AC7802x Reference Manual

Version: 1.0

Release date: 2023-03-20

Document Revision History

Revision	Date	Author	Description
1.0	2023-03-20	AutoChips	Initial Version

Legal Notice

This reference manual contains information that is confidential to AUTOCHIPS Inc. Unauthorized use or disclosure of the information contained herein is prohibited. You may be held responsible for any loss or damages suffered by Autochips Inc. for your unauthorized disclosure hereof, in whole or in part.

Information herein is subject to change without noticed. AUTOCHIPS Inc. does not assume any responsibility for any use of, or reliance on, the information contained herein.

THIS REFERENCE MANUAL AND ALL INFORMATION CONTAINED HEREIN IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE. AUTOCHIPS INC. SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE. NEITHER DOES AUTOCHIPS INC. PROVIDE ANY WARRANTY WHATSOEVER WITH RESPECT TO THE SOFTWARE OF ANY THIRD PARTY WHICH MAY BE USED BY, INCORPORATED IN, OR SUPPLIED WITH THIS REFERENCE MANUAL, AND USER AGREES TO LOOK ONLY TO SUCH THIRD PARTY FOR ANY WARRANTY CLAIM RELATING THERETO. AUTOCHIPS INC. SHALL ALSO NOT BE RESPONSIBLE FOR ANY AUTOCHIPS DELIBERABLES MADE TO USER'S SPECIFICATION OR TO CONFORM TO A PARTICULAR STANDARD OR OPEN FORUM.

Table of Contents

Document Revision History **2**
Legal Notice **3**
Table of Contents **4**
List of Figures **18**
List of Tables **21**
Abbreviation **25**
1 Introduction **26**
1.1 Overview 26
1.2 Module description 26
2 Memory and Bus Architecture **28**
2.1 System architecture 28
2.2 Functional description 29
2.2.1 Memory layout 29
2.2.2 Internal SRAM 30
2.2.3 Fast GPIO memory map 30
2.2.4 Memory map 30
2.2.5 Embedded Flash memory 31
2.2.6 Embedded Flash memory read 32
2.2.7 Chip Model 32
2.2.8 Chip UUID 32
2.2.9 AHB to APB bridge 32
2.2.10 Nested Vectored Interrupt Controller(NVIC) 32
2.2.11 Boot configuration 34
2.3 Address assignment of Peripherals 35
3 Reset **36**
3.1 Features 36
3.2 Block diagram 36
3.3 Function description 37

3.3.1 Power On Reset (POR) 37
3.3.2 System Reset 38
3.4 Register Definition 39
3.4.1 Chip Reset Control Register(RESET_CTRL) 39
3.4.2 Chip Reset Status Register(RESET_STATUS) 40
4 Clock 43
4.1 Introduction 43
4.2 Block diagram 43
4.2.1 Clock control diagram 43
4.3 Register Definition 44
4.3.1 Clock Control Register(CKGEN_CTRL) 44
4.3.2 Peripheral Clock Enable Control 0(CKGEN_PERI_CLK_EN_0) 46
4.3.3 Peripheral Clock Enable Control 1(CKGEN_PERI_CLK_EN_1) 47
4.3.4 Peripheral Software Reset Control 0(CKGEN_PERI_SFT_RST0) 48
4.3.5 Peripheral software reset control 1(CKGEN_PERI_SFT_RST1) 50
5 Power Modes 52
5.1 Introduction 52
5.2 Functional description 52
5.3 Application note 52
5.3.1 Entering and exiting low power modes 52
5.3.2 Module operation in low power modes 52
6 System Power Management(SPM) 55
6.1 Introduction 55
6.2 Features 55
6.3 Application notes 55
6.3.1 SPM power control program guide 55
6.3.2 XOSC power control 55
6.4 Register Definition 56
6.4.1 Power Manager Configuration 0 (SPM_PWR_MGR_CFG0) 56
6.4.2 Power Manager Configuration 1(SPM_PWR_MGR_CFG1) 57

6.4.3 Peripheral Sleep Ack Status(SPM_PERIPH_SLEEP_ACK_STATUS) 58
6.4.4 Peripheral Sleep Ack Enable(SPM_EN_PERIPH_SLEEP_ACK) 59
6.4.5 Peripheral Wakeup Enable(SPM_EN_PERIPH_WKUP) 61
6.4.6 SPM Wakeup IRQ Flags Status(SPM_WAKEUP_IRQ_STATUS) 62
7 Universal Asynchronous Receiver/Transmitter(UART) 65
7.1 Introduction 65
7.2 Features 65
7.3 Block diagram 67
7.4 Function description 68
7.4.1 Noise detection 68
7.4.2 Baud rate description 69
7.4.3 LIN function 70
7.4.4 Two power mode 71
7.5 Application note 72
7.5.1 Baud rate configuration note 72
7.5.2 UART configuration note 73
7.6 Register Definition 74
7.6.1 RX/TX Data Register(UART_RBR/THR) 74
7.6.2 Divisor Low 8 Bits Register(UART_DIV_L) 75
7.6.3 Divisor High 8 Bits Register(UART_DIV_H) 75
7.6.4 UART Supplementary Control Register 0(UART_LCR0) 76
7.6.5 UART Supplementary Control Register 1(UART_LCR1) 77
7.6.6 FIFO Control Register(UART_FCR) 78
7.6.7 Interrupt Enable Register(UART_IER) 78
7.6.8 Status register 0(UART_LSR0) 80
7.6.9 Status register 1(UART_LSR1) 82
7.6.10 UART Sample Counter Register(UART_SMP_CNT) 83
7.6.11 Guard Time Added Register(UART_GUARD) 83
7.6.12 Sleep enable register(UART_SLEEP_EN) 84
7.6.13 Fractional Divider Register(UART_DIV_FRAC) 84

7.6.14 Idle Interrupt Enable Register(UART_IDLE) 85
7.6.15 Software LIN Control Register(UART_LINCR) 86
7.6.16 Software LIN Sync Break Length Control(UART_BRKLH) 87
8 Analog-to-Digital Converter(ADC) 88
8.1 Introduction 88
8.2 Features 88
8.3 ADC functional description 89
8.4 Function Description 90
8.4.1 Power on sequence 90
8.4.2 Operation modes 90
8.4.3 Trigger mode 98
8.4.4 Linear calibration 98
8.4.5 Analog monitor 99
8.4.6 Status flag 101
8.4.7 DATA alignment 102
8.4.8 Sampling conversion time 103
8.4.9 Temperature sensor 103
8.4.10 Low power mode 104
8.5 Application note 105
8.5.1 Reset and enable 105
8.5.2 ADC power-on delay 105
8.6 Register definition 106
8.6.1 Status Register(ADC_STR) 107
8.6.2 Control Register 0(ADC_CTRL0) 108
8.6.3 Control Register 0(ADC_CTRL1) 109
8.6.4 Sample Time Selection Register 0(ADC_SPT0) 110
8.6.5 Sample Time Selection Register 1(ADC_SPT1) 110
8.6.6 Injected Group Offset Register(ADC_IOFRx) 111
8.6.7 AMO High Threshold Register(ADC_AMOHR) 111
8.6.8 AMO Low Threshold Register(ADC_AMOLR) 112

8.6.9 Regular Group Sequence Configuration Register 0(ADC_RSQR0) 113
8.6.10 Regular Group Sequence Configuration Register 1(ADC_RSQR1) 113
8.6.11 Regular Group Sequence Configuration Register 2(ADC_RSQR2) 114
8.6.12 Injected Group Sequence Configuration Register(ADC_ISQR) 114
8.6.13 Injected group data register(ADC_IDRx) 115
8.6.14 Regular group data register(ADC_RDRx) 115
8.6.15 Gain Error Value Calibration Register(ADC_CGV) 115
8.6.16 Offset Error Value Calibration Register(ADC_COV) 116
8.6.17 Regular Group Conversion End Status Register(ADC_REOC) 116
8.6.18 Regular Group Conversion End Interrupt Enable Register (ADC_REOCEN) 117
8.6.19 Injected Group Conversion End Status Register(ADC_IEOC) 118
8.6.20 Injected Group conversion end interrupt enable register (ADC_IEOCEN) 118
8.6.21 Regular Group Sequence Configuration Register 3(ADC_RSQR3) 119
8.6.22 ADC Analog Configuration Register 0(ADC_CFG0) 119
8.6.23 ADC Analog Configuration Register 1(ADC_CFG1) 120

9 Analog Comparator(ACMP) 121

9.1 Introduction 121
9.2 Features 121
9.3 Block diagram 121
9.4 Functional description 122
9.4.1 Normal mode 122
9.4.2 Polling mode 122
9.4.3 HALL output in polling mode 123
9.4.4 Hysteresis 123
9.4.5 DAC output 124
9.4.6 Low power mode wakeup 124
9.5 Register definition 124
9.5.1 Configuration Register 0(ACMP_CR0) 125

9.5.2 Configuration Register 1(ACMP_CR1) 126
9.5.3 Configuration Register 2(ACMP_CR2) 127
9.5.4 Configuration Register 3(ACMP_CR3) 127
9.5.5 Configuration Register 4(ACMP_CR4) 128
9.5.6 Data Output Register(ACMP_DR) 128
9.5.7 Status register(ACMP_SR) 130
9.5.8 Polling Frequency Divider Register(ACMP_FD) 131
9.5.9 Hall A Output Setting Register(ACMP_OPA) 132
9.5.10 Hall B Output Setting Register(ACMP_OPB) 132
9.5.11 Hall C Output Setting Register(ACMP_OPC) 133
9.5.12 DAC Reference Source Select Register(ACMP_DACSR) 134
9.5.13 Analog Configuration Register(ACMP_CFG) 134

10 Pulse Width Modulation(PWM) 136

10.1 Introduction 136
10.2 Features 136
10.3 Block diagram 137
10.4 Functional description 138
10.4.1 Clock source 138
10.4.2 Counter 138
10.4.3 Operation mode 139
10.4.4 Input capture mode 140
10.4.5 Output Compare mode 141
10.4.6 Edge-Aligned PWM(EPWM) mode 142
10.4.7 Center-Aligned PWM(CPWM) mode 143
10.4.8 Combine mode 144
10.4.9 Dual Edge Capture mode 152
10.4.10 Quadrature decoder mode 153
10.4.11 Write protection 156
10.4.12 Initialization 156
10.4.13 Polarity control 156

10.4.14 Output mask 157
10.4.15 Software output control 157
10.4.16 Initialization trigger 157
10.4.17 Channel match trigger 157
10.4.18 Fault control 158
10.4.19 Registers updated from write buffers 159
10.4.20 PWM synchronization 160
10.4.21 Features priority 168
10.4.22 Global time base (GTB) 169
10.4.23 PWM interrupts 169
10.4.24 Low power mode 170
10.5 Register definition 170
10.5.1 Initialization Register(PWM_INIT) 171
10.5.2 Counter Value Register(PWM_CNT) 172
10.5.3 Maximum Count Value Register(PWM_MCVR) 172
10.5.4 Channel Status and Control Register(PWM_CHnSCR) 173
10.5.5 Channel Value(PWM_CHnV) 174
10.5.6 Counter Initial Value Register(PWM_CNTIN) 175
10.5.7 Capture and Compare Status Register(PWM_STR) 175
10.5.8 Function Mode Selection Register(PWM_FUNCSEL) 176
10.5.9 Synchronization Register(PWM_SYNC) 178
10.5.10 Initial State for Channels Output(PWM_OUTINIT) 180
10.5.11 Output Mask Control Register(PWM_OMCR) 181
10.5.12 Mode Selection Register(PWM_MODESEL) 182
10.5.13 Dead-time Setting Register(PWM_DTSET) 184
10.5.14 External Trigger Register(PWM_EXTTRIG) 185
10.5.15 Channel Output Polarity Control Register(PWM_CHOPOLCR) 186
10.5.16 Fault Detect Status Register(PWM_FDSR) 188
10.5.17 Input Capture Mode Filter Control(PWM_CAPFILTER) 189
10.5.18 Fault Filter and Fault Enable Register(PWM_FFAFER) 190
10.5.19 Quadrature Decoder Control and Status(PWM_QDI) 191

10.5.20 Configuration Register(PWM_CONF) 193
10.5.21 Fault Input Polarity Register(PWM_FLTPOL) 195
10.5.22 Synchronization Configuration Register(PWM_SYNCONF) 196
10.5.23 Inverting Control Register(PWM_INVCR) 198
10.5.24 Channel Software Output Control Register(PWM_CHOSWCR) 199
11 Pulse Width Detect Timer(PWDT) 201
11.1 Introduction 201
11.2 Features 201
11.3 Block diagram 202
11.4 Functional description 202
11.4.1 Pulse Width Measurement function 202
11.4.2 Timer function 206
11.4.3 Interrupt request 206
11.4.4 Low power mode 206
11.5 Application note 207
11.5.1 Pulse Width Measurement function program guide 207
11.5.2 Timer function program guide 207
11.6 Register definition 207
11.6.1 Initialization Register 0(PWDT_INIT0) 207
11.6.2 NPW count value(PWDT_NPW) 209
11.6.3 Initialization Register 1(PWDT_INIT1) 209
12 TIMER 211
12.1 Introduction 211
12.2 Features 211
12.3 Block diagram 211
12.4 Functional description 212
12.4.1 General mode 212
12.4.2 Link mode 212
12.4.3 Interrupts 212
12.5 Register definition 212

12.5.1 Module Controller Register(TIMER_MCR) 213
12.5.2 Initial Value Register(TIMER_LDVAL) 213
12.5.3 Current Value Register(TIMER_CVAL) 214
12.5.4 Initial Register(TIMER_INIT) 214
12.5.5 Interrupt Flag Register(TIMER_TF) 215
13 Connection Transmission Unit(CTU) 216
13.1 Introduction 216
13.2 Features 216
13.3 Block diagram 217
13.4 Function description 217
13.4.1 ACMP output capture 217
13.4.2 RTC capture 218
13.4.3 ADC hardware trigger 218
13.4.4 PWM software synchronization 218
13.4.5 Low power mode 218
13.5 Register definition 218
13.5.1 CTU Configuration Register 0(CTU_CONFIG0) 218
13.5.2 CTU Configuration Register 1(CTU_CONFIG1) 220
14 General Purpose Input/Output(GPIO) 222
14.1 Introduction 222
14.2 Features 222
14.3 Block diagram 223
14.4 Functional description 224
14.4.1 External interrupt 224
14.4.2 Multi-Function 226
14.4.3 Low power mode 227
14.5 Application note 228
14.5.1 External input 228
14.5.2 Multi-Function 228
14.5.3 Open-drain output 228

14.5.4 HIGH_Z mode 228
14.5.5 GPIO function 229
14.5.6 Programming guide 229
14.6 Register description 229
14.6.1 Port Configuration Register(GPIO_CR) 230
14.6.2 Port Input Data Register(GPIO_IDR) 230
14.6.3 Port Output Data Register(GPIO_ODR) 231
14.6.4 Port Set/Reset Register (GPIO_BSRR) 232
14.6.5 Port reset register(GPIO_BRR) 233
14.6.6 Pull-down Enable Register(GPIO_PD) 233
14.6.7 Pull-Up Enable Register(GPIO_PU) 234
14.6.8 Driving Capability Selection Register(GPIO_E4_E2) 234
14.6.9 Input Enable Register(GPIO_IES) 235
14.6.10 Multi-function Selection Register(GPIO_PINMUX) 235
14.6.11 External interrupt flag pending(GPIO_PR) 236
14.6.12 Interrupt mask register(GPIO_IMR) 237
14.6.13 Rising Edge Trigger Event Configuration(GPIO_RTSR) 237
14.6.14 Falling Edge Trigger Event Configuration(GPIO_FTSR) 237
14.6.15 External interrupt register(GPIO_EXTICR) 238
15 Inter-IC(I2C) 239
15.1 Introduction 239
15.2 Features 239
15.3 Block diagram 240
15.3.1 I2C signal 240
15.3.2 Baud rate component 241
15.3.3 Data flow 241
15.4 Functional description 242
15.4.1 Master mode 242
15.4.2 Slave mode 243
15.4.3 SMBus 244

15.4.4 Interrupt request 244
15.4.5 Slave low power wakeup 245
15.5 Application note 245
15.5.1 Data transmission 245
15.5.2 ACK control 246
15.6 Register description 247
15.6.1 Address Register 0(I2C_ADDR0) 247
15.6.2 Address Register 1(I2C_ADDR1) 248
15.6.3 Baud Rate Configuration Register 0(I2C_SAMPLE_CNT) 248
15.6.4 Baud Rate Configuration Register 1(I2C_STEP_CNT) 248
15.6.5 Control Register 0(I2C_CTRL0) 249
15.6.6 Control Register 1(I2C_CTRL1) 250
15.6.7 Control Register 2(I2C_CTRL2) 251
15.6.8 Control Register 3(I2C_CTRL3) 252
15.6.9 Status register 0(I2C_STATUS0) 252
15.6.10 Status register 1(I2C_STATUS1) 254
15.6.11 Deglitch configuration register(I2C_DGLCFG) 256
15.6.12 Data Register(I2C_DATA) 257
15.6.13 Master START STOP Signal Control Register(I2C_STARTSTOP) 257
16 Serial Peripheral Interface(SPI) 259
16.1 Introduction 259
16.2 Features 259
16.3 Block diagram 260
16.4 Functional description 260
16.4.1 Data flow & Algorithm 260
16.4.2 Input & Output timing 261
16.4.3 Master SCK output timing setting 263
16.4.4 Master mode fault detect 263
16.4.5 Slave low power wakeup 264
16.4.6 Interrupt 265

16.5 Application note 266
16.5.1 Master CS continuous mode 266
16.5.2 Master CS discontinuous output 266
16.5.3 Slave mode 267
16.6 Register definition 267
16.6.1 SPI configuration register 0(SPI_CFG0) 268
16.6.2 SPI Configuration Register 1(SPI_CFG1) 268
16.6.3 SPI Command Register(SPI_CMD) 271
16.6.4 SPI Status Register(SPI_STATUS) 272
16.6.5 SPI Data Register(SPI_DATA) 274
16.6.6 SPI Configuration Register 2(SPI_CFG2) 274
17 Watchdog Timer(WDG) 276
17.1 Introduction 276
17.2 Features 276
17.3 Block diagram 277
17.4 Functional description 277
17.4.1 Basic Watchdog 277
17.4.2 Watchdog default timeout behavior 277
17.4.3 Window Watchdog 278
17.4.4 Low-power behavior 278
17.4.5 Debug mode 278
17.5 Application note 278
17.5.1 Configuring the Watchdog 278
17.5.2 Watchdog refresh mechanism 278
17.5.3 Watchdog interrupt 279
17.6 Register definition 279
17.6.1 Watchdog Control and Status Register 0(WDG_CS0) 279
17.6.2 Watchdog Control and Status Register 1(WDG_CS1) 280
17.6.3 Watchdog Counter Register(WDG_CNT) 281
17.6.4 Watchdog Timeout Value Register(WDG_TOVAL) 281

17.6.5 Watchdog Window Value Register(WDG_WIN) 282
18 Real Time Counter(RTC) 283
18.1 Introduction 283
18.2 Features 283
18.3 Block diagram 283
18.4 Functional description 283
18.4.1 Clock source selection 283
18.4.2 Counting 284
18.4.3 RTC timing signal output 284
18.4.4 Low power wake up 284
18.5 Application note 284
18.5.1 Basic use of RTC 284
18.5.2 RTC low power wake up 284
18.6 Register definition 285
18.6.1 RTC Control and Status Register(RTC_SC) 285
18.6.2 RTC Modulo Register(RTC_MOD) 287
18.6.3 RTC Counter Register(RTC_CNT) 287
18.6.4 RTC Clock Prescaler Register(RTC_PS) 288
18.6.5 RTC Prescaler Counter Register(RTC_PSCNT) 288
19 Embedded Flash 289
19.1 Introduction 289
19.2 Features 289
19.3 Block diagram 290
19.4 Functional description 290
19.4.1 Embedded flash memory organization 290
19.4.2 Embedded flash protection 291
19.4.3 Flash command ID 293
19.5 Application note 293
19.5.1 Page erase 294
19.5.2 Block erase 295

19.5.3 Mass erase 296
19.5.4 Page program 296
19.5.5 Page erase verify 298
19.5.6 Mass erase verify 298
19.5.7 Option byte erase 299
19.5.8 Option byte program 300
19.6 Register definition 301
19.6.1 Unlock Sequence Register(EFLASH_UKR) 302
19.6.2 Global Status Register(EFLASH_GSR) 302
19.6.3 Global Control Register(EFLASH_GCR) 303
19.6.4 Command Status Register(EFLASH_CSR) 304
19.6.5 Command Control Register(EFLASH_CCR) 306
19.6.6 Command Address Register(EFLASH_CAR) 307
19.6.7 Command Data Register(EFLASH_CDR) 307
19.6.8 P-Flash Write Protection Register 0(EFLASH_PWPR0) 308
19.6.9 P-Flash Write Protection Register 1(EFLASH_PWPR1) 308
19.6.10 D-Flash Write Protection Register(EFLASH_DWPR) 309
20 SRAM ECC 310
20.1 Introduction 310
20.2 Features 310
20.3 Functional description 310
20.4 Register definition 311
20.4.1 ECC Control and Status Register(ECC_SRAM_CTRL) 312
20.4.2 ECC 1 Bit Error Address Register(ECC_SRAM_ERR1_ADDR) 313
20.4.3 ECC 2 Bits Error Address Register(ECC_SRAM_ERR2_ADDR) 313
21 Debug 314
21.1 Introduction 314
21.2 Features 314

List of Figures

Figure 2-1 System architecture 28
Figure 3-1 Reset block diagram 37
Figure 4-1 Clock control diagram 44
Figure 7-1 UART block diagram 67
Figure 7-2 UART transmitter flow 68
Figure 7-3 UART receiver flow 68
Figure 7-4 UART noise detection 68
Figure 7-5 LIN frame flow 70
Figure 7-6 Run mode and Stop mode condition 71
Figure 7-7 Typical flow for waking up the chip by UART 72
Figure 7-8 Diagram for baud rate generator 72
Figure 8-1 ADC block diagram 89
Figure 8-2 ADC power on sequence 90
Figure 8-3 Regular group sequence 92
Figure 8-4 Valid regular group sequence 92
Figure 8-5 Injected group sequence 92
Figure 8-6 Valid injected group sequence 92
Figure 8-7 Mode 1 operation flow 93
Figure 8-8 Mode 2 operation flow 93
Figure 8-9 Mode 3 operation flow with injected trigger scan mode 94
Figure 8-10 Mode 3 operation flow with injected trigger at ADC idle state 94
Figure 8-11 Mode 3 operation flow with injected trigger discontinuous mode 94
Figure 8-12 Mode 4 operation flow 95
Figure 8-13 Mode 5 operation flow of injected group scan mode 95
Figure 8-14 Mode 5 operation flow with injected trigger at ADC idle state 96
Figure 8-15 Mode 5 operation flow with injected trigger discontinuous mode 96
Figure 8-16 Mode 6 operation flow 96
Figure 8-17 Mode 7 operation flow 97
Figure 8-18 Mode 8 operation flow 98
Figure 8-19 ADC GEOE calibration conversion diagram 99
Figure 8-20 Monitor region in level trigger mode 100
Figure 8-21 Monitor detecting region in edge trigger mode 101
Figure 8-22 Three flag behaviors 102
Figure 8-23 ADC data arrangement in data register 102
Figure 8-24 CPU power mode switching flow 104
Figure 9-1 ACMP block diagram 121
Figure 9-2 Operation flow in polling mode 123
Figure 9-3 Hysteresis 123
Figure 9-4 DAC output configuration 124
Figure 10-1 PWM block diagram 137
Figure 10-2 Up counting 138
Figure 10-3 Up-Down counting 139
Figure 10-4 Input capture mode 141
Figure 10-5 Match setting output compare mode 141
Figure 10-6 Match clear output compare mode 142
Figure 10-7 Match toggle output compare mode 142
Figure 10-8 Edge-aligned PWM mode 143
Figure 10-9 CPWM waveform 144
Figure 10-10 CH(n) output waveform in up counting mode 145
Figure 10-11 CH(n) output waveform if (CNTIN < CHnV/CH(n+1)V < MCVR) &(CHnV < CH(n+1)V) 145
Figure 10-12 CH(n) output waveform if (CNTIN < CHnV < MCVR)&(CH(n+1)V = MCVR) 146

Figure 10-13 CH(n) output waveform if (CHnV = CNTIN)&(CNTIN < CH(n+1)V < MCVR) 146
Figure 10-14 CH(n) output if (CNTIN < CHnV/ CH(n+1)V < MCVR) & (CHnV > CH(n+1)V) ... 147
Figure 10-15 CH(n) output if (CH(n+1)V < CNTIN)&(CNTIN < CHnV < MCVR) 147
Figure 10-16 CH(n) output if (CH(n+1)V > MCVR) & (CNTIN < CHnV < MCVR) 148
Figure 10-17 Up-down counting range define 148
Figure 10-18 CHn matching point DIR=1(Up), CH(n+1) matching point DIR=1(Up)..... 149
Figure 10-19 CHn matching point DIR=1(Up), CH(n+1) matching point DIR=0(Down) 149
Figure 10-20 CHn matching point DIR=0(Down), CH(n+1) matching point DIR=1(Up) 149
Figure 10-21 CHnV matching point DIR=0(Down), CH(n+1)V matching point DIR=0(Down) ... 150
Figure 10-22 Output in complementary mode 150
Figure 10-23 Dead-time insertion 151
Figure 10-24 CH(n+1)V output in the next period matching 152
Figure 10-25 Multi-channel phase shift output waveform 152
Figure 10-26 Dual edge capture mode 153
Figure 10-27 Quadrature Decoder—Count and Direction Encoding mode 154
Figure 10-28 Quadrature Decoder—Phase A and Phase B Encoding Mode 155
Figure 10-29 PWM Counter Overflow in Up Counting for Quadrature Decoder Mode..... 155
Figure 10-30 PWM Counter Overflow in Down Counting for Quadrature Decoder Mode 156
Figure 10-31 Hardware trigger event with HWTRIGMODESEL = 0..... 160
Figure 10-32 Software trigger event 161
Figure 10-33 Synchronization points 162
Figure 10-34 PWM_MCVR register synchronization flowchart 163
Figure 10-35 PWM_CNT register synchronization flow 164
Figure 10-36 PWM_OMCR register synchronization flow chart..... 165
Figure 10-37 PWM_INVCR register synchronization flowchart 166
Figure 10-38 PWM_CHOSWCR register synchronization flowchart..... 167
Figure 10-39 PWM_CHOPOLCR register synchronization flowchart 168
Figure 10-40 Features priority 169
Figure 11-1 PWDT block diagram 202
Figure 11-2 Four basic measurement modes(HALLEN=0) 203
Figure 11-3 Hall measurement modes(HALLEN=1) 203
Figure 11-4 Two common installation ways 204
Figure 11-5 Example for low level noise and filter..... 205
Figure 11-6 Example for high level noise and filter 205
Figure 11-7 PWDTC counter and counting error 206
Figure 11-8 Modify the TIMLDTVAL during TIMEN=0 206
Figure 11-9 Modify the TIMLDTVAL during TIMEN=1 206
Figure 12-1 TIMER block diagram..... 211
Figure 13-1 CTU block diagram 217
Figure 14-1 GPIO block diagram..... 223
Figure 14-2 GPIO external interrupt 224
Figure 14-3 GPIO multi-function 226
Figure 15-1 I2C block diagram 240
Figure 15-2 START and STOP conditions 240
Figure 15-3 Data transmission format..... 240
Figure 15-4 Baud rate generation 241
Figure 15-5 Data flow of transmitter 242
Figure 15-6 Data flow of Receiver 242
Figure 15-7 Master combined mode 243
Figure 15-8 BND sequence of master write slave mode 245
Figure 15-9 BND sequence of master read slave mode..... 246
Figure 15-10 Typical I2C slave interrupt routine 246
Figure 16-1 SPI system connection 259
Figure 16-2 SPI block diagram 260
Figure 16-3 Data flow of master 260

