default value.

21.6.3.13 OTGFS controller ID register (OTGFS_GUID)

This is a read-only register containing the production ID.

Bit	Register	Reset value	Type	Description
31: 0	USERID	0x0000 1000	rw	Product ID field The application can program the ID field.

21.6.3.14 OTGFS host periodic Tx FIFO size register (OTGFS_HPTXFSIZ)

This register contains the size and memory start address of the periodic transmit FIFO.

Bit	Register	Reset value	Type	Description
Bit 31: 16	PTXFSIZE	0x02000	ro/rw	Host periodic TxFIFO depth Values are in terms of 32-bit words. Minimum value is 16 Maximum value is 512
Bit 15: 0	PTXFSTADDR	0x0600	ro/rw	Host Periodic TxFIFO start address The power-on reset value of this register is the sum of the largest receive FIFO depth and the largest non-periodic transmit FIFO depth.

21.6.3.15 OTGFS device IN endpoint Tx FIFO size register (OTGFS_DIEPTXFn) (x=1…7, where n is the FIFO number)

This register holds the depth and memory start address of the IN endpoint transmit FIFO in device mode. Each of the FIFOs contains an IN endpoint data. This register can be used repeatedly for instantiated IN endpoint FIFO1~15. The GNPTXFSIZ register is used to program the depth and memory start address of the IN endpoint FIFO 0.

Bit	Register	Reset value	Type	Description
Bit 31: 16	INEPTXFDEP	0x0200	ro/rw	IN Endpoint TxFIFO depth Values are in terms of 32-bit words. Minimum value is 16 Maximum value is 512 The reset value is the maximum possible IN endpoint transmit FIFO depth
Bit 15: 0	INEPTXFSTADDR	0x0400	ro/rw	IN Endpoint FIFOn transmit SRAM start address This field contains the SRAM start address of the IN endpoint n transmit FIFO

21.6.4 Host-mode registers

Host-mode registers affect the operation of the controller in host mode. Host-mode register are not accessible in device mode (as the results are undefined in device mode). Host-mode registers contain as follows:

21.6.4.1OTGFS host mode configuration register (OTGFS_HCFG)

This register is used to configure the controller after power-on. Do not change this register after initialization.

Bit	Register	Reset value	Type	Description
Bit 31: 3	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 2	FSLSSUPP	0x0	ro	FS- and LS-only support The application uses this bit to control the controller's enumeration speed. With this bit, the application can make the controller enumerate as a full-speed host mode, even if the connected device supports high-speed communication. Do not change this bit after initial programming. 0: FS/LS, depending on the largest speed supported by the connected device.

				1: FS/LS-only, even if the connected device supports high-speed.
Bit 1: 0	FSLSPCLKSEL	0x0	rw	FS/LS PHY clock select When the controller is in FS host mode: 01: PHY clock is running at 48MHz Others: Reserved When the controller is in LS host mode: 00: Reserved 01: PHY clock is running at 48 MHz 10: PHY clock is running at 6 MHz. If 6 MHz clock is selected, reset must be done by software. 11: Reserved

21.6.4.2 OTGFS host frame interval register (OTGFS_HFIR)

This register is used to program the current

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	HFIRRLDCTRL	0x0	rw	Reload control This bit is used to disable/enable dynamic reload for the host frame register at runtime. 1: Reload control disable 0: Reload control enable This bit must be configured at initialization. Do not change its value at runtime.
Bit 15: 0	FRINT	0xEA60	rw	Frame interval The application uses this filed to program the interval between two consecutive SOFs (full speed) The number of PHY locks in this field indicates the frame interval. The application can write a value to the host frame interval register only after the port enable bit in the host port control and status register has been set. If no value is programmed, the controller calculates the value based on the PHY clock frequency defined in the FS/LS PHY clock select bit of the host configuration register. Do not change the value of this field after initial configuration. 1 ms * (FS/LS PHY clock frequency)

21.6.4.3 OTGFS host frame number/frame time remaining register (OTGFS_HFNUM)

This register indicates the current frame number, and also the time remaining in the current frame (in terms of the number of PHY clocks).

Bit	Register	Reset value	Type	Description
Bit 31: 16	FTREM	0x0000	ro	Frame time remaining Indicates the time remaining in the current frame (FS/HS), in terms of the number of PHY clocks. This field decrements with the number of PHY clocks. When it reaches zero, this filed is reloaded with the value of the frame interval register, and a new SOF is transmitted on the USB bus.
Bit 15: 0	FRNUM	0x3FFF	ro	Frame number This field increments every time a new SOP is transmitted on the USB bus, and is cleared to 0 when the value reaches 16'h3FFF.

21.6.4.4 OTGFS host periodic Tx FIFO/request queue register (OTGFS_HPTXSTS)

This is a ready-only register containing the free space information of the periodic Tx FIFO and the periodic transmit request queue.

Bit	Register	Reset value	Type	Description
Bit 31: 24	PTXQTOP	0x00	ro	Top of the periodic transmit request queue) Indicates that the MAC is processing the request from the periodic transmit request queue. This register is used for debugging. Bit [31]: Odd/Even frame 0: Transmit in even frame 1: Transmit in odd frame Bit [30: 27]: Channel/Endpoint number Bit [26: 25]: Type 00: IN/OUT 01: Zero-length packet 10: Reserved 11: Channel command disable Bit [24]: Terminate (last request for the selected channel or endpoint)
Bit 23: 16	PTXQSPCAVAIL	0x08	ro	Periodic transmit request queue space available Indicates the number of free space available to be written in the periodic transmit request queue. This queue contains both IN and OUT requests. 00: Periodic transmit request queue is full 01: 1 space available 10: 2 space available N: n space available (0 ≤ n ≤ 8) Others: Reserved
Bit 15: 0	PTXFSPCAVAIL	0x0100	rw	Periodic transmit data FIFO space available Indicates the number of free space available to be written in the periodic transmit FIFO, in terms of 32-bit words. 0000: Periodic transmit FIFO is full 0001: 1 space available 0010: 2 space available N: n space available (0 ≤ n ≤ 512) Others: Reserved

21.6.4.5 OTGFS host all channels interrupt register (OTGFS_HAINT)

When a flag event occurs on a channel, the host all channels interrupt register interrupts the application through the host channels interrupt bit of the controller interrupt register, as shown in Figure 21-2. There is one interrupt bit for each channel, up to 16 bits. The application sets or clears this register by setting or clearing the appropriate bit in the corresponding host channel-n interrupt register.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	HAINT	0x0000	ro	Channel interrupts One bit per channel: bit 0 for channel 0, bit 15 for channel 15.

21.6.4.6 OTGFS host all channels interrupt mask register (OTGFS_HAINTMSK)

The host all channels interrupt mask register works with the host all channels interrupt register to interrupt the application when an event occurs on a channel. There is one interrupt mask bit per one channel, 16 bits in total.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	HAINTMSK	0x0000	rw	Channel interrupt mask One bit per channel: bit 0 for channel 0, bit 15 for channel 15.

21.6.4.7 OTGFS host port control and status register (OTGFS_HPRT)

This register is valid only in host mode. Currently, the OTG host supports only one port.

This register contains USB port-related information such as USB reset, enable, suspend, resume, connect status and test mode, as show in Figure 21-2. The register of type rw1c can interrupt the application through the host port interrupt bit in the controller interrupt register. Upon a port interrupt, the application must read this register and clear the bit that caused the interrupt. For the register of type rw1c, the application must write 1 to clear the interrupt.

Bit	Register	Reset value	Type	Description
Bit 31: 19	Reserved	0x0000	resd	Kept at its default value.
Bit 18: 17	PRTSPD	0x0	ro	Port speed Indicates the speed of the device connected to this port. 00: Reserved 01: Full speed 10: Low speed 11: Reserved
Bit 16: 13	PRTTSTCTL	0x0	rw	Port test control The application writes a non-zero value to this field to put the port into test mode, and the port gives a corresponding signal. 0000: Test mode disabled 0001: Test_J mode 0010: Test_K mode 0011: Test_SE0_NAK mode 0100: Test_Packet mode 0101: Test_Force_Enable Others: Reserved
Bit 12	PRTPWR	0x0	rw	Port power The application uses this bit to control power supply to this port (by writing 1 or 0) 0: Power off 1: Power on Note: This bit is not associated with interfaces. The application must follow the programming manual to set this bit for various interfaces.
Bit 11: 10	PRTLNSTS	0x0	ro	Port line status Indicates the current logic status of the USB data lines. Bit [10]: Logic level of D+ Bit [11]: Logic level of D–
Bit 9	Reserved	0x0	resd	Kept at its default value.
Bit 8	PRTRST	0x0	rw	Port reset When this bit is set by the application, a reset sequence is started on this port. The application must calculate the time required for the reset sequence, and clear this bit after the reset sequence is complete. 0: Port not in reset 1: Port in reset The application must keep this bit set for a minimum duration defined in Section 7.1.7.5 of USB 2.0 specification to start a reset on the port. In addition to this, the application can make this bit set for another 10 ms to the minimum duration, before clearing this bit. There is no maximum limit set by the USB standard.
Bit 7	PRTSUSP	0x0	rw1s	Port suspend The application sets this bit to put this port in suspend mode. In this case, the controller only stops sending SOF. The application must set the port clock stop bit in order to disable the PHY clock. The read value of this bit reflects the current suspend status of the port. This bit is cleared by the controller when a remote wakeup signal is detected or when the application sets the port reset bit or port resume bit in this register, or sets the

				resume/remote wakeup detected interrupt bit or disconnect detected interrupt bit in the controller interrupt register. The controller can still clear this bit, even if the device is disconnected with the host. 0: Port not in suspend mode 1: Port in suspend mode
Bit 6	PRTRES	0x0	rw	Port resume The application sets this bit to drive resume signaling on the port. The controller continues to trigger the resume signal until the application clears this bit. If the controller detects a USB remote wakeup sequence (as indicated by the port resume/remote wakeup detected interrupt bit of the controller interrupt register), the controller starts driving resume signaling without the intervention of the application. The read value of this bit indicates whether the controller is currently driving resume signaling. 0: No resume triggered 1: Resume triggered
Bit 5	PRTOVRCCHNG	0x0	rw1c	Port overcurrent change The controller sets this bit when the status of the port overcurrent active bit (bit 4) in this register changes. This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 4	PRTOVRCACT	0x0	ro	Port overcurrent active Indicates the overcurrent status of the port. 0: No overcurrent 1: Overcurrent condition
Bit 3	PRTENCHNG	0x0	rw1c	Port enable/disable change The controller sets this bit when the status of the port enable bit 2 in this register changes. This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 2	PRTENA	0x0	rw1c	Port enable A port is enabled only by the controller after a reset sequence. This port is enabled by an overcurrent condition, a disconnected condition ro by the application. The application cannot set this bit by a register write operation. It can only clear this bit to disable the port. This bit does not trigger any interrupt. 0: Port disabled 1: Port enabled
Bit 1	PRTCONDET	0x0	rw1c	Port connect detected On a device connection detected, the controller sets this bit using the host port interrupt bit in the controller register. This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 0	PRTCONSTS	0x0	ro	Port connect status 0: No device is connected to the port 1: A device is connected to the port

21.6.4.8 OTGFS host channelx characteristics register (OTGFS_HCCHARx) (x = 0...15, where x= channel number)

Bit	Register	Reset value	Type	Description
Bit 31	CHENA	0x0	rw1s	Channel enable This bit is set by the application and cleared by the OTG host. 0: Channel disabled 1: Channel enabled
Bit 30	CHDIS	0x0	rw1s	Channel disable The application sets this bit to stop transmitting or receiving data on a channel, even before the transfer on

				that channel is complete. The application must wait for the generation of the channel disabled interrupt before treating the channel as disabled.
Bit 29	ODDFRM	0x0	rw	Odd frame This bit is set / cleared by the application to indicate that the OTG host must perform a transfer in an odd frame. This bit is applicable for periodic transfers (synchronous and interrupt) only. 0: Even frame 1: Odd frame
Bit 28: 22	DEVADDR	0x00	rw	Device address This field is used to select the device that can serve as the data source or receiver.
Bit 21: 20	MC	0x0	rw	Multi count (MC) This field indicates to the host the number of transfers that must be performed per frame for the periodic endpoint. 00: Reserved. This field generates undefined results. 01: 1 transaction 10: 2 transactions per frame 11: 3 transactions per frame This field must be set to at least 0x01.
Bit 19: 18	EPTYPE	0x0	rw	Endpoint type Indicates the transfer type selected. 00: Control transfer 01: Synchronous transfer 10: Bulk transfer 11: Interrupt transfer
Bit 17	LSPDDEV	0x0	rw	Low-speed device The application sets this bit to indicate that this channel is communicating to a low-speed device.
Bit 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	EPTDIR	0x0	rw	Endpoint direction Indicates whether the transfer is in IN or OUT. 0: OUT 1: IN
Bit 14: 11	EPTNUM	0x0	rw	Endpoint number Indicates the endpoint number on the device (serving as data source or receiver)
Bit 10: 0	MPS	0x000	rw	Maximum packet size Indicates the maximum packet size of the corresponding port.

21.6.4.9 OTGFS host channelx interrupt register (OTGFS_HCINTx) (x = 0...15, where x= channel number)

This register contains the status of a channel related to USB and AHB events, as shown in Figure 21-2. The application must read this register when the host channels interrupt bit is set in the controller interrupt register. Before reading this register, the application must read the host all channels interrupt register to get the exact channel number of the host channel-n interrupt register. The application must clear the corresponding bit in this register to clear the corresponding bits in the OTGFS_HAIN and OTGFS_GINTSTS registers.

Bit	Register	Reset value	Type	Description
Bit 31: 11	Reserved	0x000000	resd	Kept at its default value.
Bit 10	DTGLERR	0x0	rw1c	Data toggle error This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 9	FRMOVRUN	0x0	rw1c	Frame overrun This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 8	BBLERR	0x0	rw1c	Babble error This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 7	XACTERR	0x0	rw1c	Transaction error Indicates one of the following errors occurred on the USB

				bus: CRC check failure Timeout Bit stuffing error EOP error This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	ACK	0x0	rw1c	ACK response received/Transmitted interrupt This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 4	NAK	0x0	rw1c	NAK response received interrupt This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 3	STALL	0x0	rw1c	STALL response received interrupt This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	CHHLTD	0x0	rw1c	Channel hated Indicates that the transfer completed abnormally either because of any transfer error or in response to a disable request by the application.
Bit 0	XFERC	0x0	rw1c	Transfer completed Transfer completed normally, without any error. This bit can only be set by the controller. The application must write 1 to clear this bit.

21.6.4.10 OTGFS host channelx interrupt mask register (OTGFS_HCINTMSKx) (x = 0...15, where x= channel number)

This register is used to mask the channels described in the previous section.

Bit	Register	Reset value	Type	Description
Bit 31: 11	Reserved	0x000000	resd	Kept at its default value.
Bit 10	DTGLERRMSK	0x0	rw	Data toggle error mask
Bit 9	FRMOVRUNMSK	0x0	rw	Frame overrun mask
Bit 8	BBLERRMSK	0x0	rw	Babble error mask
Bit 7	XACTERRMSK	0x0	rw	Transaction error mask
Bit 6	NYETMSK	0x0	rw	NYET response received interrupt mask
Bit 5	ACKMSK	0x0	rw	ACK response received/transmitted interrupt mask
Bit 4	NAKMSK	0x0	rw	NAK response received interrupt mask
Bit 3	STALLMSK	0x0	rw	STALL response received interrupt mask
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	CHHLTDMSK	0x0	rw	Channel halted mask
Bit 0	XFERCMSK	0x0	rw	Transfer completed mask

21.6.4.11 OTGFS host channelx transfer size register (OTGFS_HCTSIZx) (x = 0...15, where x= channel number)

Bit	Register	Reset value	Type	Description
Bit 31	Reserved	0x0	resd	Kept at its default value.
Bit 30: 29	PID	0x0	rw	PID (Pid) The application programs this field with the type of PID used for the initial transfer. The host controls this filed for the rest of transfers. 00: DATA0 01: DATA2 10: DATA1 11: MDATA(non-control)/SETUP(control)
Bit 28: 19	PKTCNT	0x000	rw	Packet count The application programs this field with the expected number of packets to be transmitted or received. The host decrements the packet count on every successful transmission or reception of an OUT/IN packet. When this count reaches zero, the application is interrupted to

				indicate normal completion of the transfer.
Bit 18: 0	XFERSIZE	0x00000	rw	Transfer size For an OUT transfer, this field indicates the number of data bytes the host sends during a transfer. For an IN transfer, this field indicates the buffer size that the application has reserved for the transfer. For an IN transfer (periodic and non-periodic), the application must program this field as an integer multiple of the maximum packet size.

21.6.5 Device-mode registers

These registers are applicable in device mode only. They are not supported in host mode due to unknown access results. Some of the registers affect all the endpoints, while some affect only one endpoint.

21.6.5.1 OTGFS device configure register (OTGFS_DCFG)

This register configures the controller in device mode after power-on or after certain control commands or enumeration. Do not change this register after initial programming.

Bit	Register	Reset value	Type	Description
Bit 31: 13	Reserved	0x0110	resd	Kept at its default value.
Bit 12: 11	PERFRINT	0x0	rw	Periodic frame interval This field indicates the time within a frame at which the periodic frame end interrupt is generated. The application can use this interrupt to determine if the synchronous transfer has been completed in a frame. 00: 80% of the frame interval 01: 85% of the frame interval 10: 90% of the frame interval 11: 95% of the frame interval
Bit 10: 4	DEVADDR	0x00	rw	Device address The application must program this field every time a SetAddress command is received.
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2	NZSTSOUTHSHK	0x0	rw	Non-zero-length status OUT handshake The application can use this field to select the handshake the controller sends on receiving a non-zero-length data packet during a control transfer' status stage. 1: Send a STALL handshake on a non-zero-length status OUT transfer and do not send the received OUT packet to the application 0: Send the received OUT packet to the application (zero-length or non-zero-length), and send a handshake based on the NAK and STALL bits in the device endpoint control register.
Bit 1: 0	DEVSPD	0x0	rw	Device speed This field indicates the speed at which the application needs the controller to enumerate, or the maximum speed the application can support. However, the actual bus speed is determined only after the entire sequence is complete, and is based on the speed of the USB host to which the controller is connected. 00: Reserved 01: Reserved 10: Reserved 11: Full speed (USB1.1 transceiver, clock is 48MHz)

21.6.5.2 OTGFS device control register (OTGFS_DCTL)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value.
Bit 11	PWROPRGDNE	0x0	wo	Power-on programming done The application uses this bit to indicate that the register configuration is complete after a wakeup from power-down mode.
Bit 10	CGOUTNAK	0x0	wo	Clear global OUT NAK

				Writing 1 to this bit clears the global OUT NAK.
Bit 9	SGOUTNAK	0x0	wo	Set global OUT NAK Writing to this bit sets the global OUT NAK. The application uses this bit to send a NAK handshake on all OUT endpoints. The application must set this bit only after checking that the global OUT NAK effective bit in the controller interrupt register is cleared.
Bit 8	CGNPINNAK	0x0	wo	Clear Global Non-periodic IN NAK Writing to this bit clears the global Non-periodic OUT NAK.
Bit 7	SGNPINNAK	0x0	wo	Set global Non-periodic IN NAK Writing to this bit sets the global Non-periodic OUT NAK. The application uses this bit to send a NAK handshake on all non-periodic IN endpoints. The application must set this bit only after checking that the global IN NAK effective bit in the controller interrupt register is cleared.
Bit 6: 4	TSTCTL	0x0	rw	Test control 000: Test mode disabled 001: Test_J mode 010：Test_K mode 011: Test_SE0_NAK mode 100: Test_Packet mode 101: Test_Force_Enable Others: Reserved
Bit 3	GOUTNAKSTS	0x0	ro	Global OUT NAK status 0: A handshake is sent based on the FIFO status, NAK and STALL bit settings. 1: No data is written to the receive FIFO, irrespective of space availability. Sends a NAK handshake on all packets (except on SETUP transfers). Drops all synchronous OUT packets.
Bit 2	GNPINNAKSTS	0x0	ro	Global Non-periodic IN NAK status 0: A handshake is sent based on the data status in the transmit FIFO 1: A NAK handshake is sent on all non-periodic IN endpoints, irrespective of the data status in the transmit FIFO.
Bit 1	SFTDISCON	0x1	rw	Software disconnect The application uses this bit to indicate the OTGFS controller to perform software disconnected. Once this bit is set, the host finds the device disconnected, and the device does not receive signals on the USB bus. The controller stays in the disconnected state until the application clears this bit. 0: Normal operation. When this bit is cleared after a software disconnect, the controller issues a device connect event to the host. Then the USB host restarts device enumeration.
Bit 0	RWKUPSIG	0x0	rw	Remote wakeup signaling When this bit is set by the application, the controller initiates a remote signal to wakeup the USB host. The application must set this bit to indicate the controller to exit the suspend mode. Per USB2.0 standards, the application must clear this bit 1-15 ms after setting it.

Table 21-5 lists the minimum duration at which the software disconnect bit must be set in various states for the USB host to detect a device disconnect. To accommodate clock jitter, it is advised that the application adds some extra delay to the specified minimum duration.

Table 21-5 Minimum duration for software disconnect

Operating speed	Device state	Minimum duration
Full speed	Suspend	1ms + 2.5us
Full speed	Idle	2.5us
Full speed	No idle or suspend (performing transfers)	2.5us

21.6.5.3 OTGFS device status register (OTGFS_DSTS)

This register indicates the status of the controller related to OTGFS events. It must be read on interrupt events from the device all interrupts register (OTGFS_DAINT).

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x000	resd	Kept at its default value.
Bit 21: 8	SOFFN	0x0000	ro	Frame number of the received SOF Note: The read value of this field immediately after power-on reset reflects a non-zero value. If a non-zero value is returned after reading this field immediately after power-on reset, it does not mean that the host has received a SOP. The read value of this field is valid only when the host is connected to the device.
Bit 7: 4	Reserved	0x1	resd	Kept at its default value.
Bit 3	ETICERR	0x0	ro	Erratic error This error causes the controller to enter suspend mode, and interrupt is generated with the early suspend bit of the controller interrupt register. If the early suspend is asserted due to an erratic error, the application can only perform a software disconnect recover.
Bit 2: 1	ENUMSPD	0x0	ro	Enumerated speed Indicates the speed at which the controller has determined after speed detection through a sequence. 01: Reserved 10: Reserved 11: Full speed (PHY clock is running at 48MHz) Others: Reserved
Bit 0	SUSPSTS	0x0	ro	Suspend status In device mode, this bit is set as long as a suspend condition is detected on the USB bus. The controller enters the suspend state when there is no activity on the USB bus. The controller exits the suspend state on the following conditions: When there is an activity on the USB bus When the application writes to the remote wakeup signal bit in the device control register.

21.6.5.4 OTGFS device OTGFSIN endpoint common interrupt mask register (OTGFS_DIEPMSK)

This register works with each of the device IN endpoint interrupt register for all endpoints to generate an IN endpoint interrupt. The IN endpoint interrupt for a specific status in the OTGFS_DIEPINTx register can be masked by writing to the corresponding bit in the OTGFS_DIEPMSK register. Status bits are masked by default.

Bit	Register	Reset value	Type	Description
Bit 31: 10	Reserved	0x000000	resd	Kept at its default value.
Bit 9	BNAINMSK	0x0	rw	BNA interrupt mask 0: Interrupt masked 1: Interrupt unmasked
Bit 8	TXFIFOUDRMSK	0x0	rw	FIFO underrun mask 0: Interrupt masked 1: Interrupt unmasked
Bit 7	Reserved	0x0	resd	Kept at its default value.

Bit 6	INEPTNAKMSK	0x0	rw	IN endpoint NAK effective mask 0: Interrupt masked 1: Interrupt unmasked
Bit 5	INTKNEPTMISMSK	0x0	rw	IN token received with EP mismatch mask 0: Interrupt masked 1: Interrupt unmasked
Bit 4	INTKNTXFEMPMSK	0x0	rw	IN token received when TxFIFO empty mask 0: Interrupt masked 1: Interrupt unmasked
Bit 3	TIMEOUTMSK	0x0	rw	Timeout condition mask (Non-isochronous endpoints)) 0: Interrupt masked 1: Interrupt unmasked
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	EPTDISMSK	0x0	rw	Endpoint disabled interrupt mask 0: Interrupt masked 1: Interrupt unmasked
Bit 0	XFERCMSK	0x0	rw	Transfer completed interrupt mask 0: Interrupt masked 1: Interrupt unmasked

21.6.5.5 OTGFS device OUT endpoint common interrupt mask register (OTGFS_DOEPMSK)

This register works with each of the OTGFS_DOEPINTx registers for all endpoints to generate an OUT endpoint interrupt. Each of the bits in the OTGFS_DOEPINTx registers can be masked by writing to the register. All interrupts are masked by default.

Bit	Register	Reset value	Type	Description
Bit 31:10	Reserved	0x000000	resd	Kept at its default value.
Bit 9	BNAOUTMSK	0x0	rw	BNA interrupt mask 0: Interrupt masked 1: Interrupt unmasked
Bit 8	OUTPERRMSK	0x0	rw	OUT packet error mask 0: Interrupt masked 1: Interrupt unmasked
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6	B2BSETUPMSK	0x0	rw	Back-to-back SETUP packets received mask 0: Interrupt masked 1: Interrupt unmasked
Bit 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	OUTTEPDMSK	0x0	rw	OUT token received when endpoint disabled mask 0: Interrupt masked 1: Interrupt unmasked
Bit 3	SETUPMSK	0x0	rw	SETUP phase done mask Applies to control endpoints only. 0: Interrupt masked 1: Interrupt unmasked
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	EPTDISMSK	0x0	rw	Endpoint disabled interrupt mask 0: Interrupt masked 1: Interrupt unmasked
Bit 0	XFERCMSK	0x0	rw	Transfer completed interrupt mask 0: Interrupt masked 1: Interrupt unmasked

21.6.5.6 OTGFS device all endpoints interrupt mask register (OTGFS_DAINT)

When an event occurs on an endpoint, The IN/OUT endpoint interrupt bits in the OTGS_DAINT register can be used to interrupt the application. There is one interrupt pit per endpoint, up to 8 interrupt bits for OUT endpoints and 8 bits for IN endpoints. For a bidirectional endpoint, the corresponding IN and OUT interrupt bits are used at the same time. The corresponding bits in this register are set and cleared when the application sets and clears the bits in the corresponding device endpoint-x interrupt register.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x0000	resd	Kept at its default value.
Bit 23: 16	OUTEPTINT	0x0000	ro	OUT endpoint interrupt bits One OUT endpoint per bit. Bit 16 for OUT endpoint 0, bit 18 for OUT endpoint 2.
Bit 15: 8	Reserved	0x0000	resd	Kept at its default value.
Bit 7: 0	INEPTINT	0x0000	ro	IN endpoint interrupt bits One IN endpoint per bit. Bit 0 for IN endpoint 0, bit 7 for IN endpoint 7.

21.6.5.7 OTGFS all endpoints interrupt mask register (OTGFS_DAINTMSK)

When an event occurs on a device endpoint, the device endpoint interrupt mask register works with the device endpoint interrupt register to interrupt the application. However, the device all endpoints interrupt register corresponding to this interrupt is still set.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x0000	resd	Kept at its default value.
Bit 23: 16	OUTEPTMSK	0x0000	rw	OUT EP interrupt mask bits One OUT endpoint per bit. Bit 16 for OUT endpoint 0, bit 18 for OUT endpoint 2. 0: Interrupt masked 1: Interrupt unmasked
Bit 15: 8	Reserved	0x0000	resd	Kept at its default value.
Bit 7: 0	INEPTMSK	0x0000	rw	IN EP interrupt mask bits One IN endpoint per bit. Bit 0 for IN endpoint 0, bit 7 for IN endpoint 7. 0: Interrupt masked 1: Interrupt unmasked

21.6.5.8 OTGFS device IN endpoint FIFO empty interrupt mask register (OTGFS_DIEPEMPMSK)

This register works with the TXFE_OTGFS_DIEPINTx register to generate an interrupt.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0000	resd	Kept at its default value.
Bit 7: 0	INEPTXFEMSK	0x0000	rw	IN endpoint Tx FIFO empty interrupt mask bits These bits serve as mask bits for the device IN endpoint interrupt register. A transmit FIFO empty interrupt bit per IN endpoint. Bit 0 for IN endpoint 0, bit 7 for IN endpoint 7. 0: Interrupt masked 1: Interrupt unmasked

21.6.5.9 OTGFS device control IN endpoint 0 control register (OTGFS_DIEPCTL0)

This section describes the control IN endpoint 0 control register. Nonzero control endpoint uses registers for endpoints 1-7.

Bit	Register	Reset value	Type	Description
Bit 31	EPTENA	0x0	rw1s	Endpoint enable The application sets this bit to start data transmission on the endpoint 0. The controller clears this bit before generating the following interrupts:

				Endpoint disabled Transfer completed.
Bit 30	EPTDIS	0x0	ro	Endpoint disable The application sets this bit to stop data transmission on an endpoint. The application must wait for the endpoint disabled interrupt before treating the endpoint as disabled. The controller clears this bit before setting the endpoint disabled interrupt. The application must set this bit only when the endpoint is enabled.
Bit 29: 28	Reserved	0x0	resd	Kept at its default value.
Bit 27	SNAK	0x0	wo	Set NAK A write to this bit sets the NAK bit of the endpoint. The application can use this bit to control the transmission of NAK handshakes on the endpoint. The controller also sets this bit when a SETUP data packet is received on the endpoint.
Bit 26	CNAK	0x0	wo	Clear NAK A write to this bit clears the NAK bit for the endpoint.
Bit 25: 22	TXFNUM	0x0	rw	TxFIFO number The endpoint 0 can only use FIFO0.
Bit 21	STALL	0x0	rw1s	STALL handshake The application sets this bit, and the controller clears this bit when a SETUP token is received. If a NAK bit, a global non-periodic IN NAK or global OUT NAK bit is set along with this bit, the STALL bit has priority.
Bit 20	Reserved	0x0	resd	Kept at its default value.
Bit 19: 18	EPTYPE	0x0	ro	Endpoint type Set to 0 by hardware for control endpoints.
Bit 17	NAKSTS	0x0	ro	NAK status Indicates the following: 0: The controller is transmitting non-NAK handshakes based on the FIFO status 1: The controller is transmitting NAK handshakes on this endpoint When this bit is set, either by the application or controller, the controller stops transmitting data, even if there are space available in the receive FIFO. The controller always responds to SETUP data packets with an ACK handshake, irrespective of this bit's setting.
Bit 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	USBACEPT	0x0	ro	USB active endpoint This bit is always set to 1, indicating that the control endpoint 0 is always active in all configurations and interfaces.
Bit 14: 2	Reserved	0x0000	resd	Kept at its default value.
Bit 1: 0	MPS	0x0	rw	Applies to IN and OUT endpoints The application uses this bit to program the maximum packet size for the current logical endpoint. 00: 64 bytes 01: 32 bytes 10: 16 bytes 11: 8 bytes

21.6.5.10 OTGFS device IN endpoint-x control register (OTGFS_DIEPCTLx) (x=x=1…7, where x is endpoint number)

The application uses this register to control the behavior of the endpoints other than endpoint 0.

Bit	Register	Reset value	Type	Description
Bit 31	EPTENA	0x0	rw1s	Endpoint enable The application sets this bit to start transmitting data on an endpoint. The controller clears this bit before the generation one of the following interrupts on this endpoint: SETUP stage done Endpoint disabled Transfer completed
Bit 30	EPTDIS	0x0	rw1s	Endpoint disable The application sets this bit to stop transmitting data on an endpoint, even if the transfer on that endpoint is incomplete. The application must wait for the endpoint disabled interrupt before treating the endpoint as disabled. The controller clears this bit before setting the endpoint disabled interrupt. The application must set this bit only when the endpoint enabled set.
Bit 29	SETD1PID/ SETODDFR	0x0	wo	Set DATA1 PID Applies to interrupt/bulk IN endpoints only. Writing to this bit sets the endpoint data PID bit in this register to DATA1. Set odd frame Applies to synchronous IN endpoints only. Writing to this bit sets the Even/Odd frame to odd frame. 0: Disabled Set DATA1 PID disabled or Do not force odd frame 1: Set DATA1 PID enabled or forced odd frame
Bit 28	SETD0PID/ SETEVENFR	0x0	rw	Set DATA0 PID Applies to interrupt/bulk IN endpoints only. Writing to this bit sets the endpoint data PID bit in this register to DATA0. Set Even frame Applies to synchronous IN endpoints only. Writing to this bit sets the Even/Odd frame to even frame. 0:Disabled Set DATA0 PID disabled or Do not force even frame 1: Set DATA0PID or set the EOFRNUM to even frame
Bit 27	SNAK	0x0	wo	Set NAK A write to this bit sets the NAK bit for the endpoint. The application uses this bit to control the transmission of NAK handshakes on an endpoint. The controller sets this bit on a Transfer completed interrupt or after receiving a SETUP packet. Values: 0: Do not set NAK 1: Set NAK
Bit 26	CNAK	0x0	wo	Clear NAK A write to this bit clears the NAK bit for this endpoint. 0: Not clear NAK 1: Clear NAK
Bit 25: 22	TXFNUM	0x0	rw	TxFIFO number Allocate FIFO number to the corresponding endpoint. A separate FIFO number is allocated to each valid IN endpoint. This bit applies to IN endpoints only.
Bit 21	STALL	0x0	rw	STALL handshake Applies to non-control, non-synchronous IN and OUT endpoints. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, global non-periodic IN NAK bit or global OUT NAK bit is set along with this bit, the STALL bit has priority. Only the application can clear this bit, but the controller never.

				0: Stall all invalid tokens 1: Stall all valid tokens
Bit 20	Reserved	0x0	resd	Kept at its default value.
Bit 19: 18	EPTYPE	0x0	rw	Endpoint type This is the transfer type supported by this logical endpoint. 00: Control 01: Synchronous 10: Bulk 11: Interrupt
Bit 17	NAKSTS	0x0	ro	NAK status Indicates the following status: 0: The controller is sending non-NAK handshakes based on the FIFO status 1: The controller is sending NAK handshakes When this bit is set (either by the application or the controller), the controller stops receiving any data on an OUT endpoint, even if there is space in the receive FIFO to accommodate the incoming data packets. For non-synchronous IN endpoints: the controller stops transmitting data on the endpoint, even if there is data pending in the transmit FIFO. For synchronous IN endpoints: the controller sends a zero-length data packet, even if there is space in the transmit FIFO. The controller always responds to SETUP data packets with an ACK handshake, regardless of whether this bit is set or not.
Bit 16	DPID/ EOFRNUM	0x0	ro	Endpoint data PID Applies to interrupt/bulk IN endpoints only. This bit contains the PID of the packet to be transmitted on this endpoint. The application must program the PID of the initial data packet to be received or transmitted on this endpoint, after the endpoint is enabled. The application programs DATA0 or DATA1 PID through the SetD1PID and SetD0PID of this register. 0: DATA0 1: DATA1 Even/Odd frame Applies to synchronous IN endpoints only. Indicates the frame number in which the controller transmits synchronous data on this endpoint. The application must program the even/odd frame number in which it tends to transmit or receive synchronous data through the SETEVNFR and SETODDFR bits in this register. 0: Even frame 1: Odd frame
Bit 15	USBACEPT	0x0	rw	USB active endpoint Indicates whether this endpoint is active in the current configuration and interface. The controller clears this bit for all endpoints except for endpoint 0 after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the application must program the endpoint registers and set this bit. 0: Inactive 1: Active
Bit 14: 11	Reserved	0x0	resd	Kept at its default value.
Bit 10: 0	MPS	0x000	rw	Maximum packet size The application uses this field to set the maximum packet size for the current logical endpoint. The values are in bytes.

21.6.5.11 OTGFS device control OUT endpoint 0 control register (OTGFS_DOEPCTL0)

This section describes the control OUT endpoint 0 control register. Non-zero control endpoints use registers for endpoints 1-7.

Bit	Register	Reset value	Type	Description
Bit 31	EPTENA	0x0	rw1s	Endpoint enable The application sets this bit to start transmitting data on endpoint 0.The controller clears this bit before setting any one of the following interrupts on this endpoint: SETUP stage done Endpoint disabled Transfer completed
Bit 30	EPTDIS	0x0	ro	Endpoint disable The application cannot disable control OUT endpoint 0.
Bit 29: 28	Reserved	0x0	resd	Kept at its default value.
Bit 27	SNAK	0x0	wo	Set NAK A write to this bit sets the NAK bit for this endpoint. The application can use this bit to control the transmission of NAK handshakes on an endpoint. The controller sets this bit on a transfer completed interrupt or when a SETUP data packet is received.
Bit 26	CNAK	0x0	wo	Clear NAK A write to this bit clears the NAK for the endpoint.
Bit 25: 22	Reserved	0x0	resd	Kept at its default value.
Bit 21	STALL	0x0	rw1s	STALL handshake The application sets this bit and the controller clears this bit when a SETUP token is received for this endpoint. If a NAK bit, global non-periodic OIT NAK bit is set along with this bit, the STALL bit has priority. The controller always responds to SETUP data packets, regardless of whether this bit is set or not.
Bit 20	SNP	0x0	rw	Snoop mode This bit configures the endpoint to Snoop mode. In this mode, the controller does not check the correctness of OUT packets before transmitting OUT packets to the application memory.
Bit 19: 18	EPTYPE	0x0	ro	Endpoint type Hardware sets this bit to 0 to control endpoint type.
Bit 17	NAKSTS	0x0	ro	NAK status Indicates the followings: 0: The controller is sending non-NAK handshakes based on the FIFO status 1: The controller is sending NAK handshakes When this bit is set (either by the application or the controller), the controller stops receiving any data on an OUT endpoint, even if there is space in the receive FIFO. The controller always responds to SETUP data packets with an ACK handshake, regardless of whether this bit is set or not.
Bit 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	USBACEPT	0x1	ro	USB active endpoint This bit is always set to 1, indicating that a control endpoint 0 is always active in all configurations and interfaces.
Bit 14: 2	Reserved	0x0000	resd	Kept at its default value.
Bit 1: 0	MPS	0x0	ro	Maximum packet size The maximum packet size of the control OUT endpoint 0 is the same as that of the control IN endpoint 0. 00: 64 bytes 01: 32 bytes 10: 16 bytes 11: 8 bytes

21.6.5.12 OTGFS device control OUT endpoint-x control register (OTGFS_DOEPCTLx) (x= x=1…7, where x if endpoint number)

This application uses this register to control the behavior of all endpoints other than endpoint 0.

Bit	Register	Reset value	Type	Description
Bit 31	EPTENA	0x0	rw1s	Endpoint enable Indicates that the descriptor structure and data buffer for data reception has been configured. The controller clears this bit before setting any one of the following interrupts on this endpoint: – SETUP stage done – Endpoint disabled – Transfer completed
Bit 30	EPTDIS	0x0	ro	Endpoint disable The application sets this bit to stop transmitting data on an endpoint, even if the transfer on that endpoint is incomplete. The application must wait for the endpoint disabled interrupt before treating the endpoint as disabled. The controller clears this bit before setting the endpoint disabled interrupt. The application must set this bit only when the endpoint enabled set. 0: No effect 1: Endpoint disabled
Bit 29	SETD1PID/ SETODDFR	0x0	rw	Set DATA1 PID Applies to interrupt/bulk OUT endpoints only. Writing to this bit sets the endpoint data PID bit in this register to DATA1. Set odd frame Applies to synchronous OUT endpoints only. Writing to this bit sets the Even/Odd frame to odd frame. 0: Disabled Set DATA1 PID disabled or Do not force odd frame 1: Set DATA1 PID enabled or forced odd frame
Bit 28	SETD0PID/ SETEVENFR	0x0	rw	Set DATA0 PID Applies to interrupt/bulk OUT endpoints only. Writing to this bit sets the endpoint data PID bit in this register to DATA0. Set Even frame Applies to synchronous OUT endpoints only. Writing to this bit sets the Even/Odd frame to even frame. 0:Disabled Set DATA0 PID disabled or Do not force even frame 1: Set DATA0PID or set the EOFRNUM to even frame
Bit 27	SNAK	0x0	wo	Set NAK A write to this bit sets the NAK bit for the endpoint. The application uses this bit to control the transmission of NAK handshakes on an endpoint. The controller sets this bit on a Transfer completed interrupt or after receiving a SETUP packet. Values: 0: Do not set NAK 1: Set NAK
Bit 26	CNAK	0x0	wo	Clear NAK A write to this bit clears the NAK bit for the endpoint. 0: Not clear NAK 1: Clear NAK
Bit 25: 22	Reserved	0x0	resd	Kept at its default value.
Bit 21	STALL	0x0	rw	Applies to non-control, non-synchronous IN and OUT endpoints. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, global non-periodic IN NAK bit or global OUT NAK bit is set along with this bit, the STALL bit has priority. Only the application can clear

				this bit, but the controller never.
Bit 20	SNP	0x0	rw	Snoop mode This bit configures the endpoint to Snoop mode. In this mode, the controller does not check the correctness of OUT packets before transmitting OUT packets to the application memory.
Bit 19: 18	EPTYPE	0x0	rw	Endpoint type This is the transfer type supported by this logical endpoint. 00: Control 01: Synchronous 10: Bulk 11: Interrupt
Bit 17	NAKSTS	0x0	ro	NAK status Indicates the followings: 0: The controller is sending non-NAK handshakes based on the FIFO status 1: The controller is sending NAK handshakes When this bit is set (either by the application or the controller), the controller stops receiving any data on an OUT endpoint, even if there is space in the receive FIFO to accommodate the incoming data packets. For non-synchronous IN endpoints: the controller stops transmitting data on the endpoint, even if there is data pending in the transmit FIFO. For synchronous IN endpoints: the controller sends a zero-length data packet, even if there is space in the transmit FIFO. The controller always responds to SETUP data packets with an ACK handshake, regardless of whether this bit is set or not.
Bit 16	DPID/ EOFRNUM	0x0	ro	Endpoint data PID Applies to interrupt/bulk OUT endpoints only. This bit contains the PID of the packet to be transmitted on this endpoint. The application must program the PID of the initial data packet to be received or transmitted on this endpoint, after the endpoint is enabled. The application programs DATA0 or DATA1 PID through the SetD1PID and SetD0PID of this register. 0: DATA0 1: DATA1 Even/Odd frame Applies to synchronous OUT endpoints only. Indicates the frame number in which the controller transmits synchronous data on this endpoint. The application must program the even/odd frame number in which it tends to transmit or receive synchronous data through the SETEVNFR and SETODDFR bits in this register. 0: Even frame 1: Odd frame
Bit 15	USBACEPT	0x0	rw	USB active endpoint Indicates whether this endpoint is active in the current configuration and interface. The controller clears this bit for all endpoints except for endpoint 0 after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the application must program the endpoint registers and set this bit. 0: Inactive 1: Active
Bit 14: 11	Reserved	0x0	resd	Kept at its default value.
Bit 10: 0	MPS	0x000	rw	Maximum packet size The application uses this field to set the maximum packet size for the current logical endpoint. The values are in bytes.

21.6.5.13 OTGFS device IN endpoint-x interrupt register (OTGFS_DIEPINTx) (x=0…7, where x if endpoint number)

This register indicates the status of an endpoint when USB and AHB-related events occurs, as shown in Figure 21-2 When the IEPINT bit of the OTGFS_GINTSTS register is set, the application must first read the OTGFS_DAINT register to get the exact endpoint number in which the event occurs, before reading the endpoint interrupt registers. The application must clear the appropriate bit in this register to clear the corresponding bits in the OTGFS_DAINT and OTGFS_GINTST registers.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7	TXFEMP	0x0	ro	Transmit FIFO empty This interrupt is generated when the transmit FIFO for this endpoint is half or completely empty. The half or completely empty status depends on the transmit FIFO empty level bit in the controller AHB configuration register.
Bit 6	INEPTNAK	0x0	rw1c	IN endpoint NAK effective This bit can be cleared by writing 1 to the CNAK bit in the DIEPCTLx register. This interrupt indicates that the IN endpoint NAB bit set by the application has taken effect. This interrupt does not guarantee that a NAK handshake is sent on the USB line. A STALL bit has priority over a NAK bit. This bit applies to the scenario only when the endpoint is enabled.
Bit 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	INTKNTXFEMP	0x0	rw1c	N token received when TxFIFO is empty Indicates that an IN token was received when the associated transmit FIFO (periodic or non-periodic) was empty. An interrupt is generated on the endpoint for which an IN token was received.
Bit 3	TIMEOUT	0x0	rw1c	Timeout condition Applies to control IN endpoints only. This bit indicates that the controller has detected a timeout condition for the last IN token on this endpoint.
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	EPTDISD	0x0	rw1c	Endpoint disabled interrupt This bit indicates that the endpoint is disabled according to the application's request.
Bit 0	XFERC	0x0	rw1c	Transfer completed interrupt Indicates that the programmed transfers are complete on the AHB and on the USB for this endpoint.

21.6.5.14 OTGFS device OUT endpoint-x interrupt register (OTGFS_DOEPINTx) (x=0…7, where x if endpoint number)

This register indicates the status of an endpoint with respect to USB and AHB-related events, as shown in Figure 21-2. When the OEPINT bit of the OTGFS_GINTSTS register is set, the application must first read the OTGFS_DAINT register to get the exact endpoint number in which the event occurs, before reading the endpoint interrupt registers. The application must clear the appropriate bit in this register to clear the corresponding bits in the OTGFS_DAINT and OTGFS_GINTST registers.

Bit	Register	Reset value	Type	Description
Bit 31: 7	Reserved	0x0000001	resd	Kept at its default value.
Bit 6	B2BSTUP	0x0	rw1c	Back-to-back SETUP packets received Indicates that more than three back-to-back SETUP packets are received.
Bit 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	OUTTEPD	0x0	rw1c	OUT token received when endpoint disabled Applies to control OUT endpoints only. Indicates that an OUT token was received when the endpoint has not yet been enabled. An interrupt is generated on the endpoint for which an OUT token was

				received.
Bit 3	SETUP	0x0	rw1c	SETUP phase done Applies to control OUT endpoints only. Indicates that the SETUP stage for the control endpoint is complete and no more back-to-back SETUP packets were received for the current control transfer. Upon this interrupt, the application can decode the received SETUP data packets.
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	EPTDISD	0x0	rw1c	Endpoint disabled interrupt Indicates that the endpoint is disabled according to the application's request.
Bit 0	XFERC	0x0	rw1c	Transfer completed interrupt Indicates that the programmed transfers are complete on the AHB and on the USB for this endpoint.

21.6.5.15 OTGFS device IN endpoint 0 transfer size register (OTGFS_DIEPTSIZ0)

The application must set this register before enabling endpoint 0. Once the endpoint 0 is enabled using the endpoint enable pin in the device endpoint 0 control register, the controller modifies this register. The application can only read this register as long as the controller clears the endpoint enable bit.

Bit	Register	Reset value	Type	Description
Bit 31: 21	Reserved	0x000	resd	Kept at its default value.
Bit 20: 19	PKTCNT	0x0	rw	Packet count Indicates the total number of USB packets that constitute the transfer size of data for the endpoint 0. This field is decremented every time a packet is read from the transmit FIFO (maximum packet size or short packet)
Bit 18: 7	Reserved	0x000	resd	Kept at its default value.
Bit 6: 0	XFERSIZE	0x00	rw	Transfer size Indicates the transfer size (in bytes) for the endpoint 0. The controller interrupts the application when the transfer size becomes 0. The transfer size can be set to the maximum packet size of the endpoint at the end of each packet. The controller decrements this field every time a packet from the external memory is written to the transmit FIFO.

21.6.5.16 OTGFS device OUT endpoint 0 transfer size register (OTGFS_DOEPTSIZ0)

The application must set this register before enabling endpoint 0. Once the endpoint 0 is enabled using the endpoint enable pin in the device endpoint 0 control register, the controller modifies this register. The application can only read this register as long as the controller clears the endpoint enable bit.

Bit	Register	Reset value	Type	Description
Bit 31	Reserved	0x0	resd	Kept at its default value.
Bit 30: 29	SUPCNT	0x0	rw	SETUP packet count Indicates the number of back-to-back SETUP data packets the endpoint can receive. 01: 1 packet 10: 2 packets 11: 3 packets
Bit 28: 20	Reserved	0x000	resd	Kept at its default value.
Bit 19	PKTCNT	0	rw	Packet count This bit is decremented to 0 after a packet is written to the receive FIFO.
Bit 18: 7	Reserved	0x000	resd	Kept at its default value.
Bit 6: 0	XFERSIZE	0x00	rw	Transfer size Indicates the transfer size (in bytes) for the endpoint 0. The controller interrupts the application when the transfer size becomes 0. The transfer size can be set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The controller decrements this field every time a packet

				from the external memory is written to the transmit FIFO. The controller decrements this field every time a packet from the receive FIFO is written to the external memory.

21.6.5.17 OTGFS device IN endpoint-x transfer size register (OTGFS_DIEPTSIZx) (x=1…7, where x is endpoint number)

The application must set this register before enabling endpoint x. Once the endpoint x is enabled using the endpoint enable pin in the device endpoint x control register, the controller modifies this register. The application can only read this register as long as the controller clears the endpoint enable bit.

Bit	Register	Reset value	Type	Description
Bit 31	Reserved	0x0	resd	Kept at its default value.
Bit 30: 29	MC	0x0	rw	Multi count For periodic IN endpoints, this field indicates the number of packets to be transmitted on the USB for each frame. The controller uses this field to calculate the data PID transmitted on synchronous IN endpoints. 01: 1 packet 10: 2 packets 11: 3 packets
Bit 28: 19	PKTCNT	0x000	rw	Packet count Indicates the total number of USB packets (data transfer size on the endpoint) this field is decremented every time a packet is read from the transmit FIFO (maximum packet size and short packet).
Bit 18: 0	XFERSIZE	0x00000	rw	Transfer Size Indicates the transfer size (in bytes) for the current endpoint. The controller interrupts the application when the transfer size becomes 0. The transfer size can be set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The controller decrements this field every time a packet from the external memory is written to the transmit FIFO.

21.6.5.18 OTGFS device IN endpoint transmit FIFO status register (OTGFS_DTXFSTSx) (x=1…7, where x is endpoint number)

This is a ready-only register containing the free space information for the device IN endpoint transmit FIFO.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	INEPTXFSAV	0x0200	ro	IN endpoint TxFIFO space available Indicates the amount of free space in the endpoint transmit FIFO. Values are in terms of 32-bit words. 0x0: Endpoint transmit FIFO is full 0x1: 1 word available 0x02: 2 words available 0xn: n words available (0 < n < 512) 0x200: Remaining 512 words Others: Reserved

21.6.5.19 OTGFS device OUT endpoint-x transfer size register (OTGFS_DOEPTSIZx) (x=1…7, where x is endpoint number)

The application must set this register before enabling endpoint x. Once the endpoint x is enabled using the endpoint enable pin in the device endpoint x control register, the controller modifies this register. The application can only read this register as long as the controller clears the endpoint enable bit.

Bit	Register	Reset value	Type	Description
Bit 31	Reserved	0x0	resd	Kept at its default value.
Bit 30: 29	RXDPID	0x0	ro	Received data PID Applies to synchronous OUT endpoints only. This is the data PID received in the last packet. 00: DATA0 01: DATA2 10: DATA1 11: MDATA SETUP packet count Applies to synchronous OUT endpoints only. Indicates the number of back-to-back SETUP data packets the endpoint can receive. 01: 1 packet 10: 2 packets 11: 3 packets
Bit 28: 19	PKTCNT	0x000	rw	Packet count Indicates the number of USB packets transmitted on the endpoint. This field is decremented every time a packet is written to the receive FIFO (maximum packet size and short packet)
Bit 18: 0	XFERSIZE	0x00000	rw	Transfer size Indicates the transfer size (in bytes) for the current endpoint. The controller interrupts the application when the transfer size becomes 0. The transfer size can be set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The controller decrements this field every time a packet is read from the receive FIFO and written to the external memory.

21.6.6 Power and clock control registers

21.6.6.1 OTGFS power and clock gating control register (OTGFS_PCGCCTL)

This register is available in host and device modes.

Bit	Register	Reset value	Type	Description
Bit 31: 5	Reserved	0x0000000	resd	Kept at its default value.
Bit 4	SUSPENDM	0x0	ro	PHY suspend Indicates that the PHY has been suspended.
Bit 3: 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	STOPPCLK	0x0	rw	Stop PHY clock The application uses this bit to stop PHY clock when the USB is suspended, session is invalid or device is disconnected. The application clears this bit when the USB is resumed or a new session starts.

22 HICK auto clock calibration (ACC)

22.1 ACC introduction

HICK auto clock calibration (HICK ACC), which uses the SOF signal (1 ms of period) generated as a reference signal, implements the sampling and calibration for the HICK clocks.

The main purpose of this module is to provide a clock of 48MHz±0.25% for the USB device.

It is able to make the calibrated frequency as close to the target frequency as possible by means of “cross and return” algorithm.

22.2 Main features

- Programmable center frequency
- Programmable boundary frequency that triggers calibration function
- Center frequency precision ±0.25%
- Status detection flags
  - Calibration ready flag
- Error detection flags
  - Reference signal lost error flag
- Two interrupt source flag
  - Calibration ready flag
  - Reference signal lost error flag
- Two calibration modes: coarse calibration and fine calibration

22.3 Interrupt requests

Table 22-1 ACC interrupt requests

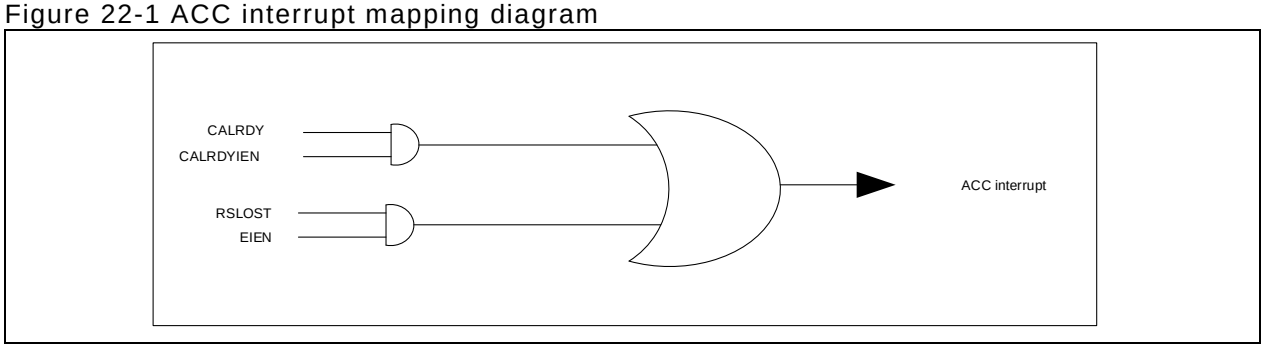
Interrupt event	Event flag	Enable bit
Calibration ready	CALRDY	CALRDYIEN
Reference signal lost	RSLOST	EIEN

ACC interrupt events are linked to the same interrupt vector (see Figure 25-1). Interrupt events include:

- During calibration process: When the calibration gets ready or reference signal lost occurs,

the corresponding interrupt will be generated if the corresponding enable bit is enabled.

Figure 22-1 ACC interrupt mapping diagram



22.4 Functional description

Auto clock calibration (HICK ACC), which uses the SOF signal (1 ms of period) generated as a reference signal, implements the sampling and calibration for the HICK clocks. In particular, the HICK clock frequency can be calibrated to a precision of ±0.25% so as to meet the needs of the high-precision clock applications such as USB.

The signals of the module are connected to the CRM and HICK inside the microcontroller instead of being connected to the pins externally.

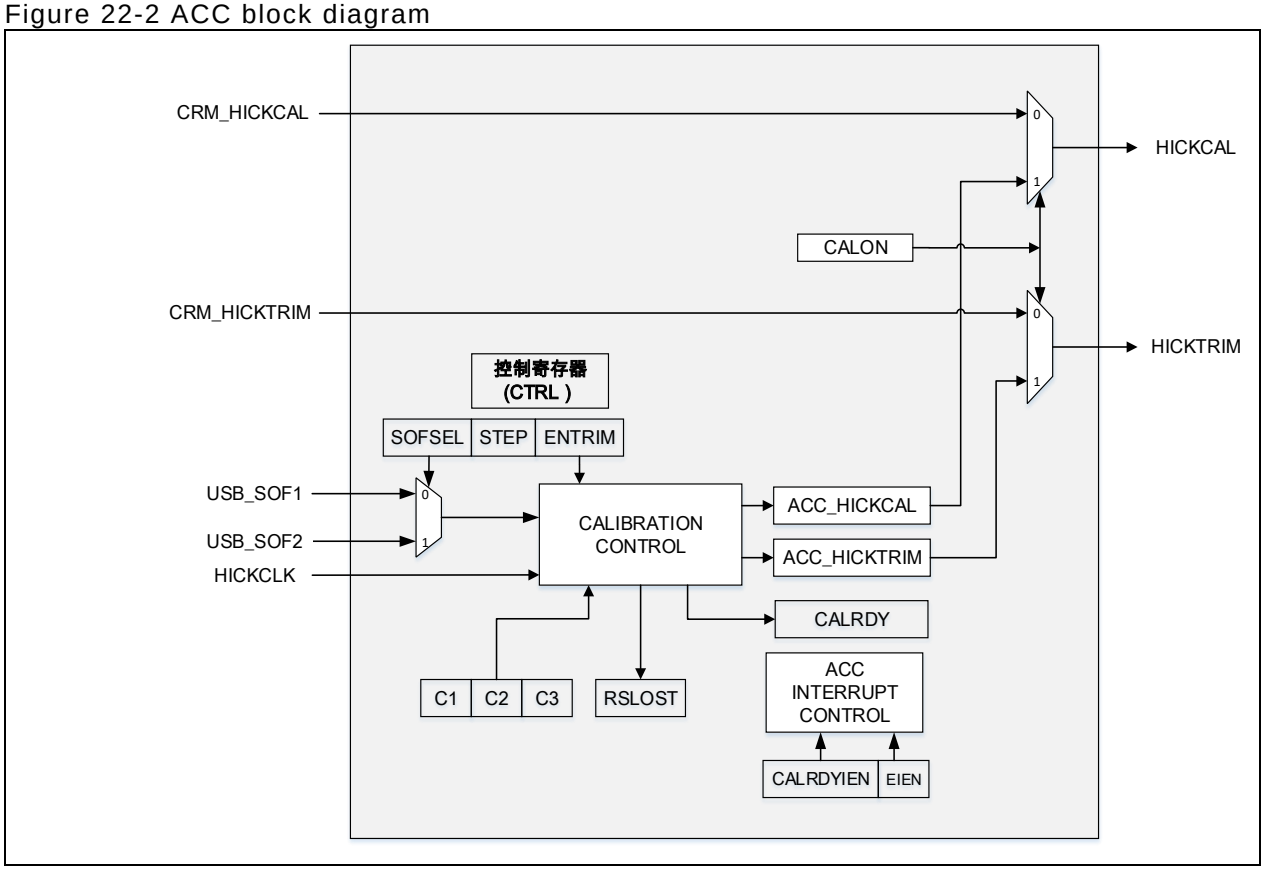
- CRM_HICKCAL: the HICKCAL bit in the CRM module. This signal is used to calibrate the HICK in bypass mode. The value is defined by the HICKCAL[7: 0] in the CRM_CTRL register.
- CRM_HICKTRIM: the HICKTRIM bit in the CRM module. This signal is used to calibrate the HICK in bypass mode. The value is defined by the HICKTRIM[5: 0] in the CRM_CTRL register.

The default value of the HICK is 32, which can be calibrated to 8MHz±0.25%. The HICK frequency can be adjusted by 20kHz (design value) each time when the CRM_HICKTRIM value changes. In other words, the HICK output frequency will increase by 20kHz each time the CRM_HICKTRIM value is decremented by one; the HICK output frequency will reduce by 20kHz each time the CRM_HICKTRIM value is decremented by one.

- USB_SOF: USB Start-of-Frame signal given by the USB device. Its high-level width is 12 system clock cycles, a pulse signal of 1 ms.
- HICKCLK: HICK clock. The original HICK output frequency is 48MHz, but the sampling clock used by the HICK calibration module is frequency divider (1/6) clock, about 8MHz.
- HICKCAL: HICK module calibration signal. For the HICK clock after frequency division (1/6), the HICK clock frequency will change by 40KHz (design value) each time the HICKCAL changes, which is positively correlated. In other words, the HICK clock frequency will increase by 40KHz (design value) each time the HICKCAL is incremented by one; the HICK clock frequency will reduce by 40KHz each time the HICKCAL is decremented by one.
- HICKTRIM: HICK module calibration signal. For the HICK clock after frequency division (1/6), the HICK clock frequency will change by 20KHz (design value) each time the HICKCAL changes, which is positively correlated.

Refer to Section 22.6 for more information about the bit definition in the registers.

Figure 22-2 ACC block diagram



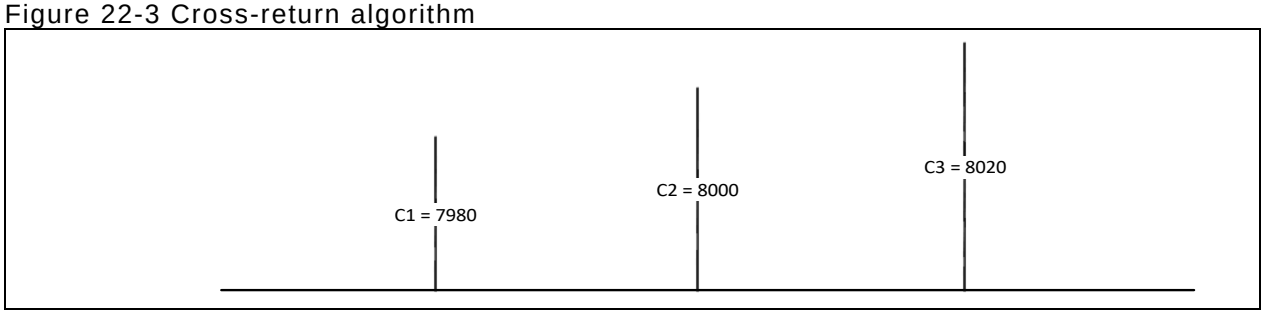
22.5 Principle

USB_SOF period signal: 1ms of period must be accurate, which is a prerequisite of the normal operation of an auto calibration module.

cross-return algorithm: This is used to calculate a calibration value closest to the theoretic value. In theory, the actual frequency after calibration can be adjusted to be within an accuracy range of about

0.5 steps from the target frequency (8MHz)

Figure 22-3 Cross-return algorithm



From the above figure, auto calibration function will adjust the HICKCAL or HICKTRIM according to the specified step as soon as the condition for triggering auto calibration is reached.

Cross:

If the auto calibration condition is met, the actual sampling data in the first 1ms period will be either less than C2, or greater than C2.

When this value is less than C2, the auto calibration module will start increasing either the HICKCAL or HICKTRIM according to the step definition until the actual sampling value is greater than C2. In this way, the actual value will cross over C2 from small to large.

When this value is greater than C2, the auto calibration module will start decrease either the HICKCAL or HICKTRIM according to the step definition until the actual sampling value become less than C1. In this way, the actual value will cross over C2 from large to small.

Return:

After cross operation is completed, the actual value closest to C2 can be obtained by comparing the difference (calculated as absolute value) between the actual sampling value and C2 before and after crossing C2 so as to get the best calibration value HICKCAL or HICKTRIM.

If the difference after crossing is less than the one before crossing C2, the calibration value after crossing prevails, and stops the calibration process until the next condition for auto calibration appears.

If the difference after crossing is greater than the one before crossing C2, the calibration value before crossing prevails, and it will return by one step to the one before crossing, and stops the calibration process until the next condition for auto calibration appears.

According to the cross-return strategy, in theory, it is possible to get the frequency accuracy that is 0.5 steps away from the center frequency.

Four conditions for enabling auto calibration function are as follows:

1. The rising edge of the CANLON (from 0 to 1)
2. When CALON=1, reference signal is lost and restored
3. When the sample counter is less than C1
4. When the sample counter is greater than C3

Even though the sampling counter is between C1 and C3, at the rising edge the CANLON, the auto calibration module can also be activated so that the HICK frequency can be adjusted to be within a range of 0.5 steps of the center frequency as soon as the CANLON is enabled.

Under one of the above-mentioned circumstances, the HICK frequency can be calibrated to be within 0.5 steps of the center frequency. To achieve the best calibration accuracy, it is recommended to remain step as 1 (default value). If the step is set to 0, either HICKCAL or HICKTRIM will not be able to be calibrated.

22.6 Register description

Refer to the list of abbreviations used in register descriptions.

These peripheral registers must be accessed by words (32 bits).

22.6.1 ACC register map

Table 22-2 ACC register map and reset values

Register name	Offset	Reset value
ACC_STS	0x00	0x0000 000
ACC_CTRL1	0x04	0x0000 0100
ACC_CTRL2	0x08	0x0000 2080
ACC_C1	0x0C	0x0000 1F2C
ACC_C2	0x10	0x0000 1F40
ACC_C3	0x14	0x00000 1F54

22.6.2 Status register (ACC_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 2	Reserved	0x0000000	resd	Kept at its default value.
Bit 1	RSLOST	0x0	ro	Reference Signal Lost 0: Reference Signal is not lost 1: Reference Signal is lost Note: During the calibration, when the sample counter of the calibration module is twice that of C2, if a SOF reference signal is not detected, it means that the reference signal is lost. The internal statue machine will move to the idle state unless another SOF signal is detected, otherwise, the internal clock sample counter remains 0. The RSLOST bit is immediately cleared after the CALON bit is cleared or when the RSLOST is written with 0. Reference signal detection occurs only when CALON=1.
Bit 0	CALRDY	0x0	ro	Internal high-speed clock calibration ready 0: Internal 8MHz oscillator calibration is not ready 1: Internal 8MHz oscillator calibration is ready Note: This bit is set by hardware to indicate that internal 8MHz oscillator has been calibrated to the frequency closest to 8MHz. The CALRDY is immediately cleared after the CALON bit is cleared or when the CALRDY is written with 0.

22.6.3 Control register 1 (ACC_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Forced by hardware to 0
Bit 11: 8	STEP	0x1	rw	Calibrated step This field defines the value after each calibration. Note: It is recommended to set the step bit in order to get a more accurate calibration result. While ENTRIM=0, only the HICKCAL is calibrated. If the step is incremented or decremented by one, the HICKCAL will be incremented or decremented by one accordingly, and the HICK frequency will increase or decrease by 40KHz (design value). This is a positive relationship. While ENTRIM=1, only the HICKTRIM is calibrated. If the step is incremented or decremented by one, the HICKTRIM will be incremented or decremented by one accordingly, and the HICK frequency will increase or decrease by 20KHz (design value). This is a positive relationship.
Bit 7: 6	Reserved	0x0	rw	Forced by hardware to 0
Bit 5	CALRDYIEN	0x0	rw	CALRDY interrupt enable This bit is set or cleared by software.

				0: Interrupt generation disabled 1: ACC interrupt is generated when CALRDY=1 in the ACC_STS register
Bit 4	EIEN	0x0	rw	RSLOST error interrupt enable This bit is set or cleared by software. 0: Interrupt generation disabled 1: ACC interrupt is generated when RSLOST=1 in the ACC_STS register
Bit 3: 2	Reserved	0x0	rw	Forced by hardware to 0
Bit 1	ENTRIM	0x0	rw	Enable trim This bit is set or cleared by software. 0: HICKCAL is calibrated. 1: HICKTRIM is calibrated. Note: It is recommended to set ENTRIM=1 in order to get higher calibration accuracy.
Bit 0	CALON	0x0	rw	Calibration on This bit is set or cleared by software. 0: Calibration disabled 1: Calibration enabled, and starts searching for a pulse on the USB_SOF. Note: This module cannot be used without the USB_SOF reference signal. If there are no requirements on the accuracy of the HICK clock, it is unnecessary to enable this module.

22.6.4 Control register 2 (ACC_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 31: 14	Reserved	0x00000	resd	Forced to 0 by hardware
Bit 13: 8	HICKTRIM	0x20	ro	Internal high-speed auto clock trimming This field is read only, but not written. Internal high-speed clock is adjusted by ACC module, which is added to the ACC_HICKCAL[7: 0] bit. These bits allow the users to input a trimming value to adjust the frequency of the HICKRC oscillator according to the variations in voltage and temperature. The default value is 32, which can trim the HICK to 8MHz±0.25%. The trimming value is 20kHz (design value) between two consecutive ACC_HICKTRIM steps.
Bit 7: 0	HICKCAL	0x80	ro	Internal high-speed auto clock calibration This field is read only, but not written. Internal high-speed clock is adjusted by ACC module. These bits allow the users to input a trimming value to adjust the frequency of the HICKPC oscillator according to the variations in voltage and temperature. The default value is 128, which can trim the HICK to 8MHz±0.25%. The trimming value is 40kHz (design value) between two consecutive ACC_HICKCAL steps.

22.6.5 Compare value 1 (ACC_C1)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Forced to 0 by hardware
Bit 15: 0	C1	0x1F2C	rw	Compare 1 This value is the lower boundary for triggering calibration, and its default value is 7980. When the number of clocks sampled by ACC in 1ms period is less than or equal to C1, auto calibration is triggered automatically. When the actual sampling value (number of clocks in 1ms) is greater than C1 but less than C3, auto calibration is not enabled.

22.6.6 Compare value 2 (ACC_C2)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Forced to 0 by hardware
Bit 15: 0	C2	0x1F40	rw	Compare 2 This value defines the number of clocks sampled for 8MHz (ideal frequency) clock in 1ms period , and its default value is 8000 (theoretical value). As a center point of cross-return strategy, this value is used to calculate the calibration value closest to the theoretical value. In theory, the actual frequency after calibration can be trimmed to be within an accuracy of 0.5 steps from the target frequency (8MHz).

22.6.7 Compare value 3 (ACC_C3)

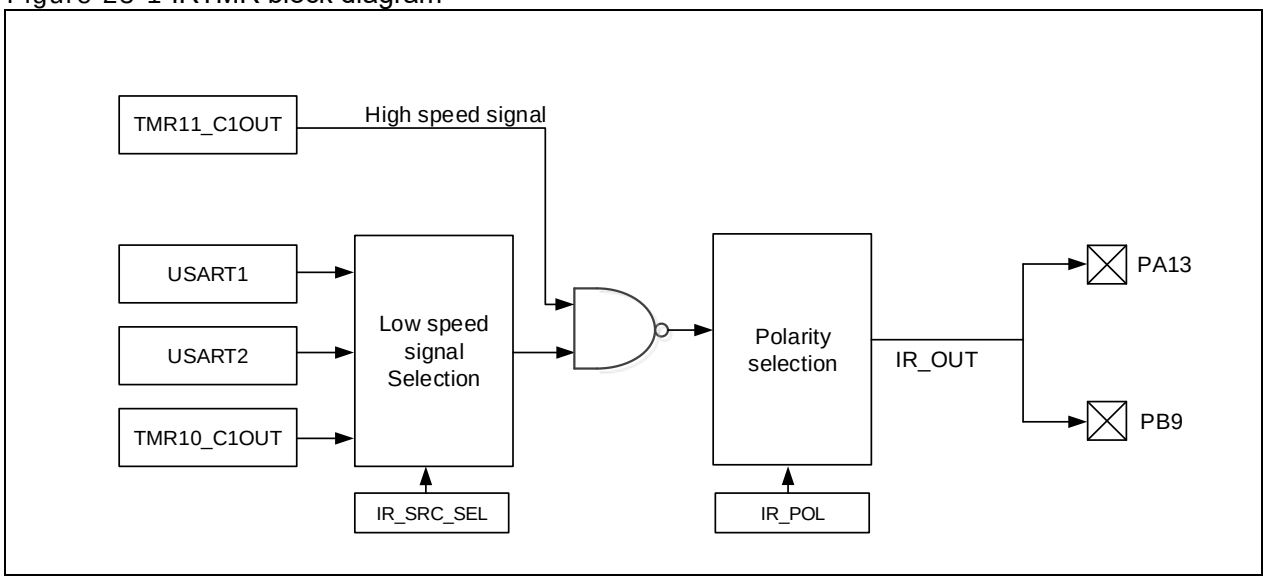
Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Forced to 0 by hardware
Bit 15: 0	C3	0x1F54	rw	Compare 3 This value is the upper boundary for triggering calibration. When the number of clock sampled by ACC in 1ms period is greater than or equal to C3, auto calibration is triggered automatically. When the actual sampling value (number of clocks in 1ms period) is greater than C1 but less than C3, auto calibration is not enabled.

23 Infrared timer (IRTMR)

The IRTMR (Infrared Timer) is used to generate the IR_OUT signal that drives the infrared LED so as to achieve infrared control.

The IR_OUT signals consists of a low-frequency modulation envelope and high-frequency carrier signals. The low-frequency modulation envelope signal selects from TMR10_C1OUT, USART1 and USART through the IR_SRC_SEL[1: 0] bit in the SCFG_CFG1 register, while the high-frequency carrier signal is provided by the TMR11_C1OUT register. The IR_POL bit in the SCFG_CFG1 register controls whether the IR_OUT output is reversed or not. The IR_OUT signal is output through multiplexed function via PB9 or PA13 (multiplexed mode needs to be configured in advance).

Figure 23-1 IRTMR block diagram



24 External memory controller (XMC)

24.1 XMC introduction

XMC peripheral block can translate the AHB signals into the external memory signals and vice versa. It boasts two chip-select signals for interfacing up to external memories at a time. The supported external memories include a NAND Flash and a static memory device featuring multiplexed signals or additional address latch function. Such static memory device includes a static random memory (SRAM), NOR Flash and PSRAM.

24.2 XMC main features

NOR/PSRAM has the following features:

- Chip-select signal supports 4 external memories, each of which has their own control register
- Support access to static memory devices, including
  - Static random access memory (SRAM)
  - NOR Flash
  - PSRAM
- 8-bit or 16-bit wide memory
- Various timing mode selection
  - Two modes with the same timings for read and write
  - Four modes with different timings for read and write
  - Multiplexed address/data mode
  - Synchronous mode
- Programmable timing control registers
- Translate the AHB data size into the appropriate external memory data size

NAND has the following features:

- Chip-select signal supports 2 external memories
- 8-bit or 16-bit wide NAND Flash
- Two storage spaces with programmable timing control registers
- Translate the AHB data size into the appropriate external memory data size
- Support ECC calculation

PC card has the following features:

- Chip-select signal supports 1 external memory
- 16-bit wide PC card
- Three storage spaces with programmable timing control registers
- Translate the AHB data size into the appropriate external memory data size

SDRAM card has the following features:

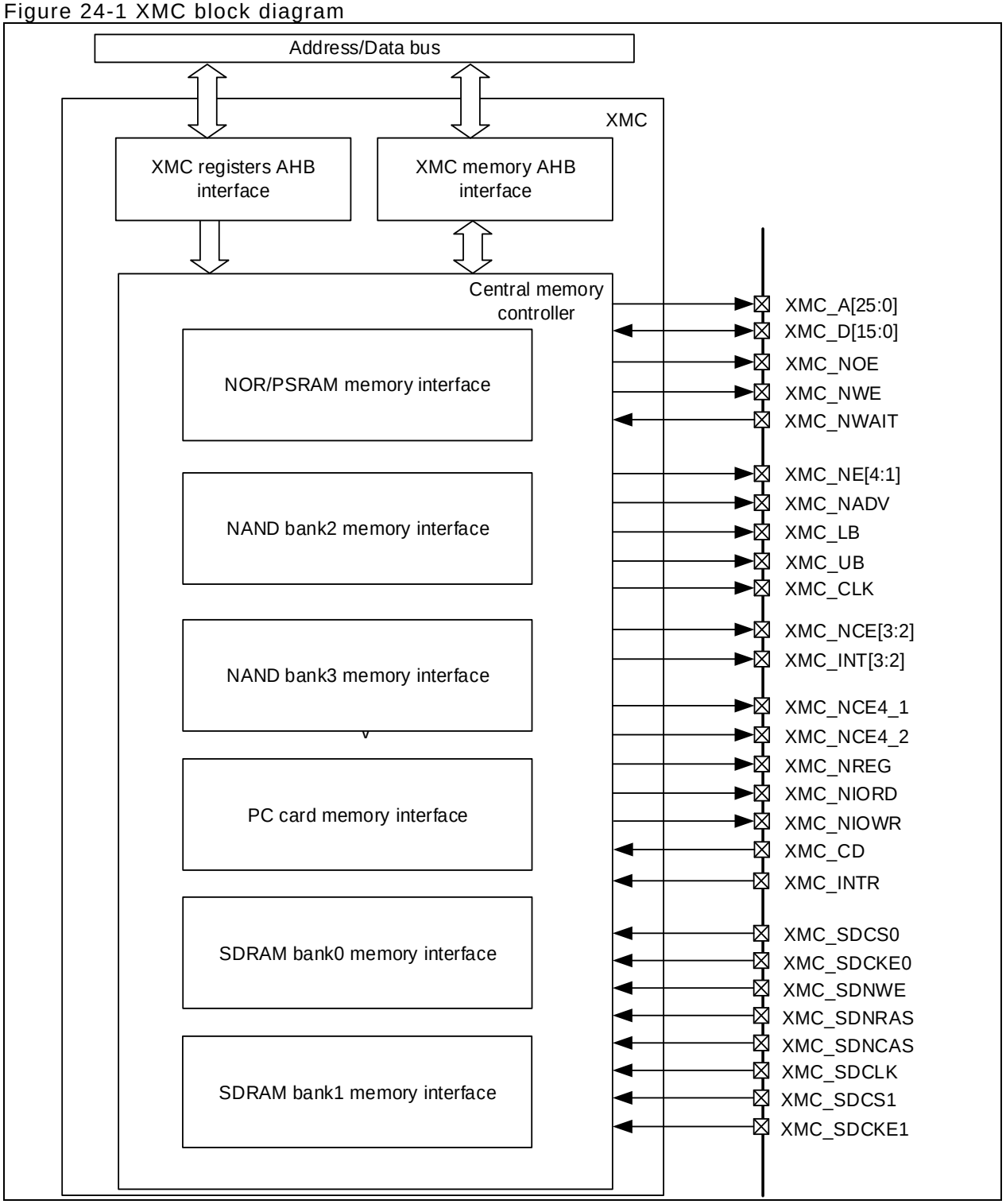
- Connected to 2 independent SDRAM devices externally
- 8-bit or 16-bit wide SDRAM data bus
- Supports up to 13-bit address row, 11-bit address column and 4 internal memory areas (4x16Mx16bit :128 MB and 4x16Mx8bit: 64 MB)
- SDRAM clock sources: HCLK/2, HCLK/3 or HCLK/4
- Supports AHB byte, halfword and word accesses
- Programmable SDRAM access timing parameters
- Automatic row and memory boundary management

- Multi-memory area ping-pong access
- Automatic refresh operation of software-programmable refresh rate
- Supports self-refresh mode
- Supports power-down mode
- SDRAM power-on initialization through software
- CAS latency can be configured 1/2/3 SDRAM clock cycles
- Cacheable Read FIFO with 6x 32-bit depth

24.3 XMC architecture

24.3.1 Block diagram

Figure 24-1 XMC block diagram



While interfacing to the external memory, NOR/PSRAM, NAND, PC card and SDRAM uses different pins respectively, as shown in Table 24-1, Table 24-2 and Table 24-3.

Table 24-1 NOR/PSRAM pins

Pin name	I/O	Description
XMC_CLK	Output	Clock
XMC_NE[x], x=1,4	Output	Chip select
XMC_NADV	Output	Address latch or address valid (NL) signal

XMC_A[x]	Output	Address bus
XMC_NOE	Output	Output enable signal
XMC_NWE	Output	Write enable signal
XMC_LB and XMC_UB	Output	Byte select signal
XMC_D[15: 0]	Read input/write output	Data bus/multiplexed address data
XMC_NWAIT	Input	Wait signal

Table 24-2 NAND pins

Pin name	I/O	Description
XMC_NCE[2]	Output	Chip select
XMC_A[17]	Output	Address latch (ALE) signal
XMC_A[16]	Output	Command latch (CLE) signal
XMC_NOE	Output	Output enable (NRE) signal
XMC_NWE	Output	Write enable signal
XMC_D[15: 0]	Read input/write output	Data bus
XMC_NWAIT	Input	Ready/Busy (R/B)

Table 24-3 PC card pins

Pin name	I/O	Description
XMC_NCE4_1	Output	Chip select 1 (CE1)
XMC_NCE4_2	Output	Chip select 2 (CE2)
XMC_A[10:0]	Output	Address bus
XMC_NOE	Output	Output enable signal for general-purpose space and attribute
XMC_NWE	Output	Write enable signal for general-purpose space
XMC_NIORD	Output	Output enable signal for IO space
XMC_NIOWR	Output	Write enable signal for IO space
XMC_NREG	Output	Attribute space select signal
XMC_D[15: 0]	Read input/write output	Data bus
XMC_CD	Input	PC card detection signal, active high
XMC_NWAIT	Input	Ready/Busy (R/B)
XMC_INTR	Input	PC card interrupt signal

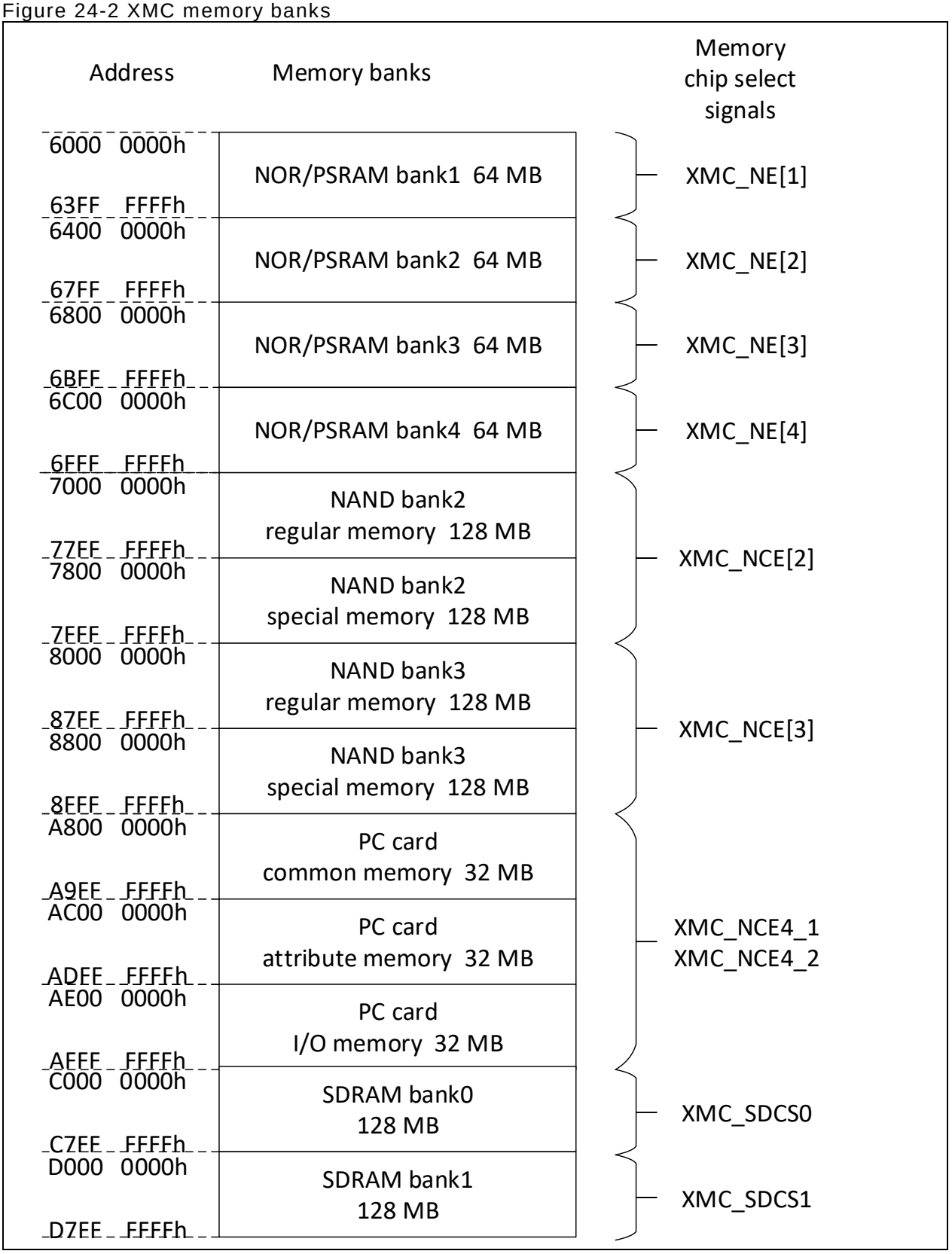
Table 24-4 SDRAM pins

Pin name	I/O	Description
XMC_SDCKE0	Output	DEVICE0 clock enable
XMC_SDCKE1	Output	DEVICE1 clock enable
XMC_SDCS0	Output	DEVICE0 chip-select
XMC_SDCS1	Output	DEVICE1 chip-select
XMC_SDCLK	Output	Clock signal
XMC_SDNRAS	Output	Row select
XMC_SDNCAS	Output	Column select
XMC_SDNWE	Output	Write enable
XMC_LB, XMC_UB	Output	Byte select
XMC_A[12:0]	Output	Address bus
XMC_A[14]	Output	BANK address low
XMC_A[15]	Output	B BANK address high
XMC_D[15:0]	Read input/write output	Data bus

24.3.2 Address mapping

XMC address is divided into multiple memory banks, as shown below.

Figure 24-2 XMC memory banks



Some HADDR bits are used to select which bank to access to, as shown in Table 24-3.

Table 24-5 Memory bank selection

HADDR[31: 28]	HADDR[27: 26]	
0110: NOR/PSRAM	00: bank1	
	11: bank4	
HADDR[31: 28]	**HADDR[27]**	**HADDR[17: 16]**
0111: NAND bank2	0: Regular space	00: Data area
		01: Command area
		1x: Address area
	1: Special space	00: Data area
		01: Command area
		1x: Address area
1000: NAND bank3	0: Regular space	00: Data area
		01: Command area
		1x: Address area
	1: Special space	00: Data area
		01: Command area
		1x: Address area
HADDR[31: 28]	**HADDR[26: 25]**	
1010: PC card	00: General-purpose memory space	
	10: Attribute memory space	
	11: I/O space	
HADDR[31: 28]	**HADDR[26: 0]**	
1100: SDRAM BANK1	BANK and row/column address mapping	
HADDR[31: 28]	**HADDR[26: 0]**	
1101: SDRAM BANK2	BANK and row/column address mapping	

Table 24-6 8-bit SDRAM address mapping

Row size	HADDR (AHB internal address line)																										
	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
11-bit	Reserved							Bank [1:0]		Row[10:0]											Column[7:0]							
	Reserved								Bank [1:0]		Row[10:0]										Column[8:0]							
	Reserved									Bank [1:0]		Row[10:0]									Column[9:0]							
	Reserved										Bank [1:0]		Row[10:0]								Column[10:0]							
12-bit	Reserved									Bank [1:0]		Row[11:0]									Column[7:0]							
	Reserved										Bank [1:0]		Row[11:0]								Column[8:0]							
	Reserved											Bank [1:0]		Row[11:0]							Column[9:0]							
	Reserved												Bank [1:0]		Row[11:0]						Column[10:0]							
13-bit	Reserved										Bank [1:0]		Row[12:0]								Column[7:0]							
	Reserved											Bank [1:0]		Row[12:0]							Column[8:0]							
	Reserved												Bank [1:0]		Row[12:0]						Column[9:0]							
	Reserved													Bank [1:0]		Row[12:0]					Column[10:0]							

Table 24-7 16-bit SDRAM address mapping

Row size	HADDR (AHB address line) 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
11-bit	Reserved | Bank [1:0] | Row[10:0] | Column[7:0]
	Reserved | Bank [1:0] | Row[10:0] | Column[8:0]
	Reserved | Bank [1:0] | Row[10:0] | Column[9:0]
	Reserved | Bank [1:0] | Row[10:0] | Column[10:0]
12-bit	Reserved | Bank [1:0] | Row[11:0] | Column[7:0]
	Reserved | Bank [1:0] | Row[11:0] | Column[8:0]
	Reserved | Bank [1:0] | Row[11:0] | Column[9:0]
	Reserved | Bank [1:0] | Row[11:0] | Column[10:0]
13-bit	Reserved | Bank [1:0] | Row[12:0] | Column[7:0]
	Reserved | Bank [1:0] | Row[12:0] | Column[8:0]
	Reserved | Bank [1:0] | Row[12:0] | Column[9:0]
	Res. | Bank [1:0] | Row[12:0] | Column[10:0]

24.4 NOR/PSRAM

NOR/PSRAM offers multiple access modes with different timings to drive multiple memories including NOR Flash, SRAM, PSRAM and Cellular RAM.

There are two banks, bank 1 and bank, with independent control registers. Such two banks can be accessed by means of different timings and different chip-select signals.

24.4.1 Operating mode

Pin function:

Pin signals vary from external memory to external memory. Table 24-8 lists typical pin signals.

Table 24-8 Pin signals for NOR and PSRAM

XMC pin name	NOR Flash	PSRAM
XMC_CLK	Clock (synchronous mode)	Clock (synchronous mode)
XMC_NE[x]	Chip-select	Chip-select
XMC_NADV	Address latch or address valid	Address latch or address valid
XMC_A[23:16], XMC_A[0]	Address bus	Address bus
XMC_NOE	Output enable	Output enable
XMC_NWE	Write enable	Write enable
XMC_LB, XMC_UB	Without using XMC_LB, XMC_UB	XMC_LB: lower byte XMC_UB: upper byte
XMC_D[15: 0]	Data bus multiplexed address data bus (multiplex and synchronous mode)	Data bus multiplexed address data bus (multiplex and synchronous mode)
XMC_NWAIT	NOR Flash wait request	PSRAM wait request

Note: If the memory data size is 8-bit, the typical data bus is XMC_D[7: 0].

Access address

The upper bytes of the HADDR bit is used to select a memory bank while the lower bytes to data memory address. HADDR is a byte address whereas the XMC supports the memory addressed in words or half words. Address translation between them is shown in Table 22-5. As long as read/write access to a specific address occurs, the XMC will enable chip-select signals and write/read operation to the external memories according to the HADDR bit.

Table 24-9 Address translation between HADDR and external memory

External memory data width	Address connection	Accessible maximum memory space (bits)
8-bit	HADDR[25: 0] is linked to XMC_A[25: 0]. In multiplexed and synchronous mode, HADDR[15: 0] is connected to XMC_D[15: 0] during address latch period.	64 Mbyte x8 =512 Mbit
16-bit	HADDR[26: 1] is connected to XMC_A[25: 0]. In multiplexed and synchronous mode, HADDR[16: 1] is connected to XMC_D[15: 0] during address latch period	(64 Mbyte x 16)/2=512 Mbit

Data access

In case that the AHB data width is not equal to that of the memories, the XMC will make appropriate arrangement according to the typical signals of the external memories. Table 22-6 lists the operation modes supported by XMC.

Table 24-10 Data access width vs. external memory data width

Memory	Mode	AHB data width	Memory width	Description
SRAM	Asynchronous read/write	8/16/32	8	One-time access, or split into 2 or 4 accesses
	Asynchronous read/write	8/16/32	16	XMC_LB and XMC_UB, One-time, or split into two access
NOR Flash	Asynchronous read	8	16	
	Asynchronous read/write	16	16	
	Asynchronous read/write	32	16	Split into 2 XMC accesses
	Synchronous read	16	16	
	Synchronous read	32	16	Split into 2 XMC accesses
PSRAM	Asynchronous read	8	16	
	Asynchronous write	8	16	Use XMC_LB and XMC_UB
	Asynchronous read/write	16	16	
	Asynchronous read/write	32	16	Split into 2 XMC accesses
	Synchronous write	8	16	Use XMC_LB and XMC_UB
	Synchronous read/write	16	16	
	Synchronous read/write	32	16	Split into 2 XMC accesses

24.4.2 Access mode

The XMC offers various access modes. Each access mode is operated based on the timing parameters, as shown in Table 24-11.Users can perform programming operations according to the specifications of the external memory and application needs.