Figure 16-4 Data flow of slave 261
Figure 16-5 CPHA=0 transmission format 262
Figure 16-6 CPHA=1 transmission format 263
Figure 16-7 Baud rate generation 263
Figure 16-8 SCK output timing with mode fault detect enable 264
Figure 16-9 Limitation of mode fault detect 264
Figure 16-10 Wakeup sequence 265
Figure 16-11 CS continuous mode 266
Figure 17-1 WDG block diagram 277
Figure 18-1 RTC block diagram 283
Figure 19-1 Block diagram for eflash and eflash controller 290
Figure 19-2 Data flow for eflash and eflash controller 290
Figure 19-3 Page erase command operation flow 295
Figure 19-4 Block erase command operation flow 295
Figure 19-5 Mass erase command operation flow 296
Figure 19-6 Page program command operation flow 297
Figure 19-7 Page erase verify command operation flow 298
Figure 19-8 Mass erase verify command operation flow 299
Figure 19-9 Option byte erase command operation flow 300
Figure 19-10 Option byte program command operation flow 301

List of Tables

Table 1-1 AC7802x module description 26
Table 2-1 Memory layout in Little Endian format 29
Table 2-2 Device memory map 30
Table 2-3 AC7802x interrupt table 33
Table 2-4 Boot configuration 34
Table 2-5 Address assignment of each peripheral 35
Table 3-1 Reset register map 39
Table 3-2 RESET_CTRL register 39
Table 3-3 RESET_STATUS register 40
Table 4-1 Clock register map 44
Table 4-2 CKGEN_CTRL register 44
Table 4-3 CKGEN_PERI_CLK_EN_0 register 46
Table 4-4 CKGEN_PERI_CLK_EN_1 register 47
Table 4-5 CKGEN_PERI_SFT_RST0 register 48
Table 4-6 CKGEN_PERI_SFT_RST1 register 50
Table 5-1 Module functionality in low-power modes 52
Table 6-1 SPM register map 56
Table 6-2 SPM_PWR_MGR_CFG0 register 56
Table 6-3 SPM_PWR_MGR_CFG1 register 57
Table 6-4 SPM_PERIPH_SLEEP_ACK_STATUS register 58
Table 6-5 SPM_EN_PERIPH_SLEEP_ACK register 59
Table 6-6 SPM_EN_PERIPH_WKUP register 61
Table 6-7 SPM_WAKEUP_IRQ_STATUS register 62
Table 7-1 UART function classification and configuration 65
Table 7-2 Typical baud rate and error rate@bclock=32MHz 69
Table 7-3 Typical baud rate and error rate@bclock=8MHz 69
Table 7-4 UART register map 74
Table 7-5 UART_RBR/THR register 74
Table 7-6 UART_DIV_L register 75
Table 7-7 UART_DIV_H register 75
Table 7-8 UART_LCR0 register 76
Table 7-9 UART_LCR1 register 77
Table 7-10 UART_FCR register 78
Table 7-11 UART_IER register 78
Table 7-12 UART_LSR0 register 80
Table 7-13 UART_LSR1 register 82
Table 7-14 UART_SMP_CNT register 83
Table 7-15 UART_GUARD register 83
Table 7-16 UART_SLEEP_EN register 84
Table 7-17 UART_DIV_FRAC register 84
Table 7-18 UART_IDLE register 85
Table 7-19 UART_LINCR register 86
Table 7-20 UART_BRKLH register 87
Table 8-1 ADC operation modes and its corresponding configuration 91
Table 8-2 Responsive behavior under different triggering methods 98
Table 8-3 Analog monitor channel configuration 100
Table 8-4 Analog sampling rate 103
Table 8-5 ADC reset range 105
Table 8-6 ADC register map 106
Table 8-7 ADC_STR register 107
Table 8-8 ADC_CTRL0 register 108
Table 8-9 ADC_CTRL1 register 109

Table 8-10 ADC_SPT0 register 110
Table 8-11 ADC_SPT1 register 110
Table 8-12 ADC_IOFRx (x=0~3)register 111
Table 8-13 ADC_AMOHR register 111
Table 8-14 ADC_AMOLR register 112
Table 8-15 ADC_RSQR0 register 113
Table 8-16 ADC_RSQR1 register 113
Table 8-17 ADC_RSQR2 register 114
Table 8-18 ADC_ISQR register 114
Table 8-19 ADC_IDRx(x=0~3) register 115
Table 8-20 ADC_RDRx register 115
Table 8-21 ADC_CGV register 115
Table 8-22 ADC_COV register 116
Table 8-23 ADC_REOC register 116
Table 8-24 ADC_REOCEN register 117
Table 8-25 ADC_IEOC register 118
Table 8-26 ADC_IEOCEN register 118
Table 8-27 ADC_RSQR3 register 119
Table 8-28 ADC_CFG0 register 119
Table 8-29 ADC_CFG1 register 120
Table 9-1 ACMP register map 124
Table 9-2 ACMP_CR0 register 125
Table 9-3 ACMP_CR1 register 126
Table 9-4 ACMP_CR2 register 127
Table 9-5 ACMP_CR3 register 127
Table 9-6 ACMP_CR4 register 128
Table 9-7 ACMP_DR register 128
Table 9-8 ACMP_SR register 130
Table 9-9 ACMP_FD register 131
Table 9-10 ACMP_OPA register 132
Table 9-11 ACMP_OPB register 132
Table 9-12 ACMP_OPC register 133
Table 9-13 ACMP_DACSR register 134
Table 9-14 ACMP_ANACFG register 134
Table 10-1 Operation mode configuration 139
Table 10-2 Software output control behavior in combine mode 157
Table 10-3 Fault Source and Number Table 158
Table 10-4 PWM_CNTIN register update buffer 159
Table 10-5 PWM_CH(n)V register update buffer 159
Table 10-6 PWM_MCVR register update buffer 159
Table 10-7 PWM low power mode 170
Table 10-8 PWM register map 170
Table 10-9 PWM_INIT register 171
Table 10-10 PWM_CNT register 172
Table 10-11 PWM_MCVR register 172
Table 10-12 PWM_CHnSCR register 173
Table 10-13 PWM_CHnV register 174
Table 10-14 PWM_CNTIN register 175
Table 10-15 PWM_STR register 175
Table 10-16 PWM_FUNCSEL register 176
Table 10-17 PWM_SYNC register 178
Table 10-18 PWM_OUTINIT register 180
Table 10-19 PWM_OMCR register 181
Table 10-20 PWM_MODESEL register 182
Table 10-21 PWM_DTSET register 184

Table 10-22 PWM_EXTTRIG register 185
Table 10-23 PWM_CHOPOLCR register 186
Table 10-24 PWM_FDSR register 188
Table 10-25 PWM_CAPFILTER register 189
Table 10-26 PWM_FFAFER register 190
Table 10-27 PWM_QDI register 191
Table 10-28 PWM_CONF register 193
Table 10-29 PWM_FLTPOL register 195
Table 10-30 PWM_SYNCONF register 196
Table 10-31 PWM_INVCR register 198
Table 10-32 PWM_CHOSWCR register 199
Table 11-1 Filterable pulse width range 204
Table 11-2 PWDT interrupt summary 206
Table 11-3 PWDT low power mode 206
Table 11-4 PWDT register map 207
Table 11-5 PWDT_INIT0 register 207
Table 11-6 PWDT_NPW register 209
Table 11-7 PWDT_INIT1 register 209
Table 12-1 TIMER register map 212
Table 12-2 TIMER_MCR register 213
Table 12-3 TIMER_LDVAL register 213
Table 12-4 TIMER_CVAL register 214
Table 12-5 TIMER_INIT register 214
Table 12-6 TIMER_TF register 215
Table 13-1 CTU low power mode 218
Table 13-2 CTU register map 218
Table 13-3 CTU_CONFIG0 register 218
Table 13-4 CTU_CONFIG1 register 220
Table 14-1 Corresponding relationship between GPIO external interrupt and ISR 225
Table 14-2 GPIO multi-function 226
Table 14-3 GPIO register map 229
Table 14-4 GPIO_CR register 230
Table 14-5 GPIO_IDR register 230
Table 14-6 GPIO_ODR register 231
Table 14-7 GPIO_BSRR register 232
Table 14-8 GPIO_BRR register 233
Table 14-9 GPIO_PD register 233
Table 14-10 GPIO_PU register 234
Table 14-11 GPIO_E4_E2 register 234
Table 14-12 GPIO_IES register 235
Table 14-13 GPIO_PINMUX register 235
Table 14-14 GPIO_PR register 236
Table 14-15 GPIO_IMR register 237
Table 14-16 GPIO_RTSR register 237
Table 14-17 GPIO_FTSR register 238
Table 14-18 GPIO_EXTICR register 238
Table 15-1 I2C Interrupt summary 244
Table 15-2 I2C register map 247
Table 15-3 I2C_ADDR0 register 247
Table 15-4 I2C_ADDR1 register 248
Table 15-5 I2C_SAMPLE_CNT register 248
Table 15-6 I2C_STEP_CNT register 248
Table 15-7 I2C_CRTL0 register 249
Table 15-8 I2C_CRTL1 register 250
Table 15-9 I2C_CRTL2 register 251

Table 15-10 I2C_CRTL3 register 252
Table 15-11 I2C_STATUS0 register 252
Table 15-12 I2C_STATUS1 register 254
Table 15-13 I2C_DGLCFG register 256
Table 15-14 I2C_DATA register 257
Table 15-15 I2C_STARTSTOP register 257
Table 16-1 Interrupt summary 265
Table 16-2 SPI register map 267
Table 16-3 SPI_CFG0 register 268
Table 16-4 SPI_CFG1 register 268
Table 16-5 SPI_CMD register 271
Table 16-6 SPI_STATUS register 272
Table 16-7 SPI_DATA register 274
Table 16-8 SPI_CFG2 register 274
Table 17-1 WDG register map 279
Table 17-2 WDG_CS0 register 279
Table 17-3 WDG_CS1 register 280
Table 17-4 WDG_CNT register 281
Table 17-5 WDG_TOVAL register 281
Table 17-6 WDG_WIN register 282
Table 18-1 RTC register map 285
Table 18-2 RTC_SC register 285
Table 18-3 RTC_MOD register 287
Table 18-4 RTC_CNT register 287
Table 18-5 RTC_PS register 288
Table 18-6 RTC_PSCNT register 288
Table 19-1 Embedded Flash memory organization 290
Table 19-2 Content list in option byte page 291
Table 19-3 Read protection setting 292
Table 19-4 Write protection setting 292
Table 19-5 Flash command ID 293
Table 19-6 Embedded flash register map 301
Table 19-7 EFLASH_UKR register 302
Table 19-8 EFLASH_GSR register 302
Table 19-9 EFLASH_GCR register 303
Table 19-10 EFLASH_CSR register 304
Table 19-11 EFLASH_CCR register 306
Table 19-12 EFLASH_CAR register 307
Table 19-13 EFLASH_CDR register 307
Table 19-14 EFLASH_PWPR0 register 308
Table 19-15 EFLASH_PWPR1 register 308
Table 19-16 EFLASH_DWPR register 309
Table 20-1 SRAM ECC register map 311
Table 20-2 ECC_SRAM_CTRL register 312
Table 20-3 ECC_SRAM_ERR1_ADDR register 313
Table 20-4 ECC_SRAM_ERR2_ADDR register 313

Abbreviation

AAI	Auto Address Increment
AHB	Advanced High Performance Bus
APB	Advanced Peripheral Bus
CKGEN	Clock Generator
ECC	Error Checking and Correction
HSE	High Speed External Clock
HSI	High Speed Internal Clock
LRU	Least Recently Used
LSI	Low Speed Internal Clock
LVR	Low Voltage Reset
NMI	Non Maskable Interrupt
OSC	Oscillator
POR	Power On Reset
LVD	Low Voltage Detect
SPM	System Power Manager
SRAM	Static Random-Access Memory
XOSC	External Crystal Oscillator

1 Introduction

1.1 Overview

AC7802x is a high-performance, low-power MCU using ARM Cortex™-M0+ core.

- Up to 32 MHz CPU frequency
- Temperature range (ambient): -40°C to +125°C
- Voltage range: 2.7 V to 5.5 V

1.2 Module description

The following sections describe the modules assigned to each category in more detail.

Table 1-1 AC7802x module description

Module	Description
ARM Cortex™-M0+ core	▪ ARM Cortex™-M0+ 32-bit MCU core ▪ Up to 32 MHz CPU frequency
Memories	▪ Up to 32 Kbyte P-Flash memory and 2 Kbyte D-Flash memory ▪ Up to 4 Kbyte SRAM with ECC
Clocks	▪ External crystal oscillator or resonator(HSE) – Range: 8 MHz ~ 20 MHz ▪ External input clock – ≤ 20 MHZ ▪ Internal oscillator – 32 MHz oscillator(HSI) – 32 kHz oscillator(LSI)
System	▪ One Clock module ▪ One Reset module ▪ One system power management(SPM) module ▪ One watchdog timer(WDG) module
Analog	▪ One 12-bit analog-to-digital converter(ADC) with up to 18 external channels and 1 internal channels ▪ One analog comparators (ACMP) with one 6-bit digital-to-analog converter (DAC), which can output to pin
Timers	▪ One 4-channel PWM controller ▪ Two 2-channel PWM controllers ▪ Four 32bit general timers(Timer) ▪ Real time clock(RTC) ▪ One pulse width detection timers(PWDT) ▪ System tick timer(SysTick)

Communications	▪ One serial peripheral interface(SPI) module ▪ One inter-integrated circuit(I2C) module ▪ Two universal asynchronous receiver/transmitter(UART) modules, only UART0 supports LIN
Human-Machines Interfaces	▪ Up to 27 general purpose input/output(GPIO) controllers ▪ Non-maskable interrupt(NMI) ▪ Five external Interrupts(IRQ)
Debug Interfaces	▪ Serial Wire Debug(SWD) interfaces

2 Memory and Bus Architecture

2.1 System architecture

The main system of AC7802x consists of:

- One master
  - Cortex™-M0+ core AHB-Lite bus
- Four slaves
  - Internal SRAM
  - Internal Flash memory
  - Fast IO and GPIO
  - AHB to APB bridge(AHB_APB), which connect all the APB peripherals.

These are interconnected using a multilayer AHB bus architecture as shown in Figure 2-1.

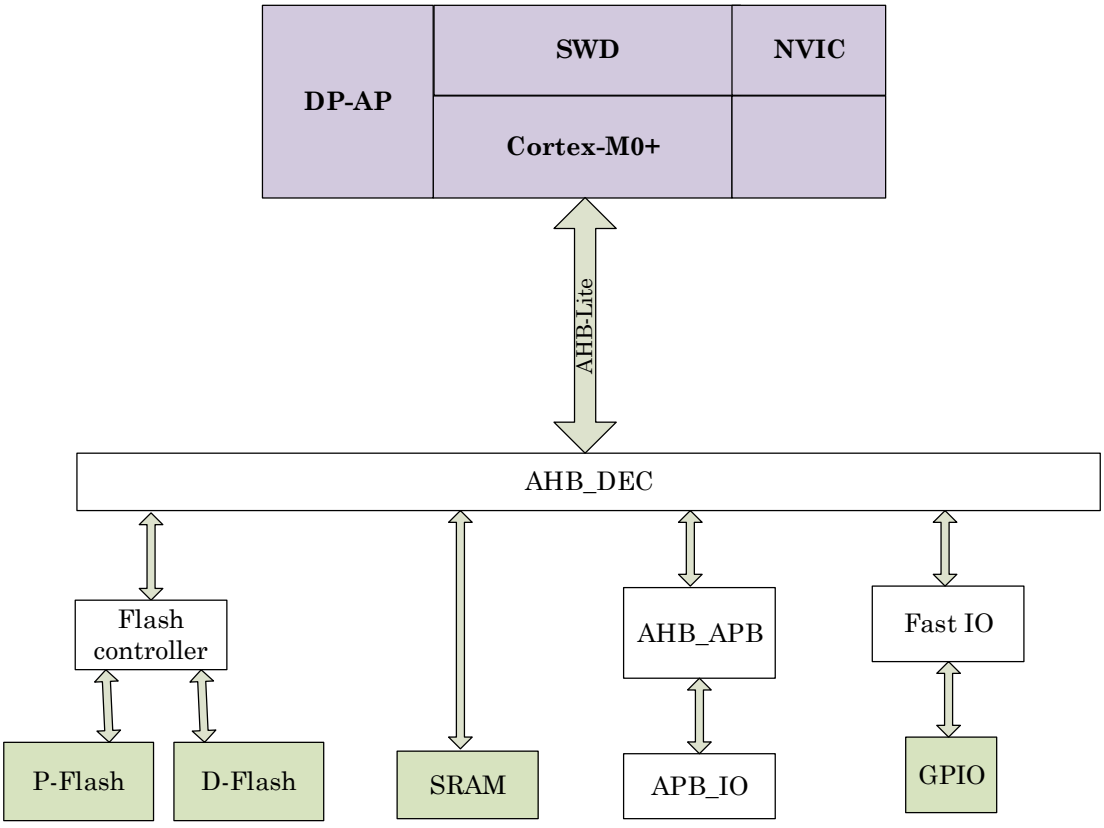

Figure 2-1 System architecture

AHB-Lite bus

This bus sends instruction and data requests of the Cortex™-M0+ core and accesses memories and peripherals.

AHB_DEC

The AHB decode module.

The arbitrated address will be decoded according to the address range, and then the decoded address can access different slavers.

BusMatrix

The BusMatrix consists of a master bus and a slave bus.

Internal Flash can only be accessed by the AHB-Lite bus.

AHB2APB bridge

The AHB2APB bridge provides full synchronous connections between the AHB and the APB buses(the AHB_APB unit in Figure 2-1). APB is limited to ½ frequency of AHB frequency by default.

DP-AP

DP-AP is the debug access port, which is composed of DP (debug port) and AP(access port).

The DP interface module (AC7802x only supports SW-DP) is to convert external signals into 32-bit debug bus signals.

The AP interface module is equivalent to a bus bridge, which is used to convert debug bus commands into data on the AHB bus and then transmit them.

Through the cooperation of DP and AP, SWD can access the address of AC7802x.

2.2 Functional description

2.2.1 Memory layout

Program memory, data memory, registers and I/O ports are organized within the same linear 4-Gbyte address space.

The bytes are coded in memory in Little Endian format, which means the lowest numbered byte in a word is considered the word's least significant byte and the highest numbered byte the most significant. For example, the storage method of the 16-bit wide number(0x1234) in the little-endian format CPU memory (assuming the memory start address is 0x4000) is as follows.

Table 2-1 Memory layout in Little Endian format

Memory address	0x4000	0x4001
Data	0x34	0x12

For the detailed mapping of peripheral registers, please refer to the related chapters about peripheral. The addressable memory space is divided into 8 main blocks, each of 512 Mbyte.

2.2.2 Internal SRAM

There is a 4 Kbytes of static SRAM. It can be accessed as bytes, half-words (16 bits) or full words (32 bits). The SRAM start address is 0x2000 0000.

2.2.3 Fast GPIO memory map

The Fast IO bridge provides a channel to access GPIO through AHB bus with more efficiency, each has 4 Kbyte address range.

Fast GPIO address range: 0x2008 0000 to 0x2008 0FFF.

2.2.4 Memory map

Table 2-2 is the AC7802x device memory map, which includes two different memory map tables based on different boot up configuration, such as embedded Flash memory boot up and SRAM boot up.

All the memory sections that are not allocated to on-chip memories and peripherals are considered "Reserved".

Table 2-2 Device memory map

0xE010 0000	Reserved		0xE010 0000	Reserved
0xE00F FFFF …… 0xE000 0000	Cortex™-M0+'s internal peripherals		0xE00F FFFF …… 0xE000 0000	Cortex™-M0+'s internal peripherals
0xDFFF FFFF …… 0x6000 0000	Reserved		0xDFFF FFFF …… 0x6000 0000	Reserved
0x5FFF FFFF …… 0x4010 0000	Reserved		0x5FFF FFFF …… 0x4010 0000	Reserved
0x400F FFFF …… 0x4000 0000	Peripheral APB Address		0x400F FFFF …… 0x4000 0000	Peripheral APB Address
0x3FFF FFFF …… 0x2008 1000	Reserved		0x3FFF FFFF …… 0x2008 1000	Reserved
0x2008 0FFF …… 0x2008 0000	AHB Fast IO		0x2008 0FFF …… 0x2008 0000	AHB Fast IO

0x2007 FFFF …… 0x2000 1000	Reserved		0x2007 FFFF …… 0x2000 1000	Reserved
0x2000 0FFF …… 0x2000 0000	AHB SRAM		0x2000 0FFF …… 0x2000 0000	AHB SRAM
0x1FFF FFFF …… 0x0804 3000	Reserved		0x1FFF FFFF …… 0x0804 3000	Reserved
0x0804 27FF …… 0x0804 2000	Reserved		0x0804 27FF …… 0x0804 2000	Reserved
0x0804 1FFF …… 0x0804 0800	Reserved		0x0804 1FFF …… 0x0804 0800	Reserved
0x0804 01FF …… 0x0804 0000	Option byte		0x0804 01FF …… 0x0804 0000	Option byte
0x0802 07FF …… 0x0802 0000	D-Flash memory		0x0802 07FF …… 0x0802 0000	D-Flash memory
0x0800 7FFF …… 0x0800 0000	P-Flash memory		0x0800 7FFF …… 0x0800 0000	P-Flash memory
0x07FF FFFF …… 0x0000 8000	Reserved		0x07FF FFFF …… 0x0000 1000	Reserved
0x0000 7FFF …… 0x0000 0000	Flash memory		0x0000 0FFF …… 0x0000 0000	AHB SRAM
Flash memory boot up			**SRAM boot up**	

2.2.5 Embedded Flash memory

The high-performance embedded Flash memory module has the following key features:

- Up to 32 Kbytes Flash memory.
- Memory organization: the Flash memory is organized as P-Flash memory, D-Flash memory and information block.
  - Size of P-Flash memory: up to 8 K × 32 bits of the main memory block, and each memory block is divided into 64 pages of 512 bytes.

- Size of D-Flash memory: 2 Kbytes, and each memory block is divided into 256 pages of 8 bytes.
- Size of information block: 512 bytes.

The Flash memory controller features:

- Flash Program/Erase operation.
- Read/Write protection.
- Erase and blank check.
- Cache controller is used to improve read efficiency, the maximum efficiency is zero wait.

2.2.6 Embedded Flash memory read

Flash memory instructions and data access are performed through the AHB bus.

2.2.7 Chip Model

The chip model information can be accessed by users through the eFlash read operation interface. The chip model information is stored in the continuous space (1*32Bit) of 0x40002050~0x40002053. Among them, the upper 8 bits are fixed to 0xFF, and the lower 24 bits are chip model information.

2.2.8 Chip UUID

The UUID has a total of 128 Bit, which is generated by a random number and can be used as the unique identification mark of the chip. It can be accessed through the eFlash read operation interface. The UUID information is stored in a continuous space(4*32Bit) from 0x40002054 to 0x40002063.

2.2.9 AHB to APB bridge

The AHB to APB bridge translates the AHB protocol to the APB protocol. Most peripherals are the APB interface. For the detailed address assignment of each peripheral, refer to Table 2-5.

2.2.10 Nested Vectored Interrupt Controller(NVIC)

Features

- 32 maskable interrupt channels (not including the 16 interrupt lines of Cortex™-M0+).
- 4 programmable priority levels (2 bits of interrupt priority are used).
- Low-latency exception and interrupt handling.
- Power management control.
- Implementation of System Control Registers.

The NVIC and the processor core interface are closely coupled, which enables low latency interrupt processing and efficient processing of late arriving interrupts.

All interrupts including the core exceptions are managed by the NVIC. For more information about exceptions and NVIC programming, refer to Chapter 5 Exceptions & Chapter 8 Nested Vectored Interrupt Controller of ARM Cortex™-M0+ Technical Reference Manual.

AC7802x interrupt vector is shown in Table 2-3.

Table 2-3 AC7802x interrupt table

Interrupt vector number	Priority	Type of priority	Acronym	Description
-15	-3	fixed	Reset	System reset
-14	-2	fixed	NMI	Non-maskable interrupt
-13	-1	fixed	HardFault	All class of fault
			Reserved	Reserved
			Reserved	Reserved
			Reserved	Reserved
			Reserved	Reserved
			Reserved	Reserved
			Reserved	Reserved
			Reserved	Reserved
-5	0	settable	SVCall	Exception caused by executing the System Service Call Instruction(SVC)
			Reserved	Reserved
			Reserved	Reserved
-2	0	settable	Pend SV	Pendable request for system service
-1	0	settable	SysTick	System tick timer
0	0	settable	PWDT0	PWDT0 interrupt
1	0	settable	PWM0	PWM 0 interrupt
2	0	settable	PWM1	PWM 1 interrupt
3	0	settable	PWM2	PWM 2 interrupt
4	0	settable	ACMP0	ACMP0 interrupt
5	0	settable	UART0	UART0 interrupt
6	0	settable	UART1	UART1 interrupt
			Reserved	Reserved
8	0	settable	WDG	Watch dog timer interrupt
9	0	settable	SPI0	SPI0 global interrupt
			Reserved	Reserved
11	0	settable	I2C0	I2C0 interrupt
			Reserved	Reserved
			Reserved	Reserved
			Reserved	Reserved

Interrupt vector number	Priority	Type of priority	Acronym	Description
			Reserved	Reserved
			Reserved	Reserved
17	0	settable	TIMER_CHANNEL0	Timer 0 interrupt
18	0	settable	TIMER_CHANNEL1	Timer 1 interrupt
19	0	settable	TIMER_CHANNEL2	Timer 2 interrupt
20	0	settable	TIMER_CHANNEL3	Timer 3 interrupt
21	0	settable	RTC	RTC interrupt
22	0	settable	LVD	LVD interrupt
23	0	settable	SPM	SPM interrupt
			Reserved	Reserved
25	0	settable	ADC0	ADC0 interrupt
26	0	settable	ECC_SRAM	ECC SRAM error detection interrupt
27	0	settable	EXTI0	EXTI 0 interrupt
28	0	settable	EXTI1	EXTI 1 interrupt
29	0	settable	EXTI2	EXTI 2 interrupt
30	0	settable	EXTI3_8	EXTI 3_8 interrupt
31	0	settable	EXTI9_15	EXTI 9_15 interrupt

2.2.11 Boot configuration

Four different boot modes can be selected through BOOT(PA6), PA1 and PA0 pins as shown in Table 2-4 .

Table 2-4 Boot configuration

PIN name	BOOT(PA6)	PA1	PA0
eFlash boot	0	x	x
SRAM boot	1	1	0

Note: x means do not care.

The values on the boot up configuration pins are latched on the 8th rising edge of 8 MHz clock after system reset. It is up to the user to set these pins to select the required boot mode, user should keep these pins stable before latched.

Due to its fixed memory map, the code section always starts from the address 0x0000 0000. The Cortex™-M0+ CPU always fetches the reset vector on the AHB-Lite bus, which implies to have the boot space available only in the code section (typically, boot from main Flash memory). AC7802x implements a special mechanism to be able to boot not only from the main Flash memory and system memory, but also from SRAM.

Depending on the selected boot mode, the embedded Flash memory are accessible as follows:

- **Boot from embedded Flash memory:** the embedded Flash memory is aliased in the boot memory space(0x0000 0000), but still is accessible from its original memory space (0x800 0000). In other words, the embedded Flash memory contents can be accessed starting from the address 0x0000 0000 or 0x800 0000.
- **Boot from the embedded SRAM:** the SRAM is aliased in the boot memory space (0x0000 0000), but still can be accessible from its original memory space (0x2000 0000).

2.3 Address assignment of Peripherals

Table 2-5 Address assignment of each peripheral

APB Memory Map	Base_Address	Size(byte)
CKGEN	0x40000000	4K
GPIO	0x40001000 / 0x20080000	4K
Embedded Flash Controller	0x40002000	4K
ADC0	0x40003000	4K
ACMP0	0x40005000	4K
SPM	0x40008000	1K
RTC	0x40008400	1K
WDG	0x4000B000	4K
SPI0	0x4000C000	4K
I2C0	0x4000E000	4K
Cortex™-M0+ controller	0x40010000	2K
TIMER	0x40011000	4K
PWM0	0x40013000	4K
PWM1	0x40014000	4K
PWM2	0x40015000	4K
CTU	0x40016000	4K
PWDT0	0x40017000	2K
UART0	0x40018000	4K
UART1	0x4001A000	4K

3 Reset

3.1 Features

POR reset:

- POR Reset: IC power-on reset.

System reset:

- External Reset: external pin reset from IC pad, low active.
- LVR Reset: low voltage detection reset, low active.
- Software Reset: Cortex™-M0+ software reset.
- ECC 2 Bit Error Reset: SRAM ECC detects a 2-bit error reset, disabled by default.
- Lock up Reset: Cortex™-M0+ lock up reset.
- Watchdog Reset Normal Mode: watch dog timer reset, active in normal mode.
- Watchdog Reset Stop Mode: watch dog timer reset, active in stop mode.
- XOSC Lost Reset: when an external XOSC clock is detected to be abnormal, a maskable system reset will be generated.

Each of the system reset sources has an associated bit in the status register RESET_STATUS.

3.2 Block diagram

The block diagram is described in Figure 3-1. Each Reset signal is high by default, and low-level reset signal triggers the system reset.

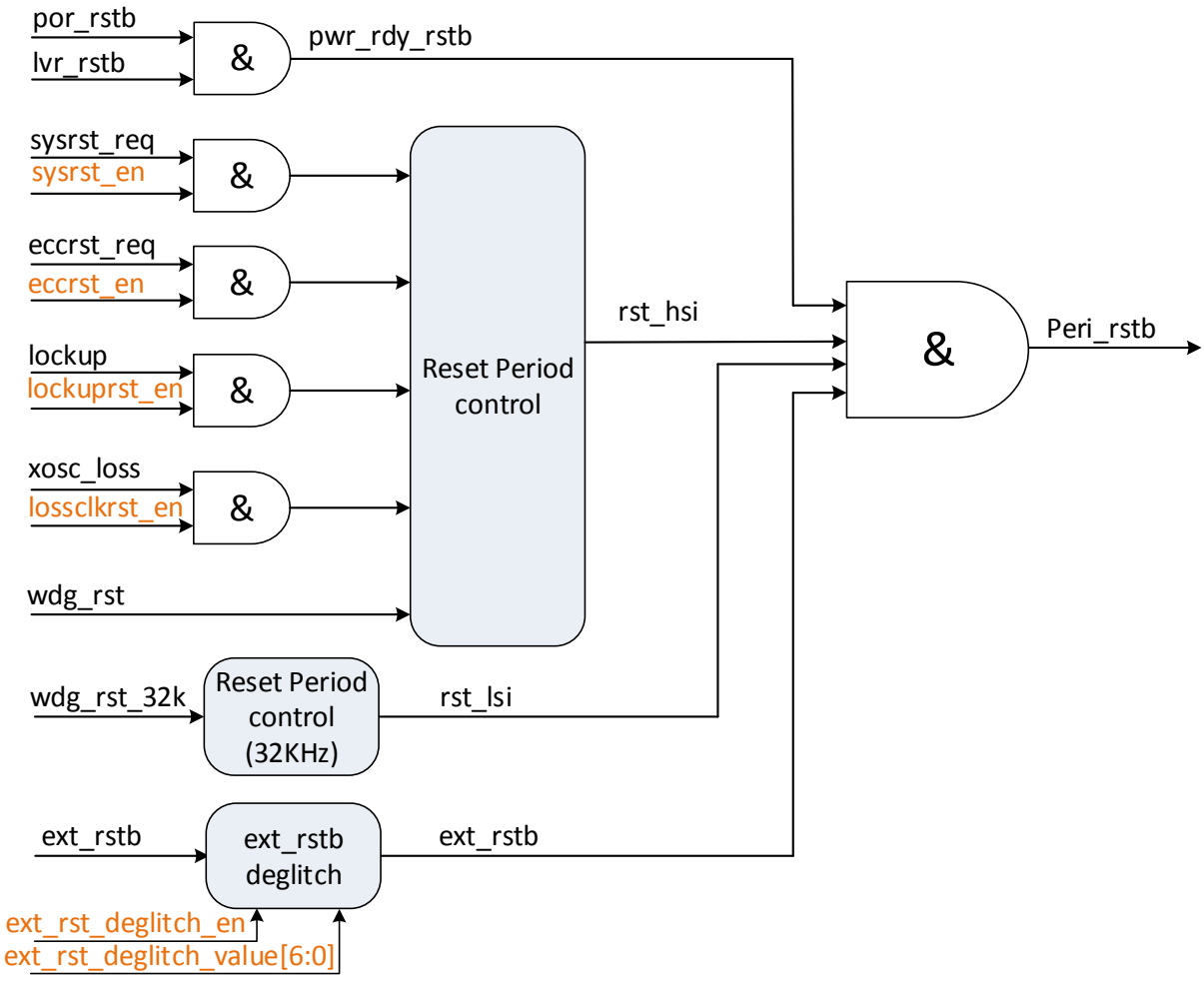

Figure 3-1 Reset block diagram

3.3 Function description

3.3.1 Power On Reset (POR)

The POR circuit causes a POR reset condition when:

1. Power is initially applied to the MCU
2. Supply voltage drops below the power-on reset voltage level (VPOR)

As the supply voltage rises, the LVR circuit holds the MCU in reset until the supply has risen above the LVR low threshold (V_{LVDL}), please refer to Table 6-2 LVD / POR / AVDD voltage reset specifications of ATC_AC7802x_Datasheet_EN.

3.3.2 System Reset

System reset begins with the on-chip regulator in full regulation and system clocking generation from an internal reference. When the processor exits reset, it performs the following:

- Read the initial value of SP (SP_main) from vector-table offset 0.
- Read the initial value of program counter (PC) from vector-table offset 4.
- The Link Register (LR) is set to 0xFFFF_FFFF.

3.3.2.1 External Reset

There is a dedicated pin in the MCU, and it is used to reset the whole MCU function and restart. As it is low active, it is suggested to add pull up function in external PCB to prevent noise. The external pin reset function can set the register EXT_RST_EN=0 to the mask state.

3.3.2.2 LVR Reset

This device includes a system to protect against low-voltage conditions in order to protect memory contents and control MCU system states during supply voltage variations. This system consists of a power-on reset (POR) circuit, and a LVR circuit with a user selectable trip voltage. LVR can be configured as different reset references, and the reset threshold high or low voltage is determined by the register LVR_THL.

For details, please refer to 5.1.1 DC characteristics of ATC_AC7802x_Datasheet_EN.

3.3.2.3 Software Reset

Setting RESET_CTRL[24] to 1 will enable Cortex™-M0+ software system reset. A software reset request is issued by the core to generate a system reset.

3.3.2.4 ECC 2 Bit Error Reset

Setting RESET_CTRL[23] to 1 will enable ECC 2 Bit Error Reset. When ECC detects a 2 Bit error, it will issue a system reset request to generate a system reset.

3.3.2.5 Lock up Reset

The Lock up gives immediate indication of seriously errant from kernel software. This is the result of the core being locked because of an unrecoverable exception following the activation of the processor's built in system state protection hardware.

When a lock up occurs, a reset can be automatically generated to recover the system.

3.3.2.6 Watchdog Reset

The watchdog timer(WDG) monitors the operation of the system by expecting periodic communication from the software. This communication is generally known as servicing (or refreshing) the watchdog.

If this periodic refreshing does not occur, the watchdog issues a system reset.

Please refer to 17 Watchdog Timer(WDG for detail.

3.3.2.7 XOSC monitor

XOSC monitor system can be activated by setting CKGEN_CTRL[16] to 1. In this case, the clock detector is enabled after the HSE oscillator startup delay, and disabled when this oscillator is stopped. If a failure is detected on the HSE oscillator clock, this oscillator is automatically disabled, XOSC loss status flag is active and NMI interrupt is generated to inform the software about the failure allowing the MCU to perform rescue operations, such as switching the internal clock. Or a reset signal is generated to reset the system after detecting the XOSC failure.

3.4 Register Definition

Table 3-1 Reset register map

CKGEN base address: 0x40000000

Address	Register name	Width	Description
CKGEN base address +0x0C	RESET_CTRL	32	chip reset control
CKGEN base address +0x10	RESET_STATUS	32	chip reset status

3.4.1 Chip Reset Control Register(RESET_CTRL)

Table 3-2 RESET_CTRL register

RESET_CTRL **Chip Reset Control** **Reset:0x0f418004**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name					EXT_RST_EN	XOSC_LOSS_RST_EN	CPU_LOCKUP_RST_EN	CPU_SYSRST_EN	ECC2_RST_EN							
Type					RW	RW	RW	RW	RW							
Reset					1	1	1	1	0							
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									EXT_RST_DEGLITCH_VALUE							EXT_RST_DEGLITCH_EN
Type									RW							RW
Reset									0	0	0	0	0	1	0	0

Field	Description
27 EXT_RST_EN	**External reset enable** 0 : Disabled 1 : Enabled
26 XOSC_LOSS_RST_EN	**XOSC loss reset enable** 0 : Disabled 1 : Enabled
25 CPU_LOCKUP_RST_EN	**CPU lock up reset enable** 0 : Disabled 1 : Enabled
24 CPU_SYSRST_EN	**CPU system reset enable** 0 : Disabled 1 : Enabled
23 ECC2_RST_EN	**SRAM ECC 2bit error reset enable** 0 : Disabled 1 : Enabled
22:8 Reserved	**Reserved**
7:1 EXT_RST_DEGLITCH_VALUE	**External reset deglitch value** With 32kHz as the period of the counting clock source. For example, when the value is 2, one cycle(below 31us) reset signal can be filtered out.
0 EXT_RST_DEGLITCH_EN	**External reset deglitch enable** 0 : Disabled 1 : Enabled

3.4.2 Chip Reset Status Register(RESET_STATUS)

Table 3-3 RESET_STATUS register

RESET_STATUS **Chip Reset Status** **Reset:0x80000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																CLEAR_RESET_STATUS
Type																RW
Reset																0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name							XOSC_LOSS_RST_STATUS		CPU_LOCKUP_RST_STATUS	CPU_SYSRESET_STATUS	WDG_RESET_STATUS	WDG_32K_RESET_STATUS	ECC2_RESET_STATUS	EXT_RESET_STATUS	LVR_RESET_STATUS	POR_RESET_STATUS
Type							RO		RO	RO	RO	RO	RO	RO	RO	RO
Reset							0		0	0	0	0	0	0	0	0

Field	Description
16 CLEAR_ RESET_STATUS	**Clear reset status** 0 : allow reset status to update 1 : clear all reset status
9 XOSC_LOSS_RST_STATUS	**XOSC Loss reset status** 0 : invalid 1 : valid
8 Reserved	**Reserved**
7 CPU_LOCKUP_ RST_STATUS	**CPU lock up reset status** 0 : invalid 1 : valid
6 CPU_SYSRESET_STATUS	**CPU system reset status** 0 : invalid 1 : valid
5 WDG_RESET_STATUS	**Watchdog normal reset status** 0 : invalid 1 : valid
4 WDG_32K_RESET_STATUS	**Watchdog reset status in low power mode** 0 : invalid 1 : valid
3 ECC2_RESET_STATUS	**SRAM ECC 2bit Error reset status** 0 : invalid 1 : valid

Field	Description
2 EXT_RESET_STATUS	**External pin reset status** 0 : invalid 1 : valid
1 LVR_RESET_STATUS	**LVR reset status** 0 : invalid 1 : valid
0 POR_RESET_STATUS	**POR reset status** 0 : invalid 1 : valid

4 Clock

4.1 Introduction

The clock control module provides clock source choices for the MCU.

4.2 Block diagram

4.2.1 Clock control diagram

This device contains the following on-chip clock sources:

- High speed internal RC(HSI): the internal RC OSC provides 32 MHz clock source.
- High speed external RC(HSE): the external OSC provides 8MHZ ~20MHz crystal oscillator or external clock input up to 20MHz.
- Low speed internal RC(LSI): the internal low speed RC OSC to provide 32kHz clock source.

Each peripheral has dedicated clock enable signal to control clock on/off, please refer to register control (4.3) to know the detail address.

Note:

- System clock up to 32MHz
- hclk(AHB) up to 32MHz
- pclk(APB) up to 16MHz. When hclk is 32MHz, APBCLK_DIV can be configured as 2.

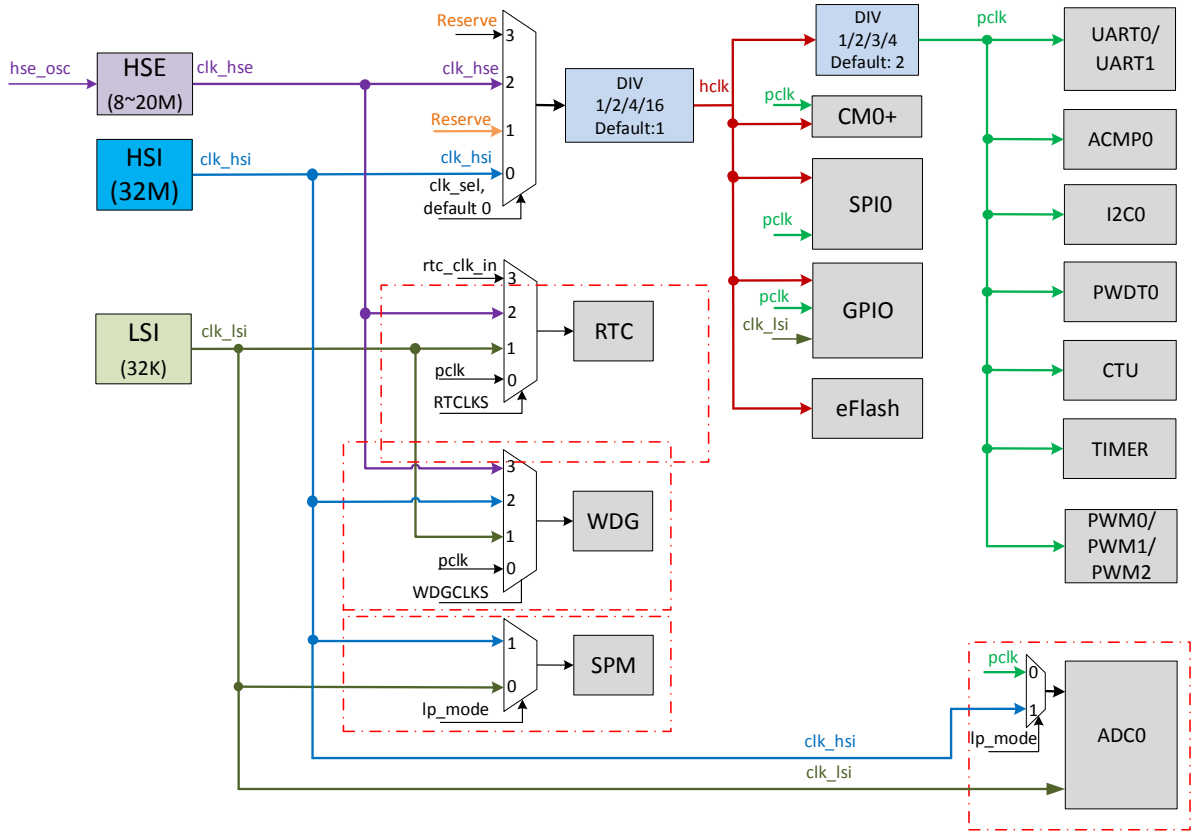

Figure 4-1 Clock control diagram

4.3 Register Definition

Table 4-1 Clock register map

CKGEN base address: 0x40000000

Address	Register name	Width	Description
CKGEN base address+0x00	CKGEN_CTRL	32	clock control
CKGEN base address+0x04	CKGEN_PERI_CLK_EN_0	32	peripheral clock enable control 0
CKGEN base address+0x08	CKGEN_PERI_CLK_EN_1	32	peripheral clock enable control 1
CKGEN base address+0x18	CKGEN_PERI_SFT_RST0	32	peripheral software reset control 0
CKGEN base address+0x1C	CKGEN_PERI_SFT_RST1	32	peripheral software reset control 1

4.3.1 Clock Control Register(CKGEN_CTRL)

Table 4-2 CKGEN_CTRL register

CKGEN_CTRL **Clock control** **Reset: 0x00000100**

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

Type																RW
Reset																0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name							APBCLK_DIV				SYSCLK_DIV				SYSCLK_SEL	
Type							RW				RW				RW	
Reset							0	1			0	0			0	0

Field	Description
31:17 Reserved	**Reserved**
16 XOSC_MON_EN	**XOSC monitor enable** 0 : disable 1 : enable
9:8 APBCLK_DIV	**APB clock divider by system clock** 00b : divided by 1 01b : divided by 2 10b : divided by 3 11b : divided by 4
5:4 SYSCLK_DIV	**System clock divider** 00b : divided by 1 01b : divided by 2 10b : divided by 3 11b : divided by 4
5:4 Reserved	**Reserved**
1:0 SYSCLK_SEL	**System clock source select** 00b : internal oscillator HSI 01b : reserved 10b : external oscillator HSE 11b : reserved

4.3.2 Peripheral Clock Enable Control 0(CKGEN_PERI_CLK_EN_0)

Table 4-3 CKGEN_PERI_CLK_EN_0 register

CKGEN_PERI_CLK_EN_0 **Peripheral clock enable control 0** **Reset: 0x02800001**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name							WDG_EN		GPIO_EN			RTC_EN	TIMER_EN			
Type							RW		RW			RW	RW			
Reset							1		1			0	0			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name			PWM2_EN	PWM1_EN	PWM0_EN	PWDT0_EN		I2C0_EN		SPI0_EN					UART1_EN	UART0_EN
Type			RW	RW	RW	RW		RW		RW					RW	RW
Reset			0	0	0	0		0		0					0	1

Note:

Before writing to the peripheral register, the clock of the corresponding peripheral needs to be enabled, otherwise Hardfault will occur.

Field	Description
31:26 Reserved	**Reserved**
25 WDG_EN	**WDG clock enable** 0 : clock disable 1 : clock enable
24 Reserved	**Reserved**
23 GPIO_EN	**GPIO AHB clock enable** 0 : clock disable 1 : clock enable
22:21 Reserved	**Reserved**
20 RTC_EN	**RTC clock enable** 0 : clock disable 1 : clock enable
19 TIMER_EN	**TIMER clock enable** 0 : clock disable

Field	Description
	1 : clock enable
18:14 Reserved	**Reserved**
13 PWM2_EN	**PWM2 timer clock enable** 0 : clock disable 1 : clock enable
12 PWM1_EN	**PWM1 timer clock enable** 0 : clock disable 1 : clock enable
11 PWM0_EN	**PWM0 timer clock enable** 0 : clock disable 1 : clock enable
10 PWDT0_EN	**PWDT0 clock enable** 0 : clock disable 1 : clock enable
9 Reserved	**Reserved**
8 I2C0_EN	**I2C0 clock enable** 0 : clock disable 1 : clock enable
7 Reserved	**Reserved**
6 SPI0_EN	**SPI0 clock enable** 0 : clock disable 1 : clock enable
5:2 Reserved	**Reserved**
1 UART1_EN	**UART1 clock enable** 0 : clock disable 1 : clock enable
0 UART0_EN	**UART0 clock enable** 0 : clock disable 1 : clock enable

4.3.3 Peripheral Clock Enable Control 1(CKGEN_PERI_CLK_EN_1)

Table 4-4 CKGEN_PERI_CLK_EN_1 register

CKGEN_PERI_CLK_EN_1 **Peripheral clock enable control 1** **Reset:0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name													ACMP0_EN	ADC0_EN	CTU_EN	
Type													RW	RW	RW	
Reset													0	0	0	

Note:
Before writing to the peripheral register, the clock of the corresponding peripheral needs to be enabled, otherwise Hardfault will occur.

Field	Description
31:4 Reserved	**Reserved**
3 ACMP0_EN	**ACMP0 clock enable** 0 : clock disable 1 : clock enable
2 ADC0_EN	**ADC0 clock enable** 0 : clock disable 1 : clock enable
1 CTU_EN	**CTU clock enable** 0 : clock disable 1 : clock enable
0 Reserved	**Reserved**

4.3.4 Peripheral Software Reset Control 0(CKGEN_PERI_SFT_RST0)

Table 4-5 CKGEN_PERI_SFT_RST0 register

CKGEN_PERI_SFT_RST0 **Peripheral software reset control 0** **Reset:0x02900001**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name							SRST_WDG		SRST_GPIO			SRST_RTC	SRST_TIMER			

Type							RW		RW			RW	RW			
Reset							1		1			1	0			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name			SRST_PWM2	SRST_PWM1	SRST_PWM0	SRST_PWDT0		SRST_I2C0		SRST_SPI0					SRST_UART1	SRST_UART0
Type			RW	RW	RW	RW		RW		RW					RW	RW
Reset			0	0	0	0		0		0					0	1

Field	Description
31:26 Reserved	**Reserved**
25 SRST_WDG	**Watch dog timer software reset** 0 : reset active 1 : reset inactive
24 Reserved	**Reserved**
23 SRST_GPIO	**GPIO AHB software reset** 0 : reset active 1 : reset inactive
22:21 Reserved	**Reserved**
20 SRST_RTC	**RTC software reset** 0 : reset active 1 : reset inactive
19 SRST_TIMER	**TIMER software reset** 0 : reset active 1 : reset inactive
18:14 Reserved	**Reserved**
13 SRST_PWM2	**PWM2 software reset** 0 : reset active 1 : reset inactive
12 SRST_PWM1	**PWM1 software reset** 0 : reset active 1 : reset inactive
11 SRST_PWM0	**PWM0 software reset** 0 : reset active 1 : reset inactive

Field	Description
10 SRST_PWDT0	**PWDT0 software reset** 0 : reset active 1 : reset inactive
9 Reserved	**Reserved**
8 SRST_I2C0	**I2C0 software reset** 0 : reset active 1 : reset inactive
7 Reserved	**Reserved**
6 SRST_SPI0	**SPI0 software reset** 0 : reset active 1 : reset inactive
5:2 Reserved	**Reserved**
1 SRST_UART1	**UART1 software reset** 0 : reset active 1 : reset inactive
0 SRST_UART0	**UART0 software reset** 0 : reset active 1 : reset inactive

4.3.5 Peripheral software reset control 1(CKGEN_PERI_SFT_RST1)

Table 4-6 CKGEN_PERI_SFT_RST1 register

CKGEN_PERI_SFT_RST1 **Peripheral software reset control 1** **Reset:0x00000010**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name												SRST_ANA_REG	SRST_ACMP0	SRST_ADC0	SRST_CTU	
Type												RW	RW	RW	RW	
Reset												1	0	0	0	

Field	Description
31:5 Reserved	**Reserved**
4 SRST_ANA_REG	**ANA register software reset** 0 : reset active 1 : reset inactive
3 SRST_ACMP0	**ACMP0 software reset** 0 : reset active 1 : reset inactive
2 SRST_ADC0	**ADC0 software reset** 0 : reset active 1 : reset inactive
1 SRST_CTU	**CTU software reset** 0 : reset active 1 : reset inactive
0 Reserved	**Reserved**

5 Power Modes

5.1 Introduction

This chapter describes the AC7802x chip power modes and functionality of the individual modules in these modes.

5.2 Functional description

The device supports Run, Sleep and Stop modes. I/O states are held in Run, Sleep and Stop modes.

- Run mode—CPU clocks can be run at full speed.
- Sleep mode—CPU enters into the sleep mode, system clock and bus clock are running.
- Stop mode— CPU enters into the deep sleep mode and some modules can wake up CPU.

5.3 Application note

5.3.1 Entering and exiting low power modes

1. Enable the wakeup source required using SPM_EN_PERIPH_WKUP.
2. Disable modules that has been enabled.
3. Call the WFI command to enter the low-power mode.
4. The processor exits the low-power mode via an interrupt.
5. Re-enable the module.

Note:

1. GPIO/RTC/WDG does not needs to be disabled.
2. In Stop mode, the shutdown of the UART module will not affect the wake-up.

5.3.2 Module operation in low power modes

The following table illustrates the functionality of each module while the chip is in each of the low-power modes. The standard behavior is shown with some exceptions.

Table 5-1 Module functionality in low-power modes

Module	Sleep mode	Stop mode
CM0+	Standby	Standby
SRAM	on	Standby
Embedded Flash	on	off

I2C	on	Standby [1]
SPI	on	Standby [2]
WDG	on	Optional on
PWDT	on	off
UART	on	Standby[3]
TIMER	on	off
PWM	on	off
CTU	on	off
RTC	on	Optional on
SPM	on	on
XOSC	on	off
HSI(32 MHz)	on	off
LSI(32 kHz)	on	on
GPIO	on	on [6]
ADC	on	Standby [5]
LVR	Optional on	Optional on
LVD	Optional on	Optional on [7]
ACMP	Optional on	Standby [8]
DAC	Optional on	Standby
T-sensor	Optional on	off
DIGLDO	on	Low power
FLHLDO	on	off
POR	on	on
BG	on	Optional on [9]

Note:

1. Supports address match wake-up in Stop mode.
2. Supports slave mode receive and wake-up in Stop mode.
3. Supports the edge wake-up in Stop mode (UART pin goes low directly to SPM).
4. Supports the edge wake-up in Stop mode, open the filter is optional.
5. Supports ADC wake-up in Stop mode.
6. The I/O state is maintained in Stop mode, supports all GPIO interrupt wake-ups.
7. Supports LVD Warning interrupt wake-up in Stop mode.
8. Supports ACMP setting voltage comparison wake-up in Stop mode.
9. Enables ADC wake-up, or enables LVR, LVD, BG is turned on, otherwise turn off BG in Stop mode.