Access modes available in the XMC:

- Read/write operation with the same timings: Mode 1 and Mode 2
- Read/write operation with different timings: Mode A, B, C and D
- Multiplexed address data lines
- Clock-based synchronous mode

Table 24-11 NOR/PSRAM parameter registers

Parameter register	Function	Access mode	Unit
ADDRST	Address set-up time	1, 2, A, B, C, D and multiplexed	HCLK cycle
ADDRHT	Address-hold time	D and multiplexed	HCLK cycle
DTST	Data set-up time	1, 2, A, B, C, D and multiplexed	HCLK cycle
DTLAT	Data latency time	Synchronous	XMC_CLK cycle
CLKPSC	Clock prescaler	Synchronous	HCLK cycle

In addition to timing parameter registers for timing control, if the wait enable bit (NWASEN or NWSEN) is enabled, the XMC will start to check whether the XMC_NWAIT signal is in wait request state during data set time. If so, the XMC will wait until the XMC_NWAIT returns to the ready state before data transfer.

24.4.2.1 Read/write operation with the same timings

The timing of read and write operation in mode 1 and mode 2 is based on the XMC_BK1TMG register configuration.

Mode 1

As configured in Table 24-12 and Table 24-13, the XMC uses mode 1 to access the external memory. The timing of read operation is shown in Figure 24-3. The timing of write operation is shown in Figure 24-4.

Table 24-12 Mode 1— SRAM/NOR Flash chip select control register (XMC_BK1CTRL) configuration

Bit	Description	Configuration
Bit 31: 20	Reserved	0x0
Bit 19	MWMC: Memory write mode control	0x0
Bit 18: 16	CRPGS: CRAM page size	0x0
Bit 15	NWASEN: NWAIT in asynchronous transfer enable	Configure according to memory specifications
Bit 14	RWTD: Read-write timing different	0x0
Bit 13	NWSEN: NWAIT in synchronous transfer enable	0x0
Bit 12	WEN: Write enable	Configure according to needs
Bit 11	NWTCFG: NWAIT timing configuration	0x0
Bit 10	WRAPEN: Wrapped enable	0x0
Bit 9	NWPOL: NWAIT polarity	Configure according to memory specifications
Bit 8	SYNCBEN: Synchronous burst enable	0x0
Bit 7	Reserved	0x1
Bit 6	NOREN: NOR flash access enable	0x0
Bit 5: 4	EXTMDBW: External memory data bus width	Configure according to memory specifications
Bit 3: 2	DEV: Memory device type	Configure according to memory specifications. It is valid except 0x2 (NOR Flash)
Bit 1	ADMUXEN: Address/data multiplexing enable	0x0
Bit 0	EN: Memory bank enable	0x1

Table 24-13 Mode 1— SRAM/NOR Flash chip select timing register (XMC_ BK1TMG) configuration

Bit	Description	Configuration
Bit 31: 30	Reserved	0x0
Bit 29: 28	ASYNCM: Asynchronous mode	0x0
Bit 27: 24	DTLAT: Data latency	0x0
Bit 23: 20	CLKPSC: Clock prescale	0x0
Bit 19: 16	BUSLAT: Bus latency	Indicates the time the XMC_NE[x] from the rising edge to the falling edge. Configure according to needs and memory specifications
Bit 15: 8	DTST: Data setup time	Refer to Figure 24-3 and Figure 24-4. Configure according to needs and memory specifications.
Bit 7: 4	ADDRHT: Address-hold time	0x0
Bit 3: 0	ADDRST: Address setup time	Refer to Figure 24-3 and Figure 24-4. Configure according to needs and memory specifications.

Figure 24-3 NOR/PSRAM mode 1 read access

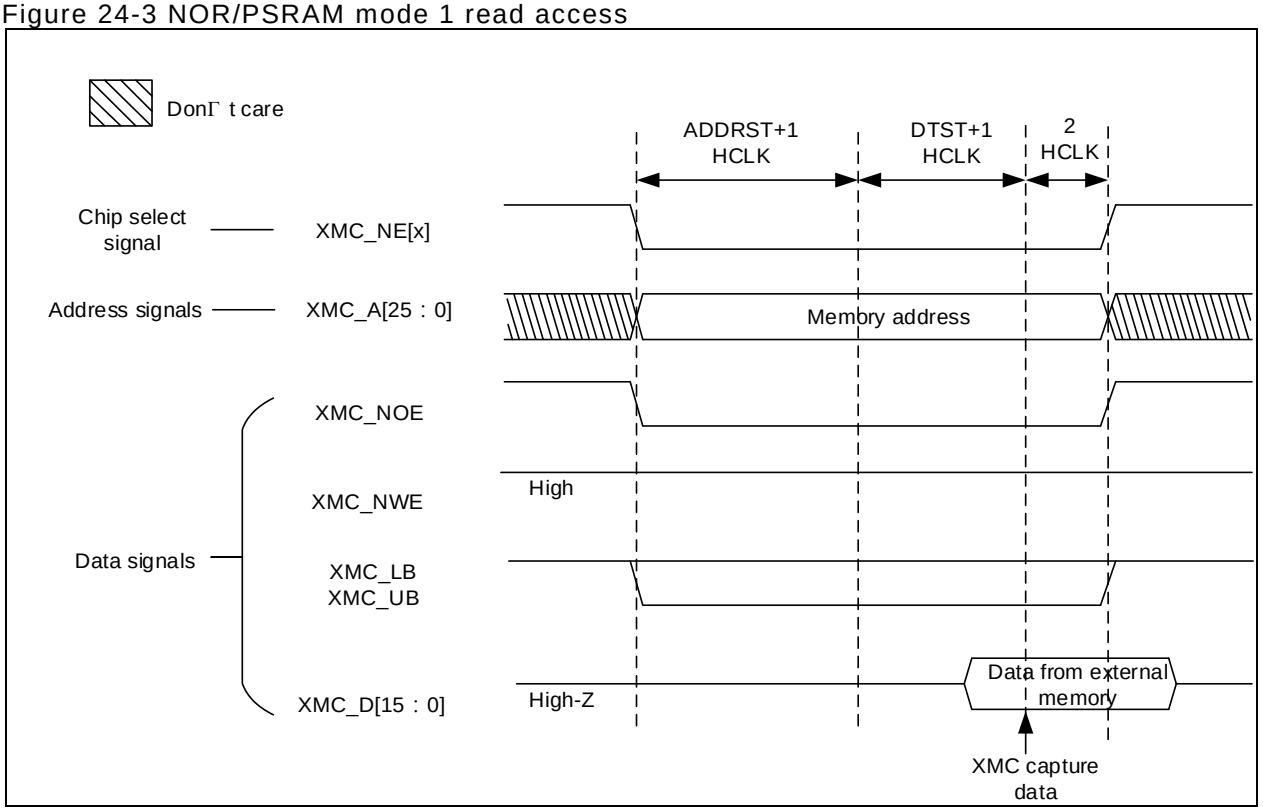
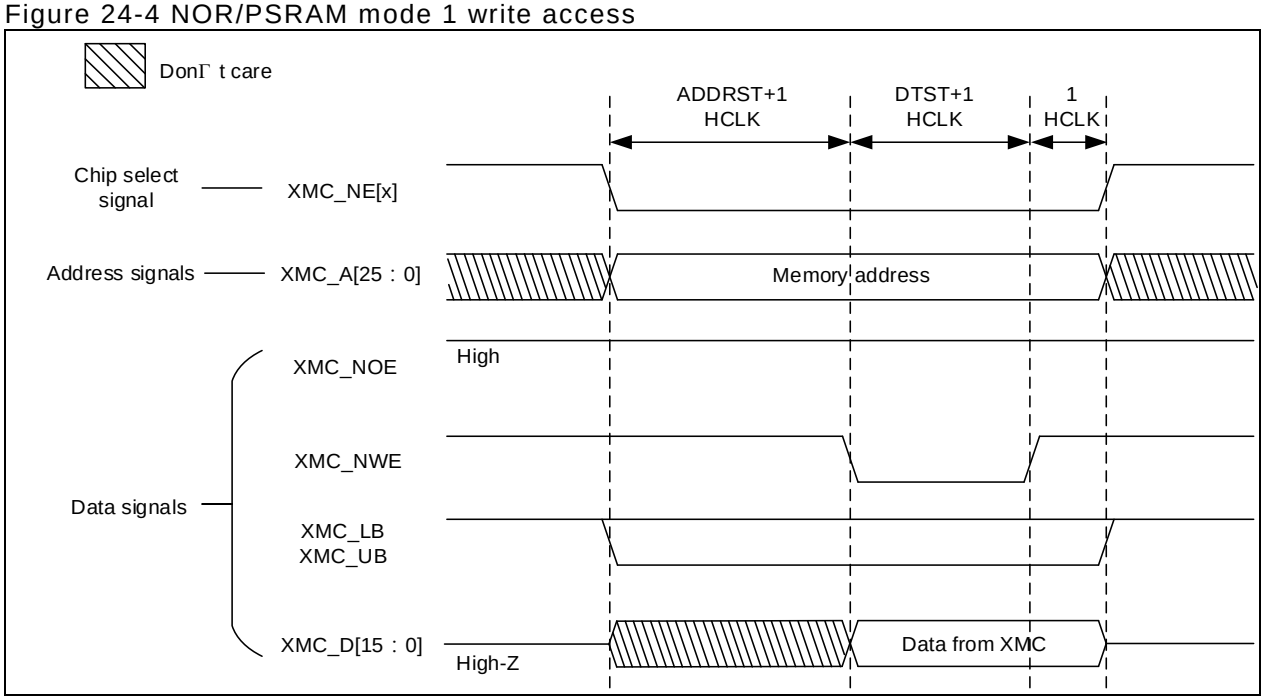


Figure 24-4 NOR/PSRAM mode 1 write access



Mode 2

As configured in Table 24-14 and Table 24-15, the XMC uses mode 2 to access the external memory. The timing of read operation is shown in Figure 24-5. The timing of write operation is shown in Figure 24-6.

Table 24-14 Mode 2 — SRAM/NOR Flash chip select control register

Bit	Description	Configuration
Bit 31: 20	Reserved	0x0
Bit 19	MWMC: Memory write mode control	0x0
Bit 18: 16	CRPGS: CRAM page size	0x0
Bit 15	NWASEN: NWAIT in asynchronous transfer enable	Configure according to memory specifications.

Bit 14	RWTD: Read-write timing different	0x0
Bit 13	NWSEN: NWAIT in synchronous transfer enable	0x0
Bit 12	WEN: Write enable	Configure according to needs.
Bit 11	NWTCFG: NWAIT timing configuration	0x0
Bit 10	WRAPEN: Wrapped enable	0x0
Bit 9	NWPOL: NWAIT polarity	Configure according to memory specifications.
Bit 8	SYNCBEN: Synchronous burst enable	0x0
Bit 7	Reserved	0x1
Bit 6	NOREN: NOR Flash access enable	0x1
Bit 5: 4	EXTMDBW: External memory data bus width	Configure according to memory specifications.
Bit 3: 2	DEV: Memory device type	0x2 (NOR Flash)
Bit 1	ADMUXEN: Address/data multiplexing enable	0x0
Bit 0	EN: Memory bank enable	0x1

Table 24-15 Mode 2 — SRAM/NOR Flash chip select timing register

Bit	Description	Configuration
Bit 31: 30	Reserved	0x0
Bit 29: 28	ASYNCM: Asynchronous mode	0x0
Bit 27: 24	DTLAT: Data latency	0x0
Bit 23: 20	CLKPSC: Clock prescale	0x0
Bit 19: 16	BUSLAT: Bus latency	Indicates the time the XMC_NE[x] from the rising edge to the falling edge. Configure according to needs and memory specifications
Bit 15: 8	DTST: Data setup time	Refer to Figure 24-5 and Figure 24-6. Configure according to needs and memory specifications.
Bit 7: 4	ADDRHT: Address-hold time	0x0
Bit 3: 0	ADDRST: Address setup time	Refer to Figure 24-5 and Figure 24-6. Configure according to needs and memory specifications.

Figure 24-5 NOR/PSRAM mode 2 read access

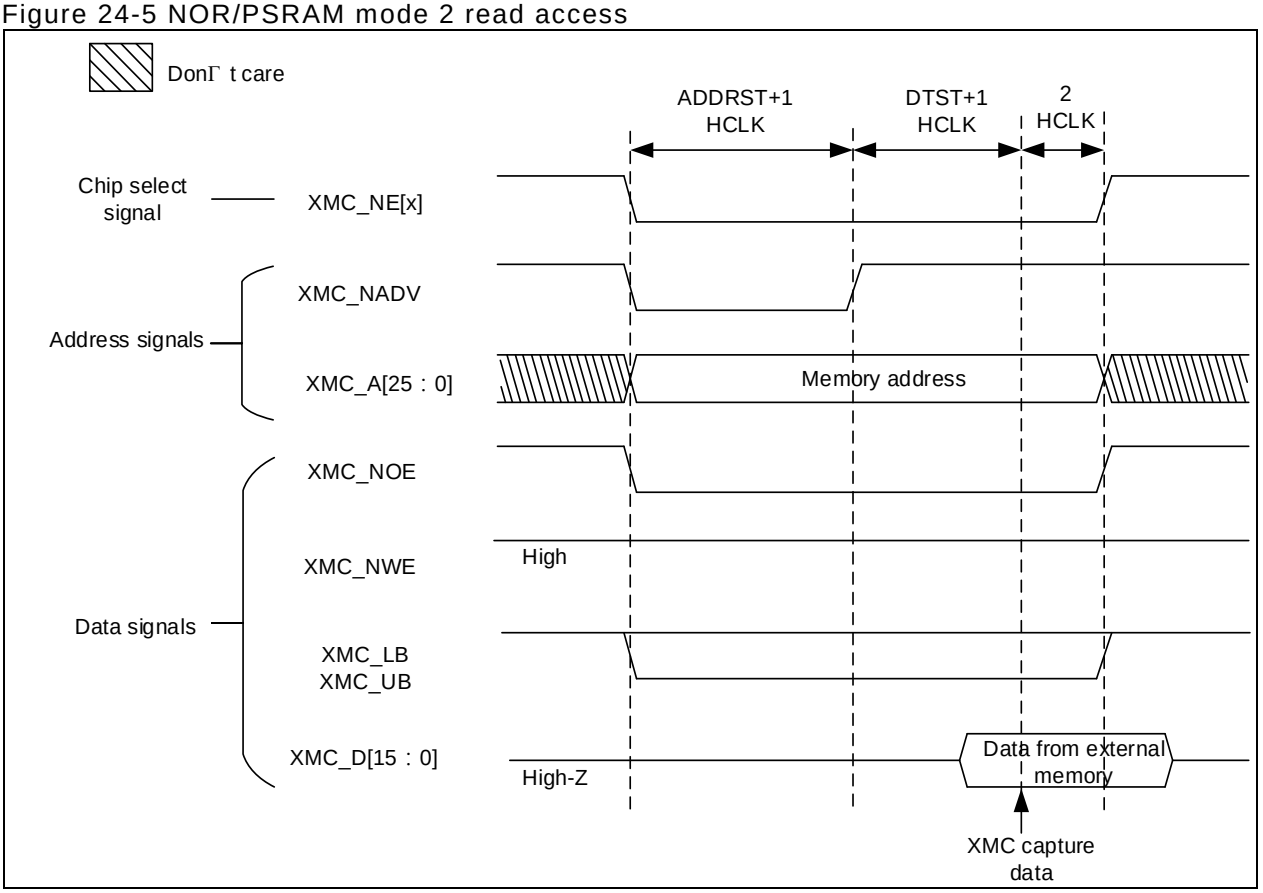
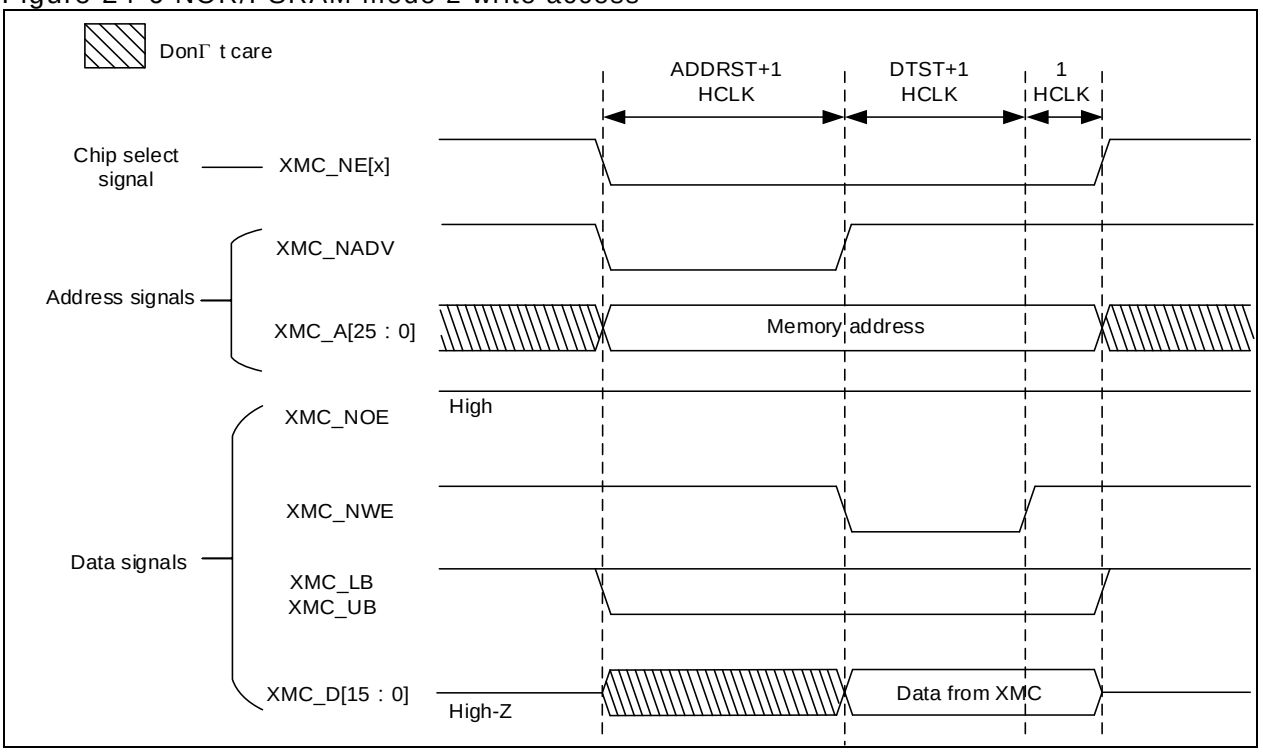


Figure 24-6 NOR/PSRAM mode 2 write access



24.4.2.2 Read/write operation with different timings

The timing of read operation in mode A/B/C/D is based on the SRAM/NOR Flash chip select timing register (XMC_BK1TMG). The timing of write operation is based on SRAM/NOR Flash write timing register (XMC_BK1TMGWR). In addition to this, it is possible to mix A, B, C and D modes for read and write operations.

Mode A

As configured in Table 24-16 and Table 24-17, the XMC uses mode A to access the external memory. The timing of read operation is shown in Figure 24-7. The timing of write operation is shown in Figure 24-8.

Table 24-16 Mode A— SRAM/NOR Flash chip select control register

Bit	Description	Configuration
Bit 31: 20	Reserved	0x0
Bit 19	MWMC: Memory write mode control	0x0
Bit 18: 16	CRPGS: CRAM page size	0x0
Bit 15	NWASEN: NWAIT in asynchronous transfer enable	Configure according to memory specifications.
Bit 14	RWTD: Read-write timing different	0x1
Bit 13	NWSEN: NWAIT in synchronous transfer enable	0x0
Bit 12	WEN: Write enable	Configure according to needs.
Bit 11	NWTCFG: NWAIT timing configuration	0x0
Bit 10	WRAPEN: Wrapped enable	0x0
Bit 9	NWPOL: NWAIT polarity	Configure according to memory specifications.
Bit 8	SYNCBEN: Synchronous burst enable	0x0
Bit 7	Reserved	0x1
Bit 6	NOREN: NOR Flash access enable	0x0
Bit 5: 4	EXTMDBW: External memory data bus width	Configure according to memory specifications.
Bit 3: 2	DEV: Memory device type	Configure according to memory specifications. It is valid except 0x2 (NOR Flash)
Bit 1	ADMUXEN: Address/data multiplexing enable	0x0
Bit 0	EN: Memory bank enable	0x1

Table 24-17 Mode A— SRAM/NOR Flash chip select timing register

Bit	Description	Configuration
Bit 31: 30	Reserved	0x0
Bit 29: 28	ASYNCM: Asynchronous mode	0x0 (Mode A)
Bit 27: 24	DTLAT: Data latency	0x0
Bit 23: 20	CLKPSC: Clock prescale	0x0
Bit 19: 16	BUSLAT: Bus latency	Indicates the time the XMC_NE[x] from the rising edge to the falling edge. Configure according to needs and memory specifications
Bit 15: 8	DTST: Data setup time	Refer to Figure 24-7.Configure according to needs and memory specifications.
Bit 7: 4	ADDRHT: Address-hold time	0x0
Bit 3: 0	ADDRST: Address setup time	Refer to Figure 24-7.Configure according to needs and memory specifications.

Table 24-18 Mode A— SRAM/NOR Flash write timing register

Bit	Description	Configuration
Bit 31: 30	Reserved	0x0
Bit 29: 28	ASYNCM: Asynchronous mode	0x0 (Mode A)
Bit 27: 20	Reserved	0x0
Bit 19: 16	BUSLAT: Bus latency	Indicates the time the XMC_NE[x] from the rising edge to the falling edge. Configure according to needs and memory specifications
Bit 15: 8	DTST: Data setup time	Refer to Figure 24-8. Configure according to needs and memory specifications.
Bit 7: 4	ADDRHT: Address-hold time	0x0
Bit 3: 0	ADDRST: Address setup time	Refer to Figure 24-8. Configure according to needs and memory specifications.

Figure 24-7 NOR/PSRAM mode A read access

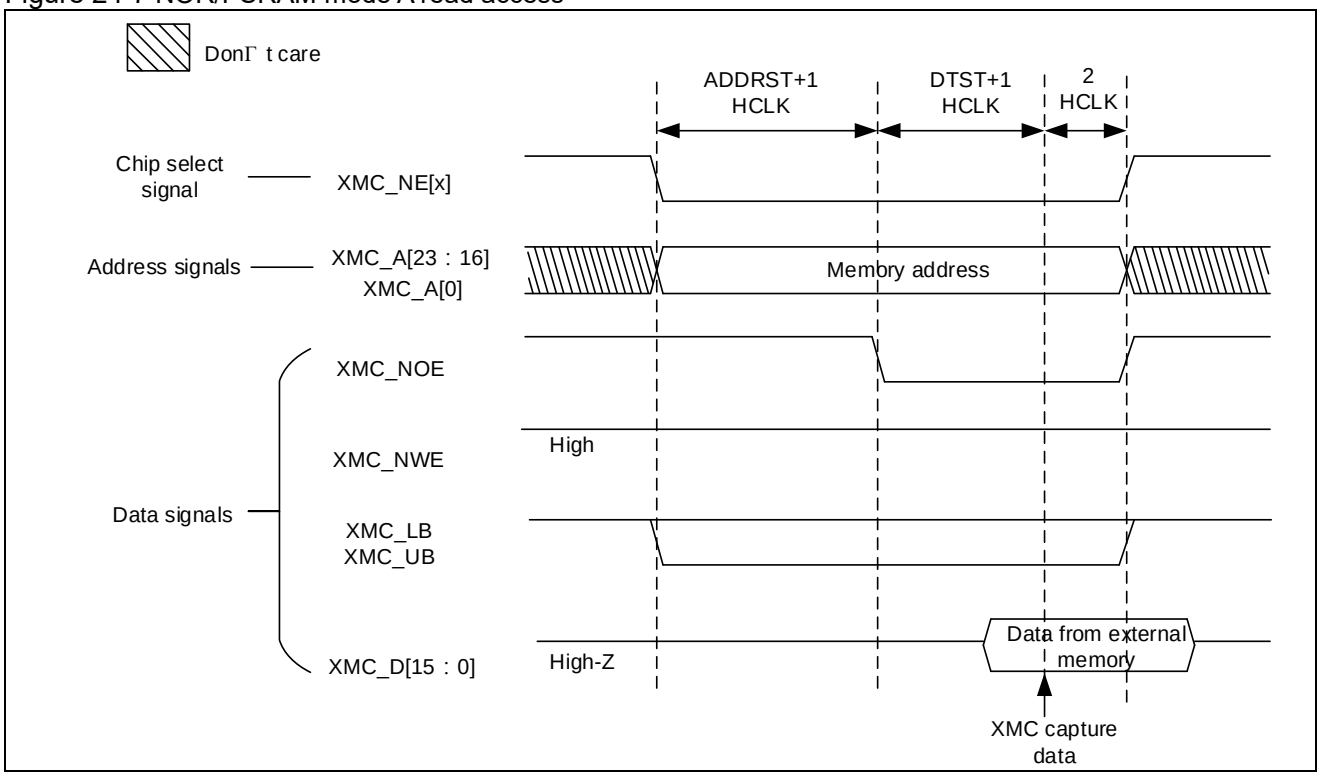
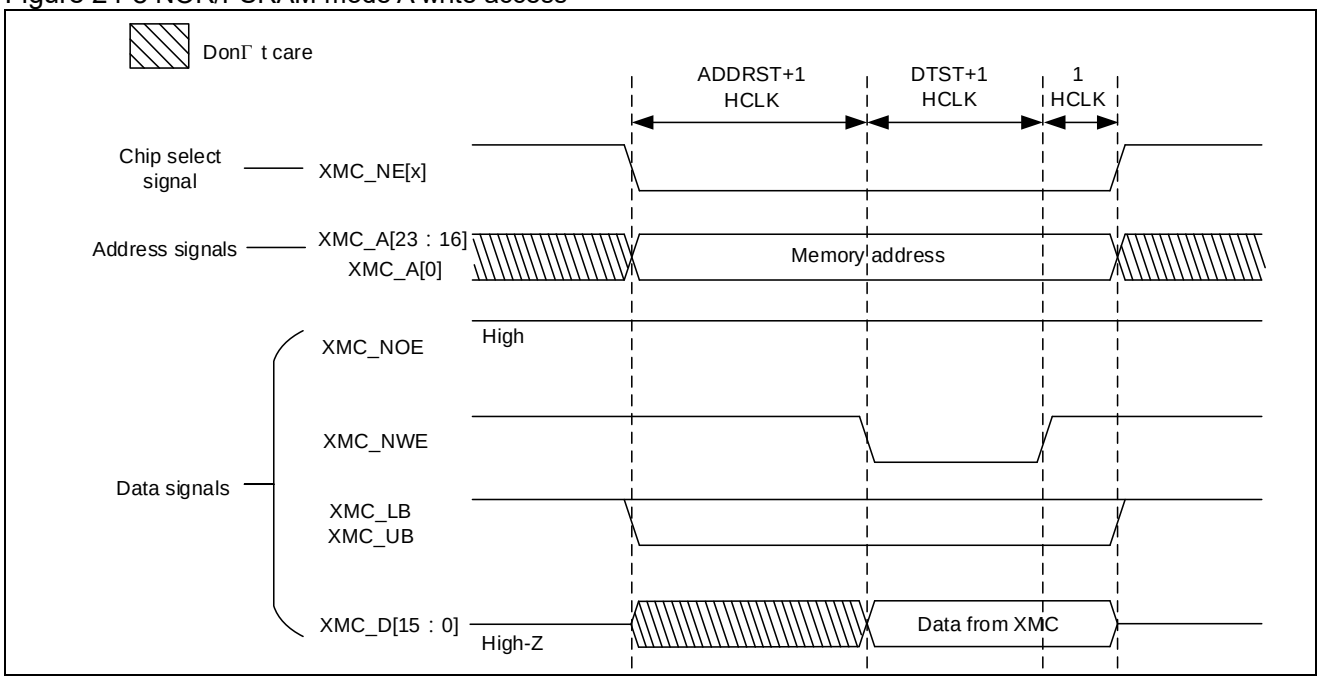


Figure 24-8 NOR/PSRAM mode A write access



Mode B

As configured in Table 24-19, Table 24-20, and Table 24-21, the XMC uses mode B to access the external memory. The timing of read operation is shown in Figure 24-9. The timing of write operation is shown in Figure 24-10.

Table 24-19 Mode B— SRAM/NOR Flash chip select register

Bit	Description	Configuration
Bit 31: 20	Reserved	0x0
Bit 19	MWMC: Memory write mode control	0x0
Bit 18: 16	CRPGS：CRAM page size	0x0
Bit 15	NWASEN: NWAIT in asynchronous transfer enable	Configure according to memory specifications.
Bit 14	RWTD: Read-write timing different	0x1
Bit 13	NWSEN: NWAIT in synchronous transfer enable	0x0
Bit 12	WEN: Write enable	Configure according to needs.
Bit 11	NWTCFG: NWAIT timing configuration	0x0
Bit 10	WRAPEN: Wrapped enable	0x0
Bit 9	NWPOL: NWAIT polarity	Configure according to memory specifications.
Bit 8	SYNCBEN: Synchronous burst enable	0x0
Bit 7	Reserved	0x1
Bit 6	NOREN: NOR Flash access enable	0x1
Bit 5: 4	EXTMDBW: External memory data bus width	Configure according to memory specifications.
Bit 3: 2	DEV: Memory device type	0x2 (NOR Flash)
Bit 1	ADMUXEN: Address/data multiplexing enable	0x0
Bit 0	EN: Memory bank enable	0x1

Table 24-20 Mode B— SRAM/NOR Flash chip select timing register

Bit	Description	Configuration
Bit 31: 30	Reserved	0x0
Bit 29: 28	ASYNCM: Asynchronous mode	0x1 (Mode B)
Bit 27: 24	DTLAT: Data latency	0x0
Bit 23: 20	CLKPSC: Clock prescale	0x0
Bit 19: 16	BUSLAT: Bus latency	Indicates the time the XMC_NE[x] from the rising edge to the falling edge. Configure according to needs and memory specifications
Bit 15: 8	DTST: Data setup time	Refer to Figure 24-9. Configure according to needs and memory specifications.
Bit 7: 4	ADDRHT: Address-hold time	0x0
Bit 3: 0	ADDRST: Address setup time	Refer to Figure 24-9. Configure according to needs and memory specifications.

Table 24-21 Mode B— SRAM/NOR Flash write timing register

Bit	Description	Configuration
Bit 31: 30	Reserved	0x0
Bit 29: 28	ASYNCM: Asynchronous mode	0x1 (Mode B)
Bit 27: 20	Reserved	0x0
Bit 19: 16	BUSLAT: Bus latency	Indicates the time the XMC_NE[x] from the rising edge to the falling edge. Configure according to needs and memory specifications
Bit 15: 8	DTST: Data setup time	Refer to Figure 24-10. Configure according to needs and memory specifications.
Bit 7: 4	ADDRHT: Address-hold time	0x0
Bit 3: 0	ADDRST: Address setup time	Refer to Figure 24-10. Configure according to needs and memory specifications.

Figure 24-9 NOR/PSRAM mode B read access

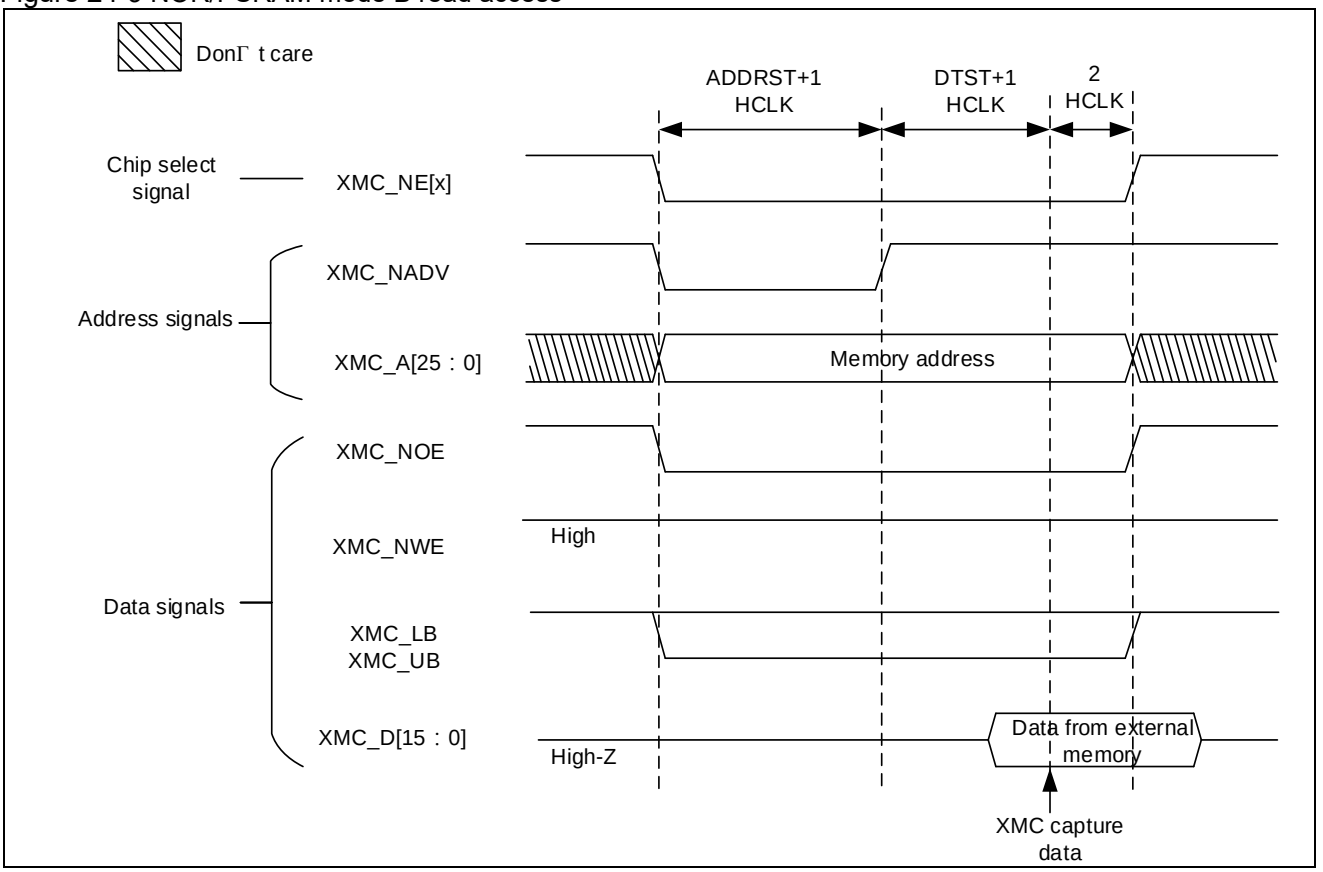
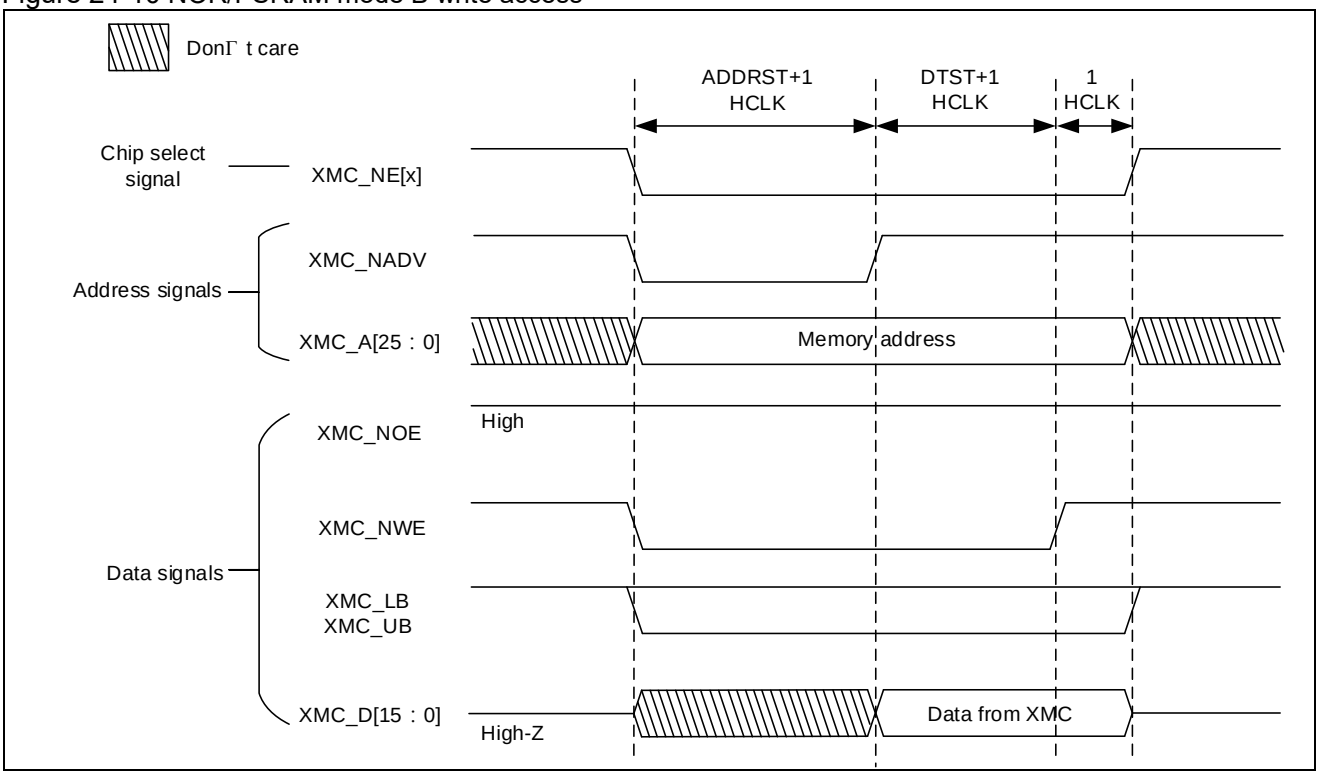


Figure 24-10 NOR/PSRAM mode B write access



Mode C

As configured in Table 24-22, Table 24-23 and Table 24-24, the XMC uses mode C to access the external memory. The timing of read operation is shown in Figure 24-11. The timing of write operation is shown in Figure 24-12.

Table 24-22 Mode C— SRAM/NOR Flash chip select register

Bit	Description	Configuration
Bit 31: 20	Reserved	0x0
Bit 19	MWMC: Memory write mode control	0x0
Bit 18: 16	CRPGS：CRAM page size	0x0
Bit 15	NWASEN: NWAIT in asynchronous transfer enable	Configure according to memory specifications.
Bit 14	RWTD: Read-write timing different	0x1
Bit 13	NWSEN: NWAIT in synchronous transfer enable	0x0
Bit 12	WEN: Write enable	Configure according to needs.
Bit 11	NWTCFG: NWAIT timing configuration	0x0
Bit 10	WRAPEN: Wrapped enable	0x0
Bit 9	NWPOL: NWAIT polarity	Configure according to memory specifications.
Bit 8	SYNCBEN: Synchronous burst enable	0x0
Bit 7	Reserved	0x1
Bit 6	NOREN: NOR Flash access enable	0x1
Bit 5: 4	EXTMDBW: External memory data bus width	Configure according to memory specifications.
Bit 3: 2	DEV: Memory device type	0x2 (NOR Flash)
Bit 1	ADMUXEN: Address/data multiplexing enable	0x0
Bit 0	EN: Memory bank enable	0x1

Table 24-23 Mode C—SRAM/NOR Flash chip select timing register

Bit	Description	Configuration
Bit 31: 30	Reserved	0x0
Bit 29: 28	ASYNCM: Asynchronous mode	0x2 (Mode C)
Bit 27: 24	DTLAT: Data latency	0x0
Bit 23: 20	CLKPSC: Clock prescale	0x0
Bit 19: 16	BUSLAT: Bus latency	Indicates the time the XMC_NE[x] from the rising edge to the falling edge. Configure according to needs and memory specifications
Bit 15: 8	DTST: Data setup time	Refer to Figure 24-11. Configure according to needs and memory specifications.
Bit 7: 4	ADDRHT: Address-hold time	0x0
Bit 3: 0	ADDRST: Address setup time	Refer to Figure 24-11. Configure according to needs and memory specifications.

Table 24-24 Mode C— SRAM/NOR Flash write timing register

Bit	Description	Configuration
Bit 31: 30	Reserved	0x0
Bit 29: 28	ASYNCM: Asynchronous mode	0x1 (Mode C)
Bit 27: 20	Reserved	0x0
Bit 19: 16	BUSLAT: Bus latency	Indicates the time the XMC_NE[x] from the rising edge to the falling edge. Configure according to needs and memory specifications
Bit 15: 8	DTST: Data setup time	Refer to Figure 24-12. Configure according to needs and memory specifications.
Bit 7: 4	ADDRHT: Address-hold time	0x0
Bit 3: 0	ADDRST: Address setup time	Refer to Figure 24-12. Configure according to needs and memory specifications.

Figure 24-11 NOR/PSRAM mode C read access

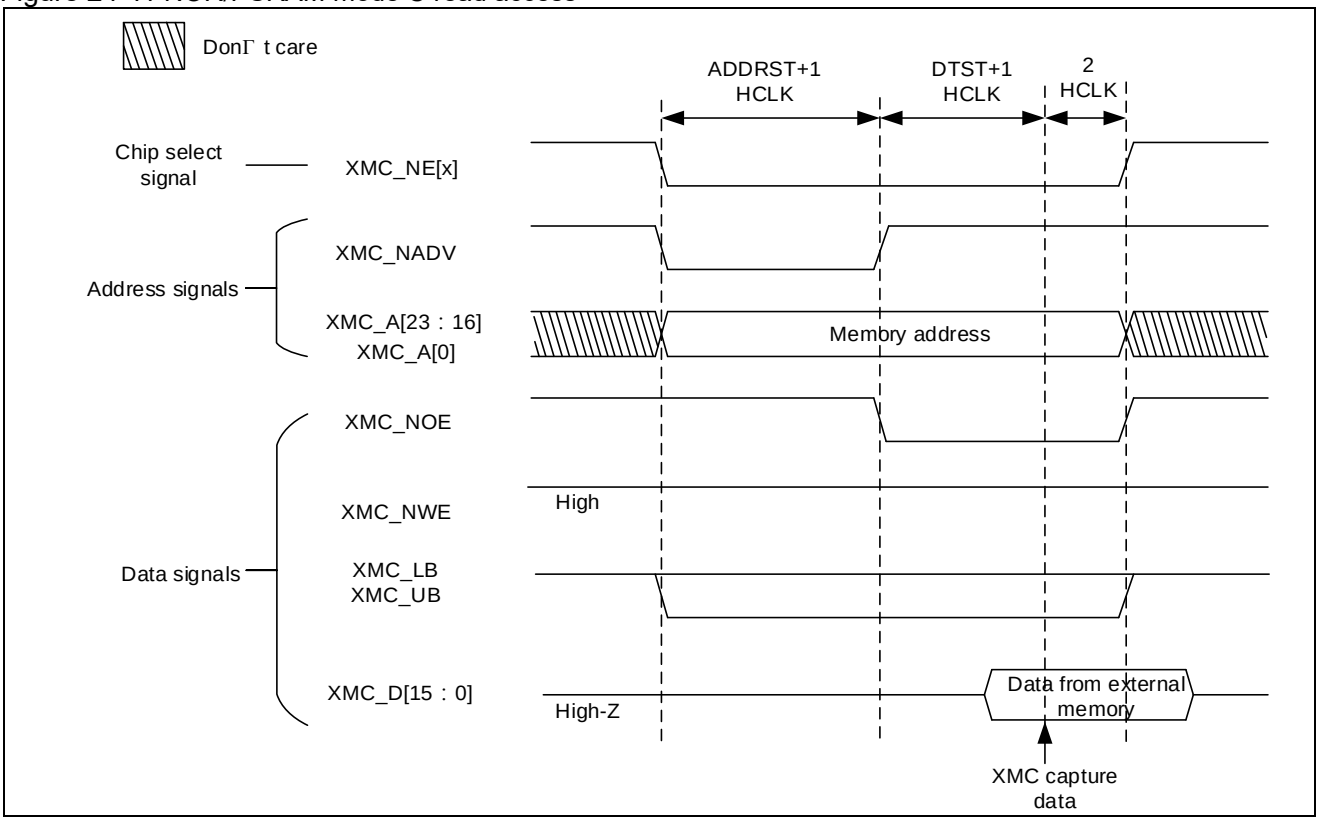
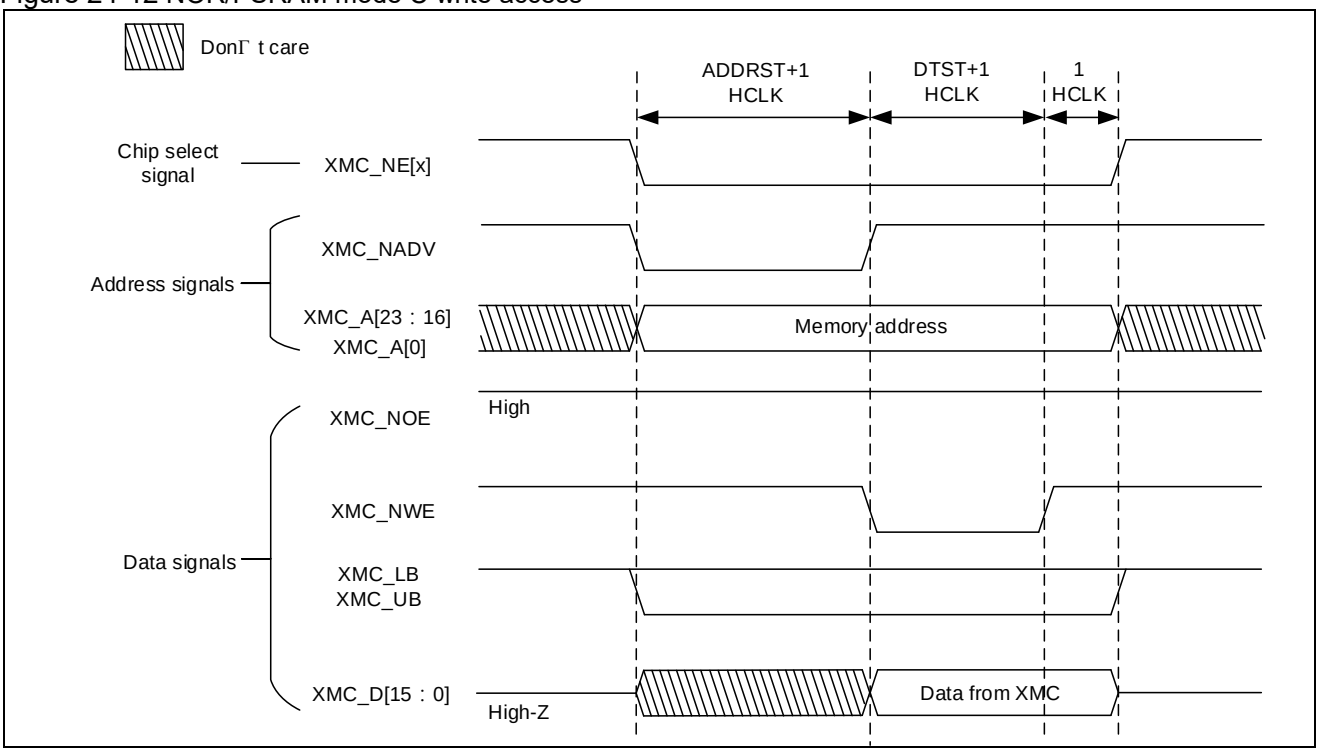


Figure 24-12 NOR/PSRAM mode C write access



Mode D

As configured in Table 24-25, Table 24-26, and Table 24-27, the XMC uses mode D to access the external memory. The timing of read operation is shown in Figure 24-13. The timing of write operation is shown in Figure 24-14.

Table 24-25 Mode D— SRAM/NOR Flash chip select register (XMC_BK1CTRL) configuration

Bit	Description	Configuration
Bit 31: 20	Reserved	0x0
Bit 19	MWMC: Memory write mode control	0x0
Bit 18: 16	CRPGS: CRAM page size	0x0
Bit 15	NWASEN: NWAIT in asynchronous transfer enable	Configure according to memory specifications.
Bit 14	RWTD: Read-write timing different	0x1
Bit 13	NWSEN: NWAIT in synchronous transfer enable	0x0
Bit 12	WEN: Write enable	Configure according to needs.
Bit 11	NWTCFG: NWAIT timing configuration	0x0
Bit 10	WRAPEN: Wrapped enable	0x0
Bit 9	NWPOL: NWAIT polarity	Configure according to memory specifications.
Bit 8	SYNCBEN: Synchronous burst enable	0x0
Bit 7	Reserved	0x1
Bit 6	NOREN: NOR Flash access enable	Configure according to memory specifications.
Bit 5: 4	EXTMDBW: External memory data bus width	Configure according to memory specifications.
Bit 3: 2	DEV: Memory device type	Configure according to memory specifications.
Bit 1	ADMUXEN: Address/data multiplexing enable	0x0
Bit 0	EN: Memory bank enable	0x1

Table 24-26 Mode D—SRAM/NOR Flash chip select timing register

Bit	Description	Configuration
Bit 31: 30	Reserved	0x0
Bit 29: 28	ASYNCM: Asynchronous mode	0x3 (Mode D)
Bit 27: 24	DTLAT: Data latency	0x0
Bit 23: 20	CLKPSC: Clock prescale	0x0
Bit 19: 16	BUSLAT: Bus latency	Indicates the time the XMC_NE[x] from the rising edge to the falling edge. Configure according to needs and memory specifications
Bit 15: 8	DTST: Data setup time	Refer to Figure 24-13. Configure according to needs and memory specifications.
Bit 7: 4	ADDRHT: Address-hold time	Refer to Figure 24-13. Configure according to needs and memory specifications.
Bit 3: 0	ADDRST: Address setup time	Refer to Figure 24-13. Configure according to needs and memory specifications.

Table 24-27 Mode D— SRAM/NOR Flash write timing register

Bit	Description	Configuration
Bit 31: 30	Reserved	0x0
Bit 29: 28	ASYNCM: Asynchronous mode	0x3 (Mode D)
Bit 27: 20	Reserved	0x0
Bit 19: 16	BUSLAT: Bus latency	Indicates the time the XMC_NE[x] from the rising edge to the falling edge. Configure according to needs and memory specifications
Bit 15: 8	DTST: Data setup time	Refer to Figure 24-14. Configure according to needs and memory specifications.
Bit 7: 4	ADDRHT: Address-hold time	Refer to Figure 24-14. Configure according to needs and memory specifications.
Bit 3: 0	ADDRST: Address setup time	Refer to Figure 24-14. Configure according to needs and memory specifications.

Figure 24-13 NOR/PSRAM mode D read access

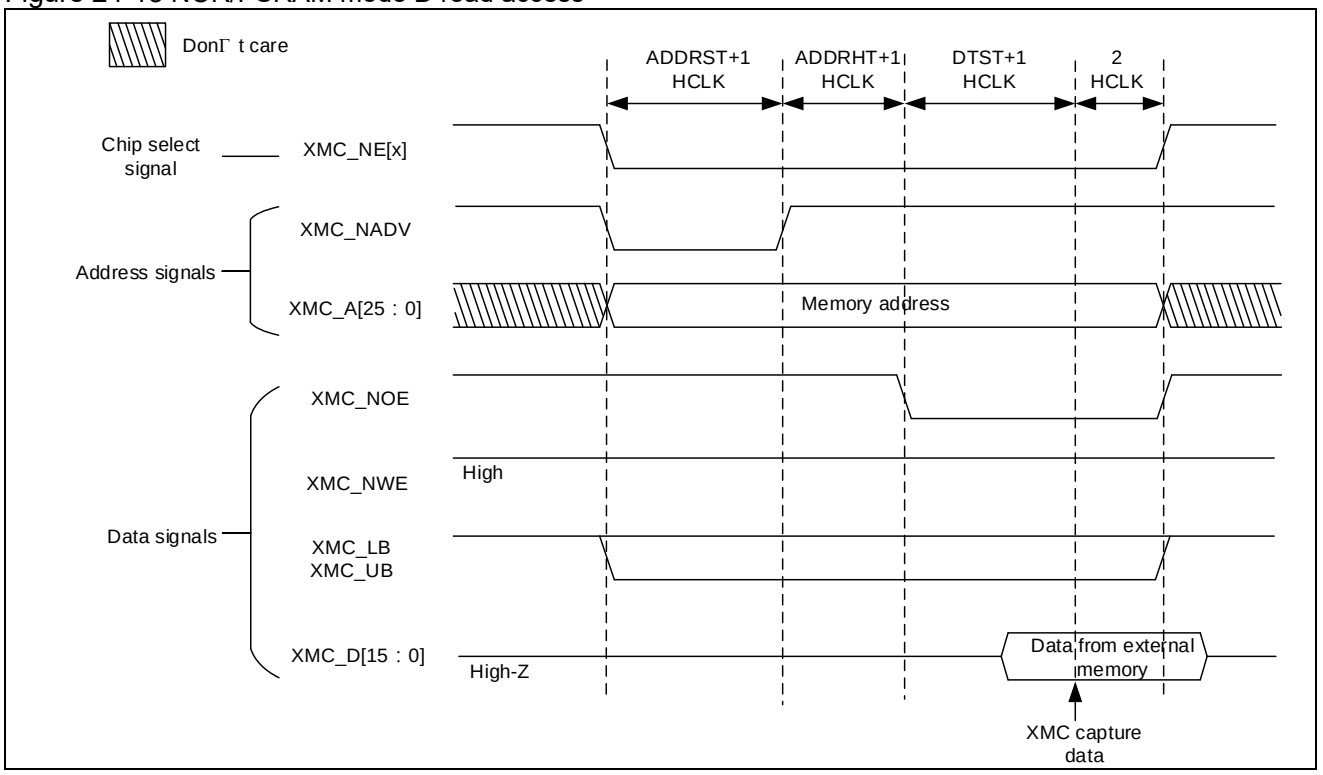
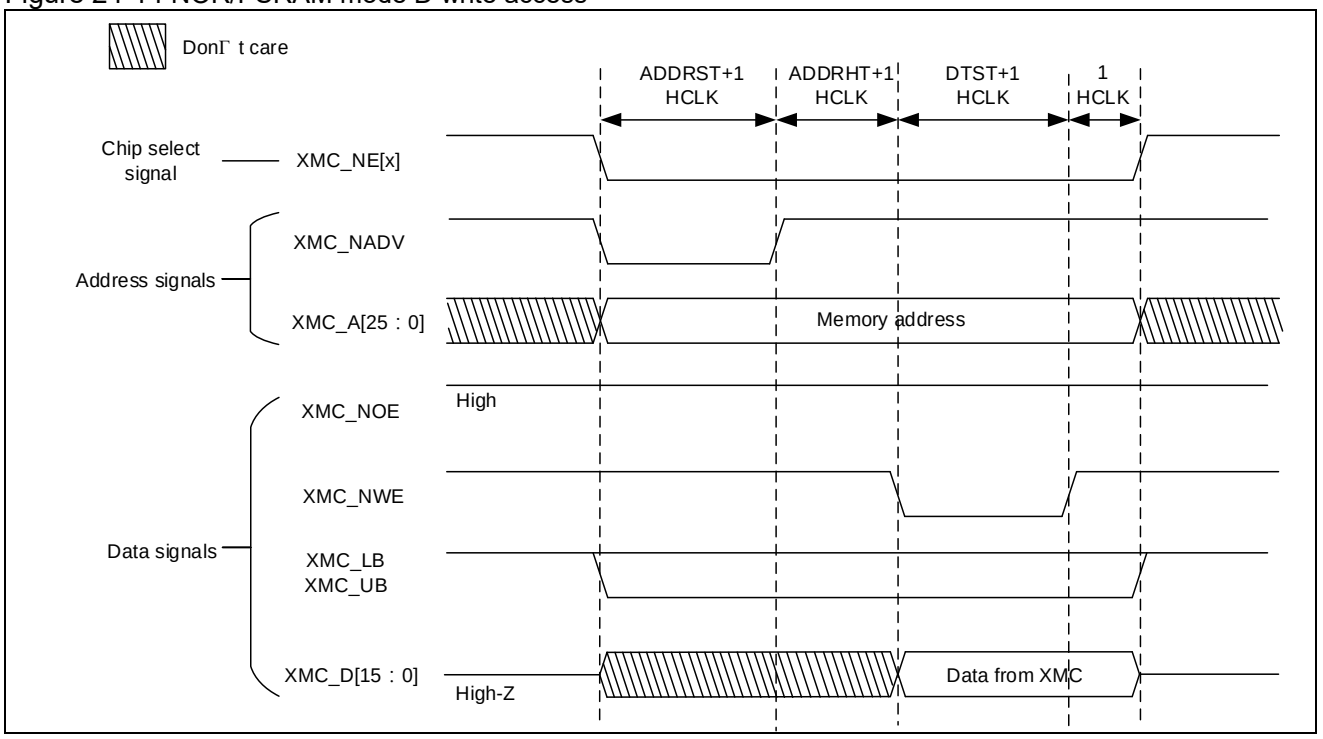


Figure 24-14 NOR/PSRAM mode D write access



24.4.2.3 Multiplexed mode

As configured in Table 24-28 and Table 24-29, the XMC uses mode A to access the external memory. The timing of read operation is shown in Figure 24-15. The timing of write operation is shown in Figure 24-16.

Table 24-28 Multiplexed mode — SRAM/NOR Flash chip select control register

Bit	Description	Configuration
Bit 31: 20	Reserved	0x0
Bit 19	MWMC: Memory write mode control	0x0
Bit 18: 16	CRPGS: CRAM page size	0x0
Bit 15	NWASEN: NWAIT in asynchronous transfer enable	Configure according to memory specifications.
Bit 14	RWTD: Read-write timing different	0x0
Bit 13	NWSEN: NWAIT in synchronous transfer enable	0x0
Bit 12	WEN: Write enable	Configure according to needs.
Bit 11	NWTCFG: NWAIT timing configuration	0x0
Bit 10	WRAPEN: Wrapped enable	0x0
Bit 9	NWPOL: NWAIT polarity	Configure according to memory specifications.
Bit 8	SYNCBEN: Synchronous burst enable	0x0
Bit 7	Reserved	0x1
Bit 6	NOREN: NOR Flash access enable	Configure according to memory specifications.
Bit 5: 4	EXTMDBW: External memory data bus width	Configure according to memory specifications.
Bit 3: 2	DEV: Memory device type	Configure according to memory specifications. It is valid except 0x0 (SRAM)
Bit 1	ADMUXEN: Address/data multiplexing enable	0x1
Bit 0	EN: Memory bank enable	0x1

Table 24-29 Multiplexed mode—SRAM/NOR Flash chip select timing register (XMC_BK1TMG) configuration

Bit	Description	Configuration
Bit 31: 30	Reserved	0x0
Bit 29: 28	ASYNCM: Asynchronous mode	0x0
Bit 27: 24	DTLAT: Data latency	0x0
Bit 23: 20	CLKPSC: Clock prescale	0x0
Bit 19: 16	BUSLAT: Bus latency	Indicates the time the XMC_NE[x] from the rising edge to the falling edge. Configure according to needs and memory specifications
Bit 15: 8	DTST: Data setup time	Refer to Figure 24-15 and Figure 24-16. Configure according to needs and memory specifications.
Bit 7: 4	ADDRHT: Address-hold time	Refer to Figure 24-15 and Figure 24-16. Configure according to needs and memory specifications.
Bit 3: 0	ADDRST: Address setup time	Refer to Figure 24-15 and Figure 24-16. Configure according to needs and memory specifications.

Figure 24-15 NOR/PSRAM multiplexed mode read access

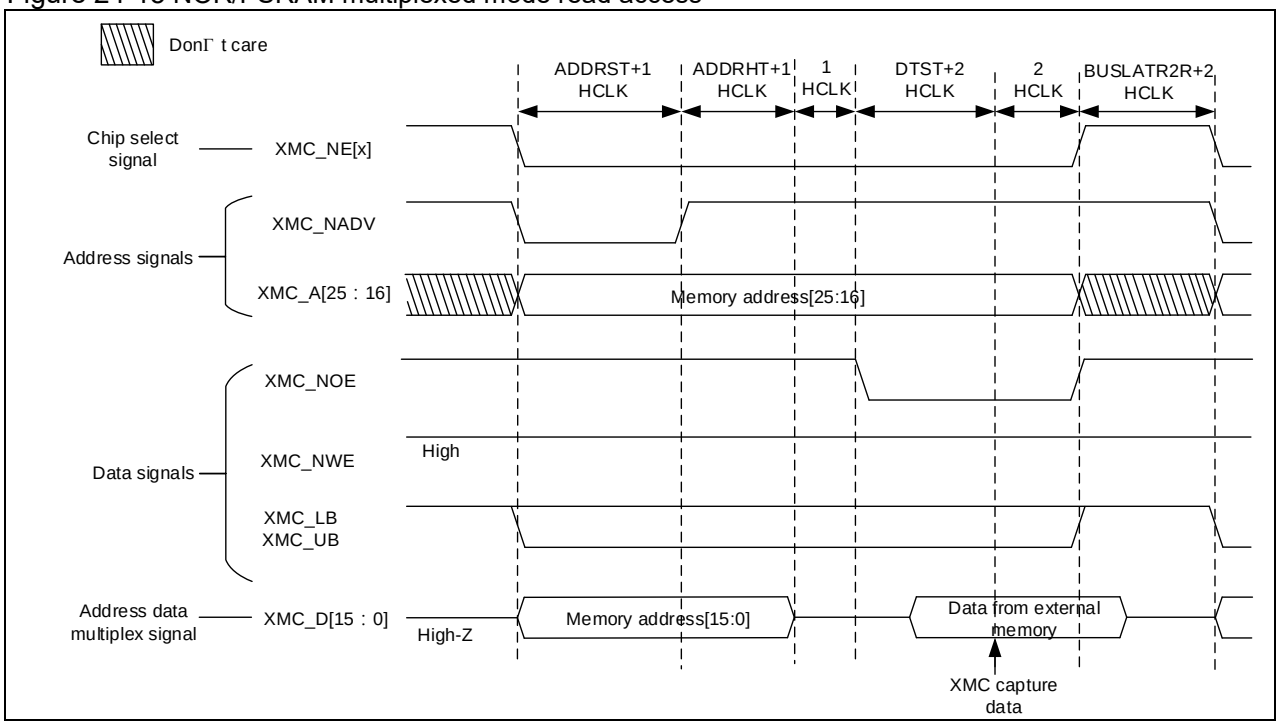
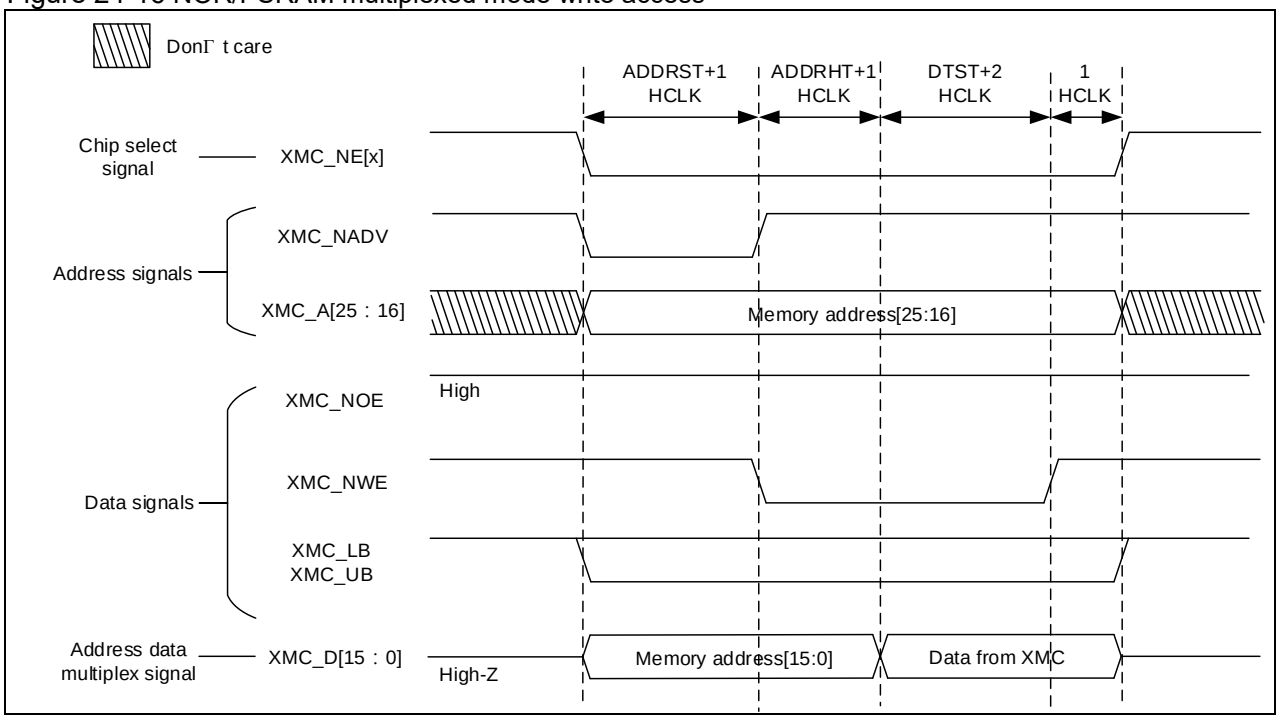


Figure 24-16 NOR/PSRAM multiplexed mode write access



24.4.2.4 Synchronous mode

As configured in Table 24-30 and Table 24-31, the XMC uses synchronous mode to access the external memories.

If the memory inserts XMC_NWAIT signal between the address latch and data transfer, the XMC will not only wait (DTLAT+1) CLK clock cycles but also have to take into account the XMC_NWAIT signal. During data transmission, the XMC will, depending on the NWTCFG configuration, select to wait either one cycle after the XMC_NWAIT signal is active or when the XMC_NWAIT signal is active.

Figure 24-17 shows the timing for read access, while Figure 24-18 shows the timing for write access. Figure 24-17 and Figure 24-18 are examples of XMC waiting in the next cycle of XMC_NWAIT signal (NWTCFG=0)

Table 24-30 Synchronous mode — SRAM/NOR Flash chip select control register

Bit	Description	Configuration
Bit 31: 20	Reserved	0x0
Bit 19	MWMC: Memory write mode control	0x1
Bit 18: 16	CRPGS: CRAM page size	Configure according to memory specifications.
Bit 15	NWASEN: NWAIT in asynchronous transfer enable	0x0
Bit 14	RWTD: Read-write timing different	0x0
Bit 13	NWSEN: NWAIT in synchronous transfer enable	Configure according to memory specifications.
Bit 12	WEN: Write enable	Configure according to needs.
Bit 11	NWTCFG: NWAIT timing configuration	Configure according to memory specifications.
Bit 10	WRAPEN: Wrapped enable	Configure according to needs.
Bit 9	NWPOL: NWAIT polarity	c
Bit 8	SYNCBEN: Synchronous burst enable	0x1
Bit 7	Reserved	0x1
Bit 6	NOREN: NOR Flash access enable	Write synchronization: 0x0 Read synchronization: Configure according to memory specifications.
Bit 5: 4	EXTMDBW: External memory data bus width	Configure according to memory specifications.
Bit 3: 2	DEV: Memory device type	Write synchronization: 0x1 Read synchronization: Configure according to memory specifications. It is valid except 0x0 (SRAM)
Bit 1	ADMUXEN: Address/data multiplexing enable	Configure according to needs.

Bit 0	EN: Memory bank enable	0x1

Table 24-31 Synchronous mode—SRAM/NOR Flash chip select timing register (XMC_BK1TMG)

Bit	Description	Configuration
Bit 31: 30	Reserved	0x0
Bit 29: 28	ASYNCM: Asynchronous mode	0x0
Bit 27: 24	DTLAT: Data latency	Refer to Figure 24-17 and Figure 24-18
Bit 23: 20	CLKPSC: Clock prescale	XMC_CLK cycle is HCLK cycle*(CLKPSC+1). Refer to Figure 24-17 and Figure 24-18
Bit 19: 16	BUSLAT: Bus latency	Indicates the time the XMC_NE[x] from the rising edge to the falling edge. Configure according to needs and memory specifications
Bit 15: 8	DTST: Data setup time	0x0
Bit 7: 4	ADDRHT: Address-hold time	0x0
Bit 3: 0	ADDRST: Address setup time	0x0

Figure 24-17 NOR/PSRAM synchronous multiplexed mode read access

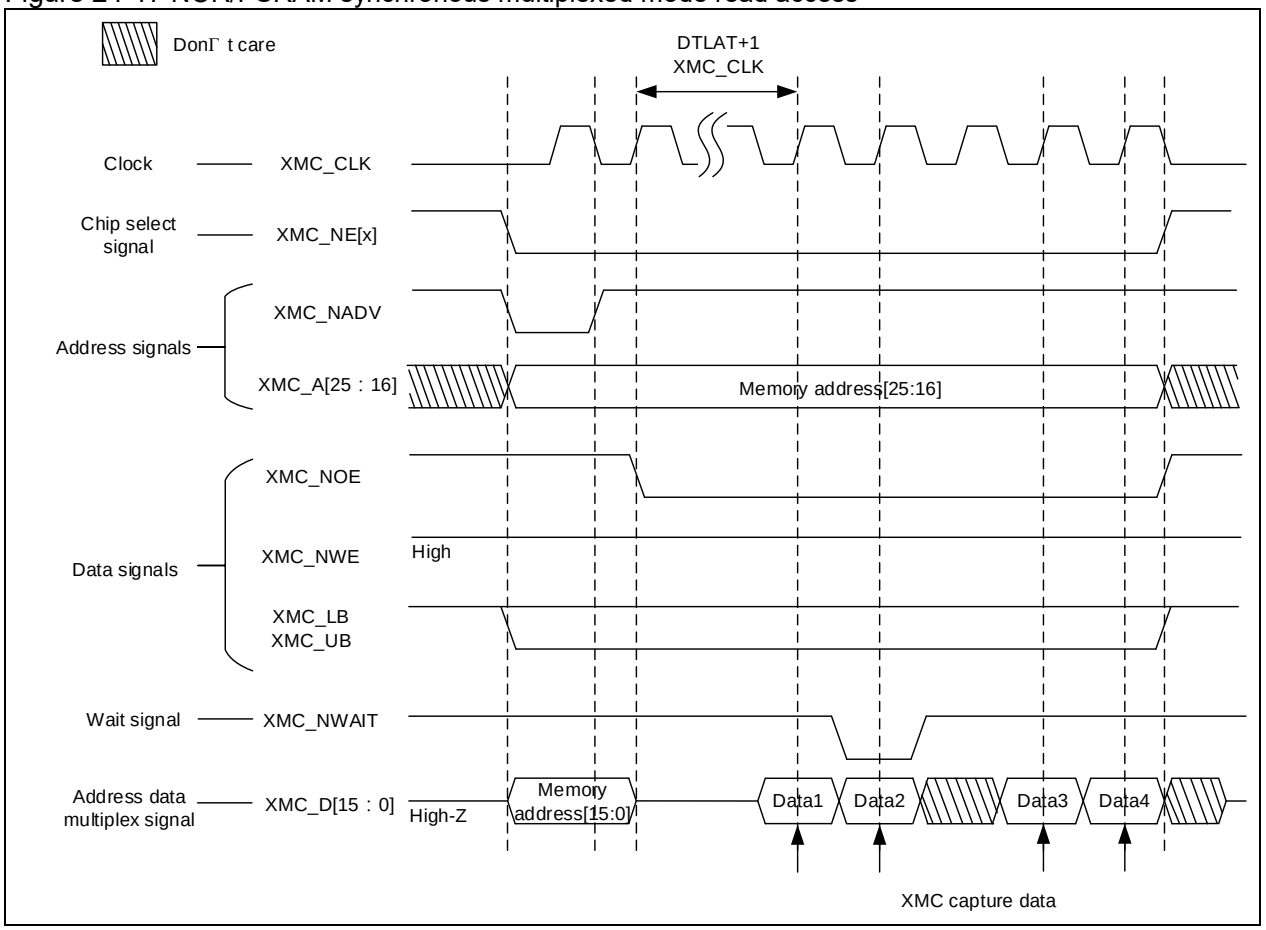
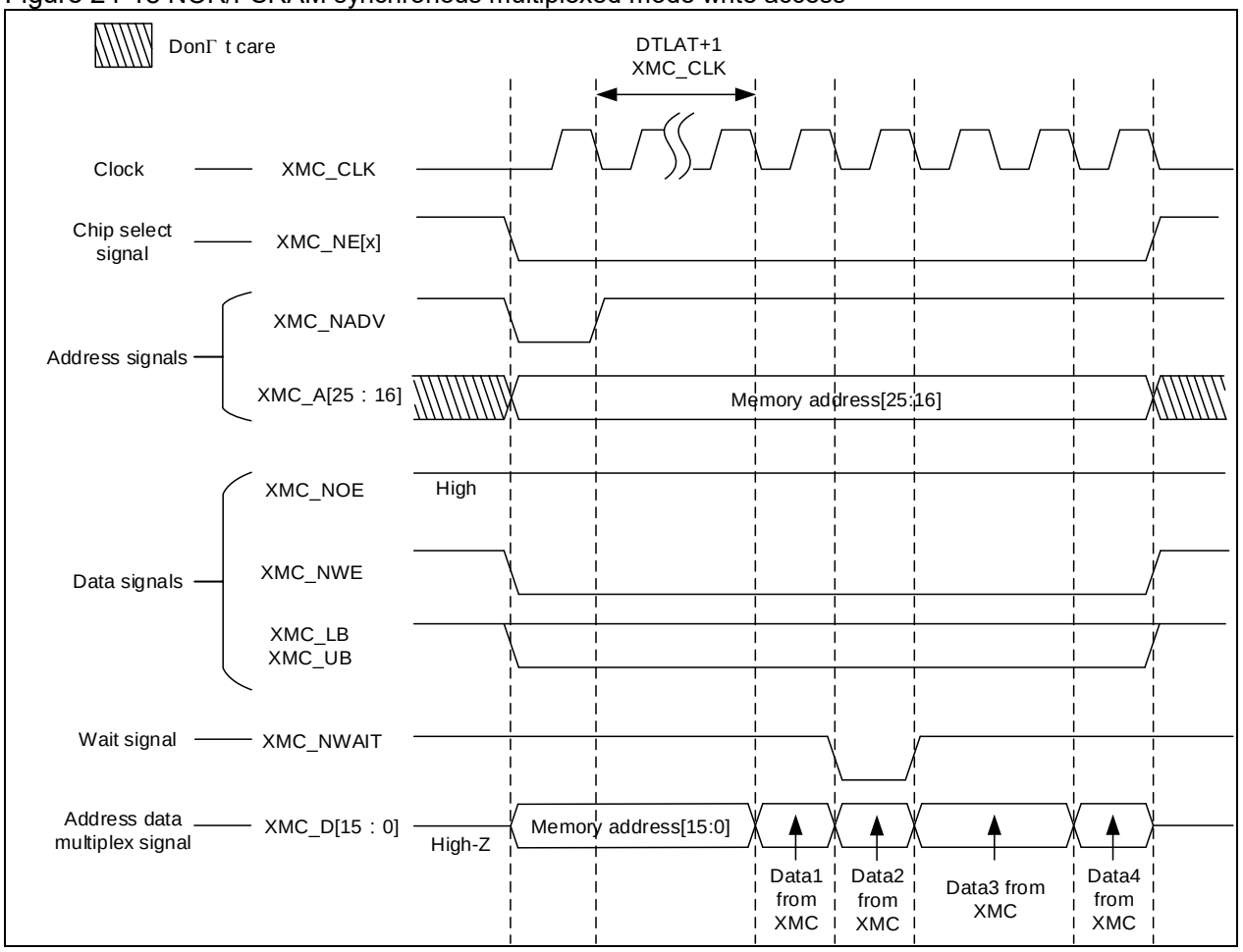


Figure 24-18 NOR/PSRAM synchronous multiplexed mode write access



24.4.3 NAND

NAND interface can be used to drive NAND Flash. It is divided into two storage banks: regular bank and special bank, each with its separate timing registers. Both banks can be accessible with different timings.

24.4.4 Operating mode

Pin function:

Pin signals vary from external memory to external memory. Table 24-32 lists typical pin signals.

Table 24-32 Typical pin signals for NAND Flash

XMC pin name	8-bit NAND Flash	16-bit NAND Flash
XMC_NCE[2]	Chip-select	Chip-select
XMC_A[17]	Address latch enable (ALE)	Address latch enable (ALE)
XMC_A[16]	Command latch enable (CLE)	Command latch enable (CLE)
XMC_NOE	Output enable (NRE)	Output enable (NRE)
XMC_NWE	Write enable	Write enable
XMC_D[15: 0]	Data bus	Do not use XMC_D[15: 8] Use XMC_D[7: 0] as data bus.
XMC_NWAIT	Ready/Busy (R/B)	Ready/Busy (R/B)

Access address

The HADDR is only used to select memory banks. Refer to Table 24-5 for more information.

The user writes the command value in the command section, the destination address in the address section, and reads or writes the data from or to the data section. As the access addresses are transmitted through data bus, the HADDR is actually not associated with NAND Flash size, so theoretically the XMC has no limitation on the NAND Flash capacity accessible.

Data access

In case that the AHB data width is not equal to that of the memories, the XMC will make appropriate arrangement according to the typical signals of the external memories. Table 24-33 lists the operation

modes supported by XMC.

Table 24-33 Data access width vs. external memory data width

Memory	Mode	AHB data width	Memory width	Description
8-bit NAND	R/W	8	8	
	R/W	16	8	Split into 2 XMC accesses
	R/W	32	8	Split into 4 XMC accesses
16-bit NAND	R	8	16	
	R/W	16	16	
	R/W	32	16	Split into 4 XMC accesses

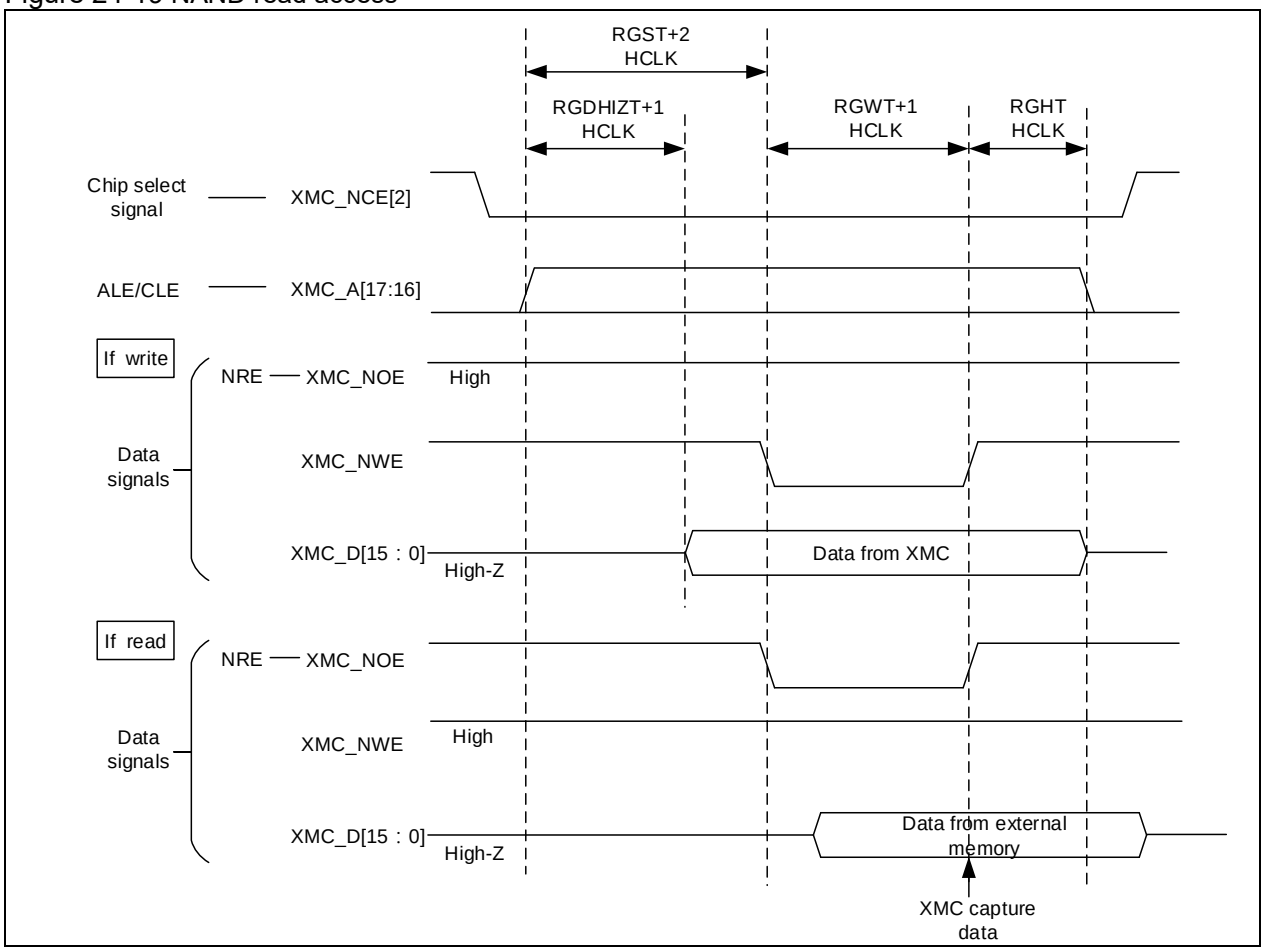
24.4.5 Access timings

The XMC access the NAND Flash according to the timing parameters, as shown in Table 24-34 and Figure 24-19. Users can perform programming operations according to the specifications of the external memory and application needs.

Table 24-34 NAND parameter registers

Parameter register	Function	Access mode	Unit
RGDHIZT/SPDHIZT	Number of cycles during which the data bus is kept in high-Z state	W	HCLK cycle
RGST/SPST	Memory set up time	R/W	HCLK cycle
RGWT/SPWT	Memory set up time	R/W	HCLK cycle
RGHT/SPHT	Memory set up time	R/W	HCLK cycle

Figure 24-19 NAND read access

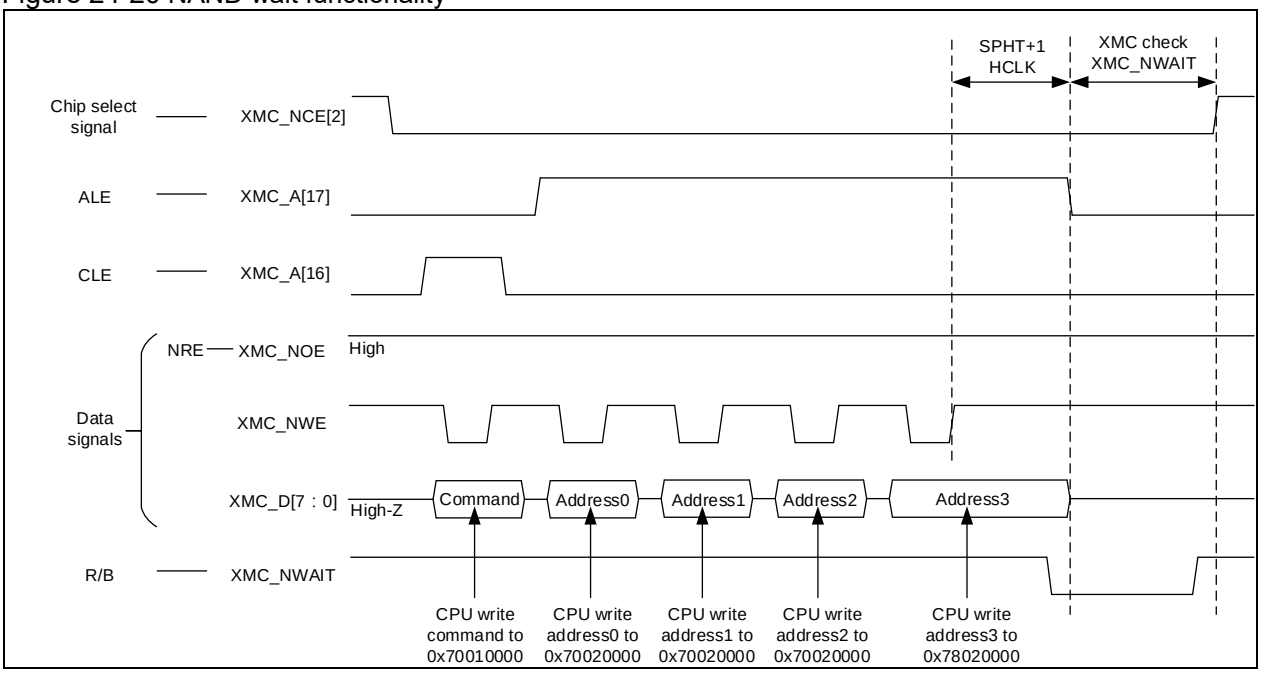


When the NWEN bit is enabled, the XMC will monitor whether the XMC_NWAIT signal is pulled low or not at the end of memory hold-up period, if so, it will keep the XMC_NCE[2] low until the XMC_NWAIT goes high.

Some NAND Flash devices require that, after receiving the last address byte, that the XMC_NCE[2] remains low until it enters ready state. This can be done by means of a special timing register and NWEN bits:

The user has to configure the time duration during which the NAND Flash shifts from the rising edge of the XMC_NWE to the falling edge of the XMC_NWAIT into a SPHT register, and write the last address byte into a special memory address section so that the XMC can perform write operations based on the timings of the special memory timing register, as shown in Address 3 in Figure 24-20.

Figure 24-20 NAND wait functionality



24.4.6 ECC computation

The NAND interface contains ECC computation module so that the data will be used for ECC computation when the NAND interface access the NAND Flash. The computed value is stored into the XMC_BK2ECC register.

To perform an ECC computation:

1. Configure the ECCPGS bit to select the number of bytes to be computed by ECC module: 256, 512, 1024, 2048, 4096 or 8192 bytes.
2. Enable the ECCEN bit.
3. Read/write from and to the data section.
4. After receiving/sending the same number of bytes as the value programed in the ECCPGS, the XMC will store the ECC computed value into the XMC_BK2ECC registers.
5. Software reads/write the last byte and waits until the FIFO flag is set.
6. Software reads the XMC_BK2ECC register and performs the corresponding error correction routine.
7. Clear the ECCEN bit by software. Repeat from 2 to 6.

Table 24-35 lists the ECC result bits corresponding to the number of bytes

ECCPGS	000	001	010	011	100	101
Number of bytes	256	512	1024	2048	4096	8192
ECC result bits	ECC[21: 0]	ECC[23: 0]	ECC[25: 0]	ECC[27: 0]	ECC[29: 0]	ECC[31:0]

24.5 PC card

The PC card interface can drive PC cards. They are made of three memory spaces: IO memory space, common memory space and attribute space, each of which has their individual on-chip signal and timing register. The three memory spaces are accessible through different timings.

24.5.1 Operating mode

Pin function:

Pin signals vary from external memory to external memory. Table 24-36 lists typical pin signals.

Table 24-36 Typical pin signals for PC card

Pin name	PC card
XMC_NCE4_1	Chip-select 1 (CE1)
XMC_NCE4_2	Chip-select 2 (CE2)
XMC_A[10:0]	Address bus
XMC_NOE	Output enable for common and attribute spaces
XMC_NWE	Write enable for common and attribute spaces
XMC_NIORD	Output enable for IO space
XMC_NIOWR	Write enable for IO space
XMC_NREG	attribute space select
XMC_D[15: 0]	Data bus
XMC_CD	PC card detection signal, active high
XMC_NWAIT	Ready/Busy (R/B)
XMC_INTR	PC card interrupt signal

Access address and Data access

The upper bytes of the HADDR bit is used to select a memory bank while the lower bytes to data memory address. Address translation between them is shown in Table 24-5. As long as read/write access to a specific address occurs, the XMC will enable chip-select signals and write/read operation to the external memories according to the HADDR bit.

In case that the AHB data width is not equal to that of the memories, the XMC will make appropriate arrangement according to the typical signals of the external memories. Table 24-37 lists the operation modes supported by XMC.

Table 24-37 Access data width and PC card data width

Memory	Mode	AHB data size	Memory data size	Description
PC card	Read/Write	8	16	Use XMC_NCE4_1
	Read/Write	16	16	Use XMC_NCE4_1 and XMC_NCE4_2
	Read/Write	32	16	Split into 2 XMC accesses and use XMC_NCE4_1 and XMC_NCE4_2

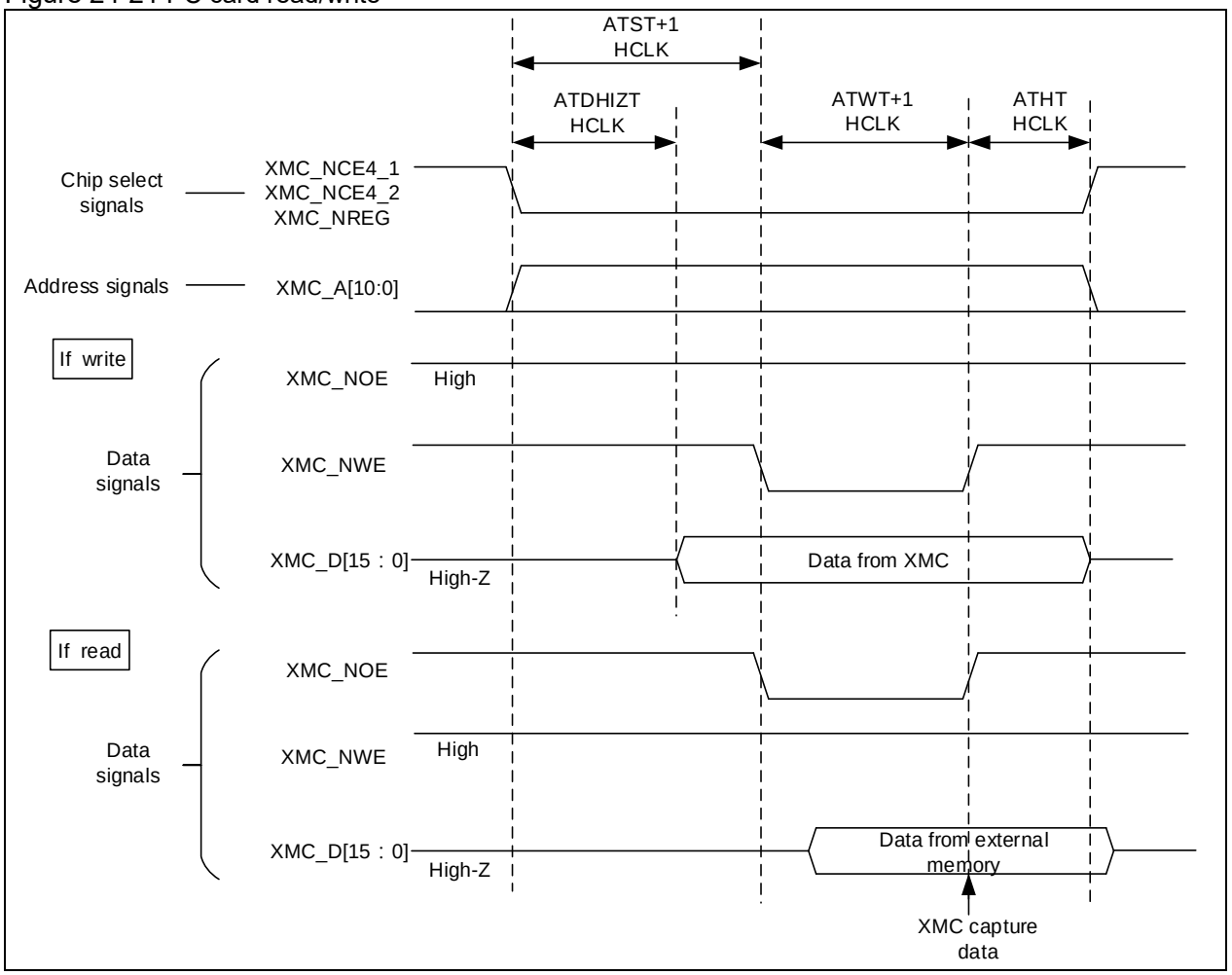
24.5.2 Access timings

The XMC access the NAND Flash according to the timing parameters, as shown in Table 24-38 and Figure 24-21. Users can perform programming operations according to the specifications of the external memory and application needs.

Table 24-38 PC card parameter register

Parameter register	Description	Access mode	Unit
CMDHIZT/ATDHIZT/IOHIZT	Memory data bus high-Z time	Write	HCLK cycles
CMST/ATST/IOST	Memory setup time	Read/Write	HCLK cycles
CMWT/ATWT/IOWT	Memory wait time	Read/Write	HCLK cycles
CMHT/ATHT/IOHT	Memory hold time	Read/Write	HCLK cycles

Figure 24-21 PC card read/write



24.6 SDRAM card

24.6.1 SDRAM access management

SDRAM initialization

The SDRAM initialization sequence defined in the JEDEC specification must be respected before using SDRAM.

The SDRAM initialization is managed by software. To initialize two devices, the BK1 and BK2 bit must be set in the SDRAM_CMD register. The initialization command is applicable to both devices.

1. Program the SDRAM_CTRLx register according to the external SDRAM device row/column and bus width and the number of BANK. Read and write operations from/to the SDRAM must be also programmed into the SDRAM_CTRLx register.
2. Program the memory device timing into the SDRAM_TMx register. The TRP and TRC timings must be programmed into the SDRAM_TM1 register.
3. Set CMD=001 to start providing the clock to the memory.
4. Wait for 100us
5. Set CMD=010 to issue a “Precharge All” command.
6. Set CMD=011 and configure the number of consecutive Auto-refresh commands (ART). Typical number is 8.
7. Configure the MRD field (controller supports BL=1 only). Set CMD=100 to issue a “Load Mode Register” command. If the mode registers of these two SDRAM devices are different from each other, this step has to be repeated twice (by setting the BK1 and BK2)
8. Program the refresh counter in the SDRAM_RCNT register to determine the SDRAM auto-refresh rate.

After the initialization, the SDRAM is ready to accept commands. If a system reset occurs during an on-going SDRAM access, the SDRAM device has to be reset after re-initialization in order to issue a new access to the SDRAM.

If the “Load Mode Register” and “Self-Refresh” commands are applied to both devices at the same time, access commands are issued using the timing parameters configured for SDRAM device 1.

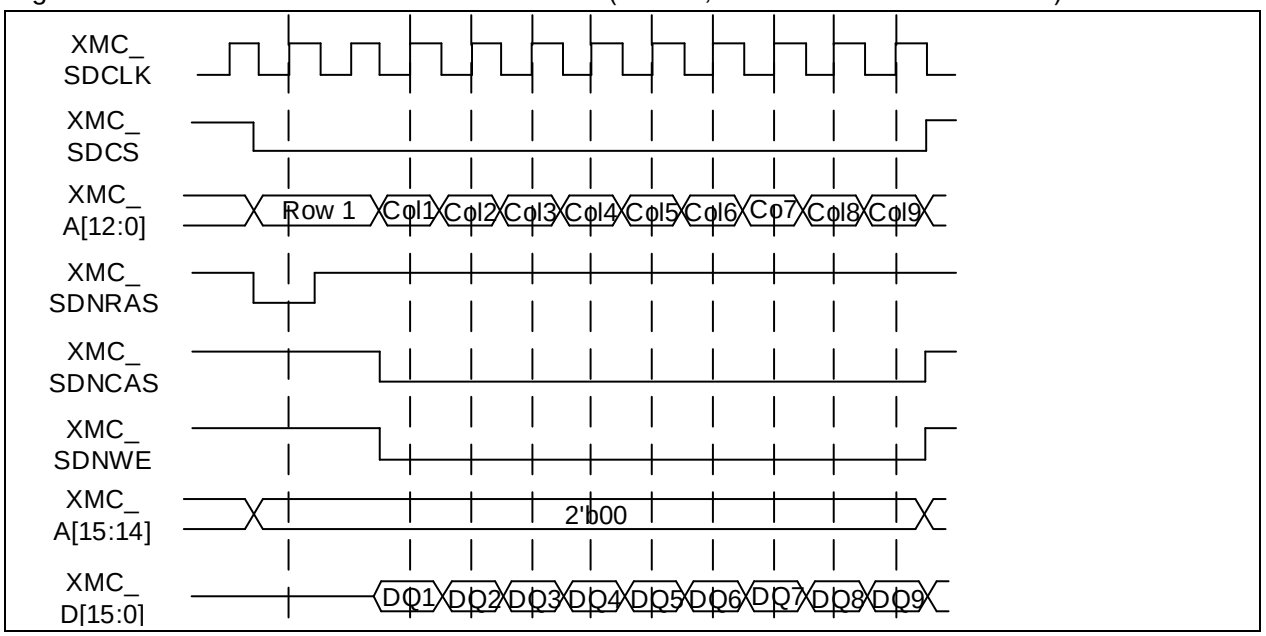
SDRAM controller write operation

Before performing any write access, the SDRAM write protection must be disabled by clearing the WRP bit in the SDRAM_CTRLx register, otherwise, an AHB error may occur.

The SDRAM controller translates single or burst AHB write requests into a single SDRAM write access, which can be done by setting BL=1. In both cases, the SDRAM controller always tries to translate consecutive or discontinuous AHB write requests into a single SDRAM write request in order to increase efficiency.

The SDRAM controller keeps track of the active row for each BANK in order to be able to perform consecutive write accesses to different banks (multibank ping-pong access). If the next write access is in the same row or in another active row, the write operation is performed. If the next write access targets an inactive row, the SDRAM controller generates a precharge command (The BANK where the inactive row is has other open rows, if no other rows, the precharge is unnecessary), and activates the new row and initiates a write operation.

Figure 24-22 SDRAM write access waveforms (Trcd=2, 9 consecutive write access)



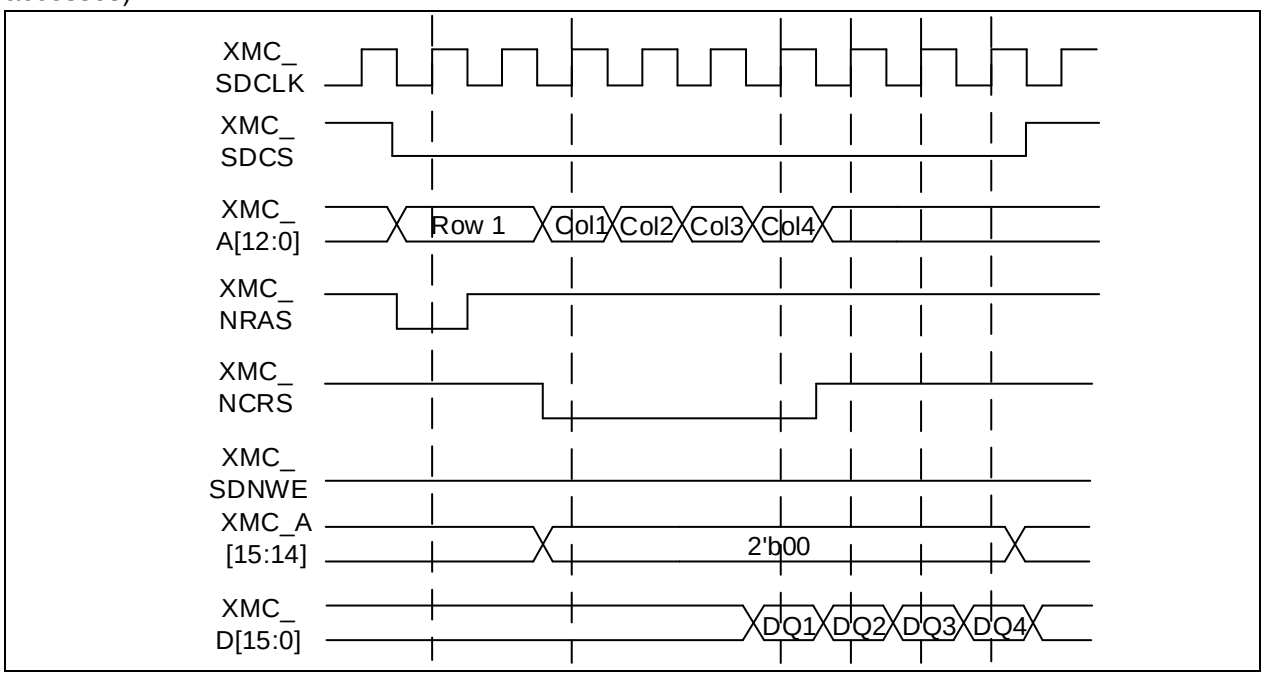
SDRAM controller read operation

The SDRAM controller translates single or burst AHB read requests into a single SDRAM read access, which can be done by setting BL=1.

The SDRAM controller keeps track of the active row for each BANK in order to be able to perform consecutive read accesses to different banks (multibank ping-pong access). If the next read access is in the same row or in another active row, the read operation is performed. If the next read access targets an inactive row, the SDRAM controller generates a precharge command (The BANK where the inactive row is has other open rows, if no other rows, the precharge is unnecessary), and activates the new row and initiates a read operation.

Note: The above-mentioned description applies to the scenario where the SDRAM consecutive read feature is not enabled (that is, read FIFO is unused.)

Figure 24-23 SDRAM read access without using read FIFO (Trcd=2,CL=3, and 4 consecutive read accesses)



When the BSTR is set in the SDRAM_CTRL1 register, the SDRAM controller continues to send another read command after issuing the current read command. The number of read commands to be issues depends on the CAS latency and RD read delay. The data read in advance during the CAS latency is stored into the read FIFO. Each line of the read FIFO has its address tag. The number of the anticipated read commands (M) is calculated as follows:

M = CAS cycles+RD/2+1

For example, CAS=3 and RD= 3xHCLK, then 5 data (not committed) are stored in the FIFO.

Each time a read request occurs, the SDRAM controller will check whether the FIFO address tag matches the address to be read. If matched, take the desired data from the FIFO; if unmatched, issue a read command to the memory, and update FIFO at the same time.

Row and BANK boundary management

When a read or write access crosses a row boundary, if the next read or write access is sequential and the current access was performed to a row boundary, then the SDRAM controller precharges the currently active row in order to disable the current row, activates a new row, and starts a read/write command to a new row.

If the next access is sequential and the current access crosses BANK boundary, two cases are possible:

- If the current BANK is not the last one, the SDRAM will perform access to the next BANK. If the next BANK is an active row and the active row matches the one to be accessed, a read operation is issued; if such active row is not the one to be accessed, disable the active row and activate a new row; if no active row, activate directly a new row to be accessed, and issue a read/write command.
- If the current BANK is the last memory area, an AHB error occurs.

SDRAM controller self-refresh operation

The SDRAM device needs an auto-refresh operation. The SDRAM controller periodically issues auto-refresh commands. The refresh rate is defined by the RC value in the SDRAM_RCNT register. The auto-refresh commands have priority over read/write accesses. In other words, if the auto-refresh commands and read/write access commands are generated simultaneously, the read/write requests are suspended temporarily, and are processed when the auto-refresh is complete.

If the SDRAM has active rows, the SDRAM controller generates all precharge commands before auto-refresh operations.

If a new auto-refresh request occurs while the previous one was not complete, the ERR (Refresh Error) bit is set in the SDRAM_STS register. An interrupt is generated if the ERIEN has been set.

24.6.2 Self-refresh mode and Power-down mode

The SDRAM controller supports two low-power modes: self-refresh mode and power-down mode.

Self-refresh mode

This mode is selected by setting CMD=101 and by configuring the Target Bank bits (BK1 and/or BK2) in the SDRAM_CMD register.

The SDRAM clock stops running after a TRAS delay. The internal refresh time stops counting if one of the following conditions is present:

When both SDRAM devices have been initialized, a self-refresh command is issued to both devices; or when one of the devices is not initialized, a self-refresh command is issued to one of the device.

After entering the self-refresh mode, the SDRAM device must remain in this mode for a minimum period of time of TRAS, which is determined by the SDRAM device characteristics. To guarantee this minimum period, the BUSY flag will be set after the self-refresh is entered. The BUSY flag is cleared automatically only after a TRAS time. This bit can be used to judge whether to exit self-refresh mode.

It is possible to leave self-refresh mode by setting CMD=000 or by read/write access to the SDRAM.

Power-down mode

This mode is selected by setting CMD=110 and by configuring the Target Bank bits (BK1 and/or BK2) in the SDRAM_CMD register.

During power-down mode, the SDRAM device can also perform auto-refresh operation. When an auto-refresh command occurs, the SDRAM exists from the power-down mode before performing auto-refresh operation. After that, the SDRAM will enter power-down mode again.

It is possible to leave power-down mode by setting CMD=000 or by read/write access to the SDRAM.

24.7 XMC registers

These peripheral registers must be accessed by word (32 bits).

Table 24-39 XMC register address mapping

Register name	Offset	Reset value
XMC_BK1CTRL1	0x000	0x0000 30DB
XMC_BK1TMG1	0x004	0x0FFF FFFF
XMC_BK1CTRL2	0x008	0x0000 30D2
XMC_BK1TMG2	0x00C	0x0FFF FFFF
XMC_BK1CTRL3	0x010	0x0000 30D2
XMC_BK1TMG3	0x014	0x0FFF FFFF
XMC_BK1CTRL4	0x018	0x0000 30D2
XMC_BK1TMG4	0x01C	0x0FFF FFFF
XMC_BK2CTRL	0x060	0x0000 0018
XMC_BK2IS	0x064	0x0000 0040
XMC_BK2TMGRG	0x068	0xFCFC FCFC
XMC_BK2TMGSP	0x06C	0xFCFC FCFC
XMC_BK2ECC	0x074	0x0000 0000
XMC_BK3CTRL	0x080	0x0000 0018
XMC_BK3IS	0x084	0x0000 0040
XMC_BK3TMGRG	0x088	0xFCFC FCFC
XMC_BK3TMGSP	0x08C	0xFCFC FCFC
XMC_BK3ECC	0x094	0x0000 0000
XMC_BK4CTRL	0x0A0	0x0000 0018
XMC_BK4IS	0x0A4	0x0000 0040

	XMC_BK4TMGCM	0x0A8	0xFCFC FCFC
	XMC_BK4TMGAT	0x0AC	0xFCFC FCFC
	XMC_BK4TMGIO	0xB0	0xFCFC FCFC
	XMC_BK1TMGWR1	0x104	0x0FFF FFFF
	XMC_BK1TMGWR2	0x10C	0x0FFF FFFF
	XMC_BK1TMGWR3	0x114	0x0FFF FFFF
	XMC_BK1TMGWR4	0x11C	0x0FFF FFFF
	XMC_EXT1	0x220	0x0000 0808
	XMC_EXT2	0x224	0x0000 0808
	XMC_EXT3	0x228	0x0000 0808
	XMC_EXT4	0x22C	0x0000 0808
	SDRAM_CTRL1	0x140	0x0000 02D0
	SDRAM_ CTRL2	0x144	0x0000 02D0
	SDRAM_TM1	0x148	0x0FFF FFFF
	SDRAM_TM2	0x14C	0x0FFF FFFF
	SDRAM_CMD	0x150	0x0000 0007
	SDRAM_RCNT	0x154	0x0000 0000
	SDRAM_STS	0x158	0x0000 0000

24.7.1 NOR Flash and PSRAM control registers

24.7.1.1 SRAM/NOR Flash chip select control register 1 (XMC_BK1CTRL1)

Accessed by words

Bit	Register	Reset value	Type	Description
Bit 31: 20	Reserved	0x000	resd	Kept at its default value.
Bit 19	MWMC	0x0	rw	Memory write mode control 0: Write operations are performed in asynchronous mode 1: Write operations are performed in synchronous mode
Bit 18: 16	CRPGS	0x0	rw	CRAM page size Cellular RAM 1.5 does not allow synchronous access to cross the address boundaries between pages. When these bits are configured in synchronous mode, the XMC will automatically split the access when the page size is reached. 000: No split access when crossing address boundary (default value) 001: 128 bytes 010: 256 bytes 011: 512 bytes 100: 1024 bytes Others: Reserved.
Bit 15	NWASEN	0x0	rw	NWAIT in asynchronous transfer enable 0: NWAIT signal is disabled 1: NWAIT signal is enable
Bit 14	RWTD	0x0	rw	Read-write timing different Different timings are used for read and write operations, that is, the XMC_BK1TMGWR register is enabled. 0: Same timings for read and write operations 1: Different timings for read and write operations
Bit 13	NWSEN	0x1	rw	NWAIT enable during synchronous transfer 0: NWAIT signal is disabled

				1: NWAIT signal is enabled
Bit 12	WEN	0x1	rw	Write enable 0: Disabled 1: Enabled
Bit 11	NWTCFG	0x0	rw	NWAIT timing configuration It is valid only in synchronous mode. 0: NWAIT signal is active one data cycle before the wait state 1: NWAIT signal is active one data cycle during the wait state
Bit 10	WRAPEN	0x0	rw	Wrapped enable This bit defines whether the XMC will split a wrapped AHB access into two accesses. 0: Direct wrapped access is not allowed 1: Direct wrapped access is allowed
Bit 9	NWPOL	0x0	rw	NWAIT polarity This bit defines the polarity of the NWAIT signal in synchronous mode. 0: NWAIT active low 1: NWAIT active high
Bit 8	SYNCBEN	0x0	rw	Synchronous burst enable This bit allows synchronous access to Flash memories. 0: Synchronous burst disabled 1: Synchronous burst enabled
Bit 7	Reserved	0x1	resd	Kept at its default value.
Bit 6	NOREN	0x1	rw	Nor flash access enable 0: Nor flash access is disabled 1: Nor flash access is enabled
Bit 5: 4	EXTMDBW	0x1	rw	External memory data bus width This field defines the external memory data bus width. 00: 8 bits 01: 16 bits 10: Reserved 11: Reserved
Bit 3: 2	DEV	0x2	rw	Memory device type 00: SRAM/ROM 01: PSRAM (Cellular RAM or CRAM) 10: NOR Flash 11: Reserved
Bit 1	ADMUXEN	0x1	rw	Address/data multiplexing enable 0: Address/data not multiplexed 1: Address/data multiplexed
Bit 0	EN	0x1	rw	Memory bank enable 0: Memory bank disabled 1: Memory bank enabled

24.7.1.2 SRAM/NOR Flash chip select control register x (x=2, 3, 4)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 20	Reserved	0x000	resd	Kept at its default value.
Bit 19	MWMC	0x0	rw	Memory write mode control 0: Write operations are performed in asynchronous mode 1: Write operations are performed in synchronous mode
Bit 18: 16	CRPGS	0x0	rw	CRAM page size Cellular RAM 1.5 does not allow synchronous access to cross the address boundaries between pages. When these bits are configured in synchronous mode, the XMC will automatically split the access when the page size is reached. 000: No split access when crossing address boundary (default value) 001: 128 bytes 010: 256 bytes 011: 512 bytes

				100: 1024 bytes Others: Reserved.
Bit 15	NWASEN	0x0	rw	NWAIT enable during asynchronous transfer 0: NWAIT signal is disabled 1: NWAIT signal is enable
Bit 14	RWTD	0x0	rw	Read-write timing different Different timings are used for read and write operations, that is, the XMC_BK1TMGWR register is enabled. 0: Same timings for read and write operations 1: Different timings for read and write operations
Bit 13	NWSEN	0x1	rw	NWAIT enable during synchronous transfer 0: NWAIT signal is disabled 1: NWAIT signal is enabled
Bit 12	WEN	0x1	rw	Write enable 0: Disabled 1: Enabled
Bit 11	NWTCFG	0x0	rw	NWAIT timing configuration It is valid only in synchronous mode. 0: NWAIT signal is active one data cycle before the wait state 1: NWAIT signal is active one data cycle during the wait state
Bit 10	WRAPEN	0x0	rw	Wrapped enable This bit defines whether the XMC will split a wrapped AHB access into two accesses. 0: Direct wrapped access is not allowed 1: Direct wrapped access is allowed
Bit 9	NWPOL	0x0	rw	NWAIT polarity This bit defines the polarity of the NWAIT signal in synchronous mode. 0: NWAIT active low 1: NWAIT active high
Bit 8	SYNCBEN	0x0	rw	Synchronous burst enable This bit allows synchronous access to Flash memories. 0: Synchronous burst disabled 1: Synchronous burst enabled
Bit 7	Reserved	0x1	resd	Kept at its default value.
Bit 6	NOREN	0x1	rw	Nor flash access enable 0: Nor flash access is disabled 1: Nor flash access is enabled
Bit 5: 4	EXTMDBW	0x1	rw	External memory data bus width This field defines the external memory data bus width. 00: 8 bits 01: 16 bits 10: Reserved 11: Reserved
Bit 3: 2	DEV	0x0	rw	Memory device type 00: SRAM/ROM 01: PSRAM (Cellular RAM or CRAM) 10: NOR Flash 11: Reserved
Bit 1	ADMUXEN	0x1	rw	Address/data multiplexing enable 0: Address/data not multiplexed 1: Address/data multiplexed
Bit 0	EN	0x0	rw	Memory bank enable 0: Memory bank disabled 1: Memory bank enabled

24.7.1.3 SRAM/NOR Flash chip select timing register x (x=1,2,3,4)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value.
Bit 29: 28	ASYNCM	0x0	rw	Asynchronous mode This field is valid only when the RWTD bit is enabled. 00: Mode A 01: Mode B 10: Mode C 11: Mode D
Bit 27: 24	DTLAT	0xF	rw	Data latency This field is valid only in synchronous mode. 0000: 0 XMC_CLK cycle is inserted 0001: 1 additional XMC_CLK cycle is inserted …… 1111: 15 additional XMC_CLK cycles are inserted
Bit 23: 20	CLKPSC	0xF	rw	Clock prescaler This field is valid only in synchronous mode. It defines the frequency of the XMC_CLK clock. 0000: Reserved 0001: XMC_CLK cycle= 2 x HCLK clock cycles 0010: XMC_CLK cycle =3 x HCLK clock cycles …… 1111: XMC_CLK cycle = 6 x HCLK cycles
Bit 19: 16	BUSLAT	0xF	rw	Bus latency To avoid data bus conflict, a latency is inserted on the data bus after one read operation in multiplexed or synchronous mode. 0000: 1 HCLK cycle is inserted 0001: 2 HCLK cycles are inserted …… 1111: 16 HCLK cycles are inserted
Bit 15: 8	DTST	0xFF	rw	Data setup time 0000: 0 HCLK cycle is inserted 0001: 1 additional HCLK cycle is inserted …… 1111: 15 additional HCLK cycles are inserted
Bit 7: 4	ADDRHT	0xF	rw	Address-hold time 0000: 0 HCLK cycle is inserted 0001: 1 additional HCLK cycle is inserted …… 1111: 15 additional HCLK cycles are inserted
Bit 3: 0	ADDRST	0xF	rw	Address setup time 0000: 0 HCLK cycle is inserted 0001: 1 additional HCLK cycle is inserted …… 1111: 15 additional HCLK cycles are inserted

24.7.1.4 SRAM/NOR Flash write timing register x (x=1,2,3,4)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value.
Bit 29: 28	ASYNCM	0x0	rw	Asynchronous mode This field is valid only when the RWTD bit is enabled. 00: Mode A 01: Mode B 10: Mode C 11: Mode D
Bit 27: 20	Reserved	0xFF	resd	Kept at its default value.
Bit 19: 16	BUSLAT	0xF	rw	Bus latency To avoid data bus conflict, a latency is inserted on the data

				bus after one read operation in multiplexed or synchronous mode. 0000: 1 HCLK cycle is inserted 0001: 2 HCLK cycles are inserted …… 1111: 16 HCLK cycles are inserted
Bit 15: 8	DTST	0xFF	rw	Data setup time 0000: 0 HCLK cycle is inserted 0001: 1 additional HCLK cycle is inserted …… 1111: 15 additional HCLK cycles are inserted
Bit 7: 4	ADDRHT	0xF	rw	Address-hold time 0000: 0 HCLK cycle is inserted 0001: 1 additional HCLK cycle is inserted …… 1111: 15 additional HCLK cycles are inserted
Bit 3: 0	ADDRST	0xF	rw	Address setup time 0000: 0 HCLK cycle is inserted 0001: 1 additional HCLK cycle is inserted …… 1111: 15 additional HCLK cycles are inserted

24.7.1.5 SRAM/NOR Flash extra timing register x (XMC_EXTx) (x=1,2,3,4)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 8	BUSLATR2R	0x08	rw	Bus turnaround phase for consecutive read duration This field is used to define the bus turnaround phase duration for consecutive read operations. A delay is inserted between two consecutive read operations in order to avoid bus conflicts. 00000000: 1 HCLK cycle is inserted for consecutive read operations 00000001: 2 HCLK cycles are inserted for consecutive read operations …… 00001000: 9 HCLK cycles are inserted for consecutive read operations (default value) …… 11111111: 256 HCLK cycles are inserted for consecutive read operations
Bit 7: 0	BUSLATW2W	0x08	rw	Bus turnaround phase for consecutive write duration This field is used to define the bus turnaround phase duration for consecutive write operations. A delay is inserted between two consecutive write operations in order to avoid bus conflicts. 00000000: 1 HCLK cycle is inserted for consecutive write operations 00000001: 2 HCLK cycles are inserted for consecutive write operations …… 00001000: 9 HCLK cycles are inserted for consecutive write operations (default value) …… 11111111: 256 HCLK cycles are inserted for consecutive write operations

24.7.2 NAND Flash control registers

24.7.2.1 NAND Flash control register x (XMC_BKxCTRL) (x=2,3)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 20	Reserved	0x000	resd	Kept at its default value.
Bit 19: 17	ECCPGS	0x0	rw	ECC page size 000: 256 bytes 001: 512 bytes 010: 1024 bytes 011: 2048 bytes 100: 4096 bytes 101: 8192 bytes
Bit 16: 13	TAR	0x0	rw	ALE to RE delay This field specifies the delay from the falling edge of the ALE to that of the RE. 0000: 1 HCLK cycle …… 1111: 16 HCLK cycles
Bit 12: 9	TCR	0x0	rw	CLE to RE delay This field specifies the delay from the falling edge of the CLE to that of the RE. 0000: 1 HCLK cycle …… 1111: 16 HCLK cycles
Bit 8: 7	Reserved	0x0	resd	Kept at its default value.
Bit 6	ECCEN	0x0	rw	ECC enable 0: ECC disabled 1: ECC enabled
Bit 5: 4	EXTMDBW	0x1	rw	External memory data bus width This field specifies the external NAND Flash width. 00: 8 bits 01: 16 bits 10: Reserved 11: Reserved
Bit 3	DEV	0x1	rw	Memory device type 0: Reserved 1: NAND Flash
Bit 2	EN	0x0	rw	Memory bank enable 0: Memory bank disabled 1: Memory bank enabled
Bit 1	NWEN	0x0	rw	Wait feature enable This bit is used to enable NAND Flash wait function. 0: Disabled 1: Enabled
Bit 0	Reserved	0x0	resd	Kept at its default value.

24.7.2.2 Interrupt enable and FIFO status register x (XMC_BKxIS) (x=2,3)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 7	Reserved	0x000000	resd	Kept at its default value.
Bit 6	FIFOE	0x1	rw	FIFO empty This bit is set by hardware when the FIFO is empty. 0: FIFO is not empty 1: FIFO is empty XMC FIFO size is 16 words. It is used to store the data from AHB.
Bit 5	FEIEN	0x0	rw	Falling edge interrupt enable 0: Falling edge interrupt disabled 1: Falling edge interrupt enabled
Bit 4	HLIEN	0x0	rw	High-level interrupt enable

				0: High-level interrupt disabled 1: High-level interrupt enabled
Bit 3	REIEN	0x0	rw	Rising edge interrupt enable 0: Rising edge interrupt disabled 1: Rising edge interrupt enabled
Bit 2	FES	0x0	rw	Falling edge status This bit is set by hardware and cleared by software. 0: No falling edge interrupt generated 1: Falling edge interrupt generated
Bit 1	HLS	0x0	rw	High-level status This bit is set by hardware and cleared by software. 0: No high level interrupt generated 1: High level interrupt generated
Bit 0	RES	0x0	rw	Rising edge status This bit is set by hardware and cleared by software. 0: No rising edge interrupt generated 1: Rising edge interrupt generated

24.7.2.3 Regular memory timing register x (XMC_ BKxTMGRG) (x=2,3)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	RGDHIZT	0xFC	rw	Regular memory databus High resistance time This field defines the databus high resistance duration when write access to NAND Flash is started in a regular space. 00000000: 0 HCLK cycle is inserted 00000001: 1 additional HCLK cycle is inserted …… 11111111: 255 additional HCLK cycles are inserted
Bit 23: 16	RGHT	0xFC	rw	Regular memory hold time This field defines the databus hold time when access to NAND Flash in a regular memory. 00000000: Reserved 00000001: 1 HCLK cycle is inserted …… 11111111: 255 HCLK cycles are inserted
Bit 15: 8	RGWT	0xFC	rw	Regular memory wait time Specifies the regular memory wait time when the XMC_NWE and XMC_NOE is low. 00000000: 0 HCLK cycle is inserted 00000001: 1 additional HCLK cycle is inserted …… 11111111: 255 additional HCLK cycles are inserted
Bit 7: 0	RGST	0xFC	rw	Regular memory setup time This field defines the address setup time when access to NAND Flash in a regular memory. 00000000: 0 HCLK cycle is inserted 00000001: 1 additional HCLK cycle is inserted …… 11111111: 255 additional HCLK cycles are inserted

24.7.2.4 Special memory timing register x (XMC_ BKxTMGSP) (x=2,3)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	SPDHIZT	0xFC	rw	Special memory databus High resistance time This field defines the databus high resistance duration when write access to NAND Flash is started in a special space. 00000000: 0 HCLK cycle is inserted 00000001: 1 additional HCLK cycle is inserted …… 11111111: 255 additional HCLK cycles are inserted
Bit 23: 16	SPHT	0xFC	rw	Special memory hold time

				This field defines the databus hold time when access to NAND Flash in a special memory. 00000000: Reserved 00000001: 1 HCLK cycle is inserted …… 11111111: 255 HCLK cycles are inserted
Bit 15: 8	SPWT	0xFC	rw	Special memory wait time Specifies the special memory wait time when the XMC_NWE and XMC_NOE is low. 00000000: 0 HCLK cycle is inserted 00000001: 1 additional HCLK cycle is inserted …… 11111111: 255 additional HCLK cycles are inserted
Bit 7: 0	SPST	0xFC	rw	Special memory setup time This field defines the address setup time when access to NAND Flash in a special memory. 00000000: 0 HCLK cycle is inserted 00000001: 1 additional HCLK cycle is inserted …… 11111111: 255 additional HCLK cycles are inserted

24.7.2.5 ECC value register x (XMC_ BKxCC) (x=2,3)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	ECC	0x0000 0000	ro	EECC value This field contains the computed ECC value.

24.7.3 PC card controller registers

24.7.3.1 PC card control register (XMC_BK4CTRL)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 3	Reserved	0x000	resd	Kept at its default value.
Bit 2	EN	0x0	rw	Memory bank enable 0: Memory bank disabled 1: Memory bank enabled
Bit 1	NWEN	0x0	rw	Wait feature enable This bit is used to enable PC card memory bank wait feature. 0: Wait feature disabled 1: Wait feature enabled
Bit 0	Reserved	0x0	resd	Kept at its default value.

24.7.3.2 Interrupt enable and FIFO status register 4 (XMC_BK4IS)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 7	Reserved	0x000000	resd	Kept at its default value.
Bit 6	FIFOE	0x1	rw	FIFO empty This bit is set by hardware when the FIFO is empty. 0: FIFO is not empty 1: FIFO is empty XMC FIFO size is 16 words. It is used to store the data from AHB.
Bit 5	FEIEN	0x0	rw	Falling edge interrupt enable 0: Falling edge interrupt disabled 1: Falling edge interrupt enabled
Bit 4	HLIEN	0x0	rw	High-level interrupt enable 0: High-level interrupt disabled 1: High-level interrupt enabled
Bit 3	REIEN	0x0	rw	Rising edge interrupt enable 0: Rising edge interrupt disabled 1: Rising edge interrupt enabled

Bit 2	FES	0x0	rw	Falling edge status This bit is set by hardware and cleared by software. 0: No falling edge interrupt generated 1: Falling edge interrupt generated
Bit 1	HLS	0x0	rw	High-level status This bit is set by hardware and cleared by software. 0: No high level interrupt generated 1: High level interrupt generated
Bit 0	RES	0x0	rw	Rising edge status This bit is set by hardware and cleared by software. 0: No rising edge interrupt generated 1: Rising edge interrupt generated

24.7.3.3 Common memory timing register 4 (XMC_ BK4TMGCM)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	CMDHIZT	0xFC	rw	Common memory databus High resistance time This field defines the databus high resistance duration when write access to NAND Flash is started in a common space. 00000000: 0 HCLK cycle is inserted 00000001: 1 additional HCLK cycle is inserted …… 11111111: 255 additional HCLK cycles are inserted
Bit 23: 16	CMHT	0xFC	rw	Common memory hold time This field defines the databus hold time when access to NAND Flash in a common space 00000000: Reserved 00000001: 1 HCLK cycle is inserted …… 11111111: 255 HCLK cycles are inserted
Bit 15: 8	CMWT	0xFC	rw	Common memory wait time Specifies the common memory wait time when the XMC_NWE and XMC_NOE is low. 00000000: 0 HCLK cycle is inserted 00000001: 1 additional HCLK cycle is inserted …… 11111111: 255 additional HCLK cycles are inserted
Bit 7: 0	CMST	0xFC	rw	Common memory setup time This field defines the address setup time when access to NAND Flash in a regular memory. 00000000: 0 HCLK cycle is inserted 00000001: 1 additional HCLK cycle is inserted …… 11111111: 255 additional HCLK cycles are inserted

24.7.3.4 Attribute memory timing register 4 (XMC_ BK4TMGAT)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	ATDHIZT	0xFC	rw	Attribute memory databus High resistance time This field defines the databus high resistance duration when write access to NAND Flash is started in an attribute space. 00000000: 0 HCLK cycle is inserted 00000001: 1 additional HCLK cycle is inserted …… 11111111: 255 additional HCLK cycles are inserted
Bit 23: 16	ATHT	0xFC	rw	Attribute memory hold time This field defines the databus hold time when access to NAND Flash in an attribute space. 00000000: Reserved 00000001: 1 HCLK cycle is inserted …… 11111111: 255 HCLK cycles are inserted

Bit 15: 8	ATWT	0xFC	rw	Attribute memory wait time Specifies the attribute memory wait time when the XMC_NWE and XMC_NOE is low. 00000000: 0 HCLK cycle is inserted 00000001: 1 additional HCLK cycle is inserted …… 11111111: 255 additional HCLK cycles are inserted
Bit 7: 0	ATST	0xFC	rw	Attribute memory setup time This field defines the address setup time when access to NAND Flash in an attribute space. 00000000: 0 HCLK cycle is inserted 00000001: 1 additional HCLK cycle is inserted …… 11111111: 255 additional HCLK cycles are inserted

24.7.3.5IO space timing register 4 (XMC_ BK4TMGIO)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	IODHIZT	0xFC	rw	I/O space databus High resistance time This field defines the databus high resistance duration when write access to NAND Flash is started in an IO space. 00000000: 0 HCLK cycle is inserted 00000001: 1 additional HCLK cycle is inserted …… 11111111: 255 additional HCLK cycles are inserted
Bit 23: 16	IOHT	0xFC	rw	I/O space hold time This field defines the databus hold time when access to NAND Flash in an IO space. 00000000: Reserved 00000001: 1 HCLK cycle is inserted …… 11111111: 255 HCLK cycles are inserted
Bit 15: 8	ATWT	0xFC	rw	Attribute memory wait time Specifies the IO wait time when the XMC_NWE and XMC_NOE is low. 00000000: 0 HCLK cycle is inserted 00000001: 1 additional HCLK cycle is inserted …… 11111111: 255 additional HCLK cycles are inserted
Bit 7: 0	IOST	0xFC	rw	IO space setup time This field defines the address setup time when access to NAND Flash in an IO space. 00000000: 0 HCLK cycle is inserted 00000001: 1 additional HCLK cycle is inserted …… 11111111: 255 additional HCLK cycles are inserted

24.7.4 SDRAM controller registers

24.7.4.1SDRAM control register 1, 2 (SDRAM_CTRL1,SDRAM_CTRL2)

This register contains the control parameters for each SDRAM memory bank.

Bit	Register	Reset value	Type	Description
Bit 31: 15	Reserved	0x00000	resd	Kept at its default value.
Bit 14: 13	RD	0x0	rw	Read delay This field defines the delay (in HCLK clock cycles) for reading data after CAS latency. 00: No HCLK clock cycle delay 01: 1 HCLK clock cycle delay 10: 2 HCLK clock cycle delay 11: Reserved, do not use. Note: The corresponding bits in the CTRL2 register are “don’t care bit”
Bit 12	BSTR	0x0	rw	Burst read

				When this bit is set, it indicates that single AHB requests (single or burst) are processed as bursts. Several data will be prefetched and stored into the FIFO. 0: Single read requests are not processed as bursts 1: ingle read requests are always processed as bursts Note: The corresponding bits in the CTRL2 register are “don’t care bit”
Bit 11: 10	CLKDIV	0x0	rw	Clock division configuration SDRAM clock configuration. 00: SDCLK clock disabled 01: HCLK/4 10: HCLK/2 11: HCLK/3 Note: The corresponding bits in the CTRL2 register are “don’t care bit”
Bit 9	WRP	0x0	rw	Write protection This bit is set to enable the SDRAM write protection. 0: Write access allowed 1: Write access forbidden
Bit 8: 7	CAS	0x0	rw	CAS latency This field is used to select CAS latency. 00: Reserved, do not use. 01: 1 cycle 10: 2 cycles 11: 3 cycles
Bit 6	INBK	0x0	rw	Internal banks This bit is used to define the number of internal banks. 0: 2 internal BANKs 1: 4 internal BANKs
Bit 5: 4	DB	0x0	rw	SDRAM data bus This field enables 8-bit or 16-bit data bus width. 00: 8 bits 01: 16 bits 10: Reserved, do not use. 11: Reserved, do not use.
Bit 3: 2	RA	0x0	rw	Row address This field defines the number of a row address, including 11 bits, 12 bits and 13 bits. 00: 11 bits 01: 12 bits 10: 13 bits 11: Reserved, do not use.
Bit 1: 0	CA	0x0	rw	Column address This field defines the number of a column address, including 8 bits, 9 bits, 10 bits and 13 bits. 00: 8 bits 01: 9 bits 10: 10 bits 11: 11 bits

24.7.4.2 SDRAM timing register 1, 2 (SDRAM_TM1,SDRAM_TM2)

This register contains the timing parameters of each SDRAM bank.

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0	resd	Kept at its default value.
Bit 27: 24	TRCD	0xF	rw	Row active to Read/Write delay This field defines the delay between the activate command and a read/write command in number of clock cycles. 0000: 1 cycle 0001: 2 cycles .... 1111: 16 cycles
Bit 23:20	TRP	0xF	rw	Precharge to active delay This field defines the delay between a precharge command and another command in number of clock

				cycles. 0000: 1 cycle 0001: 2 cycles .... 1111: 16 cycles
Bit 19: 16	TWR	0xF	rw	Write Recovery delay This field defines the delay between a write command and a precharge command in number of clock cycles. 0000: 1 cycle 0001: 2 cycles .... 1111: 16 cycles
Bit 15: 12	TRC	0xF	rw	Refresh to active delay This field defines the delay between the refresh command and the activate command, as well as the delay between two consecutive refresh commands. 0000: 1 cycle 0001: 2 cycles .... 1111: 16 cycles
Bit 11: 8	TRAS	0xF	rw	Self refresh time This field defines the minimum self-refresh period in umber of clock cycles. 0000: 1 cycle 0001: 2 cycles .... 1111: 16 cycles
Bit 7: 4	TXSR	0xF	rw	Exit Self-refresh to active delay This field defines the delay from exiting the self-refresh command to issuing an activate command in number of clock cycles. 0000: 1 cycle 0001: 2 cycles .... 1111: 16 cycles
Bit 3: 0	TMRD	0xF	rw	Load mode register program to active delay This field defines the delay between a load mode register command and an activate or refresh command in number of clock cycles. 0000: 1 cycle 0001: 2 cycles .... 1111: 16 cycles

Note: If two SDRAM devices are used, the TRP and TRC timings must be programmed with the timings of the slowest devices.

24.7.4.3 SDRAM command register (SDRAM_CMD)

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x000	resd	Kept at its default value.
Bit 22: 9	MRD	0x0000	rw	Mode Register data Refer to the SDRAM specifications for details.
Bit 8: 5	ART	0x0	rw	Auto-refresh times This field defines the number of consecutive auto-refresh commands issued when MODE =“011”. 0000: 1 auto-refresh cycle 0001: 2 auto-refresh cycles .... 1110: 15 auto-refresh cycles 1111: Reserved
Bit 4	BK1	0x0	wo	SDRAM Bank 1 This bit indicates whether the command will be sent to the SDRAM Bank 1. 0: Command not sent to SDRAM Bank 1 1: Command sent to SDRAM Bank 1
Bit 3	BK2	0x0	wo	SDRAM Bank 2 This bit indicates whether the command will be sent to the SDRAM Bank 2. 0: Command not sent to SDRAM Bank 2 1: Command sent to SDRAM Bank 2
Bit 2: 0	CMD	0x0	wo	SDRAM Command This field defines the command issued to the SDRAM device. 000: Normal mode 001: Clock configuration enable 010: Precharge all banks 011: Auto refresh 100: Load mode register 101: Self refresh 110: Power-down command 111: Reserved

24.7.4.4 SDRAM refresh timer register (SDRAM_RCNT)

This register is used to set the refresh rate of the SDRAM, in number of SDRAM CLK clock cycles.

The RC has to be configured as a non-zero value in order to perform a correct refresh operation. The RC value cannot be changed after initialization.

Refresh operation has priority over a read/write operation. However, if a read/write operation is in progress while a new refresh request occurs, the refresh operation starts only after the read/write operation is complete. It is recommended that the RC value is smaller than the calculated value in order to ensure the anticipated refresh rate.

This register is shared by the SDRAM Bank 1 and Bank 2.

Bit	Register	Reset value	Type	Description
Bit 31: 15	Reserved	0x00000	resd	Kept at its default value.
Bit 14	ERIEN	0x0	rw	Error interrupt enable 0: Error interrupt disabled 1: Error interrupt enabled
Bit 13: 1	RC	0x0000	rw	Refresh counter This 13-bit field defines the refresh rate of the SDRAM device. It is expressed in number of clock cycles. It must be set at least to 41 clock cycles. Refresh rate = (RC + 1) x SDRAM clock frequency RC = (SDRAM refresh period/number of rows) - 20
Bit 0	ERRC	0x0	wo	Error flag clear This bit is used to clear the error flag (ER) in the status register. 0: No effect 1: Refresh error flag is cleared.

Note: The programmed COUNT value must not be equal to the sum of the following timings: *TWR+TRP+TRC+TRCD+4 memory clock cycles.*

24.7.4.5 SDRAM status register (SDRAM_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 6	Reserved	0x0000000	resd	Kept at its default value.
Bit 5	BUSY	0x0	ro	Busy This bit indicates the status of the current SDRAM controller. 0: Idle 1: Busy
Bit 4: 3	BK2STS	0x0	ro	Bank2 status This field defines the status mode of the SDRAM Bank 2. 00: Normal mode 01: Auto-refresh mode 10: Power-down mode
Bit 2: 1	BK1STS	0x0	ro	Bank 1 Status This field defines the status mode of the SDRAM Bank 1. 00: Normal mode 01: Auto-refresh mode 10: Power-down mode
Bit 0	ERR	0x0	ro	Error flag 0: No refresh error has been detected 1: A refresh error has been detected

25 SDIO interface

25.1 SDIO introduction

The SD/SDIO MMC card host interface (SDIO) provides an interface between the AHB peripheral bus and MultiMediaCards (MMC), SD memory cards and SDIO cards.

SD memory card and SDI/O card system specifications are available through the SD card association website www.sdcard.org.

The MultiMediaCard system specifications published by the MMCA technical committee are available through the MultiMediaCard association website www.mmca.org.

25.2 SDIO main features

- Full compatibility with SD memory card specifications version 2.0
- Full compatibility with SDI/O card specification version 2.0 and support 1-bit and 4-bit databus modes.
- Full compatibility with MultiMedia card specification version 4.2 and support 1-bit, 4-bit and 8-bit databus modes.
- Full compatibility with previous versions of MultiMedia card specifications
- DMA transfer
- Data transfer up to 50 MHz in 8-bit bus mode
- Interrupt requests

Note: The SDIO is not compatible with SPI communication mode. It supports only one SD/SDIO/MMC 4.2 card at any one time.

Communication on the bus is based on command and data transfers.

- Command: A command is a token that starts an operation. Commands are sent from the host either to a single card (addressed command) or to all connected cards (broadcast command). Commands are transferred serially on the CMD line.
- Response: A response is a token that is sent from a card to the host as an answer to a previously received command. Responses are transferred serially on the CMD line.
- Data: Data can be transferred from the card to the host or vice versa. Data is transferred via the SDIO_D data line.

The basic operation on the MMC card/SD/SDI I/O bus is the command/response structure. These types of bus operation transfer their information through the command or bus mechanism. In addition, some operations have a data token.

Data transfers to/from SD/SDIO memory cards are done in data blocks. Data blocks are always followed by CRC bit, defining single and multiple block operations. Data transfers to/from MMC are done in data blocks or streams, as shown in Figure below.

Figure 25-1 SDIO “no response” and “response” operations

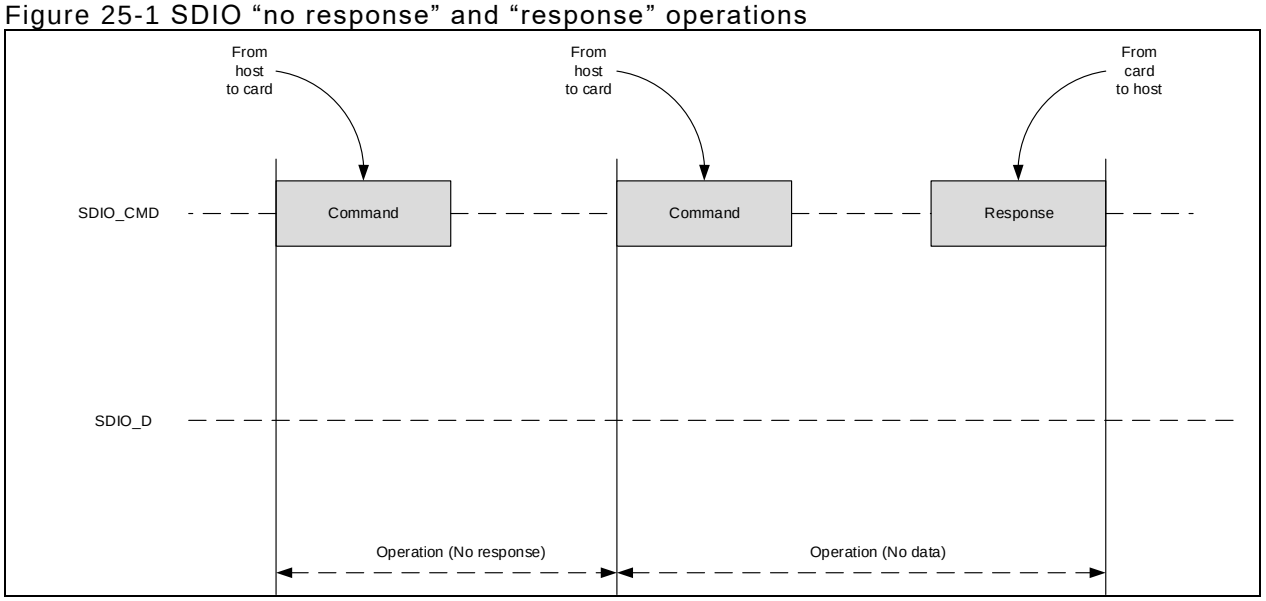


Figure 25-2 SDIO multiple block read operation

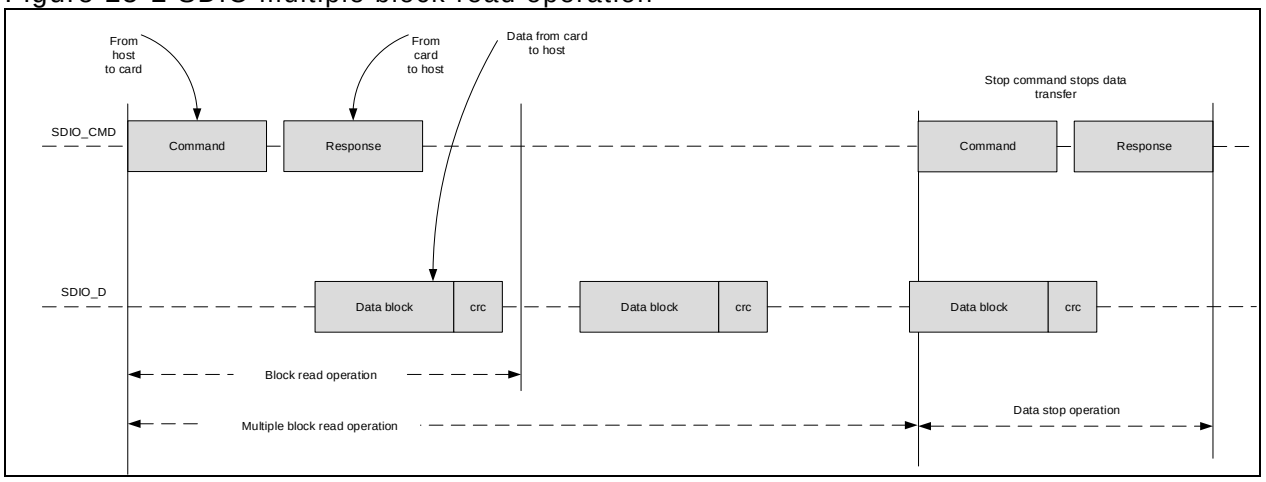
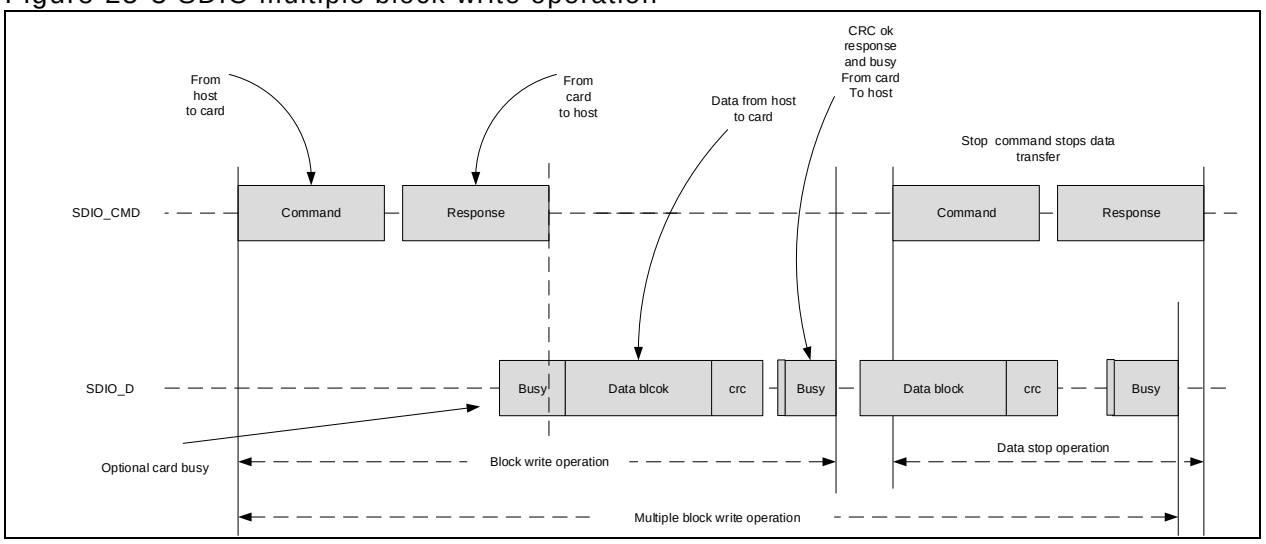


Figure 25-3 SDIO multiple block write operation



Note: The SDIO will not send any data as long as the Busy signal is set (SDIO_D0 pulled low).

Figure 25-4 SDIO sequential read operation

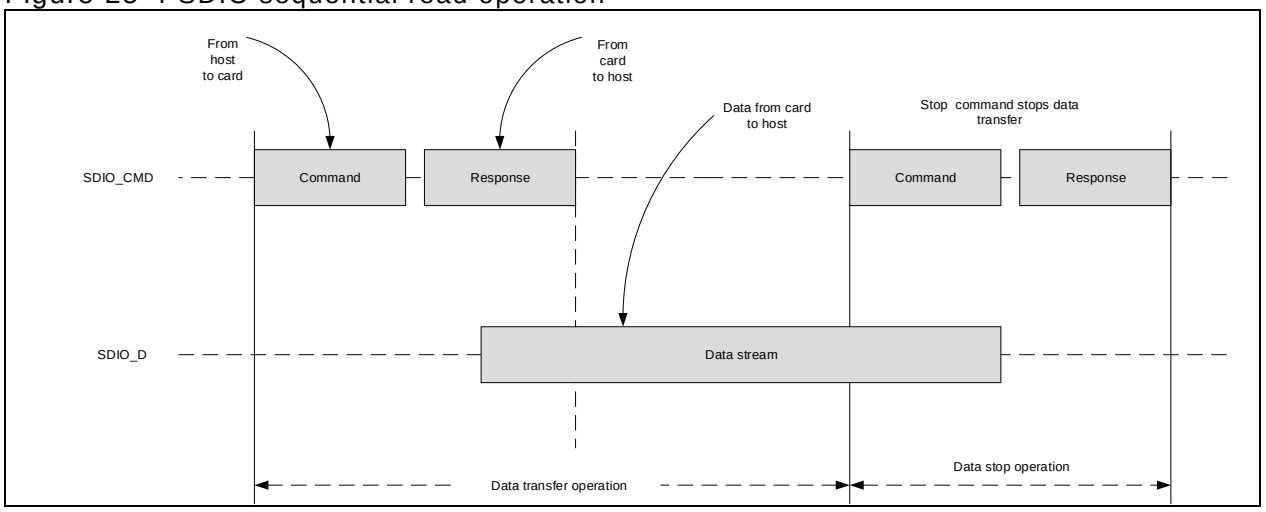
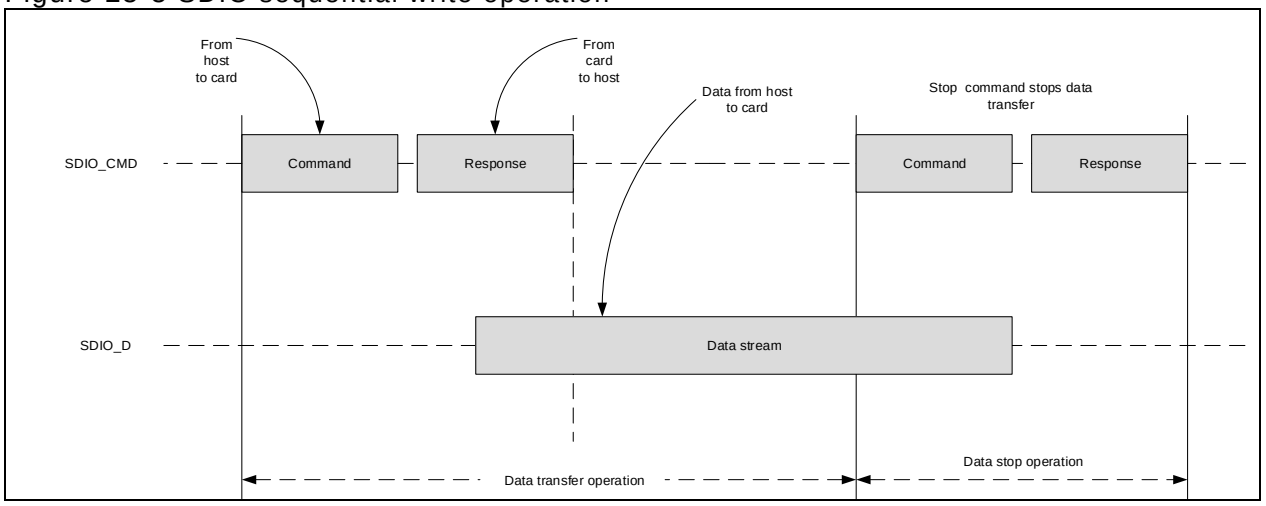


Figure 25-5 SDIO sequential write operation



25.3 Functional description

25.3.1 Card functional description

All the communications between the host and the cards are controlled by the card. The host sends two different types of commands: broadcast command and addressed (point-to-point) command.

- Broadcast command: applicable to all cards, some need responses
- Addressed command: sent to the addressed card, and responses received from the card

Memory card defines two types of operational modes:

- Card identification mode
- Data transfer mode

25.3.1.1 Card identification mode

In card identification mode, the host resets all cards, validates the operation voltage range, identifies cards and sets a relative card address (RCA) for each card on the CMD. All communications in the card identification mode use the command line (CMD).

Card identification process

The card identification process varies from card to card, and the host sends different commands. There are SD, SDI/O and MMC cards. It is possible to send a CMD5 command to identify the type of a card. If the host receives a response, it is a SDI/O card. If no response is received, the host will continue to send ACMD41 command, if a response is received, it is a SD card, otherwise a MMC card.

The card identification process is described as follows:

1. The bus is activated to confirm whether the card is connected or not. The clock frequency is at 0-400kHz during the card identification process.
2. The SDIO host sends a SD card, SDI/O card or MMC card.
3. Card Initialization
   - SD card: The SDIO host sends CMD2 (ALL_SEND_CID) to obtain its unique CID number. After receiving a response (CID number) from the card, the host will send CMD3 (SEND_RELATIVE_ADDR), instructing the card to issue the relative card address (RCA), which is shorter than the CID and is used to address the card in the data transfer mode.
   - SDI/O card: The SDIO host sends CMD3 (SEND_RELATIVE_ADDR) to instruct the card to release the relative card address (RCA), which is shorter than the CID and is used to address the card in the data transfer mode.
   - MMC card: The SDIO host sends CMD1 (SEND_OP_COND), followed by CMD2 and CMD3.
4. If the host wants to assign another RCA number, it can instruct the card to issue a new number by sending another CMD3 command. The last RCA is the actual RCA number of the card. The host repeats the card identification process (CMD2 and CMD3 cycle of each card)

25.3.1.2 Data transfer mode

The host will enter data transfer mode after identifying all cards on the bus. In data transfer mode, the host can operate cards within the range of 0 - 50MHz. It can send CMD9 (SEND_CSD) to get data specific to a card (CSD register) such as block length and car memory size. All communications between the host and the selected card are point-to-point transferred, and the CMD bus will confirm all addressed commands as a response. Data transfer read/write can be done in data block mode or stream mode, configured by the TFRMODE bit in the SDIO_DTCTRL register. In the data stream mode, data is transferred in bytes and without CRC appended to the end of each data block.

Wide bus selection/deselection

Wide bus (4-bit bus width) operation mode is selected or deselected by using ACMD6 (SET_BUS_WIDTH). The default bus width after power-up or CMD0 (GO_IDLE_STATE) is 1 bit. The ACMD6 is only valid in a transfer state, indicating that the bus width can be changed only after a card is selected by CMD7.

Stream read/write (MultiMedia card only)

Read**:**

1. The host sends CMD11 (READ_DAT_UNTIL_STOP) for stream read.
2. Until the host sends CMD12 (STOP_TRANSMISSION). The stop command has an execution delay due to the serial command transmission and the data transfers stops after the end bit of the stop command.

Write:

1. The host sends CMD20 (WRITE_DAT_UNTIL_STOP) for stream write.
2. Until the host sends CMD12 (STOP_TRANSMISSION). As the amount of data to be transferred is not determined in advance, the CRC cannot be used. When the memory range is reached, the command will be discarded by the card and remain in a transfer state, and a response is issued by setting the ADDRESS_OUT_OF_RANGE bit.

Data block read

In block read mode, the basic unit of data transfer is a block whose maximum size (fixed length 512 bytes) is defined in the CSD (READ_BL_LEN). If the READ_BL_PARTIAL is set, smaller blocks whose start and end addresses are entirely contained within 512 bytes may also be transmitted. A CRC is appended to the end of each block to ensuring data transfer integrity. Several commands related to data block read are as follows:

- CMD17 (READ_SINGLE_BLOCK): initiates a data block read and returns to the transfer state after the completion of the transfer.
- CMD18 (READ_MULTIPLE_BLOCK): starts a transfer of several consecutive data blocks.

Data blocks will keep transferring until the host sends CMD12(STOP_TRANSMISSION). The stop command has an execution delay due to the serial command transmission and the data transfers stops after the end bit of the stop command.

Data block write

During block write (CMD24-27), one or more blocks of data are transferred from the host to the card with a CRC appended to the end of each block. If the CRC failed, the card indicates a failure on the SDIO_D signal line and the transferred data are discarded and not written, and all transmitted data blocks are ignored.

If the host uses partial blocks with accumulated length is not block aligned, and block misalignment is not allowed (CSD parameter WRITE_BLK_MISALIGN is not set), the card will detect the block misalignment error before the beginning of the first misaligned block. The card sets the ADDRESS_ERROR bit in the SDIO_STS register and waits the stop command in a receive state while ignoring all further data transfer. If the host attempts to perform write operation on a write-protected area, the write operation will be aborted. In this case, however, the card should set the WP_VIOLATION bit.

Programming of the CID and CSD registers does not require a block length setting in advance. The transferred data is also CRC protected. If a part of the CSD or CID register is stored in the ROM, then the unchangeable part must match the corresponding part of the receive buffer. If this match failed, then the card will report an error and does not change any register contents. Some cards may require long and unpredictable times to write a block of data. After receiving a block of data and completing the CRC check, the card begins writing and holds the SDIO_D signal line low if its write buffer is full and unable to accept new data from a new WRITE_BLOCK command. The host can check the status of the card with SEND_STATUS command (CMD13) at any time, and the card will respond with its status.

The READY_FOR_DATA status bit indicates whether the car can accept new data or whether the write process in still in progress. The host can deselect the card by issuing CMD7 (select another card), which will place the card in the disconnect state and release the SDIO_D line without interrupting the write operation. When reselecting the card, it will reactivate busy indication by pulling SDIO_D line to low if the programming is still in progress and the write buffer is unavailable.

25.3.1.3 Erase

The erasable unit of the MultiMedia card and SD card is the erase group. The erase group is calculated in write blocks, which are the basic writable units of the card. The size of the erase group is a parameter specific to the card and defined in the CSD.

The host can erase a contiguous range of erase groups. There are three steps to start the erase process, but the commands sent by the MultiMedia card and SD card are different.

1. The host defines the start address of the range using the following command:
   - SD card: issue CMD32 (ERASE_WR_BLK_START)
   - MMC car: issue CMD35 (ERASE_GROUP_START)
2. The host defines the end address of the rang using the following command:
   - SD card: issue CMD33 (ERASE_WR_BLK_END)
   - MMD car: issue CMD36 (ERASE_GROUP_END)
3. The host starts the erase process by sending CMD38 (ERASE)

25.3.1.4 Protection management

Three write protection methods are supported in the SDIO card host module to ensure that the protected data is not erased or changed.

Mechanical write protect switch

There is a mechanical sliding switch on the side of the card to allow the user to set/clear the write protection on the card. When the sliding switch is positioned with the window open, the card is write-protected, and when the window is closed, the card is not write-protected.

Internal card write protection

Card data can be protected against write and erase. The entire card can be permanently write-protected by the manufacturer or the content provider by setting the permanent or temporary write-protect bits in the CSD. For cards that support write protection of groups of sectors by setting the WP_GRP_ENABLE

bit in the CSD, part of the data can be protected, and the write protection can be changed by the application. The SET_WRITE_PROT commands set the write protection of the addressed group. The CLR_WRITE_PROT commands clear the write protection of the addressed group. The SEND_WRITE_PROT command is similar to a single block read command. The card sends a data block containing 32 write protection bits (representing 32 write protect groups starting at the specified address) followed by 16 CRC bits. The address filed in the write protect commands is a group address in byte units.

Password protect

The password protection function enables the SDIO card host to lock and unlock a card with a password. The password is stored in the 128-bit PWD register and its size is set in the 8-bit PWD_LEN register. These registers are nonvolatile so that the content is not erased after power-off.

Locked cards can support certain commands, indicating that the host is allowed to reset, initialize and query for status, but not allowed to access data on the card. When the password is set (PWD_LEN is nonzero value), the card is locked automatically after power-up.

Like the CSD and CID register write commands, the lock/unlock commands are valid only in a transfer state. The command does not include an address parameter and thus the card must be selected before using it.

The card lock/unlock commands have the structure and bus transaction types of a regular single-block write command. The transferred data block include all the information required for the command (the password setting mode, PWD content and card lock/unlock indication). The command data block size is defined by the SDIO card host module before it sends the card lock/unlock command. The lock/unlock command structure is shown below:

Table 25-1 Lock/unlock command structure

Byte	Bit7 Bit6 Bit5 Bit4	Bit3	Bit2	Bit1	Bit0
0	Reserved(set to 0)	ERASE	LOCK_UNLOCK	CLR_PWD	SET_PWD
1	PWDS_LEN				
2	password data				
…					
PWDS_LEN+1					

- ERASE: Setting it will force an erase operation. All other bits must be zero, and only the command byte is sent.
- LOCK_UNLOCK: Setting it will lock the card. Clearing it will unlock the card. LOCK_UNLOCK can be set simultaneously with SET_PWD, but not with the CLR_PWD.
- CLR_PWD: Setting it will clear the password data.
- SET_PWD: Setting this bit will save the password data to memory.
- PWD_LEN: This bit defines the length of the password in bytes. When the password is changed, the length is the combination of the old and new passwords.
- PWD: password (new or currently used, depending on the command)

The data block size should be defined by the host before it sends the card lock/unlock command. The block length should be equal to or greater than the data structure required for the lock/unlock command.

The following sections list the command sequence to set/clear a password, lock/unlock a card, and force an erase operation.

Setting the password

1. Select a card using CMD7 (SELECT/DESELECT_CARD), if none is selected previously
2. Define the block length with CMD16(SET_BLOCKLEN) to send in the 8-bit card lock/unlock mode, 8-bit PWD_LEN, and the number of bytes of the new password. When a password is replaced, the block size must take into account the length of both the old and the new passwords sent with the command.
3. Send CMD42(LOCK/UNLOCK) with the appropriate data block size on the data line including the 16-bit CRC. The data block indicates the operation mode (SET_PWD=1), the length (PWD_LEN) and the password (PWD). When a password replacement is done, the length value includes the length of the both passwords, the old and the new one, and the PWD field includes the old password (currently used) followed by the new password.

4. When the old password is matched, the new password and its size are saved into the PWD and PWD_LEN fields, respectively. When the old password sent is not correct (in size and/or content), the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register, and the old password is not changed. When the old password sent is correct (in size and content), the given new password and its size will be saved in the PWD and the PWD_LEN registers, respectively.

The password length field (PWD_LEN) indicates whether a password is currently set or not. When the field is a zero value, it means that a password is not set currently. When the field is nonzero, it means that a password is set and the card locks itself after power-up. It is possible to lock the card immediately in the current power session by setting the LOCK_UNLOCK bit or sending an additional card lock command.

Clearing the password

1. Select a card using CMD7 (SELECT/DESELECT_CARD), if none is selected previously.
2. Define the block length with CMD16(SET_BLOCKLEN) to send in the 8-bit card lock/unlock mode, 8-bit PWD_LEN, and the number of bytes of the new password.
3. Send CMD42(LOCK/UNLOCK) with the appropriate data block size on the data line including the 16-bit CRC. The data block indicates the operation mode (SET_PWD=1), the length (PWD_LEN) and the password (PWD). When a password is matched, the PWD field is cleared and PWD_LEN is set to 0. If the password sent does not correspond to the expected password (in size and/or content), the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register, and the password is not changed.

Locking a card

1. Select a card using CMD7 (SELECT/DESELECT_CARD), if none is selected previously.
2. Define the block length with CMD16(SET_BLOCKLEN) to send in the 8-bit card lock/unlock mode, 8-bit PWD_LEN, and the number of bytes of the new password.
3. Send CMD42(LOCK/UNLOCK) with the appropriate data block size on the data line including the 16-bit CRC. The data block indicates the operation mode (SET_PWD=1), the length (PWD_LEN) and the password (PWD).
4. When a password is matched, the card is locked and CARD_IS_LOCKED is set in the SDIO_STS register. If the password sent does not correspond to the expected password (in size and/or content), the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register, and the lock fails.

If the password is previously set (PWD_LEN is not 0), the card is locked automatically after power-on reset. An attempt to lock a locked card or to lock a card that does not have a password fails and the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register.

Unlocking a card

1. Select a card using CMD7 (SELECT/DESELECT_CARD), if none is selected previously
2. Define the block length with CMD16(SET_BLOCKLEN) to send in the 8-bit card lock/unlock mode, 8-bit PWD_LEN, and the number of bytes of the new password.
3. Send CMD42(LOCK/UNLOCK) with the appropriate data block size on the data line including the 16-bit CRC. The data block indicates the operation mode (SET_PWD=1), the length (PWD_LEN) and the password (PWD).
4. When a password is matched, the card is unlocked and CARD_IS_LOCKED is cleared in the SDIO_STS register. If the password sent does not correspond to the expected password (in size and/or content), the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register, and the lock remains locked.

The unlocking function is valid only for the current power session. The card is locked automatically on the next power-up as long as the PWD field is not cleared.

An attempt to unlock an unlocked card fails and the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register.

Forcing erase

If the user forgot the password (PWD content), it is possible to access the card after clearing all the data on the card. This forced erase operation will erase all card data and all password data.

1. Select a card using CMD7 (SELECT/DESELECT_CARD), if none is selected previously
2. Define the block length with CMD16(SET_BLOCKLEN) to send in the 8-bit card lock/unlock mode, 8-bit PWD_LEN, and the number of bytes of the new password.
3. Send CMD42(LOCK/UNLOCK) with the appropriate data block size on the data line including the 16-bit CRC. The data block indicates the operation mode (ERASE =1). All other bits must be zero.
4. When the ERASE bit is the only bit in the data field, all card content will be erased, including the PWD and the PWD_LEN field, and the card is no longer locked. When any other bits are not zero, the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register, and the card data are retained, and the card remains locked.

An attempt to force erase an unlocked card fails and the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register.

25.3.2 Commands and responses

25.3.2.1 Commands

Command types

Four commands are available to control the SD memory card:

1. Broadcast command: sent to all cards, no responses returned
2. Broadcast command with response: sent to all cards, responses received from all cards simultaneously
3. Addressed command: sent to the selected card, and no data transfer on the SDIO_D line
4. Addressed data transfer command: sent to the selected card, and data transfer is present on the SDIO_D line

Command description

The SDIO host module system is designed to provide a standard interface for a variety of application types. In the meantime, specific customer/application features must also be taken into account. Because of this, two types of general commands are defined in the standard: general commands (GEN_CMD) and application-specific commands (ACMD).

To use the application-specific commands, the SDIO host must send CMD55(APP_CMD) first, and waits the response from the card, which indicates that the APP_CMD bit is set and an ACMD is expected, before sending ACMD.

Table 25-2 Commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD0	bc	[31: 0]=stuff bits	-	GO_IDLE_ STATE	Reset all cards to idle state
CMD1	bc	[31: 0]=OCR	R3	SEND_OP_ COND	In idle state, request a card to send OCR register content through the CMD bus
CMD2	bcr	[31: 0]=stuff bits	R2	ALL_Send_CID	Request all cards to send CID data through the CMD bus
CMD3	bcr	[31: 0]=stuff bits	R6	SEND_RELATIV E_AD DR	Request a card to issue a new relative card address
CMD4	bc	[31: 16]=DSR [15: 0]= stuff bits	-	SET_DSR	Set the DSR register of all cards
CMD5	bcr	[31: 24]Reserved [23: 0] I/O OCR	R4	IO_SEND_OP_C OND	Used only for the SDI/O card to query the voltage range of the required I/O card

CMD6	ac	[31: 26] set to 0 [25: 24] access [23: 16] index [15: 8] value [7: 3] set to 0 [2: 0] command set	R1b	SWITCH	Used only for the MMC card to switch the operation modes or modify the EXT_CSD register
CMD7	ac	[31: 16]=RCA [15: 0]= stuff bits	R1b	SELECT/DESELECT_ CARD	This command is used to switch a card between the standby state and the send state, or between the programmed and the disconnected state. The relative card address is used to select a card. Address 0 is used to deselect the card.
CMD8 (SD)	bcr	[31: 12] Reserved [11: 8]operating voltage (VHS) [7: 0]check mode	R7	SEND_IF_ COND	Send the host power supply voltage to the SD card and check whether the card supports the voltage or not.
CMD8 (MMC)	adtc	[31: 0]= stuff bits	R1	SEND_EXT_ CSD	Used only for the MMC card to send its own EXT_CSD register as a data block
CMD9	ac	[31: 16]=RCA [15: 0]= stuff bits	R2	SEND_CSD	The selected card sends CSD (card-specific data) through the CMD bus
CMD10	ac	[31: 16]=RCA [15: 0]= stuff bits	R2	SEND_CID	The selected card sends CID (card flag) through the CMD bus
CMD12	ac	[31: 0]= stuff bits	R1b	STOP_TRANSMISSION	Force the card to stop transmission
CMD13	ac	[31: 16]=RCA [15: 0]= stuff bits	R1	SEND_ STATUS	Selected card send status register
CMD15	ac	[31: 16]=RCA [15: 0]= stuff bits	-	GO_INACTIVE_ STATE	Selected card switch to inactive state

Table 25-3 Data block read commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD16	ac	[31: 0]=data block length	R1	SET_ BLOCKLEN	This command is used to set the length of data blocks (in bytes) for all block commands. The default value is 512 bytes.
CMD17	adtc	[31: 0]=data address	R1	READ_ SINGLE_ BLOCK	Read a data block of the size set by CMD16
CMD18	adtc	[31: 0]=data address	R1	READ_ MULTIPLE_ BLOCK	Continuously read data from the card to the host until the STOP_TRANSMISSION is received

Table 25-4 Data stream read/write commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD11	adtc	[31: 0]= data address	R1	READ_DAT_UNTIL_STOP	Read data stream form the card starting from a given address until the STOP_TRANSMISSION is received.
CMD20	adtc	[31: 0]= data address	R1	WRITE_DAT_UNTIL_STOP	Read data stream form the host starting from a given address until the STOP_TRANSMISSION is received.

Table 25-5 Data block write commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD16	ac	[31: 0]=data block length	R1	SET_BLOCKLEN	This command is used to set the length of data blocks (in bytes) for all block commands. The default value is 512 bytes
CMD23	ac	[31: 16]=set to 0 [15: 0]=data block size	R1	SET_BLOCK_COUNT	Define the number of blocks to be transferred in the data block read/write that follows
CMD24	adtc	[31: 0]=data address	R1	WRITE_BLOCK	Write a data block of the size set by the CMD16
CMD25	adtc	[31: 0]=data address	R1	WRITE_MULTIPLE BLOCK	Continuously write data blocks until the STOP_TRANSMISSION is received
CMD26	adtc	[31: 0]= stuff bits	R1	PROGRAM_CID	Program the card identification register
CMD27	adtc	[31: 0]= stuff bits	R1	PROGRAM_CSD	Program the programmable bits of the CSD

Table 25-6 Block-based write protect commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD28	ac	[31: 0]= data address	R1b	SET_WRITE_PROT	If the card has write protection features, this command sets the write protection bit of the specified group. The properties of write protection are placed in the card-specific area (WP_GRP_SIZE).
CMD29	ac	[31: 0]= data address	R1b	CLR_WRITE_PROT	If the card has write protection features, this command clears the write protection bit of the specified group.
CMD30	adtc	[31: 0]=write protect data address	R1	SEND_WRITE_PROT	If the card has write protection features, this command asks the card to send the status of the write protection bits.

Table 25-7 Erase commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD32 … CMD34	Reserved. These command indexes cannot be used in order to maintain backward compatibility with older versions of the MultiMedia card.				
CMD35	ac	[31: 0]=data address	R1	ERASE_GROUP_START	Sets the address of the first erase group within a range to be selected for erase
CMD36	ac	[31: 0]=data address	R1	ERASE_GROUP_END	Sets the address of the last erase group within a continuous range to be selected for erase
CMD37	Reserved. These command indexes cannot be used in order to maintain backward compatibility with older versions of the MultiMedia card.				
CMD38	ac	[31: 0]=stuff bits	R1b	ERASE	Erase all previously selected data blocks

Table 25-8 I/O mode commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD39	ac	[31: 16]=RCA [15]=register write flag [14: 8]=register address [7: 0]=register data	R4	FAST_IO	Used to write and read 8-bit (register) data fields. The command specifies a card and a register and provides the data for writing if the write flag is set. The R4 response contains data read from the specified register. This command accesses application-specific registers that are not defined in the MultiMedia card standard.
CMD40	bcr	[31: 0]=stuff bits	R5	GO_IRQ_STATE	Place the system in the interrupt mode

Table 25-9 Card lock commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD42	adtc	[31: 0]=stuff bits	R1	LOCK_ UNLOCK	Sets/clears the password or locks/unlocks the card. The size of the data block is set by CMD16.

Table 25-10 Application-specific commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD55	ac	[31: 16]=RCA [15: 0]=stuff bits	R1	APP_CMD	Indicates to the card that the next command is an application-specific command rather than a standard command.
CMD56	adtc	[31: 1]=stuff bits [0]=RD/WR	R1	GEN_CMD	Used either to transfer a data block to the card or to read a data block from the card for general-purpose/application-specific commands. The size of the data block is defined by the SET_BLOCK_LEN command.
CMD57 … CMD59	Reserved.				
CMD60 … CMD63	Reserved for manufacturer				

25.3.2.2 Response formats

All responses are sent via the CMD bus. The response transmission always starts with the left bit of the bit string corresponding to the response code word. The length depends on the response type.

A response always starts with a start bit (always 0), followed by the transmission bit indicating the direction of transmission (card =0). A value denoted by – in the tables below stands for a variable entry. All responses, except the R3 response type, are protected by a CRC. Every command code word is terminated with the end bit (always 1).

25.3.2.2.1 R1 (normal response command)

Code length = 48 bits. The 45:40 bits indicate the index of the command to be responded to. This value is interpreted as a binary-coded number (between 0 and 63). The status of the card is coded in 32 bits. Note that if it involves a data transfer to a card, a busy signal may appear on the data line after each data block is transmitted. The host should check the busy signal after data block transfer.

Table 25-11 R1 response

Bit	47	46	[45: 40]	[39: 8]	[7: 1]	0
Field width	1	1	6	32	7	1
Value	0	0	-	-	-	1
Description	Start bit	Transmission bit	Command index	Card status	CRC7	End bit

25.3.2.2.2 R1b

It is the same as R1 with an optional busy signal transmitted on the data line. The card may become busy after receiving these commands based on its state prior to the command reception. The host should check the busy signal.

25.3.2.2.3 R2 (CID & CSD registers)

Code length = 136 bits. The contents of the CID register are sent as a response to the CMD2 and CMD10 commands. The contents of the CSD register are sent as a response to the CMD9. Only the bits [127 ... 1] of the CID and CSD are transmitted, and the reserved bit [0] of these registers is replaced with the end bit of the response.

Table 25-12 R2 response

Bit	135	134	[133 : 128]	[127 : 1]	0
Field width	1	1	6	127	1
Value	1	0	111111	-	1
Description	Start bit	Transmission bit	Reserved	CID or CSD register	End bit

25.3.2.2.4 R3 (OCR register)

Code length = 48 bits. The contents of the OCR register are sent as a response to ACMD41.

Table 25-13 R3 response

Bit	**47**	**46**	**[45 : 40]**	**[39: 8]**	**[7: 1]**	**0**
Field width	1	1	6	32	7	1
Value	1	0	111111	-	111111	1
Description	Start bit	Transmission bit	Reserved	OCR register	Reserved	End bit

25.3.2.2.5 R4 (Fast I/O)

Code length = 48 bits. The parameter field contains the RCA of the specified card, the register address to be read out or written to, and its contents.

Table 25-14 R4 response

Bit	**47**	**46**	**[45: 40]**	**[39: 8]**			**[7: 1]**	**0**
Field width	1	1	6	16	8	8	7	1
Value	1	0	100111	-	-	-	-	1
Description	Start bit	Transmission bit	CMD39	RCA	Register address	Read register contents	CRC7	End bit

25.3.2.2.6 R4b

For SD I/O only, an SDIO card will respond with a unique SDIO response R4 after receiving the CMD5.

Table 25-15 R4b response

Bit	**47**	**46**	**[45: 40]**			**[39: 8]**			**[7: 1]**	**0**
Field width	1	1	6	1	3	1	3	24	7	1
Value	1	0	-	-	-	-	-	-	-	1
Description	Start bit	Tx bit	Res.	Card ready	Number of I/O functions	Current memory	Stuff bit	I/O OCR	Res.	End bit

25.3.2.2.7 R5 (interrupt request)

For MultiMedia card only. Code length = 48 bits. If the response is generated by the host, the RCA field in the parameter will be 0x0.

Table 25-16 R5 response

Bit	**47**	**46**	**[45: 40]**	**[39: 8]**		**[7: 1]**	**0**
Field width	1	1	6	16	16	7	1
Value	1	0	101000	-	-	-	1
Description	Start bit	Start bit	Tx bit	RCA[31:16] of a successful card or of the host	Not defined. Maybe used for interrupt data	CRC7	End bit

25.3.2.2.8 R6 (interrupt request)

For SD I/O card only. This is a normal response to CMD3 by a memory device.

Table 25-17 R6 response

Bit	47	46	[45: 40]		[39: 8]	[7: 1]	0
Field width	1	1	6	16	16	7	1
Value	1	0	000011	-	-	-	1
Description	Start bit	Tx bit	CMD3	RCA[31:16] of a successful card or of the host	Card status	CRC7	End bit

The card status bit [23: 8] will be changed when the CMD3 is sent to an I/O-only card. In this case, the 16 bits of response are the SD I/O-only values.

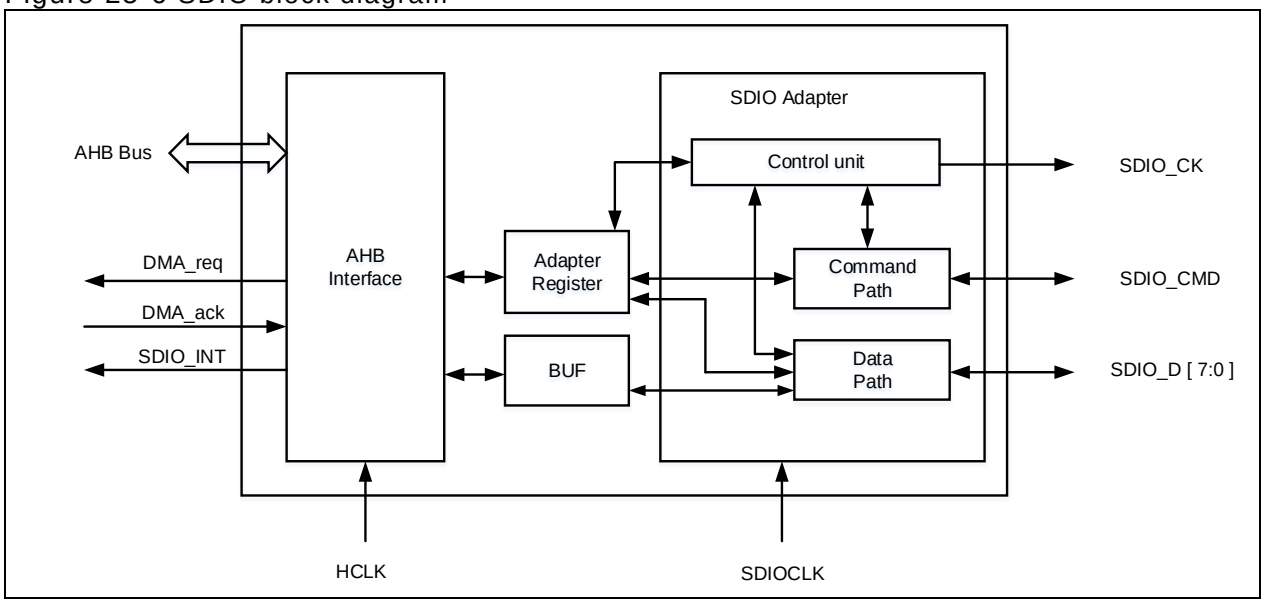
- Bit 15=COM_CRC_ERROR
- Bit 14=ILLEGAL_COMMAND
- Bit 13=ERROR
- Bit [12: 0]=Reserved

25.3.3 SDIO functional description

SDIO consists of four parts:

- SDIO adapter block: contains a control unit, command path and data path that provides all functions specific to the MMC/SD/SD I/O card such as the clock generation, command and data transfer
  - Control unit: manages and generates clock signals
  - Command path: manages command transfer
  - Data path: manages data transfer
- AHB interface: generates interrupt and DMA request signals
- Adapter register: system register
- BUF: used for data transfer

Figure 25-6 SDIO block diagram



25.3.3.1 SDIO adapter

SDIO_CK is a clock to the MultiMedia/SD/SDIO car provided by the host. One bit of command or data is transferred on both command and data lines with each clock cycle. The clock frequency can very between different cards and different protocols.

- MultiMedia card
  - V3.31 protocol 0 – 20MHz
  - V4.0/4.2 protocol 0 – 50MHz
- SD card
  - 0 – 50MHz
- SD I/O card
  - 0 – 50MHz

SDIO_CMD is a bidirectional command channel and used for the initialization of a card and command transfer. When the host sends a command to a card, the card will issue a response to the host. The SDIO_CMD has two operational modes:

- Open-drain mode for initialization (only for MMCV3.31 or previous)
- Push-pull mode for command transfer (SD/SD I/O card and MMC V4.2 also use push-pull drivers for initialization)

SDIO_D [7:0] is a bidirectional data channel. After initialization, the host can change the width of the data bus. After reset, the SDIO_D0 is used for data transfer by default. MMCV3.31 or previous supports only one bit of data line, so only SDIO_DO can be used.

The table below is used for the MultiMedia card/SD/SD I/O card bus:

Table 25-18 SDIO pin definitions

Pin	Direction	Description
SDIO_CK	Output	MultiMedia card/SD/SDIO card clock. This pin is the clock from the host to a card.
SDIO_CMD	Bidirectional	MultiMedia card/SD/SDIO card command. This pin is the bidirectional command/response signal.
SDIO_D[7: 0]	Bidirectional	MultiMedia card/SD/SDIO card data. This pin is the bidirectional databus.

Control unit

The control unit consists of a power management sub-unit and a clock management sub-unit. The power management subunit is controlled by the SDIO_PWRCTRL register. The PS bit is used to define power-up/power-off state. During the power-off and power-up phases, the power management subunit will disable the card bus output signals. The clock management subunit is controlled by the SDIO_CLKCTRL

register where the CLKDIV bit is used to define the divider factor between the SDIOCLK and the SDIO output clock. If BYPSEN = 0, the SDIO_CK output signal is driven by the SDIOCLK divided according to the CLKDIV bit; if BYPSEN = 1, the SDIO_CK output signal is directly driven by the SDIOCLK. The HFCEN is set to enable hardware flow control feature in order to avoid the occurrence of an error at transmission underflow or reception overflow. The PWRSVEN bit can be set by software to enable power save mode, and the SDIO_CK can be output only when the bus is active.

Command path

The command path unit sends commands to and receives responses from the cards. When the CCSMEN bit is set in the SDIO_CMDCTRL register, a command transfer starts. First sends a command to a card by the SDIO_CMD, the command length is 48 bits. The data on the SDIO_CMD is synchronized with the rising edge of the SDIO_CK. A block of data is transferred with each SDIO_CK, including start bit, transfer bit, command index defined by the SDIO_CMDCTRL_CMDIDX bit, parameters defined by the SDIO_ARG, 7-bit CRC and end bit. Then receives responses from the card. There are two response types: 48-bit short response and 136-bit long response. Both use CRC error check. The received responses are saved in the area from SDIO_RSP1 to SDIO_RSP4. The command path can generate command flag, which can be defined by the SDIO_STS register.

Table 25-19 Command formats

Bit	47	46	[45: 40]	[39: 8]	[7: 1]	0
Width	1	1	6	32	7	1
Value	0	1	-	-	-	1
Description	Start bit	Tx bit	Command index	Parameter	CRC7	End bit

— Response: A response is sent from a specified card to the host (or synchronously from all cards for MMCV3.31 or previous), as an answer to a previously received command. Responses are transferred serially on the CMD line.

Table 25-20 Short response format

Bit	47	46	[45: 40]	[39: 8]	[7: 1]	0
Width	1	1	6	32	7	1
Value	0	0	-	-	-	1
Description	Start bit	Tx bit	Command index	Parameter	CRC7 (or 1111111)	End bit

Table 25-21 Long response format

Bit	135	134	[133: 128]	[127: 1]	0
Width	1	1	6	127	1
Value	0	0	111111	-	1
Description	Start bit	Tx	Reserved	CID or CSD (including internal CRC7)	End bit

Table 25-22 Command path status flags

Flag	Description
CMDRSPCMPL	A response is already received (CRC OK)
CMDFAIL	A command response is already received (CRC fails)
CMDCMPL	A command is sent (does not require a response)
CMDTIMEOUT	Command response timeout (64 SDIO_CK cycles)
DOCMD	Command transfer is in progress

Command channel state machine (CCSM)

When the CCSMEN bit is set in the SDIO_CMDCTR register, command transfer starts. When the command has been sent, the command channel state machine (CCSM) will set the status flags and enters the idle state if a response is not required. When a response is received, the received CRC code and the internally generated CRC code are compared, and the appropriate status flags are set.

- The CCSM remains in the idle state for at least 8 SDIO_CK cycles to meet the Ncc (the minimum delay between two host commands) and NRC (the minimum delay between the host command and the card response).
- When the wait state is entered, the command timer is enabled. If the NCR timeout (response time to a command), that is, 64 SDIO_CK periods, is reached before the CCSM moves to the receive state, the timeout flag is set (CMDTIMEOUT) and the idle state is entered.

If the interrupt bit is set in the command register, the timer is disabled and the CCSM waits for an interrupt request from one card. If the pending bit is set in the command register, the CCSM will enter pend state and wait for a CmdPend signal from the data path subunit. When detecting the CmdPend, the CCSM goes to the send state, which triggers the data counter to send the stop command.