Note:

- **on:** both Power and Clock of the module are provided normally.
- **Standby:** Power of the module is normal, and the Clock is turned off.
- **off:** both Power and Clock of the module are turned off.

6 System Power Management(SPM)

6.1 Introduction

System Power Management (SPM) provides software developers with flexible system management, including sleep/wake-up functions, power domain management and power consumption control of each module.

6.2 Features

- Supports power management in Stop mode.

6.3 Application notes

6.3.1 SPM power control program guide

Stop mode is supported in AC7802x.

For the working status and the wake-up source of each module in Stop mode, please refer to Table 5-1.

The WFI instruction calls the Stop mode for the chip, and the processor exits the low power mode via an interrupt.

Program sequence:

1. Configure the wake-up source to work normally and generate the interrupt normally.
2. Set the wake-up source: SPM_EN_PERIPH_WKUP.
3. Enable the SPM power control: PWR_EN.
4. Execute the WFI command.

6.3.2 XOSC power control

XOSC is off by default. When needed, XOSC can be powered on/off by configuring the SPM_PWR_MGR_CFG1 register.

SPM register SPM_PWR_MGR_CFG1:

- **XOSC_HSEON**: control external high-speed clock enable.
- **XOSC_HSEBYP**: external high-speed clock bypass.

When the corresponding bit is set to 1'b1, SPM will power XOSC on following the power on sequence and it may take some time. So you should be waiting for XOSC power on complete and clock ready before using it.

XOSC power on status can be determined by reading SPM register : SPM_PWR_MGR_CFG1.

- **XOSC_RDY:** external high-speed clock ready flag.

When the device wakes up from the Stop mode, SPM will keep XOSC on or off, the same as state before sleep.

6.4 Register Definition

Table 6-1 SPM register map

SPM base address: 0x40008000

Address	Register name	Width	Description
SPM base address+0x00	SPM_PWR_MGR_CFG0	32	Power Manager Configuration 0
SPM base address+0x04	SPM_PWR_MGR_CFG1	32	Power Manager Configuration 1
SPM base address+0x0C	SPM_PERIPH_SLEEP_ACK_STATUS	32	Peripheral Sleep Ack Status
SPM base address+ 0x10	SPM_EN_PERIPH_SLEEP_ACK	32	Peripheral Sleep Ack Enable
SPM base address+ 0x14	SPM_EN_PERIPH_WKUP	32	Peripheral Wakeup Enable
SPM base address+ 0x1C	SPM_WAKEUP_IRQ_STATUS	32	SPM Wakeup IRQ Flags Status

6.4.1 Power Manager Configuration 0 (SPM_PWR_MGR_CFG0)

Table 6-2 SPM_PWR_MGR_CFG0 register

SPM_PWR_MGR_CFG0 Power Manager Configuration 0 Reset:0x00000018

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									EN_IO_SUS			EN_LVR	EN_DPWRLVR	EN_LVD	EN_FAST_BOOT	PWR_EN
Type									RW			RW	RW	RW	RW	RW
Reset									0			1	1	0	0	0

Field	Description
31:8 Reserved	**Reserved**
7 EN_IO_SUS	**Disable IO in Stop mode** 0: I/O status hold when enter stop mode 1: I/O suspend when enter stop mode
6:5 Reserved	**Reserved**

4 EN_LVR	**Enable lower voltage detect reset** 0: disable 1: enable Check the chip VCC voltage, if VCC is under voltage, trigger LVR reset.
3 EN_DPWRLVR	**Enable LDO low voltage detection** 0: disable 1: enable Check the internal LDO voltage. If the LDO is under voltage, the LVR reset will also be triggered.
2 EN_LVD	**Enable low voltage detect warning** 0: disable 1: enable Check the chip VCC voltage, if VCC is under voltage, trigger the LVD interrupt warning.
1 EN_FAST_BOOT	**Enable fast boot mode** 0: disable 1: enable Fast boot mode: chip will stop the sleep sequence and wake up immediately when receiving a wakeup interrupt during wake-up.
0 PWR_EN	**SPM power control enable** 0: disable 1: enable SPM power control

6.4.2 Power Manager Configuration 1(SPM_PWR_MGR_CFG1)

Table 6-3 SPM_PWR_MGR_CFG1 register

SPM_PWR_MGR_CFG1 Power Manager Configuration 1 Reset:0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	XOSC_RDY		XOSC_HSEON	XOSC_HSEBYP												
Type	RO		RW	RW												
Reset	0		0	0												
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name													LVD_THL	LVR_THL		
Type													RW	RW		
Reset													0	0		

Field	**Description**

31 XOSC_RDY	**XOSC clock ready flag** 0: unready 1: ready
30 Reserved	**Reserved**
29 XOSC_HSEON	**External high-speed clock enable** 0: disable XOSC 1: enable XOSC
28 XOSC_HSEBYP	**External high-speed clock bypass** 0: disable bypassing the oscillator with an external clock 1: bypass the oscillator with an external clock(external clock input)
27:4 Reserved	**Reserved**
3 LVD_THL	**LVD voltage threshold** 0: threshold low 2.9 V 1: threshold high 4.6 V
2 LVR_THL	**LVR voltage threshold** 0: threshold low 2.6 V 1: threshold high 4.3 V
1:0 Reserved	**Reserved**

6.4.3 Peripheral Sleep Ack Status(SPM_PERIPH_SLEEP_ACK_STATUS)

Table 6-4 SPM_PERIPH_SLEEP_ACK_STATUS register

SPM_PERIPH_SLEEP_ACK_STATUS Peripheral Sleep Ack Status **Reset:0xffffffff**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name														EFLASH		ADC0
Type														RO		RO
Reset														1		1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name						UART1	UART0					SPI0		I2C0		ACMP0
Type						RO	RO					RO		RO		RO
Reset						1	1					1		1		1

Field	**Description**
31:19 Reserved	**Reserved**

18 EFLASH	**eFLASH idle status** 0: busy 1: idle
17 Reserved	**Reserved**
16 ADC0	**ADC0 sleep ack status** 0: not ACK 1: ACK
15:11 Reserved	**Reserved**
10 UART2	**UART1 sleep ack status** 0: not ACK 1: ACK
10 UART1	**UART1 sleep ack status** 0: not ACK 1: ACK
9 UART0	**UART0 sleep ack status** 0: not ACK 1: ACK
8:5 Reserved	**Reserved**
4 SPI0	**SPI0 sleep ack status** 0: not ACK 1: ACK
3 Reserved	**Reserved**
2 I2C0	**I2C0 sleep ack status** 0: not ACK 1: ACK
1 Reserved	**Reserved**
0 ACMP0	**ACMP0 sleep ack status** 0: not ACK 1: ACK

6.4.4 Peripheral Sleep Ack Enable(SPM_EN_PERIPH_SLEEP_ACK)

Table 6-5 SPM_EN_PERIPH_SLEEP_ACK register

SPM_EN_PERIPH_SLEEP_ACK Peripheral Sleep Ack Enable **Reset:0x00050615**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name														EFLASH		ADC0
Type														RW		RW
Reset														1		1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name						UART1	UART0					SPI0		I2C0		ACMP0
Type						RW	RW					RW		RW		RW
Reset						1	1					1		1		1

Field	Description
31:19 Reserved	**Reserved**
18 EFLASH	**Enable eFlash Sleep ACK Waiting** 0: disable 1: enable
17 Reserved	**Reserved**
16 ADC0	**Enable ADC0 Sleep ACK Waiting** 0: disable 1: enable
15:11 Reserved	**Reserved**
10 UART1	**Enable UART1 Sleep ACK Waiting** 0: disable 1: enable
9 UART0	**Enable UART0 Sleep ACK Waiting** 0: disable 1: enable
8:5 Reserved	**Reserved**
4 SPI0	**Enable SPI0 Sleep ACK Waiting** 0: disable 1: enable
3 Reserved	**Reserved**
2 I2C0	**Enable I2C0 Sleep ACK Waiting** 0: disable 1: enable
1 Reserved	**Reserved**

0 ACMP0	**Enable ACMP0 Sleep ACK Waiting** 0: disable 1: enable

6.4.5 Peripheral Wakeup Enable(SPM_EN_PERIPH_WKUP)

Table 6-6 SPM_EN_PERIPH_WKUP register

SPM_EN_PERIPH_WKUP Peripheral Wakeup Enable **Reset:0x00028000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name													LVD	NMI	GPIO	ADC0
Type													RW	RW	RW	RW
Reset													0	0	1	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RTC					UART1	UART0					SPI0		I2C0		ACMP0
Type	RW					RW	RW					RW		RW		RW
Reset	1					0	0					0		0		0

Field	Description
31:20 Reserved	**Reserved**
19 LVD	**Enable LVD warning interrupt** 0: disable 1: enable(It needs to be enabled when generating interrupt or wakeup)
18 NMI	**Enable NMI wakeup** 0: disable 1: enable(It needs to be enabled when generating interrupt or wakeup)
17 GPIO	**Enable GPIO wakeup** 0: disable 1: enable
16 ADC0	**Enable ADC0 wakeup** 0: disable 1: enable
15 RTC	**Enable RTC wakeup** 0: disable 1: enable
14:11 Reserved	**Reserved**

10 UART1	**Enable UART1 wakeup** 0: disable 1: enable
9 UART0	**Enable UART0 wakeup** 0: disable 1: enable
8:5 Reserved	**Reserved**
4 SPI0	**Enable SPI0 wakeup** 0: disable 1: enable
3 Reserved	**Reserved**
2 I2C0	**Enable I2C0 wakeup** 0: disable 1: enable
1 Reserved	**Reserved**
0 ACMP0	**Enable ACMP0 wakeup** 0: disable 1: enable

6.4.6 SPM Wakeup IRQ Flags Status(SPM_WAKEUP_IRQ_STATUS)

Table 6-7 SPM_WAKEUP_IRQ_STATUS register

SPM_WAKEUP_IRQ_STATUS **SPM Wakeup IRQ Flags Status** **Reset:0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name												OVER_COUNT	LVD	NMI	GPIO	ADC0
Type												R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
Reset												0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RTC					UART1	UART0					SPI0		I2C0		ACMP0
Type	R/W1C					R/W1C	R/W1C					R/W1C		R/W1C		R/W1C
Reset	0					0	0					0		0		0

Field	Description
31:21 Reserved	**Reserved**
20 OVER_COUNT	**SPM over count wake-up flag** 0: invalid 1: before entering Stop mode, the SPM will wait for all peripheral ACK at specified period. If there is a peripheral NO ACK, the OVER_COUNT is set, and exit Stop mode, resulting in an SPM interrupt. **Note: write "1" to clear this bit.**
19 LVD	**LVD interrupt or wake-up flag** 0 : invalid 1 : valid **Note: write "1" to clear this bit.**
18 NMI	**NMI interrupt or wake-up flag** 0 : invalid 1 : valid **Note: write "1" to clear this bit**
17 GPIO	**GPIO wake-up flag** 0 : invalid 1 : valid **Note: write "1" to clear this bit**
16 ADC	**ADC wake-up flag** 0 : invalid 1 : valid **Note: write "1" to clear this bit**
15 RTC	**RTC wake-up flag** 0 : invalid 1 : valid **Note: write "1" to clear this bit**
14:11 Reserved	**Reserved**

10 UART1	**UART1 wake-up flag** 0 : invalid 1 : valid **Note: write "1" to clear this bit**
9 UART0	**UART0 wake-up flag** 0 : invalid 1 : valid **Note: write "1" to clear this bit**
8:5 Reserved	**Reserved**
4 SPI0	**SPI0 wake-up flag** 0 : invalid 1 : valid **Note: write "1" to clear this bit**
3 Reserved	**Reserved**
2 I2C0	**I2C0 wake-up flag** 0 : invalid 1 : valid **Note: write "1" to clear this bit**
1 Reserved	**Reserved**
0 ACMP0	**ACMP0 wake-up flag** 0 : invalid 1 : valid **Note: write "1" to clear this bit.**

7 Universal Asynchronous Receiver/Transmitter(UART)

7.1 Introduction

The UART (Universal Asynchronous Receiver/Transmitter) is a basic protocol for serial communication. It operates to achieve many functions mainly by transmitter and receiver. The main function is composed of 2 bits of the register: LINEN, ILEN, as shown in Table 7-1.

Table 7-1 UART function classification and configuration

Functions	LINEN
BASIC UART	0
LIN	1

Note

1. Only UART0 supports software LIN.
2. LIN mode supports only 8-bit data format and 16 times oversample. Meanwhile, if auto baud rate function is enabled(LABAUDEN=1), the synchronous field data(0x55) will be discarded.

7.2 Features

- Full duplex, standard NRZ format.
- Programmable baud rate (16-bit divisor).
  - Supports the baud rate range of transmission/reception: 600 bps~2 Mbps, the baud rate error is less than 1%
- Interrupt or polling is used to check these status flags.
  - Transmitter data register is empty and transmission is completed.
  - Receiver data register full.
  - Receive overflow error, frame error, parity error.
  - Idle line detect.
  - Break detect supporting LIN and optional 10/11 bits LIN break character detection.
  - Active edge detects for wake up from Stop mode.
- Programmable 7-bit, 8-bit or 9-bit data length, 1-bit or 2-bits stop bits, hardware automatically generates parity bit.
- Selectable transmitter output and receiver input polarity.

- Idle line detection.
- 13-28 bits break character generation.
- For polling mode, baud rate supports up to 4M bps. For interrupt mode, baud rate supports up to 2M bps (affected by user interrupt callback function execution efficiency)

7.3 Block diagram

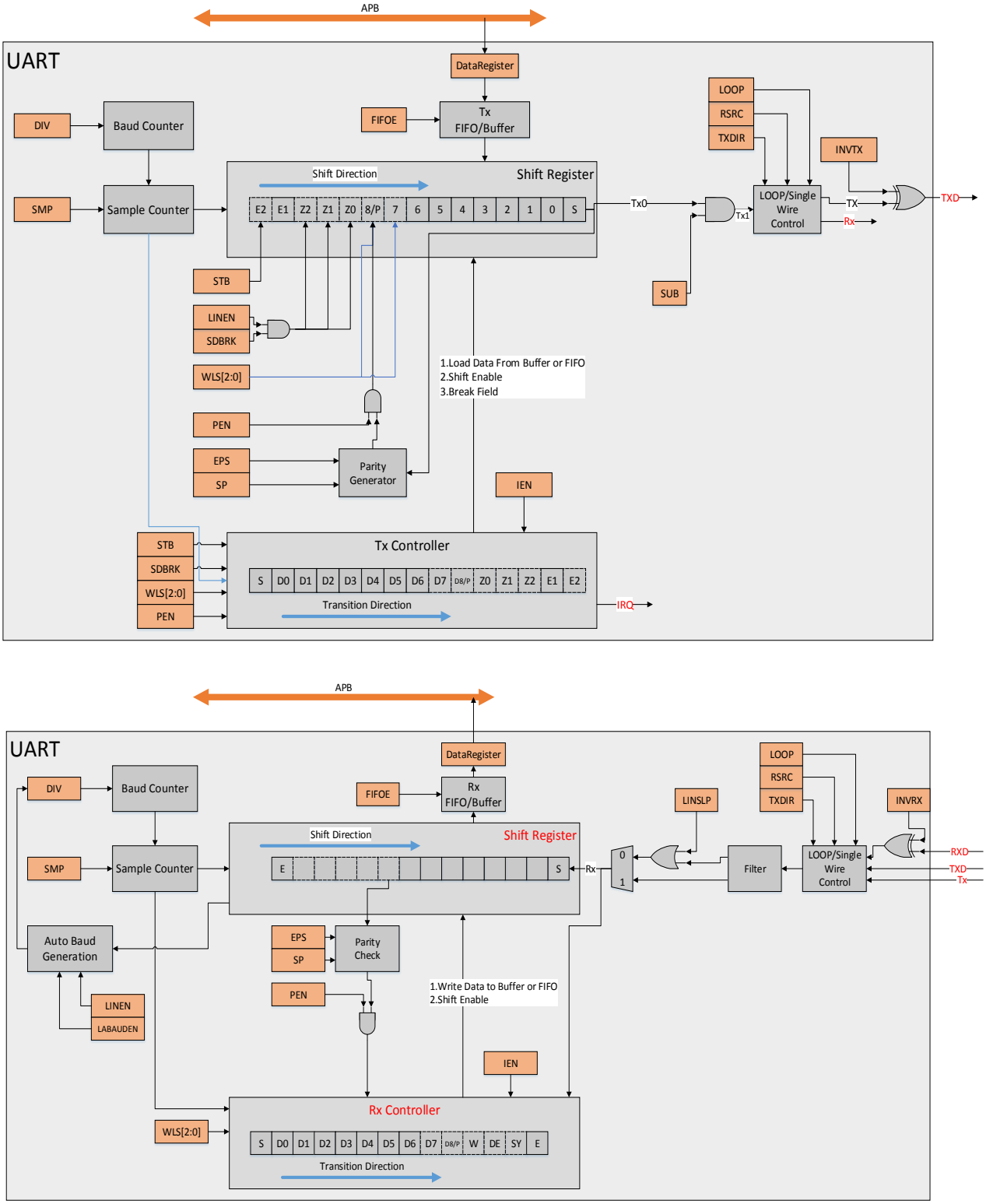

Figure 7-1 UART block diagram

7.4 Function description

Basic UART is used to transmit and receive the serial data bit by bit. In Figure 7-2 and Figure 7-3, it illustrates the full data bits including start bit, data bits, parity bit, stop bits and guard time. But the bit6, bit7, bit8, bit9, parity, stop2 bit and guard time bit can be configured by user described in UART_LCR0 and UART_LCR1 in detail. One bit corresponds to one-bit time controlled by baud rate.

There exist many statuses for UART transmitting and receiving. User had better know when the statuses are generated in the transmitting or receiving process. In this way, user can use the UART function better. The status bits THRE and TC just occur in transmitting process illustrated in Figure 7-2. During the initialization condition after global reset or power on, THRE and TC turn to 1 just after TXEN is set to 1. But in the process of transmitting, THRE turns to 1 just after start bit and TC turns to 1 just after the last bit, such as the guard time if GUARDEN=1.

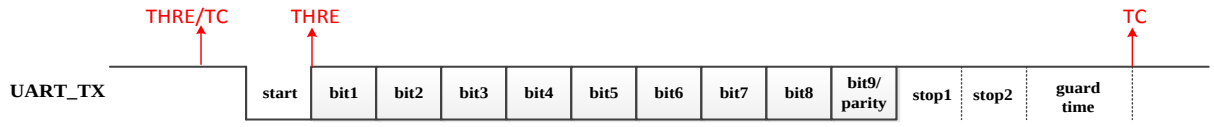

Figure 7-2 UART transmitter flow

The status bit DR, OE, PE, FE, BI and NE are set to 1 just after the stop1 bit if related events occur during receiving process as illustrated in Figure 7-3.

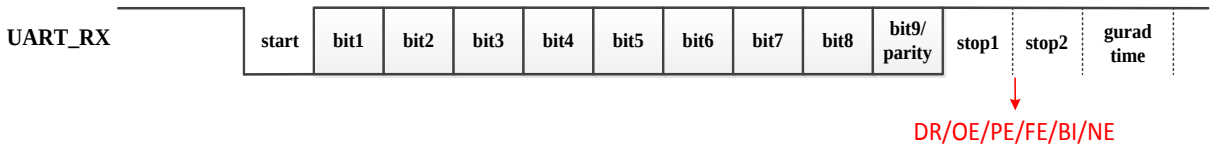

Figure 7-3 UART receiver flow

It is necessary to note that the statuses of PE, FE and NE just aim at the current receiving data byte and will be cleared automatically just when the next data is received completely. For other statuses, they hold the value all the time until they are cleared by reading or writing 1.

7.4.1 Noise detection

For NE status, it is generated during receiving process when the noise exists in UART_RX signal. In order to detect the noise, UART_RX is sampled three times at the middle position as illustrated in Figure 7-4.

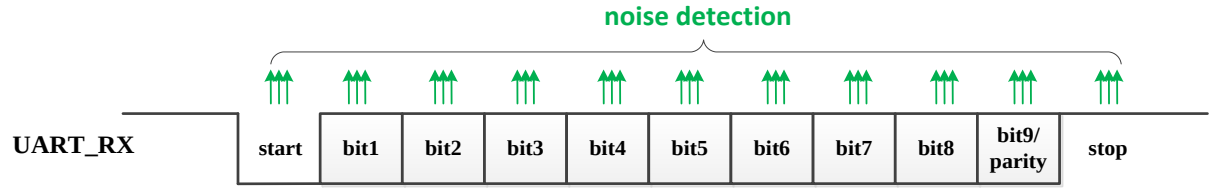

Figure 7-4 UART noise detection

To explain clearly and conveniently, the value of three times sample is called SM1, SM2 and SM3 respectively. If there are more than two '1' among SM1, SM2 and SM3 as start bit, the start bit is invalid and the receiver is reset to receive again. Other than this case for start bit, if the values of SM1, SM2 and SM3 are different with each other, the noise is detected with the NE status setting to 1.

7.4.2 Baud rate description

UART Baud rate accuracy is determined by many aspects including UART clock, oversample time and so on. So, some specific and too high baud rates are not realized or realized with large error. Table 7-2 and Table 7-3 describe the typical baud rate configurations at different system clocks and corresponding errors.

Table 7-2 Typical baud rate and error rate@bclock=32MHz

No.	Theoretical value(bps)	Practical value(bps)	DIV_MAN[15:0]	DIV_FRAC[4:0]	Oversample times	Error rate
1	600	599.998125	3333	11	16	0.000%
2	2400	2399.97	833	11	16	-0.001%
3	9600	9599.520024	208	11	16	-0.005%
4	19200	19201.92019	104	5	16	0.010%
5	57600	57605.76058	34	23	16	0.010%
6	115200	115107.9137	17	12	16	-0.080%
7	230400	230215.8273	8	22	16	-0.080%
8	460800	460431.6547	4	11	16	-0.080%
9	921600	927536.2319	2	5	16	0.644%
10	1843200	1828571.429	1	3	16	-0.794%

Table 7-3 Typical baud rate and error rate@bclock=8MHz

No.	Theoretical Value(bps)	Practical Value(bps)	DIV_MANTI[15:0]	DIV_FRAC[4:0]	Oversample times	Error rate
1	600	599.9925001	833	11	16	0.001%
2	2400	2399.880006	208	11	16	0.005%
3	9600	9598.080384	52	3	16	0.020%
4	19200	19207.68307	26	1	16	0.040%
5	57600	57553.95683	8	22	16	0.080%
6	115200	115107.9137	4	11	16	0.080%
7	230400	231884.058	2	5	16	0.644%
8	460800	457142.8571	1	3	16	0.794%
9	921600	914285.7143	1	3	8	0.794%

7.4.3 LIN function

The UART LIN is just a software LIN which transmits the break field, synchronous field and data respectively controlled by user as Figure 7-5. User can learn more details in the newest LIN protocol. Figure 7-5 just illustrates a basic frame condition. User should pay more attention to the UART_LINCR register in addition to the basic UART registers. In UART module, there exists a hardware unit for LIN detection and it is enabled when LINEN is configured to 1. It should be noted that only UART0 supports LIN function.

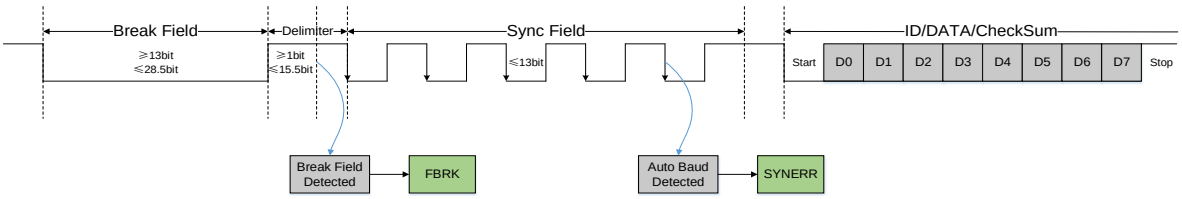

Figure 7-5 LIN frame flow

The UART-LIN frame flow is shown as Figure 7-5, the receiving process follows the steps:

1. When UART receives 10 (LBRKDL=0) or 11 (LBRKDL=1) bit zero, UART LIN detection logics unit treats it as a valid break field for LIN frame and the LIN break flag FBRK is set to 1 by hardware to indicate a valid LIN break field has occurred. It should be noted that the LIN sync break is not a data and is not stored into the RX data register or FIFO.
2. After delimiter bits period in Figure 7-5, sync field namely 0x55 acts as a normal data for receiving. If LABAUDEN is configured to 1, auto baud rate detection is in operation during synchronous field and the auto baud rate detection operation is finished just after the fifth falling edge illustrated in Figure 7-5. But the data 0x55 will not be stored into RX data register or FIFO. If LABAUDEN is configured to 0, auto baud rate detection is not performed and the data 0x55 is stored into the RX data register or FIFO. The baud rate synchronization function supports -50%~+100% deviation of the master baud rate, otherwise a sync field error SYNERR may occur.
3. After sync field, the data streams are received by UART without difference.

To avoid the module halting when exception condition occurs, time out mechanism is brought in, as illustrated in Figure 7-5.

1. When the sync break field received by the LIN slave exceeds 28.5 bits, it is considered to be an invalid synch break field.
2. More than 15 bits time for break delimiter can lead the module to reset the receiver and LIN detection logics.
3. If the sync field fails, the hardware sets the SYNERR bit of UART_LCR1. If the SYNERRIE bit of UART_LINCR is enabled, when the synchronization fails, the hardware generates a sync field error interrupt.

For the UART-LIN transmitting process, refer to the following steps:

1. As software LIN, how to transmit the sync break field is the key procedure. When user wants to transmit the sync break field, user should check the UART_LSR0[THRE] status. The length of the sync break field sent depends on the configuration of the register UART_BRKLH.

   The value of register UART_BRKLH is 0-15, corresponding to the break field of the 13-28 bit. At this time, if the value of UART_LSR0[THRE] is 1, the user can send a break field by writing 1 to the register UART_LINCR[SDBRK]. The UART_LINCR[SDBRK] will be cleared automatically by hardware.

2. And then the sync field(0x55) and other following data can be written to the UART_RBR/THR data register and sent as normal data.

UART-LIN sleep/wakeup function:

1. User can set UART_LINCR[LINSLP] to 1, so that LIN is in sleep state, and data cannot be received in this state;
2. When LIN is in sleep state, UART RX receives a low-level wake-up signal longer than 150μs, and LIN will exit the sleep state. If UART_LINCR[LINWAKIE] is enabled, the UART_LSR1[LINWAK] interrupt will be triggered, and then data can be received normally. User can also set LINSLP to 0 to exit the sleep state.