Figure 25-7 Command channel state machine (CCSM)

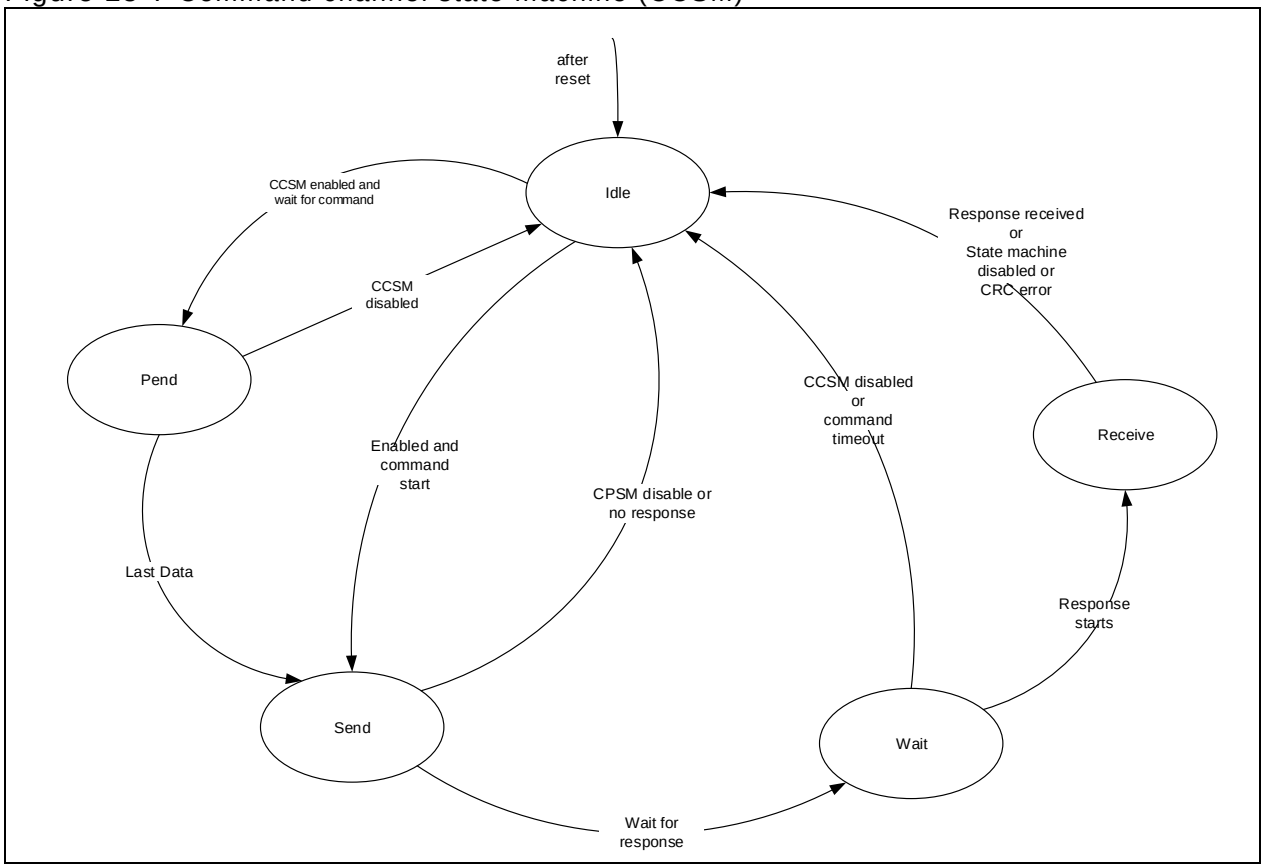
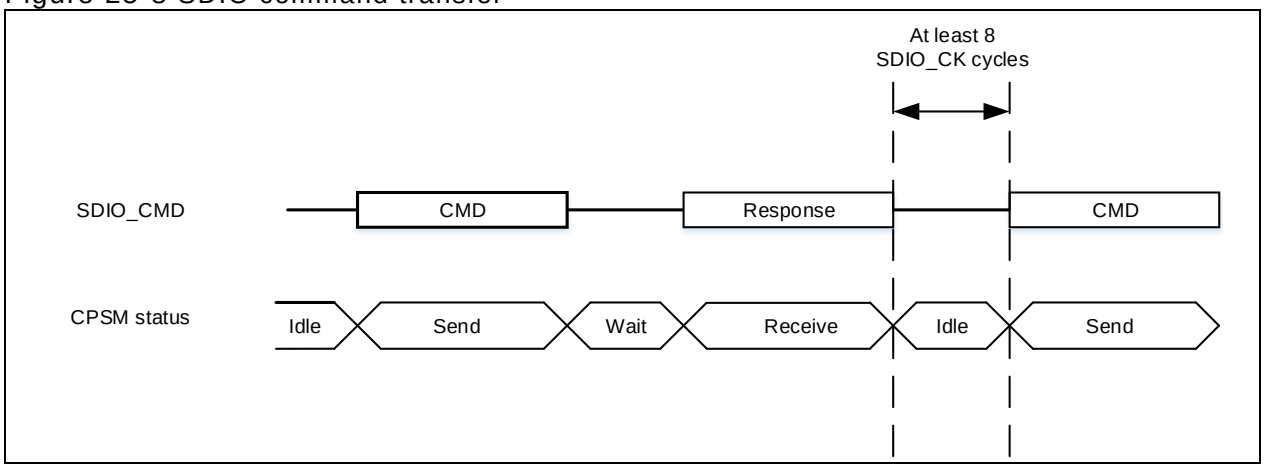


Figure 25-8 SDIO command transfer



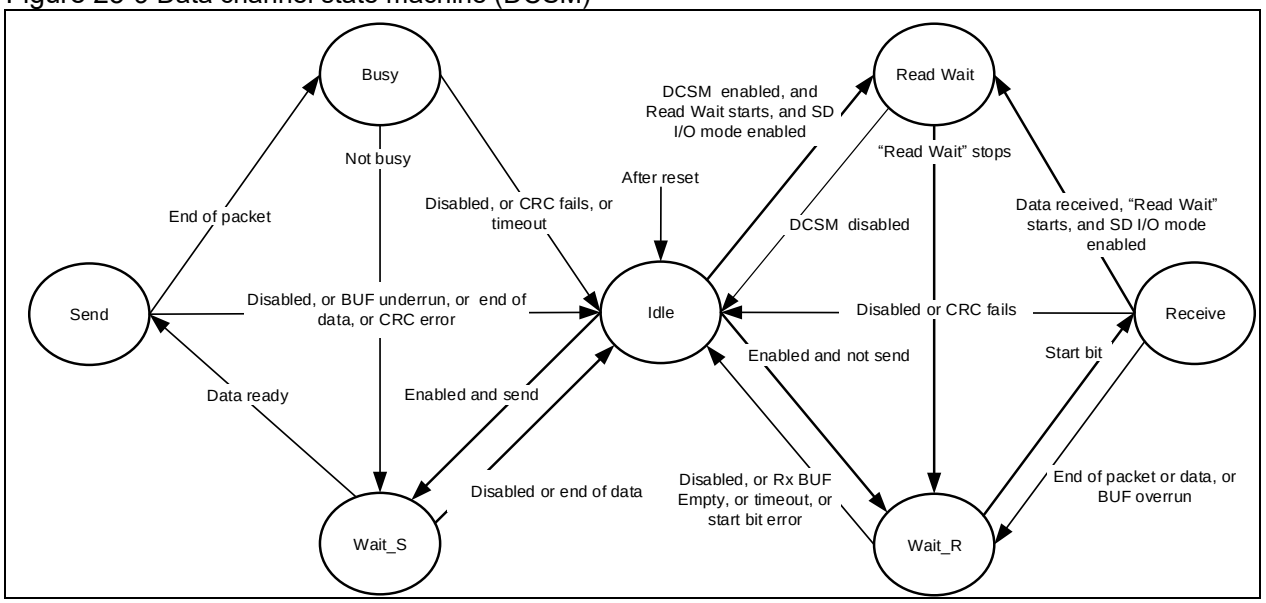
Data channel

The data path subunit transfers data between the host and the cards. The databus width can be configured using the BUSWS bit in the SDIO_CLKCTRL register. By default, only the SDIO_DO signal line is used for transfer. Only one bit of data is transferred with each clock cycle. If the 4-bit wide bus mode is selected, four bits are transferred per clock cycle over the SDIO_D [3:0] signal line. If the 8-bit wide bus mode is selected, eight bits are transferred per clock cycle over the SDIO_D [7: 0] signal line. The TFRDIR bit is set in the SDIO_DTCTR register to define the transfer direction. If TFRDIR=0, it indicates that the data is transferred from the controller to the card; if TFRDIR=1, it indicates that the data is transferred from the card to the controller. The TFRMODE bit can be used to select block data transfer or stream transfer for the MultiMedia card. If the TFREN bit is set, data transfer starts. Depending on the TFRDIR bit, the DCSM enters Wait_S or Wait_R state.

Data channel state machine (DCSM)

The DCSM has seven states, in send and receive mode, as shown in the Figure below:

Figure 25-9 Data channel state machine (DCSM)



Send mode

- Idle: The data channel is inactive, either in the Wait_S or the Wait_R state.
- Wait_S: Waits until the BUF flag becomes empty or the data transmission is completed. The DCSM must remain in the Wait_S state for at least two clock periods to meet the N_{WR} timing requirements where the N_{WR} is the interval between the reception of the card response and the start of the data transfer from the host.
- Send: The DCSM sends data to a card, and the data transfer mode can be either block or stream, depending on the SDIO_DTCTRL_TFRMODE bit. If an overflow error occurred, the DCSM then moves to the idle state

- Busy: The DCSM waits for the CRC flag. If the DCSM receives a correct CRC status and is not busy, it will enter the Wait_S state. If it does not receive a correct CRC status or a timeout occurs while the DCSM is in the busy state, a CRC fail flag or timeout flag is generated.
- Wait_R: The start bit of the Wait_R. If a timeout occurs before it detects a start bit, the DCSM moves to the idle state and generates a timeout flag.
- Receive: Data is received from a card and written to the BUF. The data transfer mode can be either block or stream, depending on the SDIO_DTCTRL _TFRMODE bit. If an overflow error occurs, it then returns to Wait_R state.

Table 25-23 Data token formats

Description	Start bit	Data	CRC16	End bit
Block data	0	-	Y	1
Stream data	0	-	N	1

25.3.3.2 Data BUF

The data BUF contains a transmit and receive unit. It is a 32-bit wide and 32-word deep data buffer. Because the data BUF operates in the AHB clock domain (HCLK), all signals connected to the SDIO clock domain (SDIOCLK) are resynchronized.

- Transmit BUF: Data can be written to the transmit BUF via the AHB interface when the SDIO transmission feature is enabled.

The transmit BUF has 32 sequential addresses. It contains a data output register that holds the data word pointed by the read pointer. When the data path has loaded its shift register, it moves its read pointer to the next data and outputs data.
If the transmit BUF is disabled, all status flags are inactive. The data path sets the DOTX when it transmits data.

- Receive BUF: When the data path receives a data word, it will write the data to the BUF. The write pointer is incremented automatically after the end of the write operation. On the other side, a read pointer always points to the current data in the BUF. If the receive BUF is disabled, all status flags are cleared, and the read and write pointers are reset as well. The data path sets the DORX when it receives data.

25.3.3.3 SDIO AHB interface

The AHB interface generates the interrupt and DMA requests, and access the SDIO interface registers and the data BUF.

SDIO interrupts

The interrupt logic generates an interrupt request when one of the selected status flags is high. The SDIO_INTEN register is used to select the conditions that will generate an interrupt.

SDIO/DMA interface: data transfer process between the SDIO and memory

In the following examples, data is transferred from the host to the card. The SDIO BUF is filled with data stored in a memory through the DMA controller.

1. Card identification process
2. Increase the SDIO_CK frequency
3. Select a card by sending CMD7
4. Enable the DMA2 controller and clear all interrupt flag bits, configure the DMA2 channel4 source address register as the memory buffer's base address, and the DMA2 channel4 destination address register as the SDIO_BUF register address. Then configure the DMA2 channel4 control register (memory increment, non-peripheral increment, and peripheral and source data width is word width). Finally enable DMA2 channel4.
5. Send CMD24 (WRITE_BLOCK) as follows:

   Program the SDIO data length register (SDIO_DTLEN), the BLKSIZE bit in the SDIO data control register (SDIO_DTCTRL), and the SDIO parameter register (SDIO_ARG) with the address of the card where data is to be transferred, and program the SDIO command register (SDIO_CMD), enable the CCSMEN bit, wait for SDIO_STS [6]=CMDRSPCMPL interrupt,

and then program the SDIO data control register (SDIO_DTCTRL): TFREN=1 (enable the SDIO card host to send data), TFRDIR=0 (from the controller to the card), TFRMODE=0 (block data transfer), DMAEN=1 (enable DMA), BLKSIZE=9 (512 bytes), and wait from SDIO_STS [10]=DTBLKCMPL.

6. Check that no channels are still enabled by confirming the DMA channel enable SDIO status register (SDIO_STS)

25.3.3.4 Hardware flow control

The HFCEN bit is set in the SDIO_CLKCTRL register to enable hardware flow control, which is sued to avoid BUF underflow and overflow errors. Read/write access to the BUF is still active even if flow control is enabled.

25.3.4 SDIO I/O card-specific operations

The SDIO can support the following operations (except read suspend, for it does not require specific hardware operation) when the SDIO_DTCTRL [11] is set.

SDIO read wait operation by SDIO_D2 signal lines

The optional read wait operation is used only for SD card 1-bit or 4-bit mode. The read wait operation can instruct the host to stop data transfer temporarily while the host is reading from multiple registers (IO_RW_EXTENDED, CMD53), and also allows the host to send commands to other functions in the SD I/O device in order to start a read wait process after the reception of the first data block. The detailed process as follows:

- Enable data path (SDIO_DTCTRL [0] = 1)
- Enable SDIO-specific operation (SDIO_DTCTRL [11] = 1)
- Start read wait (SDIO_DTCTRL [10]=0 and SDIO_DTCTRL [8]=1)
- Data direction is from a card to the SDIO host (SDIO_DTCTRL [1]=1)
- The data unit in the SDIO adapter will enter read wait state, and drive the SDIO_D2 to 0 after 2 SDIO_CK cycles
- The data unit starts waiting to receive data from a card. The DCSM will not enter read wait even if read wait start is set. The read wait process will start after the CRC is received. The RDWTSTOP has to be cleared to start a new read wait operation.

During the read wait period, the SDIO host can detect the SDIO interrupts over the SDIO_D1.

SDIO read wait operation by stopping clock

If the SDIO card does not support the mentioned above read wait operation, the SDIO can enter a read wait by stopping SDIO_CK, described as follows:

- Enable data path (SDIO_DTCTRL [0] = 1)
- Enable SDIO-specific operation (SDIO_DTCTRL [11] = 1)
- Start read wait (SDIO_DTCTRL [10]=0 and SDIO_DTCTRL [8]=1)

The DCSM stops the clock two SDIO_CK cycles after the end bit of the current received data block and starts the clock again after the read wait end bit is set.

Note that as the SDIO_CK is stopped, the SDIO host cannot send any command to the card. The SDIO host can detect the SDIO interrupt over the SDIO_D1.

SDIO suspend/resume operation (write and read operation suspend)

To free the bus to provide higher-priority transfers for other functions or memories, the host can suspend data transfer to certain functions or memories. As soon as the higher-priority transfer is completed, the previous transfer operation will restart at the suspended location.

While sending data to a card, the SDIO can suspend the write operation. The SDIO_CMD [11] bit is set and indicates to the CCSM that the current command is a suspend command. The CCSM analyzes the response and when an ACK signal is received from the card (suspend accepted), it acknowledges the DCSM that enters the idle state after receiving the CRC of the current data block.

SDIO interrupts

There is a pin with interrupt feature on the SD interface in order to enable the SD I/O card to interrupt the MultiMedia card/SD module. In 4-bit SD mode, this pin is SDIO_D1. The SD I/O interrupts are detected when the level is active. In other words, the interrupt signal line must be active (low) before it is recognized and responded by the MultiMedia card/SD module, and will remain inactive (high) at the end of the interrupt routine.

When the SDIO_DTCTRL [11] bit is set, the SDIO interrupts are detected on the SDIO_D1 signal line.

25.4 SDIO registers

The device communicates with the system through 32-bit control registers accessible via AHB.

The peripheral registers must be accessed by words (32-bit).

Table 25-24 A summary of the SDIO registers

Register name	Offset	Reset value
SDIO_PWRCTRL	0x00	0x0000 0000
SDIO_CLKCTRL	0x04	0x0000 0000
SDIO_ARG	0x08	0x0000 0000
SDIO_CMD	0x0C	0x0000 0000
SDIO_RSPCMD	0x10	0x0000 0000
SDIO_RSP1	0x14	0x0000 0000
SDIO_RSP2	0x18	0x0000 0000
SDIO_RSP3	0x1C	0x0000 0000
SDIO_RSP4	0x20	0x0000 0000
SDIO_DTTMR	0x24	0x0000 0000
SDIO_DTLEN	0x28	0x0000 0000
SDIO_DTCTRL	0x2C	0x0000 0000
SDIO_DTCNTR	0x30	0x0000 0000
SDIO_STS	0x34	0x0000 0000
SDIO_INTCLR	0x38	0x0000 0000
SDIO_INTEN	0x3C	0x0000 0000
SDIO_BUFCNTR	0x48	0x0000 0000
SDIO_BUF	0x80	0x0000 0000

25.4.1 SDIO power control register (SDIO_ PWRCTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 2	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 1: 0	PS	0x0	rw	Power switch These bits are set or cleared by software. They are used to define the current status of the card clock. 00: Power-off, the card clock is stopped. 01: Reserved 10: Reserved 11: Power-on, the card clock is started.

Note: Write access to this register is not allowed within seven HCLK clock periods after data is written.

25.4.2 SDIO clock control register (SDIO_ CLKCTRL)

The SDIO_CLKCTRL register controls the SDIO_CK output clock.

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16: 15	CLKDIV	0x0	rw	Clock division This field is set or cleared by software. It defines the clock division relations between the SDIOCLK and the SDIO_CK: SDIO_CK frequency=SDIOCLK / [CLKDIV[9: 0] + 2].
Bit 14	HFCEN	0x0	rw	Hardware flow control enable This bit is set or cleared by software. 0: Hardware flow control disabled 1: Hardware flow control enabled Note: When hardware flow control is enabled, refer to the SDIO_STS register for the meaning of the TXBUF_E and RXBUF_F interrupt signals.
Bit 13	CLKEGS	0x0	rw	SDIO_CK edge selection This bit is set or cleared by software. 0: SDIO_CK generated on the rising edge of the master clock SDIOCLK 1: SDIO_CK generated on the falling edge of the master clock SDIOCLK
Bit 12: 11	BUSWS	0x0	rw	Bus width selection This bit is set or cleared by software. 00: Default bus mode, SDIO_D0 used 01: 4-bit bus mode,SDIO_D[3: 0] used 10: 8-bit bus mode, SDIO_D[7: 0] used
Bit 10	BYPSEN	0x0	rw	Clock divider bypass enable bit This bit is set or cleared by software. When disabled, the SDIO_CK output signal is driven by the SDIOCLK that is divided according to the CLKDIV value. When enabled, the SDIO_CK output signal is directly driven by the SDIOCLK. 0: Clock divider bypass disabled 1: Clock divider bypass enabled
Bit 9	PWRSVEN	0x0	rw	Power saving mode enable This bit is set or cleared by software. When disabled, the SDIO_CK is always output; when enabled, the SDIO_CK is only output when the bus is active. 0: Power saving mode disabled 1: Power saving mode enabled
Bit 8	CLKOEN	0x0	rw	Clock output enable This bit is set or cleared by software. 0: Clock output disabled 1: Clock output enabled
Bit 7: 0	CLKDIV	0x00	rw	Clock division This field is set or cleared by software. It defines the clock division relations between the SDIOCLK and the SDIO_CK: SDIO_CK frequency=SDIOCLK / [CLKDIV[9: 0] + 2].

Note: 1. While the SD/SDIO card or MultiMedia car is in identification mode, the SDIO_CK frequency must be less than 400kHz.

2. When all card are assigned with relative card addresses, the clock frequency can be changed to the maximum card frequency.

3. This register cannot be written within seven HCLK clock periods after data is written. The SDIO_CK can be stopped during the read wait period for SD I/O cards. In this case, the SDIO_ CLKCTRL register does not control the SDIO_CK.

25.4.3 SDIO argument register (SDIO_ARG)

The SDIO_ARG register contains 32-bit command argument, which is sent to a card as part of a command.

Bit	Register	Reset value	Type	Description
Bit 31: 0	ARGU	0x0000 0000	rw	Command argument Command argument is sent to a card as part of a command. If a command contains an argument, it must be loaded into this register before writing a command to the command register.

25.4.4 SDIO command register (SDIO_CMD)

The SDIO_CMD register contains the command index and command type bits. The command index is sent to a card as part of a command. The command type bits control the command channel state machine (CCSM).

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value.
Bit 11	IOSUSP	0x0	rw	SD I/O suspend command This bit is set or cleared by software. If this bit is set, the command to be sent is a suspend command (used for SDIO only). 0: SD I/O suspend command disabled 1: SD I/O suspend command enabled
Bit 10	CCSMEN	0x0	rw	Command channel state machine (CCSM) enable bit This bit is set or cleared by software. 0: Command channel state machine disabled 1: Command channel state machine enabled
Bit 9	PNDWT	0x0	rw	CCSM Waits for ends of data transfer (CmdPend internal signal) This bit is set or cleared by software. If this bit is set, the CCSM waits for the end of data transfer before it starts sending a command. 0: Disabled 1: Enabled
Bit 8	INTWT	0x0	rw	CCSM waits for interrupt request This bit is set or cleared by software. If this bit is set, the CCSM disables command timeout and waits for an interrupt request. 0: Disabled 1: Enabled
Bit 7: 6	RSPWT	0x0	rw	Wait for response bits This bit is set or cleared by software. This bit indicates whether the CCSM is to wait for a response, and if yes, it will indicates the response type. 00: No response 01: Short response 10: No response 11: Long response
Bit 5: 0	CMDIDX	0x00	rw	Command index The command index is sent to a card as part of a command.

Note: 1.This register cannot be written within seven HCLK clock periods after data is written.
2.MultiMedia card can sent two types of responses: 48-bit short response or 136-bit short response. The SD card and SD I/O card can send only short responses, and the argument can vary according to the type of response. The software will distinguish the type of response according to the command sent.

25.4.5 SDIO command response register (SDIO_RSPCMD)

The SDIO_RSPCMD register contains the command index of the last command response received. If the command response transmission does not contain the command index (long or OCR response), the SDIO_RSPCMD field is unknown, although it should have contained 111111b (the value of the reserved field from a response)

Bit	Register	Reset value	Type	Description
Bit 31: 6	Reserved	0x0000000	resd	Kept at its default value.
Bit 5: 0	RSPCMD	0x00	ro	Response command index This field contains the command index of the command response received.

25.4.6 SDIO response 1..4 register (SDIO_RSPx)

The SDIO_RSPx (x=1..4) register contains the status of a card, which is part of the response received.

Bit	Register	Reset value	Type	Description
Bit 31: 0	CARDSTSx	0x0000 0000	ro	See Table 23-25

The card status size is 32 or 127 bits, depending on the response type.

Table 25-25 Response type and SDIO_RSPx register

Register	Short response	Long response
SDIO_RSP1	Card status [31: 0]	Card status [127: 96]
SDIO_RSP2	Unused	Card status [95: 64]
SDIO_RSP3	Unused	Card status [63: 32]
SDIO_RSP4	Unused	Card status [31: 1]

The most significant bit of the card status is always received first. The least significant bit of the SDIO_RSP4 register is always 0.

25.4.7 SDIO data timer register (SDIO_DTTMR)

The SDIO_DTTMR register contains the data timeout period in the unit of card bus clock periods. A counter loads the value from the SDIO_DTTMR register and starts decrementing when the DCSM enters the Wait_R or busy state. If the counter reaches 0 while the DCSM is in either of these states, a timeout status flag will be set.

Bit	Register	Reset value	Type	Description
Bit 31: 0	TIMEOUT	0x0000 0000	rw	Data timeout period Data timeout period in card bus clock cycles.

Note: A data transfer must be written to the SDIO_DTCNTR and the SDIO_DTLEN register before being written to the SDIO data control register (SDIO_DTCTRL).

25.4.8 SDIO data length register (SDIO_DTLEN)

The SDIO_DTLEN register contains the number of data bytes to be transferred. The value is loaded into the data counter when data transfer starts.

Bit	Register	Reset value	Type	Description
Bit 31: 25	Reserved	0x00	resd	Kept at its default value.
Bit 24: 0	DTLEN	0x0000000	rw	Data length value Number of data bytes to be transferred.

Note: For a block data transfer, the value in the SDIO_DTLEN must be a multiple of the block data size.

A data transfer must be written to the SDIO_DTCNTR and the SDIO_DTLEN register before being written to the SDIO data control register (SDIO_DTCTRL).

25.4.9 SDIO data control register (SDIO_DTCTRL)

The SDIO_DTCTRL register controls the data channel statue machine (DCSM).

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value.
Bit 11	IOEN	0x0	rw	SD I/O enable functions This bit is set or cleared by software. If the bit is set, the DCSM performs an SD IO card-specific operation. 0: Disabled 1: Enabled
Bit 10	RDWTMODE	0x0	rw	Read wait mode This bit is set or cleared by software. If disabled, the SDIO_D2 controls the read wait; if enabled, the SDIO_CK controls the read wait. 0: Disabled 1: Enabled
Bit 9	RDWTSTOP	0x0	rw	Read wait stop This bit is set or cleared by software. While the RDWTSTART is set, If this bit is set, it indicates that read wait is stopped; if this bit cleared, it indicates that the read wait is in progress. 0: Read wait is in progress if the RDWTSTART is set. 1: Read wait is stopped if the RDWTSTART is set.
Bit 8	RDWTSTART	0x0	rw	Read wait start This bit is set or cleared by software. When this bit is set, read wait starts; when this bit is cleared, no actions occurs. 0: Read wait disabled 1: Read wait enabled
Bit 7: 4	BLKSIZE	0x0	rw	Data block size This bit is set or cleared by software. This field defines the length of data block when the block data transfer is selected. 0000: block length = 2^0 = 1 byte 0001: block length = 2^1 = 2 bytes 0010: block length = 2^2= 4 bytes 0011: block length = 2^3= 8 bytes 0100: block length = 2^4= 16 bytes 0101: block length = 2^5= 32 bytes 0110: block length = 2^6= 64 bytes 0111: block length = 2^7= 128 bytes 1000: block length = 2^8= 256 bytes 1001: block length = 2^9= 512 bytes 1010: block length = 2^{10}= 1024 bytes 1011: block length = 2^{11}= 2048 bytes 1100: block length = 2^{12}= 4096 bytes 1101: block length = 2^{13}= 8192 bytes 1110: block length = 2^{14} = 16384 bytes 1111: Reserved
Bit 3	DMAEN	0x0	rw	DMA enable bit This bit is set or cleared by software. 0: Disabled 1: Enabled
Bit 2	TFRMODE	0x0	rw	Data transfer mode selection This bit is set or cleared by software. If this bit is set, it indicates stream data transfer; if this bit cleared, it indicates block data transfer. 0: Disabled 1: Enabled
Bit 1	TFRDIR	0x0	rw	Data transfer direction selection This bit is set or cleared by software. If this bit is set, data transfer is from a card to a controller; if this bit is cleared, data transfer is from a controller to a card. 0: Disabled

				1: Enabled
Bit 0	TFREN	0x0	rw	Data transfer enabled bit This bit is set or cleared by software. If this bit is set, data transfer starts. The DCSM enters the Wait_S or Wait_R state, depending on the direction bit TFRDIR. The DCSM goes to the read wait state if the RDWTSTART bit is set from the beginning of the transfer. It is not necessary to clear the enable bit after the end of data transfer but the SDIO_DTCTRL must be updated to enable a new data transfer. 0: Disabled 1: Enabled

Note: This register cannot be written within seven HCLK clock periods after data is written.

25.4.10 SDIO data counter register (SDIO_DTCNTR)

The SDIO_DTCNTR register loads the value from the SDIO_DTLEN register when the DCSM moves from the idle state to the Wait_R or Wait_S state. During the data transfer, the counter value decrements to 0, and then the DCSM enters the idle state and sets the data status end flag bit DTCMPL.

Bit	Register	Reset value	Type	Description
Bit 31: 25	Reserved	0x00	resd	Kept at its default value.
Bit 24: 0	CNT	0x0000000	ro	Data count value When this register is read, the number of data bytes to be transferred is returned. Write access has no effect.

Note: This register can be read only when the data transfer is complete.

25.4.11 SDIO status register (SDIO_STS)

The SDIO_STS is a read-only register, containing two types of flags:

- Static flags (bits [23: 22, 10: 0]): These bits can be cleared by writing to the SDIO_INTCLR register.
- Dynamic flags (bit [21: 11]): These bit status changes with the state of the corresponding logic (for example, BUT full or empty flag is set or cleared as data written to the BUF)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value.
Bit 22	IOIF	0x0	ro	SD I/O interrupt received
Bit 21	RXBUF	0x0	ro	Data available in receive BUF
Bit 20	TXBUF	0x0	ro	Data available in transmit BUF
Bit 19	RXBUFE	0x0	ro	Receive BUF empty
Bit 18	TXBUFE	0x0	ro	Transmit BUF empty If hardware flow control is enabled, the TXBUF_E signal becomes valid when the BUF contains two words.
Bit 17	RXBUFF	0x0	ro	Receive BUF full If hardware flow control is enabled, the RXBUF_F becomes valid two words before the BUF is full.
Bit 16	TXBUFF	0x0	ro	Transmit BUF full
Bit 15	RXBUFH	0x0	ro	Receive BUF half full There are at least 8 words in the BUF. This flag bit can be used as DMA request.
Bit 14	TXBUFH	0x0	ro	Transmit BUF half empty: At least 8 words can be written to the BUF. This flag bit can be used as DMA request.
Bit 13	DORX	0x0	ro	Data receive in progress
Bit 12	DOTX	0x0	ro	Data transmit in progress
Bit 11	DOCMD	0x0	ro	Command transfer in progress
Bit 10	DTBLKCMPL	0x0	ro	Data block sent/received CRC check passed)
Bit 9	SBITERR	0x0	ro	Start bit not detected on all data signals in wide bus mode
Bit 8	DTCMPL	0x0	ro	Data end (data counter, SDIO CNT, is zero)
Bit 7	CMDCMPL	0x0	ro	Command sent (no response required)
Bit 6	CMDRSPCMPL	0x0	ro	Command response (CRC check passed)
Bit 5	RXERRO	0x0	ro	Received BUF overrun error

Bit 4	TXERRU	0x0	ro	Transmit BUF underrun error
Bit 3	DTTIMEOUT	0x0	ro	Data timeout
Bit 2	CMDTIMEOUT	0x0	ro	Command response timeout The command timeout is a fixed value of 64 SDIO_CK clock periods.
Bit 1	DTFAIL	0x0	ro	Data block sent/received (CRC check failed)
Bit 0	CMDFAIL	0x0	ro	Command response received (CRC check failed)

25.4.12 SDIO clear interrupt register (SDIO_INTCLR)

The SDIO_INTCLR is a read-only register. Writing 1 to the corresponding register bit will clear the correspond bit in the SDIO_STS register.

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value.
Bit 22	IOIF	0x0	rw	SD I/O interface flag clear bit This bit is set by software to clear the IOIF flag.
Bit 21: 11	Reserved	0x000	resd	Kept at its default value.
Bit 10	DTBLKCMPL	0x0	rw	DTBLKCMPL flag clear bit This bit is set by software to clear the DTBLKCMPL flag.
Bit 9	SBITERR	0x0	rw	SBITERR flag clear bit This bit is set to clear the SBITERR flag.
Bit 8	DTCMPL	0x0	rw	DTCMPL flag clear bit This bit is set by software to clear the DTCMPL flag.
Bit 7	CMDCMPL	0x0	rw	CMDCMPL flag clear bit This bit is set by software to clear the CMDCMPL flag.
Bit 6	CMDRSPCMPL	0x0	rw	MDRSPCMPL flag clear bit This bit is set by software to clear the CMDRSPCMPL flag.
Bit 5	RXERRO	0x0	rw	RXERRO flag clear bit This bit is set by software to clear the RXERRO flag.
Bit 4	TXERRU	0x0	rw	TXERRU flag clear bit This bit is set by software to clear the TXERRU flag.
Bit 3	DTTIMEOUT	0x0	rw	DTTIMEOUT flag clear bit This bit is set by software to clear the DTTIMEOUT flag.
Bit 2	CMDTIMEOUT	0x0	rw	CMDTIMEOUT flag clear bit This bit is set by software to clear the CMDTIMEOUT flag.
Bit 1	DTFAIL	0x0	rw	DTFAIL flag clear bit This bit is set by software to clear the DTFAIL flag.
Bit 0	CMDFAIL	0x0	rw	CMDFAIL flag clear bit This bit is set by software to clear the CMDFAIL flag.

25.4.13 SDIO interrupt mask register (SDIO_INTEN)

The SDIO_INTEN register determines which status bit generates an interrupt by setting the corresponding bit.

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value.
Bit 22	IOIFIEN	0x0	rw	SD I/O mode received interrupt enable This bit is set or cleared by software to enable/disable the SD I/O mode received interrupt function. 0: Disabled 1: Enabled
Bit 21	RXBUFIEN	0x0	rw	Data available in RxBUF interrupt enable This bit is set or cleared by software to enable/disable the Data Available in RxBUF Interrupt. 0: Disabled 1: Enabled
Bit 20	TXBUFIEN	0x0	rw	Data available in TxBUF interrupt enable This bit is set or cleared by software to enable/disable the Data Available in TxBUF Interrupt. 0: Disabled 1: Enabled
Bit 19	RXBUFEIEN	0x0	rw	RxBUF empty interrupt enable This bit is set or cleared by software to enable/disable the

				RxBUF empty interrupt. 0: Disabled 1: Enabled
Bit 18	TXBUFEIEN	0x0	rw	TxBUF empty interrupt enable This bit is set or cleared by software to enable/disable the TxBUF empty interrupt. 0: Disabled 1: Enabled
Bit 17	RXBUFFIEN	0x0	rw	RxBUF full interrupt enable This bit is set or cleared by software to enable/disable the RxBUF full interrupt. 0: Disabled 1: Enabled
Bit 16	TXBUFFIEN	0x0	rw	TxBUF full interrupt enable This bit is set or cleared by software to enable/disable the TxBUF full interrupt. 0: Disabled 1: Enabled
Bit 15	RXBUFHIEN	0x0	rw	RxBUF half full interrupt enable This bit is set or cleared by software to enable/disable the RxBUF half full interrupt. 0: Disabled 1: Enabled
Bit 14	TXBUFHIEN	0x0	rw	TxBUF half empty interrupt enable This bit is set or cleared by software to enable/disable the TxBUF half empty interrupt. 0: Disabled 1: Enabled
Bit 13	DORXIEN	0x0	rw	Data receive acting interrupt enable This bit is set or cleared by software to enable/disable the Data receive acting interrupt. 0: Disabled 1: Enabled
Bit 12	DOTXIEN	0x0	rw	Data transmit acting interrupt enable This bit is set or cleared by software to enable/disable the Data transmit acting interrupt. 0: Disabled 1: Enabled
Bit 11	DOCMDIEN	0x0	rw	Command acting interrupt enable This bit is set or cleared by software to enable/disable the Command acting interrupt. 0: Disabled 1: Enabled
Bit 10	DTBLKCMPLIEN	0x0	rw	Data block end interrupt enable This bit is set or cleared by software to enable/disable the Data block end interrupt. 0: Disabled 1: Enabled
Bit 9	SBITERRIEN	0x0	rw	Start bit error interrupt enable This bit is set or cleared by software to enable/disable the Start bit error interrupt. 0: Disabled 1: Enabled
Bit 8	DTCMPLIEN	0x0	rw	Data end interrupt enable This bit is set or cleared by software to enable/disable the Data end interrupt. 0: Disabled 1: Enabled
Bit 7	CMDCMPLIEN	0x0	rw	Command sent interrupt enable This bit is set or cleared by software to enable/disable the Command sent interrupt. 0: Disabled 1: Enabled
Bit 6	CMDRSPCMPLIEN	0x0	rw	Command response received interrupt enable

				This bit is set or cleared by software to enable/disable the Command response received interrupt. 0: Disabled 1: Enabled
Bit 5	RXERROIEN	0x0	rw	RxBUF overrun error interrupt enable This bit is set or cleared by software to enable/disable the RxBUF overrun error interrupt. 0: Disabled 1: Enabled
Bit 4	TXERRUIEN	0x0	rw	TxBUF underrun error interrupt enable This bit is set or cleared by software to enable/disable the TxBUF underrun error interrupt. 0: Disabled 1: Enabled
Bit 3	DTTIMEOUTIEN	0x0	rw	Data timeout interrupt enable This bit is set or cleared by software to enable/disable the Data timeout interrupt. 0: Disabled 1: Enabled
Bit 2	CMDTIMEOUTIEN	0x0	rw	Command timeout interrupt enable This bit is set or cleared by software to enable/disable the Command timeout interrupt. 0: Disabled 1: Enabled
Bit 1	DTFAILIEN	0x0	rw	Data CRC fail interrupt enable This bit is set or cleared by software to enable/disable the Data CRC fail interrupt. 0: Disabled 1: Enabled
Bit 0	CMDFAILIEN	0x0	rw	Command CRC fail interrupt enable This bit is set or cleared by software to enable/disable the Command CRC fail interrupt. 0: Disabled 1: Enabled

25.4.14 SDIOBUF counter register (SDIO_BUFCNTR)

The SDIO_BUFCNTR register contains the number of words to be written to or read from the BUF. The BUF counter loads the value from the SDIO_DTLEN register when the data transfer bit TFREN is set in the SDIO_DTCTRL register. If the data length is not word-aligned, the remaining 1 to 3 bytes are regarded as a word.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23: 0	CNT	0x000000	ro	Number of words to be written to or read from the BUF.

25.4.15 SDIO data BUF register (SDIO_BUF)

The receive and data BUF is group of a 32-bit wide registers that can be written or read. The BUF contains 32 registers on 32 sequential addresses. The CPU can use BUF for read/write multiple operations.

Bit	Register	Reset value	Type	Description
Bit 31: 0	DT	0x0000 0000	rw	Receive and transmit BUF data The BUF data occupies 32x 32-bit words, the address: SDIO base + 0x80 to SDIO base + 0xFC

26 Ethernet media access control (EMAC)

This module applies only to AT32F437 series, not including AT32F435 series.

26.1 EMAC introduction

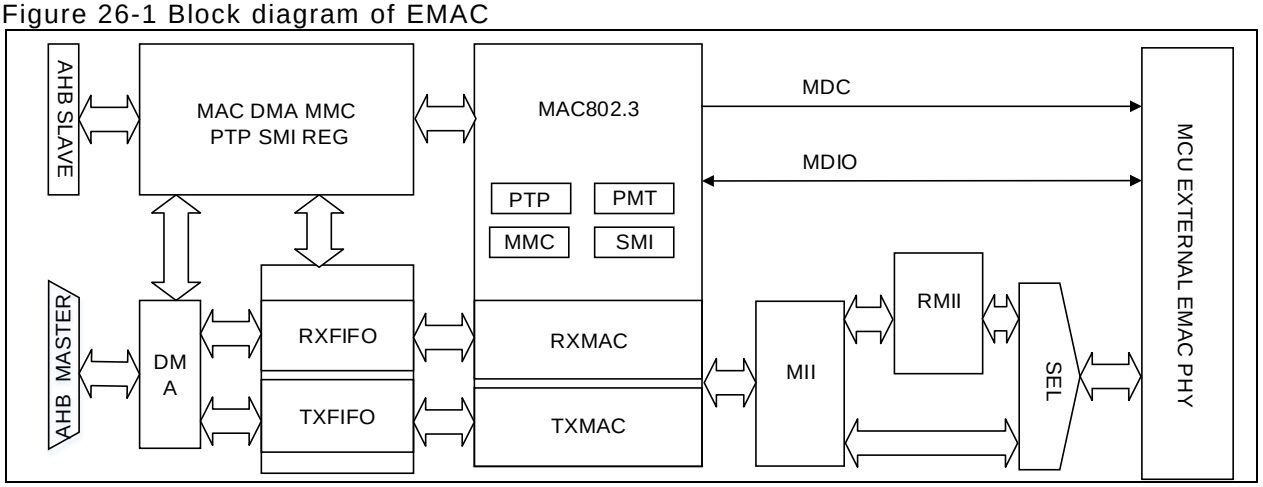
Copyright Synopsys, Inc. All rights reserved.

The Ethernet peripheral enables the AT32F437 to transmit and receive data (10/100Mbps) through Ethernet in compliance with IEEE 802.3-2002 standard.

The AT32F437 Ethernet peripheral supports two standard interfaces to the external PHY: media independent interface (MII) defined in the IEEE 802.3 standard and the reduced media independent interface (RMII).

26.1.1 EMAC structure

Figure 26-1 Block diagram of EMAC



26.1.2 EMAC main features

The Ethernet peripheral contains an EMAC core and a DMA controller. The transmission and reception of the frame is scheduled by DMA.

DMA features

- Programmable AHB burst transfer types
- Supports ring or chained descriptor
- Each descriptor can transfer up to 8 KB of data
- Poll or fixed-priority arbitration between transmission and reception
- Programmable interrupts for different operational conditions
- Status information report for each transfer

EMAC core features

- Supports 10/100Mbps data transfer rates
- IEEE 802.3-compliant MII interface to communicate with an external high-speed Ethernet PHY
- Supports flow control for full-duplex operation, and CSMA/CD protocol for half-duplex operations
- Automatic CRC and pad generation controllable on transmission frames
- Automatically remove pad bits/CRC on reception frames
- Programmable frame length up to 16 KB
- Programmable frame gap (40-96 bits)
- Supports a variety of address filtering modes and promiscuous modes
- Supports IEEE 802.1Q VLAN tag detection for reception frames
- Supports mandatory network statistics with RMON/MIB counters (RFC2819/RFC2665)
- Detection of LAN remote wakeup frames and AMD Magic Packet™ frames

- Supports checking IPv4 header checksum and IPv4, TCP, UDP or ICMP (packaged in IPv4 or IPv6 data formats) checksum
- Supports Ethernet frame time stamp as defined in IEEE 1588-2008. 64-bit time stamps are recorded in the transmit or receive status
- Two 2 KB FIFOs: one for transmit, and one for receive with a configurable threshold
- Filter received error frames and not forward them to the application in store-forward mode
- Supports store-forward mechanism for data transfer to the MAC controller
- Discard frames on late collision, excessive collisions, excessive deferral and underflow conditions
- Clear FIFO by software
- Calculates and inserts IPv4 header checksum and TCP, UDP or ICMP checksum in frames transmitted in store-forward mode
- Supports loopback mode on the MII interface for debugging
- Programmable time stamps of receive and transmit frames as defined in IEEE 1588-2008 standard
- Supports two correction methods: coarse and fine correction
- Second pulse output (programmable)
- Trigger interrupts when the system is greater the specified time

26.2 EMAC functional description

The Ethernet peripheral consists of a MAC 802.3 (media access controller), MII interface and a dedicated DMA controller.

It implements the following functions:

- Data transmit and receive
  - Framing (frame boundary and frame synchronization)
  - Handling of source and destination addresses
  - Error detection
- Media access management in half-duplex mode
  - Medium allocation (avoid collision)
  - Collision resolve (handle collision)

Usually there are two operating modes for the MAC sublayer:

- Half-duplex mode: The stations compete for the use of the physical medium using the CSMA/CD algorithm. Two stations can only communicate in a single transfer direction at the same time.
- Full-duplex mode: CSMA/CD algorithm is unnecessary, but the following conditions must be met:
  - Physical medium supports simultaneous transmission and reception
  - Only two stations connected to the LAN
  - Both two stations configured as full-duplex mode

26.2.1 EMAC communication interfaces

The EMAC allows to configure the station management interface (SMI) of PHY, media-independent interface (MII) for Ethernet frame communication, and reduced media-independent interface RMII.

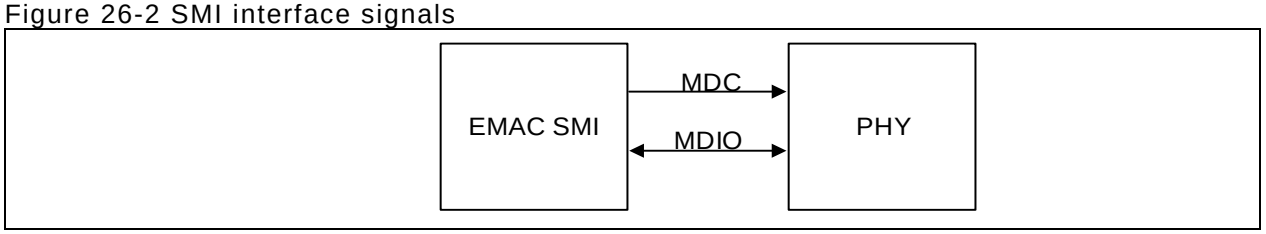
Station management interface (SMI)

The PHY management interface (SMI) accesses PHY registers through a clock and data line. It supports up to 32 PHYs.

MDC: PHY configures clock signals, at the maximum frequency of 2.5 MHz. The minimum high and low times for MDC must be 160ns, and the minimum period is 400ns. In idle state, the MDC clock signal remains low.

MDIO: Bidirectional port, data input/output lines.

Figure 26-2 SMI interface signals



Before write operation, PHY address, MII register and EMAC_MACMIIDT register must be configured first, followed by the MII MW and MB bits, and then the SMI interface will transfer the PHY address, PHY register address and data to the PHY. During the transaction, the contents of the EMAC_MACMIIADDR and the EMAC_MACMIIDT registers are not allowed to change.

Before read operation, the PHY address and MII register must be configured first, and then the MB bit is set and MW bit is set to 0 in the EMAC_MACMIIADDR register.

The SMI interface sends the PHY address and PHY register address, and then starts reading the PHY register contents. During the transaction, the MB bit is always set. It is cleared by the SMI interface at the end of read operation. Attention should be paid to the fact that the contents of the EMAC_MACMIIADDR and EMAC_MACMIIDT registers are not allowed to change (The application should not change these register contents. After the transaction, the EMAC_MACMIIDT register is automatically updated with the data read from the SMI).

The SMI clock source is a divided AHB clock. The divide factor depends on the ABH clock frequency. Note that the divide factor must be configured correctly since the MDC frequency must not be greater than 2.5 MHz.

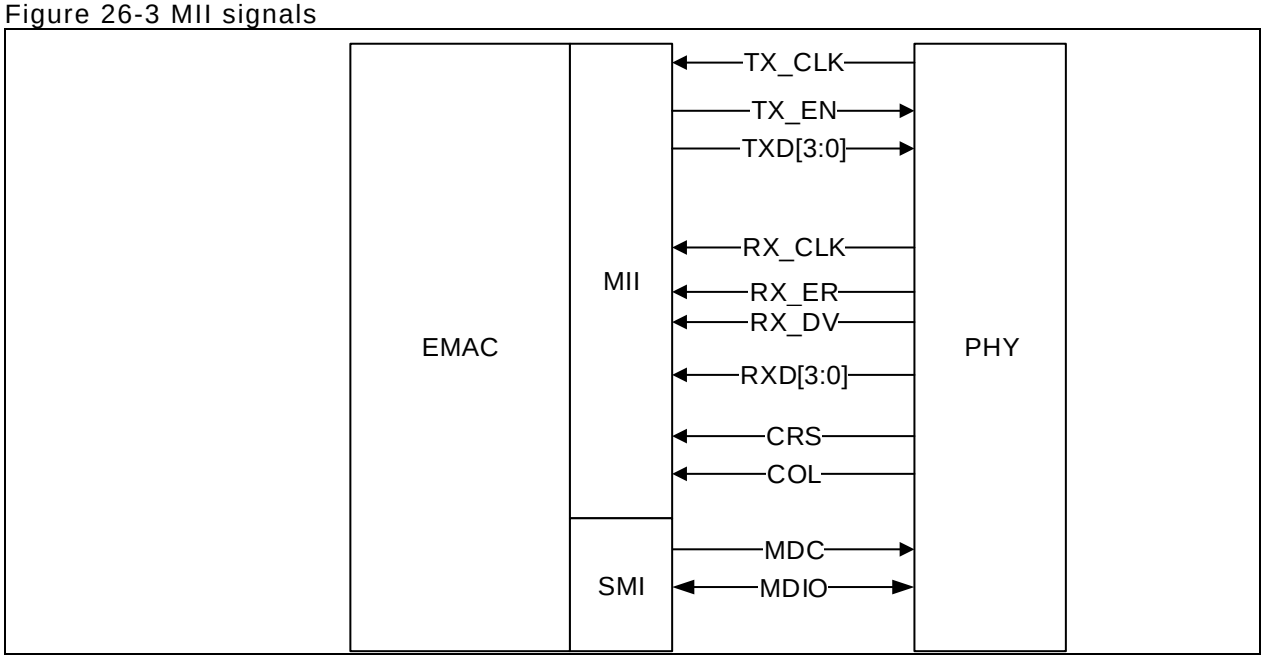
Table 26-1 shows the clock range.

Selection bit	AHB clock	MDC clock
0000	60~100MHz	AHB clock/42
0001	100~150MHz	AHB clock/62
0010	20~35MHz	AHB clock/16
0011	35~60MHz	AHB clock/26
0100	150~250MHz	AHB clock/102
0101	250~288MHz	AHB clock/124
0110,0111	Reserved	--

Media-independent interface: MII

The media-independent interface (MII) acts an interconnection between the MAC sublayer and the PHY for data transfer at 10Mbit/s and 100Mbit/s.

Figure 26-3 MII signals



MII_TX_CLK: Transmit data clock signal. This clock is 2.5MHz at 10Mbps speed; 25MHz at 100Mbps speed.

MII_RX_CLK: Receive data clock signal. This clock is 2.5MHz at 10Mbps speed; 25MHz at 100Mbps speed.

MII_TX_EN: Transmit enable signal. It must be set synchronously with the start bit of the preamble, and must remain asserted until all bits to be transmitted are transmitted.

MII_TXD[3: 0]: Transmit data. 4-bit data are transmitted each time synchronously. Data is valid when the MII_TX_EN signal is active. The MII_TXD[0] is the least significant bit while the MII_TXD[3] is the most significant bit.

MII_CRS: carrier sense signal defined in the CSMA/CD. It is controlled by the PHY and can be valid only in half-duplex mode. This signal is active when either the transmit or receive medium is not idle. The PHY must ensure that the MII_CS signal remains active throughout the duration of a collision condition. This signal is not required to be synchronized with the TX and RX clocks.

MII_COL: Collision detection signal defined in CSMA/CD. It is controlled by the PHY and can be valid only in half-duplex mode. This signal is enabled upon detection of a collision on the medium and will remain enabled during the duration of the collision. This signal is not required to be synchronized with the TX and RX clocks.

MII_RXD[3: 0]: It is controlled by the PHY. Four-bit data to be received are transmitted synchronously. Data is valid when the MII_RX_DV signal is active. The MII_RXD[0] is the least significant bit, while the MII_RXD[3] is the most significant bit. While the MII_RX_DV is inactive but the MII_RX_ER is active, the PHY will transmit a specific MII_RXD[3: 0] value to indicate specific information.

MII_RX_DV: Receive data valid signal. It is controlled by the PHY. This signal is valid when the PHY has kept data on the MII for reception. It must be enabled synchronously with the first bit (MII_RX_CLK) and must remain enabled until data transfer is fully completed. It must be cleared prior to the first clock following the transmission of the final four-bit data. In order to receive a frame correctly, the MII_RX_DV signal must remain valid throughout the frame transmission. The active level must be no later than the SFO bit.

MII_RX_ER: Receive error signal. It must be active for one or more clock periods (MII_RX_CLK) to indicate to the MAC that an error was detected in a frame. Detailed error information depends on the state of the MII_RX_DV and MII_RXD[3: 0] data.

Table 26-2 Transmit interface signal encode

MII_TX_EN	MII_TXD[3: 0]	Description
0	0000 to 1111	Normal frame interval
1	0000 to 1111	Normal data transfer

Table 26-3 Receive interface signal encode

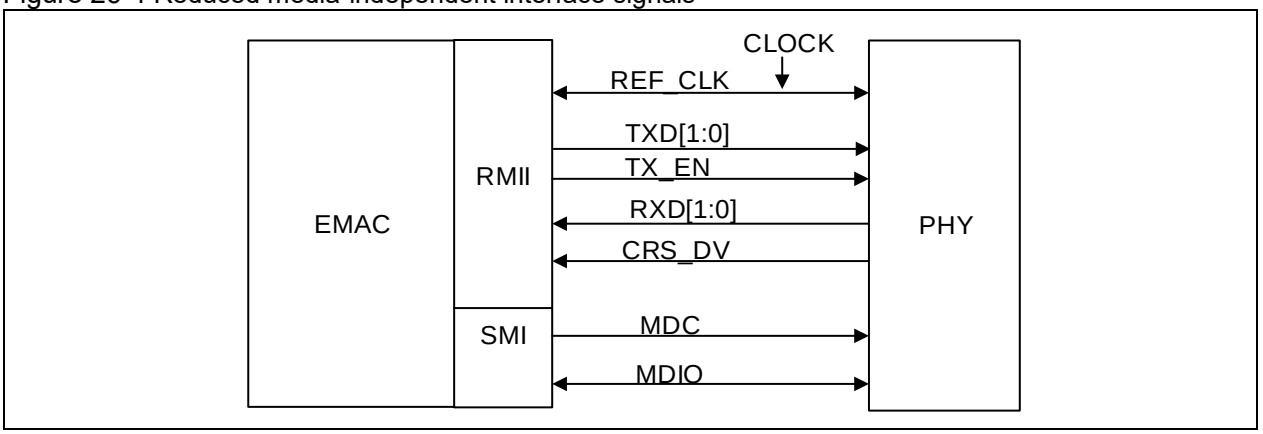
MII_RX_DV	MII_RX_ER	MII_RXD[3: 0]	Description
0	0	0000 to 1111	Normal frame interval
0	1	0000	Normal frame interval
0	1	0001 to 1101	Reserved
0	1	1110	False carrier indication
0	1	1111	Reserved
1	0	0000 to 1111	Normal data reception
1	1	0000 to 1111	Data reception error

Reduced media-independent interface: RMII

The reduced media-independent interface reduces the pin count between the AT32F437xx Ethernet peripheral and the external Ethernet. According to the IEEE802.3u standard, the MII interface requires 16 pins for data and control signals, while the RMII reduces the pin count to 7 pins (a 62.5% decrease in pin count)

The RMII interface is used to connect the EMAC and the PHY. This helps translate the MAC MII signal to the RMII.

Figure 26-4 Reduced media-independent interface signals



MII/RMII selection and clock sources

Either the MII or RMII mode can be selected using the 23rd bit MII_RMII_SEL in the SCFG_CFG2 register. The MII/RMII mode must be selected when the Ethernet controller is in reset state or before the clock is enabled.

MII clock sources

The EMAC TX_CLK and RX_CLK clock signals are provided by the PHY. External PHY module is driven by an external 25MHz clock, which can be provided by either an oscillator or the MCU CLKOUT pin. Refer to CRM section for more information.

Figure 26-5 MII clock sources (provided by CLKOUT pin)

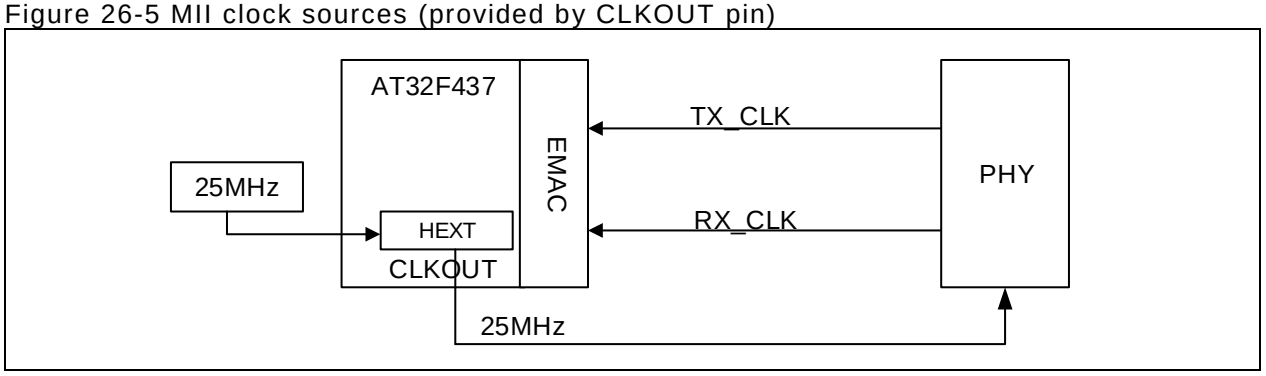
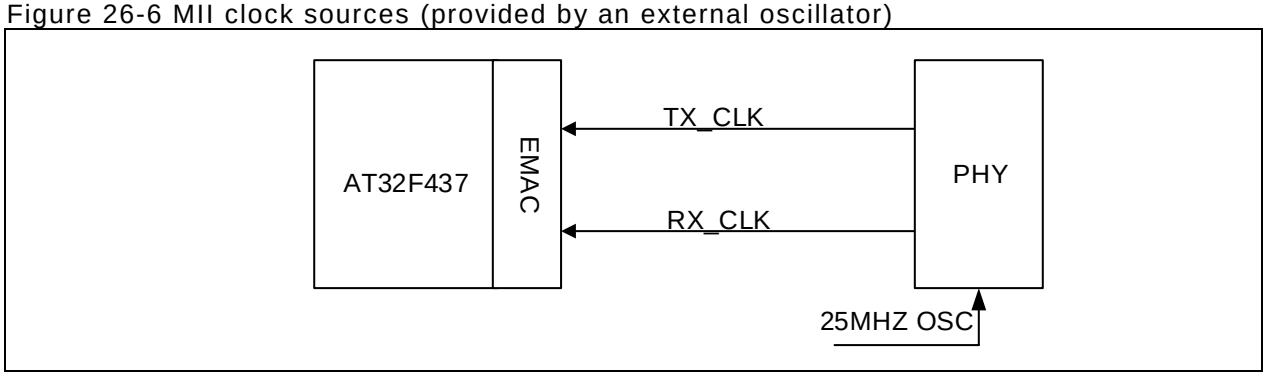


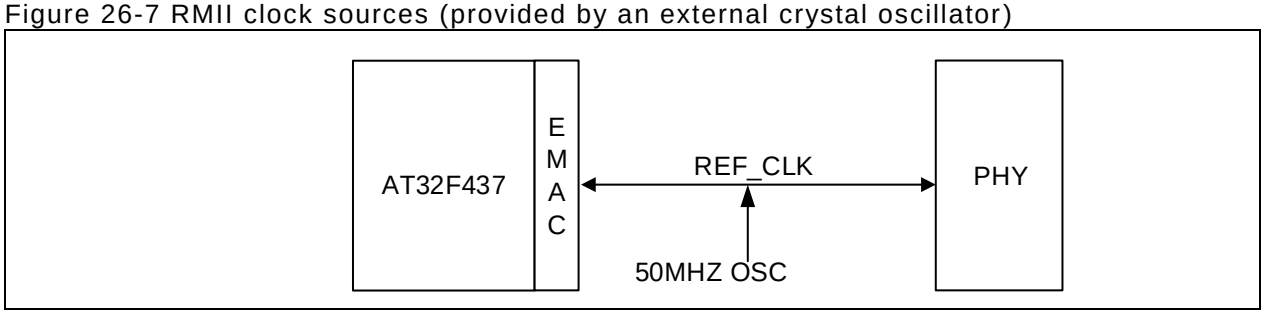
Figure 26-6 MII clock sources (provided by an external oscillator)



RMII clock sources

As shown in Figure 26-7, both the EMAC and PHY require 50MHz clock sources, which can be done with an external crystal oscillator.

Figure 26-7 RMII clock sources (provided by an external crystal oscillator)



EMAC pin allocation and multiplexing

Table 26-4 Ethernet peripheral pin configuration

EMAC signal	MII	RMII	Pin	Pin description
EMAC_MDC	MDC	MDC	PC1	Multiplexed push-pull output
EMAC_MII_TXD2	TXD2	-	PC2	Multiplexed push-pull output
EMAC_MII_TX_CLK	TX_CLK	-	PC3	Floating input (reset state)
EMAC_MII_CRS	CRS	-	PA0	Floating input (reset state)
EMAC_MII_RX_CLK EMAC_RMII_REF_CLK	RX_CLK	REF_CLK	PA1	Floating input (reset state)
EMAC_MDIO	MDIO	MDIO	PA2	Multiplexed push-pull output
EMAC_MII_COL	COL	-	PA3	Floating input (reset state)
EMAC_MII_RX_DV EMAC_RMII_CRS_DV	RX_DV	CRS_DV	PA7	Floating input (reset state)
EMAC_MII_RXD0 EMAC_RMII_RXD0	RXD0	RXD0	PC4	Floating input (reset state)
EMAC_MII_RXD1 EMAC_RMII_RXD1	RXD1	RXD1	PC5	Floating input (reset state)
EMAC_MII_RXD2	RXD2	-	PB0	Floating input (reset state)
EMAC_MII_RXD3	RXD3	-	PB1	Floating input (reset state)
EMAC_MII_RX_ER	RX_ER	-	PB10	Floating input (reset state)
EMAC_MII_TX_EN EMAC_RMII_TX_EN	TX_EN	TX_EN	PB11	Multiplexed push-pull output
EMAC_MII_TXD0 EMAC_RMII_TXD0	TXD0	TXD0	PB12	Multiplexed push-pull output
EMAC_MII_TXD1 EMAC_RMII_TXD1	TXD1	TXD1	PB13	Multiplexed push-pull output
EMAC_PPS_OUT	PPS_OUT	PPS_OUT	PB5	Multiplexed push-pull output
EMAC_MII_TXD3	TXD3	-	PB8	Multiplexed push-pull output
EMAC_RMII_CRS_DV	RX_DV	CRS_DV	PD8	Floating input (reset state)
EMAC_MII_RXD0 EMAC_RMII_RXD0	RXD0	RXD0	PD9	Floating input (reset state)
EMAC_MII_RXD1 EMAC_RMII_RXD1	RXD1	RXD1	PD10	Floating input (reset state)
EMAC_MII_RXD2	RXD2	-	PD11	Floating input (reset state)
EMAC_MII_RXD3	RXD3	-	PD12	Floating input (reset state)

26.2.2 EMAC frame communication

Frame format

Figure 26-8 shows the MAC frame format and tagged MAC frame format (Refer to IEEE 802.C-2002 for more information on MAC frame formats)

Figure 26-8 MAC frame format

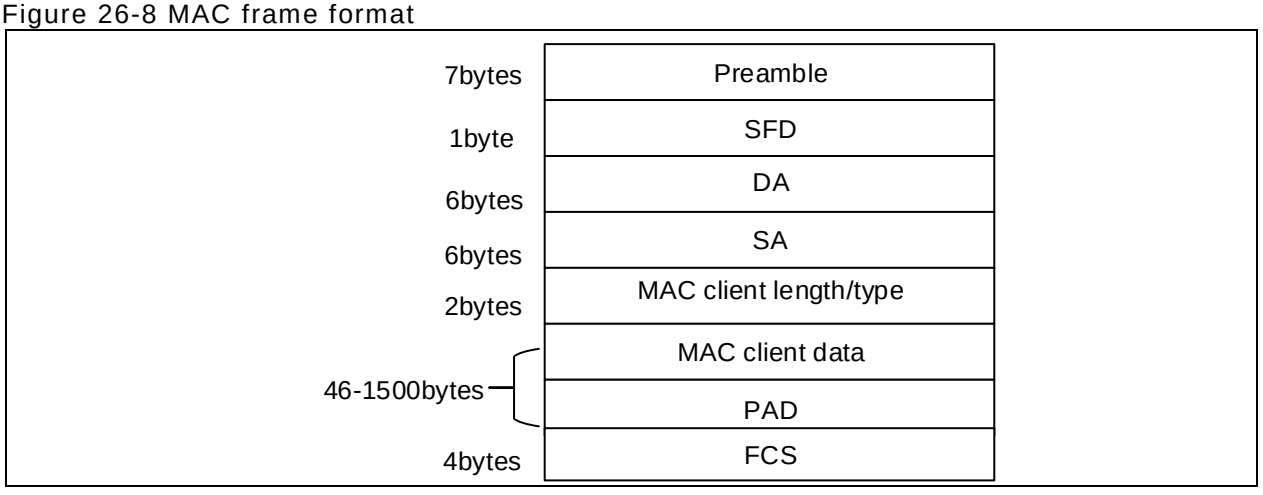
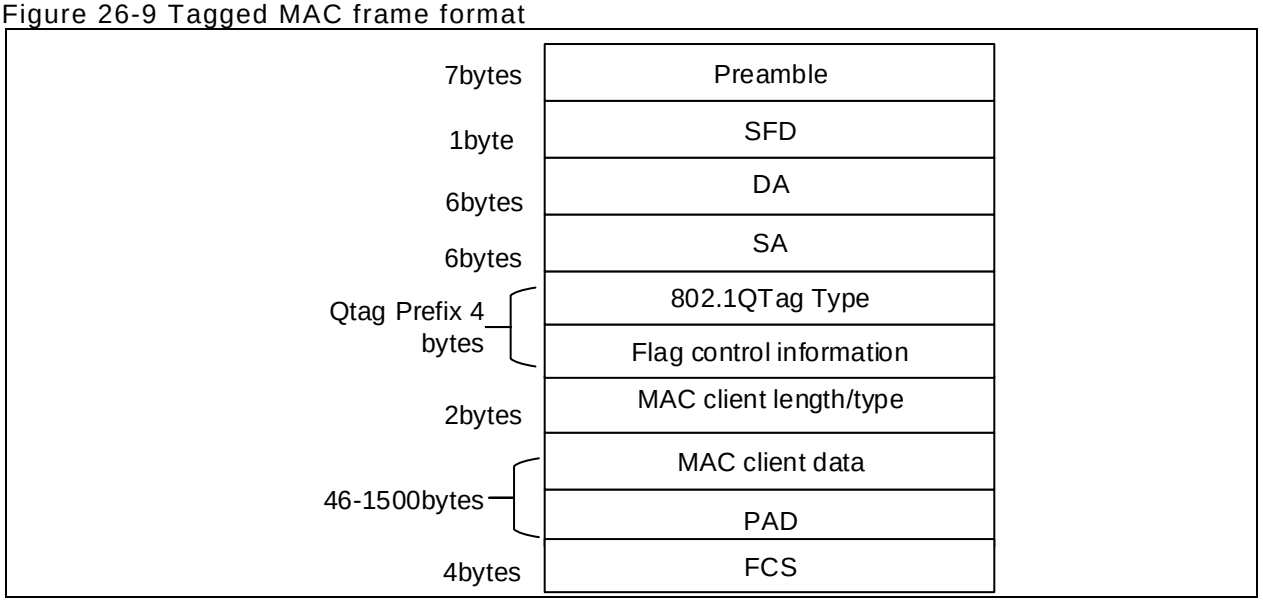


Figure 26-9 Tagged MAC frame format



EMAC frame filtering

EMAC supports source and destination address filtering.

Address filtering

Based on frame filtering register chosen by the application, address filtering checks the source and destination addresses over all received frames using the MAC address and multicast HASH table. The address filtering status is reported accordingly.

Unicast destination address filter

There are two filtering modes: perfect address filtering and HASH table filtering.

1. Perfect address filtering: It is enabled by setting HUC=0 in the frame filtering register. Four MAC addresses are used for perfect filtering. The MACADDR0 is always enabled. The MACADDR1, MACADDR2 and MACADDR3 bits are enabled using their respective AE bit. The MBC bit in the address register is set to mask the address comparison of the corresponding bytes. If the MBC bit is all 0, the EMAC will compare all 48 bits of the received unicast address with the programmed MAC address, if matched, the unicast address is said to have passed through the filtering.
2. HASH table filtering: The HUC must be set in the frame filtering register. The EMAC performs imperfect filtering for unicast addresses through 64-bit HASH table. For HASH filtering, the MAC calculates the CRC value of the received destination address (see the note below) and uses the 6 upper CRC bits to index the HASH table. A value of 000000 corresponds to bit 0 in the HASH table register, and a value of 111111 corresponds to bit 63 in the HASH table register. If the corresponding bit in the HASH table relative to the CRC value is set to 1, it indicates that the frame has passed through the HASH filter, otherwise, it has failed the HASH filter.

Multicast destination address filter

1. The MAC can be programmed to receive all multicast frames by setting PMC=1 in the frame filter register
2. If the PMC is set to 0 and the HMC is set to 0, perfect address filtering can be done using the MACADDR0/1/2/3 addresses.
3. If the PMC is set to 0, and the HMC is set to 1, the 64-bit HASH table is used to perform imperfect filtering.

Broadcast address filter

If the DBF is set in the frame filter register, the EMAC will reject all broadcast frames. If DBF=0, the EMAC will accept all broadcast frames.

Unicast source address filter

If the bit 30 is set in the MAC address register 1/2/3, the filter will compare source address instead of destination address of the received frames.

If the SAF bit is set in the frame filter address, the frames that failed the SA filter will be dropped by the

EMAC. In this case, the frames that pass through both SA and DA filtering can be forwarded to the application; otherwise, they will be dropped.

Inverse filtering operation

The DAIF and SAIF bits in the frame filter register are used to invert the filtering output result for both destination and source address filtering. The DAIF bit is applicable for both unicast and multicast destination frames, while the SAIF bit for the unicast source address.

Table 26-5 Destination address filtering

Frame type	PMC	HPF	HUC	DAIF	HMC	PMC	DBF	DA filter operation
Broadcast	1	X	X	X	X	X	X	Pass
	0	X	X	X	X	X	0	Pass
	0	X	X	X	X	X	1	Fail
Unicast	1	X	X	X	X	X	X	Pass all frames
	0	X	0	0	X	X	X	Pass on perfect/group filter match
	0	X	0	1	X	X	X	Fail on perfect/group filter match
	0	0	1	0	X	X	X	Pass on HASH filter match
	0	0	1	1	X	X	X	Fail on HASH filter match
	0	1	1	0	X	X	X	Pass on HASH or perfect/group filter match
	0	1	1	1	X	X	X	Fail on HASH or perfect/group filter match
Multicast	1	X	X	X	X	X	X	Pass all frames
	X	X	X	X	X	1	X	Pass all frames
	0	X	X	0	0	0	X	Pass on perfect/group filter match and drop PAUSE frames if PCF= 0x
	0	0	X	0	1	0	X	Pass on HASH filter match and drop PAUSE frames if PCF= 0x
	0	1	X	0	1	0	X	Pass on HASH or perfect/group filter match and drop PAUSE frames if PCF= 0x
	0	X	X	1	0	0	X	Fail on perfect/group filter match and drop PAUSE frames if PCF= 0x
	0	0	X	1	1	0	X	Fail on HASH filter match and drop PAUSE frames if PCF= 0x
	0	1	X	1	1	0	X	Fail on HASH or perfect/group filter match and drop PAUSE frames if PCF= 0x

Table 26-6 Source address filtering

Frame type	PR	SAIF	SAF	SA filter operation
Unicast	1	X	X	Pass all frames
	0	0	0	Pass status on perfect/group filter match but do not drop frames that failed
	0	1	0	Fail status on perfect/group filter match but do not drop frames that failed
	0	0	1	Pass on perfect/group filter match and drop frames that failed
	0	1	1	Fail on perfect/group filter match and drop frames that failed

EMAC frame transmission

The EMAC frame transmission is controlled by the DMA controller and MAC. Once a transmission command is sent by the application, Ethernet frames read from the user data buffer (such as SRAM) are stored into TXFIFO by the DMA. The frames are then popped out and transferred to the MAC core. The MAC core sends the frames to external Ethernet PHY via MII/RMII interface so as to communicate with external stations. When the end-of-frame is transferred, the status of the transmission is generated from the MAC core and transferred back to the DMA controller.

When the SOF is detected, the MAC accepts data and begins transmitting to the MII/RMII. The time required to transmit the data frame to the MII/RMII after the application initiates is variable, due to various delay factors like frame interval, time to transmit preamble/SFD and any back-off delays caused by

CSMA/CD algorithm in half-duplex mode. After the EOF is transferred to the MAC core, the core completes normal transmission and then gives the status of the transmission back to the DMA.

There are two modes of operation for popping data from TXFIFO to the MAC core:

- In threshold mode: If the number of bytes in the FIFO crosses the configured threshold (or when the EOF is written before the threshold is crossed), the data is ready to be popped out and forwarded to the MAC core. The threshold level is configured using the TTC bits in the EMAC_DMAOPM register.
- In store-and-forward mode: Only after a complete frame is written to the FIFO, the data is transferred to the MAC core. If the TXFIFO size is smaller than the Ethernet frame to be transmitted, then the data is also transferred to the MAC core when the TXFIFO becomes almost full.

  The FTF bit (the bit 20 in the EMAC_DMAOPM register) can be set to flush the TXFIFO. If the FTF bit is mistakenly set in the middle of transferring a frame to the MAC core, the FIFO is flushed and the frame transmission is aborted. An underflow event is marked in the EMAC transmitter and the corresponding status is given to the DMA core.

Automatic CRC and pad generation

If the length of a transmitting fame is less than 46 bytes, the minimum data size for an Ethernet frame defined in IEEE802.3 standard, the EMAC can be programmed to append padding (zeroes are appended) automatically so that zeroes are appended to the transmitting frame to make the data length exactly 46 bytes.

During a frame transmission, the EMAC can be programmed either to append CRC to the frame check sequence or not to append CRC. When the EMAC is programmed to append pads for frames (DA+SA+LT+Data) less than 60 bytes, the CRC value will be appended at the end of the padded frames.

Collision detection in CSMA/CD

The collision detection is applicable only to half-duplex mode, not to full-duplex mode (refer to Ethernet protocol for more information).

In MII mode, if a collision occurs at any time from the beginning of a frame to the end of the CRC, the collision signal is sent to the EMAC core. The EMAC core will send a 32-bit jam signal of 0x5555 5555 to inform all other stations on the LAN that a collision has occurred. If the collision happened during the preamble transmission phrase, the EMAC completes the transmission of the preamble and SFD and then sends the jam signal.

Jabber timer

The EMAC jabber timer is enabled (set EMAC_MACCTRL[20]=0) to prevent certain station on the LAN from occupying the LAN for a long time. Once enabled, the EMAC will stop transmitting if the application tries to send a frame more than 2048 bytes.

Interframe gap management

The EMAC enables transmission after satisfying the interframe gap and backoff delays. The MAC maintains an idle period of the programmed interframe gap (IFG bits in the EMAC_MACCTRL register) between any two transmitted frames. The EMAC starts its IFG counter as soon as the carrier signal of the MII becomes inactive. If a frame arrives sooner that the configured IFG time, it will be delayed for transmission until the interframe time is reached. The MAC starts its IFG counter as soon as the carrier signal of the MII becomes inactive. In full-duplex mode, the MAC enables transmission at the end of the configured IFG value. In half-duplex mode, if the IFG is configured as 96 bit times, the MAC will follow the rule defined in Section 4.2.3.2.1 of the IEEE 802.3 specification. The MAC will reset its IFG counter if a carrier is detected during the first two-thirds of the IFG interval (64-bit times). If the carrier is detected during the final one-third of the IFG interval, the MAC will continue the IFG count and enables transmission after the IFG interval is reached. In half-duplex mode the MAC follows the truncated binary exponential backoff algorithm.

Transmit flow control

In full-duplex mode the EMAC implements Transmit flow control using pause frames. The EMA can request the suspension of the frame transmission by sending Pause frames in two ways.

When the FCB bit in the EMAC_MACFCTRL register is set, the EMAC generates and transmits a single Pause frame. The value of the pause time in the generated frame holds the programmed pause time

value in the EMAC_MACFCTRL register. To extend the pause time or cancel the remaining pause time, the EMAC must send another pause frame (PT=0 will cancel the remaining pause time).

If flow control is enabled (EFT=1 in the EMAC_MACFCTRL register), when the RXFIFO is full, the EMAC generates and transmits a Pause time. If the RXFIFO remains full while the programmed pause time threshold is satisfied, the MAC will transmit another Pause frame. The process is repeated as long as the receive FIFO remains full. If the RXFIFO is not full prior to the pause time, the MAC transmit a Pause time with zero pause time to indicate to the remote station that the receive buffer is ready to receive new data frames.

Retransmission during collision

When a frame is being transferred to the MAC, a collision event may occur on the MAC line in half-duplex mode. The MAC would then try retransmission even before the end of the frame is received. At this point, if retransmission is enabled, the frame is popped out again from the FIFO. After 96 bytes have been popped towards the MAC core, the FIFO controller will release that space to allow the DMA to push in more data. This means that the retransmission is not possible if this threshold (96 bytes) is crossed or when the MAC core indicates a late collision event.

Transmit FIFO flush operation

The FTF bit (bit 20) in the EMAC_DMAOPM register is set to flush TXFIFO. The TXFIFO flush operation is immediate and the corresponding points are reset to their initial states even if the TXFIFO is in the process of transferring a frame to the MAC core. This operation will abort the current frame transmission and result in an underflow event in the EMAC. The status of such a frame is generated (TDES0 bits 1 and 13). The flush operation is completed when the application (DMA) has received all of the status words of the frames that were flushed. Then the FTF bit is cleared in the EMAC_DMAOPM register. In this case, TXFIFO is allowed to receive new frames from the application (DMA). All data that do not start with an SOF marker will be discarded after the flush operation.

Transmit status word and time stamp

At the EMAC controller has completed the transmission of the Ethernet frame, the transmit status is given to the application. If IEEE 1588 time stamp is enabled, a 64-bit time stamp, along with the transmit status, will be written to the transmit descriptor.

Transmit checksum offload

The most widespread use of Ethernet is to encapsulate TCP or UDP over IP datagrams, so the Ethernet controller has a transmit checksum feature that supports checksum calculation and insertion in the transmit path, and error detection in the receive path.

Note: This function is enabled only when the TXFIFO is configured as store-and-forward mode (that is, when the TSF is set in the EMAC_DMAOPM register). If the TXFIFO depth is less than the input Ethernet frame size, the checksum function is invalid and only the IPv4 header checksum is calculated and modified by the EMAC, even in store-and-forward mode.

See IETF specifications RFC791, RFC 793, RFC 768, RFC 792, RFC 2460 and RFC 4443 for IPv4, TCP, UDP, ICMP, IPv6 and ICMPv6 specifications.

IP header checksum

The checksum module will detect an IPv4 datagrams when the Ethernet frame's type field has the value of 0x0800 and the IP datagram's version field has the value of 0x4. The input frame's checksum field is ignored and replaced by the calculated value. IPv6 headers do not have a checksum field, thus the checksum module does not modify IPv6 header fields. The result of this IP header checksum calculation is indicated by the IP header error status bit (TDES0 bit 16). This status bit is set whenever the values of the Ethernet type field and the IP header's version filed are not consistent, or when the Ethernet frame does not have enough data. In other words, this bit is set when the following errors occurred.

- For IPv4 datagrams
  - The received Ethernet type is 0x0800, but the IP header's version field is not equal to 0x4
  - The IPv4 header length field has a value less than 0x5 (20 bytes)
  - The total frame length is less than the value given in the IPv4 header length field
- For IPv6 datagrams
  - The received Ethernet type is 0x86DD, but the IP header's version field is not equal to 0x6

— The frame ends before the IPv6 header (40 bytes) or extension header (including header length field) has been completely received. Even if the checksum module detects such an IP header error, it inserts an IPv4 header checksum if the Ethernet type field indicates an IPv4 payload.

TCP/UDP/ICMP checksum

The TCP/UDP/ICMP checksum determines whether the encapsulated data is TCP, UDP or ICMP by analyzing the IPv4 or IPv6 header (including extension headers).

Note: 1. For non-TCP, - UDP or - ICMP/ICMPv6 data, this checksum function is invalid and the frame data is not modified.

2. Fragmented IP frames (IPv4 or IPv6), IP frames with security features (such as an authentication header or encapsulated security data) and IPv6 frames with routing headers are not processed by the checksum.

The checksum is calculated for the TCP, UDP or ICMP data and inserted into its corresponding field in the header. It can work in the following two modes:

1. The TCP, UDP or ICMPv6 pseudo-header is not included in the checksum calculation and is assumed to be present in the input frame's checksum field. The checksum field is included in the checksum calculation, and then replaced by the final calculated checksum.
2. The checksum field is ignored, and the TCP, UDP or ICMPv6 pseudo-header data are included into the checksum calculation, and the checksum field is overwritten with the final calculated value.

The result of this operation is indicated by the checksum error status bit in the transmit status vector (TDES0 bit 12). The data checksum error status bit is set when either of the following is detected:

1. The frame has been forwarded to the MAC transmitter in store-and-forward mode without the end of the frame being written to the TXFIFO.

2. The packet ends before the number of bytes indicated by the data length field in the IP header is received.

When the packet is longer than the indicated data length, the bytes are ignored as stuff bytes, and no error is reported. When the first type of error is detected, the TCP, UDP or ICMP header is not modified. For the second type of error, the calculated checksum is still inserted into the corresponding header field.

EMAC frame reception

The MAC received frames are stored into the RXFIFO and sent out by the DMA using the AHB interface. There are two modes: Cut-through mode and store-and-forward mode.

In the default Cut-through mode, when the frame data length greater than the programmed threshold (configured with the RTC bit in the EMAC_DMAOPM register) or a full packet of data are received in the RXFIFO, the data are popped out and the DMA is notified of its availability. The DMA will keep popping out the data from the RXFIFO until the data transfer is completed. Upon completion of the EOF frame transfer, the status word is popped out and sent to the DMA controller.

In store-and-forward mode (configured by the RSF bit in the EMAC_DMAOPM register), a frame is read by the DMA only after being written completely into the RXFIFO.

If the EMAC core is configured to drop all error frames, behavior varies in different reception modes:

1. In store-and-forward mode, only the valid frames are read and forwarded to the application by the DMA.
2. In Cut-through mode, some error frames are not dropped because the error status is received at the end of the frame.

A reception operation is initiated when the EMAC detects an SFD on the MII. The MAC core strips the preamble and SFD before processing the frame. The header fields are checked for the filtering and the FCS field used to verify the CRC of the frame. The frame is dropped in the core if it failed the address filter.

If IEEE1588 time stamping is enabled, a snapshot of the system time is taken when any frame's SFD is detected on the MII. This time stamp is passed onto the application when the EMAC has completed the current frame.

If the received frame length/type field is less than 0x600 and if the EMAC is configured for the auto

CRC/pad stripping option, the EMAC sends the data of the frame to RXFIFO up to the count specified in the length/type field, then starts dropping bytes (including the FCS filed). If the length/type field is greater than 0x600, the MAC sends all received Ethernet frame data to RXFIFO, regardless of the value on the programmed auto-CRC strip option.

The EMAC watchdog timer is enabled by default, frames above 2048 bytes (DA + SA + LT + Data + pad + FCS) are cut off. This feature can be disabled by the WD bit in the EMAC_MACCTRL register. However, even if the watchdog timer is disabled, frame length greater than 16 KB are cut off and a watchdog timeout event is reported.

Receive checksum offload

Both IPv4 and IPv6 frames are detected for data integrity by setting the IPC bit in the EMAC_MACCTRL register. The EMAC identifies IPv4 or IPv6 frames by checking for the Ethernet type field value. The receive checksum offload calculates IPv4 header checksums and checks that they match the received IPv4 header checksums. The result of the header checksum is indicated by the bit 7 of the receive descriptor (RDES0). The IP header error bit is set either one of the following conditions:

1. Ethernet type field does not match the IP header version field
2. Received frames are less than the length indicated by the IPv4 header length field
3. IPv4 or IPv6 headers less than 20 bytes

The receive checksum offload also identifies a TCP, UDP or ICMP payload in the received IP datagrams (IPv4 or IPv6) and calculates the checksum of such payloads properly, as defined in the TCP, UDP or ICMP specifications. It includes the TCP/UDP/ICMPv6 pseudo-header bytes for checksum calculation and the result of CRC is indicated by the bit 0 in the receive descriptor (RDES0). This bit is set either one of the following conditions:

1. Received TCP, UDP or ICMP data length does not match that of the IP headers
2. The calculated checksum does not equal the value of the TCP, UDP or ICMP checksum field

Receive flow control

In Full-duplex mode, the MAC detects the receiving Pause frame and pauses the frame transmission according to the delay specified within the received Pause frame.

The ERF bit in the EMAC_MACFCTRL register to enable or disable Pause frame detection function. Once receive flow control is enabled, the EMAC will decode the Pause frames.

During the pause period, if another Pause frame is detected with a zero Pause time value, the MAC clears the PAUSE time and retransmits the data. If it is not a zero Pause time value, the pause time in the frame will be immediately loaded into the pause time counter.

Receive operation multiframe handling

Since the status is available immediately following the data, the RXFIFO is capable of storing any number of frames into it, as long as it is not full.

Receive status word

At the end of the Ethernet frame reception, the EMAC will send the receive status to the application (DMA). The detailed information of the receive status is given in the RDES0.

EMAC loopback mode

The EMAC supports loopback of transmitted frames onto its receiver. The loopback mode is very useful to debug the Ethernet communication. This feature is enabled by setting the LM bit in the EMAC_MACCTRL register.

Note that the loopback mode is applicable only to the MII interface.

EMAC frame counter

The MAC management counters (MMC) contain a set of registers for gathering statistics on the received and transmitted frames. These include the EMAC_MMCCTR, EMAC_MMCRI, EMAC_MMCRIM, EMAC_MMCTI and EMAC_MMCTIM registers.

If a frame transmission is aborted due to any of the following errors, this frame will not be counted:

1. No carrier/Loss of carrier
2. Jabber timeout
3. Late collision
4. Excessive collision

5. Excessive deferral
6. Frame overflow

If a frame reception is aborted due to any of the following errors, this frame will not be counted:

1. CRC error
2. Runt frame (shorter than 64 bytes)
3. Alignment error
4. Length error (length field value does not match the received frame length)
5. Out of range (frame length beyond the maximum size, untagged frame maximum size=1518 bytes, tagged frame maximum size=1522 bytes)
6. MII_RXER input error

26.2.3 Ethernet frame transmission and reception using DMA

The transmission and reception of the Ethernet frames are scheduled through DMA.

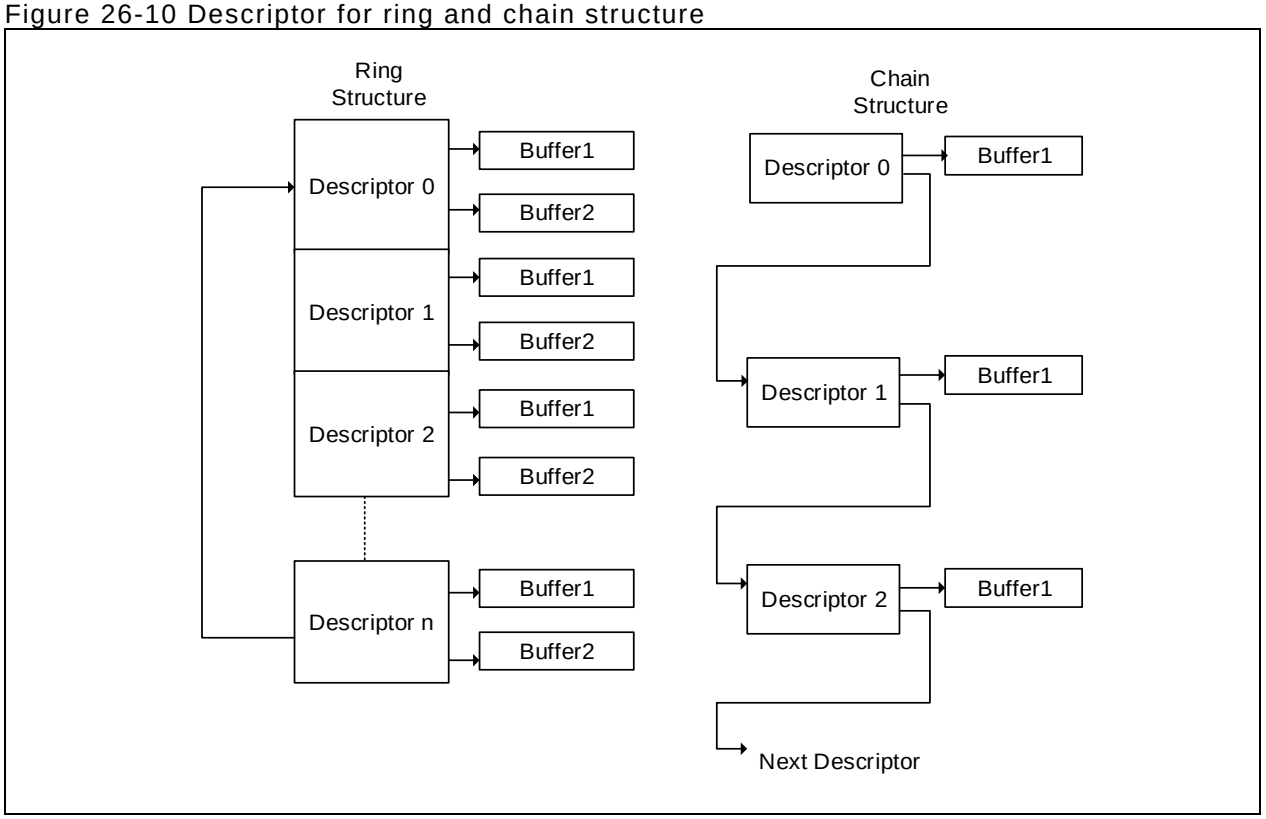
For transmission, the DMA reads out the Ethernet frames from the user system buffer (such as SRAM) via the AHB master interface and forwards them to the TXFIFO. The EMAC core then transfers the data frames in the TXFIFO to the MII/RMII interface.

For reception, the EMAC core sends the received Ethernet frames from the MII/RMII interface to the RXFIFO so that the DMA controller reads out the Ethernet frames from the RXFIFO and transfers them to the user data buffer (SRAM) via the AHB master interface.

DMA control and status register and descriptor table are used to manage the whole transmission and reception process, which has its respective descriptor list. The descriptor list is usually stored in the system buffer area (SRAM). When the transmission and reception is enabled, the DMA polls the descriptor table through the transmit and receive poll register to start the transmission and reception process. The base address the descriptor list for transmission and reception is stored into the transmit descriptor list register and receive descriptor list register.

There are two descriptor structures: ring structure and chain structure. In a ring structure, each descriptor may point to two buffers. In a chain structure (TDES0[20]=1 is configured for transmission, but RDES1[14]=1 for reception), each descriptor points to the only one buffer. The contents of the TDES3 and RDES3 for the current frames refer to the next descriptor address for transmission and reception.

Figure 26-10 Descriptor for ring and chain structure



DMA AHB host burst access

The DMA executes a fixed-length burst access on the AHB master interface if the FB bit is set in the EMAC_DMABM register. The maximum burst length is defined by the PBL filed (bit [13: 8] in the EMAC_DMABM register). The receive and transmit descriptors are always accessed in the maximum possible burst size (limited by PBL) for the 16 bytes to read.

The DMA provides the start address and the number of transfers to the AHB master interface before starting one transfer.

Note that one of the following conditions must be respected for transmission:

1. TXFIFO space is greater than the programmed burst size
2. The number of bytes before the end of a frame is less than the burst size and the TXFIFO can accommodate these bytes

One of the following conditions must be respected for reception:

1. The data available in the RXFIO is greater than the programmed burst size
2. The number of bytes before the end of a frame is less than the burst size and the end of the frame is detected in the RXFIFO

AHB host data alignment

The DMA always initiates transfers with address aligned to the bus width. But the start address of the buffers can aligned to any of the four bytes.

- Example of buffer read: If the transmit buffer address is 0x2000 0AA3, and 15 bytes are to be transferred, then the DMA will read five words (32 bits) from the address 0x2000 0AA0, but when transferring data to the TXFIFO, the first three bytes and the last two bytes will be ignored. The DMA always ensures that is transfer a 32-bit data to the TXFIFO, unless it is the end of the frame.
- Example of buffer write: If the receive buffer address is 0x2000 0BB2 and 16 bytes are to be transferred, the DMA will read five 32-bit data from the address 0x2000 0BB0. But the first two bytes and the last two bytes are dummy data.

Buffer size calculation

For transmission, software needs to calculate the buffer size. The TXDMA transfers the exact number of bytes programmed by buffer size field in the TDES1 to the EMAC core. If the FS bit is set in the TDES0, the DMA marks the first transfer from the buffer as the start of frame. If the LS bit is set in the TDES0, the DMA marks the last transfer from the buffer as the end of frame.

During a frame reception, if the receive buffer address is word-aligned, the valid length of the buffer refers to the value programmed in the RDES1 if the receive buffer address is not word-aligned, the valid length of the buffer is less than the value configured in the RDES1. The valid length value of the buffer is the value indicated by the RDES1 minus the lower two bit value of the buffer address. For example, if the total buffer size is 1024 bytes and the buffer address is 0x2000 0001, the lower 2-bit value of the address is 0x01, then the valid buffer size is 1023 bytes.

The FS bit is set by the DMA controller when an SOF is received. The LS is set when an EOF is received. If the receive buffer length field is big enough to accommodate a full frame, then both the FS and LS bits will be set in the same descriptor. The actual length of the received frame is indicated by the FL bit in the RDES0.

DMA arbiter

Two types of arbitrations are used for the arbitration between transmit and receive controller: round-bin, and fixed-priority. When round-robin is selected (DA bit is set to 0 in the EMAC_DMABM register), the arbiter allocates the databus according to the ratio set by the PR bit in the EMAC_DMABM register, when both transmit and RXDMA request access to the AHB bus simultaneously. When the DA bit is set to 1, the RXDAM always has priority over the TXDMA for data access.

Error response to DMA

If an error response is received during DMA transfer, then the DMA stops all operations and updates the error bit and the fatal bus error bit in the EMAC_DMASTS register. The DMA can resume operation after software or hardware resets the Ethernet peripherals and re-initiates the DMA.