7.4.4 Two power mode

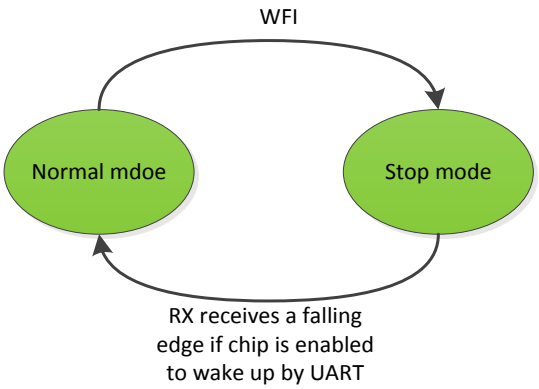

Figure 7-6 Run mode and Stop mode condition

The UART status in two power modes designed for the chip is introduced here. As the Figure 7-6 illustrates, user can execute the WFI instruction to make the chip go into the Stop mode, in which the chip power consumption will be much less obviously and the UART module will power down and be reset by default. The register configuration of the UART module is maintained and does not need to be reconfigured after awakening.

In Stop mode, the chip can wake up to normal mode when UART receives a falling edge if the chip is enabled to wake up by UART. Because of the time required for the wake-up flow, UART can normally receive data just after the total time of 5 ms. In details, TX1 can send 0xFF to act as a

falling edge and actually other data is also ok. Specially, data will be lost in RX2 if TX1 send some data to RX2 during the wake up flow.

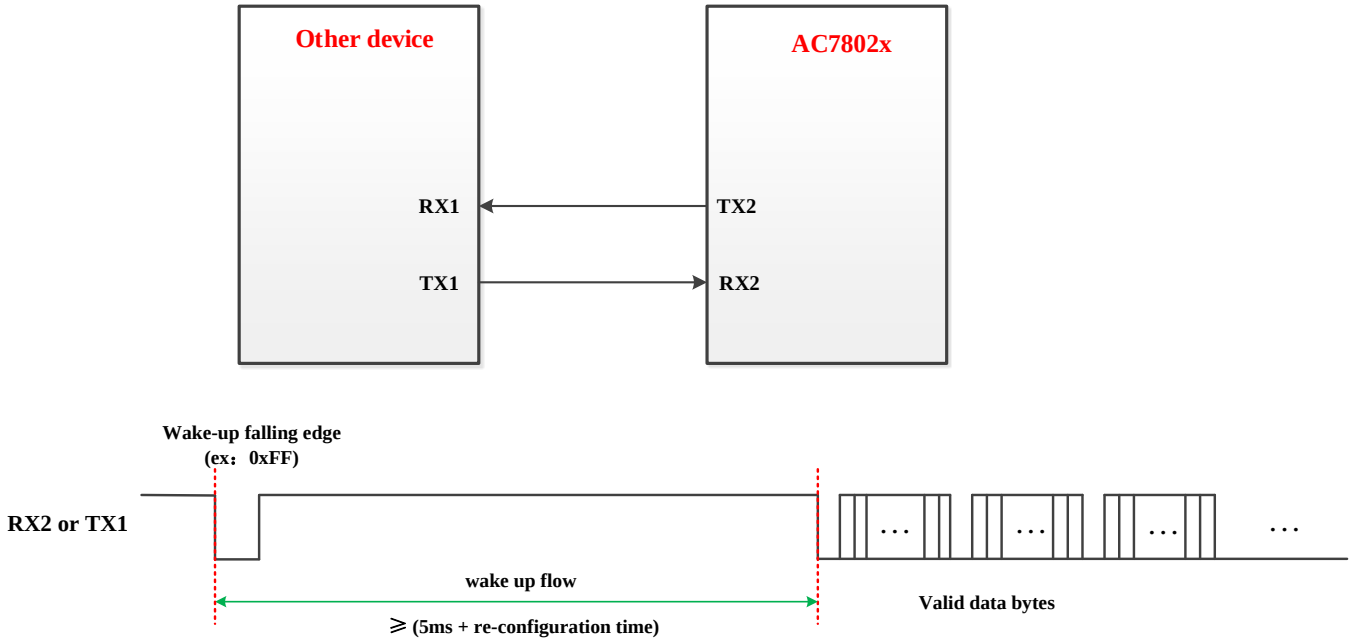

Figure 7-7 Typical flow for waking up the chip by UART

7.5 Application note

7.5.1 Baud rate configuration note

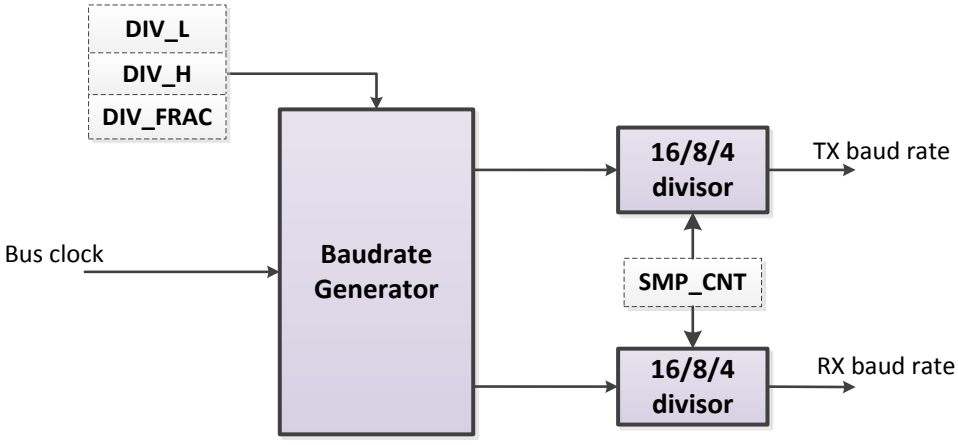

Figure 7-8 Diagram for baud rate generator

As illustrated in the Figure 7-8, the baud rate related configuration registers are as follows:

UART_SMP_CNT/UART_DIV_H/UART_DIV_L/UART_DIV_FRAC. And the detailed information is described in UART register map. For configuration, the formula below is used for baud rate configuration.

$$Baudrate = \frac{f_{clk}}{DIV * SMP\_CNT}$$

Note: DIV = {UART_DIV_H, UART_DIV_L} ; SMP_CNT = UART_SMP_CNT.

For example:

If user wants to get baud rate 230400 bps at 8 times sample mode with 24MHz bus frequency, so we can get the DIV configuration as follows. So, **UART_DIV_L=13, UART_DIV_H=0, UART_DIV_FRAC=32 * 0.0208=1.**

$$\text{DIV} = \frac{24000000}{\text{Baudrate*SMP CNT}} = \frac{24000000}{230400*8} = 13.0208$$

Combined with the example above, UART_DIV_FRAC[4:0] can be explained clearly. In the expression above, DIV is not an integer. If user removes the fractional part, the accuracy of baud rate will decrease. Especially in high baud rate condition, the discard of DIV fractional part may reduce the accuracy to a low level so that the normal data transmission will make a mistake.

In order to keep high accuracy, UART_DIV_FRAC[4:0] is configured as the DIV fractional part. Because of 5 bits width, UART_DIV_FRAC ranges from 0 to 31. So, 32 multiplied by the DIV fractional part generates the configuration value for UART_DIV_FRAC[4:0].

7.5.2 UART configuration note

Configuration steps:

(1) TX/RX data storage mode: UART_FCR.

(2) Baud rate: UART_DIV_L/UART_DIV_H/ UART_DIV_FRAC/UART_SMP_CNT

(3) Data format: UART_LCR0/UART_LCR1

Note:
SUB bit must be taken care of.

(4) Function configuration: UART_LINCR/UART_IDLE and so on.

Note:
This step is an option for different function.

(5) Interrupt enable: UART_IER

(6) Transmitter and receiver enable: UART_LCR1[TXEN] / UART_LCR1[RXEN].

(7) Transmitting or receiving data: UART_THR/UART_RBR

Note:
This step is in normal transmitting or receiving data process actually.

Notes:

1. For LIN function, data format must be configured as 8 bit with no parity check with 16 times oversample.
2. For LIN function, the sync field data(0x55) will be received and stored into FIFO or RX register when LABAUDEN=0. The sync field data(0x55) will be received and not stored into FIFO or RX register when LABAUDEN=1.

7.6 Register Definition

Table 7-4 UART register map

UART0 base address: 0x40018000
UART1 base address: 0x4001A000

Address	Register name	Width	Description
UARTx base address+0x00	UART_RBR/THR	32	TX holding register /RX buffer register
UARTx base address+0x04	UART_DIV_L	32	Divisor Low 8 bits
UARTx base address+0x08	UART_DIV_H	32	Divisor High 8 bits
UARTx base address+0x0C	UART_LCR0	32	UART supplementary control register 0
UARTx base address+0x10	UART_LCR1	32	UART supplementary control register 1
UARTx base address+0x14	UART_FCR	32	FIFO control register
UARTx base address+0x1C	UART_IER	32	Interrupt enable register
UARTx base address+0x20	UART_LSR0	32	Status register0
UARTx base address+0x24	UART_LSR1	32	Status register 1
UARTx base addres+0x28	UART_SMP_CNT	32	UART sample counter register
UARTx base address+0x34	UART_GUARD	32	Guard time added register
UARTx base address+0x3C	UART_SLEEP_EN	32	Sleep enable register
UARTx base address+0x44	UART_DIV_FRAC	32	Fractional divider register
UARTx base address+0x58	UART_IDLE	32	Idle interrupt enable register
UARTx base address+0x5C	UART_LINCR	32	Software LIN control register **Note: UART1 does not support LIN, no such register.**
UARTx base address+0x60	UART_BRKLH	32	Software LIN sync break length control register **Note: UART1 does not support LIN, no such register.**

7.6.1 RX/TX Data Register(UART_RBR/THR)

Table 7-5 UART_RBR/THR register

UART_RBR/THR **RX/TX data register** **Reset：0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name								RBR/THR								
Type								RW								
Reset								0								

Field	Description
8: 0 RBR/THR	**RX/TX Data register** The received data can be read by accessing this register and the transmitting data can be written into this register. The data width is no longer than 9 bits.

7.6.2 Divisor Low 8 Bits Register(UART_DIV_L)

Table 7-6 UART_DIV_L register

UART_DIV_L **Divisor Low 8 bits register** **Reset：0x00000001**

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

Field	Description
7: 0 DIV_L	**Baud rate divisor** Divisor low 8 bits.

7.6.3 Divisor High 8 Bits Register(UART_DIV_H)

Table 7-7 UART_DIV_H register

UART_DIV_H **Divisor High 8 bits register** **Reset：0x00000000**

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

Bits	Description
7: 0 DIV_H	**Baud rate divisor** Divisor high 8 bits.

7.6.4 UART Supplementary Control Register 0(UART_LCR0)

Table 7-8 UART_LCR0 register

UART_LCR0 **Control Register 0** **Reset：0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name										SUB	SP	EPS	PEN	STB	WLS1_WLS0	
Type										RW	RW	RW	RW	RW	RW	
Reset										0	0	0	0	0	0	

Note：

SUB bit configuration must be 0, otherwise tx transmits ‘0’ at any time.

Field	Description
6 SUB	**Set up Break** 0: No effect 1: SOUT signal is forced into the "0" state.
5 SP	**Stick parity** 0: No effect. 1: The parity bit is forced into a defined state, depending on the states of EPS and PEN If EPS = 1 & PEN = 1, the parity bit is set and checked = 0. If EPS = 0 & PEN = 1, the parity bit is set and checked = 1.

Field	Description
4 EPS	**Select even parity** 0: an odd number of ones is sent and checked. 1: an even number of ones is sent and checked.
3 PEN	**Enable parity** 0: The parity is neither transmitted nor checked. 1: The parity is transmitted and checked.
2 STB	**Number of STOP bits** 0: One STOP bit is always added. 1: Two STOP bits are added after each character is sent; unless the character length is 5 when 1 STOP bit is added.
1: 0 WLS1_WLS0	**Selects word length** **Combined with UART_LCR1[WLS2] as WSL[2:0]** 000: 7 bits 001: 7 bits 010: 7 bits 011: 8 bits 100: 9 bits ``` 111: 9 bits

7.6.5 UART Supplementary Control Register 1(UART_LCR1)

Table 7-9 UART_LCR1 register

UART_LCR1 **Control Register 1** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									INVTX	INVRX	WLS2	LOOP			TXEN	RXEN
Type									RW	RW	RW	RW			RW	RW
Reset									0	0	0	0			0	0

Field	Description
7 INVTX	**Determine whether to inverse the tx output, including idle, break, data bits, start bit, stop bit.** 0: don't inverse TX output 1: inverse TX output
6 INVRX	**Determine whether to inverse the RX input, including idle, break, data bits, start bit, stop bit.**

	0: don't inverse RX input 1: inverse RX input
5 WLS2	**Determine whether 9 bits data mode is available** 0: not available 1: available
4 LOOP	**LOOP** 0: for user normal use 1: control the uart into loop mode(used for testing uart by itself)
1 TXEN	**UART Transmitter enable** 0: disable 1: enable
0 RXEN	**UART Receiver enable** 0: disable 1: enable

7.6.6 FIFO Control Register(UART_FCR)

Table 7-10 UART_FCR register

UART_FCR **FIFO Control Register** **Reset: 0x00000000**

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

Field	Description
0 FIFOE	**Enable FIFO** 0: disable both RX and TX FIFOs 1: enable both RX and TX FIFOs.

7.6.7 Interrupt Enable Register(UART_IER)

Table 7-11 UART_IER register

UART_IER **Interrupt Enable Register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name								ETXDF		EOEBI	ENE	EFE	EPE	ETC	ETXE	ERXNE
Type								RW		RW	RW	RW	RW	RW	RW	RW
Reset								0		0	0	0	0	0	0	0

Field	Description
8 ETXDF	**Transmission register or transmit FIFO full interrupt enable** 0: disable 1: enable
6 EOEBI	**Interrupt enable of overflow error or break error** 0: disable 1: enable
5 ENE	**Interrupt enable of noise error** 0: disable 1: enable
4 EFE	**Interrupt enable of frame error** 0: disable 1: enable
3 EPE	**Interrupt enable of parity error** 0: disable 1: enable
2 ETC	**Interrupt enable of transmitting completed** 0: disable 1: enable
1 ETXE	**Interrupt enable of transmitting data empty** 0: disable 1: enable **Note: fifoe=1 represents fifo empty;** **fifoe=0 represents data register empty.**

Field	Description
0 ERXNE	**Interrupt enable of receiving data not empty** 0: disable 1: enable **Note: fifoe=1 represents fifo not empty;** **fifoe=0 represents data register not empty.**

7.6.8 Status register 0(UART_LSR0)

Table 7-12 UART_LSR0 register

UART_LSR0 **Line Status Register 0** **Reset：0x00000020**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name								TXDF	NE	TC	THRE	BI	FE	PE	OE	DR
Type								RO	R/W1C	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C	RO
Reset								0	0	0	1	0	0	0	0	0

Note：

SUB NE/PE/FE error just aims at the current byte data. Meanwhile, OE/BI will exist until it is cleared.

Field	Description
8 TXDF	**Transmission data register or transmit FIFO full flag** 0: transmission data register(fifoe=0) or transmission FIFO (fifoe=1) not full 1: transmission data register(fifoe=0) or transmission FIFO (fifoe=1) full **Note: This flag reflects the data and transmission status.**
7 NE	**Noise error flag** 0**:** No noise error**.** 1: Noise error has been occurred. **Note: write 1 to clear this flag to 0.**
6 TC	**The flag of Transmitting completed.** 0: TX FIFO(fifoe=1) or TX register(fifoe=0) is not empty, or transmitter has not finished the data shifting.

Field	Description
	1: TX FIFO(fifoe=1) or TX register(fifoe=0) is empty and transmitter has finished the data shifting. **Note: The default value at power-on is 0, and TC will only work after TXEN is 1. Write data to TX FIFO(fifoe=1)/TX register(fifoe=0) to clear this flag to 0. For LIN function, set SDBRK bit also can clear this flag to 0.**
5 THRE	**The empty flag of TX holding register or TX FIFO** 0: Reset whenever the contents of the TX FIFO are not empty, or whenever TX holding register is not empty (FIFOs are disabled). 1: Set whenever the contents of the TX FIFO are empty, or whenever TX holding register is empty (FIFOs are disabled). **Note: write data to TX FIFO(fifoe=1)/TX register(fifoe=0) to clear this flag to 0.**
4 BI	**Break error flag** 0: No break error 1: Break error has been occurred. If FIFOs are disabled, this bit is set whenever the SIN is held in the 0 state for more than one transmission time (START bit + DATA bits + PARITY + STOP bit). When a break occurs, only one zero character is loaded into FIFO or TX holding register. **Note: write 1 to clear this flag to 0.**
3 FE	**Framing error flag** 0: No framing error. 1: Framing error has been occurred. This bit will be set if the received data do not have a valid STOP bit. **Note: write 1 to clear this flag to 0.**
2 PE	**Parity error flag** 0: No parity error. 1: Parity error has been occurred. This bit will be set if the received data do not have a valid parity bit. **Note: write 1 to clear this flag to 0.**
1 OE	**Overflow error flag** 0: No RX overflow error. 1: RX overflow error has been occurred. If FIFOs are disabled, this bit will be set if the RX buffer is not read by the CPU before the new data from the RX shift register overwrites the previous contents. If FIFOs are enabled, an overflow error occurs when RX FIFO is full and the RX shift register becomes full. OE is set as soon as this happens. The character in the shift register is then overwritten, but not transferred to FIFO.

Field	Description
	Note: write 1 to clear this flag to 0.
0 DR	**Data ready flag** 0: Data not ready 1: Data ready. Set by the RX buffer becoming full or RX FIFO not empty (at least one byte being transferred into the FIFO). **Note: Read data register UART_RBR/THR, or read all RX FIFOs to clear this flag to 0 automatically if FIFO is enabled.**

7.6.9 Status register 1(UART_LSR1)

Table 7-13 UART_LSR1 register

UART_LSR1 **Line Status Register 1** **Reset：0x000000E0**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name											UART_IDLE	LINWAK		FBRK	SYNERR	IDLE
Type											RO	RO		R/W1C	R/W1C	R/W1C
Reset											1	0		0	0	0

Field	Description
5 UART_IDLE	**UART_IDLE** 0: UART is at operation. 1: UART is not at operation, namely, transmitter and receiver are not working or has finished the data transmission or reception.
4 LINWAK	**LIN wakeup flag** 0: wake-up signal has not been received 1: wake-up signal has been received
2 FBRK	**The flag of LIN BREAK occurred.** 0: LIN has not detected the break field in LIN frame 1: LIN has detected the break field in LIN frame **Note: write 1 to clear this flag to 0.**
1 SYNERR	**LIN Sync field error flag** 0: there exists no error 1: there exists error

Field	Description
	Note: write 1 to clear this flag to 0.
0 IDLE	**IDLE flag** 0: idle line has not been detected 1: idle line detected Receiver has received the data followed by a high level maintaining at least one byte data time. IDLE status flag will work after the IDLE line detection (ILEN) is enabled. **Note: write 1 to clear this flag to 0.**

7.6.10 UART Sample Counter Register(UART_SMP_CNT)

Table 7-14 UART_SMP_CNT register

UART_SMP_CNT **Sample Counter Register** **Reset：0x00000000**

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

Field	Description
1: 0 SMP_CNT	**UART sample counter** 00: Based on 16*baud_pulse, baud_rate = system clock frequency/16/{DLH, DLL} 01: Based on 8*baud_pulse, baud_rate = system clock frequency/8/{DLH, DLL} 10: Based on 4*baud_pulse, baud_rate = system clock frequency/4/{DLH, DLL} 11: reserved

7.6.11 Guard Time Added Register(UART_GUARD)

Table 7-15 UART_GUARD register

UART_GUARD **Guard time register** **Reset：0x0000000F**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name												GUARD_EN	GUARD_CNT			
Type												RW	RW			
Reset												0	F			

Note:
Adding the guard time can contribute to eliminate the accumulated error every byte, so it is significant to improve the accuracy of the baud rate by using the fraction divisor with the guard time.

Field	Description
4 GUARD_EN	**Guard interval time adds enabling signal** 0: disable 1: enable
3: 0 GUARD_CNT	**Guard interval counting value** 0~15: 0 ~ 15 bit time

7.6.12 Sleep enable register(UART_SLEEP_EN)

Table 7-16 UART_SLEEP_EN register

UART_SLEEP_EN **Sleep Enable Register** **Reset: 0x00000000**

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

Field	Description
0 SLEEP_EN	**Sleep function enable** 0: Does not deal with sleep mode indication signal 1: Activate hardware flow control according to software initial settings when the chip enters the sleep mode. Release the hardware flow when the chip wakes up.

7.6.13 Fractional Divider Register(UART_DIV_FRAC)

Table 7-17 UART_DIV_FRAC register

UART_DIV_FRAC **Fractional Divider Register** **Reset: 0x00000000**

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

Field	Description
4: 0 DIV_FRAC	**Fractional divisor** If actual divisor is 135.65, then DIV_FRAC is 0.65*32 = [20.8]=21 , and the DIV_L=135 .

7.6.14 Idle Interrupt Enable Register(UART_IDLE)

Table 7-18 UART_IDLE register

UART_IDLE **Idle interrupt enable register** **Reset: 0x00000000**

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

Field	Description
7 ILEN	**Idle line detect enable** 0: disabled 1: enabled
4 IDLEIE	**IDLE interrupt enable** 0: disabled 1: enabled

7.6.15 Software LIN Control Register(UART_LINCR)

Table 7-19 UART_LINCR register

UART_LINCR **LIN control register** **Reset：0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									LINEN	LBRKIE	LBRKDL	SDBRK	LABAUDEN	SYNERRIE	LINWAKEN	LINSLP
Type									RW	RW	RW	RW	RW	RW	RW	RW
Reset									0	0	0	0	0	0	0	0

Field	Description
7 LINEN	**LIN Mode enable** 0: disable 1: enable
6 LBRKIE	**LIN Break detect interrupt enable** 0: disable 1: enable
5 LBRKDL	**LIN Break detect length** 0: 10 bits 1: 11bits
4 SDBRK	**LIN transmit Break enable** 0: disable 1: enable transmit Break, the Break length is determined by the register BRKLGH. **Note:** It is set by software and cleared by MCU internal hardware after the Break transmits completed.
3 LABAUDEN	**Sync field auto baud rate enable** 0:0X55 is not used to auto baud rate detection 1:0x55 is used to auto baud rate detection
2 SYNERRIE	**Sync field error interrupt enable** 0: disable sync byte error interrupt 1: enable sync byte error interrupt
1 LINWAKIE	**LIN wakeup interrupt enable** 0: disable LIN wakeup interrupt 1: enable LIN wakeup interrupt

Field	Description
0 LINSLP	**LIN sleep state** 0: LIN exits sleep state 1: LIN enters sleep state

7.6.16 Software LIN Sync Break Length Control(UART_BRKLH)

Table 7-20 UART_BRKLH register

UART_BRKLH **LIN sync break length control register** **Reset: 0x00000000**

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

Field	Description
3: 0 BRKLGH	**LIN transmit Break length** 0000: 13bits sync break length 0001: 14bits sync break length …… 1111: 28bits sync break length

8 Analog-to-Digital Converter(ADC)

8.1 Introduction

The ADC is a 12-bit successive approximation analog-to-digital converter. It has up to 18 external channels and 1 internal channels. A/D conversion of the various channels can be performed in single, continuous, scan or discontinuous mode. The analog monitor feature allows the application to monitor whether the input voltage exceeds the voltage range setting.

8.2 Features

- 12-bit resolution.
- Channel input voltage range: AVSS < Vin < AVDD.
- Maximum conversion rate: 250Ksps.
- Reference voltage supports AVDD and external VREF+/VREF-.
- Channel:
  - 18 external channels.
  - 1 internal monitoring channel, which can measure 1 internal temperature sensor(T-Sensor) and 1 bandgap reference voltage(Bandgap) respectively.
- Each channel can configure sample time individually.
- Conversion sequence is classified as regular group and injection group.
  - Regular group: configurable up to 21 channels.
  - Injection group: configurable up to 4 channels.
- 8 operation modes (called mode x for convenience, x=1~8):
  - Single regular group channel single conversion (mode1).
  - Single regular group channel continuous conversion (mode2).
  - Multiple regular group channels single scan with injected trigger (mode3 scan injection).
  - Multiple regular group channels single scan with discontinuous injected trigger (mode3 discontinuous injected).
  - Multiple regular group channels single scan with automatically injected (mode4).
  - Multiple regular group channels continuous scan with injected trigger (mode5 scan injected).
  - Multiple regular group channels continuous scan with injected trigger (mode5 discontinuous injected).

- Multiple regular group channels continuous scan with automatically injected (mode6).
- Multiple regular group channels under discontinuous conversion mode (mode7).
- Multiple injected group channels under discontinuous conversion mode(mode8).

- Start ADC by internal software trigger or external hardware trigger conversion.
- Analog monitor function:
  - Monitor single or all channels voltage by configuration.
  - Monitor whether the channel voltage is less than the low threshold or more than the high threshold.
- Interrupts:
  - End of conversion (EOCx, End of Conversion, x: 0~20) for regular group.
  - End of injected group conversion (IEOCx , x: 0~3).
  - Analog Monitor event (AMO,AAMO,NAMO).