DMA initialization

1. Configure AT32F437xx bus access parameters in the EMAC_DMABM register.
2. Mask unnecessary interrupt sources in the EMAC_DMAIE register.
3. The application generates the transmit and receive descriptor lists. Then it writes the start addresses of the descriptor lists to both the EMAC_DMARDLADDR and EMAC_DMATDLADDR registers.
4. Configure address filtering registers.
5. Configure the EMAC_MACCTRL register to enable the transmit and receive operating modes Program the PS and DM bits according to the auto-negotiation result
6. Set the bit 13 and bit 1 in the EMAC_DMAOPM register to enable transmission and reception.
7. The transmit and receive controllers begin reading descriptors from the corresponding descriptor lists to process receive and transmit operations.

TXDMA operation: non-OSF mode

The TXDMA proceeds as follows, in default mode:

1. The application sets up the Ethernet frame data buffer and the transmit descriptor (TDES0-TDES3), and sets the OWN bit (TDES0[31])
2. Once the SSTC is set (EMAC_DMAOPM bit [13]), the DMA enables transmission.
3. The DMA polls the transmit descriptor to get a frame to be transmitted. If the DMA detects a descriptor that is being owned by the CPU or if an error occurs, transmission is suspended and suspend state is entered, and both the transmit buffer unavailable (EMAC_DMASTS bit 2) and normal interrupt summary bit (EMAC_DMASTS bit 16) are set. The transmit controller jumps to Step 8.
4. DMA fetches the data from the AT32F437xx memory based on the transmit descriptor indication and transfers the data to the TXFIFO.
5. If the Ethernet frame is stored in multiple descriptors with different descriptor, the DMA will close the intermediate descriptor and fetch the next descriptor. Steps 3, 4 and 5 are repeated until the end of frame data is transferred.
6. When the frame transmission is complete, if IEEE1588 time stamping was enabled for the frame (as indicated in the transmit status), the time stamp value is written to the transmit descriptor (TDES2 and TDES3) that contains the end-of-frame buffer while the transmit status information is sent to the TDES0. Then the OWN bit is cleared, and the current descriptor is disabled. If time stamping was not enabled for the frame, the DMA only updates the status information to the TDES0 without recording the time stamping
7. When the frame transmission is complete, if the Interrupt on Completion (TDES0[30]) is set, then the transmit interrupt bit (EMAC_DMASTS bit [0]) will be set. The DMA then returns to Step 3 and is ready for the next transmission.
8. In the suspend state, the DAM tries to re-fetch the descriptor (back to Step 3) when it receives a transmit poll request and the overflow interrupt flag bit is cleared.

TXDMA operation: OSF mode

In this mode, the OSF bit is set (EMAC_DMAOPM bit 2=1). When the current data frame transmission is complete, the DMA immediately polls the transmit descriptor for the second frame without the need of waiting for the status information is written.

1. The DMA operates according to steps 1-6 of the TXDMA.
2. The DMA fetches the next descriptor without waiting for the status information update of the last description for the previous frame.
3. If the DMA owns the descriptor, then it will decode the transmit buffer address. If the DMA does not own the descriptor, then it will enter into suspend state and jump to Step 7.
4. The DMA fetches the transmit frame data from the AT32F437xx memory and transfer the frame until the end of frame is transferred. If the frame is split among multiple buffers, the DMA will close the intermediate descriptors.
5. The DMA waits for the transmit status and time stamp of the previous frame. After the status information is available, the DMA writes the time stamp to TDES2 and TDES3 if such time stamp

is captured. The DMA then clears the OWN bit and closes the descriptor. If the time stamping was not enabled for the frame, the DMA will not alter the contents of TDES2 and TDES3.

6. If enabled, the transmit interrupt bit is set. The DMA fetches the next descriptor when the status information is normal, and jumps to Step 3. If the previous transmit status shows an underflow error, the DMA enters into suspend state and jumps to Step 7.
7. In suspend state, when the DMA receives a pending status information and time stamp, if the time stamping is enabled it will write the time stamp to TDES2 and TDES3, and writes the status to TDES0. It then sets relevant interrupt flag bits and returns to suspend state.
8. The DMA can exit suspend state and enter run state only after receiving a transmit poll request (EMAC_DMACTD register).

Transmit frame processing

Ethernet frames stored in the transmit buffer must contain destination address, source address, correct type/length field and valid data. As for whether to include CRC value, it depends on the transmit descriptor. If the transmit descriptor requires the EMAC core to disable CRC or pad insertion, the buffer must contain the CRC.

A frame can be stored in multiple buffers that are linked in chain structure. When the transmission starts, the TDES0 bit 28 must be set in the first descriptor, and then the data are transferred from the memory to the TXFIFO. If the TDES0 bit 29 is set, it indicates the last buffer of the frame. After the last buffer of the data has been completed, the DMA writes back the final status information to the TDES0. If the transmit complete interrupt bit (TDES0[30]) is set, the transmit interrupt bit (EMAC_DMASTS bit 0) is set, the next descriptor is fetched, and the above steps are repeated. Actual frame transmission depends on whether the store-and-forward mode or threshold mode is selected. The descriptor is disabled (TDES0[31] is cleared) when the DMA finishes frame transmission.

Transmit polling suspend

Transmit polling can be suspended by one of the following conditions:

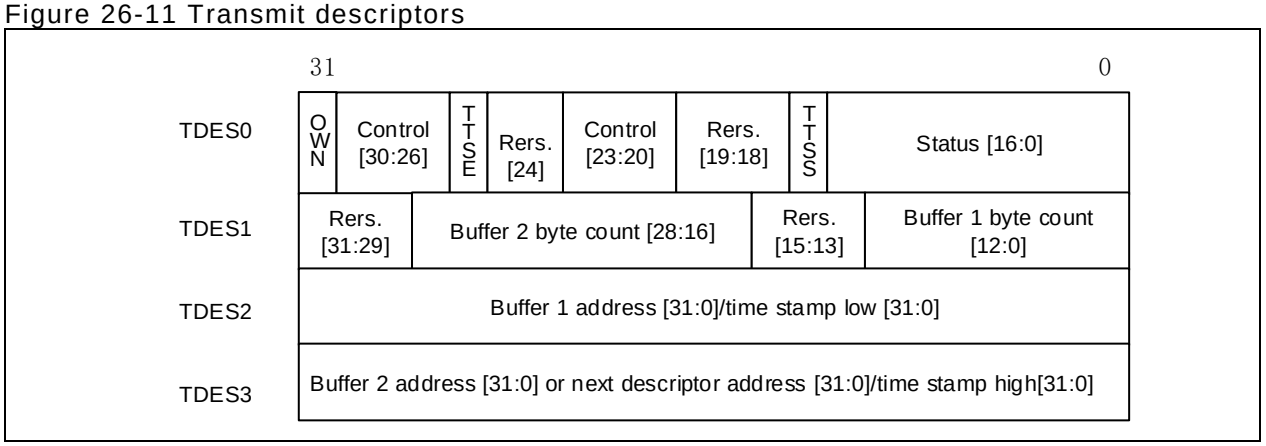
The DMA detects a descriptor owned by the CPU (TDES0[31]=0), and the DMA enters suspend state.

A frame transmission is aborted when an underflow is detected. The abnormal interrupt summary bit (EMAC_DMASTS bit 15) and transmit data underflow bit (EMAC_DMASTS bit 5) are set, and the appropriate error bit is set in the TDES0.

TXDMA descriptors

The descriptor structure consists of four 32-bit words. The bit definitions of TDES0, TDES1, TDES2 and TDES3 are as shown below:

Figure 26-11 Transmit descriptors



TDES0: Transmit descriptor word0

The software must configure the control bits [30: 26]+ [23: 20] and the OWN bit during descriptor initialization. When the DMA updates or writes the descriptor, it clears all the control bits and OWN bit, and report only the status bits.

Bit	Name	Type	Description
Bit 31	OWN	rw	Own bit 0: The descriptor is owned by the CPU 1: The descriptor is owned by the DMA This bit is cleared by the DMA when the DMA completes the frame transmission or

			when all the data in the buffer are read completely. The own bit of the frame's first descriptor can be set only after subsequent descriptors for the same frame have been set.
Bit 30	IC	rw	Interrupt on completion When set, this bit sets the transmit interrupt bit (EMAC_DMASTS bit [0]) after the present frame has been transmitted. This bit is valid only when the LS bit is set.
Bit 29	LS	rw	Last segment When set, this bit indicates that the buffer contains the last segment of the frame. When this bit is set, neither TBS1 nor TBS2 can be cleared in the TDES1.
Bit 28	FS	rw	First segment When set, this bit indicates that the buffer contains the first segment of the frame.
Bit 27	DC	rw	Disable CRC When set, the MAC does not append a CRC field to the end of the transmitted frame. This bit is valid only when the FS bit is set (TDES0[28]=1).
Bit 26	DP	rw	Disable pad 0: The MAC automatically adds padding to a frame shorter than 64 bytes, and the CRC field is added despite the state of the DC (TDES0[27]). This bit is valid only when the FS bit is set (TDES0[28]). 1: The MAC does not automatically add padding to a frame shorter than 64 bytes.
Bit 25	TTSE	rw	Transmit time stamp enable When this bit is set, the IEEE1588 hardware time stamp is activated for the transmit frame described the descriptor. This bit is valid only when both the TSE (EMAC_PTPTSCTRL[0]) and FS bits (TDES0[28]) are set.
Bit 24	Reserved	resd	Kept at its default value.
Bit 23: 22	CIC	rw	Checksum insertion control These two bits control the checksum calculation and insertion, as shown below: 00: Checksum insertion disabled 01: Only IP header checksum calculation and insertion are enabled 10: IP header checksum and data checksum calculation and insertion are enabled, but pseudo-header checksum is not calculated. 11: IP header checksum and data checksum calculation and insertion are enabled, and pseudo-header checksum is calculated.
Bit 21	TER	rw	Transmit end of ring When set, it indicates that the descriptor list reached its final descriptor. The DMA returns to the start address of the list, creating a descriptor ring.
Bit 20	TCH	rw	Second address chained When set, it indicates that the TBS2 bit in the TDES1 refers to the second descriptor address rather than the second buffer address. TDES0[21] takes precedence over TDES0[20]. This bit is valid only when the TDES0[28] is set.
Bit 19: 18	Reserved	resd	Kept at its default value.
Bit 17	TTSS	rw	Transmit time stamp status This bit is used as a status bit to indicate that a time stamp is captured for the described transmit frame. When this bit is set, it indicates the time stamp of the transmitted frame described by the descriptor has been captured, which are stored in the TDES2 and TDES3. This bit is valid only when the LS bit is set (TDES0[29]).
Bit 16	IHE	rw	IP header error When set, it indicates that the MAC transmitter detected an error in the IP data packet header. For IPv4 frames, the MAC checks whether the length field in the IPv4 datagram does match the number of he received IPv4 data. If there is a mismatch, an error is given. For IPv6 frames, an error is reported when the header length is not 40 bytes. Furthermore, the length/type field value for an IPv4 or IPv6 frame must match the IP header version. For IPv4 frame, an error is reported and this bit is set if the header length field value is shorter than 0x5.
Bit 15	ES	rw	Error summary This bit indicates the logical OR of the following bits: TDES0[14]: Jabber timeout TDES0[13]: Frame flush TDES0[11]: Loss of carrier TDES0[10]: No carrier TDES0[9]: Late collision TDES0[8]: Excessive collision TDES0[2]: Excessive deferral TDES0[1]: Underflow error TDES0[16]: IP header error

			TDES0[12]: IP data error
Bit 14	JT	rw	Jabber timeout When set, this bit indicates that the MAC transmitter has experienced a jabber timeout. This bit is set only when the JAD bit is not set in the EMAC_MACCTRL register.
Bit 13	FF	rw	Frame flushed When set, this bit indicates that the DMA or MTL flushed the frame in the FIFO due to a flush command given by the CPU.
Bit 12	IPE	rw	IP payload error When set, this bit indicates that the MAC transmitter detected an error in the TCP, UDP or ICMP. The transmitter compares the length field received in the IPv4 or IPv6 with the actual number of TCP, UDP or ICMP bytes. This bit is set as an error warning if there is a match.
Bit 11	LOC	rw	Loss of carrier When set, this bit indicates that a loss of carrier occurred during a frame transmission (The MII_CRS signal is active for one or more transmit clock periods). This bit is valid only for the frame transmitted without collision while the MC operates in half-duplex mode.
Bit 10	NC	rw	No carrier When set, this bit indicates that the carrier sense signal from the PHY was not set during a frame transmission.
Bit 9	LC	rw	Late collision When set, this bit indicates that frame transmission was aborted due to a collision detected after the collision window (64 byte times, including preamble, in MII mode). This bit is invalid if the underflow error bit is set.
Bit 8	EC	rw	Excessive collision When set, this bit indicates that the frame transmission was aborted after 16 consecutive collisions while attempting to transmit the current frame. If the RD bit (Retry disabled) bit in the EMAC_MACCTRL register is set, then this bit is set after the first collision, and the transmission of the frame is aborted.
Bit 7	VF	rw	VLAN frame When set, this bit indicates that the transmitted frame is a VLAN-type frame.
Bit 6: 3	CC	rw	Collision count This field indicates the number of collisions experienced before the frame was transmitted. This bit is invalid when the EC bit is set (TDES0[8]). It is valid only in half-duplex mode.
Bit 2	ED	rw	Excessive deferral When set, this bit indicates that the transmission ended because of excessive deferral of over 24288 bit times when the DC bit is set in the EMAC_MACCTRL register.
Bit 1	UF	rw	Underflow error When set, this bit indicates that the MAC stopped the frame transmission because data arrived late from the system memory to the MAC. Underflow error indicates that the DMA encountered an empty transmit buffer while transmitting the frame. The transmission process enters the suspend state and sets bot the bit 5 (TU) in the EMAC_DMASTS register and the transmit interrupt bit (bit 0 in the EMAC_DMASTS register).
Bit 0	DB	rw	Deferred bit When set, this bit indicates that the MAC defers frame transmission because of the presence of the carrier. This bit is valid only in half-duplex mode.

TDES1: Transmit descriptor word1

Bit	Name	Type	Description
Bit 31: 29	Reserved	resd	Kept at its default value.
Bit 28: 16	TBS2	rw	Transmit buffer 2 size This field indicates the second data buffer size in bytes. When the TDES0[20] bit is set, this field indicates the second descriptor address.
Bit 15: 13	Reserved	resd	Kept at its default value.
Bit 12: 0	TBS1	rw	Transmit buffer 1 size This field indicates the first data buffer size in bytes. If the field is 0, the DMA ignores this buffer and uses buffer 2 or the next buffer, depending on the TDES0[20] bit.

TDES2: Transmit descriptor word2

TDES2 contains the address pointer to the first buffer of the descriptor or it contains the lower 32-bit time stamp data.

Bit	Name	Type	Description
Bit 31: 0	TBAP1/TTSL	rw	Transmit buffer 1 address pointer / Transmit frame time stamp low This field has two functions: 1: The application indicates to the DMA the location of the Ethernet data in system memory. 2: After all data are transferred, the DMA can use these bits to store the time stamp of the transmit frame.
	TBAP1		When the current descriptor is owned by the DMA, these bits indicate the physical address of the buffer 1.
	TTSL		Before it releases the descriptor to the CPU, the DMA writes the 32 least significant bits of the time stamp capture for the corresponding transmit frame to this field. This field has the time stamp only when the TTSE bit in the TDES0 and the LS bit for the frame are set.

TDES3: Transmit descriptor word3

TDES3 contains the address pointer to the second buffer of the descriptor or the next descriptor, or it contains time stamp data.

Bit	Name	Type	Description
Bit 31: 0	TBAP2/TTSH	rw	Transmit buffer 2 address pointer (Next descriptor address) / Transmit frame time stamp high This field has two functions: 1: The application indicates to the DMA the location of the Ethernet data in system memory. 2: After all data are transferred, the DMA can use these bits to store the 32 most significant bits of the time stamp for the frame.
	TBAP2		When the current descriptor is owned by the DMA, these bits indicate the physical address of buffer2 if a descriptor ring structure is used. If a descriptor chain structure is used, these bits indicate the physical address of the next descriptor.
	TTSH		Transmit frame time stamp high The DMA updates these bits with the 32 most significant bits of the time stamp captured for the corresponding frame. This field has the time stamp only when the TTSE bit in the TDES0 and the LS bit for the frame are set.

RXDMA configuration

1. The application sets up receive descriptors (RDES0~RDES3), sets the OWN bit and then releases the descriptors to the DMA.
2. When the SSR bit (EMAC_DMAOPM[1]) is set, the DMA enters run state and attempts to acquire receive descriptors. If the fetched descriptor is not free (owned by the CPU), the DMA enters suspend state and jumps to Step 9.
3. The DMA decodes the receive buffer address from the acquired receive descriptors.
4. The DMA writes the frame data in the RXFIFO to the receive buffer.
5. When the buffer is full or the frame transfer ends, the receive controller will fetch the next descriptor from the descriptor queue.
6. If the current frame transfer is complete, the DMA jumps to Step 7. If the OWN bit of the next receive descriptor is cleared while the current frame is not complete (EOF is not received), when the frame flushing function is enabled, the DMA sets the descriptor error bit in the RDES0, closes the current descriptor (OWN=0) and sets the LS bit in the RDES1, and then jumps to Step 8 (Note that the LS bit in the RDES1 will not be set if the frame flushing feature is disabled). When the OWN bit of the next descriptor is set while the current frame transfer is not complete, the DMA then closes the current descriptor, marks it as intermediate and jumps to Step 4.
7. If IEEE1588 time stamp is enabled, the DMA writes the time stamp to the current descriptor's RDES2 and RDES3 while it writes the status word to the RDES0, with the OWN bit cleared and the LS bit set.

8. The receive controller checks the latest receive descriptors, if the DMA owns the descriptor, the receive controller will return to Step 4. If the CPU owns the descriptor, the RXDMA will enter suspend state and set the receive buffer unavailable bit, and the controller will flush the received frames if the receive frame flushing feature is enabled.
9. The DMA exits the suspend state when a receive poll demand is received or the start of the next frame is available in the receive FIFO. The receive controller fetches the next descriptor and jumps to Step 2.

Receive descriptor acquisition

The RXDMA always attempts to acquire another descriptor. Receive descriptor is attempted if any of the following conditions is satisfied:

- The DMA enters the run state (SSR=1 in the EMAC_DMAOPM).
- The buffer of the current descriptor is full before a full or part of the frame being transferred.
- The controller has completed the current frame reception, but the current receive descriptor has not yet been closed.
- A new frame is received but the receive process is suspended because the descriptor is owned by the CPU.
- A receive poll demand received.

Receive frame processing

The MII/RMI interface receives the receive frame and writes it to the RXFIFO. When a programmed threshold is reached, the RXDMA begins transferring the frame data to the receive buffer pointed to by the current descriptor. The DMA sets the LS bit in the RDES0 to indicate that this is the first segment of the frame in the buffer. The descriptors are released by clearing the OWN bit when the data buffer fills up or at the end of the frame reception. If the current descriptor buffer is enough to accommodate the complete frame, the current description RDES0 LS and FS bits are set.

The DMA fetches the next descriptor, sets the LS bit in the previous descriptor, writes the receive status word to the previous descriptor and then closes the previous descriptor. Then the DMA sets the receive interrupt bit (EMAC_DMASTS[6]). The same process repeats unless the DMA finds a descriptor as being owned by the CPU. If this occurs, the receive process sets the receive buffer unavailable bit (EMAC_DMASTS[7]) and then enters the suspend state. Before the suspend state is being entered, the current pointer value in the descriptor list is retained, which is used as the start address of a descriptor after exiting the suspend state.

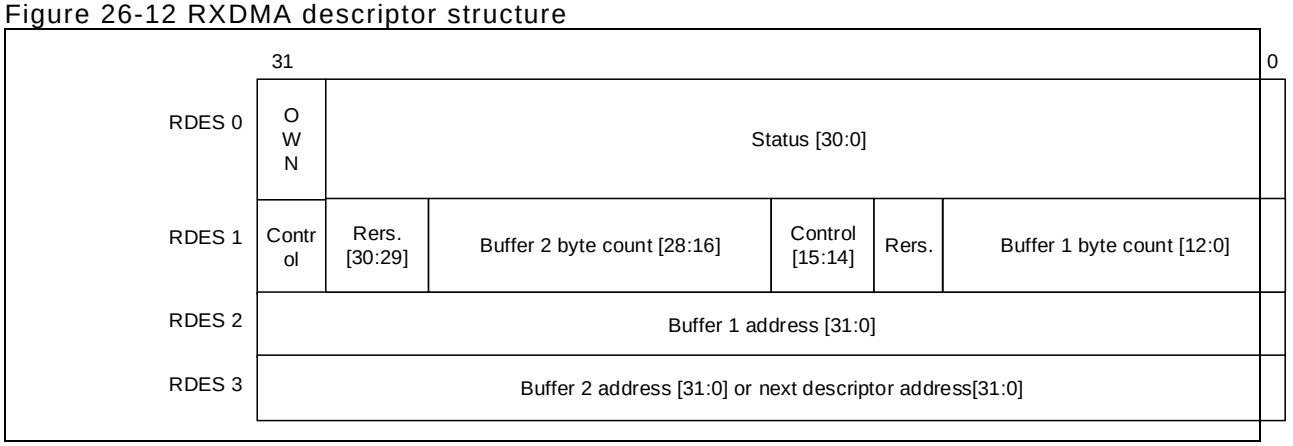
Receive process suspended

If a new receive frame arrives in the RXFIFO while the RXDMA is in suspend state, the DMA re-fetches the current descriptor in the AT32F437xx memory. If the OWN bit of the fetched descriptor is set, the receive process re-enters the run state. If the OWN bit is cleared, the DMA discards the current frame at the top of the RXFIFO and increments the missed frame counter. If more than one frame is stored in the RXFIFO, the process repeats. The flushing or discarding of the frame at the top of the receive FIFO can be avoided by setting the bit 24 (DFRF bit) in the EMAC_DMAOPM register. In this case, the receive process sets the receive buffer unavailable bit and returns to the suspend state.

RXDMA descriptors

The descriptor structure consists of four 32-bit words: RDES0, RDES1, RDES2 and RDES3.

Figure 26-12 RXDMA descriptor structure



RDES0: Receive descriptor word0

RDES0 contains the receive frame state, the frame length and the descriptor ownership information.

Bit	Name	Type	Description
Bit 31	OWN	rw	Own bit 0: The descriptor is owned by the CPU 1: The descriptor is owned by the DMA This bit is cleared by the DMA when the DMA completes the frame transmission or when the buffers associated with this descriptor are full. The descriptor is released to the CPU.
Bit 30	AFM	rw	Destination address filter fail When set, this bit indicates that a frame failed the DA filter in the MAC core.
Bit 29: 16	FL	rw	Frame length These bits indicate the byte length of the received frame that was transferred to the system memory by the DMA. This field is valid only when the LS bit (RESD0[8]) is set and descriptor error bit is cleared (RDES0[14). If the LS bit is cleared, this field indicates the accumulated number of bytes that have been transferred to the system memory. Whether the frame length includes CRC depends on the bit 7 and bit 25 in the EMAC_MACCTRL register.
Bit 15	ES	rw	Error summary This bit indicates the logical OR of the following bits: RDES0[1]: CRC error RDES0[3]: Receive error RDES0[4]: Watchdog timeout RDES0[6]: Late collision RDES0[7]: Giant frame or IP checksum error RDES0[11]: Overflow error RDES0[14]: Descriptor error
Bit 14	DE	rw	Descriptor error When set, this bit indicates a frame truncation caused by a frame that does not fit with the current descriptor buffers, and that the DMA does not own the next descriptor. This bit is valid only when the LS bit (RDES0[8])) is set.
Bit 13	SAF	rw	Source address filter fail When set, this bit indicates that the received frame failed the SA filter in the MAC core.
Bit 12	LE	rw	Length error When set, this bit indicates that the actual length of the received frame does not match the value in the Ethernet length/type field. This bit is valid only when the RDES0[5] bit is cleared. It is invalid when a CRC error occurs.
Bit 11	OE	rw	Overflow error When set, this bit indicates that the received frame was damaged due to receive FIFO overflow.
Bit 10	VLAN	rw	VLAN tag When set, this bit indicates that the frame pointed to by the current descriptor is marked as a VLAN frame by the MAC.
Bit 9	FS	rw	First descriptor When set, this bit indicates that this descriptor contains the first buffer of the frame. If the size of the first buffer is 0, the second buffer contains the beginning of the frame. If the size of the second buffer is also 0, the buffer of the next descriptor

			contains the beginning of the frame.
Bit 8	LS	rw	Last descriptor When set, this bit indicates that the buffers pointed to by this descriptor are the last buffers of the frame.
Bit 7	IPHCE	rw	IPv header checksum error When set, this bit indicates an error in the IPv4 or IPv6 header. This error can be due to mismatched Ethernet type filed and IP version field, IPv4 header checksum error or an Ethernet frame lacking the desired number of IP header bytes.
Bit 6	LC	rw	Late collision When set, this bit indicates that a late collision has occurred while receiving the frame in half-duplex mode.
Bit 5	FT	rw	Frame type When set, this bit indicates that the received frame is an Ethernet frame (the LT field is greater than or equal to 1536). When this bit is cleared, it indicates that the received frame is an IEEE802.3 frame. This bit is invalid when the received frame is a runt frame shorter than 14 bytes.
Bit 4	RWT	rw	Receive watchdog timeout When set, this bit indicates the receive watchdog timeout has occurred while receiving the current frame. The current frame is truncated after the watchdog timeout.
Bit 3	RE	rw	Receive error When set, this bit indicates that the RX_ER signal is valid while the RX_DV is valid during frame reception.
Bit 2	DE	rw	Dribble bit error When set, this bit indicates that the received frame is not an integer multiple of bytes (odd nibbles) This bit is valid only in MII mode.
Bit 1	CE	rw	CRC error When set, this bit indicates a CRC error occurred on the received frame. This bit is valid only when the LS bit is set.
Bit 0	PCE	rw	Payload checksum error When set, this bit indicates that the TCP, UDP or ICMP checksum calculated by the MAC core does not match the received TCP, UDP or ICMP checksum field. This bit is also set when the received payload size does not match the length filed value of the IPv4 or IPv6 datagram in the received Ethernet frame.

Table 26-7 shows the definitions of bit 5, 7 and 0.

Table 26-7 Receive descriptor 0

Bit 5: Frame type	**Bit 7: Checksum error**	**Bit 0: Payload checksum error**	**Frame status**
0	0	0	IEEE802.3 type frame (length filed value is less than 0x0600)
1	0	0	IPv4/IPv6 type frame, no checksum error detected
1	0	1	IPv4/IPv6 type frame with payload checksum error detected (PCE bit)
1	1	0	IPv4/IPv6 type frame with an IP header checksum error detected (IPC CE bit)
1	1	1	IPv4/IPv6 type frame with both IP header and payload checksum errors detected
0	0	1	IPv4/IPv6 type frame with no IP header checksum error detected and the payload check bypassed due to an unsupported payload
0	1	1	A type frame is neither IPv4 nor IPv6 (the checksum offload bypasses checksum inspection)
0	1	0	Reserved

RDES1: Receive descriptor 1

Bit	Name	Type	Description
Bit 31	DIC	rw	Disable interrupt on completion When set, this bit prevents setting the Ethernet DMA status register's RECV bit (EMAC_DMASTS) for the received frame pointed to by this descriptor. As a result, this disables the interrupt triggered by the RECV bit. This bit is valid only when the RDES0[8] is set.
Bit 30: 29	Reserved	resd	Kept at its default value.
Bit 28: 16	RBS2	rw	Receive buffer 2 size This field indicate the receive buffer 2 size, in bytes. It is invalid if the RDES1[14 bit is set.
Bit 15	RER	rw	Receive end of ring When set, this bit indicates that the current descriptor is the last one in the descriptor list. The DMA returns to the base address to fetch the next descriptor, creating a descriptor ring.
Bit 14	RCH	rw	Second address chained When set, this bit indicates that the second address in the descriptor is the next descriptor address rather than the second buffer address. When this bit is set, RBS2(TDES1[28: 16]) value can be ignored. RDES0[15] takes precedence over RDES0[14].
Bit 13	Reserved	resd	Kept at its default value.
Bit 12: 0	RBS1	rw	Receive buffer 1 size This field indicates the receive buffer 1 size, in bytes. If this field is 0, the DMA ignores this buffer and uses buffer 2 or next descriptor depending on the RDES[14] value.

RDES2: Receive descriptor word2

RDES2 contains the address pointer to the first data buffer in the descriptor, or it contains time stamp data.

Bit	Name	Type	Description
Bit 31: 0	RBAP1/RTSL	rw	Receive buffer 1 address pointer /Receive frame time stamp low These bits have two functions. The application uses them to indicate to the DMA where to store the data in memory. After completing data reception, the DMA may use these bits to store frame time stamp.
	RBAP1		When this descriptor is owned by the DMA, these bits indicate the physical address of buffer 1.
	RTSL		Before it releases the descriptor, the DMA updates this field with the 32 least significant bits of the time stamp. The time stamp is written by the DMA only when the time stamp and LS bit are set (indicating that the last segment of the frame is stored in memory).

RDES3: Receive descriptor word3

RDES3 contains the address pointer to the second data buffer in the descriptor, or it contains time stamp data.

Bit	Name	Type	Description
Bit 31: 0	RBAP2/RTSH	rw	Receive buffer 2 address pointer (next descriptor address)/Receive frame time stamp high Bit [31:0] These bits have two functions. The application uses them to indicate to the DMA where to store the data in memory. After completing data reception, the DMA may use these bits to store frame time stamp.
	RBAP2		When this descriptor is owned by the DMA, these bits indicate the physical address of buffer 2 when a descriptor ring structure is used. If the second address chained bit (RDES0[14]) is set, these bits point to the physical address of the next descriptor while the next descriptor is present.
	RTSH		Before it releases the descriptor, the DMA updates this field with the 32 most significant bits of the time stamp. The time stamp is written by the DMA only when the time stamp and LS bit are set (indicating that the last segment of the frame is stored in memory).

26.2.4 Enter and wake up EMAC power-down mode

The EMAC enters power-off mode when the PD bit is enabled in the EMAC_MACPMTCTRLSTS register. In this mode, all received frames are dropped by the EMAC and they are not forwarded to the application.

PMT supports the reception of remote wakeup frames and AMD Magic Packet frames and uses them to wake up the EMAC from power-off mode. This is done by setting the ERWF and EMP bits in the EMAC_MACPMTCTRLSTS register.

Remote wakeup frame filter register

There are eight wakeup frame filter registers, each of which requires to be configured one by one. The desired values of the wakeup frame filter are loaded by sequentially loading eight times the wakeup frame filter register. The read operation is identical to the write operation. To read the eight values, the user has to read the wakeup frame filter register for consecutive eight times.

Figure 26-13 Wakeup frame filter register

Wkuppktfilter_reg0	Filter 0 Byte Mask							
Wkuppktfilter_reg1	Filter 1 Byte Mask							
Wkuppktfilter_reg2	Filter 2 Byte Mask							
Wkuppktfilter_reg3	Filter 3 Byte Mask							
Wkuppktfilter_reg4	RESD	Filter 3 Cmd	RESD	Filter 2 Cmd	RESD	Filter 1 Cmd	RESD	Filter 0 Cmd
Wkuppktfilter_reg5	Filter 3 Offset		Filter 2 Offset		Filter 1 Offset		Filter 0 Offset	
Wkuppktfilter_reg6	Filter 1 CRC-16				Filter 0 CRC-16			
Wkuppktfilter_reg7	Filter 3 CRC-16				Filter 2 CRC-16			

Filter i byte mask

This register defines which bytes of the filter i (i=0~3) are used to determine whether or not the frame is a wakeup frame. The bit 31 must be zero. The bit j[30: 0] is the byte mask. If the bit j is set, then filter i offset + j of the incoming frame will be processed by the CRC block, otherwise filter i offset + j is ignored.

Filter i command

This is a 4-bit command. Bit 3 defines the address type. When the bit is set, this feature applies to only multicast addresses. When the bit is cleared, this feature applies to only unicast addresses. Bit 2 and bit 1 are reserved. Bit 0 is the filter enable bit. This filter is disabled if the bit 0 is cleared.

Filter i offset

This is an 8-bit register that defines the offset for the filter i first byte to be examined by filter i. The minimum allowed is 12, which refers to the 13th byte of the frame (offset value 0 means the first byte of the frame)

Filter i CRC-16

This register contains the CRC_16 value calculated by the filter, and the byte mask value programmed in the wakeup frame filter register block.

Remote wakeup frame detection

This mode is enabled by setting the RRWF bit in the EMAC_MACPMTCTRLSTS register.

PMT supports four programmable filters. If the incoming frame passed the address filtering of the filter, and if the filter CRC_16 matches the examined incoming frame, then the wakeup frame is received. PMT is only responsible for checking length error, FCS error, Dribble bit error, MII error, collision and ensuring that the wakeup frame is not a runt frame.

When a remote wakeup frame is received, the EMAC will move from sleep mode to normal mode. At the

same time, the RRWF bit (bit 6) is set in the EMAC_MACPMTCTRLSTS register, indicating that a remote wakeup frame is received. If a remote wakeup interrupt is enabled, an interrupt will be generated when the PMT receives the remote wakeup frame.

Magic Packet detection

Magic Packet detection is enabled by setting the EMP bit in the EMAC_MACPMTCTRLSTS register. The Magic Packet frame contains a specific packet of information that is used to wakeup the stations on the LAN.

Magic Packet frame format: 6 bytes are all 1, followed by a MAC address repeating 16 times, for instance, MAC address of a device is 0x11aabb22cc33, then the Magic Packet used to wake up this frame is shown as follows:

Destination address source address ·····FFFF FFFF FFFF

11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33

11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33

11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33

11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33

··· CRC

In Sleep mode, the PMT will constantly detect each frame transferred to the station to determine whether or not the frame meets the Magic Packet format.

When the Magic Packet is received, the RMP bit will be updated in the EMAC_MACPMTCTRLSTS register. Upon the reception of Magic Packet, the PMT will generate an interrupt if enabled.

Precautions on system in DEEPSLEEP mode

In DEEPSLEEP mode, the Ethernet PMT block is still able to detect frames as long as the EXTI 19 is enabled. However, the RE bit has to be set in the EMAC_MACCTRL register because the EMAC needs to detect Magic Packet or a remote wakeup frame.

DEEPSLEEP and wakeup sequences are recommended as follows:

1. Disable the TXDMA and wait for all data transmission to complete. These transmissions can be checked by polling the TI bit in the EMAC_DMASTS register.
2. Disable the MAC transmitter and MAC receiver by clearing the TE and RE bits in the EMAC_MACCTRL register.
3. Poll the RI bit in the EMAC_DMASTS register, and wait for the RXDMA to real all the frames in the RXFIFO, and then disable the RXDMA.
4. Configure and enable the EXTI19 to generate either an event or an interrupt.
5. Enable Magic Packet/remote wakeup frame detection by setting the EMP/ERWF bit in the EMAC_MACPMTCTRLSTS register.
6. Enable power-down mode by setting the PD bit in the MAC_MACPMTCTRLSTS register.
7. Enable the EMAC receiver by setting the RE bit in the EMAC_MACCTRL register.
8. MCU enters DEEPSLEEP mode.
9. Upon receiving Magic Packet/a remote wakeup frame, the Ethernet exits the power-down mode.
10. Read the EMAC_MACPMTCTRLSTS register to clear RRWF/RMP, enable the EMAC transmit state machine, and the TXDMA and RXDMA.
11. Configure the MCU system clock.

26.2.5 IEEE1588 precision time protocol

The PTP is used to synchronize systems that include clocks of varying precision, resolution and stability (not limited to Ethernet). Refer to IEEE1588 standard for more information.

The PTP block captures the accurate time when a PTP packet is transmitted or received from Ethernet port, and the captured time is returned to the application.

Reference clock source

According to IEEE158 standard, the system requires a reference time in a 64-bit format as the current time record, with the upper 32 bits time information in seconds, and the lower 32 bits time information in nanoseconds.

The PTP reference clock is used to generate the system time and to capture time stamps. The frequency of this reference clock must be greater or equal to the resolution of time stamp counter. The synchronization accuracy between the master node and the slave node is around 100ns.

The time synchronization accuracy depends on the PTP reference clock input period, the frequency drift of the crystal oscillator and that of the synchronization procedure.

Transmission and reception of frames with PTP feature

When the bit 0 is set in the EMAC_PTPTSCTRL register and the bit 25 is also set in the TDES0 register, a frame's SFD is output on the MII, and then a time stamp is captured. The upper 32 bits and the lower 32 bits of the time stamp are stored in the TDES3 and TDES2, respectively so that the time stamp and transmit status work will be returned to the application all together.

When the bit 0 is set in the EMAC_PTPTSCTRL register and the bit 8 is also set in the EMAC_PTPTSCTRL register, the EMAC will capture the time stamp of all the received frames on the MII. The upper 32 bits and the lower 32 bits of the time stamp are stored in the RDES3 and RDES2, respectively so that the time stamp and receive status work will be returned to the application all together.

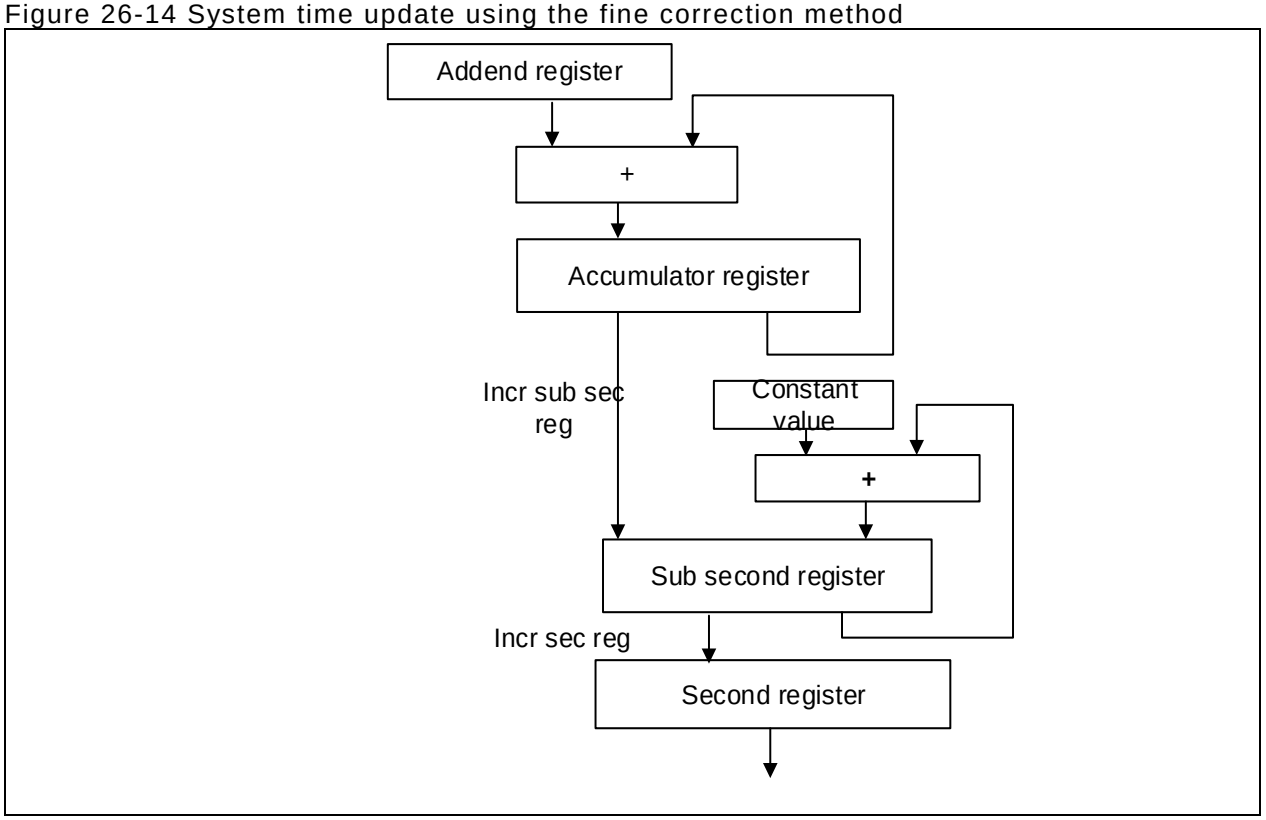
System time correction methods

The PTP input reference clock is the system clock SYSCLK, which is used to update the 64-bit time stamp of the Ethernet frames being transmitted or received. There are two correction methods: coarse and fine correction.

Coarse correction: the initial value/the offset value is written to the time stamp update registers (EMAC_PTPTSHUD and EMAC_ PTPTSLUD). When the TI bit is set in the EMAC_PTPTSCTRL register, an initialization process starts, and system clock counter is updated with the value in the time stamp update register. If the TU bit is set in the EMAC_PTPTSCTRL register, a correction process starts, and the time stamp update register value is uses as the offset value, and such offset value is added or subtracted from the system time.

Fine correction: the slave clock (reference clock) frequency drift with respect to the master clock (as defined in IEEE1588) is corrected over a period of time. An accumulator sums up the value of the addend register as shown in Figure 26-15.The pulse generated by the arithmetic carry of the accumulator is used to increment the system time counter. The addend register value depends on the system clock frequency. Both the accumulator and the addend are 32-bit registers.

Figure 26-14 System time update using the fine correction method



The subsecond register update frequency requires 50 MHz to achieve 20 ns accuracy for the system clock update circuit. Therefore, if the system clock frequency is 70 MHz, the ratio is calculated as 70/50=1.4. Thus the value written to the addend register is 2^{32}/1.4, which is equal to 0XB6DB6DB6.

The value in the addend register has to be updated according to the drift of the system clock frequency. If the subsecond register update frequency is 50 MHz, the constant value used to increment the subsecond register is $2^{31}/(50*10^6)$=43.

The software has to calculate the drift of the frequency by means of Sync, and to update the accumulator register accordingly. Initially, the slave clock addend register is programmed to be FreqCompensationValue0:

FreqCompensationValue0 = 2^{32}/frequency division ratio. If the MasterToSlaveDelay is assumed to be the same for consecutive Sync messages. The algorithm below mentioned must be applied. After a few Sync cycles, the frequency is locked. The slave clock can then determine an accurate MasterToSlaveDela value and synchronize the master with the salve clock using the new value. The algorithm is shown as follows:

When the master clock is $MasterSyncTime_n$, the master node sends the slave node a Sync message. The slave node receives this message when its local clock is $SlaveClockTime_n$, and computes $MasterClockTime_n$ as

$MasterClockTime_n = MasterSyncTime_n + MasterToSlaveDelay_n$

The master clock count for the current Sync cycle, $MasterClockCount_n$ is:

$MasterClockCount_n = MasterClockTime_n - MasterClockTime_{n-1}$, assuming that the MasterToSlaveDelay is the same for Sync cycles n Sync cycles n-1

The slave clock count for the current Sync cycle, $SlaveClockCount_n$ is:

$SlaveClockCount_n = SlaveClockTime_n - SlaveClockTime_{n-1}$

The difference between the master clock count and slave clock count for the current Sync cycle, $ClockDiffCount_n$ is

$ClockDiffCount_n = MasterClockCount_n - SlaveClockCount_n$

The frequency-ratio factor for a slave clock, $FreqScaleFactor_n$ is

$FreqScaleFactor_n = (MasterClockCount_n + ClockDiffCount_n) / SlaveClockCount_n$

The frequency compensation value for the addend register, $FreqCompensationValue_n$ is

$$FreqCompensationValue_n = FreqScaleFactor_n \times FreqCompensationValue_{n-1}$$

This algorithm comes with a self-correction feature. In theory, the frequency can be locked at a synchronized cycle. However, it may makes several cycles to synchronize the slave device.

System time initialization procedure

1. Mask the time stamp trigger interrupt by setting the bit 9 in the EMAC_MAIMR register.
2. Enable the time stamp by setting the bit 0 in the EMAC_PTPTSCTRL register.
3. Program the subsecond increment register based on the system time update precision.
4. If the fine correction method is being used, program the EMAC_PTPTSAD register, set the bit 5 in the EMAC_PTPTSCTRL register, poll the EMAC_PTPTSCTRL register until the bit 5 becomes 0. For the coarse correction method, skip this step and directly jump to step 6.
5. To select the fine correction method, program the bit 1 in the EMAC_PTPTSCTRL register.
6. Write the system time value to be configured to the EMAC_PTPTSHUD and EMAC_PTPTSLUD) registers.
7. Set the bit 2 in the EMAC_PTPTSCTRL register, and start initializing the time stamp and polling this bit until this bit becomes 0 (initialization complete).
8. The time stamp counter starts running as soon as the initialization is complete.
9. Enable the MAC transmitter and receiver for proper time stamping.

Programming steps for system time update using coarse correction method

1. Write the offset value (positive or negative) to the Ethernet PTP time stamp high update register (EMAC_PTPTSHUD) and the Ethernet PTP time stamp low update register (EMAC_PTPTSLUD).
2. Set the bit 3 (TU) in the Ethernet PTP time stamp control register (EMAC_PTPTSCTRL).
3. When the TU bit is cleared, the value in the time stamp update registers is added to or subtracted from the system time.

Programming steps for system time update using fine correction method

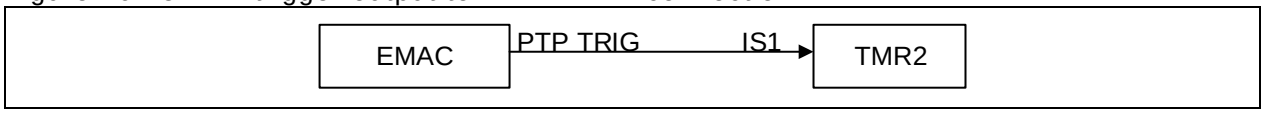
1. Use the algorithm explained in System time correction methods to calculate the value of the addend register.
2. Update the addend register.
3. Write the target value you want to the target time high and target time low registers, and clear the bit 9 in the Ethernet MAC interrupt mask register (EMAC_MAIMR) to activate the time stamp interrupt.
4. Set the bit 4 (TITE) in the Ethernet PTP time stamp control register (EMAC_PTPTSCTRL).
5. When this event generates an interrupt, read the EMAC_MAIMR register to clear the corresponding interrupt flag bits.
6. Reprogram the EMAC_PTPTSAD register with the old value and set the bit 5 in the EMAC_PTPTSCTRL register to update the addend register.

PTP trigger internal connection with TMR2

The EMAC provides a trigger interrupt when the system time is greater than the target time. Using an interrupt introduces an interrupt latency. To obtain an accurate interrupt latency time, a PTP will output a high signal to the TMR2 when the system time becomes greater than the target value. An accurate interrupt latency can be calculated since the clock of the timer and PTP reference clock are synchronous.

Set the bit [11:10]=01 in the TMR2_RMP register to enable the connection between the PTP trigger signal and the TMR2 IS1.

Figure 26-15 PTP trigger output to TMR2 ITR1 connection



PTP second pulse output signal

Refer to the EMAC_PTPPPSCR register descriptor for more information about PTP pulse second output. The following contents are based on the fact when the emac_pps_sel bit (bit 9) is cleared in the CRM_MISC2 register.

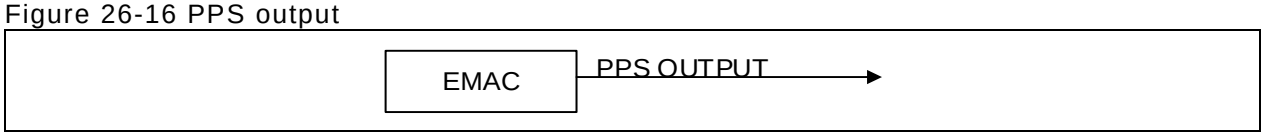
The PPS output frequency is 1 Hz by default, which can be configured to 2PPSFREQ Hz through the PPSFREQ[3: 0] in the EMAC_PTPPPSCR register.

If it is 1 Hz, the pulse width of the PPS is 125 ms when the binary rollover control is selected (TSSSR=0 in the EMAC_PTPTSCTRL register); the pulse width of the PPS is 100 ms when the digital rollover control is selected (TSSSR=1).

If it is 2 Hz or over, the duty cycle of the PPS output is 50% when the binary rollover control is selected.

Configure the PB5 multiplexed feature to enable PPS output feature.

Figure 26-16 PPS output



26.2.6 EMAC interrupts

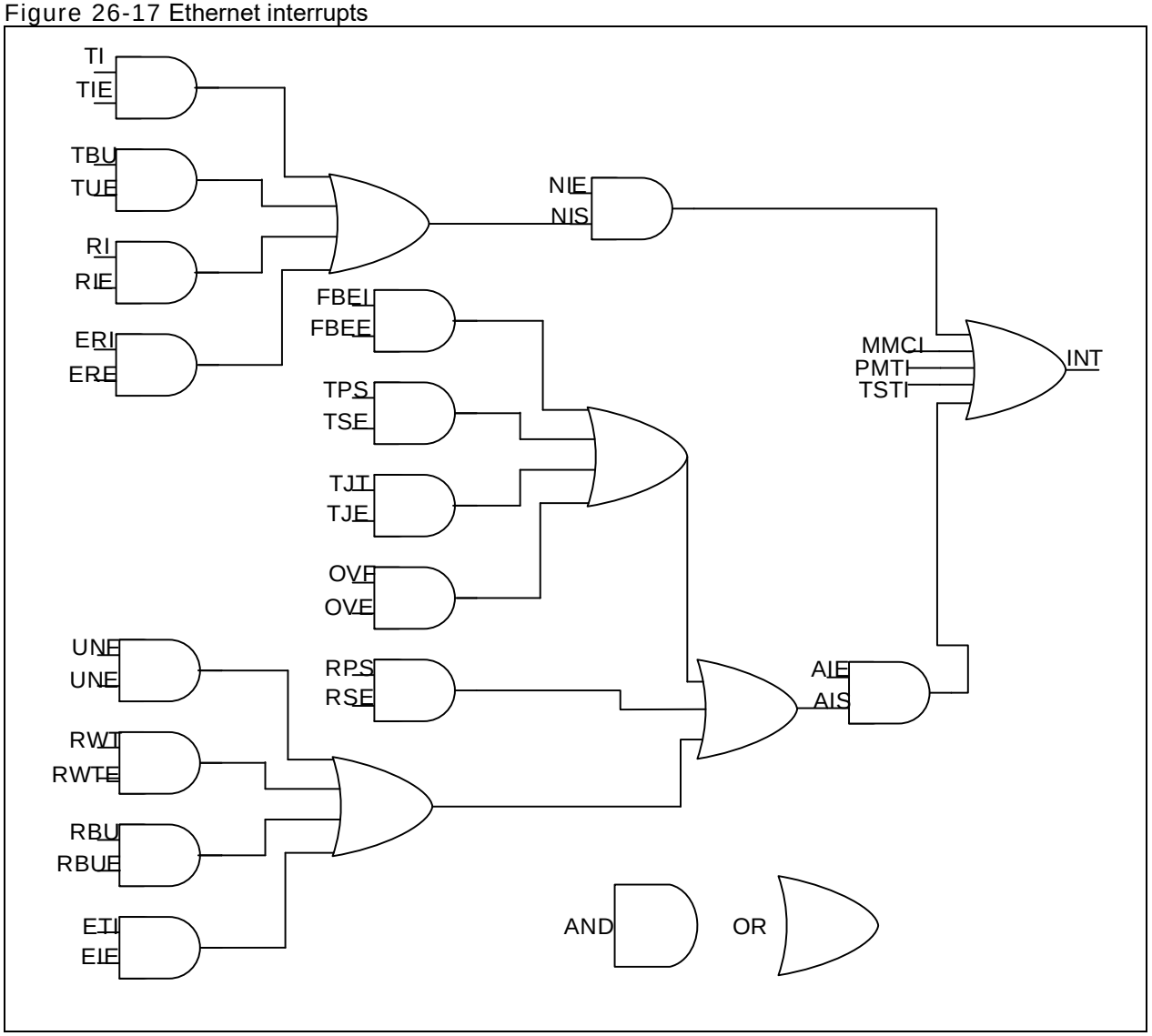
The EMAC has two interrupt vectors: one is used for normal Ethernet operations and the other for the Ethernet wakeup event (remote wakeup frame or Magic Packet detection) when it is mapped on EXINT line 19.

The first interrupt vector is used for interrupts generated by the MAC and the DMA.

The second interrupt vector is used for interrupts generated by the PMT block on wakeup events. It can cause the AT32F437xx to exit the low-power mode, and generate an interrupt.

When an Ethernet wakeup event is mapped on the EXINT line19 and the EMAC PMT interrupt is enabled and the EXINT line 19 interrupt, detected on its rising edge, is also enabled, both interrupts are generated at the same time.

Figure 26-17 Ethernet interrupts



26.3 EMAC registers

Table 26-8 shows the Ethernet register map and its reset values.

The peripheral registers can be accessed by bytes (8-bit), half words (16-bit) or words (32-bit).

Table 26-8 Ethernet register map and its reset values

Register name	Offset	Reset value
EMAC_MACCTRL	0x00	0x0000 8000
EMAC_MACFRMF	0x04	0x0000 0000
EMAC_MACHTH	0x08	0x0000 0000
EMAC_MACHTL	0x0C	0x0000 0000
EMAC_MACMIIADDR	0x10	0x0000 0000
EMAC_MACMIIDT	0x14	0x0000 0000
EMAC_MACFCTRL	0x18	0x0000 0000
EMAC_MACVLT	0x1C	0x0000 0000
EMAC_MACRWFF	0x28	0x0000 0000
EMAC_MACPMTCTRLSTS	0x2C	0x0000 0000
EMAC_MACISTS	0x38	0x0000 0000

EMAC_MAIMR	0x3C	0x0000 0000
EMAC_MACA0H	0x40	0x0010 FFFF
EMAC_MACA0L	0x44	0xFFFF FFFF
EMAC_MACA1H	0x48	0x0000 FFFF
EMAC_MACA1L	0x4C	0xFFFF FFFF
EMAC_MACA2H	0x50	0x0000 FFFF
EMAC_MACA2L	0x54	0xFFFF FFFF
EMAC_MACA3H	0x58	0x0000 FFFF
EMAC_MACA3L	0x5C	0xFFFF FFFF
EMAC_MMCCTRL	0x100	0x0000 0000
EMAC_MMCRI	0x104	0x0000 0000
EMAC_MMCTI	0x108	0x0000 0000
EMAC_MMCRIM	0x10C	0x0000 0000
EMAC_MMCTIM	0x110	0x0000 0000
EMAC_MMCTFSCC	0x14C	0x0000 0000
EMAC_MMCTFMSCC	0x150	0x0000 0000
EMAC_MMCTFCNT	0x168	0x0000 0000
EMAC_MMCRFCECNT	0x194	0x0000 0000
EMAC_MMCRFAECNT	0x198	0x0000 0000
EMAC_MMCRGUFCNT	0x1C4	0x0000 0000
EMAC_PTPTSCTRL	0x700	0x0000 2000
EMAC_PTPSSINC	0x704	0x0000 0000
EMAC_PTPTSH	0x708	0x0000 0000
EMAC_PTPTSL	0x70C	0x0000 0000
EMAC_PTPTSH	0x708	0x0000 0000
EMAC_PTPTSL	0x70C	0x0000 0000
EMAC_PTPTSHUD	0x710	0x0000 0000
EMAC_PTPTSLUD	0x714	0x0000 0000
EMAC_PTPTSAD	0x718	0x0000 0000
EMAC_PTPTTH	0x71C	0x0000 0000
EMAC_PTPTTL	0x720	0x0000 0000
EMAC_PTPTSSR	0x728	0x0000 0000
EMAC_PTPPPSCR	0x72c	0x0000 0000
EMAC_DMABM	0x1000	0x0002 0101
EMAC_DMATPD	0x1004	0x0000 0000
EMAC_DMARPD	0x1008	0x0000 0000
EMAC_DMARDLADDR	0x100C	0x0000 0000
EMAC_DMATDLADDR	0x1010	0x0000 0000
EMAC_DMASTS	0x1014	0x0000 0000
EMAC_DMAOPM	0x1018	0x0000 0000
EMAC_DMAIE	0x101C	0x0000 0000

EMAC_DMAMFBOCNT	0x1020	0x0000 0000
EMAC_DMACTD	0x1048	0x0000 0000
EMAC_DMACRD	0x104C	0x0000 0000
EMAC_DMACTBADDR	0x1050	0x0000 0000
EMAC_DMACRBADDR	0x1054	0x0000 0000

26.3.1 Ethernet MAC configuration register (EMAC_MACCTRL)

The Ethernet MAC configuration register defines the receive and transmit operation modes.

A delay greater than 4μs is required for two consecutive write accesses to this register.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23	WD	0x0	rw	Watchdog Disable When this bit is set, the MAC disables the watchdog timer on the receiver, and can receive frames of up to 16,384 bytes. When this bit is cleared, the MAC allows no more than 2048 bytes of the frames being received.
Bit 22	JD	0x0	rw	Jabber Disable When this bit is set, the MAC disables the Jabber timer on the transmitter, and can transfer frames of up to 16,384 bytes. When this bit is cleared, the MAC cuts of the transmitter if the application sends out more than 2048 bytes of data during transmission.
Bit 21: 20	Reserved	0x0	resd	Kept at its default value.
Bit 19: 17	IFG	0x0	rw	InterFrame Gap These bits are used to define the minimum interframe gap between frames during transmission. 000: 96 bit times 96 bit times 001: 88 bit times 88 bit times 010: 80 bit times 80 bit times ... 111: 40 bit times 40 bit times In half-duplex mode, the minimum IFG can be configured as 64 bit times (IFG=100). Lower values are not allowed.
Bit 16	DCS	0x0	rw	Disable Carrier Sense When this bit is set, the MAC transmitter will ignore the MII CRS signal during frame transmission in half-duplex mode. No error is reported due to loss of carrier or no carrier during transmission. When this bit is cleared, the MAC transmitter will report errors due to carrier sense and even abort the transmission. This bit is reserved in full-duplex mode.
Bit 15	Reserved	0x1	resd	Kept at its default value.
Bit 14	FES	0x0	rw	Fast EMAC Speed This bit indicates the speed of the MII, RMII interface. 0: 10 Mbps 1: 100 Mbps
Bit 13	DRO	0x0	rw	Disable Receive Own When this bit is set, the MAC disables the frame reception in half-duplex mode if the phy_txen_o is enabled. When this bit is cleared, the MAC will receive all packets that are given by the PHY during transmission. This bit is not applicable when the MAC is in full-duplex mode. This bit is reserved (with default value RO) when the MAC is configured as “For full-duplex mode only” mode.
Bit 12	LM	0x0	rw	Loopback Mode When this bit is set, the MAC MII operates in loopback

				mode. The MII receive clock input (clk_rx_i) is required for the loopback mode to work normally, for the transmit clock is not looped-back internally.
Bit 11	DM	0x0	rw	Duplex Mode When this bit is set, the MAC operates in full-duplex mode, in which it can transmit and receive simultaneously.
Bit 10	IPC	0x0	rw	IPv4 Checksum When this bit is set, the MAC calculates the 16-bit complement sum of all received Ethernet frames and enables IPv4 header checksum (assuming it is bytes 26-26 or 29-30 (VLANtagged)) for received frames, and gives the status in the receive status information. The MAC also appends the 16-bit checksum of the calculated IP header packets (bytes after the IPv4header), and adds it to the Ethernet frame that has been sent out to the application (when Type 2 COE is deselected). When this bit is cleared, this feature is disabled. When this bit is set, IPv4 header checksum feature and IPv4 or IPv6 TCP, UDP or ICMP payload checksum feature is enabled while the Type 2 COE is selected. When this bit is cleared, the COE function in the receiver is disabled, and the corresponding PCE and IP HCE status bits are always 0. This bit is reserved (with default value RO) if the IP checksum mechanism is disabled during the core configuration.
Bit 9	DR	0x0	rw	Disable Retry When this bit is set, the MAC attempts only 1 transmission. When a collision occurs on the MII interface, the MAC will ignore the current frame transmission and report a frame abort because of excessive collision error in the transmit frame status. When this bit is cleared, the MAC attempts retries based on the settings of BL ([6: 5]). This bit is applicable only in half-duplex mode. It is reserved (with default value RO) in “For full-duplex mode only” mode.
Bit 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	ACS	0x0	rw	Automatic pad/CRC Stripping When this bit is set, the MAC strips the pad/FCS field on received frames only when the frame length is shorter than 1536 bytes. All received frame with length field greater than or equal to 1536 bytes are passed on to the application without stripping the Pad or FCS field. When this bit is cleared, the MAC will forward all received frames to the master without changing its contents.
Bit 6: 5	BL	0x0	rw	Back-off Limit The Back-off limit defines the random integer number (r) of slot time delays (512 bit times for 10/100 Mbps) the MAC waits before retries after a collision. This field is applicable only in the half-duplex mode. It is reserved (RO) in “For full-duplex mode only” mode. 00: k= min (n, 10) 01: k = min (n, 8) 10: k = min (n, 4) 11: k = min (n, 1) Where n = the number of slot time delays for retransmission attempt, and r takes the random integer value in the range 0 ≤ r < 2k.
Bit 4	DC	0x0	rw	Deferral Check When this bit is set, the deferral check function is enabled in the MAC. The MAC issues a frame abort status and sets the excessive deferral error flag bit in the transmit frame status when the transmit state machine is delayed for more than 24288 bit times in 10/100 Mbit/s mode. If the Jumbo frame mode is enabled in 10/100 Mbps mode,

				the deferral threshold is 155680 bit times. Deferral begins when the transmitter is ready to transmit, but is prevented when an active carrier sense signals is detected on the MII. Deferral time is not cumulative. For instance, if the transmitter is deferred for 10000 bit times because that the CRS signals is active first, but then becomes inactive, then transmits, collides, backs off because of collision, and then has to defer again after the completion of back-off, the deferral times resets to 0 and restarts. When this bit is cleared, the deferral check function is disabled. The MAC defers until the CRS signal becomes inactive. This bit is applicable only in the half-duplex mode. It is reserved (RO) in “For full-duplex mode only” mode.
Bit 3	TE	0x0	rw	Transmitter Enable When this bit is set, the transmit state machine of the MAC is enabled. when this bit is cleared, the MAC disables the transmit state machine after the completion of the current frame transmission, and does not transmit any further frames (To modify this bit through consecutive commands, if needed, a deferral value greater than 4us is required between two consecutive operations)
Bit 2	RE	0x0	rw	Receiver Enable When this bit is set, the receive state machine of the MAC is enabled. when this bit is cleared, the MAC disables the receive state machine after the completion of the current frame reception, and does not receive any further frames (To modify this bit through consecutive commands, if needed, a deferral value greater than 4us is required between two consecutive operations).
Bit 1: 0	Reserved	0x0	resd	Kept at its default value.

26.3.2 Ethernet MAC frame filter register (EMAC_MACFRMF)

The Ethernet MAC frame filter register contains the filter control bits for receiving frames. Some of the control bits got to the address check block of the MAC to perform the first level of address filtering.

The second level of filtering is performed on the incoming frames based on other control bits (such as pass bad frames and pass control frames).

Bit	Register	Reset value	Type	Description
Bit 31	RA	0x0	rw	Receive All When this bit is set, the MAC passes all received frames onto the application, irrespective of whether they have passed through the address filter. The result (pass or fail) of the source address or destination address filtering is updated in the corresponding bits of the receive status word. When this bit is cleared, the MAC passes on to the application only those frames that have passed the source address or destination address filtering.
Bit 30: 11	Reserved	0x00000	resd	Kept at its default value.
Bit 10	HPF	0x0	rw	Hash or Perfect Filter When this bit is set, the address filter passes frames that match the perfect filter or hash filter set by the HMC or HUC bit. When this bit is cleared, if the HUC or HMC bit is set, only frames that match the hash filter can pass address filter.
Bit 9	SAF	0x0	rw	Source Address Filter When this bit is set, the MAC compares the source address of the received frame with the value programmed in the enabled source address registers. If the comparison mismatches, the MAC will drop this frame. (SAF). When this bit is cleared, the MAC forwards the received frame to the application and updates the source address filter bit (SAF) in the receive status based on the source address comparison.
Bit 8	SAIF	0x0	rw	Source Address Inverse Filtering

				When this bit is set, the address check block operates in inverse filtering mode. The frame whose source address matches the source address register is marked as failing the source address filter. When this bit is cleared, the frame whose source address does not match the source address register is marked as failing the source address filter.
Bit 7: 6	PCF	0x0	rw	Pass Control Frames These bits control the forwarding of all control frames (including unicast and multicast Pause frames). 00: MAC filters all control frames and prevents them from reaching the application 01: MAC forwards all control frames, except Pause frame, to the application even if they fail the address filter 10: MAC forwards all control frames to the application even if they fail the address filter 11: MAC forwards control frames that pass the address filter to the application The following conditions must be met when dealing with a Pause frame: 1: When the MAC is in full-duplex mode, the bit 2 (REF) is set in the register 6 (flow control register) to enable flow control. 2: When the bit 3 (UP) is set in the register 6 (flow control register), the destination address of the received frames matches the specific multicast address or MAC address 0. 3: Type field of the receive frame is 0x8808, and the OPCODE field is 0x0001.
Bit 5	DBF	0x0	rw	Disable Broadcast Frames When this bit is set, the address filters filter all incoming broadcast frames. In addition, all other filter settings will also be overwritten. When this bit is set, the address filters pass all incoming broadcast frames.
Bit 4	PMC	0x0	rw	Pass MultiCast When this bit is set, all frames with a multicast destination address (first bit in the destination address is set) are passed. When this bit is cleared, the filtering of a multicast frame depends on the HMC bit.
Bit 3	DAIF	0x0	rw	Destination Address Inverse Filtering When this bit is set, the address check block operates in inverse filtering mode for the destination address comparison for both unicast and multicast frames. When this bit is cleared, the filter work normally.
Bit 2	HMC	0x0	rw	Hash MultiCast When this bit is set, the MAC performs destination address filtering of the received multicast frames according to the hash table. When this bit is cleared, the MAC performs a perfect destination address filtering for multicast frames, that is, it compares the destination address field with the values programmed in the destination registers. This bit is reserved if Hash filter is not selected during core configuration.
Bit 1	HUC	0x0	rw	Hash UniCast When this bit is set, the MAC performs destination address filtering for unicast frames according to the hash table. When this bit is cleared, the MAC performs a perfect destination address filtering for unicast frames, that is, it compares the destination address field with the values programmed in the destination registers.
Bit 0	PR	0x0	rw	Promiscuous Mode When this bit is set, the address filters pass all incoming frames regardless of their destination or source address.

	When the PR is set, the source address or destination address error bits in the receive status word are always 0.

26.3.3 Ethernet MAC Hash table high register (EMAC_MACHTH)

The 64-bit Hash table is used for group address filtering. For Hash filtering, the contents of the destination address of the incoming frame pass through the CRC logic, and the upper 6 bits in the CRC register are used to index the Hash table. The most significant bit of the CRC determines the register to be used (EMAC_MACHTH or EMAC_MACHTL), and the other 5 bits determine which bit in the register is to be used. The Hash value 5b'00000 uses the bit 0 in the selected register, while the Hash value 5b'11111 uses the bit 31 in the selected register.

The Hash value of the destination address is calculated according to the following steps:

1. Calculate a 32-bit CRC value of the destination address (see IEEE 802.3, and refer to 3.2.8 section for more details)
2. Bit invert the value obtained in Step 1
3. Take the upper 6 bits from the values obtained in Step 2

For example, if the destination address of the incoming frame is 0x1F52419CB6AF (0x1F is the first byte received on the MII interface), the calculated 6-bit Hash value is 0x2C and the bit 12 in the EMAC_MACHTH registers checked for filtering. If the destination address of the incoming frame is 0xA00A98000045, the calculated 6-bit Hash value is 0x07, and the bit 7 in the EMAC_MACHTL register is checked for filtering.

Bit	Register	Reset value	Type	Description
Bit 31	HTH	0x0000 0000	rw	This bit contains the upper 32 bits of the Hash table.

26.3.4 Ethernet MAC Hash table low register (EMAC_MACHTL)

The EMAC_MACHTL register contains the lower 32 bits of the Hash table. If the Hash filter is disabled or either 128-bit or 256-bit Hash table is selected, both register 2 and register 3 are reserved.

Bit	Register	Reset value	Type	Description
Bit 31	HTL	0x0000 0000	rw	Hash Table Low This bit contains the lower 32 bits of the Hash table.

26.3.5 Ethernet MAC MII address register (EMAC_MACMIIADDR)

The Ethernet MAC MII address register controls the external PHY through the management interface.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 11	PA	0x00	rw	PHY Address This field indicates which of the 32 possible PHY devices are being accessed.
Bit 10: 6	MII	0x00	rw	MII Register This field select the desired MII register in the PHY device.
Bit 5: 2	CR	0x0	rw	Clock Range The CSR clock range selection determines the MDC clock frequency based on the used CSR clock frequency. Each value (when bit 5=0) has its corresponding CSR clock frequency range in order to ensure that the MDC clock frequency is roughly between 1.0 MHz and 2.5 MHz. 0000: CSR clock frequency is 60–100 MHz, and MDC clock frequency is CSR clock/42 0001: CSR clock frequency is 100–150 MHz, and MDC clock frequency is CSR clock/62 0010: CSR clock frequency is 20–35 MHz, and MDC clock frequency is CSR clock/16 0011: CSR clock frequency is 35–60 MHz, and MDC clock frequency is CSR clock/26 0100: CSR clock frequency is 150–250 MHz, and MDC clock frequency is CSR clock/102 0101: CSR clock frequency is 250–288 MHz, and MDC clock frequency is CSR clock/124 0110, 0111: Reserved

Bit 1	MW	0x0	rw	MII Write When this bit is set, it indicates that the EMAC_MACMIIDT register is used for a write operation to the PHY. When this bit is not set, it is a read operation, and the data is loaded to the EMAC_MACMIIDT register.
Bit 0	MB	0x0	rw	MII Busy This bit should read a logic 0 before writing to the EMAC_MACMIIADDR and EMAC_MACMIIDT register. During a PHY register access, this bit is set to 1'b1 by software, indicating that a read or write access is in progress. The EMAC_MACMIIDT register is invalid before this bit is cleared by the MAC. Thus, the MII data should be kept valid until this bit is cleared by the MAC during a PHY write operation. Similarly, the EMAC_MACMIIDT value is invalid until this bit is cleared by the MAC during a PHY read operation. The previous operation must be completed before performing subsequent read or write operations. This is because that there will be no acknowledgement from PHY to MAC after the completion of a read or write operation, the function of this bit will not change even if the PHY is not present.

26.3.6 Ethernet MAC MII data register (EMAC_MACMIIDT)

The Ethernet MAC MII data register stores data to be written to the PHY register located at the address specified in the EMAC_MACMIIADDR register. EMAC_MACMIIDT register also stores data read out from the PHY registers.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	MD	0x0000	rw	MII Data This field contains the 16-bit value from the PHY after a read operation, or the 16-bit value to be written to the PHY before a write operation.

26.3.7 Ethernet MAC flow control register (EMAC_MACFCTRL)

The Ethernet MAC flow control register controls the generation and reception of the control frames by the MAC flow control block. Writing 1 to the Busy bit triggers the flow control block to generate a Pause frame. The field of the control frame is selected as defined in the 802.3x specification, and the Pause Time value from this register is used in the Pause Time field of the control frame. The Busy bit remains set before the control frame is transferred onto the cable. The host must make sure that the Busy bit is cleared before writing to the register.

Bit	Register	Reset value	Type	Description
Bit 31: 16	PT	0x0000	rw	Pause Time This field contains the value to be used in the Pause Time field of the control frame. If the Pause Time bit is configured to be double-synchronized to the MII clock domain, then consecutive write operations to this register should be performed only after at least four clock cycles in the destination clock domain.
Bit 15: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7	DZQP	0x0	rw	Disable Zero-Quanta Pause When this bit is set, it disables the automatic generation of Zero-quanta Pause frame while the flow control signal of the FIFO layer is disabled. When this bit is cleared, normal operation resumes. The automatic generation of Zero-quanta Pause frame is enabled.
Bit 6	Reserved	0x0	resd	Kept at its default value.
Bit 5: 4	PLT	0x0	rw	Pause Low Threshold This field defines the threshold of the Pause timer.

				The threshold values should always be less than the Pause time defined in the [31: 16] bit. For example, if PT = 100H (256 slot times), and PLT = 01, then a second Pause frame is automatically transmitted if initiated at 228 (256-28) slot times after the first Pause frame is transmitted. Threshold selection as follows: 00: Pause time minus 4 slot times (PT minus 4 slot times) 01: Pause time minus 28 slot times (PT minus 28 slot times) 10: Pause time minus 144 slot times (PT minus 144 slot times) 11: Pause time minus 256 slot times (PT minus 256 slot times) Slot time is defined as the time taken to transmit 512 bits (64 bytes) on the MII interface.
Bit 3	DUP	0x0	rw	Detect Unicast Pause Frame The Pause frame with a unique multicast address as specified in the IEEE 802.3 will be processed. When this bit is set, the MAC detects the Pause frames with a unicast address specified in the MAC address0 high and MAC address0 low registers. When this bit is cleared, the MAC detects only a Pause frame with a unique multicast address. Note: If the multicast address of the received frame does not match the unique multicast address, the MAC will not process the Pause frame.
Bit 2	ERF	0x0	rw	Enable Receive Flow control When this bit is set, the MAC decodes the received Pause frame and disables the transmitter for a period of time. When this bit is cleared, the decode function of the Pause frame is disabled.
Bit 1	ETF	0x0	rw	Enable Transmit Flow control In full-duplex mode, when this bit is set, the MAC enables the flow control operation to transmit Pause frames. When this bit is cleared, the flow control of the MAC is disabled, and the MAC does not transmit any Pause frames. In half-duplex mode, when this bit is set, the MAC enables the back-pressure feature. When this bit is cleared, the back-pressure feature is disabled.
Bit 0	FCB/BPA	0x0	rw1c/rw	Flow Control Busy/Back Pressure Activate In full-duplex mode, this bit initiates a Pause frame; in half-duplex mode, the back-pressure feature is activated if the TFE bit is set. In full-duplex mode, this bit is read as 1'b0 before writing to the EMAC_MACFCTRL register. The application must set this bit to 1'b1 to initiate a Pause frame. During a control frame transmission, this bit remains set, indicating that a frame transmission is in progress. After the completion of the Pause frame, the MAC resets this bit to 1'b0. The Ethernet MAC flow control register (EMAC_MACFCTRL) should not be written until this bit is cleared. In half-duplex mode, when this bit is set (and the TFE is set), the back-pressure feature is activated by the MAC. During back pressure, when the MAC receives a new frame, the transmitter starts sending a JAM mode, resulting a collision. When the MAC is configured to full-duplex mode, the back-pressure (BPA) function is automatically disabled.

26.3.8 Ethernet MAC VLAN tag register (EMAC_MACVLT)

The Ethernet MAC VLAN tag register contains the IEEE 802.1Q VLAN tag to identify the VLAN frames. The MAC compares the 13th and 14th bytes of the received frame (length/type) with 16'h8100, and the following 2 bytes are compared with the VLAN tag. If the comparison matches, the VLAN bit is set in the receive frame status. The legal length of the VLAN frame is increased from 1518 bytes to 1522 bytes.

If the EMAC_MACVLT register is configured to be double-synchronized to the (G)MII clock domain, then consecutive write operations to this register should be performed at least four clock cycles in the destination clock domain.

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	ETV	0x0	ro	Enable 12-bit VLAN tag comparison When this bit is set, a 12-bit VLAN identifier, rather than a 16-bit VLAN tag, is used for comparison and filtering. The bit [11: 0] of the VLAN tag is compared with the corresponding filed in the received VLAN-tagged frame. Similarly, if enabled, only a 12-bit VLAN tag is used for hash VLAN filtering. When this bit is cleared, the 16 bits of the received VLAN frame's 15th and 16th bytes are used for comparison and VLAN hash filtering.
Bit 15: 0	VTI	0x0000	rw	VLAN Tag Identifier (for receive frames) This field contains the 802.1Q VLAN tag to identify VLAN frames, which is compared with the 15th and 16th bytes of the received VLAN frames, described as follows: Bit [15: 13]: User priority Bit 12: Canonical format indicator (CFI) or drop eligible indicator (DEI) Bit [11: 0]: VLAN tag's VLAN identifier field When the ETV bit is set, only the VID ([11: 0]) is used for comparison. If the VL is all zero (if the ETV is set, then VL[11: 0] is all zero), the MAC does not check the 15th and 16th bytes for VLAN tag comparison, and treats all frames with a type field value of 0x8100 or 0x88a8 as VLAN frames.

26.3.9 Ethernet MAC remote wakeup frame filter register (EMAC_MACRWFF)

The PMT CSR sets the request wakeup events and detects the wakeup events.

Figure 26-18 Ethernet MAC remote wakeup frame filter register (EMAC_MACRWFF)

Wkuppktfilter_reg0	Filter 0 Byte Mask							
Wkuppktfilter_reg1	Filter 1 Byte Mask							
Wkuppktfilter_reg2	Filter 2 Byte Mask							
Wkuppktfilter_reg3	Filter 3 Byte Mask							
Wkuppktfilter_reg4	RESD	Filter 3 Cmd	RESD	Filter 2 Cmd	RESD	Filter 1 Cmd	RESD	Filter 0 Cmd
Wkuppktfilter_reg5	Filter 3 Offset		Filter 2 Offset		Filter 1 Offset		Filter 0 Offset	
Wkuppktfilter_reg6	Filter 1 CRC-16				Filter 0 CRC-16			
Wkuppktfilter_reg7	Filter 3 CRC-16				Filter 2 CRC-16			

26.3.10 Ethernet MAC PMT control and status register (EMAC_MACPMTCTRLSTS)

The Ethernet MAC PMT control and status register sets the request wakeup events and detects the wakeup events.

Bit	Register	Reset value	Type	Description
Bit 31	RWFFPR	0x0	rw1s	Remote Wakeup Frame Filter Register Pointer Reset When this bit is set, it resets the remote frame filter register pointer to 3'b000. This bit is automatically cleared after one clock cycle.
Bit 30: 10	Reserved	0x000000	resd	Kept at its default value.
Bit 9	GUC	0x0	rw	Global UniCast When this bit is set, it enables all unicast packets filtered by the MAC address filtering to be remote wakeup frames.
Bit 8: 7	Reserved	0x0	resd	Kept at its default value.
Bit 6	RRWF	0x0	rrc	Received Remote Wakeup Frame When this bit is set, it indicates that the power management event was generated because of the reception of a remote wakeup frame. This bit is cleared by a read access to this register.
Bit 5	RMP	0x0	rrc	Received Magic Packet When this bit is set, it indicates that the power management event is generated because of the reception of a Magic packet. This bit is cleared by a read access to this register.
Bit 4: 3	Reserved	0x0	resd	Kept at its default value.
Bit 2	ERWF	0x0	rw	Enable Remote Wakeup Frame When this bit is set, it indicates that the power management event is generated due to a remote wakeup frame reception.
Bit 1	EMP	0x0	rw	Enable Magic Packet When this bit is set, it indicates that the power management event is generated due to a Magic packet reception.
Bit 0	PD	0x0	rw1s	Power Down When this bit is set, the MAC receiver will drop all received frames after receiving the expected Magic packet or a remote wakeup frame. Then this bit is automatically cleared and power-down mode is disabled. This bit can also be cleared by software before the expected Magic packet or a remote wakeup frame is received. After this bit is cleared, the MAC forwards the receive frames to the application. This bit must only be set when either the Magic Packet enable bit, global unicast bit or the remote wakeup frame enable bit is set high.

26.3.11 Ethernet MAC interrupt status register (EMAC_MACISTS)

The Ethernet MAC interrupt status register identify the events in the MAC that can generate an interrupt.

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at its default value.
Bit 9	TIS	0x0	rrc	Timestamp Interrupt Status When this bit is set, it indicates that the system time value equals or exceeds the value programmed in the destination time registers. This bit is cleared after the completion of a read operation to this bit.
Bit 8: 7	Reserved	0x0	resd	Kept at its default value.
Bit 6	MTIS	0x0	ro	MMC Transmit Interrupt Status This bit is set when an interrupt event is generated in the EMAC_MMCTI register. This bit is cleared when all bits in the transmit interrupt register are cleared.
Bit 5	MRIS	0x0	ro	MMC Receive Interrupt Status This bit is set when an interrupt is generated in the

				EMAC_MMCRI register. This bit is cleared when all bits in the receive interrupt register are cleared.
Bit 4	MIS	0x0	ro	MMC Interrupt Status This bit is set whenever any bit of the [7: 5] bit is set high. This bit is cleared only when these bits are set low.
Bit 3	PIS	0x0	ro	PMT Interrupt Status This bit is set when a Magic packet or a remote wakeup event is received in power-down mode (see bits 5 and 6 in the EMAC_MACPMTCTRLSTS register). This bit is cleared when both bits [6: 5] are cleared due to a read access to the EMAC_MACPMTCTRLSTS register.
Bit 2: 0	Reserved	0x0	resd	Kept at its default value.

26.3.12 Ethernet MAC interrupt mask register (EMAC_MAIMR)

The Ethernet MAC interrupt mask register is used to mask the interrupt signal generated due to the corresponding event in the EMAC_MACISTS register.

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at its default value.
Bit 9	TIM	0x0	rw	Timestamp Interrupt Mask When this bit is set, it masks the interrupt signal generated in the time stamp interrupt status bit of the EMAC_MACISTS register. This bit is applicable only when the IEEE1588 time stamp is enabled. This bit is reserved in other modes.
Bit 8: 4	Reserved	0x00	resd	Kept at its default value.
Bit 3	PIM	0x0	rw	PMT Interrupt Mask. When this bit is set, it masks the interrupt signal generated in the MPT interrupt status bit of the EMAC_MACISTS register.
Bit 2: 0	Reserved	0x0	resd	Kept at its default value.

26.3.13 Ethernet MAC address 0 high register (EMAC_MACA0H)

The EMAC_MACA0H register contains the upper 6 bits of the first 6-byte MAC address of the station. The first DA byte received on the MII interface corresponds to the LS byte (bit [7: 0]) of the MAC address low register. For example, if the 0x112233445566 (0x11 in channel 0 of the first column) is received on the MII interface as the destination address, then the MacAddress0 register [47: 0] is compared with 0x665544332211.

If the MAC address register is configured to be double-synchronized with the MII domain, the synchronization can be enabled only by writing the bit [31: 24] (in little endian mode) or the bit [7: 0] (in big-endian mode) in the Ethernet MAC address 0 low register (EMAC_MACA0L). Consecutive write operations to this address low register must be performed after at least 4 cycles in the destination clock domain so as to achieve an accurate synchronous update.

Bit	Register	Reset value	Type	Description
Bit 31	AE	0x0	rrc	Adrress Always 1.
Bit 30: 16	Reserved	0x0010	resd	Kept at its default value.
Bit 15: 0	MA0H	0xFFFF	rw	MAC Address0 [47: 32] This field contains the upper 16 bits of the first 6-byte MCU address. This is used by the MAC for filtering received frames, and for inserting the MAC address in the transmit flow control frames (Pause).

26.3.14 Ethernet MAC address 0 low register (EMAC_MACA0L)

The Ethernet MAC address 0 low register contains the lower 32 bits of the 6-byte first MAC address.

Bit	Register	Reset value	Type	Description
Bit 31: 0	MA0L	0xFFFF FFFF	rw	MAC Address0 [31: 0] This field contains the lower 16 bits of the first 6-byte MCU address. This is used by the MAC for filtering received frames, and for inserting the MAC address in the transmit flow control frames (Pause).

26.3.15 Ethernet MAC address 1 high register (EMAC_MACA1H)

The Ethernet MAC address 1 high register holds the upper 16 bits of the 6-byte second MAC address.

If the MAC address register is configured to be double-synchronized with the MII domain, the synchronization can be enabled only by writing the bit [31: 24] (in little endian mode) or the bit [7: 0] (in big-endian mode) in the Ethernet MAC address 1 low register (EMAC_MACA1L) . Consecutive write operations to this address low register must be performed after at least 4 cycles in the destination clock domain so as to achieve an accurate synchronous update.

Bit	Register	Reset value	Type	Description
Bit 31	AE	0x0	rw	Address Enable When this bit is set, the address filter uses the second MAC address for a perfect filtering. When this bit is cleared, the address filter will ignore the address for filtering.
Bit 30	SA	0x0	rw	Source Address When this bit is set, the MAC address 1 [47: 0] is used for comparison with the source address field of the received frame. When this bit is cleared, the MAC address 1 [47: 0] is used for comparison with the destination address field of the received frame.
Bit 29: 24	MBC	0x00	rw	Mask Byte Control These bits are mask control bits for comparison with each of the MAC address bytes. When this bit is set, the MAC does not compare the corresponding byte of the received DA/SA with the contents of the MAC address 1 register. Each control bit is used for controlling the mask of the bytes as follows: Bit 29: EMAC_MACA1H [15: 8] Bit 28: EMAC_MACA1H [7: 0] Bit 27: EMAC_MACA1L[31: 24] ... Bit 24: EMAC_MACA1L[7: 0] It is possible to filter group addresses (that is, group address filtering) by masking one or more bytes of the address.
Bit 23: 16	Reserved	0x00	resd	Kept at its default value.
Bit 15: 0	MA1H	0xFFFF	rw	MAC Address1 [47: 32] These bits contain the upper 16 bits (47: 32) of the 6-byte second MAC address.

26.3.16 Ethernet MAC address 1 low register (EMAC_MACA1H)

The Ethernet MAC address 1 low register contains the lower 32 bits of the 6-byte second MAC address.

Bit	Register	Reset value	Type	Description
Bit 31: 0	MA1L	0xFFFF FFFF	rw	MAC Address1 [31: 0] These bits contain the lower 32 bits of the 6-byte second MAC address. The contents of this field is undefined until loaded by the application after the initialization process.

26.3.17 Ethernet MAC address 2 high register (EMAC_MACA2H)

The Ethernet MAC address 2 high register holds the upper 16 bits of the 6-byte second MAC address.

If the MAC address register is configured to be double-synchronized with the MII domain, the synchronization can be enabled only by writing the bit [31: 24] (in little endian mode) or the bit [7: 0] (in big-endian mode) in the Ethernet MAC address 2 low register (EMAC_MACA2L). Consecutive write operations to this address low register must be performed after at least 4 cycles in the destination clock domain so as to achieve an accurate synchronous update.

Bit	Register	Reset value	Type	Description
Bit 31	AE	0x0	rw	Address Enable When this bit is set, the address filter uses the second MAC address for a perfect filtering. When this bit is cleared, the address filter will ignore the address for filtering.
Bit 30	SA	0x0	rw	Source Address When this bit is set, the MAC address2 [47: 0] is used for comparison with the source address field of the received frame. When this bit is cleared, the MAC address 2 [47: 0] is used for comparison with the destination address field of the received frame.
Bit 29: 24	MBC	0x00	rw	Mask Byte Control These bits are mask control bits for comparison with each of the MAC address bytes. When this bit is set, the MAC does not compare the corresponding byte of the received DA/SA with the contents of the MAC address 2 register. Each control bit is used for controlling the mask of the bytes as follows: Bit 29: EMAC_MACA2H [15: 8] Bit 28: EMAC_MACA2H [7: 0] Bit 27: EMAC_MACA2L[31: 24] ... Bit 24: EMAC_MACA2L[7: 0] It is possible to filter group addresses (that is, group address filtering) by masking one or more bytes of the address.
Bit 23: 16	Reserved	0x00	resd	Kept at its default value.
Bit 15: 0	MA2H	0xFFFF	rw	MAC Address2 High [47: 32] These bits contain the upper 16 bits (47: 32) of the 6-byte second MAC address.

26.3.18 Ethernet MAC address 2 low register (EMAC_MACA2L)

The Ethernet MAC address 2 low register holds the lower 32 bits of the 6-byte second MAC address.

Bit	Register	Reset value	Type	Description
Bit 31: 0	MA2L	0xFFFF FFFF	rw	MAC Address2 Low [31: 0] These bits contain the lower 32 bits of the 6-byte second MAC address. The contents of this field is undefined until loaded by the application after the initialization process.

26.3.19 Ethernet MAC address 3 high register (EMAC_MACA3H)

The Ethernet MAC address 3 high register holds the upper 16 bits of the 6-byte second MAC address.

If the MAC address register is configured to be double-synchronized with the MII domain, the synchronization can be enabled only by writing the bit [31: 24] (in little endian mode) or the bit [7: 0] (in big-endian mode) in the Ethernet MAC address 3 low register (EMAC_MACA3L). Consecutive write operations to this address low register must be performed after at least 4 cycles in the destination clock domain so as to achieve an accurate synchronous update.

Bit	Register	Reset value	Type	Description
Bit 31	AE	0x0	rw	Address Enable When this bit is set, the address filter uses the second MAC address for a perfect filtering.

				When this bit is cleared, the address filter will ignore the address for filtering.
Bit 30	SA	0x0	rw	Source Address When this bit is set, the MAC address 3 [47: 0] is used for comparison with the source address field of the received frame. When this bit is cleared, the MAC address 3 [47: 0] is used for comparison with the destination address field of the received frame.
Bit 29: 24	MBC	0x00	rw	Mask Byte Control These bits are mask control bits for comparison with each of the MAC address bytes. When this bit is set, the MAC does not compare the corresponding byte of the received DA/SA with the contents of the MAC address 3 register. Each control bit is used for controlling the mask of the bytes as follows: Bit 29: EMAC_MACA3H [15: 8] Bit 28: EMAC_MACA3H [7: 0] Bit 27: EMAC_MACA3L[31: 24] ... Bit 24: EMAC_MACA3L[7: 0] It is possible to filter group addresses (that is, group address filtering) by masking one or more bytes of the address.
Bit 23: 16	Reserved	0x00	resd	Kept at its default value.
Bit 15: 0	MA3H	0xFFFF	rw	MAC Address3 High [47: 32] These bits contain the lower 16 bits (47: 32) of the 6-byte second MAC address.

26.3.20 Ethernet MAC address 3 low register (EMAC_MACA3L)

The Ethernet MAC address 3 low register holds the lower 32 bits of the 6-byte second MAC address.

Bit	Register	Reset value	Type	Description
Bit 31: 0	MA3L	0xFFFF FFFF	rw	MAC Address3 Low [31: 0] These bits contain the lower 32 bits of the 6-byte second MAC address. The contents of this field is undefined until loaded by the application after the initialization process.

26.3.21 Ethernet DMA bus mode register (EMAC_DMABM)

The Ethernet DMA bus mode register defines the bus operation modes for the DMA.

Bit	Register	Reset value	Type	Description
Bit 31: 26	Reserved	0x00	resd	Kept at its default value.
Bit 25	AAB	0x0	rw	Address-Aligned Beats When this bit is set and the FB bit equals 1, the AHB interface generates burst transfers aligned to the start address LS bits. If the FB bit equals 0, the first burst transfer (accessing the data buffer's start address) is not aligned, but subsequent burst transfers are aligned to the address. This bit is applicable to GMAC-AHB and GMAC-AXI configurations only. It is reserved in other configurations.
Bit 24	PBLx8	0x0	rw	PBLx8 Mode When this bit is set, this bit multiples the PBL value programmed (bits [22: 17] and bits [13: 8]) by 8. Thus the DMA transfers data at 8, 16, 32, 64, 128 and 256 beats depending on the PBL value.
Bit 23	USP	0x0	rw	Use separate PBL When this bit is set, the Rx DMA uses the value programmed in bit [22: 17] as PBL. The PBL value in bit [13: 8] is applicable to Tx DMA operations only. When this bit is cleared, the PBL value in bit [13: 8] is applicable to both Tx DMA and Rx DMA operations.
Bit 22: 17	RDP	0x01	rw	Rx DMA PBL

				This field indicates the maximum number of beats to be transferred in one Rx DMA operation. This is the maximum value that is used for a single write or read operation. The Rx DMA always attempts to perform burst transfer as specified in RPBL each time it starts a burst transfer on the host bus. The RPBL can be programmed with 1, 2, 4, 8, 16 and 32. Any other value result in unexpected behavior. These bits are applicable only when the USP bit is set.
Bit 16	FB	0x0	rw	Fixed Burst This bit controls whether the AHB master interface performs fixed burst transfers or not. When this bit is set, the AHB uses only SINGLE, INCR4, INCR8 or INCR16 during start of normal burst transfers. When this bit is cleared, the AHB or AXI interface uses SINGLE and INCR burst transfer operations.
Bit 15: 14	PR	0x0	rw	Priority Ratio These bits control the priority ratio of the round-robin arbitration between Rx DMA and Tx DMA. These bits are valid only when the bit 1 (destination address) is reset. The priority ratio is either Rx: Tx or Tx: Rx, depending on whether the bit 27 (TXPR) is set or reset. 00: 1: 1 01: 2: 1 10: 3: 1 11: 4: 1
Bit 13: 8	PBL	0x01	rw	Programmable Burst Length These bits indicate the maximum number of beats to be transferred in one DMA transaction. This is the maximum that is used for a single write or read operation. The DMA always attempts to perform burst transfer as specified in PBL each time it starts a burst transfer on the host bus. The RPBL can be programmed with 1, 2, 4, 8, 16 and 32. Any other value result in unexpected behavior. When the USP is set, the PBL value is applicable to Tx DMA operations only. If the number of beats to be transferred is greater than 32, the following steps are required: 1. Set PBLx8 mode 2. Set PBL For example, if the maximum value to be transferred is greater than 64, then the PBLx8 should be set first, and then the PBL is set to 8. The PBL values have the following limitations: The maximum number of beats possible is limited by the size of the Tx FIFO and Rx FIFO on the MTL layer, as well as the data bus width on the DMA. FIFO constraint: The maximum beat supported by the FIFO is half the depth of the FIFO, unless otherwise specified.
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6: 2	DSL	0x00	rw	Descriptor Skip Length These bits define the number of words to skip between two unchained descriptors. The address skip starts from the end of the current descriptor to the start of next descriptor. When the DSL value equals 0, the descriptor is regarded as contiguous by the DMA in ring mode.
Bit 1	DA	0x0	rw	DMA Arbitration These bits specify the arbitration scheme between the transmit path and receive path of channel 0. 0: Rx: Tx or Tx: Rx The priority between round-robin channels depends on the priority as specified in the bit [15: 14] (PR) and the priority weight as specified in bit 27(TXPR). 1: Fixed priority When the bit 27 (TXPR) is set, Tx has priority over Rx.

				Otherwise, Rx has priority over Tx.
Bit 0	SWR	0x1	rw	Software Reset When this bit is set, the MAC DMA controller resets all internal registers and MAC logic. This bit is automatically cleared after all reset operations have been completed.

26.3.22 Ethernet DMA transmit poll demand register (EMAC_DMATPD)

The EMAC_DMATPD register enables the Tx DMA to check whether or not the current descriptor is owned by the DMA. The Transmit Poll Demand is used to wake up the Tx DMA from suspend mode. The Tx DMA can go into suspend mode due to an underflow error in a transmitted frame or due to the unavailability of descriptors owned by transmit DMA. The Poll demand can be issued at any time, and the Tx DMA will reset this command once it starts re-fetching the current descriptor from the host memory. This register is always read 0.

Bit	Register	Reset value	Type	Description
Bit 31: 0	TPD	0x0000 0000	rrc	Transmit Poll Demand When these bits are written with any value, the DMA reads the current descriptor pointed to by the EMAC_DMACTD. If the descriptor is not available (owned by host), the transmission suspends, and the bit 2 (TU) is set in the status register. If the descriptor is available, the transmission resumes.

26.3.23 Ethernet DMA receive poll demand register (EMAC_DMARPD)

The EMAC_DMARPD register enables the Rx DMA to check new descriptors. The Receive Poll Demand is used to wake up the Rx DMA from suspend mode. The Rx DMA can enter suspend mode due to the unavailability of descriptors owned by it.

Bit	Register	Reset value	Type	Description
Bit 31: 0	RPD	0x0000 0000	rrc	Receive Poll Demand When these bits are written with any value, the DMA reads the current descriptor pointed to by the EMAC_DMACRD. If the descriptor is not available (owned by host), the reception suspends, and the bit 7 (RU) is set in the status register. If the descriptor is available, the reception resumes.

26.3.24 Ethernet DMA receive descriptor list address register (EMAC_DMARDLADDR)

The EMAC_DMARDLADDR register points to the start of the receive descriptor list. The descriptor list is located in the host's physical memory and must be word-aligned. The DMA enables bus-width aligned address by making the corresponding LS bit low. Writing to the register is permitted only when the Rx DMA stops. After the Rx DMA stops, this register must be written before the receive start command is given.

Writing to the register is permitted only when the Rx DMA stops. In other words, the bit 1 (SR) is set 0 in the operation mode register. After the Rx DMA stops, this register can be written with a new descriptor list address.

When the SR bit is set, the DMA uses the newly programmed descriptor base address.

If the SR is cleared and this register remains unchanged, then the DMA will use the previous descriptor address when the Rx DMA stops.

Bit	Register	Reset value	Type	Description
Bit 31: 0	SRL	0x0000 0000	rw	Start of Receive List These bits contain the base address of the first descriptor in the receive descriptor list. The LSB bits (1: 0, 2: 0 or 3: 0) for 32/64/128-bit bus width are ignored and taken as zero by the DMA. Therefore these LSB bits are read only.

26.3.25 Ethernet DMA transmit descriptor list address register (EMAC_DMATDLADDR)

The EMAC_DMATDLADDR register points to the start of the transmit descriptor list. The descriptor list is located in the host's physical memory and must be word-aligned. The DMA enables bus-width aligned address by making the corresponding LS bit low.

Writing to the register is permitted only when the Tx DMA stops. In other words, the bit 13 (ST) is set 0 in the register 6 (operation mode register). After the Tx DMA stops, this register can be written with a new descriptor list address.

When the SR bit is set, the DMA uses the newly programmed descriptor base address.

If the SR is cleared and this register remains unchanged, then the DMA will use the previous descriptor address when the Tx DMA stops.

Bit	Register	Reset value	Type	Description
Bit 31: 0	STL	0x0000 0000	rw	Start of Transmit List These bits contain the base address of the first descriptor in the transmit descriptor list. The LSB bits (1: 0, 2: 0 or 3: 0) for 32/64/128-bit bus width are ignored and taken as zero by the DMA. Therefore these LSB bits are read only.

26.3.26 Ethernet DMA status register (EMAC_DMASTS)

The EMAC_DMASTS register contains all the status bits the DMA reports to the host. This register is read by the software driver during an interrupt service routine or polling. Most of the bits in this register can trigger the host to be interrupted. The bits in this register cannot be cleared when read. Writing 1'b1 to the bit [16: 0] (unreserved) in this register clears them. Writing 1'b0 has no effect. Each bit (bit [16: 0]) can be masked through the corresponding bit in the interrupt enable mask register.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value.
Bit 29	TTI	0x0	ro	Timestamp Trigger Interrupt This bit indicates an interrupt event in the time stamp generator block. The software must read the corresponding register to get interrupt sources. This bit is applicable only when the IEEE1588 time stamp feature is enabled. Otherwise, this bit is reserved.
Bit 28	MPI	0x0	ro	MAC PMT Interrupt This bit indicates an interrupt even in the PMT. The software must read the Ethernet PMT control and status register (EMAC_MACPMTCTRLSTS) to get the interrupt sources and clear them in order to reset this bit to 1'b0. This bit is applicable only when the PMT function is enabled. Otherwise, this bit is reserved.
Bit 27	MMI	0x0	ro	MAC MMC Interrupt This bit indicates an interrupt event in the MMC. The software must read the corresponding register to get interrupt sources and clear them in order to reset this bit to 1'b0. This bit is applicable only when the MAC MMC is enabled. Otherwise, this bit is reserved.
Bit 26	Reserved	0x0	resd	Kept at its default value.
Bit 25: 23	EB	0x0	ro	Error Bits These bits indicate the type of error that caused a bus error. They are applicable only when the bit 13 (FBI) is set. This filed does not generate an interrupt. 000: Error during data transfer by Rx DMA 011: Error during read transfer by Tx DMA 100: Error during Rx DMA descriptor write access 101: Error during Tx DMA descriptor write access 110: Error during Rx DMA descriptor read access 111: Error during Tx DMA descriptor read access Note: 001 and 010 are reserved.
Bit 22: 20	TS	0x0	ro	Transmit Process State

				This field indicates the Tx DMA FSM state. This field does not generate an interrupt. 3'b000: Stopped; Rest or Stop transmit command issued 3'b001: Running; Fetching transmit descriptor 3'b010: Running; Waiting for status 3'b011: Running; Reading data from host memory buffer and queuing it to Tx FIFO 3'b100: Time stamp write status 3'b101: Reserved for future use 3'b110: Suspended; Transmit descriptor unavailable or transmit buffer underflow 3'b111: Running; Closing transmit descriptor
Bit 19: 17	RS	0x0	ro	Receive Process State This field indicates the Rx DMA FSM state. This field does not generate an interrupt. 3'b000: Stopped; Rest or Stop transmit command issued 3'b001: Running; Fetching receive descriptor 3'b010: Reserved for future use 3'b011: Running; Waiting for receive packet 3'b100: Suspended; Receive descriptor unavailable 3'b101: Running; Closing receive descriptor 3'b110: Time stamp write status 3'b111: Running; Transferring the receive buffer data to host memory
Bit 16	NIS	0x0	rw1c	Normal Interrupt Summary The normal interrupt summary value is the logic OR of the following bits when the corresponding interrupt bits are enabled in the interrupt enable registers. EMAC_DMASTS[0]: Transmit interrupt EMAC_DMASTS[2]: Transmit buffer unavailable EMAC_DMASTS[6]: Receive interrupt EMAC_DMASTS[14]: Early receive interrupt Only unmasked bits affect the normal interrupt summary. This is a sticky bit and it must be cleared (by writing 1 to this bit) each time a corresponding bit (causes NIS to be set) is cleared.
Bit 15	AIS	0x0	rw1c	Abnormal Interrupt Summary The abnormal interrupt summary value is the logic OR of the following bits when the corresponding interrupt bits are enabled in the interrupt enable registers. EMAC_DMASTS[1]: Transmit process stopped EMAC_DMASTS[3]: Transmit Jabber timeout EMAC_DMASTS[4]: Receive FIFO overflow EMAC_DMASTS[5]: Transmit data underflow EMAC_DMASTS[7]: Receive buffer unavailable EMAC_DMASTS[8]: Receive process stopped EMAC_DMASTS[9]: Receive watchdog timeout EMAC_DMASTS[10]: Early transmit interrupt EMAC_DMASTS[13]: Fatal bus error Only unmasked bits affect the abnormal interrupt summary. This is a sticky bit and it must be cleared (by writing 1 to this bit) each time a corresponding bit (causes AIS to be set) is cleared.
Bit 14	ERI	0x0	rw1c	Early Receive Interrupt This bit indicates that the DMA has filled the first data buffer of the packet. This bit is cleared when the software writes 1 to this bit or when the bit 6 (RI) bit is set in this register. (Whichever occurs first)
Bit 13	FBEI	0x0	rw1c	Fatal Bus Error Interrupt This bit indicates that a bus error occurred as defined in bit [25: 23]. When this bit is set, the corresponding DMA will disable all its bus accesses.
Bit 12: 11	Reserved	0x0	resd	Kept at its default value.
Bit 10	ETI	0x0	rw1c	Early Transmit Interrupt

				This bit indicates that the frame to be transmitted was fully sent to the MTL Tx FIFO.
Bit 9	RWT	0x0	rw1c	Receive Watchdog Timeout When this bit is set, it indicates that the receive watchdog timer timeout occurs while receiving the current frame, and the current frame is cut off after the watchdog timeout happens.
Bit 8	RPS	0x0	rw1c	Receive Process Stopped This bit is set when the receive process enters the stop state.
Bit 7	RBU	0x0	rw1c	Receive Buffer Unavailable This bit indicates that the next descriptor in the receive list is owned by the host and cannot be acquired by the DMA. Thus the receive process is suspended. The host should change the ownership of the descriptor and release the receive poll demand command in order to resume receive process. If no receive poll demand command is issued, the receive process resumes when the DMA receives the next incoming frame. This bit is set only when the previous receive descriptor is owned by the DMA.
Bit 6	RI	0x0	rw1c	Receive Interrupt This bit indicates the completion of a frame reception. After the completion of a frame reception, the bit 31 of the RDES1 (interrupt disabled after reset operation) is reset in the last descriptor. Specific frame status information will be posted in the descriptor. Receive process remains in the running state.
Bit 5	UNF	0x0	rw1c	Transmit Underflow This bit indicates that the transmit buffer has an underflow during a frame transmission. Transmit process is suspended and the underflow error bit TDES0[1] is set.
Bit 4	OVF	0x0	rw1c	Receive Overflow This bit indicates that the receive buffer has an overflow during a frame reception. If the partial frame has been transferred to the application, the overflow status is set in the RDES0[11].
Bit 3	TJT	0x0	rw1c	Transmit Jabber Timeout This bit indicates that the transmit Jabber timer will expire when the current frame is greater than 2048 bytes (it is 10240 bytes if Jumbo frame is enabled). After the Jabber is expired, the transmit process is aborted and enters stop state, which causes the transmit Jabber timeout flag bit TDES0[14] to be set.
Bit 2	TBU	0x0	rw1c	Transmit Buffer Unavailable This bit indicates that the next descriptor in the transmit list is owned by the host and cannot be acquired by the DMA. Then the transmit process is suspended. Bit [22: 20] explains the transmit process state. To resume transmit process, the host should change the ownership of the descriptor by setting the TDES0[31] and issue the transmit poll demand command
Bit 1	TPS	0x0	rw1c	Transmit Process Stopped This bit is set when the transmit process stops.
Bit 0	TI	0x0	rw1c	Transmit Interrupt This bit indicates the completion of a frame transmission. The bit 31 (OWN) is reset in the TDES0. Specific frame status information will be posted in the descriptor.

26.3.27 Ethernet DMA operation mode register (EMAC_DMAOPM)

The EMAC_DMAOPM register defines the receive and transmit operation modes and commands. This register should be the last CSR to be written during DMA initialization. This register is also applicable to GMAC-MTL configuration where the unused and reserved bits are 24, 13, 2 and 1. A delay value greater than 4us is required between two consecutive write accesses to this register.

Bit	Register	Reset value	Type	Description
Bit 31: 27	Reserved	0x00	resd	Kept at its default value.
Bit 26	DT	0x0	rw	Disable Dropping of TCP/IP Checksum Error Frames When this bit is set, the MAC does not drop the frames that only have errors detected by the receive checksum offload engine. Such frames have errors in the encapsulated payload only but do not have errors (including FCS error) in the Ethernet frames received by the MAC. When this bit is cleared, all error frames are dropped if the FEF bit is reset.
Bit 25	RSF	0x0	rw	Receive Store and Forward When this bit is set, the MTL reads the Rx FIFO only after a full frame is written to the Rx FIFO, ignoring the RTC bit. When this bit is cleared, the Rx FIFO operates in cut-through mode and will be subject to the threshold defined by the RTC.
Bit 24	DFRF	0x0	rw	Disable Flushing of Received Frames When this bit is set, the Rx DMA does not flush any receive frame due to the unavailability of receive descriptors or receive buffers. When this bit is cleared, the DMA will flush receive frames in case of the above-mentioned circumstances.
Bit 23: 22	Reserved	0x000	resd	Kept at its default value.
Bit 21	TSF	0x0	rw	Transmit Store and Forward When this bit is set, transmission starts when a full frame resides in the Tx FIFO, and the TTC values specified in the bit [16: 14] are ignored. This bit can be changed only when the transmit process stops.
Bit 20	FTF	0x0	rw	Flush Transmit FIFO When this bit is set, the Tx FIFO controller logic is reset of its default values and thus all data in the Tx FIFO are either lost or flushed. This bit is cleared after the completion of the flushing operation. The operation mode register should not be written before this bit is cleared. The data that has been received by the MAC transmitter is not flushed and is going to be transferred, causing data underflow and runt frame transfer (If you want to change this bit through consecutive commands, a delay value greater than 4us is required between two consecutive operations.)
Bit 19: 17	Reserved	0x0	resd	Kept at its default value.
Bit 16: 14	TTC	0x0	rw	Transmit Threshold Control These bits control the threshold of the Tx FIFO. Transmission starts when the frame size in the Tx FIFO is greater than the threshold. In addition, full frames with a length less than the threshold are also transmitted. These bits are applicable only when the bit 21 (TSF) is reset. 000: 64 001: 128 010: 192 011: 256 100: 40 101: 32 110: 24 111: 16
Bit 13	SSTC	0x0	rw	Start or Stop Transmission Command When this bit is set, transmission is in the running state, and the DMA checks the transmit list at the current location and determines the frame to be transmitted. The DMA acquires the descriptor either from the current position in the list (the transmit list base address set by the transmit descriptor list address register) or from the position where the transmit process was stopped previously. If the current descriptor is owned by the DMA, the transmit process enters suspend state, and the bit 2 (transmit buffer

				unavailable) is set in the statue register. Transmission command is valid only when the transmission is stopped. If the transmit command were issued before setting the transmit descriptor list address register, the DMA will show unpredictable behavior. When this bit is cleared, transmit process enters stop state after the completion of a frame transmission. The next descriptor position in the transmit list is saved, and becomes the current position when transmission gets started. To change the list address, write a new value to the transmit descriptor list address register when this bit is reset. The newly written value becomes effective only when this bit is set again. The Stop Transmission Command is effective only when the current frame transmission is complete or transmit process enters suspend state.
Bit 12: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7	FEF	0x0	rw	Forward Error Frames 1: All frames except runt error frames are forwarded to the DMA 0: Rx FIFO drops error frames (CRC error, collision error, giant frame, watchdog timeout and overflow). However, if the frame's start byte point has already been transferred to the application in Threshold mode, then the frames are not dropped. The Rx FIFO drops the error frames whose start bytes have not been transferred to the AHB bus.
Bit 6	FUGF	0x0	rw	Forward Undersized Good Frames When this bit is set, the Rx FIFO forwards undersized good frames including pad bytes and CRC (with no error and length less than 64 bytes). When this bit is cleared, the Rx FIFO drops all frames with a length less than 64 bytes, unless such a frame has already been transferred to the application due to a lower value than the receive threshold (e.g. RTC=01).
Bit 5	Reserved	0x0	resd	Kept at its default value.
Bit 4: 3	RTC	0x0	rw	Receive Threshold Control These two bits control the threshold of the Rx FIFO. Transfer to DMA starts when the frame in the Rx FIFO is larger than the threshold. In addition, full frames with a length less than the threshold are also automatically transferred. Value 11 is not applicable if the Rx FIFO size is configured to be 128 bytes. These bits are applicable only when the RSF bit equals 0. These bits are ignored when the RSF bit is set. 00: 64 01: 32 10: 96 11: 128
Bit 2	OSF	0x0	rw	Operate on Second Frame When this bit is set, it instructs the DMA to process a second frame of transmit data even before the status of the first frame is obtained.
Bit 1	SSR	0x0	rw	Start or Stop Receive When this bit is set, the receive process is in the running state, and the DMA attempts to acquire the descriptor from the receive list and processes incoming frames. The DMA acquires the descriptor either from the current position in the list (the receive list base address set by the receive descriptor list address register) or from the position where the receive process was stopped previously. If the current descriptor is owned by the DMA, the receive process enters suspend state, and the bit 7 (receive buffer unavailable) is set in the statue register. Reception command is valid only when the reception is stopped. If

				the reception command were issued before setting the receive descriptor list address register, the DMA will show unpredictable behavior. When this bit is cleared, Rx DMA operation is stopped after the completion of a frame reception. The next descriptor position in the receive list is saved, and becomes the current position when reception process is restarted. The Stop Rece[topm Command is effective only when the receive process enters in the running state (waiting for receive packet) or the suspend state.
Bit 0	Reserved	0x0	resd	Kept at its default value.

26.3.28 Ethernet DMA interrupt enable register (EMAC_DMAIE)

The EMAC_DMAIE register enables the interrupts reported by the status register. Setting a bit to 1'b1 enables a corresponding interrupt. All interrupts are disabled after a software or hardware reset.

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	NIE	0x0	rw	Normal Interrupt enable When this bit is set, a normal interrupt summary is enabled. When this bit is cleared, a normal interrupt summary is disabled. This bit enables the following bits (in the statue register) EMAC_DMASTS[0]: Transmit interrupt EMAC_DMASTS[2]: Transmit buffer unavailable EMAC_DMASTS[6]: Receive interrupt EMAC_DMASTS[14]: Early receive interrupt
Bit 15	AIE	0x0	rw	Abnormal interrupt enable When this bit is set, an abnormal interrupt summary is enabled. When this bit is cleared, an abnormal interrupt summary is disabled. This bit enables the following bits (in the status register) EMAC_DMASTS[1]: Transmit process stopped EMAC_DMASTS[3]: Transmit Jabber timeout EMAC_DMASTS[4]: Transmit overflow EMAC_DMASTS[5]: Transmit data underflow EMAC_DMASTS[7]: Transmit buffer u unavailable EMAC_DMASTS[8]: Receive process stopped EMAC_DMASTS[9]: Receive watchdog timeout EMAC_DMASTS[10]: Early transmit interrupt EMAC_DMASTS[13]: Fatal bus error
Bit 14	ERE	0x0	rw	Early Receive interrupt Enable When this bit is set with the normal interrupt summary enable bit, the early receive interrupt is enabled. When this bit is cleared, the early receive interrupt is disabled.
Bit 13	FBEE	0x0	rw	Fatal Bus Error Enable When this bit is set with the abnormal interrupt summary enable bit, the fatal bus error interrupt is enabled. When this bit is cleared, the fatal bus error enable interrupt is disabled.
Bit 12: 11	Reserved	0x0	resd	Kept at its default value.
Bit 10	EIE	0x0	rw	Early transmit Interrupt Enable When this bit is set with the abnormal interrupt summary enable bit, the early transmit interrupt is enabled. When this bit is cleared, the early transmit interrupt is disabled.
Bit 9	RWTE	0x0	rw	Receive Watchdog Timeout Enable When this bit is set with the abnormal interrupt summary enable bit, the receive watchdog timeout interrupt is enabled. When this bit is cleared, the receive watchdog timeout interrupt is disabled.
Bit 8	RSE	0x0	rw	Receive Stopped Enable When this bit is set with the abnormal interrupt summary enable bit, the receive stopped interrupt is enabled. When this bit is cleared, the receive stopped interrupt is disabled.

Bit 7	RBUE	0x0	rw	Receive Buffer Unavailable Enable When this bit is set with the abnormal interrupt summary enable bit, the receive buffer unavailable interrupt is enabled. When this bit is cleared, the receive buffer unavailable interrupt is disabled.
Bit 6	RIE	0x0	rw	Receive Interrupt Enable When this bit is set with the normal interrupt summary enable bit, the receive interrupt is enabled. When this bit is cleared, the receive interrupt is disabled.
Bit 5	UNE	0x0	rw	Underflow Interrupt Enable When this bit is set with the abnormal interrupt summary enable bit, the underflow interrupt is enabled. When this bit is cleared, the underflow interrupt is disabled.
Bit 4	OVE	0x0	rw	Overflow Interrupt Enable When this bit is set with the abnormal interrupt summary enable bit, the overflow interrupt is enabled. When this bit is cleared, the overflow interrupt is disabled.
Bit 3	TJE	0x0	rw	Transmit Jabber Timeout Enable When this bit is set with the abnormal interrupt summary enable bit, the transmit Jabber timeout interrupt is enabled. When this bit is cleared, the transmit Jabber timeout interrupt is disabled.
Bit 2	TUE	0x0	rw	Transmit Buffer Unavailable Enable When this bit is set with the normal interrupt summary enable bit, the transmit buffer unavailable interrupt is enabled. When this bit is cleared, the transmit buffer unavailable interrupt is disabled.
Bit 1	TSE	0x0	rw	Transmit Stopped Enable When this bit is set with the abnormal interrupt summary enable bit, the transmit stopped interrupt is enabled. When this bit is cleared, the transmit stopped interrupt is disabled.
Bit 0	TIE	0x0	rw	Transmit Interrupt Enable When this bit is set with the normal interrupt summary enable bit, the transmit interrupt is enabled. When this bit is cleared, the transmit interrupt is disabled.

The Ethernet interrupt is generated only when the TST or PMT bit is set in the DMA status register with other interrupts unmasked, ow when the NIS/AIS is enabled with other interrupts enabled.

26.3.29 Ethernet DMA missed frame and buffer overflow counter register (EMAC_DMAMFBOCNT)

The DMA contains two counters to track the number of missed frames during reception. Tis register reports the current value of the counter. The counter is used for the purpose of diagnosis. The bit [15: 0] indicates the number of missed frames due to the host buffer being unavailable. The bit [27: 17] indicate the number of missed frames due to buffer overflow (MTL and MAC) and runt frames dropped by the MTL.

Bit	Register	Reset value	Type	Description
Bit 31: 29	Reserved	0x0	resd	Kept at its default value.
Bit 28	OBFOC	0x0	rrc	Overflow Bit for FIFO Overflow Counter This bit is set whenever an overflow occurs on the overflow frame counter ([27: 17]), that is, the Rx FIFO overflows, and the overflow frame counter reaches its maximum value. In this case, the overflow frame counter is reset to all zero, and this bit indicates that a toggle has occurred.
Bit 27: 17	OFC	0x000	rrc	Overflow Frame Counter These bits indicate the number of frames missed by the application.
Bit 16	OBMFC	0x0	rrc	Overflow Bit for Missed Frame Counter This bit is set whenever an overflow occurs on the missed frame counter ([15: 0]), that is, the DMA ignores incoming frames due to the host receive buffer being unavailable, and the missed frame counter reaches its maximum value.

				In this case, the missed frame counter is reset to all zero, and this bit indicates that a toggle has occurred.
Bit 15: 0	MFC	0x0000	rrc	Missed Frame Counter This field indicates the number of frames missed by the controller due to the host receive buffer being unavailable. This counter is incremented each time the DMA discards an incoming frame.

26.3.30 Ethernet DMA current transmit descriptor register (EMAC_DMACTD)

The EMAC_DMACTD register points to the start address of the transmit descriptor being read by the DMA.

Bit	Register	Reset value	Type	Description
Bit 31: 0	HTDAP	0x0000 0000	ro	Host Transmit Descriptor Address Pointer These bits are cleared when reset. The DMA updates the pointer during operation.

26.3.31 Ethernet DMA current receive descriptor register (EMAC_DMACRD)

The EMAC_DMACRD register points to the start address of the receive descriptor being read by the DMA.

Bit	Register	Reset value	Type	Description
Bit 31: 0	HRDAP	0x0000 0000	ro	Host Receive Descriptor Address Pointer These bits are cleared when reset. The DMA updates the pointer during operation.

26.3.32 Ethernet DMA current transmit buffer address register (EMAC_DMACTBADDR)

The EMAC_DMACTBADDR register points to the transmit buffer address being read by the DMA.

Bit	Register	Reset value	Type	Description
Bit 31: 0	HTBAP	0x0000 0000	ro	Host Transmit Buffer Address Pointer These bits are cleared when reset. The DMA updates the pointer during operation.

26.3.33 Ethernet DMA current receive buffer address register (EMAC_DMACRBADDR)

The EMAC_DMACRBADDR register points to the receive buffer address being read by the DMA.

Bit	Register	Reset value	Type	Description
Bit 31: 0	HRBAP	0x0000 0000	ro	Host Receive Buffer Address Pointer These bits are cleared when reset. The DMA updates the pointer during operation.

26.3.34 Ethernet MMC control register (EMAC_MMCCTRL)

The EMAC_MMCCTRL register defines the operating mode of the management counters.

Bit	Register	Reset value	Type	Description
Bit 31: 4	Reserved	0x0000000	resd	Kept at its default value.
Bit 3	FMC	0x0	rw	Freeze MMC Counter When this bit is set, it freezes all the MMC counters to their current value. None of the MMC counters are updated due to any transmitted or received frame until this bit is set to 0. If the Reset on Read bit is set while the MMC counter is being read, the counter is also cleared.
Bit 2	RR	0x0	rw	Reset on Read When this bit is set, the MMC counter is reset to 0 after being read. The counter is cleared when the least significant byte bit [7: 0] is read.

Bit 1	SCR	0x0	rw	Stop Counter Rollover When this bit is set, the counter does not roll over to 0 after it reaches the maximum value.
Bit 0	RC	0x0	rw	Reset Counter When this bit is set, all counters are reset. This bit is cleared automatically after 1 clock cycle.

26.3.35 Ethernet MMC receive interrupt register (EMAC_MMCRI)

The EMAC_MMCRI register contains the interrupts generated in the following conditions:

- Receive statistic counters reaches half their maximum values (32-bit counter corresponds to 0x8000_0000, and 16-bit counter corresponds to 0x8000)
- Receive statistic counters exceed their maximum values (32-bit counter corresponds to 0xFFFF_FFFF, and 16-bit counter corresponds to 0xFFFF)

When the counter stops rolling, an interrupt is set but the counter is still all 1. The EMAC_MMCRI is a 32-bit register. An interrupt bit is cleared when the MMC counter that generates the interrupt is read. The least significant byte bit [7: 0] of the corresponding counter must be read in order to clear the interrupt bit.

Bit	Register	Reset value	Type	Description
Bit 31: 18	Reserved	0x0000	resd	Kept at its default value.
Bit 17	RGUF	0x0	rrc	Received Good Unicast Frames This bit is set when the received good unicast frame counter reaches the maximum value or half the maximum value.
Bit 16: 7	Reserved	0x000	resd	Kept at its default value.
Bit 6	RFAE	0x0	rrc	Received Frames Alignment Error This bit is set when the received frame counter with alignment error reaches the maximum value or half the maximum value.
Bit 5	RFCE	0x0	rrc	Received Frames CRC Error This bit is set when the receive frame with CRC error reaches the maximum value or half the maximum value.
Bit 4: 0	Reserved	0x00	resd	Kept at its default value.

26.3.36 Ethernet MMC transmit interrupt register (EMAC_MMCTI)

The EMAC_MMCTI register contains the interrupts generated in the following conditions: when the transmit statistic counters reach half their maximum values (32-bit counter corresponds to 0x8000_0000, and 16-bit counter corresponds to 0x8000), and when the transmit statistic counters exceed their maximum values (32-bit counter corresponds to 0xFFFF_FFFF, and 16-bit counter corresponds to 0xFFFF). When the counter stops rolling, an interrupt is set but the counter is still all 1. The EMAC_MMCTI is a 32-bit register. An interrupt bit is cleared when the MMC counter that generates the interrupt is read. The least significant byte bit [7: 0] of the corresponding counter must be read in order to clear the interrupt bit.

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x000	resd	Kept at its default value.
Bit 21	TGF	0x0	rrc	Transmitted Good Frames This bit is set when the transmitted good frame counter reaches its maximum value or half its maximum value.
Bit 20: 16	Reserved	0x00	resd	Kept at its default value.
Bit 15	TGFMSC	0x0	rrc	Transmitted Good Frames More Single Collision This bit is set when the transmitted good frame after more than a single collision counter reaches its maximum value or half its maximum value.
Bit 14	TSCGFCI	0x0	rrc	Transmitted Single Collision Good Frame Counter Interrupt This bit is set when the transmitted good frame after a single collision counter reaches its maximum value or half its maximum value.
Bit 13: 0	Reserved	0x0000	resd	Kept at its default value.

26.3.37 Ethernet MMC receive interrupt register (EMAC_MMCRIM)

The EMAC_MMCRIM contains the masks for interrupts generate when the receive statistic counters reach half their maximum values or their maximum values. This register is a 32-bit register.

Bit	Register	Reset value	Type	Description
Bit 31: 18	Reserved	0x0000	resd	Kept at its default value.
Bit 17	RUGFCIM	0x0	rw	Received Unicast Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the received good unicast frame counter reaches half its maximum value or its maximum value.
Bit 16: 7	Reserved	0x000	resd	Kept at its default value.
Bit 6	RAEFACIM	0x0	rw	Received Alignment Error Frame Alignment Counter Interrupt Mask Setting this bit masks the interrupt when the received alignment error frame counter reaches half its maximum value or its maximum value.
Bit 5	RCEFCIM	0x0	rw	Received CRC Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the received CRC error frame counter reaches half its maximum value or its maximum value.
Bit 4: 0	Reserved	0x00	resd	Kept at its default value.

26.3.38 Ethernet MMC transmit interrupt register (EMAC_MMCTIM)

The EMAC_MMCTIM contains the masks for interrupts generate when the transmit statistic counters reach half their maximum values or their maximum values. This register is a 32-bit register.

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x000	resd	Kept at its default value.
Bit 21	TGFCIM	0x0	rw	Transmitted Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the transmitted good frame counter reaches half its maximum value or its maximum value.
Bit 20: 16	Reserved	0x00	resd	Kept at its default value.
Bit 15	TMCGFCIM	0x0	rw	Transmitted Multiple Collision Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the transmitted good frame after more than a single collision counter reaches half its maximum value or its maximum value.
Bit 14	TSCGFCIM	0x0	rw	Transmitted Single Collision Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the transmitted good frame after a single collision counter reaches half its maximum value or its maximum value.
Bit 13: 0	Reserved	0x0000	resd	Kept at its default value.

26.3.39 Ethernet MMC transmitted good frame single collision counter register (EMAC_MMCTFSCC)

This register maintains the number of successfully transmitted frames after a single collision in half-duplex mode.

Bit	Register	Reset value	Type	Description
Bit 31: 0	TGFSCC	0x0000 0000	ro	Transmitted Good Frames Single Collision Counter) This field maintains the transmitted good frames after a single collision counter.

26.3.40 Ethernet MMC transmitted good frame more than a single collision counter register (EMAC_MMCTFMSCC)

This register maintains the number of successfully transmitted frames after more than a single collision in half-duplex mode.

Bit	Register	Reset value	Type	Description
Bit 31: 0	TGFMSCC	0x0000 0000	ro	Transmitted Good Frame More Than a Single Collision Counter This field maintains the transmitted good frames after more than a single collision counter.

26.3.41 Ethernet MMC transmitted good frames counter register (EMAC_MMCTFCNT)

This register maintains the number of the transmitted good frames.

Bit	Register	Reset value	Type	Description
Bit 31: 0	TGFC	0x0000 0000	ro	Transmitted Good Frames Counter

26.3.42 Ethernet MMC received frames with CRC error counter register (EMAC_MMCRFCECR)

This register maintains the number of the received good frames with CRC error.

Bit	Register	Reset value	Type	Description
Bit 31: 0	RFCEC	0x0000 0000	ro	Received Frames CRC Error Counter Received frames with CRC error.

26.3.43 Ethernet MMC received frames with alignment error counter register (EMAC_MMCRFAECNT)

This register maintains the number of the received frames with alignment error.

Bit	Register	Reset value	Type	Description
Bit 31: 0	RFAEC	0x0000 0000	ro	Received Frames Alignment Error Counter Received frames with alignment error.

26.3.44 Ethernet MMC received good unicast frames counter register (EMAC_MMCRGUFCNT)

This register maintains the number of the received good unicast frames.

Bit	Register	Reset value	Type	Description
Bit 31: 0	RGUFC	0x0000 0000	ro	Received Good Unicast Frames Counter

26.3.45 Ethernet PTP time stamp control register (EMAC_PTPTSCTRL)

This register controls the generation of system time in the receiver and the generation of time stamp in a PTP packet.

Bit	Register	Reset value	Type	Description
Bit 31: 19	Reserved	0x0000	resd	Kept at its default value.
Bit 18	EMAFPFF	0x0	rw	Enable MAC Address For PTP Frame Filtering When this bit is set, the MAC address (matches any of the MAC address registers) is used for PTP frame filtering while the PTP is directly sent by the Ethernet.
Bit 17: 16	SPPFTS	0x0	rw	Select PTP Packets For Taking Snapshot 00: Normal clock 01: Boundary clock 10: End-to-End Transparent Clock 11: Point-to-Point Transparent Clock
Bit 15	ESFMRTM	0x0	rw	Enable Snapshot For Message Relevant To Master When this bit is set, it enables snapshots for messages relevant to master. Otherwise, it enables snapshots for

				messages relevant to slave.
Bit 14	ETSFEM	0x0	rw	Enable Timestamp Snapshot For Event Messages When this bit is set, it enables time stamp snapshots for event messages only (SYNC, Delay_Req, Pdelay_Req, or Pdelay_Resp). When this bit is cleared, time stamp snapshots are applicable to all the messages except Announce, Management and Signaling.
Bit 13	EPPFSIP4U	0x1	rw	Enable Processing of PTP Frames Sent over IPv4-UDP When this bit is set, the MAC receiver processes the PTP encapsulated in UDP over IPv4 packet. When this bit is cleared, the MAC ignores the PTP transferred over UDP-IPv4 packet. This bit is set by default.
Bit 12	EPPFSIP6U	0x0	rw	Enable Processing of PTP Frames Sent over IPv6-UDP When this bit is set, the MAC receiver processes the PTP encapsulated in UDP over IPv6 packet. When this bit is cleared, the MAC ignores the PTP sent over UDP-IPv6 packet.
Bit 11	EPPEF	0x0	rw	Enable Processing of PTP over EMAC Frames When this bit is set, the MAC receiver processes the PTP that is directly encapsulated in the Ethernet frames. When this bit is cleared, the MAC ignores the PTP over EMAC frames.
Bit 10	EPPV2F	0x0	rw	Enable PTP packet Processing for Version 2 Format When this bit is set, it enables PTP packet processing in the format of 1588 V2. Otherwise, 1588 V1 format is used for PTP packet processing. Refer to *PTP process and control* on page 155 for more details on IEEE 1588 V1 and V2.
Bit 9	TDBRC	0x0	rw	Timestamp Digital or Binary Rollover Control When this bit is set, time stamp low register starts rolling after the 0x3B9A_C9FF value (1 ns precision), and the time stamp (high) second is incremented. When this bit is cleared, the rollover value of the subsecond register is 0x7FFF_FFFF. The subsecond increment must be configured according to PTP reference clock frequency and the value of this bit.
Bit 8	ETAF	0x0	rw	Enable Timestamp for All Frames When this bit is set, it enables time stamp snapshot for all received frames on the MAC.
Bit 7: 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	ARU	0x0	rw	Addend Register Update When this bit is set, the Ethernet PTP time stamp addend register's contents are updated on the PTP block for fine correction. This bit is cleared when the update is completed. This register bit must be read as 0 before being set.
Bit 4	TITE	0x0	rw	Timestamp Interrupt Trigger Enable When this bit is set, a time stamp interrupt is enabled if the system time becomes greater than the value written in the target time register. This bit is cleared when the time stamp trigger interrupt is generated.
Bit 3	TU	0x0	rw	Timestamp Update When this bit is set, the system time is updated (added or subtracted from) with the value programmed in the system time second update register and system time nanosecond update register. This bit must be read as 0 before being updated. This bit is cleared after the hardware update is completed. Time stamp high word register (if enabled) is not updated.
Bit 2	TI	0x0	rw	Timestamp Initialize When this bit is set, the system time is initialized (overwritten) with the value specified in the system time second update register and system time nanosecond update register.

				This bit must be read as 0 before being updated. This bit is cleared after the initialization. Time stamp high word register (if enabled) is not updated.
Bit 1	TFCU	0x0	rw	Timestamp Fine or Coarse Update When this bit is set, it indicates that the system time is updated using a fine update method. When this bit is cleared, it indicates that the system time is updated using a coarse update method.
Bit 0	TE	0x0	rw	Timestamp Enable When this bit is set, time stamp function is enabled for transmit and receive frames. Once disabled, the time stamp function is not added for transmit and receive frames, and the time stamp generator is suspended as well. Once enabled, the time stamp (system time) should be initialized. On the receive side, the MAC processes 1588 frames only when this bit is set.

Correlation between time stamp snapshot and register bits

SPPFTS Bit 17: 16	ESFMRTM Bit 15	ETSFEM Bit 14	PTP message
00 or 01	X	0	SYNC, Follow_Up, Delay_Req, Delay_Resp
00 or 01	1	1	Delay_Req
00 or 01	0	1	SYNC
10	N/A	0	SYNC, Follow_Up, Delay_Req, Delay_Resp
10	N/A	1	SYNC, Follow_Up
11	N/A	0	SYNC, Follow_Up, Delay_Req, Delay_Resp, Pdelay_Req, Pdelay_Resp
11	N/A	1	SYNC, Pdelay_Req, Pdelay_Resp

1 : N/A= Not applicable

2 : X=Irrelevant

26.3.46 Ethernet PTP subsecond increment register (EMAC_PTPSSINC)

This register is present only when the IEEE1588 time stamp function is selected without an external time stamp input. In Coarse Update mode (TSCFUPDT bit), the value in this register is added to the system time every clk_ptp_ref_i clock cycle. In Fine Update mode, the value in this register is added to the system time whenever the accumulator has an overflow.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7: 0	SSIV	0x00	rw	Sub-Second Increment Value The value programmed in this field is incremented with the value of the subsecond register at every clock cycle (of clk_ptp_i). For example, if the PTP clock is 50 MHz (20 ns), when the system time nanosecond register is 1 ns accuracy (by setting the bit 9 in the EMAC_PTPTSCTRL register), the value of these bits should be configured to 20 (0x14). When the TSCTRLSSR is cleared, nanosecond register resolution is ~0.465ns accuracy. In this case, the value of these bits should be configured to 43 (0x2B), that is, 20ns/0.465.

26.3.47 Ethernet PTP time stamp high register (EMAC_PTPTSH)

System time second register and system time nanosecond register indicate the current value of the system time maintained by the MAC. This value is updated on a continuous basis.

Bit	Register	Reset value	Type	Description
Bit 31: 0	TS	0x0000 0000	ro	Timestamp Second This field indicates the second value of the current system time maintained by the MAC.

26.3.48 Ethernet PTP time stamp low register (EMAC_PTPTSL)

This register contains the lower 32 time bits. It is a read-only register containing the subsecond system time value.

Bit	Register	Reset value	Type	Description
Bit 31	AST	0x0	ro	Add or Subtract Time When this bit is set, the time value is subtracted from the value of the update register. When this bit is cleared, the time value is added to the value of the update register.
Bit 30: 0	TSS	0x0000 0000	ro	Timestamp Sub Seconds This field indicates the subsecond system time with 0.46 ns accuracy. When the bit 9 is set in the EMAC_PTPTSCTRL register, each bit represents 1 ns, with the value programmed not exceeding the 0x3B9A_C9FF.

26.3.49 Ethernet PTP time stamp high update register (EMAC_PTPTSHUD)

System time second update register and system nanosecond update register initializes or updates the system time maintained by the MAC. It is required to write both registers before setting the TSINIT or TSUPDT bit in the EMAC_PTPTSCTRL register.

Bit	Register	Reset value	Type	Description
Bit 31: 0	TS	0x0000 0000	rw	Timestamp Second This field indicates the second value that is to be initialized or added to the system time.

26.3.50 Ethernet PTP time stamp low update register (EMAC_PTPTSLUD)

This register is present only when the IEEE1588 time stamp function is selected without an external time stamp input.

Bit	Register	Reset value	Type	Description
Bit 31	AST	0x0	rw	Add or Subtract Time When this bit is set, the time value is subtracted from the value of the update register. When this bit is cleared, the time value is added to the value of the update register.
Bit 30: 0	TSS	0x0000 0000	rw	Timestamp Sub Seconds This field indicates the subsecond system time with 0.46 ns accuracy. When the bit 9 is set in the EMAC_PTPTSCTRL register, each bit represents 1 ns, with the value programmed not exceeding the 0x3B9A_C9FF.

26.3.51 Ethernet PTP time stamp addend register (EMAC_PTPTSAD)

This register value is used only when the system time is configured for Fine update mode. This register value is added to a 32-bit accumulator at every clock cycle (of clk_ptp_ref_i). The system time is updated whenever the accumulator overflows.

Bit	Register	Reset value	Type	Description
Bit 31: 0	TAR	0x0000 0000	rw	Timestamp Addend Register This field indicates the 32-bit time value to be added to the accumulator in order to achieve time synchronization.

26.3.52 Ethernet PTP target time high register (EMAC_PTPTTH)

Target time second register and target time subsecond register are used to schedule an interrupt event when the system time exceeds the value programmed in these registers.

Bit	Register	Reset value	Type	Description
Bit 31: 0	TTSR	0x0000 0000	rw	Target Time Seconds Register This register stores the time value in seconds. When the time stamp value equals or exceeds both target time stamp registers, the MAC starts or stops PPS signal output depending on the bit [6: 5] of the PPS control register. An interrupt is generated if enabled.

26.3.53 Ethernet PTP target time low register (EMAC_PTPTTL)

Bit	Register	Reset value	Type	Description
Bit 31: 0	TTLR	0x0000 0000	rw	Target Timestamp Low Register This register stores the time (signed) in nanoseconds. When the value of the time stamp equals both target time stamp registers, the MAC starts or stops PPS signal output depending on the TRGTMODSEL0 (bit [6: 5]) of the PPS control register. An interrupt is generated if enabled. When the bit 9 (TSCTRLSSR) is set in the MAC_PTPTSCTR register, the value of this field cannot exceed the 0x3B9A_C9FF. The actual time that starts or stops PPT signal output may have an error of up to 1 subsecond increment value.

26.3.54 Ethernet PTP time stamp status register (EMAC_PTPTSSR)

Bit	Register	Reset value	Type	Description
Bit 31: 2	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 1	TTTR	0x0	ro	Timestamp Target Time Reached When this bit is set, it indicates the value programmed when the system time equals or exceeds the target time second register and target time nanosecond register.
Bit 0	TSO	0x0	ro	Timestamp Seconds Overflow When this bit is set, it indicates that the time stamp value (V2 format supported) overflows and has exceeded the 32'hFFFF_FFFF.

26.3.55 Ethernet PTP PPS register (EMAC_PTPPPSCR)

Bit	Register	Reset value	Type	Description
Bit 31: 4	Reserved	0x0000000	resd	Kept at its default value.
Bit 3: 0	POFC	0x0	rw	PPS0 Output Frequency Control The output of this field depends on the emac_pps_sel bit (bit 15 in the CRM_MISC2 register) Emac_pps_sel=0: 0000: 1 Hz, use binary rollover control, pulse width is 125 ms; use digital rollover, pulse width is 100 ms 0001: 2 Hz, use binary rollover control, duty cycle is 50% (digital rollover is not recommended) 0010: 4 Hz, se binary rollover control, duty cycle is 50% (digital rollover is not recommended) 0011: 8 Hz, use binary rollover control, duty cycle is 50%(digital rollover is not recommended) 0100: 16 Hz, use binary rollover control, duty cycle is 50% (digital rollover is not recommended) 1111: 32.768 kHz, use binary rollover control, duty cycle is 50% (digital rollover is not recommended) Emac_pps_sel=1: 0000: 1 Hz, pulse width is one clk_ptp cycle 0001: For binary rollover, 2hz, duty cycle 50%; For digital

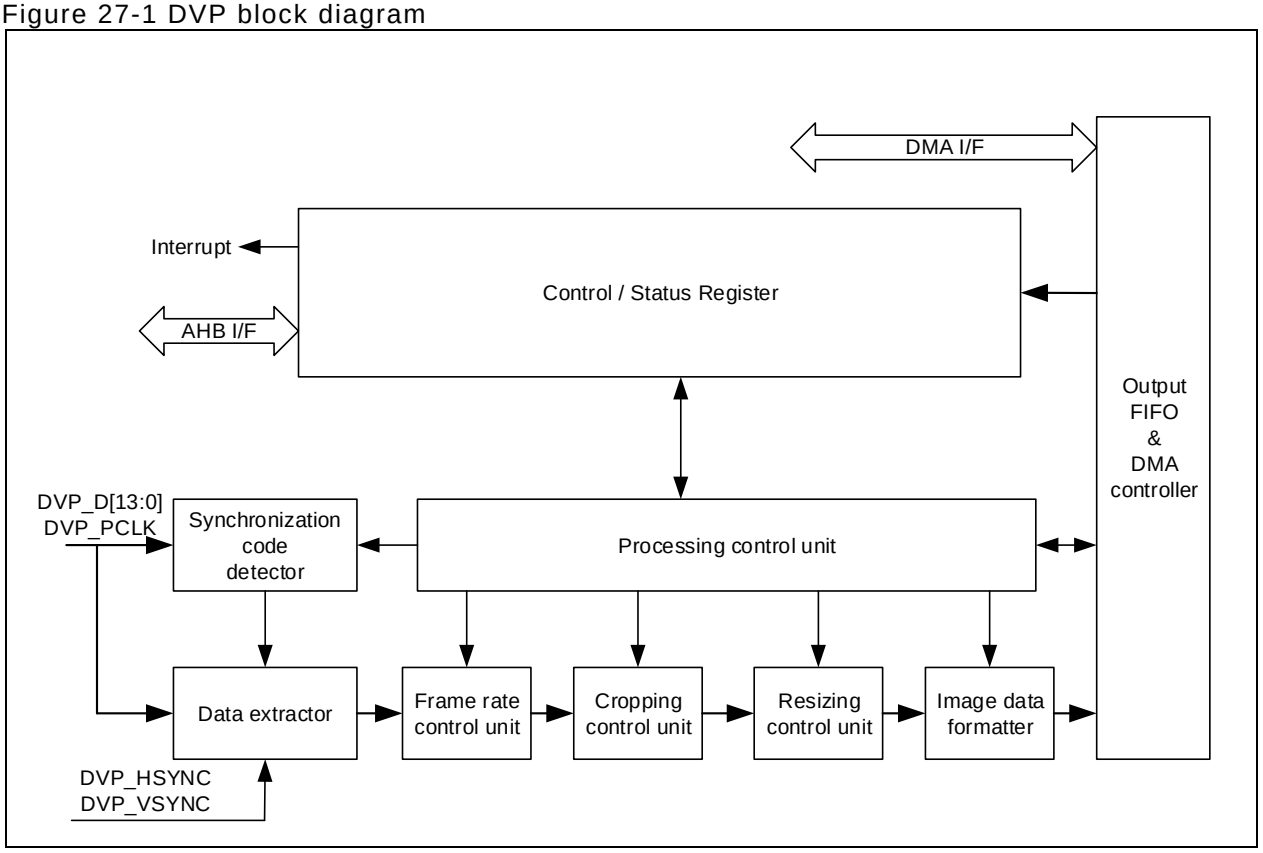
	rollover, 1hz (digital rollover is not recommended) 0010: For binary rollover, 4hz, duty cycle 50%; For digital rollover, 2hz (digital rollover is not recommended) 0011: For binary rollover, 8hz, duty cycle 50%; For digital rollover, 4hz (digital rollover is not recommended) 1111: For binary rollover, 32.768khz, duty cycle 50%; For digital rollover, 16.384khz (digital rollover is not recommended) Digital rollover is not recommended when the PPS is non-zero value, because PPS output waveforms will be irregular (although its average frequency is always correct in any one-second window) in these cases.

27 Digital Video parallel interface (DVP)

27.1 Introduction

The digital video parallel interface (DVP) is able to capture parallel data output on the CMOS video camera. It is possible to perform frame/line synchronization in either hardware or embedded synchronization code. With the frame rate control feature, the number of frames captured per second can be controlled. The crop window feature allows the users to retain the desired data and drop others. Using the image scaling feature, it is possible to scale down the number of pixels or lines. The DMA controller can be used to transfer the captured data to memory unit without the need of CPU. The users can monitor the status of data reception through status and error interrupts.

Figure 27-1 DVP block diagram



27.2 Introduction

- 8-bit, 10-bit, 12-bit or 14-bit parallel interface
- Parallel interface data alignment
- Hardware or embedded code synchronization
- Single frame or continuous capture mode
- Frame rate control
- Crop window feature
- mage scaling resize
- Monochrome image binarization
- DMA access single or burst transfer
- Frame capture completed, vertical synchronization and horizontal synchronization status interrupts
- FIFO overrun and embedded synchronization error interrupts

27.3 Data capture and synchronization

The digital video parallel interface uses the DVP pixel clock (DVP_PCLK) to capture the pixel parallel

data (DVP_D) output from the CMOS video camera. The DVP_PCLK is provided by the CMOS video camera. Data can be captured on either the rising or the falling edge of the DVP_PCLK by setting the CKP bit in the DVP_CTR register. The captured data can be divided into two parts: valid pixel data and blanking period data. The user only concerns the former. The CMOS can deliver synchronization information in a single or multiple modes for the receiver to perform frame segmentation and fetch valid pixel data. The digital video parallel interface (DVP) supports hardware or embedded code synchronization, which can be done through the SM bit in the DVP_CTRL register.

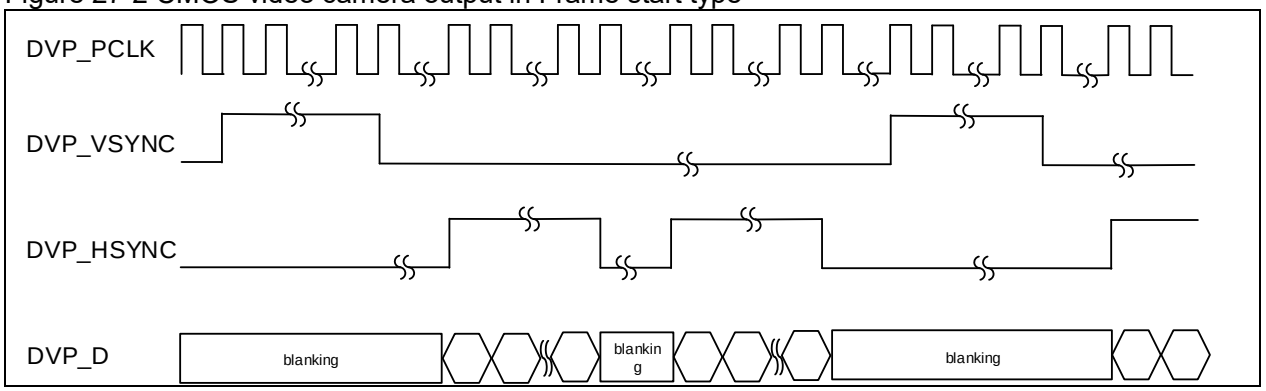
27.3.1 Hardware synchronization mode

In this mode, the CMOS video camera offers two synchronization signals: horizontal and vertical synchronization signals. The hardware synchronization mode is selected by setting SM=0 in the DVP_CTRL register. The horizontal synchronization signals are used to distinguish the valid pixel and blanking periods, and it is activated by the DVP_HSYNC pin to enable DVP to split lines and fetch valid pixel data. The polarity of the horizontal synchronization signals can be programmed through the HSP bit in the DVP_CTRL register in order to ensure that it matches the output of the digital video camera. The vertical synchronization signals are used for interframe separation, and it is activated by the DVP_VSYNC pin to enable DVP to split frames. The vertical synchronization signals are expressed in two types.

Frame start

In this mode, the vertical synchronization signals are used as a frame start signal. The Frame start signal indicates the end of the current frame and the start of the next frame. The polarity of the Frame start signal is programmable through the VSP bit in the DVP_CTRL register to ensure that it matches the output of the digital video camera. For example, in Figure 27-2, the VSP bit must be set (the DVP_VSYNC high level marks the Frame start).

Figure 27-2 CMOS video camera output in Frame start type



Frame valid

In this mode, the vertical synchronization signals are used to determine whether the captured data is in the vertical blanking period. Frames are split through the vertical blanking period. Similarly, the VSP bit in the DVP_CTR register can be used to program the polarity of the frame valid to ensure that it matches the output of the digital video camera. For example, in Figure 27-3, the VSP bit must be set to 0 (DVP_VSYNC low level indicates that the captured data is in the vertical blanking period).

Figure 27-3 CMOS video camera output in Frame valid type

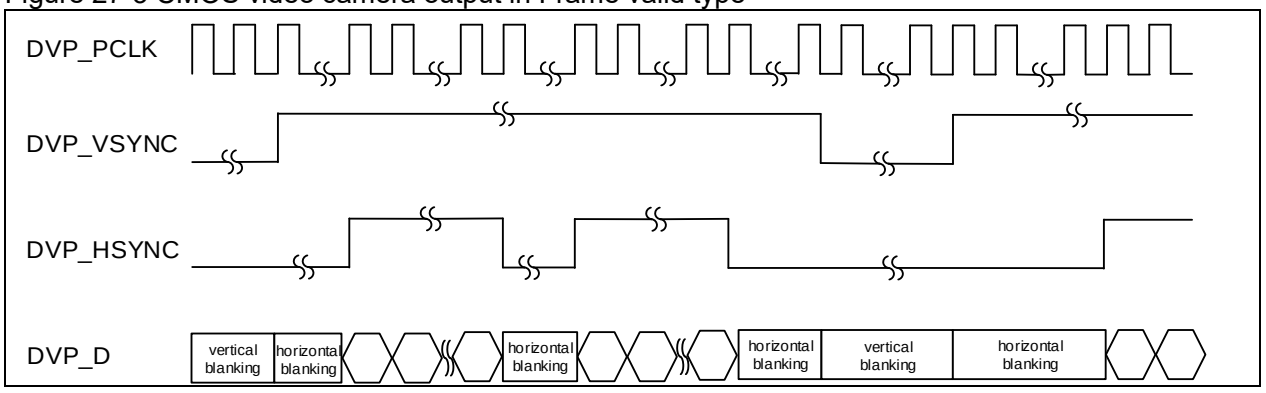


Table 27-1 lists the physical signals used by DVP in hardware synchronization mode. The use of the pixel parallel data (DVP_D) depends on the configurations of parallel data bits and alignment. Refer to Section 27.3.3 for details. Those unused signals may not be configured with the multiplexed function of the corresponding pins.

Table 27-1 DVP pin use in hardware synchronization mode

Signal name	Direction	Description
DVP_D[m:n]	Input	DVP pixel parallel data
DVP_PCLK	Input	DVP pixel clock
DVP_HSYNC	Input	DVP horizontal synchronization signal
DVP_VSYNC	Input	DVP vertical synchronization signal

27.3.2 Embedded data synchronization mode

To reduce the use of pins, some CMOS video cameras embed synchronization between valid pixel data and blanking data instead of the use of horizontal and vertical synchronization signals. The embedded synchronization mode is selected by setting SM=1 in the DVP_CTRL register. The embedded codes are composed of four data, in which the first three data have the fixed contents, with the first data all 1, and the subsequent two data all 0. The four data depends on synchronization information. The DVP_SCR register is programmed based on the synchronization type and content of the camera suppliers. The embedded code synchronization provides synchronization information in two types:

FS/FE/LS/LE type

In this mode, the CMOS camera uses four embedded synchronization codes to provide synchronization information. The Frame Start (FS) synchronization code is embedded between blanking area and valid pixels, signaling the start of a new frame. The data following the FS is the valid pixel data of line 0. For the remaining lines other than line 0, the LS (Line Start) synchronization code is used to indicate the end of blanking area and the start of a valid pixel. At the end of the last valid pixel data of each line, the LE code (Line End) is embedded indicating that the subsequent data are horizontal blanking. At the end of the line, the LE is replaced with the FE (Frame End), indicating the end of the current frame. Figure 27-4 shows the relationship between blanking area, valid pixel and synchronization code.

To enable FS/FE/LS/LE reception, the corresponding bits in the DVP_SCR register are configured as follows:

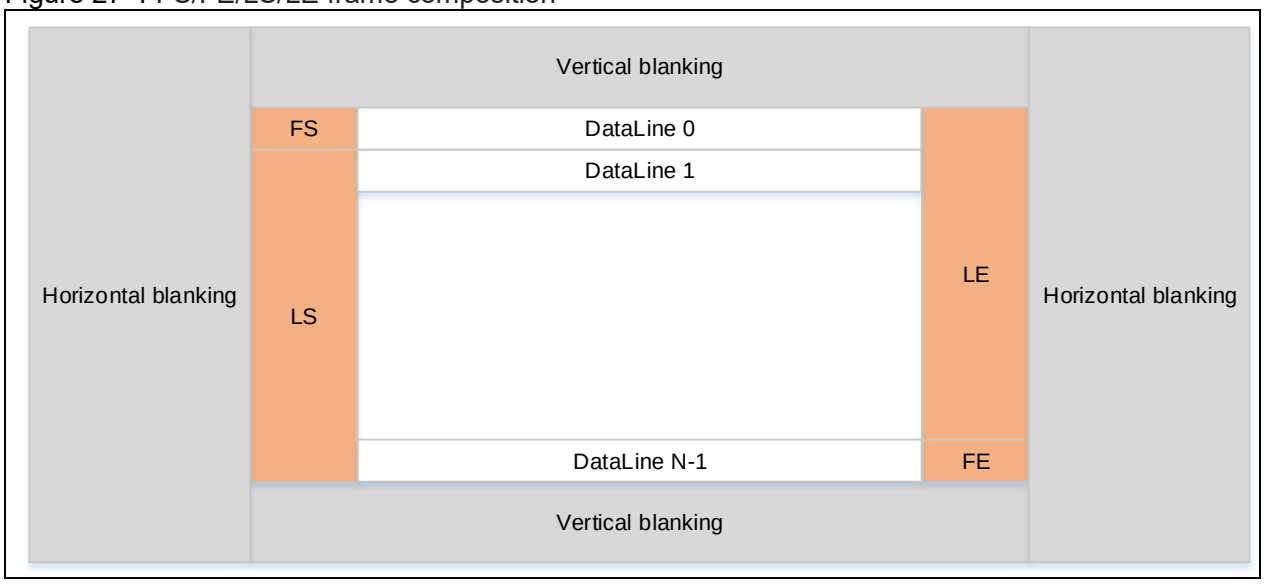
FMSC: The 4th captured data of FS is placed in the 8 most significant bits

LNSC: The 4th captured data of LS is placed in the 8 most significant bits

LNEC: The 4th captured data of LE is placed in the 8 most significant bits

FMEC: The 4th captured data of FE is placed in the 8 most significant bits

Figure 27-4 FS/FE/LS/LE frame composition



SAV/EAV type

In this mode, the CMOS camera also uses four embedded synchronization codes to deliver synchronization information. The line within a frame uses the active SAV code to indicate the end of blanking area and the start of a valid pixel. At the end of the last valid pixel data of each line, the active SAV code is embedded signaling that the data following this are vertical blanking. The vertical blanking area is used to separate frames. In a vertical blanking area, the embedded synchronization code is still based on the line level. The blanking SAV code indicates the start of the line, while the blanking EAV indicates the end of the line. Figure 27-5 shows the relationship between blanking area, valid pixel and synchronization code.

To enable SAV/EAV reception, the corresponding bits in the DVP_SCR register are configured as follows:

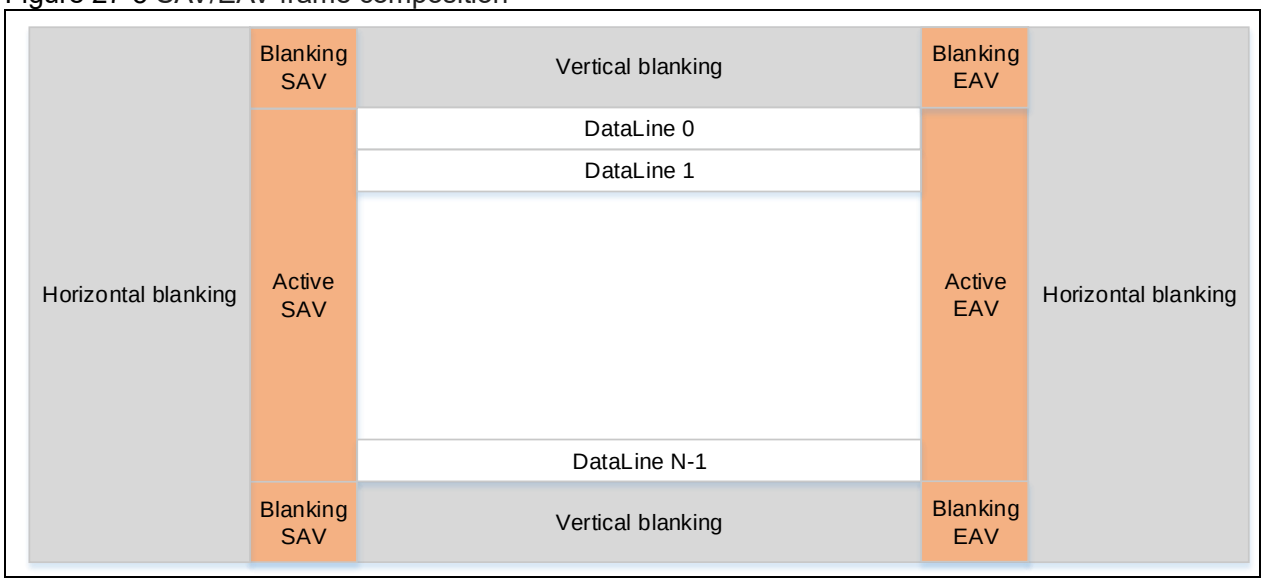
FMSC: Set as 0xff (embedded synchronization mode without frame start synchronization mode)

LNSC: The 4th captured data of Active SAV is placed in the 8 most significant bits

LNEC: The 4th captured data of Active EAV is placed in the 8 most significant bits

FMEC: Set as 0xff (embedded synchronization mode of arbitrary frame end synchronization mode)

Figure 27-5 SAV/EAV frame composition



Some of the CMOS cameras use some bits of the forth codes in one or several synchronization codes to pass on additional information. The DVP_SUR register can be used to mask the comparison behavior of decoder for the misappropriated bits of the synchronization code. The reset value 0x00 and 0xff configuration both can indicate the complete synchronization code comparison.

Table 27-2 gives the physical signals used in DVP in embedded synchronization mode. The use of the DVP_D depends on the configurations of parallel data bits and alignment. Refer to Section 27.3.3 for more information. The unused signals must not be configured with the multiplexed functions.

Table 27-2 DVP pin use in embedded synchronization mode

Signal name	Signal direction	Signal description
DVP_D[m:n]	Input	DVP pixel parallel data
DVP_PCLK	Input	DVP pixel clock

27.3.3 Data alignment

The parallel data bits and data alignment of the CMOS camera vary from supplier to supplier, and from model to model. To ensure the greatest degree of flexibility, the IDUS and IDUN bits in the DVP_ACTRL register can be used for the management of data alignment. Table 27-3 lists the CMOS camera parallel pin count, parallel data bits and alignment, and corresponding DVP register configuration and the use of DVP_D pin.

Table 27-3 DVP register configuration and DVP_D pin use

CMOS video camera parallel pin count		8	10			12					14
CMOS video camera parallel data bits		8	8		10	8		10		12	14
CMOS video camera data alignment		N/A	LSB	MSB	N/A	LSB	MSB	LSB	MSB	N/A	N/A
DVP_D	[13]										Bit 13
	[12]										Bit 12
	[11]						Bit 7		Bit 9	Bit 11	Bit 11
	[10]						Bit 6		Bit 8	Bit 10	Bit 10
	[9]			Bit 7	Bit 9		Bit 5	Bit 9	Bit 7	Bit 9	Bit 9
	[8]			Bit 6	Bit 8		Bit 4	Bit 8	Bit 6	Bit 8	Bit 8
	[7]	Bit 7	Bit 7	Bit 5	Bit 7	Bit 7	Bit 3	Bit 7	Bit 5	Bit 7	Bit 7
	[6]	Bit 6	Bit 6	Bit 4	Bit 6	Bit 6	Bit 2	Bit 6	Bit 4	Bit 6	Bit 6
	[5]	Bit 5	Bit 5	Bit 3	Bit 5	Bit 5	Bit 1	Bit 5	Bit 3	Bit 5	Bit 5
	[4]	Bit 4	Bit 4	Bit 2	Bit 4	Bit 4	Bit 0	Bit 4	Bit 2	Bit 4	Bit 4
	[3]	Bit 3	Bit 3	Bit 1	Bit 3	Bit 3		Bit 3	Bit 1	Bit 3	Bit 3
	[2]	Bit 2	Bit 2	Bit 0	Bit 2	Bit 2		Bit 2	Bit 0	Bit 2	Bit 2
	[1]	Bit 1	Bit 1		Bit 1	Bit 1		Bit 1		Bit 1	Bit 1
	[0]	Bit 0	Bit 0		Bit 0	Bit 0		Bit 0		Bit 0	Bit 0
IDUN		0	1	1	0	2	2	1	1	0	0
IDUS		0	0	1	0	0	1	0	1	0	0
PDL		0	0	0	1	0	0	1	1	2	3

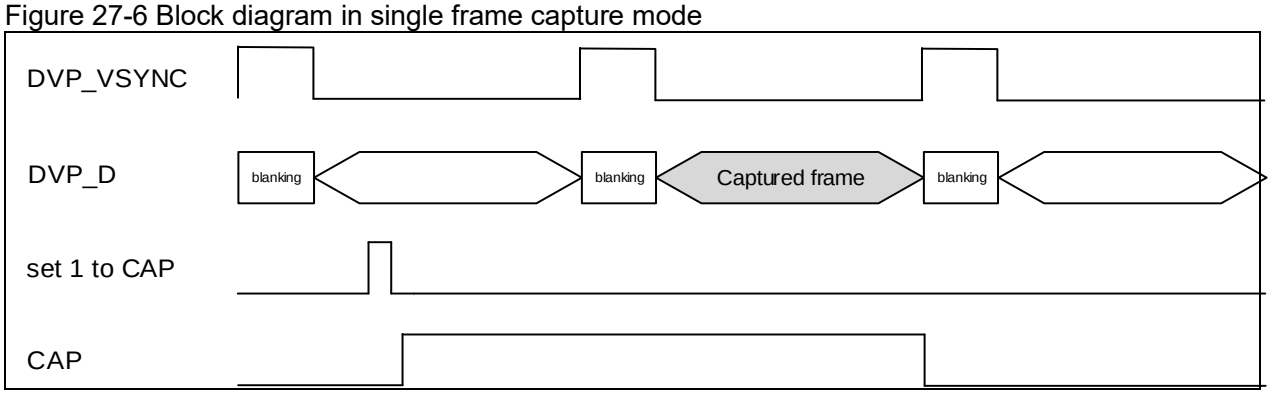
27.3.4 Single frame and continuous capture modes

The digital video parallel interface (DVP) supports two types of capture: single frame and continuous capture.

Single frame mode

A single frame mode is selected by setting CFM=1 in the DVP_CTRL register. In this mode, after the CAP bit is set in the DVP_CTR register, the DVP interface starts sampling data on the next frame start based on the received synchronization information. At the end of the frame, the DVP interface automatically resets the CAP bit and stops capturing data.

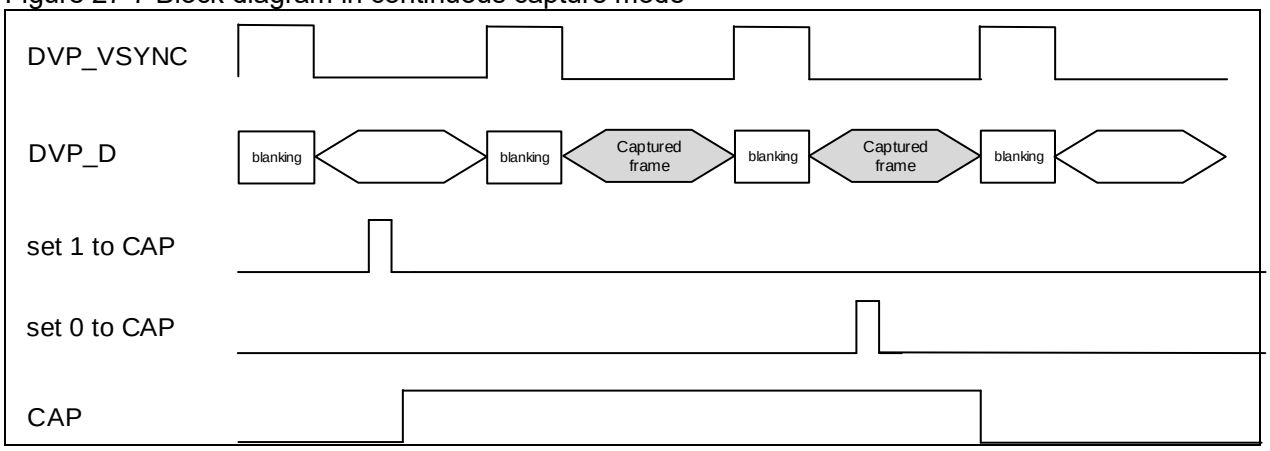
Figure 27-6 Block diagram in single frame capture mode



Continuous capture mode

The continuous capture mode is enabled by setting CFM=0 in the DVP_CTRL register. In this mode, after the CAP bit is set in the DVP_CTRL register, the DVP interface, based on the received synchronization information, starts sampling data on the start of the next frame start. This process continues. When the CAP bit is cleared, the DVP interface keeps the CAP bit in Set state and continues sampling data until the end of the current frame. At this point, the DVP interface resets the CAP bit and stops data sampling.

Figure 27-7 Block diagram in continuous capture mode



27.4 DMA access interface and data output packing

27.4.1 DMA access interface

The captured data can be transferred to memory unit using DMA interface without needing to occupy CPU resources. After the CAP is set in the DVP_CTRL register, the DMA interface is automatically activated. The DMA accesses the DVP_DT register through the AHB bus to fetch the DVP-captured data. A 32-bit data is transferred for each access. The DVP DMA interface supports single or burst transfers, depending on the DMABT bit in the DVP_ACTRL register.

Burst transfer mode

Burst transfer mode is enabled when setting DMABT=1. A DMA access is triggered each time the DVP receives four 32-bit data. After receiving a DMA request, the DMA makes four consecutive accesses to the DVP_DT register. The burst mode is able to significantly reduce the number of arbitration so as to improve bus performance. Only EDMA is supported in burst transfer mode, and the peripheral burst transfer of the EMDA must be set as INCR4 (PBURST=1).

Single transfer mode

Single transfer mode is enabled when setting DMABT=0. In this mode, a DMA access request is triggered each time the DVP receives a 32-bit data. After receiving a DMA request, the DMA interface accesses the DVP_DT register once. When the DMA interface uses DMA, only single transfer mode is supported. To use EMDA in this mode, the peripheral burst transfer must be set as single transfer mode (PBURST=0).

Note: To ensure correct data acquisition using DMA access interface, the AHB bus clock (AHB_CLK) must be higher than 2.5 DVP_PCLK clocks.

27.4.2 Data output packing

The captured data are packed into a 32-bit data register (DVP_DT) and then transferred through a DMA. The packing mode depends on the PDL bit in the DVP_CTRL register. Refer to Figure 27-8 for more information.

8-bit parallel data (PDL=0)

When the PDL is programmed to 0, the DVP captures an 8-bit data every DVP_PCLK clock, and packs four captured data into a word data. The first captured data is placed in the 8 least significant bits, and the last captured data is placed in the 8 most significant bits, and so on.

10-bit parallel data (PDL=1)

When the PDL is programmed to 1, the DVP captures a 10-bit data is captured every DVP_PCLK clock, and packs two captured data into a word data. The word data is made up of two half words. The first captured data is placed in the 10 least significant bits of the least significant half word, and the last captured data is placed in the 10 least significant bits of the most significant half word, and the remaining are cleared to zero.

12-bit parallel data (PDL=2)

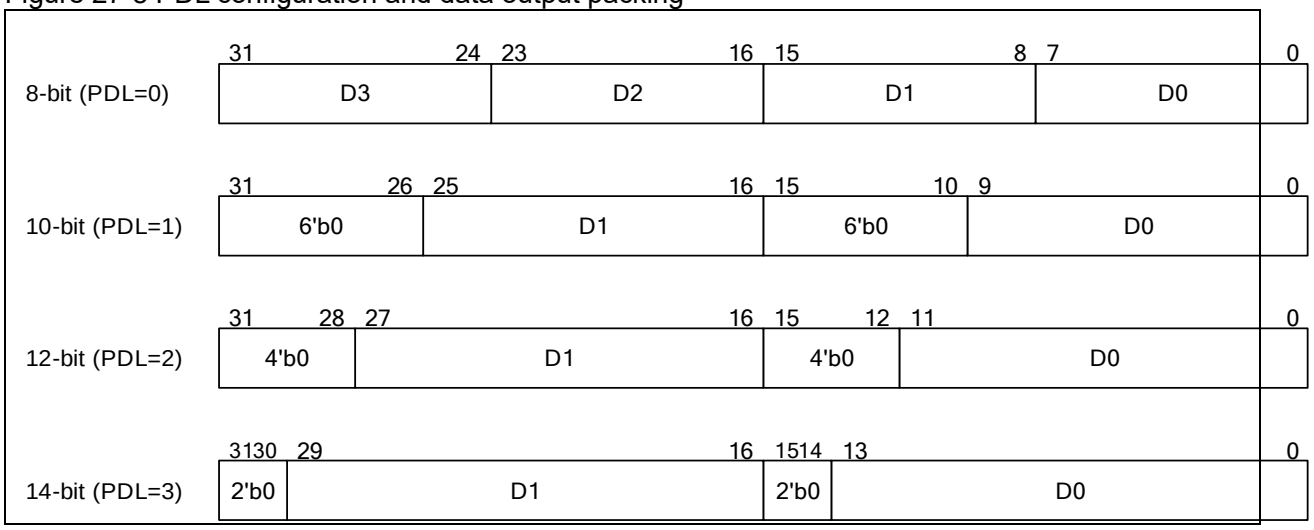
When the PDL bit is programmed to 2, the DVP captures a 12-bit data every DVP_PCLK clock, and

packs two captured data into a word data. The word data is made up of two half words. The first captured data is placed in the 12 least significant bits of the least significant half word, and the last captured data is placed in the 12 least significant bits of the most significant half word, and the remaining are cleared to zero.

14-bit parallel data (PDL=3)

When the PDL bit is programmed to 3, the DVP captures a 14-bit data every DVP_PCLK clock, and packs two captured data into a word data. The word data is made up of two half words. The first captured data is placed in the 14 least significant bits of the least significant half word, and the last captured data is placed in the 14 least significant bits of the most significant half word, and the remaining are cleared to zero.

Figure 27-8 PDL configuration and data output packing



27.5 Interrupts and interrupt control

There are four registers available to control the interrupts of the DVP interface. The read-only DVP_ESTS register is used to describe the synchronization status of error events occurred over the course of sampling. The DVP_IENA is used to control interrupt signals, and it can be programmed to enable the synchronization status or error interrupts of the corresponding bits, and send them to CPU. The interrupts, after enabled, are stored in the read-only DVP_ISTS register, which is used to check the event sources. The status in the DVP_ESTS and DVP_ISTS can be cleared by setting the corresponding bit in the DVP_ICLR register. The DVP_ICLR is a write-only register and does not need to be cleared.

DVP supports three synchronization status interrupts:

Capture frame done

When the CFDES and CFDIS bits are set (CAP enabled), it indicates the completion of the current frame capture. Based on the synchronization logic, the frame capture completed interrupt occurs when the frame end signal is detected. If the Crop feature is enabled, the frame capture completed interrupt occurs before the end of the crop window. If the Crop feature is not enabled, both CFDES and CFDIS bits have no effect.

Vertical synchronization

The VSES and VSIS bits indicate that vertical synchronization has been obtained. Vertical synchronization can be defined as the start or end of a frame, depending on the VSEID bit in the DVP_ACTRL register.

Horizontal synchronization

The HSES and HSIS bits indicate that horizontal synchronization has been obtained. The horizontal synchronization can be defined as the start or end of a line, depending on the HSEID bit in the DVP_ACTRL register.

The DVP supports two types of error interrupts:

Output data FIFO overrun

The OVRES and OVRIS bits (CAP enabled) indicate the error status of output data FIFO overrun. If

output data FIFO becomes full for the reason that the DMA is unable to capture data in time and transfer them to memory unit, in this case, the captured data will be discarded, and an output data overrun error interrupt is generated. If the CAP is not enabled, both OVRES and OVRIS bits have no effect.

Embedded synchronization error

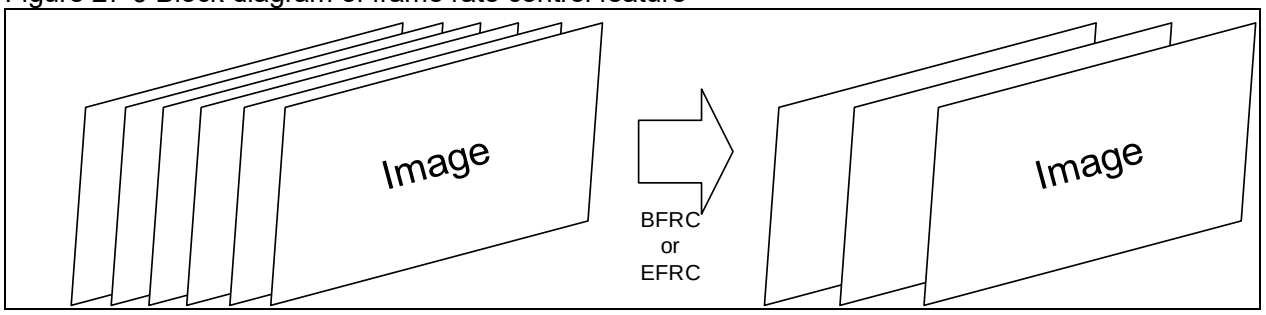
When the ESEES and ESEIS bits (CAP enabled, and SM=1) are set, it indicates the error status of embedded synchronization error. If it outputs an unexpected synchronization code, the decoder discards the current synchronization, and re-decode the subsequent data. In this case, the DVP stops data capturing and generates an embedded synchronization error. If the CAP is not enabled, both ESEES and ESEIS bits have no effect.

27.6 Functional overview

27.6.1 Frame rate control

The DVP interface can adjust the number of frame captured every second with the frame rate control feature. The frame rate control feature applies to continuous capture mode (CFM=0). It consists of a basic frame rate control and progressive frame rate control.

Figure 27-9 Block diagram of frame rate control feature



Basic frame rate control

Capture a frame per two frames or per two four frames, which is programmable by the BFRC bit in the DVP_CTR register.

Enhanced frame rate control

To acquire a fine frame rate, the advanced frame rate control is a choice. To enable this feature, set BFRC=0, and set the EFRCE bit in the DVP_ACTRL register, the DVP can adjust the number of frames through the EFRCSF and EFRCTF bits in the DVP_FRF register, based on the following formula:

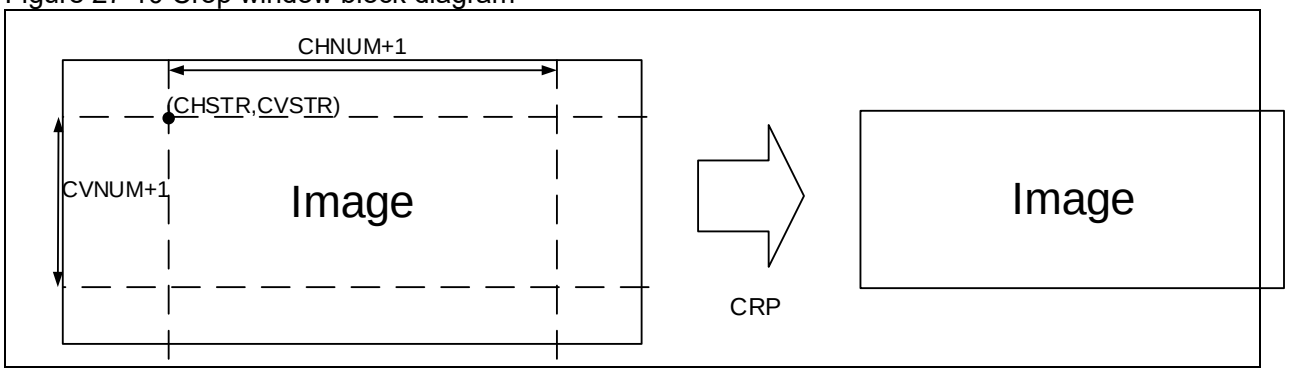
$$Target\ \ frame\ \ rate = \frac{EFRCTF}{EFRCSF} \times Original\ \ frame\ \ rate$$

Note: When the advanced frame rate control feature is used, the EFRCSF and EFRCTF bits must not be set to zero, and the EFRCTF must not be greater than the EFRCSF bit.

27.6.2 Crop window

The crop window feature can be used to retain the desired data area and discard the remaining area. When the CRP bit is set in the DVP_CTRL register, the DVP follows the configurations of the DVP_CWST and DVP_CWSZ registers to perform crop feature.

Figure 27-10 Crop window block diagram

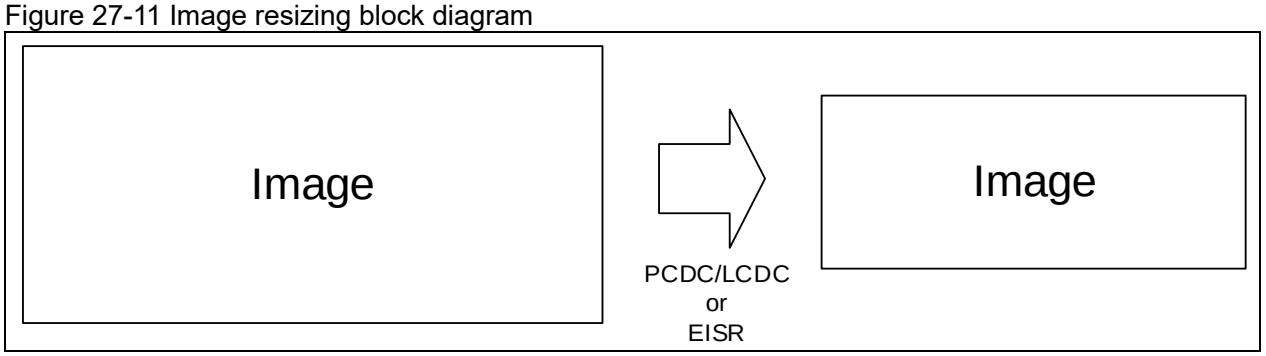


Note: As the DVP packs the captured data into a 32-bit word for DMA access, the CHNUM bit has the following limits while using crop feature: for 8-bit data (PDL=0), CHNUM+1 must be a multiple of 4; for 10-bit, 12-bit or 14-bit data (PDL≠0), CHNUM+1 must be a multiple of 2.

27.6.3 Image resizing

The image resizing feature can be used to scale down the pixels or lines of an image. It consists of a basic image capture/drop control and advanced image resizing feature.

Figure 27-11 Image resizing block diagram



Basic image capture/drop control feature

The basic image capture/drop control feature is enabled through the PCDC or LCDC bit in the DVP_CTRL register. After the LCDC bit is set to 1, drop one out of every two captured image lines to halve the size of the vertical axis of an image. The LCDS bit can be used to select whether to capture the previous one but drop the next one, or vice versa. The PCDC bit can be used to adjust the horizontal axis size of an image. The PCDS and PCDES bits determine which data to capture, as shown in Figure 27-12 and Figure 27-13.

Figure 27-12 LCDC/LCDS and frame structure

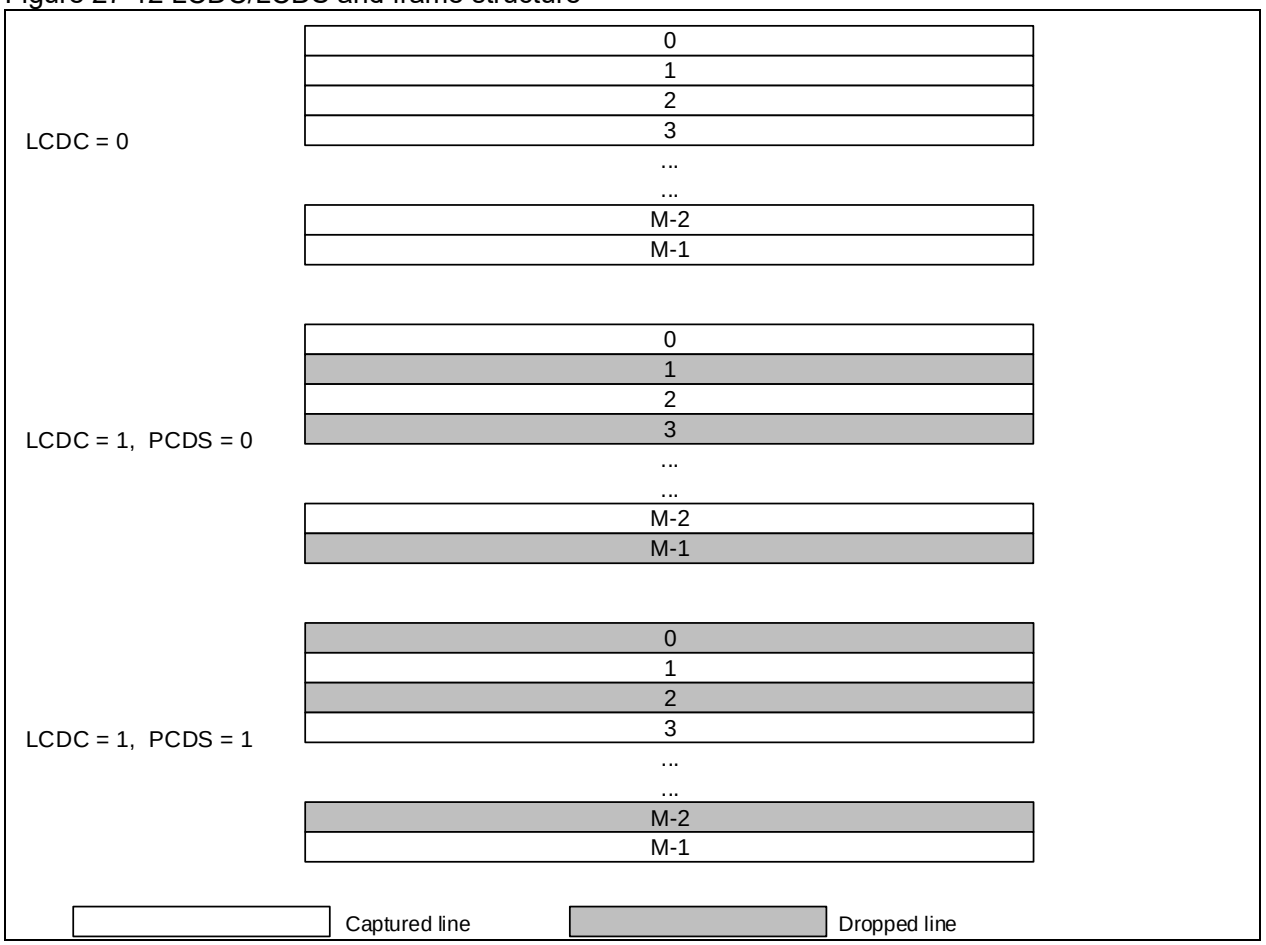
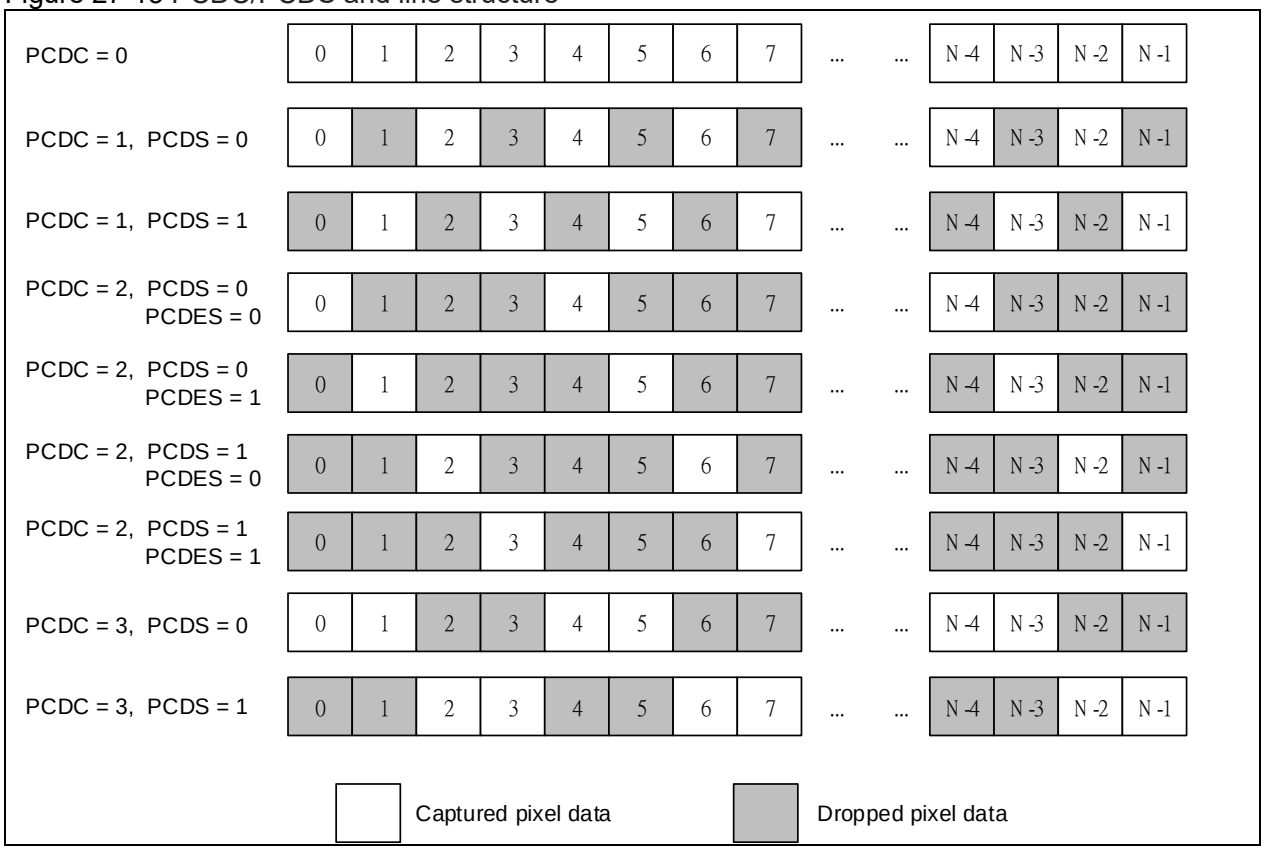


Figure 27-13 PCDC/PCDS and line structure



Note: The output format of the CMOS video camera must be determined while configuring PCDC and PCDS bits. Some configurations may cause the change in data formats. Refer to Section 27.7.3 for more information.