8.3 ADC functional description

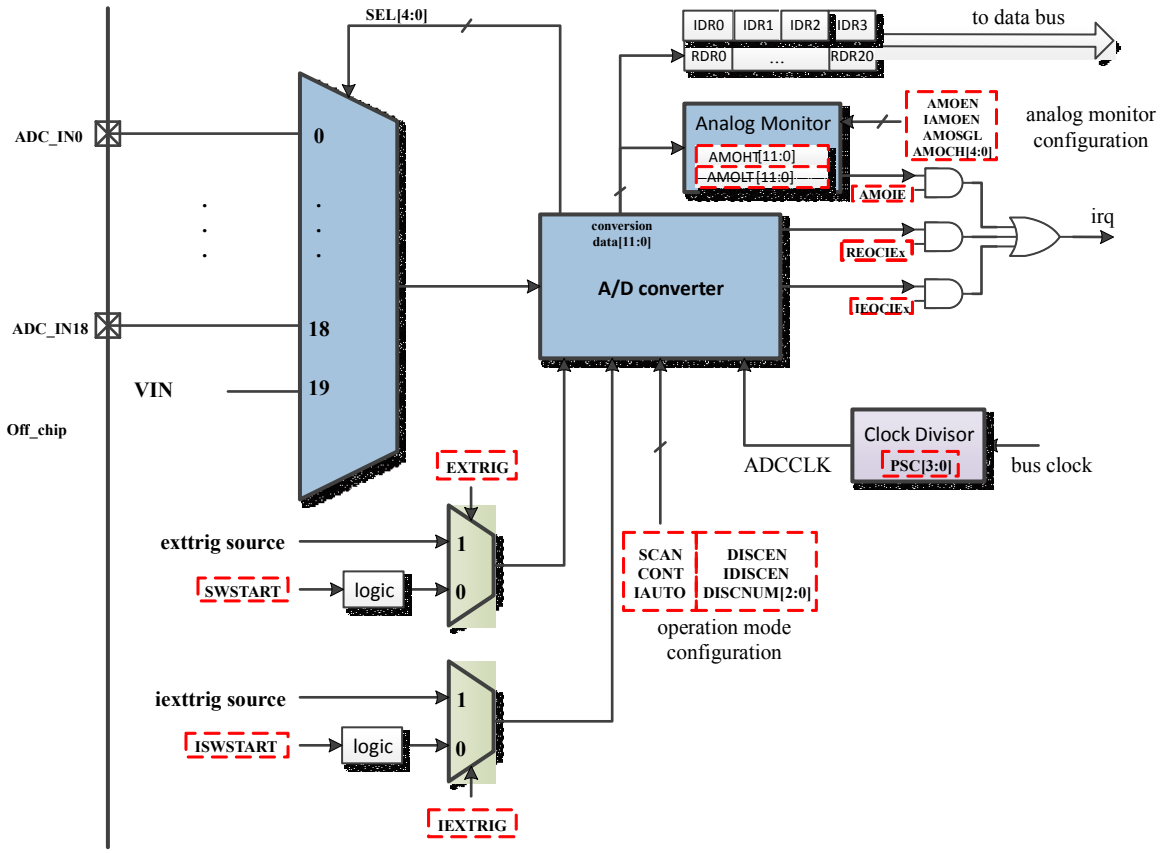

Figure 8-1 ADC block diagram

8.4 Function Description

ADC is composed of ADC converter unit, input channel selector, clock divisor, and analog monitor, etc. As illustrated in Figure 8-1 , A/D converter unit operates at ADC clock(ADCCLK) and the other circuit parts operate at bus clock.

A typical operation flow is introduced in the following paragraph.

Firstly, ADC should be powered on, ADC can be triggered by the internal ADC_CTRL0[SWSTART] or external trigger source, which is derived from other module. After trigger, the ADC converter unit starts to work and sends out the selecting signal to input channel selector to select the desired channel one by one based on the regular or injected group channels sequence. After one channel has finished the conversion, the conversion result is stored into the ADC_RDRx or ADC_IDRx based on which group the current conversion channel belongs to, and generate corresponding ADC_REOC[EOCx] or ADC_IEOC[IEOCx] flag bit. Meanwhile, the analog monitor starts to work and the related statuses flag appears if the corresponding event occurs. To point out, there exist some differences for different operation mode and detailed information will be illustrated later.

8.4.1 Power on sequence

Before starting all the functions, ADC should be powered on at first. Then a valid trigger can start the ADC to operate based on the configured mode. The power-on sequence is illustrated as follows.

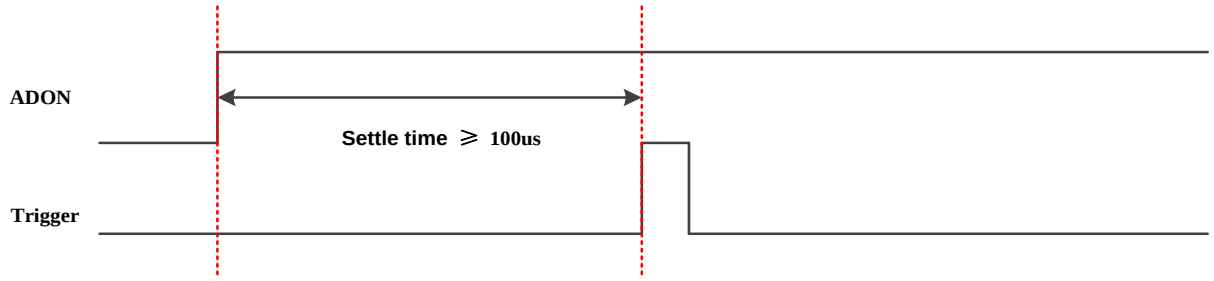

Figure 8-2 ADC power on sequence

As illustrated in the Figure 8-2, the bit ADC_CTRL1[ADON] is set to 1 to control the power on process. After the ADON is set, the waiting time for A/D Converter unit power on is not less than 100us.

8.4.2 Operation modes

Different modes can be used flexibly based on the practical application. After power on and a valid trigger, ADC operates at one of the following modes.

Table 8-1 ADC operation modes and its corresponding configuration

Operation mode	MODE_BITS	Trigger source	Conversion sequence
mode1	5'b0000x	regular trigger	single regular group channel single conversion
mode2	5'b0100x	regular trigger	single regular group channel continuous conversion
mode3 (injected group scan mode)	5'b10000 (INTERVAL=0)	regular trigger (+injected trigger)	multiple regular group channels (+injected group channels) single conversion
mode3 (injected group discontinuous mode)	5'b10000 (INTERVAL=1)	regular trigger (+injected trigger)	multiple regular group channels (+injected group channels) single conversion
mode4	5'b10001	regular trigger + auto injected trigger	multiple regular group channels + injected group channels single conversion
mode5 (injected group scan mode)	5'b11000 (INTERVAL=0)	regular trigger (+injected trigger)	multiple regular group channels (+injected group channels) continuous conversion
mode5 (injected group discontinuous mode)	5'b11000 (INTERVAL=1)	regular trigger (+injected trigger)	multiple regular group channels (+injected group channels) continuous conversion
mode6	5'b11001	regular trigger + auto injected trigger	multiple regular group channels + injected group channels continuous conversion
mode7	5'b1x10x	regular trigger	regular group subgroup scan mode conversion
mode8	5'b1x01x	injected trigger	injected group channels scan mode conversion

Note: MODE_BITS = {SCAN, CONT, DISCEN, IDISEN, IAUTO}.

Before describing the mode operation, it's necessary to introduce some terminologies, such as regular group, injected group.

For ADC input channels(19 in total), they are called ch0 to ch19 (not include ch17), of which ch0 to ch18 are the external input channels, corresponding to ADC_IN0~ADC_IN18 in Table 14-2, ch19 corresponds to internal voltage monitor channel.

The ADC module contains a regular group and an injection group. Both regular group and injection group can be configured to convert zero or more ADC input channels. Input channel configuration in the same group or in different groups, no additional restrictions other than the maximum number. That is, any input channel, such as ch0, can be configured to a regular group or an injection group, or both groups, or even multiple ch0s can be configured in one group.

Regular group and injection group can be triggered by different trigger sources. Channels in the same group are converted sequentially (except for mode 1 and mode 2).

The conversion result of the regular group will be stored in ADC_RSQR0, ADC_RSQR1, ADC_RSQR2 registers in order, the regular group is composed of up to 21 channels in sequence from RSQ0 to RSQ20.

For example, if the RSQ0 to RSQ11 are set to 9,8,12,1,5,4,7,3,13,2,0,0 respectively, a regular group is arranged in Figure 8-3.

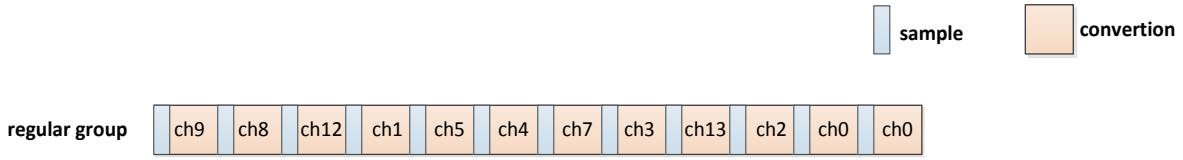

Figure 8-3 Regular group sequence

If the RSQL is set 8(length=9), the last 3 channels will be invalid and not be converted. Valid regular group sequence is illustrated in Figure 8-4.

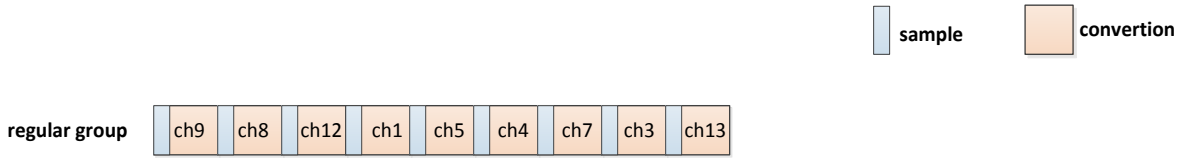

Figure 8-4 Valid regular group sequence

In the same way, the conversion results of the injected group are stored in ADC_ISQR0~ADC_ISQR3 in turn. Based on the register ADC_ISQR, the injected group is composed of up to 4 channels in sequence from ISQ0 to ISQ3.

For example, if the ISQ0 to ISQ3 are set to 12,7,13,2 respectively, an injected group is arranged as Figure 8-5.

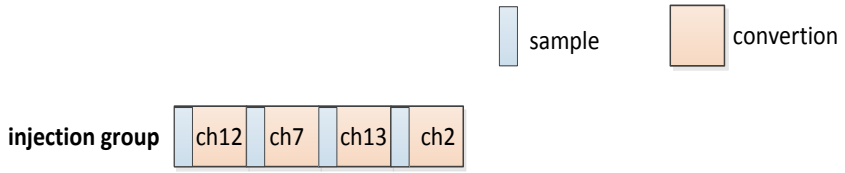

Figure 8-5 Injected group sequence

If the ISQL is 2(Length=3), the last 1 channel will be invalid and not be converted. Valid injected group sequence is illustrated in Figure 8-6 .

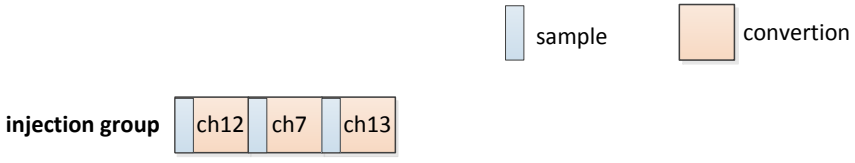

Figure 8-6 Valid injected group sequence

Obviously, regular group trigger and injected group trigger are the corresponding signals to start conversion of the regular and injected group sequence. The trigger is derived from the internal ADC_CTRL0[SWSTART] or external trigger source illustrated in the ADC block diagram. And the

regular trigger is invalid when ADC is in the process of regular group channel conversion. Based on the basic introduction, detailed information for each mode is described as follows.

8.4.2.1 Mode 1

This mode only converts the first channel in regular group, ignoring RSQL values. After the mode is configured in Table 8-1, a valid trigger can make the ADC work in this mode.

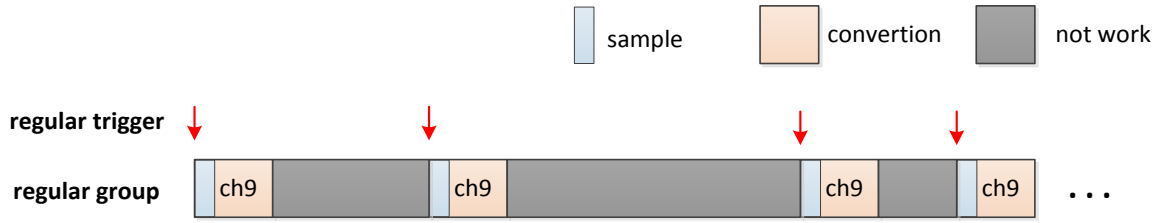

Figure 8-7 Mode 1 operation flow

As the Figure 8-7 shows, the first channel in regular group is converted once after a valid regular trigger. Then the ADC goes to idle until the next valid regular trigger for the next conversion.

8.4.2.2 Mode 2

This mode continuously converts the first channel in regular group, ignoring RSQL values. After the mode makes configuration as Table 8-1, a valid trigger can make the ADC work in this mode.

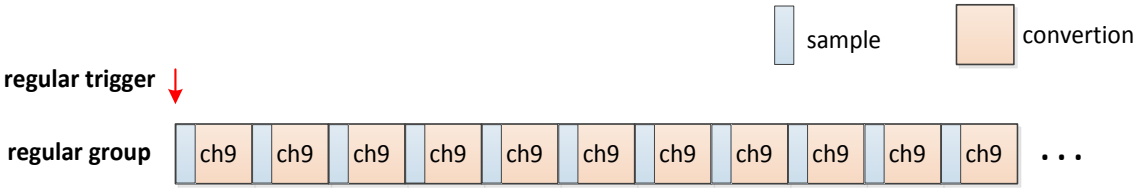

Figure 8-8 Mode 2 operation flow

As shown in Figure 8-8, the first channel in regular group is converted forever except power down, reset or mode change after a valid regular trigger.

8.4.2.3 Mode 3

8.4.2.3.1 interval bit=0, injected group is the scan mode

This mode converts regular group and injected group. The valid regular and injected group channel number are decided by RSQL and ISQL respectively. With the mode configuration in Table 8-1, a valid trigger can make the ADC work in this mode. For example, RSQL is set to 6 (Length=7), and ISQL is set to 2 (Length=3). A typical operation shows in Figure 8-9. The initial regular trigger starts conversion of the first 7 channels in group. When ADC converts the ch1 in regular group, an injected trigger is generated and switches the conversion to 3 injected group channels after the end of ch1 conversion. After all injected group channels have been converted completely, conversion

switches back to regular group channels automatically convert the ch5. When finishing the valid regular channels conversion, ADC runs to idle until the next trigger.

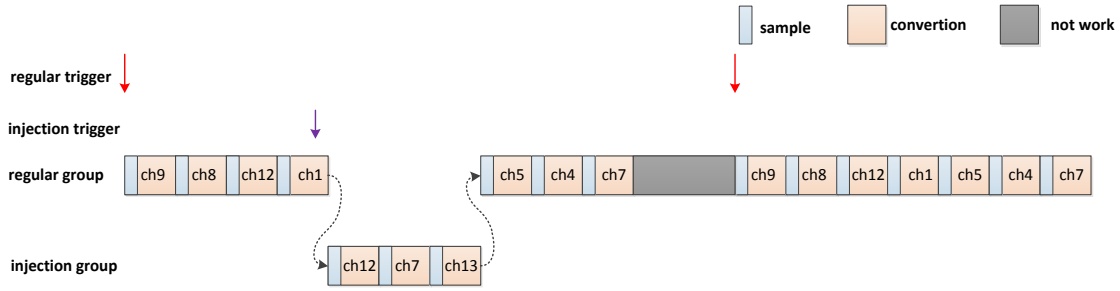

Figure 8-9 Mode 3 operation flow with injected trigger scan mode

Especially, if the injected trigger occurs when the ADC is idle, the ADC will finish the conversion of the valid injected group channels as the following Figure 8-10.

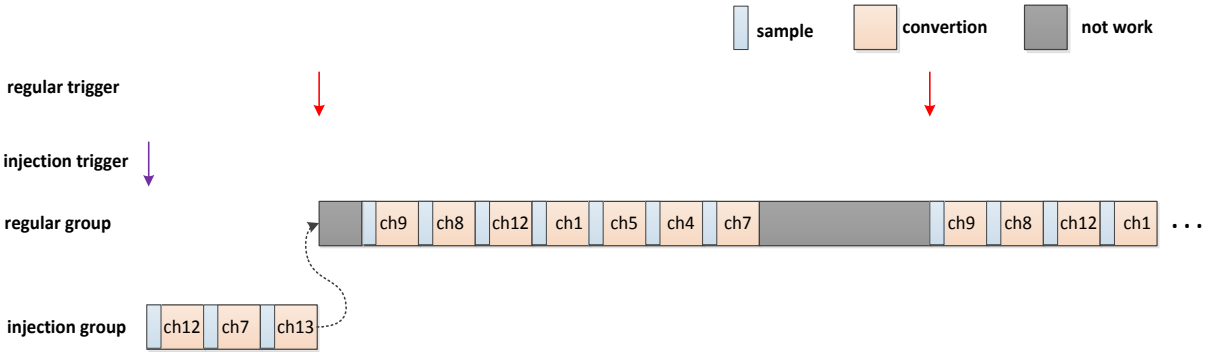

Figure 8-10 Mode 3 operation flow with injected trigger at ADC idle state

8.4.2.3.2 interval bit=1, injected group is the discontinuous mode

Compared to Figure 8-9, the generation of an injected trigger will only convert one channel of the injected group sequence, and the next time the injected trigger occurs, the next channel of the injected group sequence will be converted.

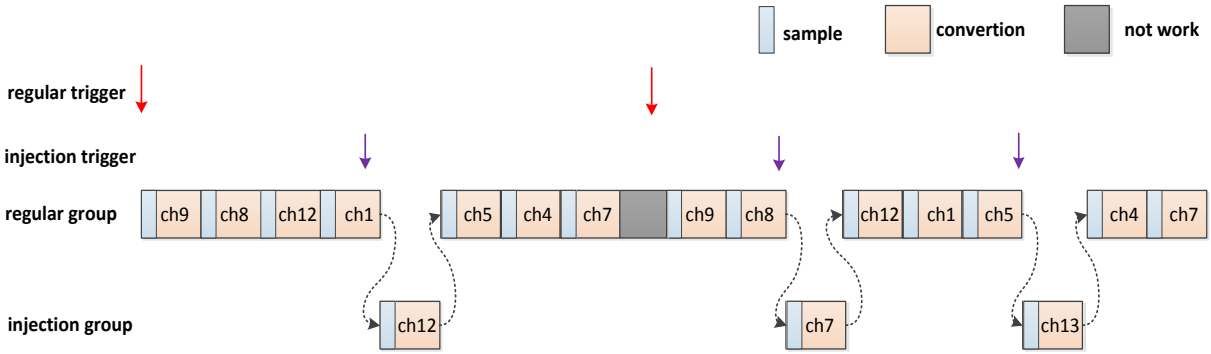

Figure 8-11 Mode 3 operation flow with injected trigger discontinuous mode

8.4.2.4 Mode 4

After this mode is triggered, the regular group is automatically converted first and then convert the injected group. The valid regular channels and injected group channels are decided by RSQL and ISQL respectively. With the mode configuration in Table 8-1, a valid trigger can make the ADC work in this mode. For example, RSQL is set to 6 and ISQL is set to 2. A typical operation shows in Figure 8-12. A regular trigger starts conversion of the first 7 regular group channels followed by 3 injected group channels automatically. After the total 10 channels has been converted completely, ADC runs to idle state until the next valid regular trigger.

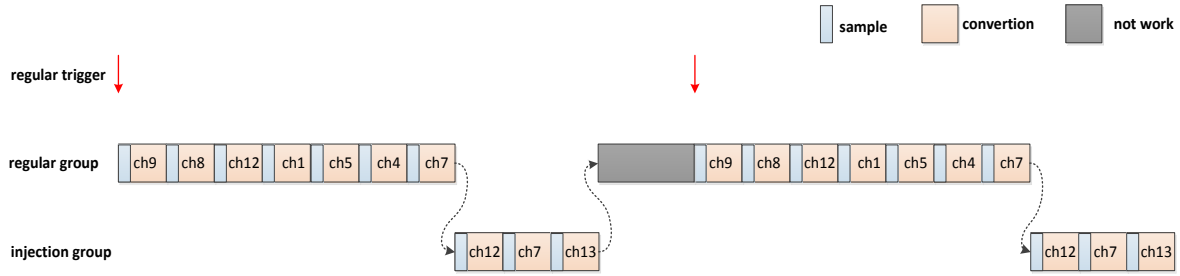

Figure 8-12 Mode 4 operation flow

8.4.2.5 Mode 5

8.4.2.5.1 interval bit=0, injected group is the scan mode

The difference with Mode 3 is that this mode is a continuous conversion. The valid regular channels and injected group channels length is decided by RSQL and ISQL respectively. Different from Mode 3, continuous conversion is used in this mode. With the mode configuration in Table 8-1, a valid trigger can make the ADC work in this mode. A key feature for this mode is that a single regular trigger can make the ADC work all the time except power down, reset or mode change. For example, RSQL is set to 6 and ISQL is set to 2. A typical operation shows in Figure 8-13. The ADC works on regular group channels in order circularly after a regular trigger or works on the injected group channels if an injected trigger occurs.

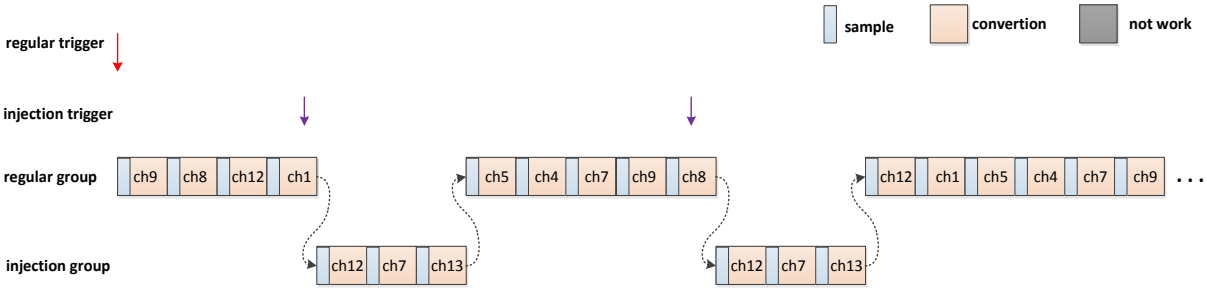

Figure 8-13 Mode 5 operation flow of injected group scan mode

Especially, if the injected trigger occurs when the ADC is idle, the ADC will finish the conversion of the valid injected group channels at first as the following Figure 8-14.

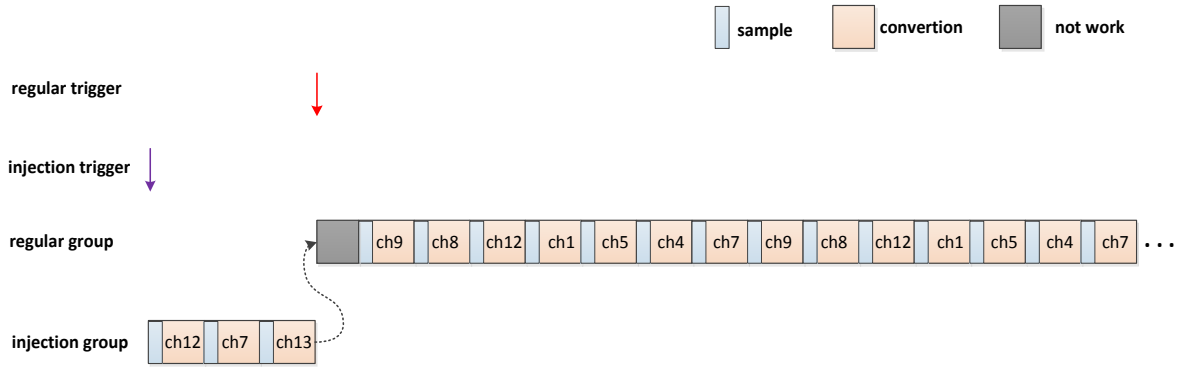

Figure 8-14 Mode 5 operation flow with injected trigger at ADC idle state

8.4.2.5.2 interval bit=1, infection group is the discontinuous mode

Compared to Figure 8-13, the generation of an injected trigger will only convert one channel of the injected group sequence, and the next time the injected trigger occurs, the next channel of the injected group sequence will be converted.

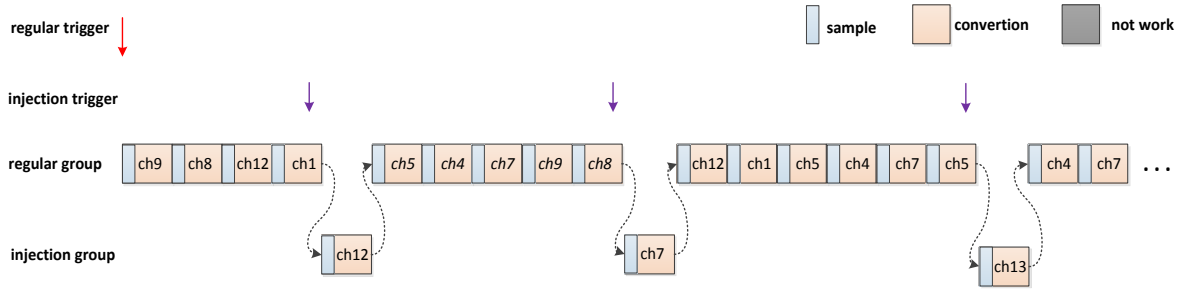

Figure 8-15 Mode 5 operation flow with injected trigger discontinuous mode

8.4.2.6 Mode 6

The difference from Mode 4 is that this mode is a continuous conversion. The valid regular channels and injected group channels length is decided by RSQL and ISQL respectively. Different from Mode 4, continuous conversion is done in this mode. With the mode configuration in Table 8-1, a valid regular trigger can make the ADC work in this mode. A key feature for this mode is that a single regular trigger can make the ADC work all the time except power down, reset and mode change. For example, RSQL is set to 6 and ISQL is set to 2. An operation flow shows in Figure 8-16. The ADC works on regular group channels in order followed by injected group channels circularly after a regular trigger.

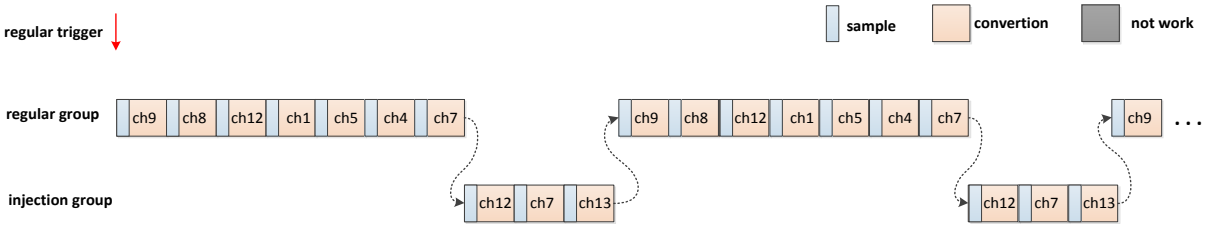

Figure 8-16 Mode 6 operation flow

8.4.2.7 Mode 7

This mode only converts regular group. The valid regular channels group channels are decided by RSQL. With the mode configuration in Table 8-1, ADC can operate in this mode. The valid regular channels are divided into several subgroups every DISCNUM channels.

For example, RSQL is set to 6 and DISCNUM is set to 1.

First regular trigger: ch9, ch8;

Second regular trigger: ch12, ch1;

Third regular trigger: ch5, ch4;

Fourth regular trigger: ch7;

Therefore, the practical conversion flow is illustrated in Figure 8-17.

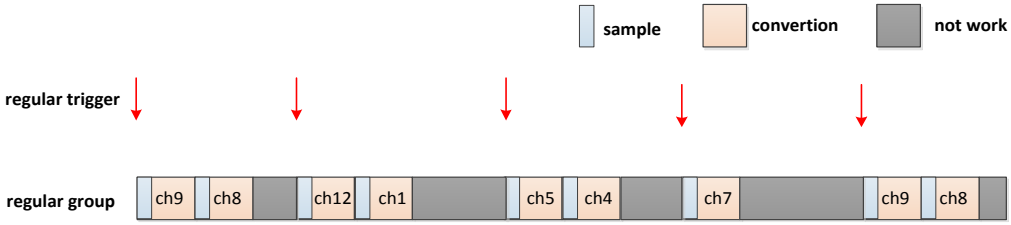

Figure 8-17 Mode 7 operation flow

8.4.2.8 Mode 8

This mode only converts injected group. The valid injected channels group channels are decided by ISQL. With the mode configuration in Table 8-1, ADC can operate in this mode. The valid injected channels are divided into several subgroups every one channel.

For example, ISQL is set to 2.

First injected trigger: ch12;

Second injected trigger: ch7;

Third injected trigger: ch13;

Fourth injected trigger: ch12;

…

Therefore, the practical conversion flow is illustrated in Figure 8-18.

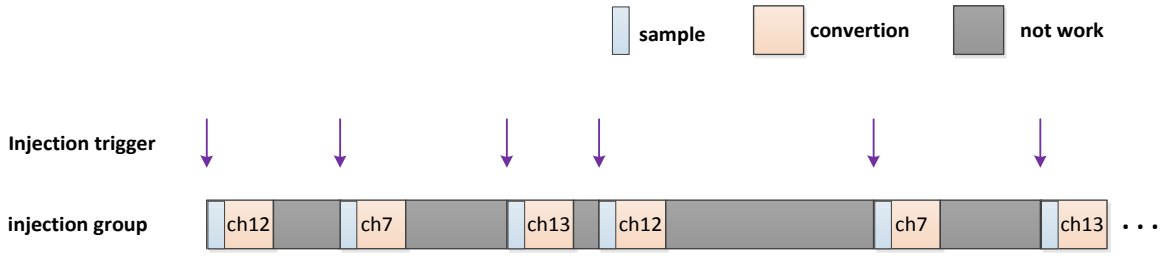

Figure 8-18 Mode 8 operation flow

8.4.3 Trigger mode

According to different triggering methods, the following 7 scenarios can be combined.

Table 8-2 Responsive behavior under different triggering methods

Trigger method	Responsive behavior
Trigger regular group	Regular group conversion
Trigger injected group	Injected group conversion
Regular trigger is generated during regular group conversion	The regular group continues to convert, and the second trigger event does not respond
Injected trigger is generated during regular group conversion	Wait for the current channel of the regular group to be converted before switching to the injected group. After the injected group conversion, switch to the original regular group to continue conversion (if the regular group sequence has not been converted).
Injected trigger is generated during injected group conversion	The injected group continues to convert, the second injected trigger event does not respond
Regular trigger is generated during injected group conversion	A regular event is generated during the injected conversion, the injected conversion will not be interrupted, but the regular sequence will be executed after the injected sequence ends up.
Regular trigger and injected trigger are generated at the same time	Convert regular group after the injected group conversion is completed.

8.4.4 Linear calibration

The ADC module of AC7802x supports the linear calibration and calculation function. This method is mainly used in medium-precision SAR ADCs. Due to the large number of channels and switches, performance degradation caused by non-ideal factors will occur in practical applications, mainly manifested in the deterioration of static performance. The purpose of code value correction is to optimize the non-ideal ADC transfer function curve into a curve close to the ideal transfer function through measurement and calculation, thereby improving static performance, as shown in Figure 8-19.

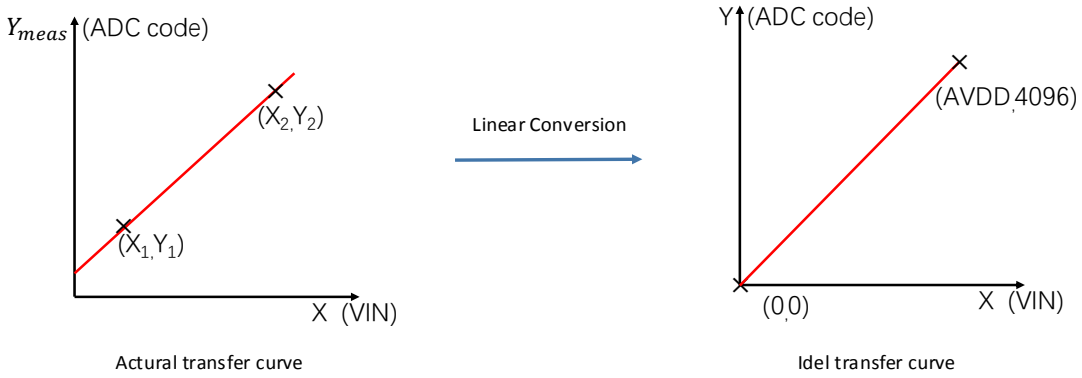

Figure 8-19 ADC GEOE calibration conversion diagram

It should be noted that this correction cannot change the transfer curve of the ADC itself, but can only optimize the shift of the transfer curve, so the dynamic performance cannot be optimized.

When the AC7802x series leave the factory, linearity calibration will be performed under 5V power supply, 5V external reference and normal temperature conditions to eliminate the linearity error caused by chip production.

During the production calibration, the compensation values of the ADC GE and OE will be obtained, and the values will be written into the flash, and the chip will automatically load the values into the ADC_CGV and ADC_COV registers after the chip is powered on.

However, since the GE OE value in the flash is only 9 bits (the most significant bit8 is the sign bit), which is different from the number of bits in these two registers. When the chip reads the GE OE value in the flash, it only reads these 9 bits into bit0-8 of the corresponding register, the sign bit expansion and filling will not be performed, so the values of these two registers need to be read out by software after power on, and then written back to these two registers after performing the corresponding shift operation. register. This operation only needs to be performed once after power on.

8.4.5 Analog monitor

Analog monitor supports level triggered monitoring event and edge triggered monitoring event mode. If the monitored channels voltage is more than the high threshold or less than the low threshold, the analog monitor will generate a monitor event. Analog monitor can be used to monitor channels based on bits AMOEN, IAMOEN, AMOSGL and AMOCH.

The thresholds are configured through the AMOHR and AMOLR registers. The value configured in each field(AMOHT/ AMOLT/ AMOHO/ AMOLO) of these two registers is the ADC code used for hardware comparison. It should be noted that the ADC raw data is used for the threshold comparison, setting data alignment or injection group offset does not affect the comparison result.

Table 8-3 Analog monitor channel configuration

Analog monitor channel	{AMOEN,IAMOEN, AMOSGL}	Configurable operation mode	Comment
None	3'b00x	All modes	-
All injected group channels	3'b010	mode3/4/5/6/8	-
All regular group channels	3'b100	Except mode8	-
All channels	3'b110	All modes	-
Single injected group channel	3'b011	mode3/4/5/6/8	Conversion sequence must contain the injected channel determined by AMOCH[4:0].
Single regular group channel	3'b101	mode1 ~ mode7	Conversion sequence must contain the regular channel determined by AMOCH[4:0].
Single regular or injected group channel	3'b111	All modes	Conversion sequence must contain the regular or injected channel determined by AMOCH[4:0].

8.4.5.1 Level trigger mode

Set AMOMODE=0, the analog monitor operates in level trigger mode.

If the monitored channels voltage is higher than the high threshold AMOHT or lower than the low threshold AMOLT, analog monitor sets the AMO flag to 1. Meanwhile, the interrupt occurs if AMOIE is configured to 1.

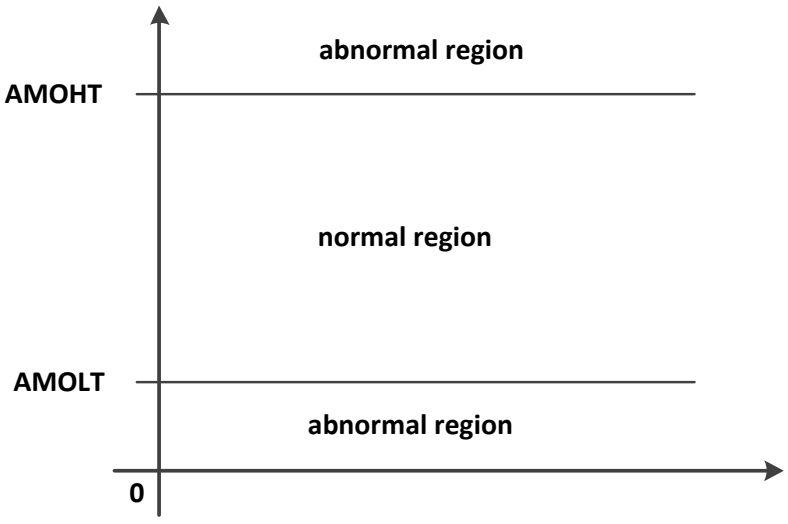

Figure 8-20 Monitor region in level trigger mode

8.4.5.2 Edge trigger mode

Set AMOMODE=1, the analog monitor operates in edge trigger mode.

If the monitored channels voltage goes from the normal region(between AMOLT and AMOHT) to the abnormal region(lower than the low threshold AMOLT or higher than the high threshold AMOHT), a monitor abnormal event is generated. Analog monitor sets the AAMO flag to 1. Meanwhile, the monitor event interrupt occurs if AMOIE is configured to 1.

When the monitored channel voltage goes from an abnormal region (lower than the low threshold AMOLT or higher than the high threshold AMOHT) to a normal region(between AMOHT-AMOHO and AMOLT+AMOLO), a monitoring recovery event is generated. The analog monitor sets the NAMO flag to 1. If the AMOIE is configured to 1, a monitoring event interrupt is generated. Boundary value = [High threshold-high offset value, low threshold + low offset value], which means [AMOHT-AMOHO, AMOLT+AMOLO].

Note:
The edge trigger mode can only be used when monitoring a single channel. Monitoring multiple channels cannot distinguish between abnormalities and recovery interrupts triggered by different channels.

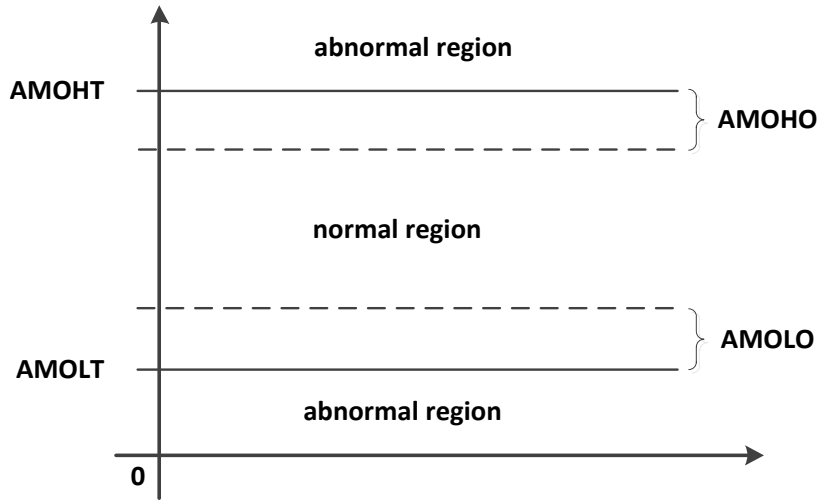

Figure 8-21 Monitor detecting region in edge trigger mode

8.4.6 Status flag

There are three flags to indicate the conversion status: EOCx, IEOCx and AMO(AAMO and NAMO are used in edge trigger mode). The EOCx flag indicates the end of the conversion for regular group channels. The IEOCx flag indicates that all the injected group channels are converted completely. The AMO flag indicates whether the analog monitor event occurs. The analog monitor event indicates whether the current conversion result is higher than high threshold or lower than low threshold based on the configuration. For different modes, The EOCx and IEOCx flags behave

identically, the flag AMO is generated at the same time for all the modes. The following descriptions are introduced with the assumption that ch5 is lower than AMOLT, ch7 is higher than the AMOHT, and the analog monitor is configured to check all the channels including regular group channels and injected group channels for example.

For any mode, the EOCx or IEOCx flag is generated when the conversion of the corresponding sequence channel is completed, and the AMO flag is generated at the same time. The detailed information about three flags is illustrated in Figure 8-22 based on mode 6.

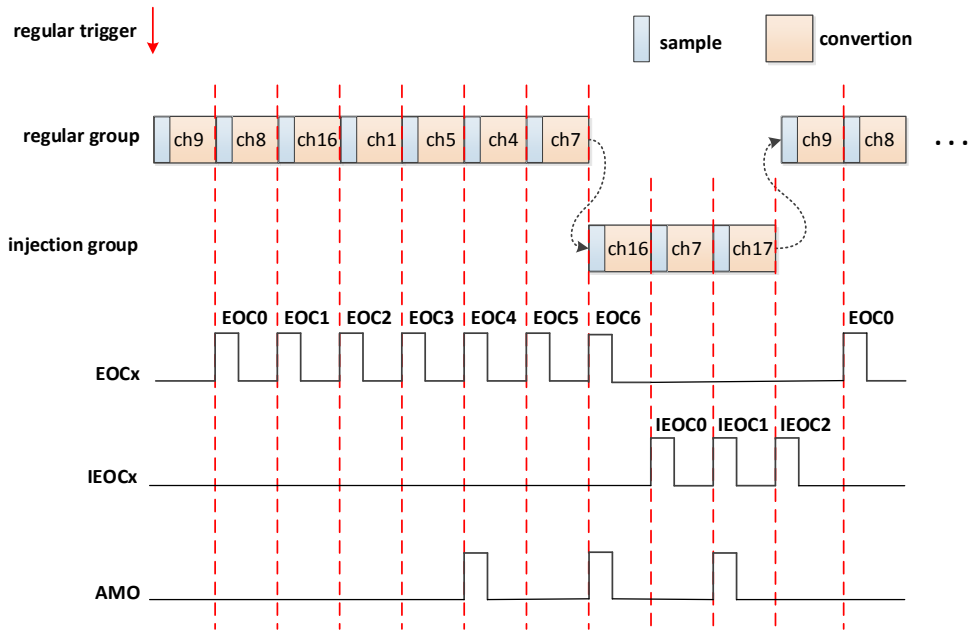

Figure 8-22 Three flag behaviors

8.4.7 DATA alignment

The data alignment function is used to control the arrangement of the ADC conversion result data in the data register, and data can be right-aligned or left-aligned. The data arrangement in the two data alignment modes is shown in Figure 8-23. It should be noted that if the injection group data register minuses ADC_IOFRx becomes negative, the sign bit will be discarded.

Figure 8-23 ADC data arrangement in data register

8.4.8 Sampling conversion time

The ADC needs to use several ADC_CLK cycles to sample the input voltage, i.e. charging the ADC's internal circuitry to the level of the external input signal, and completing the sampling before the analog to digital conversion can take place. The number of sampling cycles can be changed by the SPT[2:0] bits in the ADC_SPT register. Each channel can be sampled at different time.

Total conversion time: (SPT+ 12) * ADC cycles + 5 APB cycles

For example, if APB=16MHz, ADCCLK=8MHz, SPT=17 ADCCLK, total conversion time= (17+12)/8+(5/16) ≈ 4 μs.

The ADC clock uses pclk in the normal state of the MCU, please refer to Figure 4-1. The pclk supported by the ADC is up to 16M, and the conversion clock of the ADC, namely ADC_CLK, supports up to 8M.

The following table lists the sampling rate corresponding to each SPT configuration of the ADC under the highest clock configuration. It should be noted that the highest sampling rate supported by AC7802x ADC is 250Ksps. If the configured sampling rate is higher than this value, the accuracy of sampling results may be significantly reduced.

Table 8-4 Analog sampling rate

SPT	Sampling period(8M)	Sampling time(μs)	Conversion period((8M)	Total conversion peroid((16M)	Sampling rate(kHz)
0	9	1.125	12	47	340.43
1	7	0.875	12	43	372.09
2	17	2.125	12	63	253.97
3	33	4.125	12	95	168.42
4	64	8	12	157	101.91
5	140	17.5	12	309	51.78
6	215	26.875	12	459	34.86
7	5	0.625	12	39	410.26

8.4.9 Temperature sensor

The temperature sensor can be used to measure the ambient temperature(TA) of the device. The temperature sensor is internally connected to the ADC channel which is used to convert the sensor output voltage into a digital value.

The internal temperature sensor is more suited to applications that detect temperature variations instead of absolute temperatures. If accurate temperature measurement is needed, an external temperature sensor part should be used.

Temperature using the following formula:

Temperature (°C) = {(V_{TEMP25} - V_{SENSE}) / Slope} + 25

V_{TEMP25}: V_{SENSE} value for 25°C

V_{SENSE}: Current temperature and voltage value.

Slope = Average slope for Temperature (mV/°C).

Refer to the Electrical characteristics section for the actual values of V_{TEMP25} and Slope.

8.4.10 Low power mode

ADC has two power consumption modes available. One is the normal mode, the other is low power mode. The ADC clock operates at a lower frequency in low-power mode to reduce power consumption. ADC enters low-power mode when MCU runs into Stop mode. The ADC analog monitor event can wake up the MCU from Stop mode to Normal mode.

The power mode switching process is shown in Figure 8-24 .

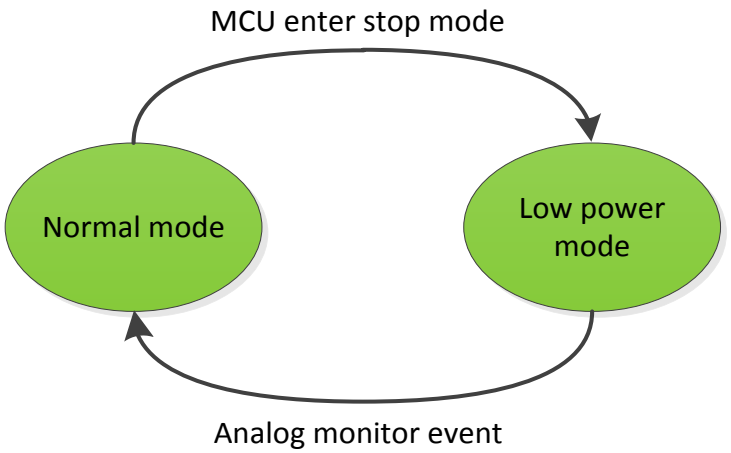

Figure 8-24 CPU power mode switching flow

If the ADC is configured to work in MCU STOP mode (configure the ADC as a wake-up source in SPM), the following situations need to be paid attention to.

1. When the MCU enters the STOP mode, the working clock of the ADC will automatically switch to HSI (32MHz). Since the highest operation clock of the ADC is 16MHz, it is necessary to adjust the frequency division of the ADC operation clock to higher than 2 frequency division before entering the STOP mode.

2. After configuring the ADC as a wake-up source in SPM, it is necessary to ensure that the clock of the ADC is enabled and the Reset signal is turned off.

3. In STOP mode, the fixed wake-up interval of the ADC is 12ms, and only one channel is converted each time the conversion is performed, that is, if it is mode3, after receiving the hardware trigger, convert one channel in the trigger group (regular group or injected group) sequentially every 12ms until all channels configured for the group are completed.

4. After the MCU exits the STOP mode, the operation clock of the ADC is automatically switched back to the original clock.

8.5 Application note

8.5.1 Reset and enable

This section introduces 3 reset ranges of the ADC. In order to use the ADC better, you need to understand these 3 reset ranges.

Table 8-5 ADC reset range

No.	Reset signal	ADC reset effect	Description
1	Set the CKGEN_PERI_SFT_RST1 [SRST_ADC0]	Reset the entire ADC, including all registers	All ADC register values are not retained. If you need to reuse the ADC, it is needed to reconfigure it, and it is needed to wait for more than 100μs after ADON is set to 1, and then trigger the conversion.
2	Set ADON to 0	Reset ADC internal state, and reset the RDR/IDR register	ADC configuration related registers will be retained. If you need to reuse the ADC, you only need to set ADON to 1 and send a trigger signal.
3	Modify ADC operation mode or modify any one of RSQR, ISQR, RSQL, ISQL	It only resets ADC internal state and it does not affect the configuration register.	ADC configuration related registers will be retained. If the ADC needs to be reused, only the trigger signal needs to be issued.

For enabling the ADC module, the following operations need to be performed in the CKGEN module:

1. Enable ADC clock
2. Disable ADC reset signal

For related register configuration, refer to section 4.3.2 and 4.3.5.

8.5.2 ADC power-on delay

As described in section 8.4.1, when the ADC module powers on for the first time, it takes about 100μs to stabilize the internal signal of the module. When the system is in low power mode(STOP) and when the ADC conversion is completed, the ADC will automatically shut down. If the ADC is configured to work in this mode, when the ADC receives a signal that triggers conversion, the ADC has a hardware automatic power-on process, and there is also a 100μs power-on delay for the ADC at this time.

8.6 Register definition

Table 8-6 ADC register map

ADC0 base address = 0x40003000

Analog base address = 0x40008800

Address	Register name	Width	Description
ADCx base address+0x0	ADC_STR	32	Status register
ADCx base address+0x4	ADC_CTRL0	32	Control register 0
ADCx base address+0x8	ADC_CTRL1	32	Control register 1
ADCx base address+0xC	ADC_SPT0	32	Sample time selection register 0
ADCx base address+0x10	ADC_SPT1	32	Sample time selection register 1
ADCx base address+0x14	ADC_IOFR0	32	Injected group offset register 0
ADCx base address+0x18	ADC_IOFR1	32	Injected group offset register 1
ADCx base address+0x1C	ADC_IOFR2	32	Injected group offset register 2
ADCx base address+0x20	ADC_IOFR3	32	Injected group offset register 3
ADCx base address+0x24	ADC_AMOHR	32	AMO high threshold register
ADCx base address+0x28	ADC_AMOLR	32	AMO low threshold register
ADCx base address+0x2C	ADC_RSQR0	32	Regular group sequence configuration register 0
ADCx base address+0x30	ADC_RSQR1	32	Regular group sequence configuration register 1
ADCx base address+0x34	ADC_RSQR2	32	Regular group sequence configuration register 2
ADCx base address+0x38	ADC_ISQR	32	Injected group sequence configuration register
ADCx base address+0x3C	ADC_IDR0	32	Injected group data register 0
ADCx base address+0x40	ADC_IDR1	32	Injected group data register 1
ADCx base address+0x44	ADC_IDR2	32	Injected group data register 2
ADCx base address+0x48	ADC_IDR3	32	Injected group data register 3
ADCx base address+0x4C	ADC_RDR0	32	Regular group data register 0
ADCx base address+0x50	ADC_RDR1	32	Regular group data register 1
ADCx base address+0x54	ADC_RDR2	32	Regular group data register 2
ADCx base address+0x58	ADC_RDR3	32	Regular group data register 3
ADCx base address+0x5C	ADC_RDR4	32	Regular group data register 4
ADCx base address+0x60	ADC_RDR5	32	Regular group data register 5
ADCx base address+0x64	ADC_RDR6	32	Regular group data register 6
ADCx base address+0x68	ADC_RDR7	32	Regular group data register 7
ADCx base address+0x6C	ADC_RDR8	32	Regular group data register 8
ADCx base address+0x70	ADC_RDR9	32	Regular group data register 9
ADCx base address+0x74	ADC_RDR10	32	Regular group data register 10
ADCx base address+0x78	ADC_RDR11	32	Regular group data register 11
ADCx base address+0x7C	ADC_RDR12	32	Regular group data register 12
ADCx base address+0x80	ADC_RDR13	32	Regular group data register 13
ADCx base address+0x84	ADC_RDR14	32	Regular group data register 14
ADCx base address+0x88	ADC_RDR15	32	Regular group data register 15
ADCx base address+0x8C	ADC_RDR16	32	Regular group data register 16
ADCx base address+0x90	ADC_RDR17	32	Regular group data register 17
ADCx base address+0x94	ADC_RDR18	32	Regular group data register 18
ADCx base address+0x98	ADC_RDR19	32	Regular group data register 19
ADCx base address+0x9C	ADC_RDR20	32	Regular group data register 20

Address	Register name	Width	Description
ADCx base address+0xCC	ADC_CGV	32	Gain error value calibration register
ADCx base address+0xD0	ADC_COV	32	Offset error value calibration register
ADCx base address+0xD4	ADC_REOC	32	Regular group conversion end status register
ADCx base address+0xD8	ADC_REOCEN	32	Regular group conversion end interrupt enable register
ADCx base address+0xDC	ADC_IEOC	32	Injected group conversion end status register
ADCx base address+0xE0	ADC_IEOCEN	32	Injected group conversion end interrupt enable register
ADCx base address+0xE4	ADC_RSQR3	32	Regular group sequence configuration register 3
ADCx base address+0x40	ADC_CFG0	32	Analog configuration register 0
ADCx base address+0x44	ADC_CFG1	32	Analog configuration register 1

8.6.1 Status Register(ADC_STR)

Table 8-7 ADC_STR register

ADC_STR **ADC status register** **Reset: 0x00000010**

Bit	31~7	6	5	4	3	2	1	0
Name		AAMO	NAMO	IDLE				AMO
Type		R/W0C	R/W0C	RO				R/W0C
Reset		0	0	1				0

Field	Description
6 AAMO	**Analog monitor event occurs (edge trigger mode)** 0: no analog monitor event 1: analog monitor event occurs, write 0 to clear
5 NAMO	**Analog monitor recovery event occurs (edge trigger mode)** 0: no recovery event 1: recovery event occurs, write 0 to clear.
4 IDLE	**ADC idle state flag** 0: ADC not idle 1: ADC idle
0 AMO	**Analog monitor event occurs (level trigger mode)** 0: no analog monitor event 1: analog monitor event occurs, write 0 to clear

8.6.2 Control Register 0(ADC_CTRL0)

Table 8-8 ADC_CTRL0 register

ADC_CTRL0 **ADC control register 0** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	SWSTART	ISWSTART						INTERVAL	AMOMODE	ALIGN	IEXTTRIG	EXTTRIG		AMOIE		
Type	RW	RW						RW	RW	RW	RW	RW		RW		
Reset	0	0						0	0	0	0	0		0		
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	SCAN	CONT	DISCEN	IDISCEN	IAUTO	DISCNUM[2: 0]			AMOEN	IAMOEN	AMOSGL	AMOCH[4: 0]				
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Field	Description
31 SWSTART	**Software trigger for regular channels** Write 1 to trigger; read as 0
30 ISWSTART	**Software trigger for injected channels** Write 1 to trigger; read as 0
24 INTERVAL	**Interval mode(used only in Mode3/5)** 0: injected group is scan mode 1: inject group is interval mode
23 AMOMODE	**Analog monitor trigger mode** 0: level trigger mode 1: edge trigger mode
22 ALIGN	**Data alignment** 0: right alignment. 1: left alignment.
21 IEXTTRIG	**Injected group trigger source selection** 0: internal (software trigger) 1: external
20 EXTTRIG	**Regular group trig source select** 0: internal (software trigger)

Field	Description
	1: external
18: AMOIE	**AMO interrupt function enable** 0: disable 1: enable
15: 11 Modes control bits	**ADC operation mode** For detailed configuration, refer to Table 8-1.
10: 8 DISCNUM	**Mode 7 regular group subgroup length** 0 ~ 7: the subgroup length under mode 7, 1~8 respectively.
7: 5 Analog monitor control bits	**Analog monitor function configuration** For detailed configuration, refer to Table 8-3.
4: 0 AMOCH	**Analog monitor channel** Specify the monitored channel when analog monitor is configured to detect just single channel.