Enhanced image reduction feature

The advanced image reduction feature can be used to obtain a fine image scaling. To enable this feature, EFDM=1 must be set, and the EFDF must be programmed based on the output formats of the CMOS video camera. Refer to Section 27.7.3 for more information. When the enhanced image reduction feature is used, PCDC=0 and LCDC=0 must be configured, and the EISRE bit must be set in the DVP_ACTRL register. The DVP adjusts the size of the horizontal axis of an image through the HSRSF and HSRTF bits in the DVP_HSCF register, and modifies the size of the vertical axis of an image through the VSRSF and VSRTF bits in the DVP_VSCF register, based on the following formula:

$$Target\ \ horizontal\ \ axis\ \ size = \frac{HSRTF}{HSRSF} \times Original\ \ horizontal\ \ axis\ \ size$$

$$Target\ \ vertical\ \ axis\ \ size = \frac{VSRTF}{VSRSF} \times Original\ \ vertical\ \ axis\ \ size$$

Note: When using the enhanced image reduction feature, the HSRSF, HSRTF, VSRSF and VSRTF bits must not be set to 0, the HSRTF bit must not be greater than that of the HSRSF bit, and the VSRTF bit must not be higher than that of the VSRSF bit. Additionally, the calculated target horizontal size and target vertical size must be integers, and the target horizontal size must be a multiple of 4, otherwise, unpredictable results may be generated.

27.6.4 Grayscale image binarization conversion

The grayscale image binarization conversion unit is used to convert the luminance into one-bit format. To enable this feature, set EFDM=1, and program the EFDF bit based on the output format of CMOS video camera. Refer to Section 27.7.3 for more information. The MIBE bit is set in the DVP_ACTRL register to enable grayscale image binarization conversion feature.

The MIBTHD bit in the DVP_BTH register is used to program the grayscale image binarization threshold. The value above the threshold is deemed to 1, while the value below the threshold is deemed to 0. This feature becomes invalid if the luminance format of an image data cannot be fetched directly.

27.7 Data formats

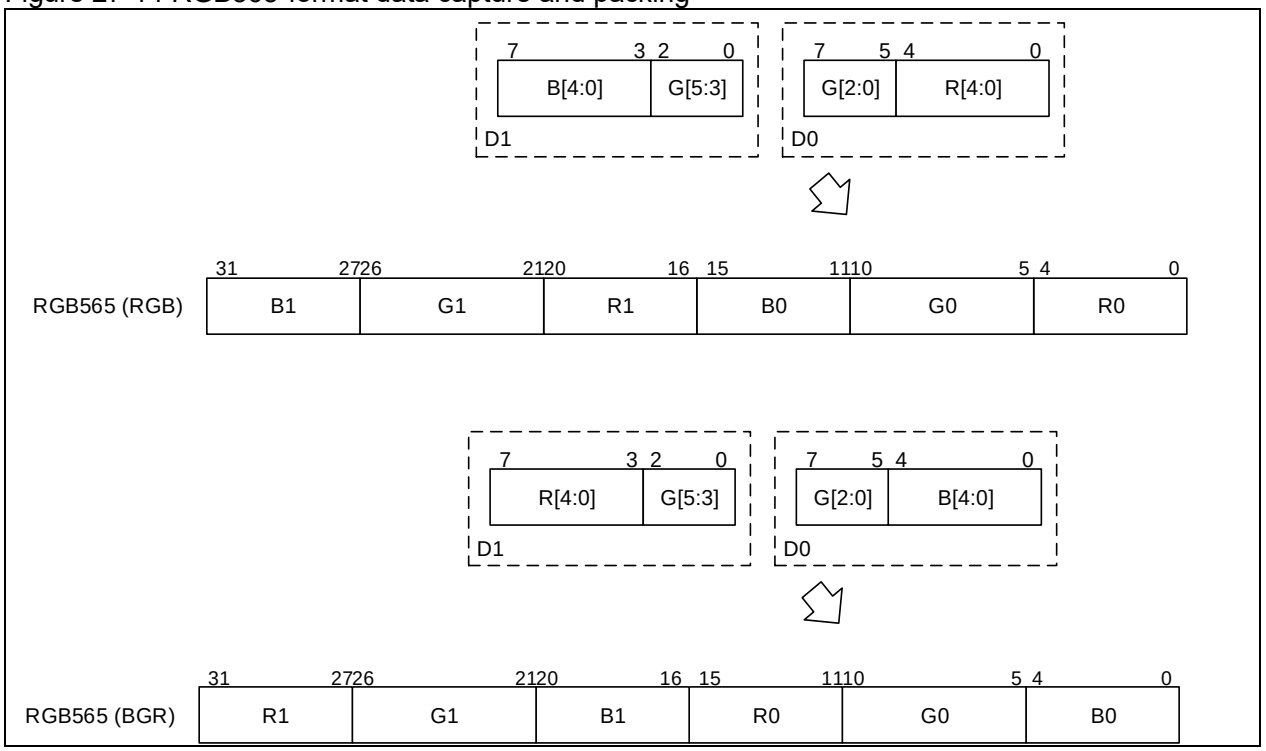
27.7.1 Common CMOS video camera output formats

The CMOS video camera offers various image output formats. This section only gives a brief account of common data formats.

RGB565 format

In this format, one half-word data pixel (16 bit) is output every two pixel clocks. Each half-word data contains three types of pixel components: R (red), Green (G) and Blue (B), which are encoded in 5-bit, 6-bit and 5-bit format respectively. The first pixel clock outputs the lower part of R and G components, the second pixel clock outputs the upper part of G component, and B component. Or the first pixel clock outputs the lower part of B and G components, and the second pixel clock outputs the upper part of G component, and R component. Figure 27-14 gives an example of DVP data capture and packing in RGB565 format.

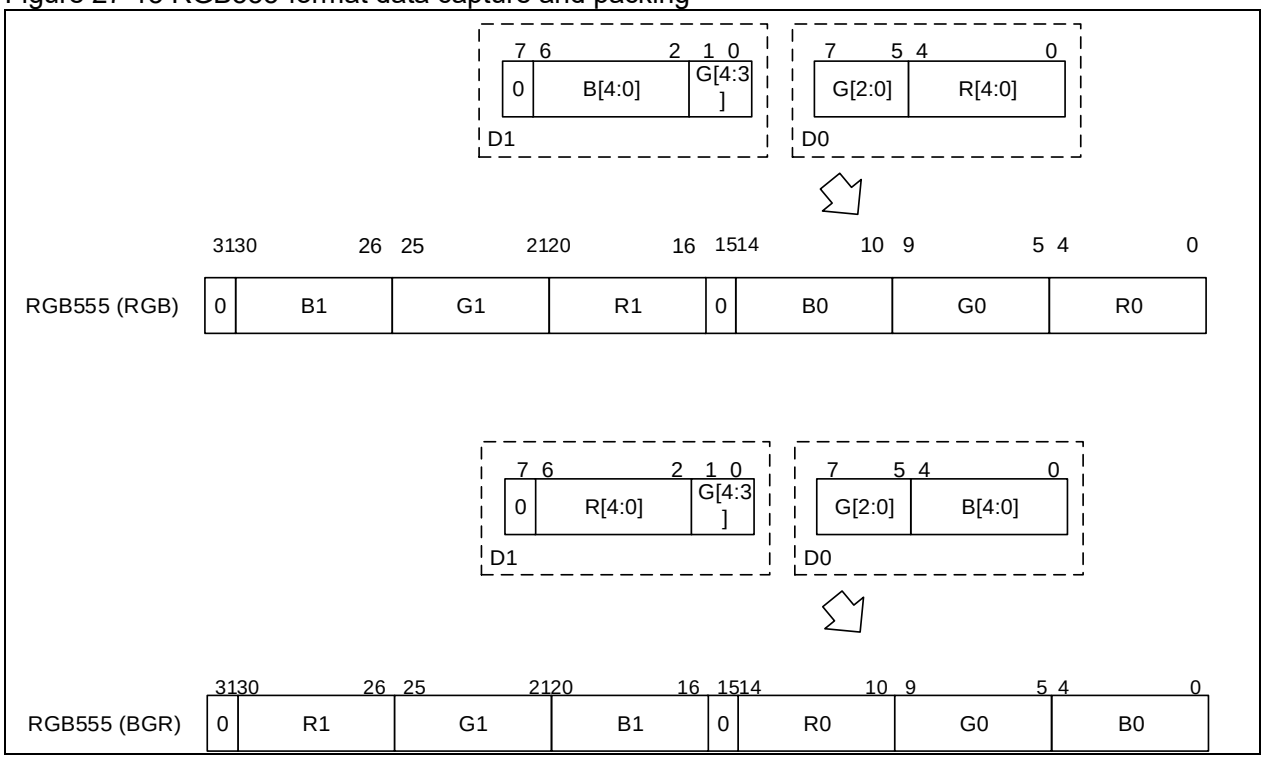
Figure 27-14 RGB565-format data capture and packing



RGB555 format

In this format, one half-word data pixel (16 bit) is output every two pixel clocks. Each half-word data contains three types of pixel components: R (red), Green (G) and Blue (B), which are encoded in 5-bit format. The first pixel clock outputs the lower part of R and G components, the second pixel clock outputs the upper part of G component, and B component. Or the first pixel clock outputs the lower part of B and G components, and the second pixel clock outputs the upper part of G component, and R component. As the valid data has a total of 15 bits, the upper part of the second pixel clock are not actually used. Typically, the CMOS video camera directly outputs a low level. Figure 27-15 gives an example of DVP data capture and packing in RGB555 format.

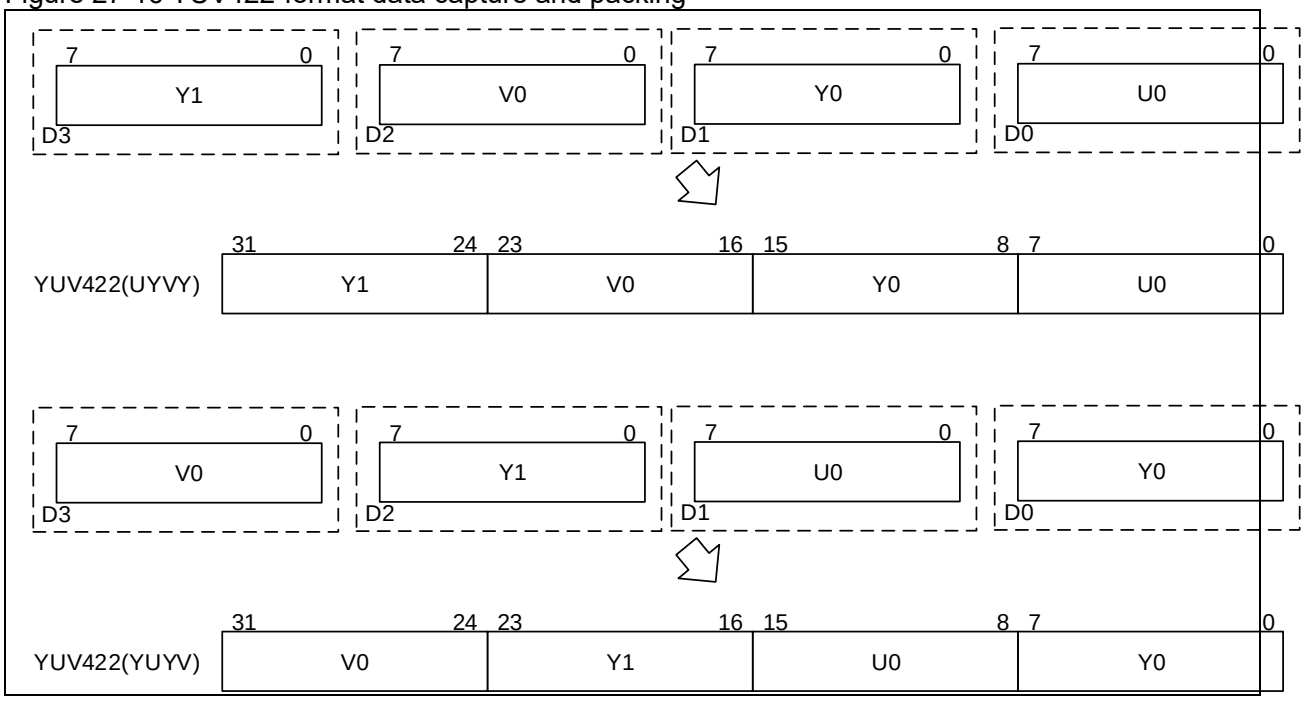
Figure 27-15 RGB555-format data capture and packing



YUV422 format

In this format, one half-word data pixel (16 bit) is output every two pixel clocks. Each half-word data contains a luminance component Y, and a chroma component U or V, which are encoded in 8-bit format. The chroma components U and V are interleaved among pixels, and two adjacent pixels use different chroma components. The camera outputs a group of pixel components every pixel clock, with Y component output on the first pixel clock, and U or V component output on the second pixel clock. Or the first pixel clock outputs U or V component, and the second pixel clock outputs Y component. Figure 27-16 gives an example of DVP data capture and packing in YUV422 format, in which the sequence of components U and V is swappable.

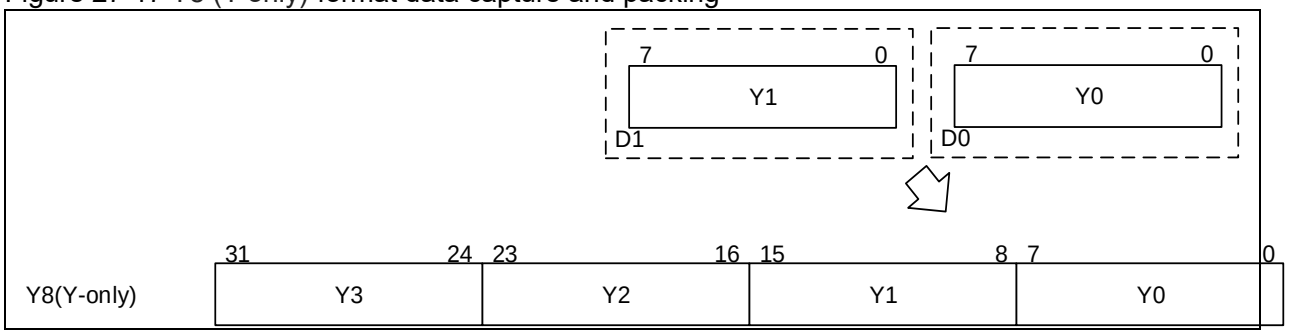
Figure 27-16 YUV422-format data capture and packing



Y8 (Y-only) format

In this format, one-byte pixel (8-bit) Y (luminance) is output every pixel clock, which is encoded in 8-bit format. There is no chroma output. Figure 27-17 gives an example of DVP data capture and packing in Y8 (Y-only) format.

Figure 27-17 Y8 (Y-only)-format data capture and packing



Raw data format

RAW data is the original pixel data obtained by CMOS video camera through Bayer color filter. One pixel data is output every pixel clock. It can be 8-bit, 10-bit, 12-bit or 14-bit. Refer to Section 27.4.2 for more information.

27.7.2 JPEG compressed format

To allow JPEG compressed data reception, it is necessary to set SM=0 and the JPEG big in the DVP_CTRL register. In this format, the vertical synchronization signal is still used to separate frames, in the form of Frame Start or Frame effective, depending on manufacturers. The horizontal synchronization signal is used as a data enable signal, without line separation information. Because the JPEG size is programmable, the remaining data are padded with 0 at the end of the current frame in order to fill a word data (32-bit).

Note: The crop feature and image scaling feature must not be used in the JPEG format.

27.7.3 Enhanced data management

Enhanced image scaling resize and monochrome image binarization are handled in different ways, depending on the captured data format. To enable both features, EFDM=1 must be asserted, and the EFDF must be programmed according to the CMOS video camera output format. Table 27-4 describes two features and DVP data formats supported.

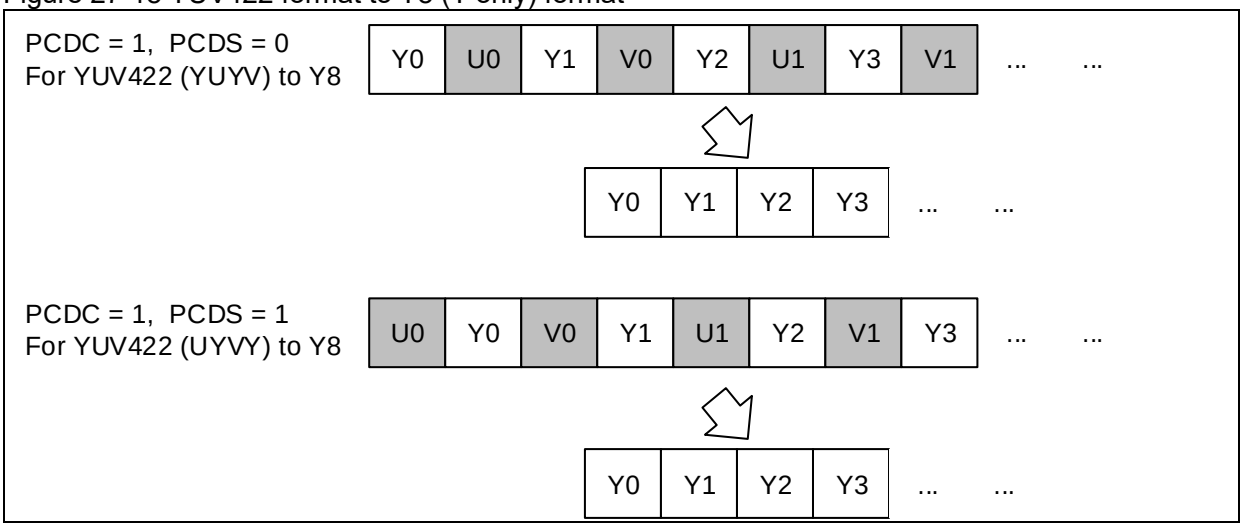
Table 27-4 Enhanced features and DVP data formats supported

EFDM	EFDF	Data format	Enhanced image scaling resize	Monochrome image binarization
0	don't care	don't care	Not supported	not supported
1	0	YUV422 (UYVY / VYUY)	Supported	Supported
1	1	YUV422 (YUYV / YVYU)	Supported	Supported
1	2	RGB565 RGB555	Supported	Not supported
1	3	Y8 (Y-only)	Supported	Supported

27.7.4 Data format conversion

In YUV422 format, the DVP interface can use a basic pixel capture/drop feature to pick up luminance component Y, and pack it in Y8 (Y-only) format, as shown in Figure 27-18.

Figure 27-18 YUV422 format to Y8 (Y-only) format



27.8 Registers

Table 27-5 shows the DVP register map and its reset values.

The peripheral registers can be accessed by words (32-bit).

Table 27-5 DVP register map and reset values

Register	Offset	Reset value
DVP_CTRL	0x000	0x0000 0000
DVP_STS	0x004	0x0000 0000
DVP_ESTS	0x008	0x0000 0000
DVP_IENA	0x00C	0x0000 0000
DVP_ISTS	0x010	0x0000 0000
DVP_ICLR	0x014	0x0000 0000
DVP_SCR	0x018	0x0000 0000
DVP_SUR	0x01C	0x0000 0000
DVP_CWST	0x020	0x0000 0000
DVP_CWSZ	0x024	0x0000 0000
DVP_DT	0x028	0x0000 0000
DVP_ACTRL	0x040	0x0000 0000
DVP_HSCF	0x048	0x0000 0000
DVP_VSCF	0x04C	0x0000 0000
DVP_FRF	0x050	0x0000 0000
DVP_BTH	0x054	0x0000 0000

27.8.1 DVP control register (DVP_CTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 21	Reserved	0x000	resd	Kept at its default value.
Bit 20	LCDS	0x0	rw	Basic line capture/drop selection 0: Capture the first line and drop the next 1: Drop the first line and capture the next This register is valid when the LCDC=1 is asserted.
Bit 19	LCDC	0x0	rw	Basic line capture/drop control 0: All frames are captured or use enhanced image scaling resize feature 1: Enable capture/drop control to capture one out of two

				lines
Bit 18	PCDS	0x0	rw	Basic pixel capture/drop selection 0: Capture the first group of data (one or two pixel data) and drop the next group 1: Drop the first group of data (one or two pixel data) and capture the next group This register is valid when the PCDC=1/2/3 is asserted.
Bit 17: 16	PCDC	0x0	rw	Basic pixel capture/drop control 0: All frames are captured or use enhanced image scaling resize feature 1: Enable capture/drop control to capture one in two pixel data 2: Enable capture/drop control to capture one in four pixel data 3: Enable capture/drop control to capture two consecutive data in four pixel data
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	ENA	0x0	rw	DVP Enable 0: DVP disabled 1: DVP enabled The DVP register configuration must be completed before enabling DVP.
Bit 13: 12	Reserved	0x0	resd	Kept at its default value.
Bit 11: 10	PDL	0x0	rw	Pixel data length 0: Interface captures 8-bit data 1: Interface captures 10-bit data 2: Interface captures 12-bit data 3: Interface captures 14-bit data
Bit 9: 8	BFRC	0x0	rw	Basic frame rate control 0: All frames are captured or use enhanced frame rate control feature 1: Enable frame rate control, every alternate frame captured 2: Enable frame rate control, one frame in 4 frames captured 3: Reserved This feature is valid only when CFM=0 is asserted.
Bit 7	VSP	0x0	rw	DVP_VSYNC polarity This bit defines the vertical synchronization signals. Frame start: 0: DVP_VSYNC low level indicates a Frame start signal 1: DVP_VSYNC high level indicates a Frame start signal Frame effective: 0: DVP_VSYNC low level indicates that the captured data is in vertical blanking 1: DVP_VSYNC high level indicates that the captured data is in vertical blanking This feature is valid only when SM=0 is asserted.
Bit 6	HSP	0x0	rw	DVP_HSYNC polarity 0: DVP_HSYNC high level indicates that the captured data is a valid pixel data, and Line start signal on the rising edge 1: DVP_HSYNC low level indicates that the captured data is a valid pixel data, and Line start signal on the falling edge This feature is valid only when SM=0 is asserted.
Bit 5	CKP	0x0	rw	DVP_PCLK polarity 0: DVP_PCLK rising edge active 1: DVP_PCLK falling edge active
Bit 4	SM	0x0	rw	Synchronization mode 0: Hardware synchronization mode 1: Embedded synchronization mode
Bit 3	JPEG	0x0	rw	JPEG format 0: Uncompressed video format 1: Compressed video format

				This feature is valid only when SM=0 is asserted.
Bit 2	CRP	0x0	rw	Cropping window function enable 0: Cropping window function disabled 1: Cropping window function enabled
Bit 1	CFM	0x0	rw	Capture function mode 0: Continuous capture mode 1: Single frame capture mode
Bit 0	CAP	0x0	rw	Capture function enable 0: Capture function disabled 1: Capture function enabled The DMA controller and DVP register configurations must be programmed before enabling this bit. When CFM=1, after this bit is set, this register is automatically reset after the completion of a single frame capture. When CFM=0, after this bit is set, this register remains in set statue. After this bit is cleared by software, this register is automatically reset after the completion of the current frame capture.

27.8.2 DVP status register (DVP_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 3	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 2	OFNE	0x0	ro	Output data FIFO status 0: FIFO empty 1: FIFO has valid data
Bit 1	VSYN	0x0	ro	Vertical synchronization status 0: Vertical synchronization is not in blanking state 1: Vertical synchronization is in blanking state This bit is valid when the CAP is set.
Bit 0	HSYN	0x0	ro	Horizontal synchronization status 0: Horizontal synchronization is not in blanking state 1: Horizontal synchronization is in blanking state This bit is valid when the CAP is set.

27.8.3 DVP event status register (DVP_ESTS)

Bit	Register	Reset value	Type	Description
Bit 31: 5	Reserved	0x0000 000	resd	Kept at its default value.
Bit 4	HSES	0x0	ro	Horizontal synchronization event status 0: No horizontal synchronization status detected 1: Horizontal synchronization status detected It is cleared by writing 1 to the HSIC bit in the DVP_ICLR register.
Bit 3	VSES	0x0	ro	Vertical synchronization event status 0: No vertical synchronization status detected 1: Vertical synchronization status detected It is cleared by writing 1 to the VSIC bit in the DVP_ICLR register.
Bit 2	ESEES	0x0	ro	Embedded synchronization error event status 0: Embedded synchronization normal 1: Embedded synchronization error It is cleared by writing 1 to the ESEIC bit in the DVP_ICLR register. This feature is valid only when SM=1 is asserted.
Bit 1	OVRES	0x0	ro	Output data FIFO overrun event status 0: No data FIFO overrun event detected 1: Data FIFO overrun event detected It is cleared by writing 1 to the OVRIC bit in the DVP_ICLR register.
Bit 0	CFDES	0x0	ro	Capture frame done raw event status 0: A frame has not been captured 1: A frame has been captured

				It is cleared by writing 1 to the CFDIC bit in the DVP_ICLR register.

27.8.4 DVP interrupt enable register (DVP_IENA)

Bit	Register	Reset value	Type	Description
Bit 30: 5	Reserved	0x0000000	resd	Kept at its default value.
Bit 4	HSIE	0x0	rw	Horizontal synchronization interrupt enable 0: Horizontal synchronization interrupt disabled 1: Horizontal synchronization interrupt enabled
Bit 3	VSIE	0x0	rw	Vertical synchronization interrupt enable 0: Vertical synchronization interrupt disabled 1: Vertical synchronization interrupt enabled
Bit 2	ESEIE	0x0	rw	Embedded synchronization error interrupt enable 0: Embedded synchronization error interrupt disabled 1: Embedded synchronization error interrupt enabled
Bit 1	OVRIE	0x0	rw	Output data FIFO overrun interrupt enable 0: Output data FIFO overrun interrupt disabled 1: Output data FIFO overrun interrupt enabled
Bit 0	CFDIE	0x0	rw	Capture frame done interrupt enable 0: Capture frame done interrupt disabled 1: Capture frame done interrupt enabled

27.8.5 DVP interrupt status register (DVP_ISTS)

Bit	Register	Reset value	Type	Description
Bit 31: 5	Reserved	0x0000 000	resd	Kept at its default value.
Bit 4	HSIS	0x0	ro	Horizontal synchronization interrupt status 0:No horizontal synchronization interrupt generated 1: Horizontal synchronization interrupt generated It is cleared by writing 1 to the HSIC bit in the DVP_ICLR register.
Bit 3	VSIS	0x0	ro	Vertical synchronization interrupt status 0: No vertical synchronization interrupt generated 1: Vertical synchronization interrupt generated It is cleared by writing 1 to the VSIC bit in the DVP_ICLR register.
Bit 2	ESEIS	0x0	ro	Embedded synchronization error interrupt status 0: No embedded synchronization error interrupt generated 1: Embedded synchronization error interrupt generated It is cleared by writing 1 to the ESEIC bit in the DVP_ICLR register. This feature is valid only when SM=1 is asserted.
Bit 1	OVRIS	0x0	ro	Output data FIFO overrun interrupt status 0: No FIFO overrun interrupt generated 1: FIFO overrun interrupt generated It is cleared by writing 1 to the OVRIC bit in the DVP_ICLR register.
Bit 0	CFDIS	0x0	ro	Capture frame done interrupt status 0: A frame has not been captured 1: Capture frame done interrupt generated It is cleared by writing 1 to the CFDIC bit in the DVP_ICLR register.

27.8.6 DVP interrupt clear register (DVP_ICLR)

Bit	Register	Reset value	Type	Description
Bit 31: 5	Reserved	0x0000000	resd	Kept at its default value.
Bit 4	HSIC	0x0	wo	Horizontal synchronization interrupt clear Writing 1 to this bit clears the HSES bit in the DVP_ESTS register, and clears the HSIS bit in the DVP_ISTS register.
Bit 3	VSIC	0x0	wo	Vertical synchronization interrupt clear Writing 1 to this bit clears the VSES bit in the DVP_ESTS register, and clears the VSIS bit in the DVP_ISTS register.
Bit 2	ESEIC	0x0	wo	Embedded synchronization error interrupt clear

				Writing 1 to this bit clears the ESEES bit in the DVP_ESTS register, and clears the ESEIS bit in the DVP_ISTS register.
Bit 1	OVRIC	0x0	wo	Output data FIFO overrun interrupt clear Writing 1 to this bit clears the OVRES bit in the DVP_ESTS register, and clears the OVRIS bit in the DVP_ISTS register.
Bit 0	CFDIC	0x0	wo	Capture frame done interrupt clear Writing 1 to this bit clears the CFDES bit in the DVP_ESTS register, and clears the CFDIS bit in the DVP_ISTS register.

27.8.7 DVP embedded synchronization code register (DVP_SCR)

Bit	Register	Reset value	Type	Description
Bit 31: 24	FMEC	0x00	rw	Frame end synchronization 4th code Frame end delimiter code consists of four consecutive data: All 1, all 0, all 0, FE4 Frame end delimiter 4th code (FE4) is composed of FMEC. In embedded synchronization mode, if the FMEC is programmed to 0xFF, the decoder performs embedded synchronization for any frame end delimiter. All the undefined bytes other than FMSC, LNSC and LNEC are interpreted as frame end delimiter 4th code FE4
Bit 23: 16	LNEC	0x00	rw	Line end synchronization 4th code The code consists of 4 consecutive data: All 1, all 0, all 0 and LE4 Line end delimiter 4th data (LE4) is composed of LNEC.
Bit 15: 8	LNSC	0x00	rw	Line start synchronization 4th code The code consists of 4 consecutive data: All 1, all 0, all 0 and LS4 Line start delimiter 4th data (LS4) is composed of LNSC.
Bit 7: 0	FMSC	0x00	rw	Frame start synchronization 4th code In embedded synchronization mode, if the FMSC and FMSU both are programmed to 0xFF, the decoder performs embedded synchronization for non-frame-start synchronization code. The first occurrence of line start synchronization code after a Frame end code will be interpreted as a start of frame synchronization code.

27.8.8 DVP embedded synchronization unmask register (DVP_SUR)

Bit	Register	Reset value	Type	Description
Bit 31: 24	FMEU	0x00	rw	Frame end synchronization code unmask This field specifies the mask to be applied to the code of the frame end synchronization. PDL=0, set bit N =0 in the FMEU, the bit N is masked PDL=1, set bit N =0 in the FMEU, the bit N+2 is masked PDL=2, set bit N =0 in the FMEU, the bit N+4 is masked PDL=2, set bit N =0 in the FMEU, the bit N+6 is masked
Bit 23: 16	LNEU	0x00	rw	Line end synchronization code unmask This field specifies the mask to be applied to the code of the line end synchronization. PDL=0, set bit N =0 in the LNEU, the bit N is masked PDL=1, set bit N =0 in the LNEU, the bit N+2 is masked PDL=2, set bit N =0 in the LNEU, the bit N+4 is masked PDL=2, set bit N =0 in the LNEU, the bit N+6 is masked
Bit 15: 8	LNSU	0x00	rw	Line start synchronization code unmask This field specifies the mask to be applied to the code of the line end synchronization. PDL=0, set bit N =0 in the LNSU, the bit N is masked

				PDL=1, set bit N =0 in the LNSU, the bit N+2 is masked PDL=2, set bit N =0 in the LNSU, the bit N+4 is masked PDL=2, set bit N =0 in the LNSU, the bit N+6 is masked
Bit 7: 0	FMSU	0x00	rw	Frame start synchronization code unmask This field specifies the mask to be applied to the code of the line end synchronization. PDL=0, set bit N =0 in the FMSU, the bit N is masked PDL=1, set bit N =0 in the FMSU, the bit N+2 is masked PDL=2, set bit N =0 in the FMSU, the bit N+4 is masked PDL=2, set bit N =0 in the FMSU, the bit N+6 is masked

27.8.9 DVP crop window start register (DVP_CWST)

Bit	Register	Reset value	Type	Description
Bit 31: 29	Reserved	0x0	resd	Kept at its default value.
Bit 28: 16	CVSTR	0x00	rw	Crop window vertical start line This field specifies the start position of crop window in vertical axis. The first line data captured after the Frame start is line 0, the second line data captured is line 1, and so on.
Bit 15: 14	Reserved	0x0	resd	Kept at its default value.
Bit 13: 0	CHSTR	0x00	rw	Crop window horizontal start pixel This field specifies the start position of crop window in horizontal axis. The first pixel data captured after the Frame start is data 0, the second line data captured is data 1, and so on.

27.8.10 DVP crop window size register (DVP_CWSZ)

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x00	resd	Kept at its default value.
Bit 29: 16	CVNUM	0x00	rw	Crop window vertical line number minus one This field specifies the number of lines of a crop window in vertical axis. CVNM1=N indicates that N+1 data has been cropped from the vertical axis.
Bit 15: 14	Reserved	0x00	resd	Kept at its default value.
Bit 13: 0	CHNUM	0x00	rw	Crop window horizontal pixel number minus one This field specifies the number of pixel data of a crop window in horizontal axis. CHNM1=N indicates that N+1 pixel data has been cropped from the vertical axis.

27.8.11 DVP data register (DVP_DT)

Bit	Register	Reset value	Type	Description
Bit 31: 0	DT	0x00000000	ro	Data Port This register is used by the DMA controller to pick up data.

27.8.12 DVP advanced control register (DVP_ACTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 18	Reserved	0x00	resd	Kept at its default value.
Bit 17	VSEID	0x0	rw	Vertical synchronization event and interrupt definition 0: VSES and VEIS indicates frame end event and interrupt 1: VSES and VEIS indicates frame start event and interrupt
Bit 16	HSEID	0x0	rw	Horizontal synchronization event and interrupt definition 0: HSES and HEIS indicates line end event and interrupt 1: HSES and HEIS indicates line start event and interrupt
Bit 15: 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	DMABT	0x0	rw	DMA burst transaction This register works with EDMA's peripheral transfer configuration (PBURST) 0: DMA burst transaction disabled. The EDMA's peripheral transfer must be programmed as a single transfer (PBURST=0), or use DMA.

				1: DMA burst transaction enabled. The EDMA's peripheral transfer must be programmed as INCR4 (PBURST=1). This configuration is enabled only when the EDMA is used for data transfer. If the DMA is used, this configuration must be disabled.
Bit 11	Reserved	0x0	resd	Kept at its default value.
Bit 10	IDUS	0x0	rw	Input data un-used setting 0: Unused data bit in the MSB 1: Unused data bit in the LSB
Bit 9: 8	IDUN	0x0	rw	Input data un-used bit number 0: No unused bits 1: 2-bit unused data 2: 4-bit unused data 3: 6-bit unused data
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6	EFDM	0x0	rw	Enhanced function data format management 0: Enhanced function data format management disabled 1: Enhanced function data format management enabled Enhanced function data format management must be enabled before enabling the enhanced image scaling resize and monochrome image binarization.
Bit 5: 4	EFDF	0x0	rw	Enhanced function data format 0: YUV422 (UYVY / VYUY) data format 1: YUV422 (YUYV / YVYU) data format 2: RGB565 and RGB555 data format 3: Y8 (Y only) data format This configuration is valid when EFDM=1 is asserted.
Bit 3	PCDES	0x0	rw	Basic pixel capture/drop extended selection This register works with a basic pixel capture/drop selection (PCDS). PCDS=0: 0: Capture the first pixel data and drop others 1: Capture the second pixel data and drop others PCDS=1: 0: Capture the third pixel data and drop others 1: Capture the forth pixel data and drop others This configuration is valid only when the basic pixel capture/drop control is enabled and PCDC=2.
Bit 2	MIBE	0x0	rw	Monochrome image binarization enable 0: Monochrome image binarization disabled 1: Monochrome image binarization enabled This configuration is valid only when EFDM=1 and EFDF is set based on CMOS video camera output format.
Bit 1	EFRCE	0x0	rw	Enhanced frame rate control enable 0: Enhanced frame rate control disabled 1: Enhanced frame rate control enabled This configuration is valid only when CFM=0 and basic frame rate control is disabled.
Bit 0	EISRE	0x0	rw	Enhanced image scaling resize enable 0: Enhanced image scaling resize disabled 1: Enhanced image scaling resize enabled This configuration is valid only when PCDC=0 and LCDC=0. This configuration works normally only when EFDM=1 and EFDT is set.

27.8.13 DVP enhanced horizontal scaling factor register (DVP_HSCF)

Bit	Register	Reset value	Type	Description
Bit 30: 29	Reserved	0x0	resd	Kept at its default value.
Bit 28: 16	HSRTF	0x00	rw	Horizontal scaling resize target factor When EISRE=1, this register must not be 0, or greater than HSRSF value.
Bit 15: 13	Reserved	0x0	resd	Kept at its default value.
Bit 12: 0	HSRSF	0x00	rw	Horizontal scaling resize source factor When EISRE=1, this register must not be 0,

27.8.14 DVP enhanced vertical scaling factor register (DVP_VSCF)

Bit	Register	Reset value	Type	Description
Bit 30: 29	Reserved	0x0	resd	Kept at its default value.
Bit 28: 16	VSRTF	0x00	rw	Vertical scaling resize target factor When EISRE=1, this register must not be 0, or greater than VSRSF value.
Bit 15: 13	Reserved	0x0	resd	Kept at its default value.
Bit 12: 0	VSRSF	0x00	rw	Vertical scaling resize source factor When EISRE=1, this register must not be 0,

27.8.15 DVP enhanced frame rate control factor register (DVP_FRF)

Bit	Register	Reset value	Type	Description
Bit 30: 13	Reserved	0x00000	resd	Kept at its default value.
Bit 12: 8	EFRCTF	0x00	rw	Enhanced frame rate control target factor When EFRE=1, this register must not be 0, or greater than EFRSF value.
Bit 7: 5	Reserved	0x0	resd	Kept at its default value.
Bit 4: 0	EFRCSF	0x00	rw	Enhanced frame rate control source factor When EFRCE=1, this register must not be 0

27.8.16 DVP binarization threshold register (DVP_BTH)

Bit	Register	Reset value	Type	Description
Bit 30: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7: 0	MIBTHD	0x00	rw	Monochrome image binarization threshold Monochrome image is binarized based on this threshold. The value above the threshold is regarded as 1, while the value below the threshold is regarded as 0. This configuration is valid only when MIBE=1.

28 Qud-SPI interface (QSPI)

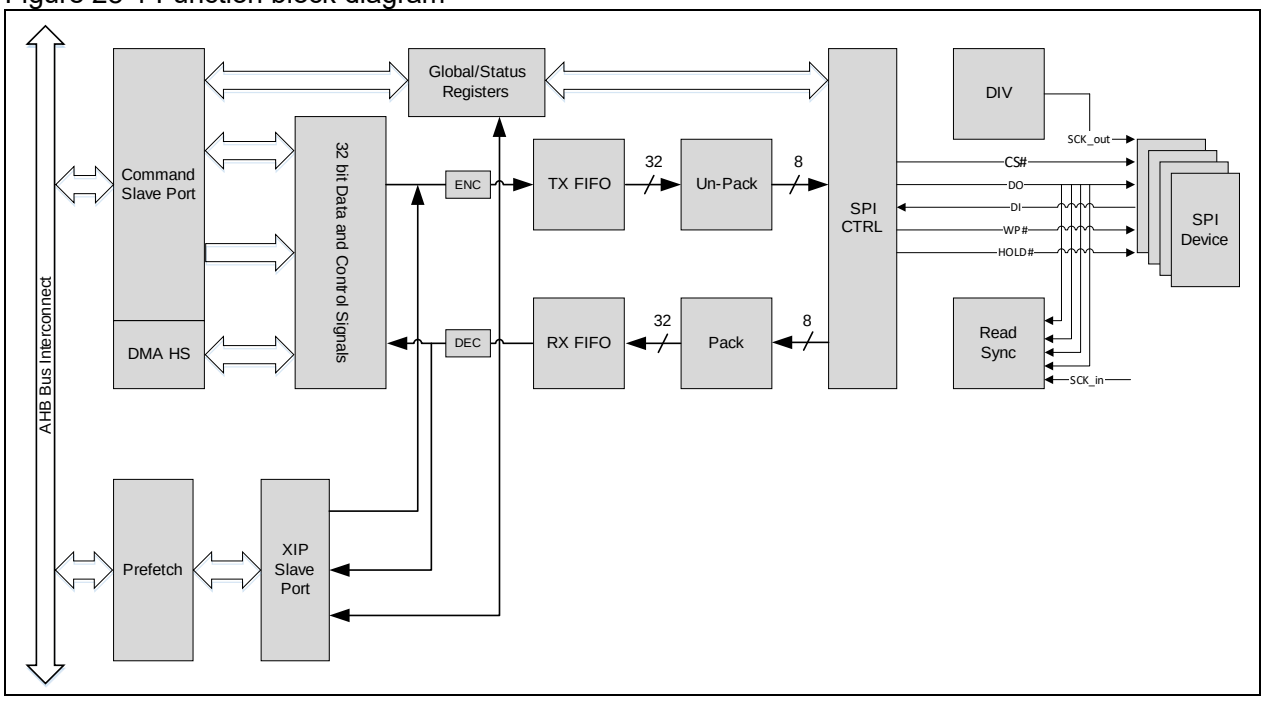
28.1 Introduction

The QSPI interface consists of a command-based slave port, an XIP port (direct address mapping access) and QSPI interface controller used for SPI Flash command execution. The command-based slave port is used to access registers and data ports, and the XIP slave port reads data from direct address mapping. Additionally, the QSPI allows AHB data port to access data in either IPO or DMA mode.

28.2 QSPI main features

- DMA handshake and CPU PIO modes
- SPI mode, dual/quad output mode, dual/quad I/O mode and DPI/QPI mode
- XIP port (direct address mapping read/write)
- XIP port prefetch function (2-channel read cache)
- 128-byte TxFIFO/RxFIFO depth
- Programmable divider
- Data encryption

Figure 28-1 Function block diagram



28.3 QSPI command slave port

28.3.1 QSPI command slave port

The QSPI has a command slave interface that contains register and data ports. The users can access registers or data ports using this interface. The command word register must be written sequentially (CMD_W3 is the last to be written) and is accessible in words, while any other registers (including data port register) can be accessed by bytes, half-words and words.

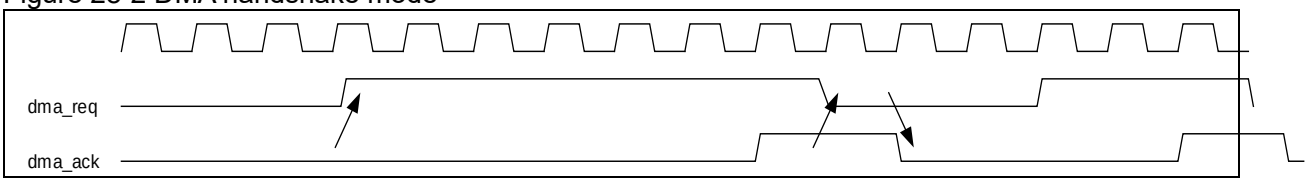
28.3.2 CPU PIO mode

The data port can be accessed in either PIO or DMA mode. The RxFIFO / TxFIFO registers (0x18) must be polled in PIO mode. When the RxFIFO is ready, the user can read or store the entire RxFIFO data. When the TxFIFO is ready, it can be written with the entire data. The polling state must be guaranteed in PIO mode.

28.3.3 DMA handshake mode

The DMA mode can also be used to access data ports. The DMA controller register must be programmed as DMA handshake mode. In this mode, the host controller sends a DMA request when the receive/transmit FIFO threshold is reached. The threshold values are in terms of words. The DMA request is still sent when the last data transfer is less than the threshold.

Figure 28-2 DMA handshake mode



28.3.4 XIP port (direct address mapping read/write)

The QSPI offers a XIP slave port (direct address mapping read/write) for users to perform a direct access to system addresses. In this case, only 0x10 register can be used to select a command slave port or a XIP port, or read other registers than data register (0x100). It should be noted that when reading Flash memory using this XIP port, the Abort function (bit [8] in the 0x10 register) cannot be used (meaning that the bit [20] of the 0x10 register is set to 1), and the XIP port cannot be aborted with this bit, which means that port switch must be performed in sequence. The user's system must have a main Flash memory, in which the user can load the port switch code and perform switching operation.

28.3.5 XIP port prefetch

The XPI port supports read prefetch with 2-channel read cache.

28.3.6 SPI device operation

Command phase

The QSPI operations are conducted through commands, which are used to indicate the intent of the operation and SPI operating mode (multiple-line communication).

Note: the command phase can be set to 1 or 2 bytes. However, in 2-byte mode, it sends two repeated 1-byte commands. Only 1-byte register is used to store command words.

Address phase

The address phase is required to perform read, write, and erase operations on the memory.

In the address phase, the address phase can be from 0 to 4byte in length; In XIP mode, the length of the address phase can be optional 3byte or 4byte.

Dummy-cycles phase

It is used to allow the to-be-read device the time to prepare for the subsequent to-be-read data. XIP and command mode must be configured through DUM2 and XIPR_DUM2, respectively. Besides, the write timing of XIP mode also reserves programmable dummy cycles, which is usually configured to 0 through the XIPW_DUM2 register.

Data phase

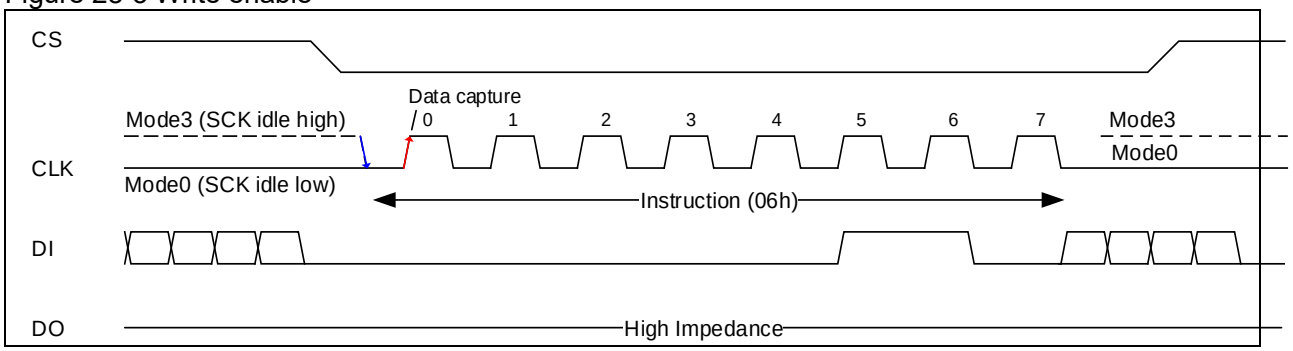
It is used as input when reading memory device data/status registers, and as an output when writing the memory device.

Serial (1-1-1) mode

The host controller supports a SPI protocol (mode 0 and mode 3). The command queue register must be programmed according to SPI device specification. Refer to Figure 28-3 and Figure 28-10 for more information on how to program a command queue.

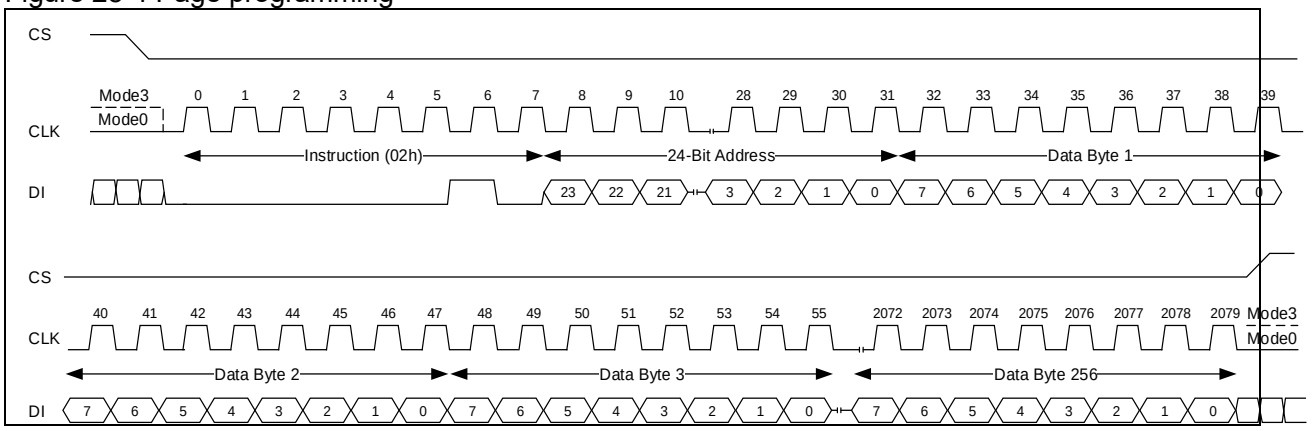
To execute a write enable command, set instruction code to 06h, write enable, and set instruction length to 1. Refer to Figure 28-3 for details.

Figure 28-3 Write enable



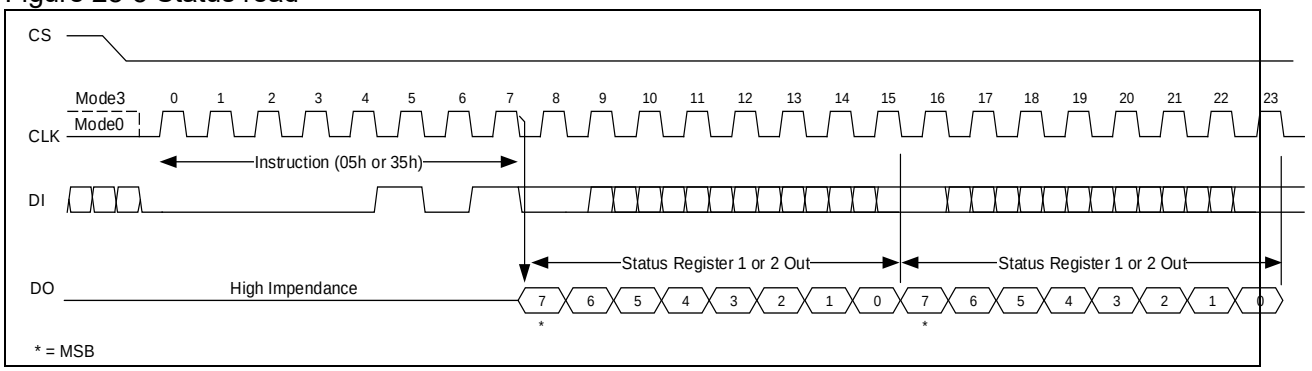
To execute page programming command, set instruction code to 02h, address register, address size, write enable and set instruction length to 1.

Figure 28-4 Page programming



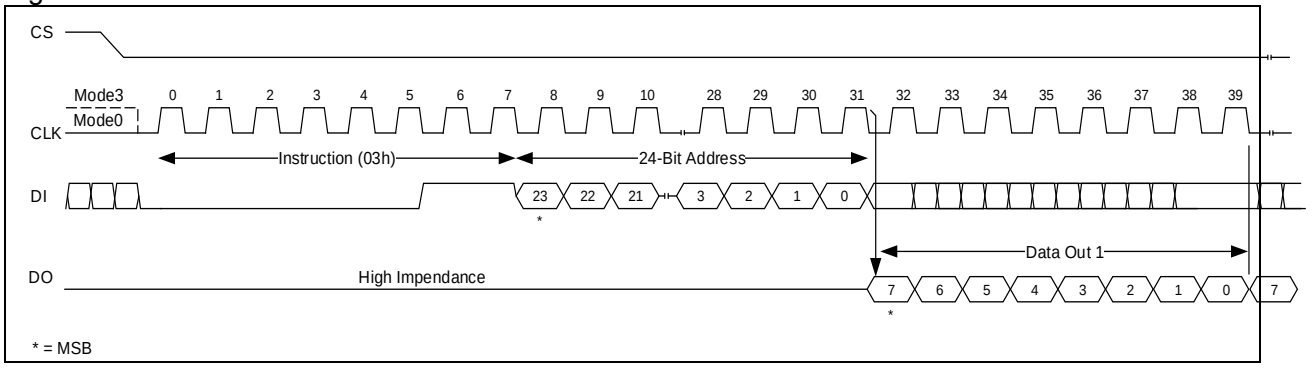
To execute a read status command, set instruction code to 05h/35h, enable read status, select read status through hardware or software, and set instruction length to 1. Refer to Figure 28-5 for more information.

Figure 28-5 Status read



To execute a read data command, set instruction code and length to 1 byte, address/address size to 3 bytes, and set write instruction to 0. Refer to Figure 28-6 for more information.

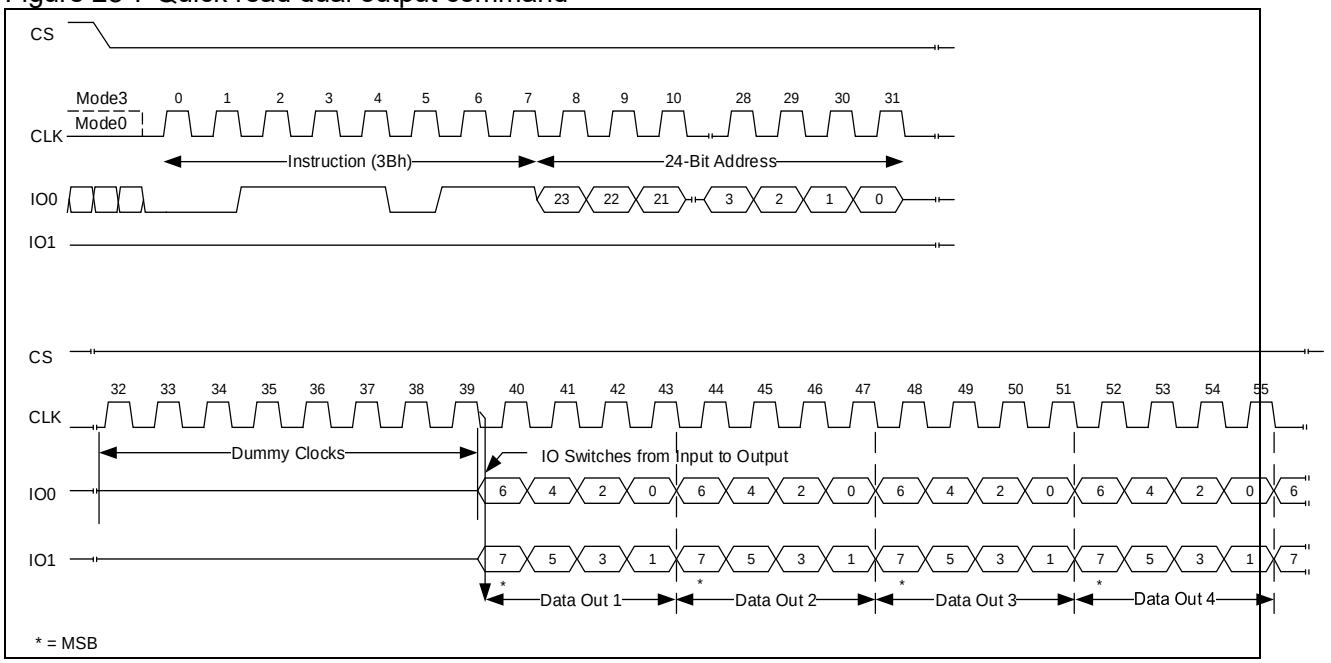
Figure 28-6 Data read



Dual (1-1-2) mode

To execute a quick read dual output command, set instruction code/length to 1, address/address size to 3 bytes, set the second dummy cycle to 8, and enable dual mode. Refer to Figure 28-7 for details.

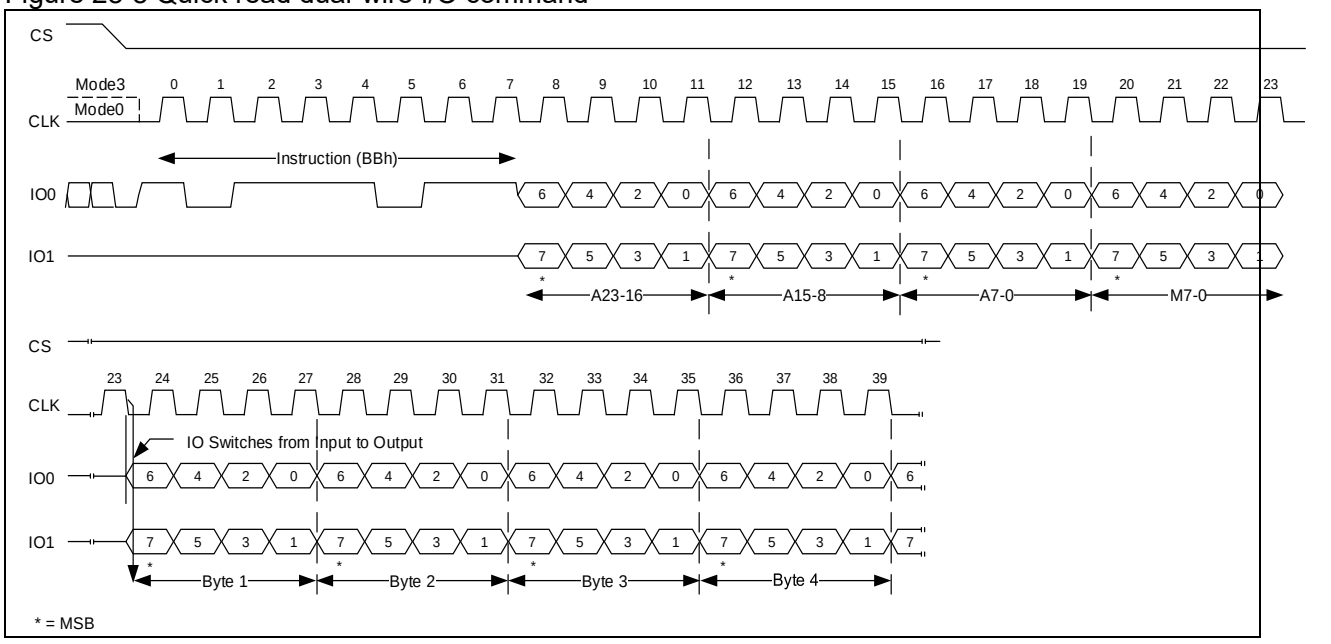
Figure 28-7 Quick read dual output command



Dual I/O (1-2-2) mode

To execute a quick read dual I/O command, set code/ length, address/address size, enable enhanced performance mode/code and dual I/O operation commands. Refer to Figure 28-8 for details.

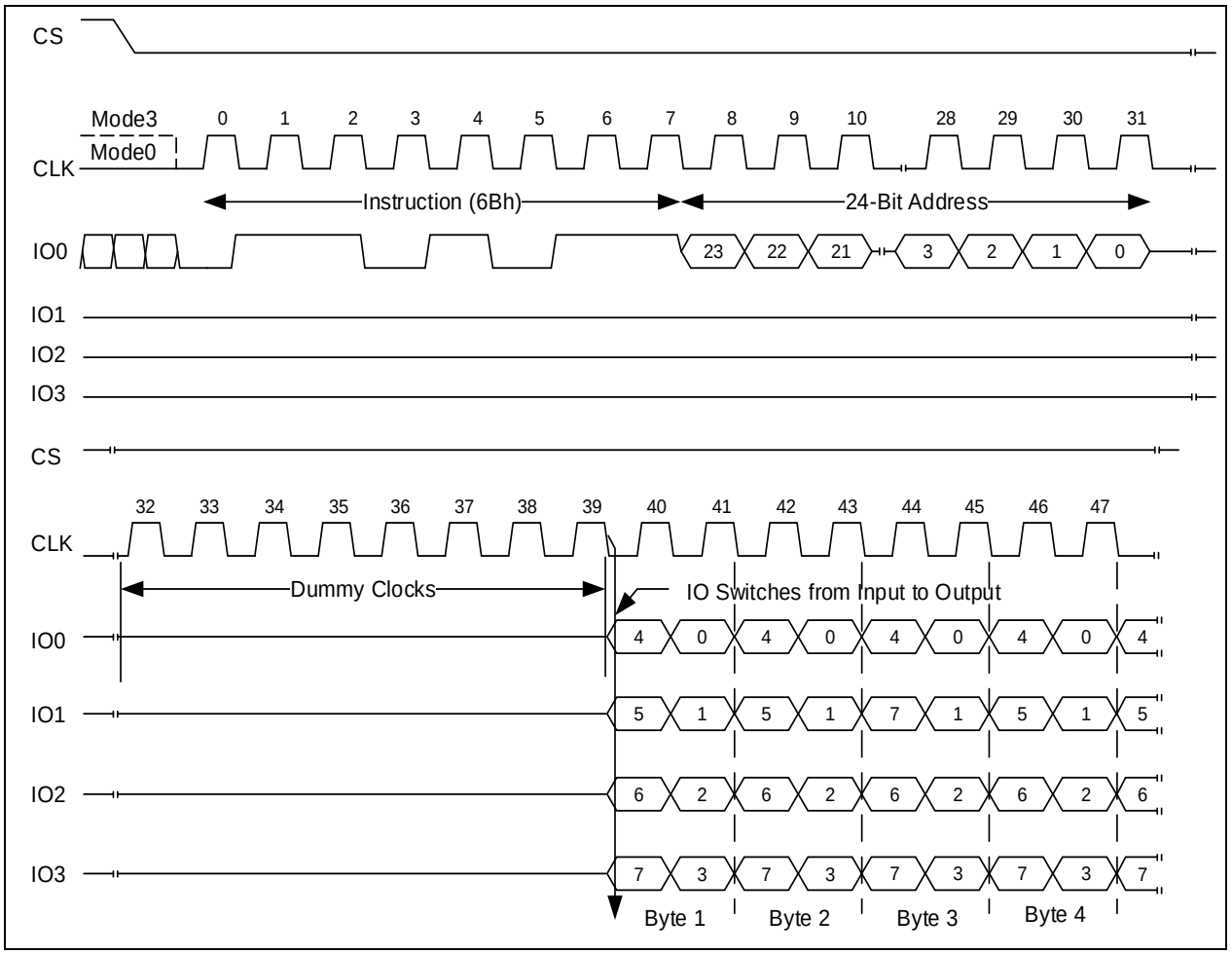
Figure 28-8 Quick read dual-wire I/O command



Quad (1-1-4) mode

To execute a quick read quad command, set instruction code/length, address/address size, enable second dummy period and enable quad mode. Refer to Figure 28-9 for more information.

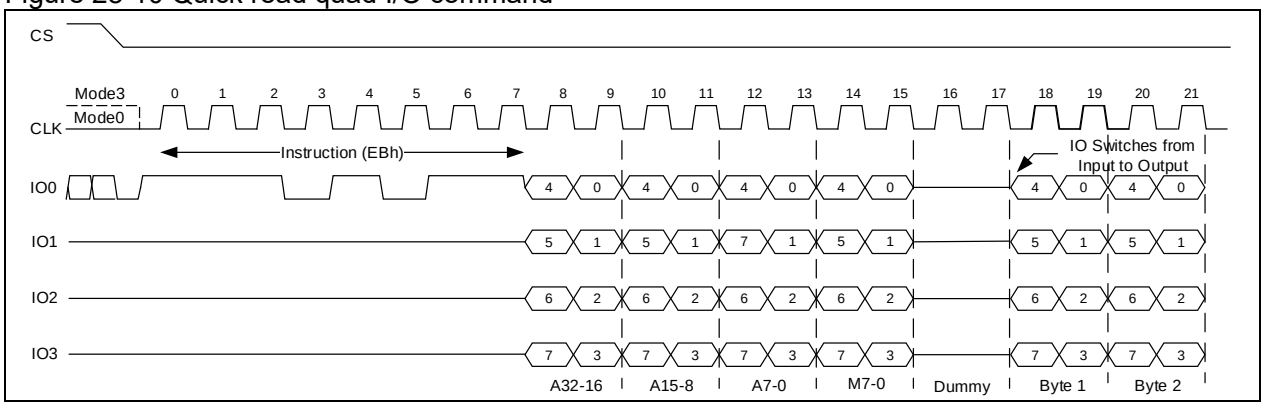
Figure 28-9 Quad read command



Quad I/O (1-4-4) mode

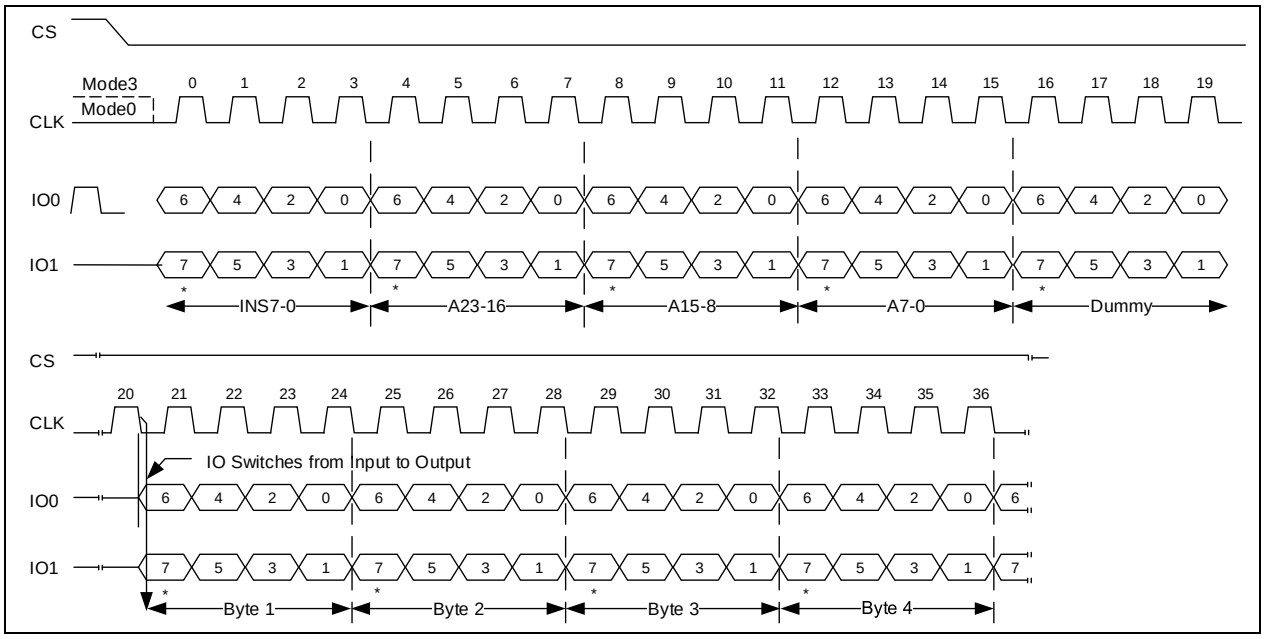
To execute a quick read quad I/I command, set instruction code/length, address/address size, the second dummy period, enable performance enhancement mode and enable quad I/O mode. Refer to Figure 28-10 for more information.

Figure 28-10 Quick read quad I/O command



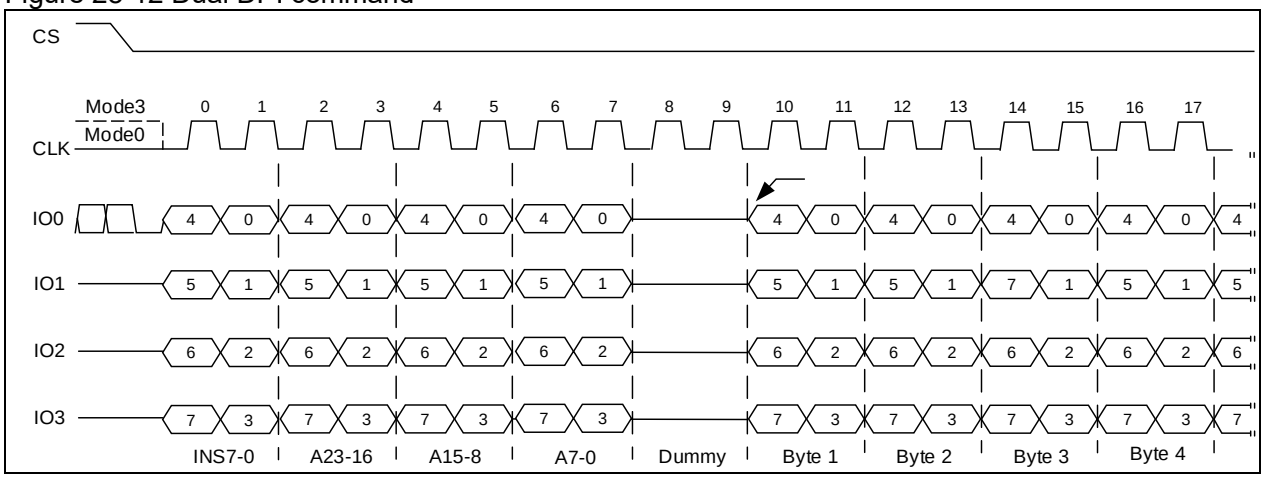
If dual DPI (2-2-2) mode is to be used, it is necessary to set code/length, address/address length, the second dummy cycle and operation command of dual DPI mode. For details, refer to the figure below.

Figure 28-11 Dual DPI command



If quad DPI (4-4-4) mode is to be used, it is necessary to set code/length, address/address length, the second dummy cycle and operation command of quad DPI mode. For details, refer to the figure below.

Figure 28-12 Dual DPI command



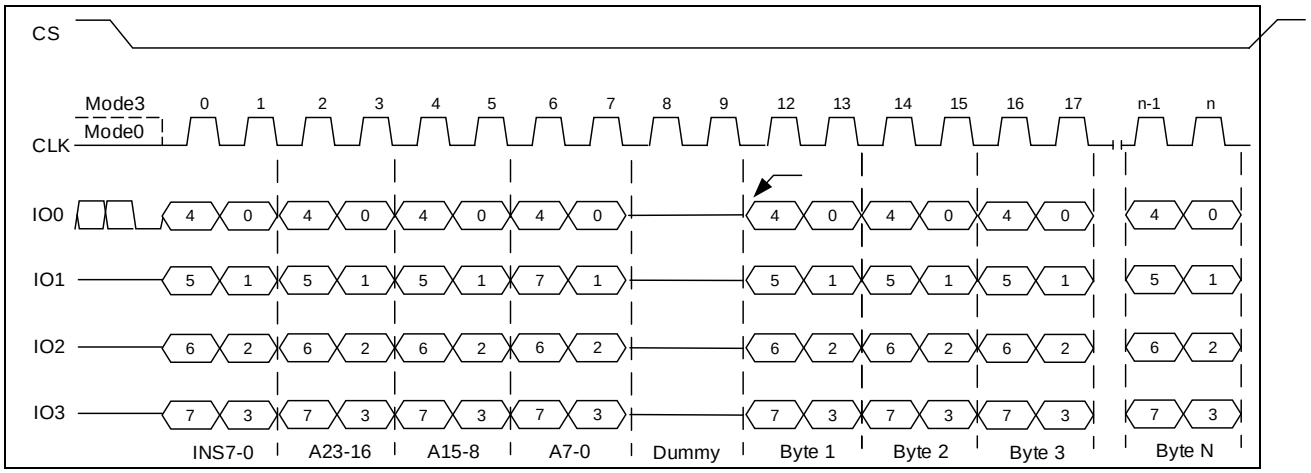
XIP read/write D mode

If XIP read/write D mode is to be used, it is necessary to set code/length, address/address length, the second dummy cycle, select serial/two-line/two-line IO/four-line/four-line IO or DPI/QPI mode, and to set read/write D mode and D mode upper limit counter.

When XIP port is reading or writing back-to-back addresses, it wil keep reading or writing until the higher threshold is reached or the address becomes discontinuous.

When reading, N(byte) = XIPR_DCNT*8; when writing, N(byte) = XIPW_DCNT*8+4.

For more details, refer to the figure below.



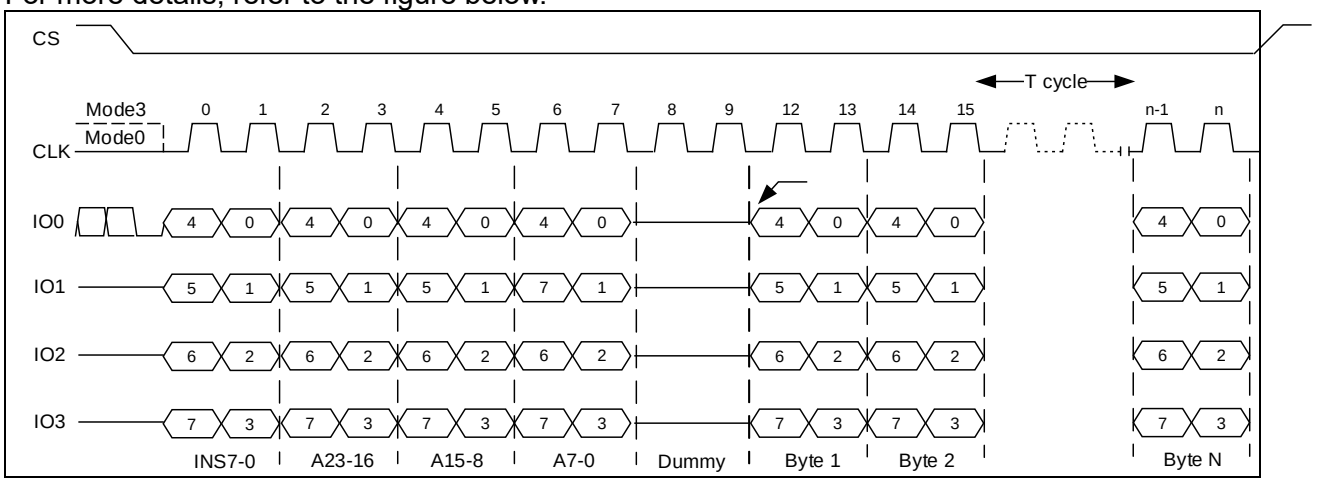
XIP read/write T mode

If XIP read/write T mode is to be used, it is necessary to set code/length, address/address length, the second dummy cycle, select serial/two-line/two-line IO/four-line/four-line IO or DPI/QPI mode, and to set read/write T mode and T mode upper limit counter.

When XIP port is reading or writing back-to-back addresses at intervals, it wil disable CLK wait aftr reading or writing data. If the wait duration is longer than the threshold or when the next address is discontinuous, read/write operations stops.

When reading, T(cycle) = XIPR_TCNT; when writing, T(cycle) = XIPW_TCNT.

For more details, refer to the figure below.



28.4 QSPI registers

These registers must be accessed by bytes (8-bit), half-words (16-bit) or words (32-bit).

Table 28-1 SPI register map and reset values

Register	Offset	Reset value
CMD_W0	0x0	0x0000 0000
CMD_W1	0x4	0x0100 0003
CMD_W2	0x8	0x0000 0000
CMD_W3	0xC	0x0000 0000
CTRL	0x10	0x0010 0083
FIFOSTS	0x18	0x0000 0001
CTRL2	0x20	0x0000 0000
CMDSTS	0x24	0x0000 0000
RSTS	0x28	0x0000 0000
FSIZE	0x2C	0xF0000 0000
XIP CMD_W0	0x30	0x0000 3000
XIP CMD_W1	0x34	0x0000 2000
XIP CMD_W2	0x38	0x0F01 0F01
XIP CMD_W3	0x3C	0x0000 0000
REV	0x50	0x0001 0500
DT	0x100	0x0000 0000

28.4.1 Command word 0 (CMD_W0)

No-wait states, assessable by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	SPIADR	0x0	rw	SPI Flash address This register defines the values of SPI Flash addresses, and sends them to SPI Flash. The address byte is based on the bit [2: 0] of 004h.

28.4.2 Command word 1 (CMD_W1)

No-wait states, assessable by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 29	Reserved	0x0	resd	Kept at its default value.
Bit 28	PEMEN	0x0	rw	Performance enhanced mode enable Locates between the address and the second dummy state. In this mode, the command status after the second read command can be removed. Do not set this bit when CMD_W2=0. 0: Performance enhanced mode disabled 1: 1-byte Performance enhanced mode enabled
Bit 27: 26	Reserved	0x0	resd	Kept at its default value.

Bit 25: 24	INSLEN	0x1	rw	Instruction code length Instruction code is required for SPI Flash command execution. The instruction code length varies from SPI Flash supplier to SPI Flash supplier. Thus this register can be used to program the desired instruction code length. Typically, the instruction code is one-byte length. However, if the user sets two-byte instruction code, the host controller sends this instruction code twice. 00: No instruction code. It cannot be used until the continuous read mode command is completed. 01: 1-byte instruction code 10: 2-byte instruction code (repeated instruction code) 11: Reserved
Bit 23: 16	DUM2	0x0	rw	Second dummy state cycle The second dummy cycle is located between the address and data state, excluding performance enhanced mode status. The user can check there is a dummy state between the address and data status in SPI Flash specification. The host controller sends logic 1 in a dummy cycle. 0: No second dummy state 1~32: 1 dummy second period~32 dummy second period
Bit 15: 3	Reserved	0x0	resd	Kept at its default value.
Bit 2: 0	ADRLEN	0x3	rw	SPI address length This field defines the number of bytes of the SPI Flash address, ranging from one to four bytes. 000: No address state 001: 1-byte address 010: 2-byte address 011: 3-byte address 100: 4-byte address Others: Reserved When the address length is 0, the second dummy period is not preset.

28.4.3 Command word 2 (CMD_W2)

No-wait states, accessible by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	DCNT	0x0	rw	Read/Write data counter This bit must be set to 0 when executing read status command. 0: No read/write data 1~FFFFFFFF: 1~FFFFFFFF byte data Note: This register must not be padded with 0 for data read or write. However, for “read status” or “write enable” instruction, this register must be set to 0.

28.4.4 Command word 3 (CMD_W3)

No-wait states, accessible by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	INSC	0x00	rw	Instruction code This code is set to enable SPI Flash command.
Bit 23: 16	PEMOPC	0x00	rw	Performance enhanced mode operation code This field works with the PEMEN bit. This code can be padded to execute performance enhanced mode. Follow the corresponding Flash specification document to write the corresponding value.

Bit 15: 10	Reserved	0x0	resd	Kept at its default value.
Bit 7: 5	OPMODE	0x0	rw	SPI Operation mode 000: Serial mode (1-1-1) 001: Dual mode (1-1-2) 010: Quad mode (1-1-4) 011: Dual I/O mode (1-2-2) 100: Quad I/O mode (1-4-4) 101: DPI mode (2-2-2) 110: QPI mode (4-4-4) Others: Reserved
Bit 4	Reserved	0x0	resd	Kept at its default value.
Bit 3	RSTSC	0x0	rw	Read SPI status configuration This bit is valid only when read state and write is enabled. The user must send a SPI read state command. 0: Hardware read. The controller keeps polling until the state is ready (not busy) and feedbacks to the status register. 1: Software read. Read status ones and feedback to the status register until the user is able to read it.
Bit 2	RSTSEN	0x0	rw	Read SPI status enable This bit is valid when WEN =“0”, and the user must send SPI read status command. 0: Read SPI status disabled 1: Read SPI status enabled
Bit 1	WEN	0x0	rw	Write data enable This bit is used to enable SPI write data, excluding read data or read status (read data return path); the user must set write enable bit=1 for other SPI commands. Note: Write enable must be set to 1 in data write or Flash erase command. The write enable must be set to 0 only in read data or read status command. 0: Disabled 1: Enabled
Bit 0	Reserved	0x0	resd	Kept at its default value.

28.4.5 Control register (CTRL)

No-wait states, accessible by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x000	resd	Kept at its default value.
Bit 21	KEYEN	0x0	rw	SPI data encryption key enable 0: SPI data encryption key disabled 1: SPI data encryption key enabled When this bit is enabled, raw data is converted into ciphertext and written into the QSPI peripheral through QSPIKEY. While read, data is decrypted into plaintext and sends to the CPU.
Bit 20	XIPSEL	0x1	rw	XIP port selection Read SPI Flash data from the following ports: 0: Command slave port 1: XIP port When this bit is switched, the QSPI sends automatically an Abort signal. The user can send a command only after the completion of Abort function.
Bit 19	XIPRCMDF	0x0	rw	XIP read command flush

Bit 18: 16	BUSY	0x0	rw	Busy bit of SPI status The host polls this busy bit and remains in hardware read state. 000~111: bit 0~bit7
Bit 15: 9	Reserved	0x00	resd	Kept at its default value.
Bit 8	ABORT	0x0	rw	Refresh all commands/FIFOs and reset state machine When an Abort even occurs, this bit must be written (This bit is automatically cleared to 0). 0: No effect 1: Enabled
Bit 7	XIPIDLE	0x1	ro	XIP port idle status 0: XIP port is busy 1: XIP port is idle
Bit 6: 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	SCKMODE	0x0	rw	Sckout mode 0: For mode 0, sck_out is low in idle state, with data capture on the first edge 1: For mode 3, sck_out is high in idle state, with data capture on the second edge
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2: 0	CLKDIV	0x3	rrw	Clk divider 000: Divided by 2 001: Divided by 4 010: Divided by 6 011: Divided by 8 100: Divided by 3 101: Divided by 5 110: Divided by 10 111: Divided by 12

28.4.6 AC timing register (ACTR)

No-wait states, accessible by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 4	Reserved	0x0000 000	resd	Kept at its default value.
Bit 3: 0	CSDLY	0xF	rw	cs delay Indicates the time from inactive cs to active us, meaning a timing from high to low. Refer to the corresponding specification for more information. The value is in terms of sck_out period, which is 16 sck_out period by default. 0000~1111: 1 period~16 periods

28.4.7 FIFO status register (FIFOSTS)

No-wait states, accessible by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 2	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 1	RXFIFORDY	0x0	ro	RxFIFO ready status When this bit is set, it indicates the following: 1: RxFIFO full 2: The remaining data in the RxFIFO is less than the depth of RxFIFO, but it is the last data.

Bit 0	TXFIFORDY	0x1	ro	TxFIFO ready status When the TxFIFO is ready, it indicates that the TxFIFO will get empty so that data can be transmitted into it until it becomes full.

28.4.8 Control register 2 (CTRL2)

No-wait states, accessible by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 14	Reserved	0x0000 0	resd	Kept at its default value.
Bit 13: 12	RXFIFO THOD	0x0	rw	This field is used to program the level value to trigger RxFIFO threshold interrupt for DMA handshake mode. The value is in terms of word. The trigger value is the data in the RxFIFO. 00: 8 WORD 01: 16 WORD 10: 24 WORD 11: Reserved
Bit 11: 10	Reserved	0x0	resd	Kept at its default value.
Bit 9: 8	TXFIFO THOD	0x0	rw	This field is used to program the level value to trigger TxFIFO threshold interrupt for DMA handshake mode. The value is in terms of word. The trigger value is the data in the TxFIFO. 00: 8 WORD 01: 16 WORD 10: 24 WORD 11: Reserved
Bit 7: 2	Reserved	0x00	resd	Kept at its default value.
Bit 1	CMDIE	0x0	rw	Command complete Interrupt enable 0: Command complete Interrupt disabled 1: Command complete Interrupt enabled
Bit 0	DMAEN	0x0	rw	DMA enable Note: This bit must be disabled before moving from command-based slave port to XIP port.

28.4.9 Command status register (CMDSTS)

No-wait states, accessible by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 1	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 0	CMDSTS	0x0	rw1c	Command complete status Set at the end of a command.

28.4.10 Read status register (RSTS)

No-wait states, accessible by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 1	Reserved	0x0000 00	resd	Kept at its default value.
Bit 7: 0	SPISTS	0x00	ro	SPI Read status The host sends a read SPI Flash status command and stores the returned data in this register. By reading it, the user can check the status of SPI Flash.

28.4.11 Flash size register (FSIZE)

No-wait states, accessible by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	SPIFSIZE	0xF000 0000	rw	SPI Flash Size In direct address map mode, system address is always greater than that of SPI Flash. The user must mask the upper bits of the system address to match SPI Flash size.

28.4.12 XIP command word 0 (XIP CMD_W0)

No-wait states, accessible by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 20	Reserved	0x000	resd	Kept at its default value.
Bit 19: 12	XIPR_INSC	0x03	rw	XIP read instruction code This field is set to execute SPI Flash command of XPI read.
Bit 11	XIPR_ADRLEN	0x0	rw	XIP read address length This bit defines the number of bytes of SPI Flash address. The user can uses this bit to program a 3-byte or 4-byte XIP read address. 0: 3-byte address 1: 4-byte address
Bit 10: 8	XIPR_OPMODE	0x0	rw	XIP read Operation mode 000: Serial mode (1-1-1) 001: Dual mode (1-1-2) 010: Quad mode (1-1-4) 011: Dual IO mode (1-2-2) 100: Quad IO mode (1-4-4) 101: DPI mode (2-2-2) 110: QPI mode (4-4-4) 111: Reserved
Bit 7: 0	XIPR_DUM2	0x00	rw	XIP Read second dummy cycle The second dummy state is located between the address and data status, excluding continuous read mode status. The user can check if there is a dummy state between the address and data status in SPI Flash specification. The host controller issues logic 1 in a dummy cycle. 0: No second dummy state 1~32: 1 dummy second period~32 dummy second periods

28.4.13 XIP command word 1 (XIP CMD_W1)

No-wait states, accessible by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 18	Reserved	0x000	resd	Kept at its default value.
Bit 19: 12	XIPW_INSC	0x02	rw	XIP write instruction code The user can set this field to enable a SPI Flash command of XIP write.
Bit 11	XIPW_ADRLEN	0x0	rw	XIP write address length This bit defines the number of bytes of SPI Flash address. The user can uses this bit to program a 3-byte or 4-byte XIP write address. 0: 3-byte address 1: 4-byte address

Bit 10: 8	XIPW_OPMODE	0x0	rw	XIP write operation mode 000: Serial mode (1-1-1) 001: Dual mode (1-1-2) 010: Quad mode (1-1-4) 011: Dual IO mode (1-2-2) 100: Quad IO mode (1-4-4) 101: DPI mode (2-2-2) 110: QPI mode (4-4-4) 111: Reserved
Bit 7: 0	XIPW_DUM2	0x00	rw	XIP Write second dummy cycle The second dummy state is located between the address and data status, excluding continuous read mode status. The user can check if there is a dummy state between the address and data status in SPI Flash specification. The host controller issues logic 1 in a dummy cycle. 0: No second dummy state 1~32: 1 dummy second period~32 dummy second periods

28.4.14 XIP command word 2 (XIP CMD_W2)

No-wait states, accessible by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31	XIPW_SEL	0x0	rw	XIP write mode select 0: Mode D 1: Mode T The XIP slave port can be used for the improvement of read/write process and performance. Mode D: When data are written to the consecutive addresses, put a limit on the maximum data count (DCNT) in a single write operation Mode T: When two consecutive data are written and the addresses are also continuous, and the interval is less than a specified time (TCNT), then they are combined into one command.
Bit 30: 24	XIPW_TCNT	0x0F	rw	This indicates the time counter that is used to judge time interval in mode T. Value is in terms of sck_out period. This counter is valid when mode T is selected.
Bit 23: 22	Reserved	0x0	resd	Kept at its default value.
Bit 21: 16	XIPW_DCNT	0x01	rw	This indicates the time counter that is used to judge the maximum data count in mode D. Value is in terms of word, and must not be 0. This counter is valid when mode D is selected.
Bit 15	XIPR_SEL	0x0	rw	XIP read mode select 0: Mode D 1: Mode T The XIP slave port can be used for the improvement of read/write process and performance. Mode D: When data are read from the consecutive addresses, put a limit on the maximum data count (DCNT) in a single write operation Mode T: When two consecutive data are read and the addresses are also continuous, and the interval is less than a specified time (TCNT), then they are combined into one command.

Bit 14: 8	XIPR_TCNT	0x0F	rw	This indicates the time counter that is used to judge time interval in mode T. Value is in terms of sck_out period. This counter is valid when mode T is selected.
Bit 7: 6	Reserved	0x0	resd	Kept at its default value.
Bit 5: 0	XIPR_DCNT	0x01	rw	This indicates the time counter that is used to judge the maximum data count in mode D. Value is in terms of word, and must not be 0. This counter is valid when mode D is selected.

28.4.15 XIP command word 3 (XIP CMD_W3)

No-wait states, accessible by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 4	Reserved	0x0000 000	resd	Kept at its default value.
Bit 3	CSTS	0x0	r	Cache Status 0: Cache verified 1: Cache failed
Bit 2: 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	BYPASSC	0x0	rw	Bypass Cache Function When this bit is set, the high-speed cache feature is deactivated, and all read transfers do not check high-speed cache.

28.4.16 Revision register (REV)

No-wait states, accessible by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	REV	0x0001 0500	ro	Indicates IP version.

28.4.17 Data port register (DT)

No-wait states, accessible by bytes, half-words and words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	DT	0x0000 0000	rw	Data port register This port is used for data read or write.

29 EDMA controller (EDMA)

29.1 Introduction

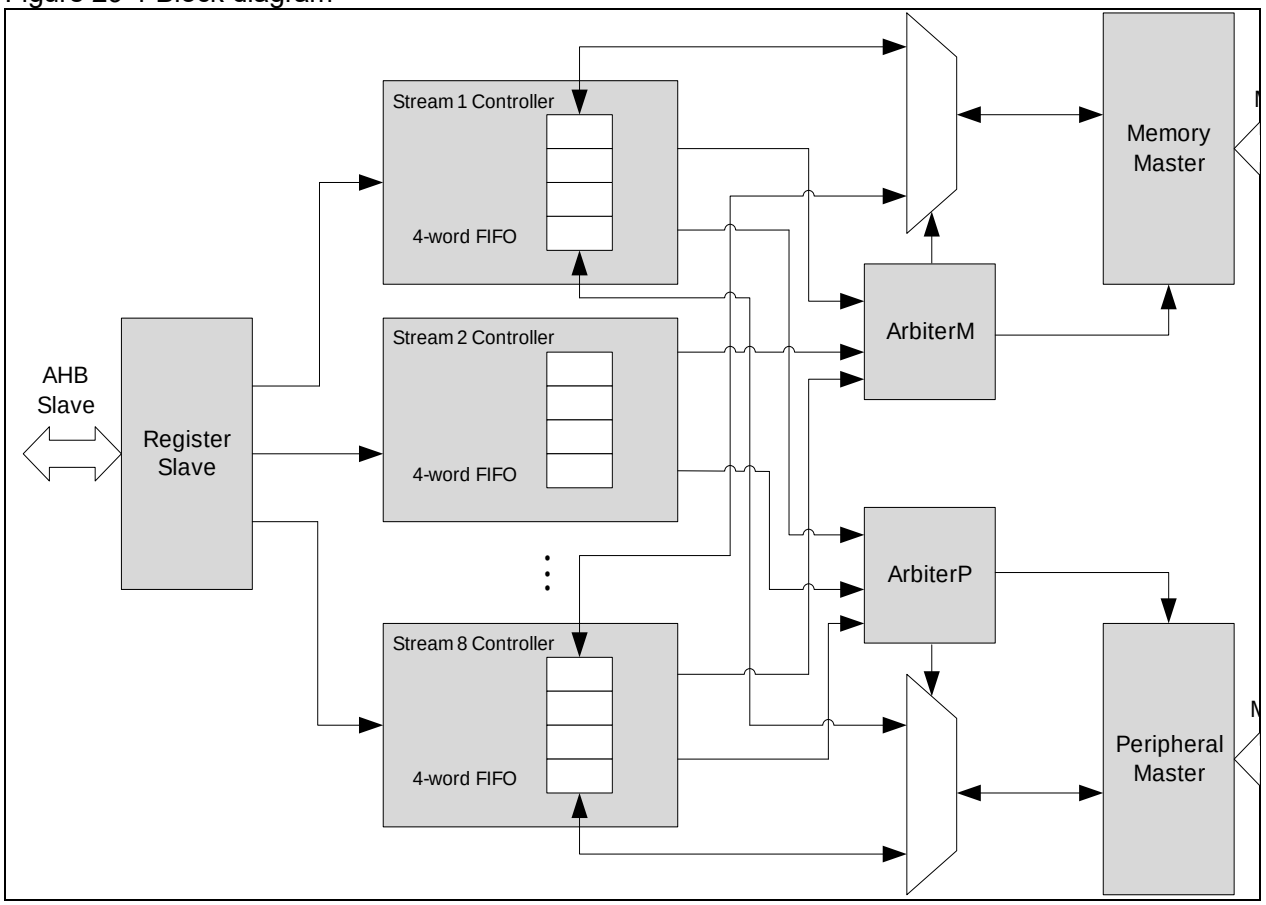
Enhanced direct memory access (EDMA) controller is designed for 32-bit MCU applications with the goal of enhancing system performance and reducing the generation of interrupts.

This controller offers 8 DMA channels to guarantee data transfers between peripheral-to-memory, memory-to-peripheral, and memory-to-memory.

29.2 Main features

- AMBA compliant (Rev. 2.0)
- Only support AHB OKAY and ERROR responses
- HBUSREQ and HGRANT of AHB master interface are not supported
- Two AHB main interfaces for data transfer
- Support 8 channels
- Peripheral-to-memory, memory-to-peripheral, and memory-to-memory transfers
- Support hardware handshake
- Support 8-bit, 16-bit and 32- bit data transfers
- Programmable amount of data to be transferred: up to 65535
- Programmable chain transmission
- Programmable 2D transfer
- Support multiplexer

Figure 29-1 Block diagram



29.3 Functional overview

29.3.1 Flow configuration

1. **Set the peripheral address in the DMA_CxPADD register**
   The initial peripheral address for data transfer remains unchanged during transmission.
2. **Set the memory address in the DMA_CxMADDR register**
   The initial memory address for data transfer remains unchanged during transmission.
3. **Configure the amount of data to be transferred in the DMA_CxDTCNT register**
   Programmable data transfer size is up to 65535. This value is decremented after each data transfer.
4. **Configure the channel setting in the DMA_CxCTRL register**
   Including channel priority, data transfer direction/width, address incremented mode, circular mode and interrupt mode

- **Channel select (CHSEL)**

Each of the streams can be selected up to 8 possible channel requests

- **Channel priority (SPL)**

There are four levels, including very high priority, high priority, medium priority and low priority.
If two channels have the same priority level, the channel with lower number will get priority over the one with higher number. For example, stream 1 has priority over stream 2.

- **Data transfer direction (DTD)**

Memory-to-peripheral (M2P), peripheral-to-memory (P2M) or memory-to-memory (M2M)

In M2M mode, circular mode, dual memory mode and direct mode cannot be used.

- **Address incremented mode (PINCM/MINCM)**

In incrementing mode, the subsequent transfer address is the previous address plus transfer width (PWIDTH/MWIDTH).

- **Data transfer mode (PBURST/MBURST)**

Single transfer or burst transfer.

In non-incrementing mode, burst transfers of 4, 8 or 16 beats are translated into 4, 8 or 16 single transfers.

In direct mode, PBURST and MBURST bits are forced to 0 (single transfer mode)

- **Peripheral flow control (PFCTRL)**

If PFCTRL = 1 (peripheral flow control is used), the value of the DMA_SxDTCNT register is forced to 0xFFFF. The dma_ch_lt signal of the peripherals indicates the completion of data transfer. The M2M mode and circular mode cannot be used in Peripheral Flow Control.

- **Circular mode (LM)**

In circular mode, the contents in the DMA_CxDTCNT register is automatically reloaded with the initially programmed value after the completion of the last transfer.

- **Dual memory mode (DMM)**

When the flow is configured in dual memory mode, the hardware will automatically enable circular mode. Two memory pointers (DMA_SxM0ADR/ DMA_SxM1ADR) are available in dual memory flow. The CM bit indicates which one of the memory pointers is being used. The software is allowed to process another memory while the DMA is filling up or using the second memory.

5. **FIFO settings through the DMA_SxFCTRL register**

This includes FIFO threshold value select, direct mode disable and FIFO error interrupt enable bit.

FIFO threshold select (FTHSEL)

1/4, 2/4, 3/4 and full FIFO threshold values are available for selection. This bit is applicable to the non-direct mode only.

FIFO mode (FEN)

Direct mode (FEN = 0) can be used only in P2M or M2P mode, and the transfer width of peripherals and memories must be equal (MWIDTH = PWIDTH), and single transfer mode is enabled (PBURST = MBURST = 0). This bit (FEN) will be forced to 1 in M2M mode.

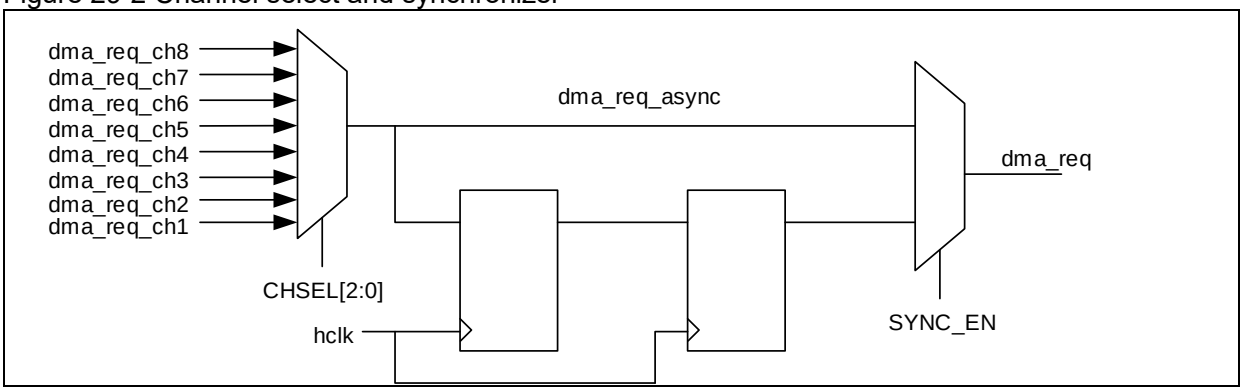
6. **Enable DMA transfer through the SEN bit in the DMA_SxCTRL register**

29.3.2 Channel selection and synchronizer

The clocks between a peripheral and DMA may be asynchronous. The user can choose whether the synchronizer of the dma_chx_req is needed.

Figure 29-2 shows the block diagram of channel selection and synchronizer.

Figure 29-2 Channel select and synchronizer



29.3.3 Handshake mechanism

In P2M and M2P mode, the peripherals need to send a request signal to the DMA controller. The DMA channel will send the peripheral transfer request (single or burst) until the signal is acknowledged. After the completion of peripheral transmission, the DMA controller sends an acknowledge signal to the peripheral. The peripheral then releases its request as soon as it receives the acknowledge signal. At the same time, the DMA controller releases the acknowledge signal as well.

In non-incrementing mode (PINCM = 0), burst transfers 4, 8 or 16 beats are translated into 4, 8 or 16 single transfers. These data will be transferred by the main peripheral controller in a request/acknowledge group.

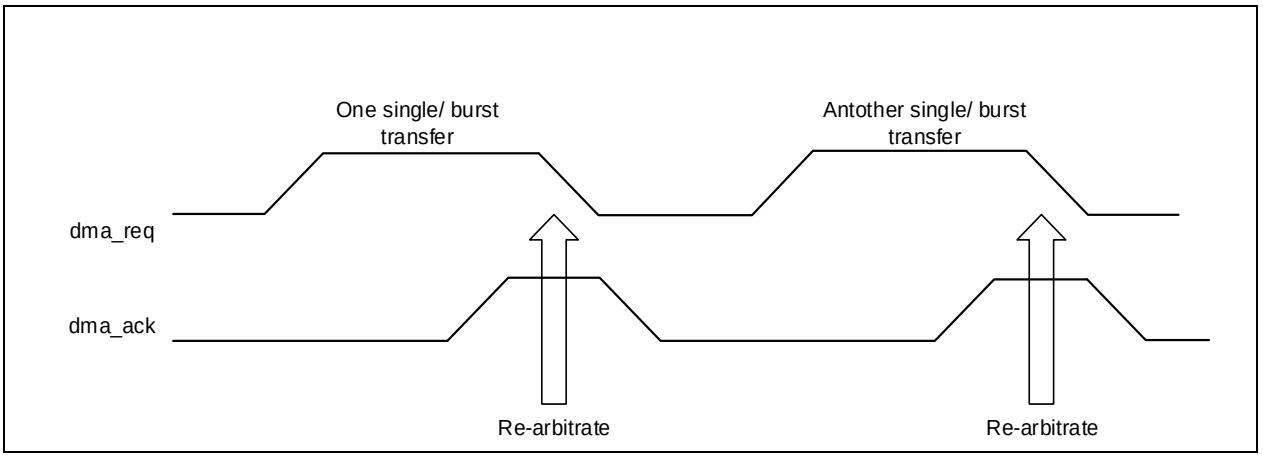
In peripheral flow control mode (PFCTRL = 1), only a single transfer is present in a request/ acknowledge group.

29.3.4 Arbiter

When several channels are enabled simultaneously, the arbiter will restart arbitration after full data transfer by the master controller. The channel with very high priority waits until the channel of the master controller has completed data transfers before taking control of it. The master controller will re-arbitrate to serve other channels as long as the channel completes a single transfer based on the master controller priority.

Figure 29-3 shows the behavior between DMA request/acknowledge and arbiter. Since there is only one single or burst transfer in a request/acknowledge group, it works in the same way as a re-arbitration mechanism. Thus the main peripheral controller will re-arbitrate to serve other channels each time a channel completes a single or burst transfer according to the priority defined by the main controller.

Figure 29-3 Re-arbitrate after Request/Acknowledge



In non-incrementing mode (PINCM = 0 or MINCM = 0), burst transfers of 4, 8 or 16 beats are translated into 4, 8 or 16 single transfers. In this case, the main controller will transfer all data before re-arbitration.

29.3.5 Programmable data transfer width

Transfer width of the source data and destination data is programmable through the PWIDTH and MWIDTH bits in the DMA_CxCTRL register. When PWIDTH is not equal to MWIDTH, a packing and unpacking mechanism is performed. When the transfer size of the source data is less than that of the destination data, the data is packed in FIFO before sending to the destination. On the contrary, if the transfer size of the source data is greater than that of the destination data, the data is unpacked before sending to the destination.

PWIDTH, MWIDTH and CNT must be configured correctly to ensure complete data transfer.

CNT = (MWIDTH/ PWIDTH) × N

- MWIDTH and PWIDTH stand for data transfer size = 1 (byte), 2 (half-words) or 4 (words)
- CNT and N are positive integers.

Figure 29-4 Example of packing mechanism

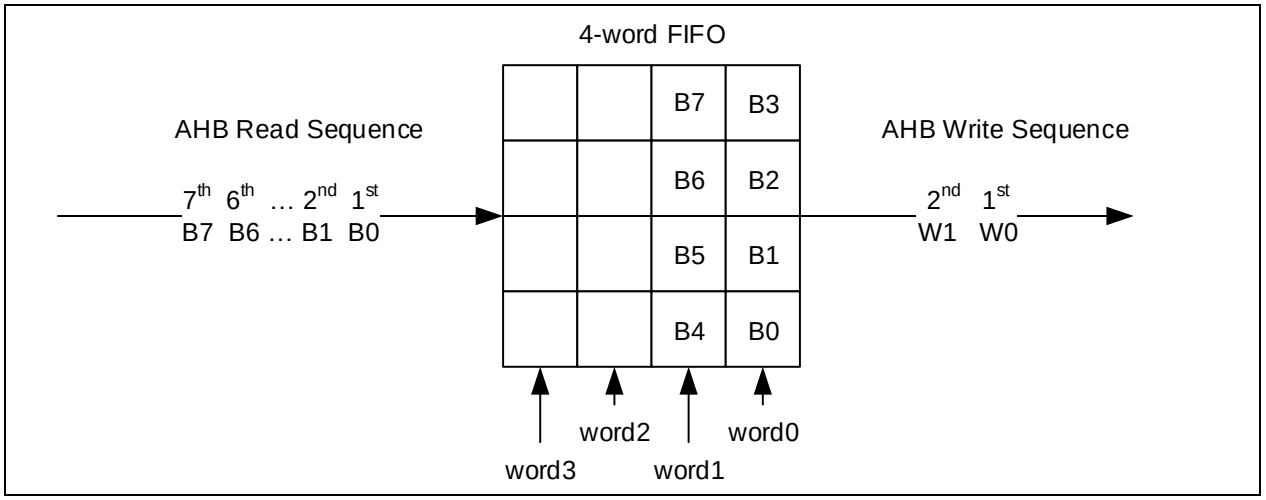


Figure 29-5 Example of unpacking mechanism

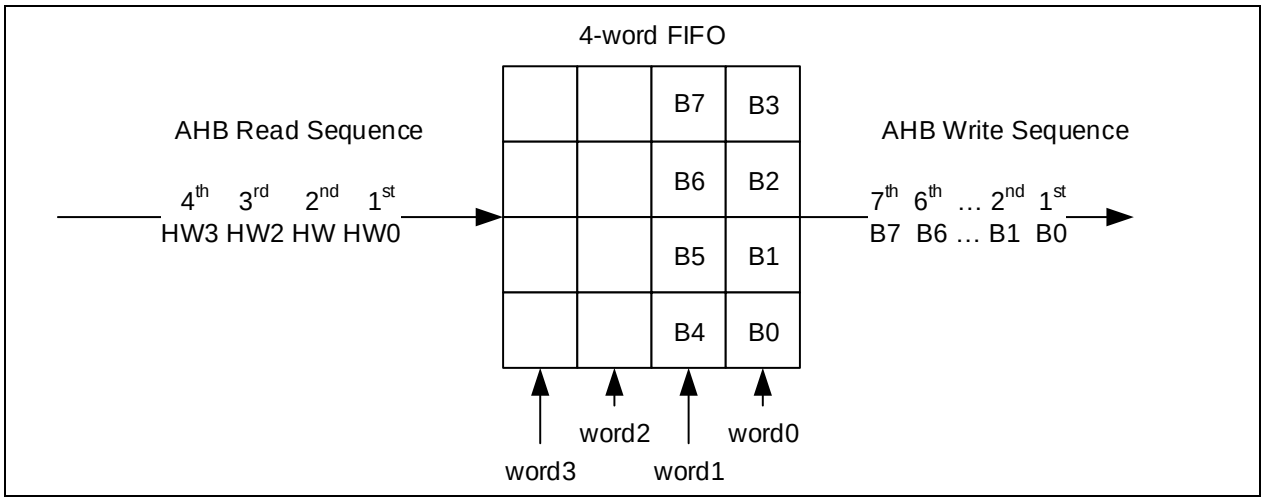
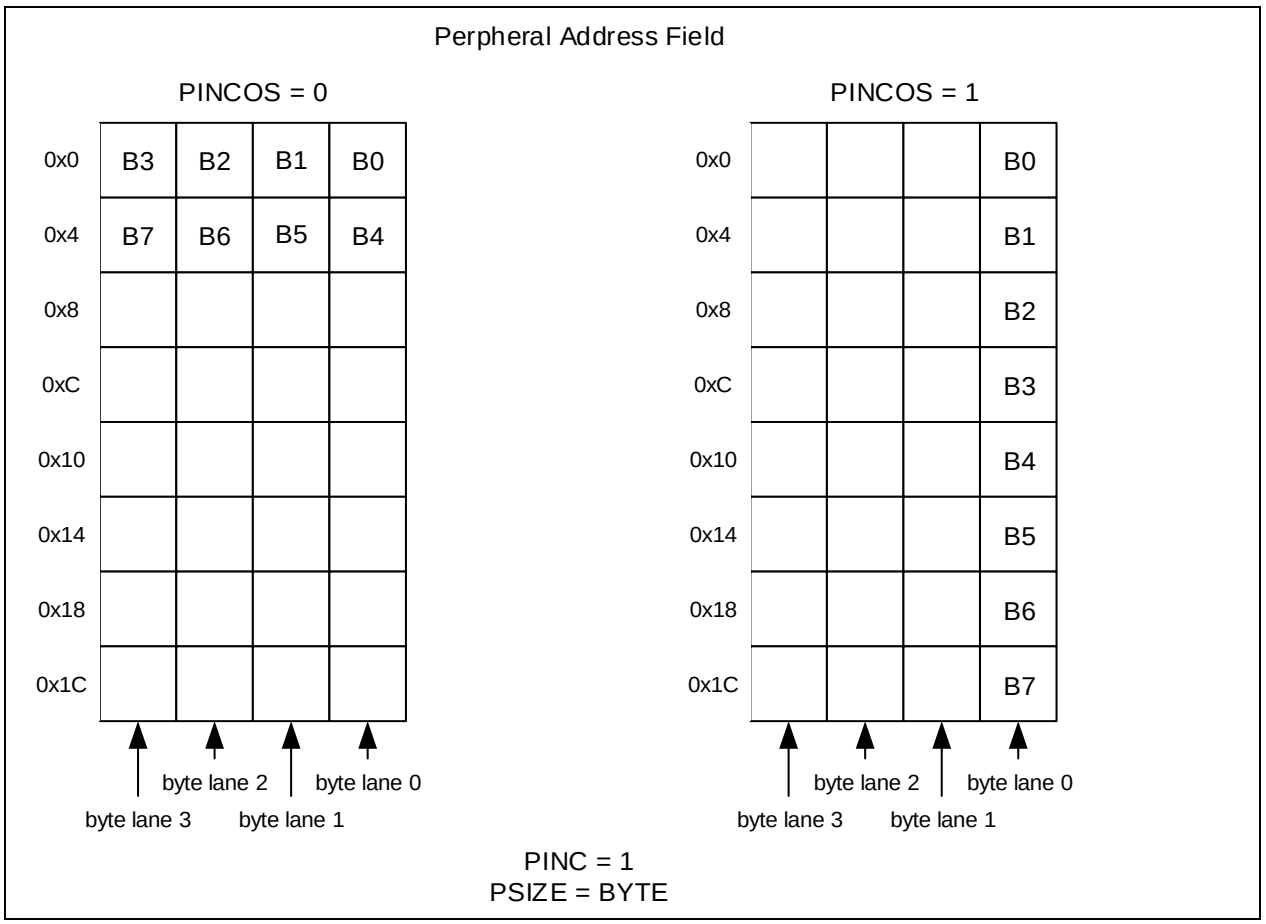


Figure 29-6 Example of PINCOS



29.3.6 FIFO and threshold

Each stream has its dedicated 4-word FIFO that is programmed with different threshold (1/4, 1/2, 3/4 or full). The FTHSEL bit must match the MBURST bit, otherwise a FIFO error (FERR) is generated while enabling a stream, causing that a stream is automatically disabled.

FTHSEL×4 = MBURST×MWIDTH×M

- FTHSEL (FIFO threshold) = 1 (1/4), 2 (1/2), 3 (3/4) or 4 (Full)
- MBURST (data transfer size) = 1 (single) or 4, 8 or 16 burst
- MWIDTH (data transfer width) = 1 (byte), 2 (half-word) or 4 (word)
- M is a positive integer

Additionally, the MBURST/PBURST and MWIDTH/PWIDTH bits must match the CNT bit for full data transfers.

CNT = PBURST×PWIDTH×N = MBURST×MWIDTH×M

- MBURST/PBURST (data transfer size) = 1 (single) or 4, 8 or 16 burst
- MWIDTH/PWIDTH (data transfer width) = 1 (byte), 2 (half-word) or 4 (word)
- M, N, CNT are positive integers

If mismatch, when the remaining data is les that the MBURST/PBURST bits, they are transmitted by hardware in a single mode, and set the FDTF bit after the completion of refresh. If the remaining data is less than the MWIDTH/PWIDTH, data are still transmitted based on the width programmed in the MWIDTH bit.

29.3.7 Linked table transfer mechanism

Using this mechanism, it is possible to link several different transfers to strengthen DMA processing capabilities. Each transfer data can be stored in a descriptor by software, and the DMA loads the descriptor from the main memory.

To enable a linked table transfer, set the SxLLSEN (where n is a stream number) bit in the DMA_LLCTRL register and the local descriptor memory base address, and prepare LDM or main memory descriptor. Then, the user can continue to set DMA_SxDTCNT=0, write the descriptor address to the DMA_SxLLP and enable a stream.

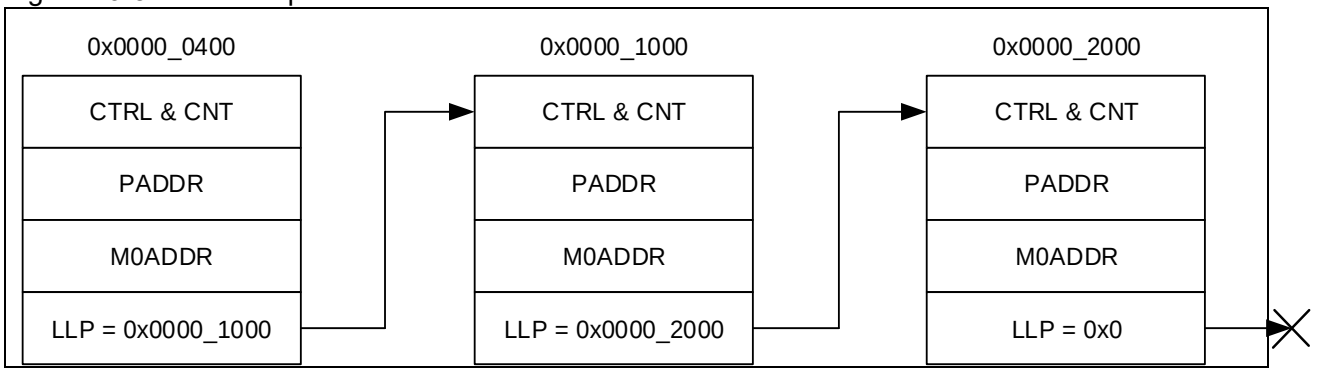
After enabling a stream, the memory controller in DMA issues a few AHB commands to load the descriptor from the main memory.

Figure 29-7 gives the descriptor format. After loading descriptors (that is, CTRL and CNT, PADDR and M0ADDR), the DMA uses these bits for data transfers. When a descriptor transaction is completed (CNT becomes 0), if the current LLP is not equal to zero, the DMA continues to load the next descriptor. If the current LLP is zero, it means that the current descriptor is the last one, and the DMA stops loading a new one. At this point, a transfer completed interrupt (FDTF bit) is generated to mark the completion of a linked table transfer. It should be noted that the TC is generated at the last descriptor.

Figure 29-7 Descriptor format

		bit offset																													
		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
	CTRL & CNT	FEN	PINCOS	SPL		MBURST		PBURST		MWIDTH		PWIDTH		MINCM	PINCM	DTD		DTCNT[15:0]															
+4	PADDR	PADDR[31:0]																													
+8	M0ADDR	M0ADDR[31:0]																													
+C	LLP	LLP[31:0]																													

Figure 29-8 Linked list pointers



29.3.8 2D transfer mechanism

The 2-dimensional (2D) transfer mechanism makes it easier to block data like images. The DMA controller offers four types of configurations for 2D transfers.

- XCOUNT bit in DMA_Sx2DCNT: Data count to be transmitted before jumping to the next stride
- YCOUTNT bit in DMA_Sx2DCNT: Iteration count
- SRCSTD bit in DMA_Sx2DSTRIDE: Source stride value. This value should be added or subtracted before source iteration
- DSTSTD bit in DMA_Sx2DSTRIDE: Destination stride value. This value should be added or subtracted before destination iteration.

Example of a 2D transfer:

Source stride (byte address)	Destination stride (byte address)	XCNT	YCNT
0x20	0x10	0x4	0x8

As shown in Figure 29-9, the user can read 2D-block data from the source memory. The XCOUNT is a 4-beat data (4 beat means four words=16 bytes) that is read by DMA before iteration. After reading data from the word address 0x0 to 0x3 (byte address from 0x0 to 0xF), the next read address is the source address plus source stride (0x0 + 0x20 = 0x20 byte address=0x8 word address). Therefore, the DMA can perform the second iteration and read four data beats from the slave word address 0x8, 0x9, 0xA and 0xB (byte address ranging from 0x20 to 0x2F). The YCOUTNT bit has up to 8 iterations, which is equal to 0x8. When both the YCOUTNT and XCOUNT bits become 0, it indicates that a total of 32-word data (8 iterations *4 beats) have been read from the source memory.

Figure 29-9 Example of a 2D transfer (source side is managed by a peripheral controller)

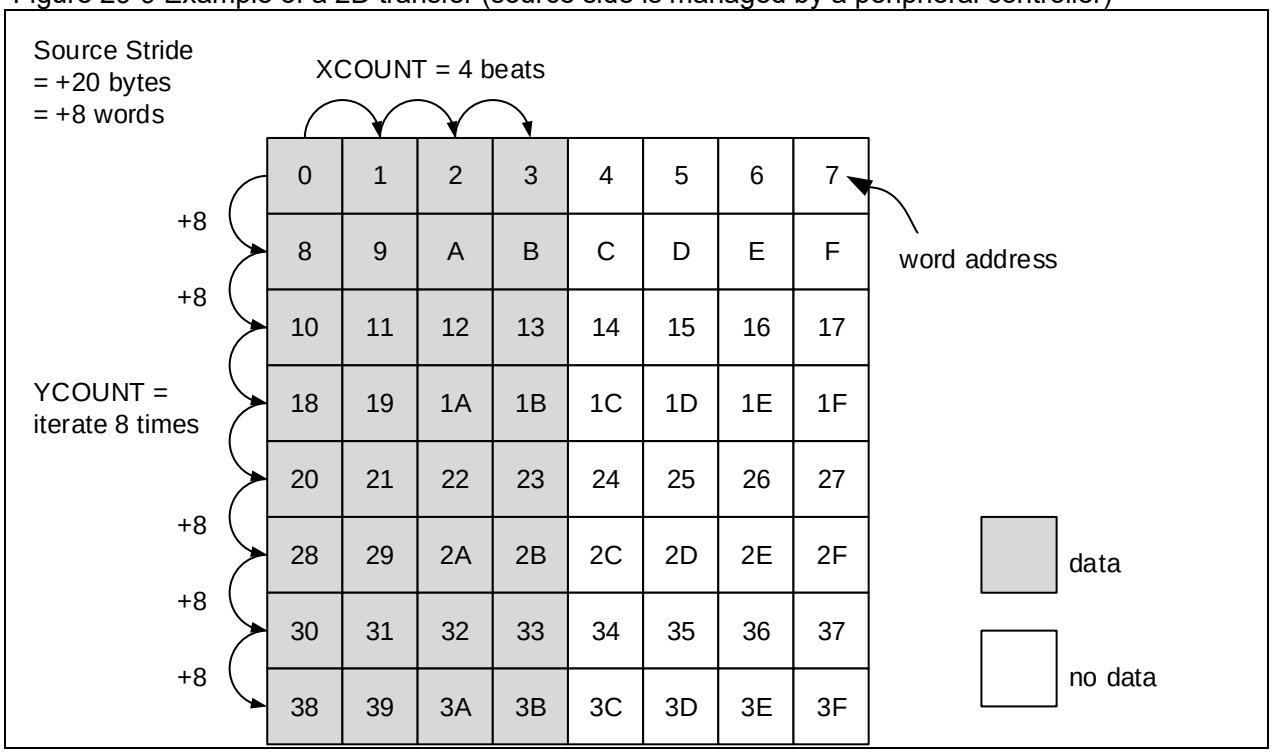
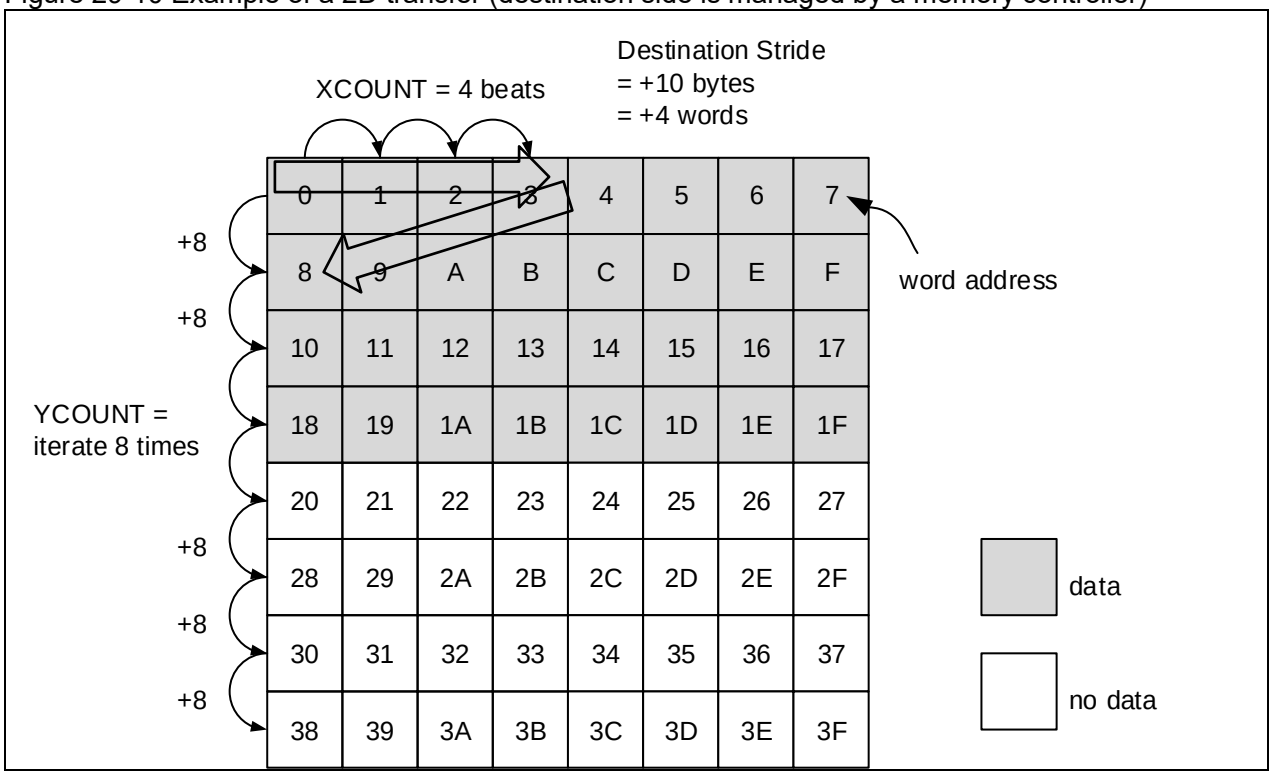


Figure 29-10 shows how to load data in source memory to destination memory. The only difference between them is the stride value. After the DMA writes 4-word data from 0x0 to 0x3, the next write address is the destination address plus the destination stride (0x0 + 0x10= 0x10 byte address = 0x4 word address). Then the DMA controller can continue a second iteration to write four beats in the word address ranging from 0x4 to 0x7 (byte address from 0x10 to 0x1F). When both the YCOTNT and XCOUTNT become zero, it indicates that all 32-word data (8 iterations*4 beats) have been written to the destination memory.

Figure 29-10 Example of a 2D transfer (destination side is managed by a memory controller)



Limits o 2D transfers

- Support M2M transfer only
- PWIDTH and MISZE must be set to 2 (word)
- Source and destination stride values must be word-aligned
- Source and destination stride values must be two's complement
- PINCM and MINCM bits must be set to 1 (incremented mode)
- PFCTRL bit must be set to 0
- PINCOS bit must be set to 0
- Circular and dual buffer mode are not supported
- Linked table transfer is not supported

29.3.9 Errors

Table 29-1 DMA error events

Error events	
Transfer error	AHB response error during DMA read/write
	Write access to the current destination memory address register in dual buffer mode
FIFO error	FIFO overrun in P2M transfer or FIFO underrun in M2P transfer
	Stream enabled when FIFO threshold bit does not match memory burst size bit
	The total transfer count is not a multiple of that of PBURST bit in peripheral flow control mode.
Direct mode error	In the case of FEN = 1 and MINCM = 0, a DMA request is generated before the previous data have been completely transmitted to memory

29.3.10 Interrupts

An interrupt can be generated on a DMA half-transfer, transfer complete, transfer error, FIFO error and direct mode error. Each channel has its specific interrupt flag, clear and enable bits, as shown in the table below.

Table 29-2 DMA interrupts

Interrupt events	Event flag	Clear control bit	Enable control bit
Half-transfer	FDTF	FDTFC	FDTIEN
Transfer complete	HDTF	HDTFC	HDTIEN
Transfer error	DTERRF	DTERRFC	DTERRIEN
FIFO error	FERRF	FERRFC	FERRIEN
Direct mode error	DMERRF	DMERRFC	DMERRIEN

29.4 DMA multiplexer (DMAMUX)

DMAMUX manages DMA requests/acknowledge between peripherals and DMA controller.

The DMA controller selects the DMA mapping table with the TBL_SEL bit in the DMA_MUXSEL register. Each DMA controller stream selects only one DMA request from the flexible mapping table. In flexible mapping mode, each channel can bypass or synchronize 127 possible channel requests from peripherals or generators through the REQSEL [6: 0] bit in the DMA_MUXCxCTRL register.

EXINT LINE is used as the trigger input for request generators and the synchronized input for requests.

29.4.1 DMAMUX functional overview

The DMAMUX consists of a request generator and a request multiplexer.

Each of the DMAMUX generator channel x has a GEN enable bit in the DMA_MUXGxCTR register. The SIGSEL bit is used to select the trigger input of the DMAMUX generator. Typically, the number of DMA requests equals GREQCNT + 1. The GPOL bit is used in the DMA_MUXGxCTRL register to select a trigger event that can be on a rising edge, falling edge or either of them.

Each of the DMAMUX stream x comes from all_req [127: 1].

In flexible mapping mode, the SYNCEN bit in the DMA_MUXSxCTRL register is used to synchronize the selected DMA request input. In synchronous mode, the SYNCSEL bit in the DMA_MUXSxCTRL register is used to select synchronized input. The selected DMA request input will be transferred to chx_mux_req [7: 0] as soon as a valid edge of the synchronized input is detected by the SYNCPOL [1: 0] in the DMA_MUXSxCTRL register. In addition, when the EVTGEN bit is set in the DMA_MUXCxCTRL register, the programmable request counter (REQCNT) is used to generate a request output and event output.

Figure 29-11 DMAMUX block diagram

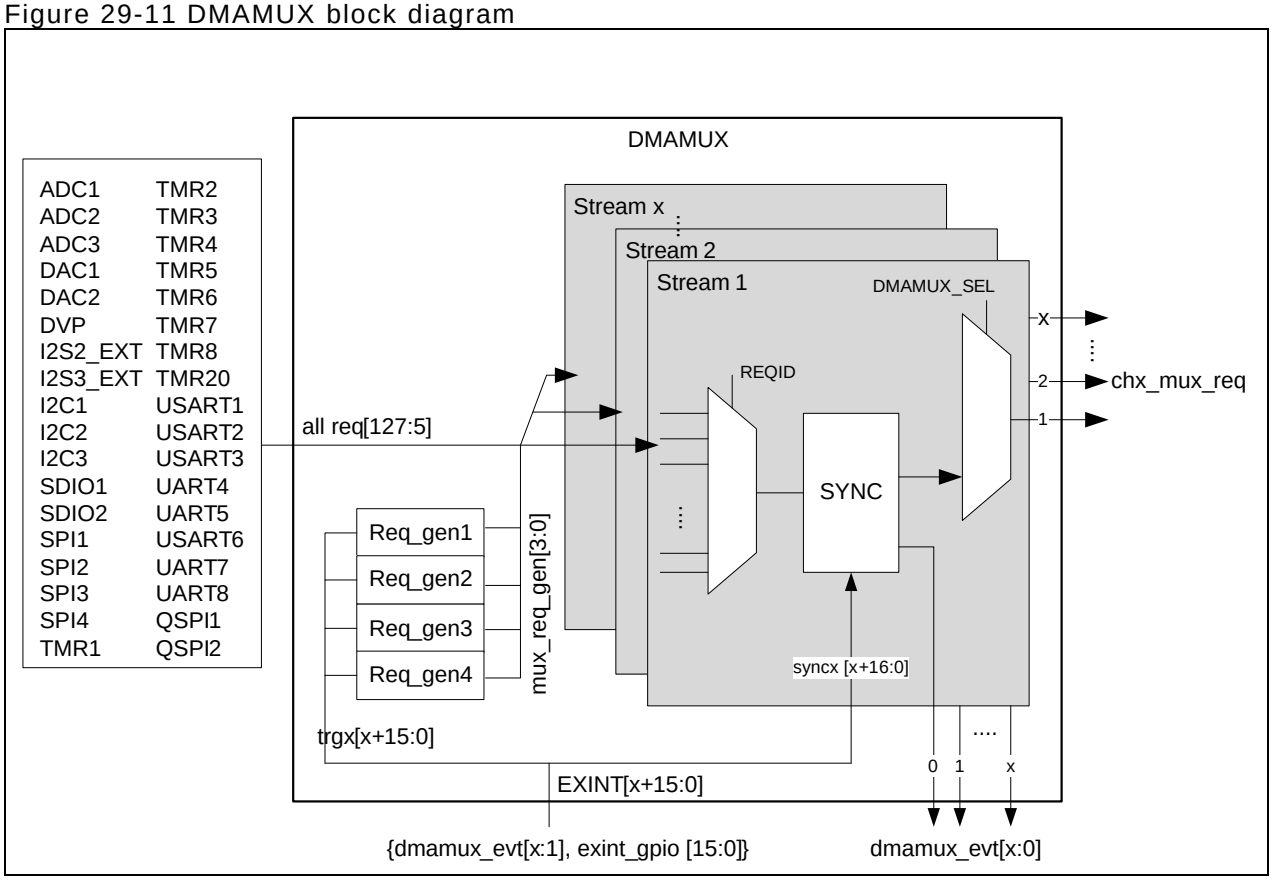


Table 29-3 EDMA flexible request mapping

DMAMUX request	Source	DMAMUX request	Source	DMAMUX request	Source	DMAMUX request	Source
1	DMA_MUXREQG1	33	UART5_TX	65	TMR3_OVERFLOW	97	reserved
2	DMA_MUXREQG2	34	reserved	66	TMR3_TRIG	98	reserved
3	DMA_MUXREQG3	35	reserved	67	TMR4_CH1	99	reserved
4	DMA_MUXREQG4	36	ADC2	68	TMR4_CH2	100	reserved
5	ADC1	37	ADC3	69	TMR4_CH3	101	reserved
6	DAC1	38	reserved	70	TMR4_CH4	102	reserved
7	reserved	39	SDIO1	71	TMR4_OVERFLOW	103	SDIO2
8	TMR6_OVERFLOW	40	QSPI1	72	TMR5_CH1	104	QSPI2
9	TMR7_OVERFLOW	41	DAC2	73	TMR5_CH2	105	DVP
10	SPI1_RX	42	TMR1_CH1	74	TMR5_CH3	106	SPI4_RX
11	SPI1_TX	43	TMR1_CH2	75	TMR5_CH4	107	SPI4_TX
12	SPI2_RX	44	TMR1_CH3	76	TMR5_OVERFLOW	108	reserved
13	SPI2_TX	45	TMR1_CH4	77	TMR5_TRIG	109	reserved
14	SPI3_RX	46	TMR1_OVERFLOW	78	reserved	110	I2S2_EXT_RX
15	SPI3_TX	47	TMR1_TRIG	79	reserved	111	I2S2_EXT_TX
16	I2C1_RX	48	TMR1_HALL	80	reserved	112	I2S3_EXT_RX
17	I2C1_TX	49	TMR8_CH1	81	reserved	113	I2S3_EXT_TX
18	I2C2_RX	50	TMR8_CH2	82	reserved	114	USART6_RX
19	I2C2_TX	51	TMR8_CH3	83	reserved	115	USART6_TX
20	I2C3_RX	52	TMR8_CH4	84	reserved	116	UART7_RX

21	I2C3_TX	53	TMR8_OVERFLOW	85	reserved	117	UART7_TX
22	reserved	54	TMR8_TRIG	86	TMR20_CH1	118	UART8_RX
23	reserved	55	TMR8_HALL	87	TMR20_CH2	119	UART8_TX
24	USART1_RX	56	TMR2_CH1	88	TMR20_CH3	120	reserved
25	USART1_TX	57	TMR2_CH2	89	TMR20_CH4	121	reserved
26	USART2_RX	58	TMR2_CH3	90	TMR20_OVERFLOW	122	reserved
27	USART2_TX	59	TMR2_CH4	91	reserved	123	reserved
28	USART3_RX	60	TMR2_OVERFLOW	92	reserved	124	reserved
29	USART3_TX	61	TMR3_CH1	93	TMR20_TRIG	125	reserved
30	UART4_RX	62	TMR3_CH2	94	TMR20_HALL	126	TMR2_TRIG
31	UART4_TX	63	TMR3_CH3	95	reserved	127	reserved
32	UART5_RX	64	TMR3_CH4	96	reserved		

Table 29-4 DMAMUX EXINT LINE for trigger input and synchronized input

EXINT LINE	Source	EXINT LINE	Source	EXINT LINE	Source	EXINT LINE	Source
0	exint_gpio[0]	8	exint_gpio[8]	16	DMA_MUXevt1	24	reserved
1	exint_gpio[1]	9	exint_gpio[9]	17	DMA_MUXevt2	25	reserved
2	exint_gpio[2]	10	exint_gpio[10]	18	DMA_MUXevt3	26	reserved
3	exint_gpio[3]	11	exint_gpio[11]	19	DMA_MUXevt4	27	reserved
4	exint_gpio[4]	12	exint_gpio[12]	20	DMA_MUXevt5	28	reserved
5	exint_gpio[5]	13	exint_gpio[13]	21	DMA_MUXevt6	29	reserved
6	exint_gpio[6]	14	exint_gpio[14]	22	DMA_MUXevt7	30	reserved
7	exint_gpio[7]	15	exint_gpio[15]	23	DMA_MUXevt8	31	reserved

29.4.2 DMAMUX overflow interrupts

During DMAMUX request generation, when a new trigger input occurs before the GREQCNT underflows, the TRGOVFx bit will be set in the DMA_MUXGSTS register. It is cleared by setting TRGOVFCx=1 in the DMA_MUXGCLR register. An interrupt will be generated if the interrupt enable bit TRGOVIEN is set in the DMA_MUXGxCTRL register.

In DMAMUX synchronous mode, when a new synchronized input occurs before the REQCNT underflows, the SYNCOVFx bit will be set in the DMA_MUXSYNCSTS register. It is cleared by setting the SYNCOVFCx bit in the DMA_MUXSYNCCLR register. An interrupt will be generated if the interrupt enable bit SYNCOVIEN is set in the DMA_MUXSxCTRL register.

Figure 29-12 DMAMUX synchronized mode

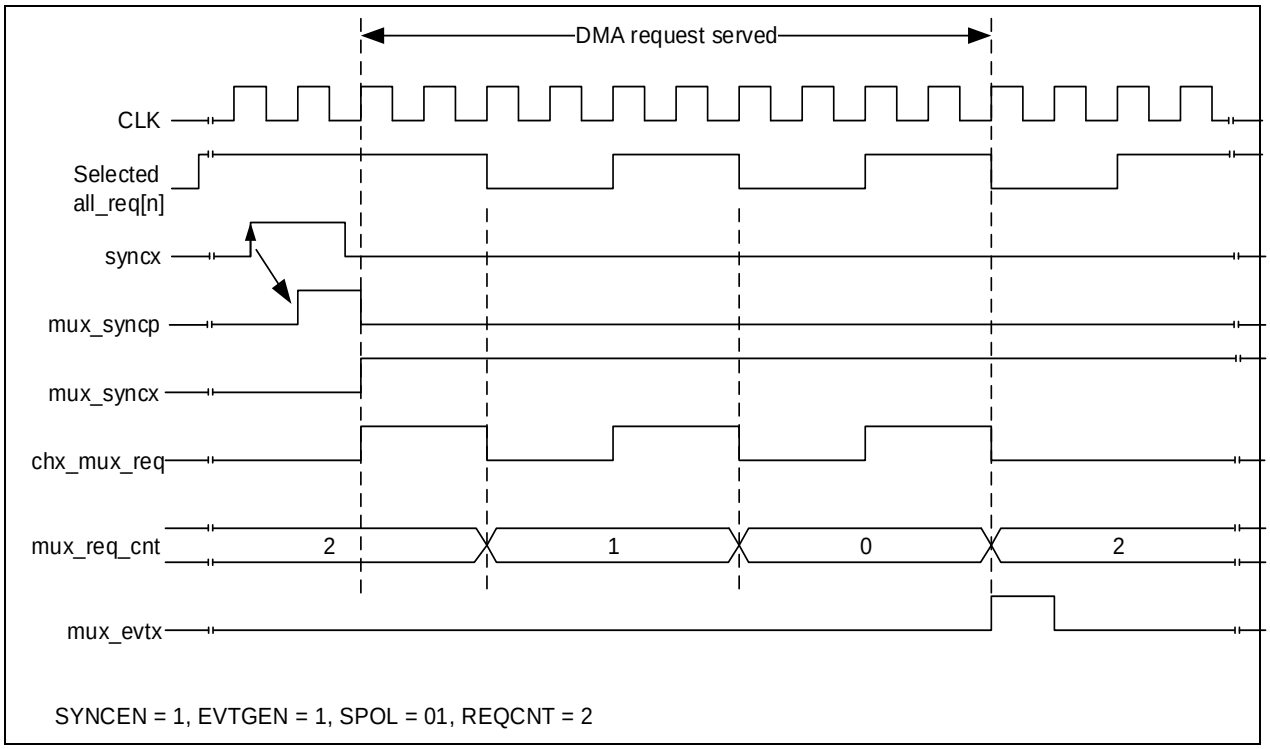
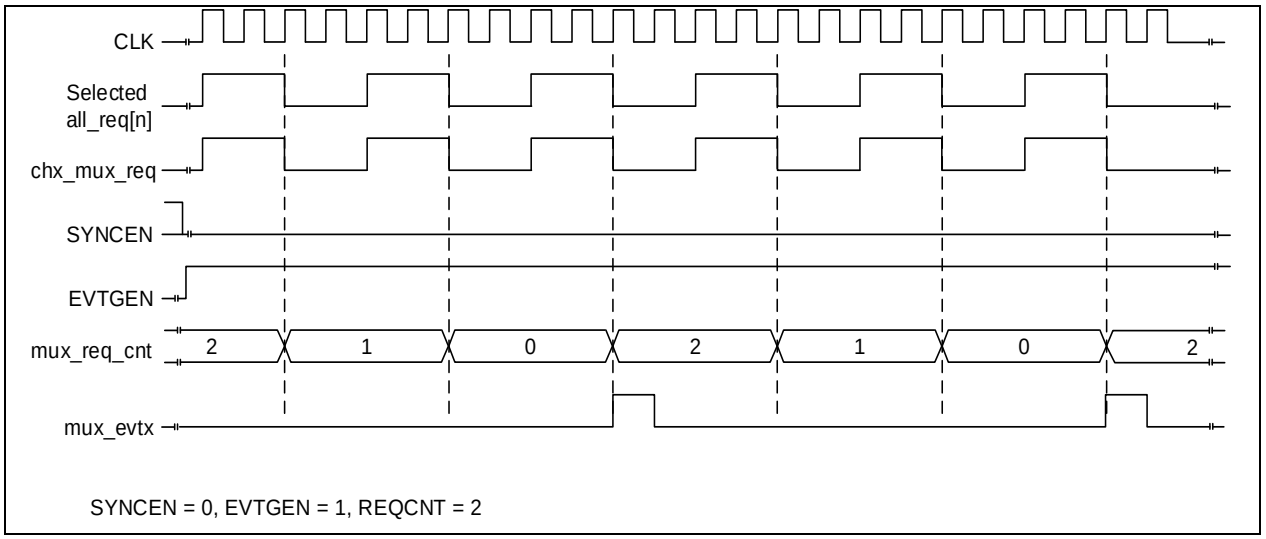


Figure 29-13 DMAMUX event generation



29.5 EDMA registers

Table 29-5 shows DMA register map and reset values.

These peripheral registers must be accessed by byte (8 bits), half-word (16 bits) or word (32 bits)

Table 29-5 BPR register map and reset values

Register	Offset	Reset value
EDMA_STS1	0x00	0x0000 0000
EDMA_STS2	0x04	0x0000 0000
EDMA_CLR1	0x08	0x0000 0000
EDMA_CLR2	0x0c	0x0000 0000
EDMA_S1CTRL	0x10	0x0000 0000
EDMA_S1DTCNT	0x14	0x0000 0000
EDMA_S1PADDR	0x18	0x0000 0000
EDMA_S1M0ADDR	0x1c	0x0000 0000
EDMA_S1M1ADDR	0x20	0x0000 0000
EDMA_S1FCTRL	0x24	0x0000 0000
EDMA_S2CTRL	0x28	0x0000 0000
EDMA_S2DTCNT	0x2c	0x0000 0000
EDMA_S2PADDR	0x30	0x0000 0000
EDMA_S2M0ADDR	0x34	0x0000 0000
EDMA_S2M1ADDR	0x38	0x0000 0000
EDMA_S2FCTRL	0x3c	0x0000 0000
EDMA_S3CTRL	0x40	0x0000 0000
EDMA_S3DTCNT	0x44	0x0000 0000
EDMA_S3PADDR	0x48	0x0000 0000
EDMA_S3M0ADDR	0x4c	0x0000 0000
EDMA_S3M1ADDR	0x50	0x0000 0000
EDMA_S3FCTRL	0x54	0x0000 0000
EDMA_S4CTRL	0x58	0x0000 0000
EDMA_S4DTCNT	0x5c	0x0000 0000
EDMA_S4PADDR	0x60	0x0000 0000
EDMA_S4M0ADDR	0x64	0x0000 0000
EDMA_S4M1ADDR	0x68	0x0000 0000
EDMA_S4FCTRL	0x6c	0x0000 0000
EDMA_S5CTRL	0x70	0x0000 0000
EDMA_S5DTCNT	0x74	0x0000 0000
EDMA_S5PADDR	0x78	0x0000 0000
EDMA_S5M0ADDR	0x7c	0x0000 0000
EDMA_S5M1ADDR	0x80	0x0000 0000
EDMA_S5FCTRL	0x84	0x0000 0000
EDMA_S6CTRL	0x88	0x0000 0000
EDMA_S6DTCNT	0x8c	0x0000 0000

EDMA_S6PADDR	0x90	0x0000 0000
EDMA_S6M0ADDR	0x94	0x0000 0000
EDMA_S6M1ADDR	0x98	0x0000 0000
EDMA_S6FCTRL	0x9c	0x0000 0000
EDMA_S7CTRL	0xa0	0x0000 0000
EDMA_S7DTCNT	0xa4	0x0000 0000
EDMA_S7PADDR	0xa8	0x0000 0000
EDMA_S7M0ADDR	0xac	0x0000 0000
EDMA_S7M1ADDR	0xb0	0x0000 0000
EDMA_S7FCTRL	0xb4	0x0000 0000
EDMA_S8CTRL	0xb8	0x0000 0000
EDMA_S8DTCNT	0xbc	0x0000 0000
EDMA_S8PADDR	0xc0	0x0000 0000
EDMA_S8M0ADDR	0xc4	0x0000 0000
EDMA_S8M1ADDR	0xc8	0x0000 0000
EDMA_S8FCTRL	0xcc	0x0000 0000
EDMA_LLCTRL	0xd0	0x0000 0000
EDMA_S1LLP	0xd4	0x0000 0000
EDMA_S2LLP	0xd8	0x0000 0000
EDMA_S3LLP	0xdc	0x0000 0000
EDMA_S4LLP	0xe0	0x0000 0000
EDMA_S5LLP	0xe4	0x0000 0000
EDMA_S6LLP	0xe8	0x0000 0000
EDMA_S7LLP	0xec	0x0000 0000
EDMA_S8LLP	0xf0	0x0000 0000
EDMA_S2DCTRL	0xf4	0x0000 0000
EDMA_S12DCNT	0xf8	0x0000 0000
EDMA_S1STRIDE	0xfc	0x0000 0000
EDMA_S22DCNT	0x100	0x0000 0000
EDMA_S2STRIDE	0x104	0x0000 0000
EDMA_S32DCNT	0x108	0x0000 0000
EDMA_S3STRIDE	0x10c	0x0000 0000
EDMA_S42DCNT	0x110	0x0000 0000
EDMA_S4STRIDE	0x114	0x0000 0000
EDMA_S52DCNT	0x118	0x0000 0000
EDMA_S5STRIDE	0x11c	0x0000 0000
EDMA_S62DCNT	0x120	0x0000 0000
EDMA_S6STRIDE	0x124	0x0000 0000
EDMA_S72DCNT	0x128	0x0000 0000
EDMA_S7STRIDE	0x12c	0x0000 0000
EDMA_S82DCNT	0x130	0x0000 0000

EDMA_S8STRIDE	0x134	0x0000 0000
EDMA_SYNCEN	0x138	0x0000 0000
EDMA_MUXSEL	0x13c	0x0000 0000
EDMA_MUXS1CTRL	0x140	0x0000 0000
EDMA_MUXS2TRL	0x144	0x0000 0000
EDMA_MUXS3CTRL	0x148	0x0000 0000
EDMA_MUXS4CTRL	0x14c	0x0000 0000
EDMA_MUXS5CTRL	0x150	0x0000 0000
EDMA_MUXS6CTRL	0x154	0x0000 0000
EDMA_MUXS7CTRL	0x158	0x0000 0000
EDMA_MUXS8CTRL	0x15c	0x0000 0000
EDMA_MUXG1CTRL	0x160	0x0000 0000
EDMA_MUXG2CTRL	0x164	0x0000 0000
EDMA_MUXG3CTRL	0x168	0x0000 0000
EDMA_MUXG4CTRL	0x16c	0x0000 0000
EDMA_MUXSYNCSTS	0x170	0x0000 0000
EDMA_MUXSYNCCLR	0x174	0x0000 0000
EDMA_MUXGSTS	0x178	0x0000 0000
EDMA_MUXGCLR	0x17c	0x0000 0000

29.5.1 DMA status register 1 (DMA_STS1)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0	resd	Kept at its default value.
Bit 27	FDTF4	0x0	ro	Stream4 full data transfer complete interrupt flag
Bit 26	HDTF4	0x0	ro	Stream4 half data transfer complete interrupt flag
Bit 25	DTERRF4	0x0	ro	Stream4 transfer error interrupt flag
Bit 24	DMERRF4	0x0	ro	Stream4 direct mode error interrupt flag
Bit 23	Reserved	0x0	ro	Kept at its default value.
Bit 22	FERRF4	0x0	ro	Stream4 fifo error interrupt flag
Bit 21	FDTF3	0x0	ro	Stream3 full data transfer complete interrupt flag
Bit 20	HDTF3	0x0	ro	Stream3 half data transfer complete interrupt flag
Bit 19	DTERRF3	0x0	ro	Stream3 transfer error interrupt flag
Bit 18	DMERRF3	0x0	ro	Stream3 direct mode error interrupt flag
Bit 17	Reserved	0x0	ro	Kept at its default value.
Bit 16	FERRF3	0x0	ro	Stream3 fifo error interrupt flag
Bit 15: 12	Reserved	0x0	resd	Kept at its default value.

Bit 11	FDTF2	0x0	ro	Stream2 full data transfer complete interrupt flag
Bit 10	HDTF2	0x0	ro	Stream2 half transfer complete interrupt flag
Bit 9	DTERRF2	0x0	ro	Stream2 transfer error interrupt flag
Bit 8	DMERRF2	0x0	ro	Stream2 direct mode error interrupt flag
Bit 7	Reserved	0x0	ro	Kept at its default value.
Bit 6	FERRF2	0x0	ro	Stream2 fifo error interrupt flag
Bit 5	FDTF1	0x0	ro	Stream1 full data transfer complete interrupt flag
Bit 4	HDTF1	0x0	ro	Stream1 half data transfer complete interrupt flag
Bit 3	DTERRF1	0x0	ro	Stream1 transfer error interrupt flag
Bit 2	DMERRF1	0x0	ro	Stream1 direct mode error interrupt flag
Bit 1	Reserved	0x0	ro	Kept at its default value.
Bit 0	FERRF1	0x0	ro	Stream1 fifo error interrupt flag

29.5.2 DMA status register 2 (DMA_STS2)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0	resd	Kept at its default value.
Bit 27	FDTF8	0x0	ro	Stream6 full data transfer complete interrupt flag
Bit 26	HDTF8	0x0	ro	Stream8 half transfer complete interrupt flag
Bit 25	DTERRF8	0x0	ro	Stream8 transfer error interrupt flag
Bit 24	DMERRF8	0x0	ro	Stream8 direct mode error interrupt flag
Bit 23	Reserved	0x0	ro	Kept at its default value.
Bit 22	FERRF8	0x0	ro	Stream8 fifo error interrupt flag
Bit 21	FDTF7	0x0	ro	Stream7 full data transfer complete interrupt flag
Bit 20	HDTF7	0x0	ro	Stream7 half data transfer complete interrupt flag
Bit 19	DTERRF7	0x0	ro	Stream7 transfer error interrupt flag
Bit 18	DMERRF7	0x0	ro	Stream7 direct mode error interrupt flag
Bit 17	Reserved	0x0	ro	Kept at its default value.
Bit 16	FERRF7	0x0	ro	Stream7 fifo error interrupt flag
Bit 15: 12	Reserved	0x0	resd	Kept at its default value.
Bit 11	FDTF6	0x0	ro	Stream6 full data transfer complete interrupt flag

Bit 10	HDTF6	0x0	ro	Stream6 half data transfer complete interrupt flag
Bit 9	DTERRF6	0x0	ro	Stream6 transfer error interrupt flag
Bit 8	DMERRF6	0x0	ro	Stream6 direct mode error interrupt flag
Bit 7	Reserved	0x0	ro	Kept at its default value.
Bit 6	FERRF6	0x0	ro	Stream6 fifo error interrupt flag
Bit 5	FDTF5	0x0	ro	Stream5 full data transfer complete interrupt flag
Bit 4	HDTF5	0x0	ro	Stream5 half data transfer complete interrupt flag
Bit 3	DTERRF5	0x0	ro	Stream5 transfer error interrupt flag
Bit 2	DMERRF5	0x0	ro	Stream5 direct mode error interrupt flag
Bit 1	Reserved	0x0	ro	Kept at its default value.
Bit 0	FERRF5	0x0	ro	Stream5 fifo error interrupt flag

29.5.3 DMA flag clear register 1 (DMA_CLR1)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0	resd	Kept at its default value.
Bit 27	FDTFC4	0x0	w	Stream4 clear transfer complete interrupt flag
Bit 26	HDTFC4	0x0	w	Stream4 clear half transfer complete interrupt flag
Bit 25	DTERRFC4	0x0	w	Stream4 clear error interrupt flag
Bit 24	DMERRFC4	0x0	w	Steam4 clear direct mode error interrupt flag
Bit 23	Reserved	0x0	resd	Kept at its default value.
Bit 22	FERRFC4	0x0	w	Stream4 clear fifo error interrupt flag
Bit 21	FDTFC3	0x0	w	Stream3 clear transfer complete interrupt flag
Bit 20	HDTFC3	0x0	w	Stream3 clear half transfer complete interrupt flag
Bit 19	DTERRFC3	0x0	w	Stream3 clear error interrupt flag
Bit 18	DMERRFC3	0x0	w	Steam3 clear direct mode error interrupt flag
Bit 17	Reserved	0x0	resd	Kept at its default value.
Bit 16	FERRFC3	0x0	w	Stream3 clear fifo error interrupt flag
Bit 15: 12	FDTFC2	0x0	w	Stream2 clear transfer complete interrupt flag
Bit 11	HDTFC2	0x0	w	Stream2 clear half transfer complete interrupt flag
Bit 10	DTERRFC2	0x0	w	Stream2 clear error interrupt flag
Bit 9	DMERRFC2	0x0	w	Steam2 clear direct mode error interrupt flag
Bit 8	Reserved	0x0	resd	Kept at its default value.

Bit 7	FERRFC2	0x0	w	Stream2 clear fifo error interrupt flag
Bit 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	FDTFC1	0x0	w	Stream1 clear transfer complete interrupt flag
Bit 4	HDTFC1	0x0	w	Stream1 clear half transfer complete interrupt flag
Bit 3	DTERRFC1	0x0	w	Stream1 clear error interrupt flag
Bit 2	DMERRFC1	0x0	w	Steam1 clear direct mode error interrupt flag
Bit 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	FERRFC1	0x0	w	Stream1 clear fifo error interrupt flag

29.5.4 DMA flag clear register 2 (DMA_CLR2)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0	resd	Kept at its default value.
Bit 27	FDTFC8	0x0	w	Stream8 clear transfer complete interrupt flag
Bit 26	HDTFC8	0x0	w	Stream8 clear half transfer complete interrupt flag
Bit 25	DTERRFC8	0x0	w	Stream8 clear error interrupt flag
Bit 24	DMERRFC8	0x0	w	Steam8 clear direct mode error interrupt flag
Bit 23	Reserved	0x0	resd	Kept at its default value.
Bit 22	FERRFC8	0x0	w	Stream8 clear fifo error interrupt flag
Bit 21	FDTFC7	0x0	w	Stream7 clear transfer complete interrupt flag
Bit 20	HDTFC7	0x0	w	Stream7 clear half transfer complete interrupt flag
Bit 19	DTERRFC7	0x0	w	Stream7 clear error interrupt flag
Bit 18	DMERRFC7	0x0	w	Steam7 clear direct mode error interrupt flag
Bit 17	Reserved	0x0	resd	Kept at its default value.
Bit 16	FERRFC7	0x0	w	Stream7 clear fifo error interrupt flag
Bit 15: 12	FDTFC6	0x0	w	Stream6 clear transfer complete interrupt flag
Bit 11	HDTFC6	0x0	w	Stream6 clear half transfer complete interrupt flag
Bit 10	DTERRFC6	0x0	w	Stream6 clear error interrupt flag
Bit 9	DMERRFC6	0x0	w	Steam6 clear direct mode error interrupt flag
Bit 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	FERRFC6	0x0	w	Stream6 clear fifo error interrupt flag
Bit 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	FDTFC5	0x0	w	Stream5 clear transfer complete interrupt flag

Bit 4	HDTFC5	0x0	w	Stream5 clear half transfer complete interrupt flag
Bit 3	DTERRFC5	0x0	w	Stream5 clear error interrupt flag
Bit 2	DMERRFC5	0x0	w	Steam5 clear direct mode error interrupt flag
Bit 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	FERRFC5	0x0	w	Stream5 clear fifo error interrupt flag

29.5.5 DMA stream-x control register (DMA_SxCTRL) (x= 1…8)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0	resd	Kept at its default value.
Bit 27: 25	CHSEL	0x0	rw	channel select 000: Channel 1 selected This field can be written only when SEN=0.
Bit 24: 23	MBURST	0x0	rw	Memory burst transfer configuration This field defines the type of memory burst transfer. 00: Single transfer 01: INCR4 (4-beat increment burst) 10: INCR8 (beat increment burst) 11: INCR16 (16-beat increment burst) This field is forced to 00 when SEN=1. Note: This field can be written only when SEN=0.
Bit 22: 21	PBURST	0x0	rw	Peripheral burst transfer configuration This field defines the type of peripheral transfers. 00: Single transfer 01: INCR4 (4-beat increment burst) 10: INCR8 (beat increment burst) 11: INCR16 (16-beat increment burst) This field is forced to 00 when SEN=1. Note: This field can be written only when SEN=0.
Bit 20	Reserved	0x0	resd	Kept at its default value.
Bit 19	CM	0x0	rw	Current memory This bit is valid in dual buffer mode only. 0: Current target is memory 0 (addressed by the DMA_SxM0ADDR) 1: Current target is memory 1 (addressed by the DMA_SxM1ADDR) This bit can be written only when SEN=0 to indicate the target memory area of the first transfer. Once SEN=1, this bit is used as a status flag, indicating which is the current target memory area.
Bit 18	DMM	0x0	rw	Double memory mode 0: Double memory mode disabled 1: Double memory mode enabled This bit can be written only when SEN=0.
Bit 17: 16	SPL	0x0	rw	Stream polarity 00: Low 01: Medium 10: High 11: Very high This field can be written only when SEN=0.

Bit 15	PINCOS	00x	rw	Peripheral increase offset 0: Peripheral increase offset disabled 1: Peripheral increase offset enabled If PINCOS bit is enabled, the offset size of the peripheral address calculation is fixed to 32-bit alignment. This bit has no meaning if PINCM=0. This bit is forced low by hardware when the direction mode is selected or if PBURST is different from 00. This bit can be written only when SEN=0.
Bit 14: 13	MWIDTH	0x0	rw	Memory data width This field defines the HSIZE (AHB bus signal) in the memory controller. 00: Byte (8-bit) 01: Half-word (16-bit) 10: Word (32-bit) 11: Reserved In direct mode, this field is forced by hardware to be same value as PWIDTH when SEN=1. This field can be written only when SEN=0
Bit 12: 11	PWIDTH	00x	rw	peripheral data width This field defines the HSIZE (AHB bus signal) in the peripheral controller. 00: Byte (8-bit) 01: Half-word (16-bit) 10: Word (32-bit) 11: Reserved This field can be written only when SEN=0.
Bit 10	MINCM	0x0	rw	Memory increment mode 0: Memory address pointer is fixed 1: Memory address pointer is incremented This bit can be written only when SEN=0.
Bit 9	PINCM	0x0	rw	peripheral increment mode 0: Peripheral address pointer is fixed 1: Peripheral address pointer is incremented This bit can be written only when SEN=0.
Bit 8	LM	0x0	rw	Loop mode 0: Loop mode is disabled 1: Loop mode is enabled This bit is forced flow by hardware if PFCTRL=1, as soon as SEN=1. This bit is forced high by hardware if DMM=1, as soon as SEN=1.
Bit 7: 6	DTD	0x0	rw	data transfer direction 00: Peripheral to memory (P2M) 01: Memory to peripheral (M2P) 10: Memory to memory (M2M) 11: Reserved This bit can be written only when SEN=0.
Bit 5	PFCTRL	0x0	rw	Peripheral flow controller 0: DMA is the flow controller 1: The peripheral is the flow controller This bit is forced low by hardware if DTD=10 (M2M). This bit can be written only when SEN=0.
Bit 4	FDTIEN	0x0	rw	Full data transfer interrupt enable 0: TC interrupt disabled 1: TC interrupt enabled
Bit 3	HDTIEN	0x0	rw	Half data transfer interrupt enable 0: HT interrupt disabled 1: HT interrupt enabled

Bit 2	DTERRIEN	0x0	rw	Data transfer error interrupt enable 0: TE interrupt disabled 1: TE interrupt enabled
Bit 1	DMERRIEN	0x0	rw	Direct mode error interrupt enable 0: DME interrupt disabled 1: DME interrupt enabled
Bit 0	SEN	0x0	rw	Stream enable 0: Stream disabled 1: Stream enabled This bit may be cleared by hardware under the following conditions: -DMA end of transfer -If a transfer error occurs on the AHB master buses -When the FIFO threshold on memory AHB port is not compatible with the burst size. Note: Before setting SEN=1 to start a new transfer, the event flags corresponding to the stream in DMA_STS1 or DMA_STS2 must be cleared.

29.5.6 DMA stream-x data register (DMA_SxDTCNT) (x= 1...8)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit15: 0	CNT	0x0000	rw	Number of data to be transferred 0~65535 This register can be written only when SEN=0. This register is read-only when SEN=1, indicating the remaining data to be transmitted. This register is decremented after each transfer. Typically, this register remains 0 after the completion of data transfer. But this register may be automatically reloaded with the previously programmed value in the following cases: ·When the stream is configured to be loop mode ·When the stream is enabled again by setting SEN=1. If this register is zero, no transaction can be served even if SEN=1

29.5.7 DMA stream-x peripheral address register (DMA_SxPADDR) (x= 1...8)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	PADDR	0x0000 0000	rw	Peripheral address Base address of the peripheral address. This field can be written only if SEN=0.

29.5.8 DMA stream-x memory 0 address register (DMA_SxM0ADDR) (x= 1...8)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	M0ADDR	0x0000 0000	rw	Memory 0 address Base address of memory area 0. This field can be written only if: ·The stream is disabled (SEN =0) ·The stream is enabled in dual buffer mode, and CM=1.

29.5.9 DMA stream-x memory 1 address register (DMA_SxM1ADDR) (x= 1...8)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	M1ADDR	0x0000 0000	rw	Memory 1 address Base address of memory area 1. This field can be written only if: ·The stream is disabled (SEN =0) ·The stream is enabled in dual buffer mode, and CM=1.

29.5.10 DMA stream-x FIFO control register (DMA_SxFCTRL) (x= 1...8)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0000 00	resd	Kept at its default value.
Bit 7	FERRIEN	0x0	rw	FIFO error interrupt enable 0: FIFO error interrupt disabled 1: FIFO error interrupt enabled
Bit 6	Reserved	0x0	resd	Kept at its default value.
Bit 5: 3	FSTS	0x4	rw	FIFO status 000: 0 <fifo_level <1/4 001: 1/4≤fifo_level <1/2 010: 1/2≤fifo_level <3/4 011: 3/4≤fifo_level < full 100: FIFO empty 101: FIFO full Others: No meaning These bits are not relevant in direct mode (FEN=0)
Bit 2	FEN	0x0	rw	FIFO mode enable 0: Direct mode 1: FIFO mode This bit is set by hardware if memory-to-memory mode is selected (DTD=10) and SEN=1.
Bit 1: 0	FTHSEL	0x1	rw	FIFO threshold select 00: 1/4 full FIFO 01: 1/2 full FIFO 10: 3/4 full FIFO 11: full FIFO These bits are not used in direct mode (FEN=0). These bits cannot be written only if SEN=0.

29.5.11 DMA linked table control register (DMA_SxLLCTRL)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	S8LLEN	0x0	rw	Stream8 link list enable
Bit 6	S7LLEN	0x0	rw	Stream7 link list enable
Bit 5	S6LLEN	0x0	rw	Stream6 link list enable
Bit 4	S5LLEN	0x0	rw	Stream5 link list enable
Bit 3	S4LLEN	0x0	rw	Stream4 link list enable
Bit 2	S3LLEN	0x0	rw	Stream3 link list enable
Bit 1	S2LLEN	0x0	rw	Stream2 link list enable
Bit 0	S1LLEN	0x0	rw	Stream1 link list enable

29.5.12 DMA linked table pointer register (DMA_SxLLP) (x = 1…8)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	LLP	0x0000 0000	rw	link list pointer After the completion of the current descriptor, the flow controller uses this pointer as a target address to read the descriptor.

29.5.13 DMA 2D transfer control register (DMA_S2DCTRL)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	S82DEN	0x0	rw	Stream8 2D enable
Bit 6	S72DEN	0x0	rw	Stream7 2D enable
Bit 5	S62DEN	0x0	rw	Stream6 2D enable
Bit 4	S52DEN	0x0	rw	Stream5 2D enable
Bit 3	S42DEN	0x0	rw	Stream4 2D enable
Bit 2	S32DEN	0x0	rw	Stream3 2D enable
Bit 1	S22DEN	0x0	rw	Stream2 2D enable
Bit 0	S12DEN	0x0	rw	Stream1 2D enable

29.5.14 DMA 2D transfer count register (DMA_S2DCNT)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	YCOUNT	0x0000	rw	C-axis count
Bit 15: 0	XCOUNT	0x0000	rw	X-axis count

29.5.15 DMA 2D transfer stride register (DMA_STRIDE)(x = 1…8)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	DSTSTD	0x0000	rw	Destination stride This is a signed value (two's complement). 0x0000: 0 … 0x7FFF:32767 0x8000:-32768 0x8001:-32767 … 0xFFFF:-1 The DST value is in terms of byte address. For example, if the destination stride is 0x8, the destination byte address is added with 0x8 before the next iteration. DSTSTD bit 1..0 must be 00, for the size of the destination position remains word.
Bit 15: 0	SRCSTD	0x0000	rw	Source stride This is a signed value (two's complement). 0x0000: 0 … 0x7FFF:32767 0x8000:-32768 0x8001:-32767 … 0xFFFF:-1 The SRC value is in terms of byte address. For example, if the source stride is 0x8, the source byte address is added with 0x8 before the next iteration. SRCSTDbit 1..0 must be 00, for the size of the source position remains word.

29.5.16 DMA synchronization enable (DMA_SYNCEN)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0		Kept at its default value.
Bit 7	S8SYNC	0x0		Stream8 synchronization enable
Bit 6	S7SYNC	0x0		Stream7 synchronization enable
Bit 5	S6SYNC	0x0		Stream6 synchronization enable
Bit 4	S5SYNC	0x0		Stream5 synchronization enable
Bit 3	S4SYNC	0x0		Stream4 synchronization enable
Bit 2	S3SYNC	0x0		Stream3 synchronization enable
Bit 1	S2SYNC	0x0		Stream2 synchronization enable
Bit 0	S1SYNC	0x0		Stream1 synchronization enable

29.5.17 DMAMUX table select (DMA_MUXSEL)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	TBL_SEL	0x0	rw	Multiplexer table select 0x1: Flexible mapping table

29.5.18 DMAMUX channel-x control register (DMA_MUXSxCTRL) (x = 1…8)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 25	Reserved	0x0	resd	Kept at its default value.
Bit 28: 24	SYNCSEL	0x0	rw	Synchronization select
Bit 23: 19	REQCNT	0x0	rw	DMA request count This field defines the number of DMA requests after synchronization events and/or before synchronization events. This field are reserved only if SYNCEN and EVTGEN both are 0.
Bit 18: 17	SYNCPOL	0x0	rw	Synchronization polarity Defines the polarity of the selected synchronization input. 0x0: No event 0x1: Rising edge 0x2: Falling edge 0x3: Rising and falling edge
Bit 16	SYNCEN	0x0	rw	Synchronization enable 0: Synchronization disabled 1: Synchronization enabled
Bit 15: 10	Reserved	0x00	resd	Kept at its default value.
Bit 9	EVTGEN	0x0		Event generation enable 0: Event generation disabled 1: Event generation enabled
Bit 8	SYNCOVIEN	0x0		Synchronization overrun interrupt enable 0: Interrupt disabled 1: Interrupt enabled
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6: 0	REQSEL	0x00		DMA request select Select a DMA request Refer to DMAMUX table for more information.

29.5.19 DMAMUX generator-x control register (DMA_MUXGxCTRL) (x = 1...8)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23: 19	GREQCNT	0x00	rw	DMA request generation count Defines the number of DMA requests (GREQCNT + 1) after an event occurs. This field is reserved only when GEN is disabled.
Bit 18: 17	GPOL	0x0	rw	DMA request generation polarity Defines the polarity of the selected trigger inputs. 0x0: No event 0x1: Rising edge 0x2: Falling edge 0x3: Rising and falling edge
Bit 16	GEN	0x0	rw	DMA request generation enable 0: DMA request generation disabled 1: DMA request generation enabled
Bit 15: 9	Reserved	0x00	resd	Kept at its default value.
Bit 8	TRGOVIEN	0x0	rw	Trigger overrun interrupt enable 0: Interrupt disabled 1: Interrupt enabled
Bit 7: 5	Reserved	0x0	resd	Kept at its default value.
Bit 4: 0	SIGSEL	0x00	rw	Signal select This field is used to select DMA signal for DMA request generation.

29.5.20 DMAMUX synchronization interrupt status register (DMA_MUXSYNCSTS)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0000 000	resd	Kept at its default value.
Bit 7: 0	SYNCOVF	0x00	ro	Synchronization overrun interrupt flag This bit is set when a new synchronization event occurs while the DMA request count is below the REQCNT.

29.5.21 DMAMUX synchronization interrupt clear flag register (BPR_MUXSYNCCLR)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0000 000	resd	Kept at its default value.
Bit 7: 0	SYNCOVFC	0x00	rw1c	Synchronization overrun interrupt flag clear Writing 1 to the corresponding bit clears the corresponding overrun flag SYNCOVF in DMA_MUXCSR register.

29.5.22 DMAMUX generator interrupt status register (DMA_MUXGSTS)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0000 000	resd	Kept at its default value.
Bit 7: 0	TRGOVF	0x00	ro	Trigger overrun interrupt flag This bit is set when a new trigger event occurs while the DMA request count is below the GNBREQ.

29.5.23 DMAMUX generator interrupt clear flag register (DMA_MUXGCLR)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0000 000	resd	Kept at its default value.
Bit 7: 0	TRGOVFC	0x00	rw1c	Trigger overrun interrupt flag clear Writing 1 to the corresponding bit clears the overrun flag TRGOVF in DMA_MUXGS register.

30 Debug (DEBUG)

30.1 Debug introduction

Cortex™-M4F core provides powerful debugging features including halt and single step support, as well as trace function that is used for checking the details of the program execution. The debug features are implemented with two interfaces: serial wire debug (SWD) and JTAG debug port. Trace information is collected by a single-wire serial wire view interface, or by TRACE interface when a larger trace bandwidth is needed. Trace and debugging interfaces can be combined into one interface.

ARM Cortex™-M4F reference documentation:

- Cortex™-M4 Technical Reference Manual (TRM)
- ARM Debug Interface V5
- ARM CoreSight Design Kit revision r1p0 Technical Reference Manual

30.2 Debug and Trace

It is possible to support debugging for different peripherals, and configure the status of peripherals during debugging. For timers and watchdogs, the user can select whether or not to stop or continue counting during debugging; For CAN, the user can select whether or not to stop or continue updating receive registers during debugging; For I2C, the user can select whether or not to stop or continue SMBUS timeout counting.

In addition, code debugging is supported in Low-power mode. In Sleep mode, the clock programmed by code remains active for HCLK and FCLK to continue to work. In DeepSleep mode, HICK oscillator is enabled to feed FCLK and HCLK.

There are several ID codes inside the MCU, which is accessible by the debugger using the DEBUG_IDCODE at address 0xE0042000. It is part of the DEBUG and is mapped on the external PPB bus. These codes are accessible using the JTAG debug port or the SWD debug port or by the user software. They are even accessible while the MCU is under system reset.

Two trace interface modes supported: single-pin mode for serial wire view and multi-pin trace interface.

30.3 I/O pin control

SWJ-DP debug is supported in different packages of AT32F435/437. It uses 5 general-purpose I/O ports. After reset, the SWJ-DP can be immediately used by the debugger as a default function. To ensure that JTAG input pins are not floating (particularly SWCLK/JTCK), the JTAG input pins are embedded with internal pull-up or pull-down feature, NJTRST, JTDI and JTMS/SWDIO with internal pull-up feature, and JTCK/SWCLK with internal pull-down feature.

GPIO and IOMUX registers can be configured to allow users to switch between debug ports or disable debug feature.

30.4 DEGUB registers

Table 30-1 shows DEBUG register map and reset values.

These peripheral registers must be accessed by word (32 bits)

Table 30-1 DEBUG register address and reset value

Register name	Offset	Reset value
DEBUG_IDCODE	0xE004 2000	0xXXXX XXXX
DEBUG_CTRL	0xE004 2004	0x0000 0000
DEBUG_APB1_PAUSE	0xE004 2008	0x0000 0000
DEBUG_APB2_PAUSE	0xE004 200C	0x0000 0000
DEBUG_SER_ID	0xE004 2020	0x0000 XX0X

30.4.1 DEBUG device ID (DEBUG_IDCODE)

MCU integrates an ID code that is used to identify MCU's revision code. The DEBUG_IDCODE register is mapped on the external PPB bus at address 0xE0042000. This code is accessible by the JTAG debug port or SW debug port or by the user code.

Bit	Register	Reset value	Type	Description
Bit 31: 0	PID	0xXXXX XXXX	ro	PID information

PID [31: 0]	AT32 part number	FLASH size	Packages
0x7008_4540	AT32F435ZMT7	4032KB	LQFP144
0x7008_3341	AT32F435ZGT7	1024KB	LQFP144
0x7008_4598	AT32F435ZDT7	448KB	LQFP144
0x7008_3242	AT32F435ZCT7	256KB	LQFP144
0x7008_4543	AT32F435VMT7	4032KB	LQFP100
0x7008_3344	AT32F435VGT7	1024KB	LQFP100
0x7008_4599	AT32F435VDT7	448KB	LQFP100
0x7008_3245	AT32F435VCT7	256KB	LQFP100
0x7008_4546	AT32F435RMT7	4032KB	LQFP64
0x7008_3347	AT32F435RGT7	1024KB	LQFP64
0x7008_459A	AT32F435RDT7	448KB	LQFP64
0x7008_3248	AT32F435RCT7	256KB	LQFP64
0x7008_4549	AT32F435CMT7	4032KB	LQFP48
0x7008_334A	AT32F435CGT7	1024KB	LQFP48
0x7008_459B	AT32F435CDT7	448KB	LQFP48
0x7008_324B	AT32F435CCT7	256KB	LQFP48
0x7008_454C	AT32F435CMU7	4032KB	QFN48
0x7008_334D	AT32F435CGU7	1024KB	QFN48
0x7008_459C	AT32F435CDU7	448KB	QFN48
0x7008_324E	AT32F435CCU7	256KB	QFN48
0x7008_454F	AT32F437ZMT7	4032KB	LQFP144
0x7008_3350	AT32F437ZGT7	1024KB	LQFP144
0x7008_459D	AT32F437ZDT7	448KB	LQFP144
0x7008_3251	AT32F437ZCT7	256KB	LQFP144
0x7008_4552	AT32F437VMT7	4032KB	LQFP100
0x7008_3353	AT32F437VGT7	1024KB	LQFP100
0x7008_459E	AT32F437VDT7	448KB	LQFP100
0x7008_3254	AT32F437VCT7	256KB	LQFP100
0x7008_4555	AT32F437RMT7	4032KB	LQFP64
0x7008_3356	AT32F437RGT7	1024KB	LQFP64
0x7008_459F	AT32F437RDT7	448KB	LQFP64
0x7008_3257	AT32F437RCT7	256KB	LQFP64

30.4.2 DEBUG control register (DEBUG_CTRL)

This register is asynchronously reset by POR Reset (not reset by system reset). It can be written by the debugger under reset.

Bit	Register	Reset value	Type	Description
Bit 31:3	Reserved	0x0000 0000	resd	Always 0.
Bit 2	STANDBY_DEBUG	0x0	rw	Debug Standby mode control bit 0: The whole 1.2V digital circuit is unpowered in Standby mode 1: The whole 1.2V digital circuit is not unpowered in Standby mode, and the system clock is provided by the internal RC oscillator (HICK)
Bit 1	DEEPSLEEP_DEBUG	0x0	rw	Debug Deepsleep mode control bit 0: In Deepsleep mode, all clocks in the 1.2V domain are disabled. When exiting from Deepsleep mode, the internal RC oscillator (HICK) is enabled, and HICK is used as the system clock source, and the software must reprogram the system clock according to application requirements. 1: In Deepsleep mode, system clock is provided by the internal RC oscillator (HICK). When exiting from Deepsleep mode, HICK is used as the system clock source, and the software must reprogram the system clock. According to application requirements.
Bit 0	SLEEP_DEBUG	0x0	rw	Debug Sleep mode control bit 0: When entering Sleep mode, CPU HCLK clock is disabled, but other clocks remain active. When exiting from Sleep mode, it is not necessary to reprogram the clock system. 1: When entering Sleep mode, all clocks keep running.

30.4.3 DEBUG APB1 pause register (DEBUG_ APB1_PAUSE)

This register is asynchronously reset by POR Reset (not reset by system reset). It can be written by the debugger under reset.

Bit	Register	Reset value	Type	Description
Bit 31: 29	Reserved	0x0	resd	Kept at its default value.
Bit 28	I2C3_SMBUS_TIMEOUT	0x0	rw	I2C3 pause control bit 0: I2C3 SMBUS timeout control works normally 1: I2C3 SMBUS timeout control stops running
Bit 27	I2C2_SMBUS_TIMEOUT	0x0	rw	I2C2 pause control bit 0: I2C2 SMBUS timeout control works normally 1: I2C2 SMBUS timeout control stops running
Bit 26	CAN2_PAUSE	0x0	rw	CAN2 pause control bit 0: CAN2 works normally 1: CAN2 receive register pauses (does not receive data)
Bit 25	CAN1_PAUSE	0x0	rw	CAN1 pause control bit 0: CAN1 works normally 1: CAN1 receive register pauses (does not receive data)

Bit 24	I2C1_SMBUS_TIMEOUT	0x0	rw	I2C1 pause control bit 0: I2C1 SMBUS timeout control works normally 1: I2C1 SMBUS timeout control stops running
Bit 23: 16	Reserved	0x00	resd	Kept at its default value.
Bit 15	ERTC_512_PAUSE	0x0	rw	ERTC 512Hz output clock pause control bit 0: ERTC 512Hz output clock works normally 1: Froze 512Hz output clock
Bit 14: 13	Reserved	0x0	rw	Kept at its default value.
Bit 12	WDT_PAUSE	0x0	rw	WDT pause control bit 0: WDT works normally 1: WDT stops running
Bit 11	WWDT_PAUSE	0x0	rw	WWDT pause control bit 0: WWDT works normally 1: WWDT stops running
Bit 10	ERTC_PAUSE	0x0	rw	ERTC pause control bit 0: ERTC works normally 1: ERTC stops running
Bit 9	Reserved	0x0	rw	Kept at its default value.
Bit 8	TMR14_PAUSE	0x0	rw	TMR14 pause control bit 0: TMR14 works normally 1: TMR14 stops running
Bit 7	TMR13_PAUSE	0x0	rw	TMR13 pause control bit 0: TMR13 works normally 1: TMR13 stops running
Bit 6	TMR12_PAUSE	0x0	rw	TMR12 pause control bit 0: TMR12 works normally 1: TMR12 stops running
Bit 5	TMR7_PAUSE	0x0	rw	TMR7 pause control bit 0: TMR7 works normally 1: TMR7 stops running
Bit 4	TMR6_PAUSE	0x0	rw	TMR6 pause control bit 0: TMR6 works normally 1: TMR6 stops running
Bit 3	TMR5_PAUSE	0x0	rw	TMR5 pause control bit 0: TMR5 works normally 1: TMR5 stops running
Bit 2	TMR4_PAUSE	0x0	rw	TMR4 pause control bit 0: TMR4 works normally 1: TMR4 stops running
Bit 1	TMR3_PAUSE	0x0	rw	TMR3 pause control bit 0: TMR3 works normally 1: TMR3 stops running
Bit 0	TMR2_PAUSE	0x0	rw	TMR2 pause control bit 0: TMR2 works normally 1: TMR2 stops running

30.4.4 DEBUG APB2 pause register (DEBUG_ APB2_PAUSE)

This register is asynchronously reset by POR Reset (not reset by system reset). It can be written by the debugger under reset.

Bit	Register	Reset value	Type	Description
Bit 31: 19	Reserved	0x0000	resd	Kept at its default value.
Bit 18	TMR11_PAUSE	0x0	rw	TMR11 pause control bit 0: TMR11 works normally 1: TMR11 stops running
Bit 17	TMR10_PAUSE	0x0	rw	TMR10 pause control bit 0: TMR10 works normally 1: TMR10 stops running
Bit 16	TMR9_PAUSE	0x0	rw	TMR9 pause control bit 0: TMR9 works normally 1: TMR9 stops running
Bit 15: 7	Reserved	0x000	resd	Kept at its default value.
Bit 6	TMR20_PAUSE	0x0	rw	TMR20 pause control bit 0: TMR20 works normally 1: TMR20 stops running
Bit 5: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	TMR8_PAUSE	0x0	rw	TMR8 pause control bit 0: TMR8 works normally 1: TMR8 stops running
Bit 0	TMR1_PAUSE	0x0	rw	TMR1 pause control bit 0: TMR2 works normally 1: TMR2 stops running

30.4.5 MCU SERIES ID register (DEBUG_ SER_ID)

MCU_SIE_ID register is used to identify MCU part number and its revision code. The DEBUG_IDCODE register is mapped on the external PPB bus. This register is asynchronously reset by POR Reset (not reset by system reset). This code is accessible by the JTAG debug port or SW debug port or by the user code.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 8	SER_ID	0xXX	ro	MCU part number ID AT32F435: 0x0D AT32F437: 0x0E
Bit 7: 3	Reserved	0x0X	resd	Kept at its default value.
Bit 2: 0	REV_ID	0xX	ro	Revision code 0x0: Revision A

31 Revision history

Document Revision History

Date	Version	Revision Note
2021.06.30	2.00	Initial release.
2021.12.01	2.01	Updated I2C description, and revised some typos.
2022.06.27	2.02	1. Updated *Section 13 Serial peripheral interface (SPI)* by adding SPI timing diagram. 2. Updated *Section 5.1 FLASH introduction.* 3. Updated *Section 13.2.6 DMA transfer* 4. Updated *Section 11.7.2 Control register2 (I2C_CTRL2)* 5. Updated *Figure 24-2* 6. Updated *Table* 6-2 7. Updated *Table 24-31* 8. Updated *Figure 1-1* 9. Updated *Section 20.6.7 Error management* 10. Updated *Section 20.7 CAN registers* 11.Updated *Section 24.7.1.5 SRAM/NOR Flash extra timing register x (XMC_EXTx) (x=1,2,3,4*) 12. Updated *11.7.1 Control register1 (I2C_CTRL1)*
2022.11.11	2.03	1. Updated descriptions of *Chapter 4* 2. Updated descriptions of *Chapter 10* 3. Updated descriptions of *Chapter 14* 4. Updated descriptions of *Chapter 17* 5. Updated descriptions of *Section 18.2*
2023.04.10	2.04	1. Added Table-3 448KB Flash memory architecture in *Section 5.1 FLASH introduction* 2. Modified the reset value of *section 22.6.4 Control register 2 (ACC_CTRL2)* 3. Updated Table 6-1 in *Section 6.2.9 IOMUX input/output* 4. Updated the descriptions of *Section 18 Analog-to-digital converter (ADC)* 5. Updated the descriptions of *Section 11 I2C interface* 6. Updated the descriptions of Table 5-5 in *Section 5.1 FLASH introduction* 7. Updated the descriptions of *Section 28 Qud-SPI interface (QSPI)* 8. Updated the descriptions of *Section 26 Ethernet media access control (EMAC)*

IMPORTANT NOTICE – PLEASE READ CAREFULLY

Purchasers are solely responsible for the selection and use of ARTERY's products and services; ARTERY assumes no liability for purchasers' selection or use of the products and the relevant services.

No license, express or implied, to any intellectual property right is granted by ARTERY herein regardless of the existence of any previous representation in any forms. If any part of this document involves third party's products or services, it does NOT imply that ARTERY authorizes the use of the third party's products or services, or permits any of the intellectual property, or guarantees any uses of the third party's products or services or intellectual property in any way.

Except as provided in ARTERY's terms and conditions of sale for such products, ARTERY disclaims any express or implied warranty, relating to use and/or sale of the products, including but not restricted to liability or warranties relating to merchantability, fitness for a particular purpose (based on the corresponding legal situation in any injudical districts), or infringement of any patent, copyright, or other intellectual property right.

ARTERY's products are not designed for the following purposes, and thus not intended for the following uses: (A) Applications that have specific requirements on safety, for example: life-support applications, active implant devices, or systems that have specific requirements on product function safety; (B) Aviation applications; (C) Auto-motive application or environment; (D) Aerospace applications or environment, and/or (E) weapons. Since ARTERY products are not intended for the above-mentioned purposes, if purchasers apply ARTERY products to these purposes, purchasers are solely responsible for any consequences or risks caused, even if any written notice is sent to ARTERY by purchasers; in addition, purchasers are solely responsible for the compliance with all statutory and regulatory requirements regarding these uses.

Any inconsistency of the sold ARTERY products with the statement and/or technical features specification described in this document will immediately cause the invalidity of any warranty granted by ARTERY products or services stated in this document by ARTERY, and ARTERY disclaims any responsibility in any form.

© 2023 ARTERY Technology - All Rights Reserved.