8.6.3 Control Register 0(ADC_CTRL1)

Table 8-9 ADC_CTRL1 register

ADC_CTRL1 **ADC control register 1** **Reset: 0x0000F002**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	PSC[3：0]														CALEN	ADON
Type	RW	RW	RW	RW											RW	RW
Reset	1	1	1	1											1	0

Field	Description
15: 12 PSC	**ADC clock Prescaler** 0 ~ 15 : 1 ~ 16 divisor
1 CALEN	**Calibration enable** 0: disable calibration 1: enable calibration Note that this function should be kept enable to assure the accuracy of ADC.
0 ADON	**ADC Power on** 0: ADC power down and reset the ADC (but the configure registers are not be reset).

Field	Description
	1: ADC power on

8.6.4 Sample Time Selection Register 0(ADC_SPT0)

Table 8-10 ADC_SPT0 register

ADC_SPT0 **ADC Sample time register 0** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name			SPT19[2：0]			SPT18[2：0]						SPT16[2：0]			SPT15[2:0]	
Type			RW													
Reset			0													
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name		SPT14[2：0]			SPT13[2：0]			SPT12[2：0]			SPT11[2：0]			SPT10[2：0]		
Type		RW														
Reset		0														

Field	Description
29: 0 SPTx(x=10 ~ 19)	**Sample time selection for each channels** The numbering rule of SPTx is: 0~18: external channels 0~18 (without 17) 19: internal voltage channel The meaning of SPTx code is as below: 000b：9 ADCCLK 001b：7 ADCCLK 010b：17 ADCCLK 011b：33 ADCCLK 100b：64 ADCCLK 101b：140 ADCCLK 110b：215 ADCCLK 111b：5 ADCCLK

8.6.5 Sample Time Selection Register 1(ADC_SPT1)

Table 8-11 ADC_SPT1 register

ADC_SPT1 **ADC Sample time register1** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name			SPT9[2：0]			SPT8[2：0]			SPT7[2：0]			SPT6[2：0]			SPT5[2:0]	
Type			RW													
Reset			0													
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name		SPT4[2：0]			SPT3[2：0]			SPT2[2：0]			SPT1[2：0]			SPT0[2：0]		

Type		RW
Reset		0

Field	Description
29: 0 SPTx (x=0 ~ 9)	**Sample time selection for each channels** The numbering rule of SPTx is: 0~18: external channels 0~18 (without 17) 19: internal voltage channel The meaning of SPTx code is as below: 000b: 9 ADCCLK 001b: 7 ADCCLK 010b: 17 ADCCLK 011b: 33 ADCCLK 100b: 64 ADCCLK 101b: 140 ADCCLK 110b: 215 ADCCLK 111b: 5 ADCCLK

8.6.6 Injected Group Offset Register(ADC_IOFRx)

Table 8-12 ADC_IOFRx (x=0~3)register

ADC_IOFRx（x= 0 ~ 3） **ADC injected group offset Register** **Reset: 0x00000000**

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

Field	Description
11: 0 IOFR	**Injected group offset value** It is used to subtract the corresponding value of this register from the injected group conversion result. Injected group channels conversion values IDFR has been minus by offset value defined in the register ADC_IOFR.

8.6.7 AMO High Threshold Register(ADC_AMOHR)

Table 8-13 ADC_AMOHR register

ADC_AMOHR **ADC AMO high threshold register** **Reset: 0x00000000**

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

Field	Description
27: 16 AMOHO	**Recovery offset corresponds to the High threshold value for analog monitor** Define offset of the high threshold value .
11: 0 AMOHT	**High threshold value for analog monitor** Define the high threshold value .

8.6.8 AMO Low Threshold Register(ADC_AMOLR)

Table 8-14 ADC_AMOLR register

ADC_AMOLR **ADC AMO low threshold register** **Reset: 0x00000000**

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

Field	Description
27: 16 AMOLO	**Recovery offset corresponds to the Low threshold value for analog monitor** Define offset of the low threshold value
11: 0 AMOLT	**Low threshold value for analog monitor** Define the low threshold value .

8.6.9 Regular Group Sequence Configuration Register 0(ADC_RSQR0)

Table 8-15 ADC_RSQR0 register

ADC_RSQR0 ADC regular group sequence configuration register 0 Reset: 0x000FFFFF

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name								RSQL[4：0]					RSQ15[4：0]			
Type								RW					RW			
Reset								0					1	1	1	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name		RSQ14[4：0]					RSQ13[4：0]					RSQ12[4：0]				
Type		RW					RW					RW				
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Field	Description
24: 20 RSQL	**Length of the regular group** 0 ~ 20: define regular group length as 1~21. **Note:** length must be less than the number of actual valid regular group sequence, the length is invalid when the length is higher than 20.
19: 0 RSQx(x=12~15)	**Channel selection for regular group** 0~18: external channels 0~18 19: internal voltage channel Other values are not used.

8.6.10 Regular Group Sequence Configuration Register 1(ADC_RSQR1)

Table 8-16 ADC_RSQR1 register

ADC_RSQR1 ADC regular group sequence configuration register 1 Reset: 0x3FFFFFFF

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name			RSQ11[4：0]					RSQ10[4：0]					RSQ9[4：0]			
Type			RW													
Reset			1	1	1	1	1	1	1	1	1	1	1	1	1	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name		RSQ8[4：0]					RSQ7[4：0]					RSQ6[4：0]				
Type		RW														
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Field	Description
29: 0 RSQx(x=6~11)	**Channel selection for regular group** 0~18: external channels 0~18 (without 17) 19: internal voltage channel Other values are not used.

8.6.11 Regular Group Sequence Configuration Register 2(ADC_RSQR2)

Table 8-17 ADC_RSQR2 register

ADC_RSQR2 ADC regular group sequence configuration register 2 Reset: 0x3FFFFFFF

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name			RSQ5[4：0]					RSQ4[4：0]					RSQ3[4：0]			
Type			RW					RW					RW			
Reset			1	1	1	1	1	1	1	1	1	1	1	1	1	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name		RSQ2[4：0]					RSQ1[4：0]					RSQ0[4：0]				
Type		RW					RW					RW				
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Field	Description
29: 0 RSQx(x=0~5)	**Channel selection for regular group** 0~18: external channels 0~18 (without 17) 19: internal voltage channel Other values are not used.

8.6.12 Injected Group Sequence Configuration Register(ADC_ISQR)

Table 8-18 ADC_ISQR register

ADC_ISQR ADC injected group sequence configure register Reset: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name											ISQL[1：0]		ISQ3[4：0]			
Type											RW					
Reset											0	0	1	1	1	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name		ISQ2[4：0]					ISQ1[4：0]					ISQ0[4：0]				
Type		RW														
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	Description
21: 20 ISQL	**Length of the injected group** 0 ~ 3: define the injected group length as 1~4. **Note:** The length must be less than the number of actual valid injected group sequence.
19: 0 ISQx(x=0~3)	**Channel selection for injected group** 0~18: external channels 0~18 (without 17) 19: internal voltage channel Other values are not used.

8.6.13 Injected group data register(ADC_IDRx)

Table 8-19 ADC_IDRx(x=0~3) register

ADC_IDRx (x=0 ~ 3) **ADC injected group data register** **Reset: 0x00000000**

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

Field	Description
15: 0 IDR	**Data registers for injected group** **Note:** **1. ADC_IDR has minuses the offset defined in the ADC_IOFR register.** **2. Refer to section 8.4.7 for data arrangement under different alignment modes.**

8.6.14 Regular group data register(ADC_RDRx)

Table 8-20 ADC_RDRx register

ADC_RDRx (x=0 ~ 20) **ADC regular group data register** **Reset: 0x00000000**

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

Field	Description
15: 0 RDR	**Data register for regular group** **Note: refer to section 8.4.7 for data arrangement under different alignment modes.**

8.6.15 Gain Error Value Calibration Register(ADC_CGV)

Table 8-21 ADC_CGV register

ADC_CGV **ADC gain error value calibration register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name					GE1											
Type					RW											
Reset					0x0											
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name					GE0											
Type					RW											
Reset					0x0											

Field	Description
27:16 GE1	**Gain error of internal channels** Save in the format of complementary code, the highest bit is the sign bit.
11:0 GE0	**Gain error of external channels** Save in the format of complementary code, the highest bit is the sign bit.

8.6.16 Offset Error Value Calibration Register(ADC_COV)

Table 8-22 ADC_COV register

ADC_COV **ADC offset error value calibration register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name						OE1										
Type						RW										
Reset						0x0										
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name						OE0										
Type						RW										
Reset						0x0										

Field	Description
26:16 OE1	**Offset error of internal channels** Save in the format of complementary code, the highest bit is the sign bit.
10:0 OE0	**Offset error of external channels** Save in the format of complementary code, the highest bit is the sign bit.

8.6.17 Regular Group Conversion End Status Register(ADC_REOC)

Table 8-23 ADC_REOC register

ADC_COV **ADC Regular group conversion end status register** **Reset: 0x00000000**

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

Name												EOC20	EOC19	EOC18	EOC17	EOC16
Type												R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
Reset												0x0	0x0	0x0	0x0	0x0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	EOC15	EOC14	EOC13	EOC12	EOC11	EOC10	EOC9	EOC8	EOC7	EOC6	EOC5	EOC4	EOC3	EOC2	EOC1	EOC0
Type	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
Reset	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0

Field	Description
20:0 EOCx(x=0~20)	**End flag of regular group conversion** 0: regular group sequence conversion does not end, or the conversion does not start. 1: regular group sequence conversion ends. Write 1 to the corresponding bit to clear the flag.

8.6.18 Regular Group Conversion End Interrupt Enable Register (ADC_REOCEN)

Table 8-24 ADC_REOCEN register

ADC_REOCEN Regular group conversion end interrupt enable register Reset: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name												REOCIE20	REOCIE19	REOCIE18	REOCIE17	REOCIE16
Type												RW	RW	RW	RW	RW
Reset												0x0	0x0	0x0	0x0	0x0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	REOCIE15	REOCIE14	REOCIE13	REOCIE12	REOCIE11	REOCIE10	REOCIE9	REOCIE8	REOCIE7	REOCIE6	REOCIE5	REOCIE4	REOCIE3	REOCIE2	REOCIE1	REOCIE0
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0

Field	Description
20:0 REOCIEx(x=0~20)	**Regular group conversion end interrupt enable** 0: regular group sequence conversion end flag does not trigger interrupt. 1: regular group sequence conversion end flag triggers interrupt.

8.6.19 Injected Group Conversion End Status Register(ADC_IEOC)

Table 8-25 ADC_IEOC register

ADC_IEOC ADC injected group conversion end status register Reset: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name													IEOC3	IEOC2	IEOC1	IEOC0
Type													R/W1C	R/W1C	R/W1C	R/W1C
Reset													0x0	0x0	0x0	0x0

Field	Description
3:0 IEOCx(x=0~3)	**Injected group conversion end flag** 0: injected group sequence conversion does not end, or the conversion does not start. 1: injected group sequence conversion ends. Write 1 to the corresponding bit to clear the flag.

8.6.20 Injected Group conversion end interrupt enable register (ADC_IEOCEN)

Table 8-26 ADC_IEOCEN register

ADC_IEOCEN Injected group conversion end interrupt enable register Reset: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name													IEOCIE3	IEOCIE2	IEOCIE1	IEOCIE0
Type													RW	RW	RW	RW
Reset													0x0	0x0	0x0	0x0

Field	Description
3:0 IEOCIEx(x=0~3)	**Injected group conversion end interrupt enable** 0: injected group sequence conversion end flag does not trigger interrupt. 1: injected group sequence conversion end flag triggers interrupt.

8.6.21 Regular Group Sequence Configuration Register 3(ADC_RSQR3)

Table 8-27 ADC_RSQR3 register

ADC_RSQR3 ADC regular group sequence configuration register 3 Reset: 0x01FFFFFF

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name								RSQ20[4：0]					RSQ19[4：0]			
Type								RW					RW			
Reset								1	1	1	1	1	1	1	1	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name		RSQ18[4：0]					RSQ17[4：0]					RSQ16[4：0]				
Type		RW					RW					RW				
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Field	Description
24：0 RSQx(x=16~20)	**Channel selection for regular group** 0~18: external channels 0~18 (without 17) 19: internal voltage channel Other values are not used.

8.6.22 ADC Analog Configuration Register 0(ADC_CFG0)

Table 8-28 ADC_CFG0 register

ADC_CFG0 ADC analog configuration register 0 Reset: 0x02492000

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

Field	Description
29 VREF_SEL	**Reference source selection** 0: the VREF+/VREF- signal is sued as the reference source 1: AVDD is used as the reference source

Field	Description
6 VBUF_EN	**Bandgap and T-sensor internal channel enable** 0: disable internal channel 1: enable internal channel

8.6.23 ADC Analog Configuration Register 1(ADC_CFG1)

Table 8-29 ADC_CFG1 register

ADC_CFG1 **ADC analog configuration register 1** **Reset: 0x00000000**

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

Field	Description
31:30 MON_SEL	**Channel selection of internal voltage signal** 00b: Bandgap 01b: T-sensor Other values are not used.

9 Analog Comparator(ACMP)

9.1 Introduction

The analog comparator module(ACMP) provides a circuit for comparing two analog input voltages. The analog MUX provides a circuit for selecting an analog input signal from 9 channels: One channel is provided by a 6-bit DAC, and others by external input. The polling mode and hall output function are designed for motor application.

9.2 Features

- One ACMP module and one DAC module
- On-chip 6-bit DAC with selectable reference voltage from VDD or internal bandgap.
- Configurable hysteresis, supporting 10/20/40 mV.
- Selectable interrupt type on rising edge, falling edge, or both rising or falling edges of comparator output.
- Up to 9 selectable comparator inputs (ACMP_IN0~ACMP_IN7 and internal DAC).
- Support Stop mode wakeup.
- Support polling mode.
- Support ACMP comparison result output.
- Support DAC output.
- Support hall output.

9.3 Block diagram

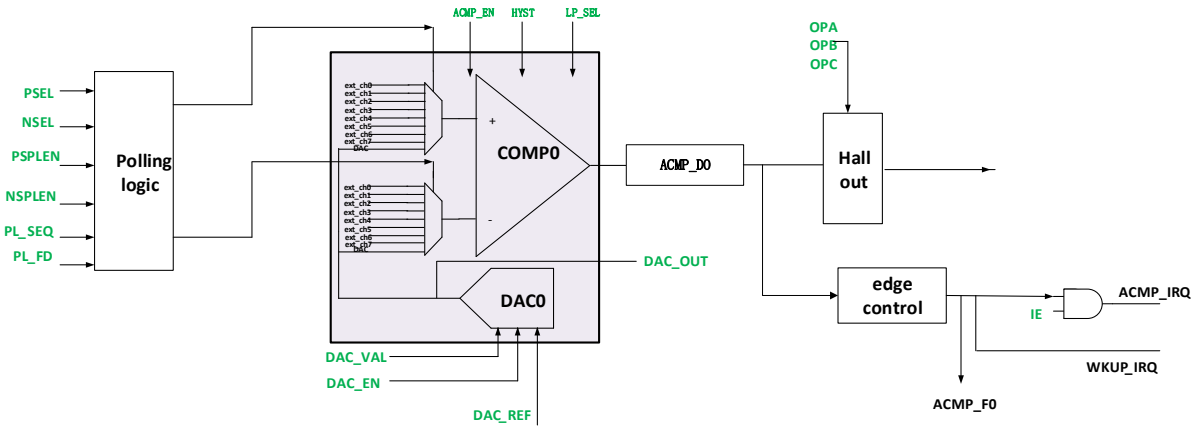

Figure 9-1 ACMP block diagram

9.4 Functional description

The ACMP module is functionally composed of two parts: digital-to-analog converter (DAC) and comparator (CMP).

The DAC includes a 64-level DAC (digital to analog converter) and related control logic. Through configuring ACMP_CR2[DACREF], DAC can select one of two reference inputs, Vdd or on-chip bandgap reference source as the DAC input source . After the DAC is enabled, it converts the data set in ACMP_CR2[DACVAL] to a stepped analog output, feeding to the input for comparison. The DAC voltage can also be output to an external pin simultaneously.

The ACMP can achieve the analog comparison between positive input and negative input, and then give out a digital output and interrupt. Both the positive input and negative input of analog comparator can be selected from the 9 common inputs.

9.4.1 Normal mode

In normal mode, positive and negative inputs are compared with fixed selected channels, the comparison result appears as a digital output. If the positive input voltage is higher than the negative input voltage, it will output 1, otherwise it will output 0.

Whenever a valid edge defined in the output occurs, the SR status bit is asserted.

If IE is set, an interrupt occurs.

The ACMP output is synchronized by the bus clock to generate the comparison result, so that CPU can read the comparison.

The data register changes following the comparison result, so it can serve as a tracking flag that continuously indicates the voltage delta on the inputs.

9.4.2 Polling mode

In polling mode, ACMP can switch the input channel for comparator's positive or negative input by logic control dynamic switch. The switch sequence is defined in ACMP_CR4[PLSEQ], and the frequency is controlled by ACMP_FD[PLFD]. ACMP_CR3[PSPLEN] and ACMP_CR3[NSPLEN] are the enable bits of polling mode, ACMP_CR3[PSPLEN] and ACMP_CR3[NSPLEN] cannot be enabled simultaneously. Both ACMP_CR3[PSPLEN] and ACMP_CR3[NSPLEN] enabled will not trigger the polling mode. So software must ensure that one of the two fields above is enabled.

This page gives an example of polling mode. Set ACMP positive input polling, polling frequency is source_clk/67, external channel 1-4 and DAC outputs are used as polling channel, ACMP negative input selects external input 0, falling edge triggers interrupt.

Step 1: ACMP_CR0[IE] = 1'b1, ACMP_CR0[MOD] = 2'b00;

Step 2: ACMP_CR2[DACEN] = 1'b1, ACMP_CR2[DACVAL]=value;

Step 3: ACMP_CR3[PSPLEN] = 1'b1, ACMP_CR3[NSPLEN] = 1'b0;

Step 4: ACMP_FD[PLFD] = 2'b01, 错误！未找到引用源。[PLSEQ] = 9'b100011110, ACMP_CR1[NSEL] = 4'b0000;

Step 5: ACMP_CR0[EN] = 1'b1.

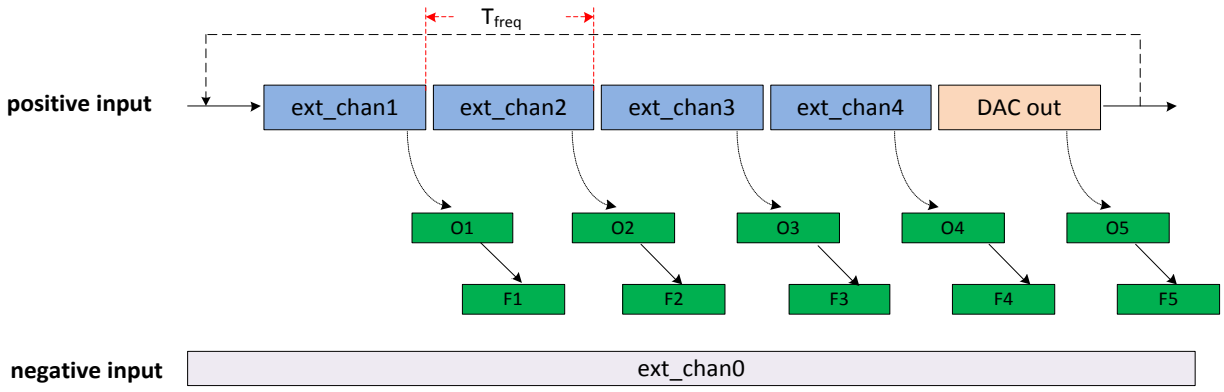

Figure 9-2 Operation flow in polling mode

9.4.3 HALL output in polling mode

ACMP has 3 hall outputs: Hall A Output, Hall B Output and Hall C Output. These signals are connected internally to PWDT. Hall output cooperates with polling function to obtain the position of the sensorless motor (The position of the rotor of the motor is detected by the BEMF). Every hall output can be selected one of the nine channels with polling mode.

For example, if polling mode ACMP_CR4[PLSEQ] = 9'b000001110, the polling sequence is external input 1- >external input 2 -> external input 3.

Set ACMP_OPA[OPASEL] = 4'b0010, then Hall A Output is ACMP_DR[O2].

9.4.4 Hysteresis

ACMP supports one hysteresis mode, as shown in Figure 9-3.

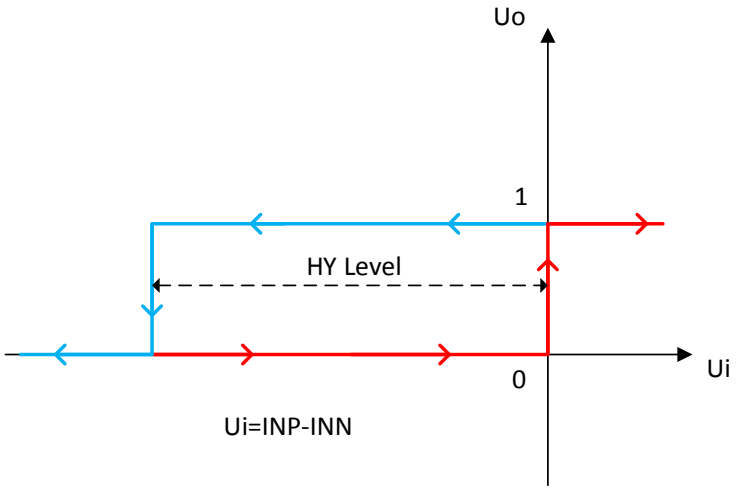

Figure 9-3 Hysteresis

9.4.5 DAC output

DAC output method support 3 ways:

- Only output to ACMP as internal signal, it will not output to pin
- Output to pin directly, not go through internal buffer
- Output to pin through internal buffer

The register configuration of above method is illustrated as below:

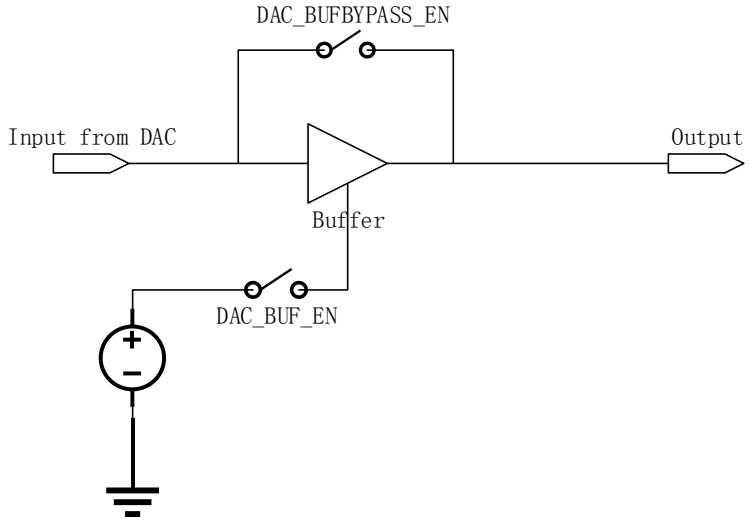

Figure 9-4 DAC output configuration

9.4.6 Low power mode wakeup

In stop mode, a valid edge on ACMP output generates an interrupt that can wake the MCU from low power mode. This interrupt can be cleared by writing 1 to WPF.

Note

The wake-up function only works in normal mode, and the polling mode does not work in low-power mode.

9.5 Register definition

Table 9-1 ACMP register map

ACMP0 base address:0x40005000

Analog base address:0x40008800

Address	Register name	Width	Description
ACMPx base address+0x0	ACMP_CR0	32	Configuration register 0
ACMPx base address+0x4	ACMP_CR1	32	Configuration register 1
ACMPx base address+0x8	ACMP_CR2	32	Configuration register 2
ACMPx base address+0xC	ACMP_CR3	32	Configuration register 3
ACMPx base address+0x10	ACMP_CR4	32	Configuration register 4

ACMPx base address+0x14	ACMP_DR	32	Data output register
ACMPx base address+0x18	ACMP_SR	32	Status register
ACMPx base address+0x1C	ACMP_FD	32	Polling frequency divider register
ACMPx base address+0x20	ACMP_OPA	32	Hall A output setting register
ACMPx base address+0x24	ACMP_OPB	32	hall B output setting register
ACMPx base address+0x28	ACMP_OPC	32	hall C output setting register
ACMPx base address+0x2C	ACMP_DACSR	32	DAC reference source select register
Analog base address+0x30	ACMP_CFG	32	Analog configuration register

9.5.1 Configuration Register 0(ACMP_CR0)

Table 9-2 ACMP_CR0 register

ACMP_CR0 **Configuration register 0** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									EN			IE	OUTEN	OPE	MOD	
Type									RW			RW	RW	RW	RW	
Reset									0			0	0	0	0	

Field	Description
7 EN	**ACMP Enable** 0: disable 1: enable
4 IE	**Interrupt enable** 0: disable 1: enable
3 OUTEN	**The comparison result outputs to the external PIN** 0: disable 1: enable
2 OPE	**Hall output enable** 0: disable 1: enable
1: 0 MOD	**Interrupt trigger mode** 00b: interrupt on output falling edge. 01b: interrupt on output rising edge. 10b: interrupt on output falling edge(00 and 10 are of the same configuration). 11b: interrupt on output falling or rising edge.

9.5.2 Configuration Register 1(ACMP_CR1)

Table 9-3 ACMP_CR1 register

ACMP_CR1 **Configuration register 1** **Reset:00000000**

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

Field	Description
7: 4 PSEL	**Positive input select** 0000b: external input 0 0001b: external input 1 0010b: external input 2 0011b: external input 3 0100b: external input 4 0101b: external input 5 0110b: external input 6 0111b: external input 7 1000b: DAC output Other values are invalid.
3: 0 NSEL	**Negative input select** 0000b: external input 0 0001b: external input 1 0010b: external input 2 0011b: external input 3 0100b: external input 4 0101b: external input 5 0110b: external input 6 0111b: external input 7 1000b: DAC output Other values are invalid.

9.5.3 Configuration Register 2(ACMP_CR2)

Table 9-4 ACMP_CR2 register

ACMP_CR2 **Configuration register 2** **Reset:00000000**

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

Field	Description
7 DACEN	**DAC enable** 0: disable 1: enable
5: 0 DACVAL	**DAC output voltage**

9.5.4 Configuration Register 3(ACMP_CR3)

Table 9-5 ACMP_CR3 register

ACMP_CR3 **Configuration register 3** **Reset:00000000**

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

Field	Description
7 PSPLEN	**Positive input polling mode enable** 0: disable 1: enable **Note:** PSPLEN and NSPLEN can't be enabled simultaneously. Both PSPLEN and NSPLEN enabled will not trigger the polling mode.
3 NSPLEN	**Negative input polling mode enable**

Field	Description
	0: disable 1: enable **Note:** PSPLEN and NSPLEN can't be enabled simultaneously. Both PSPLEN and NSPLEN enabled will not trigger the polling mode.

9.5.5 Configuration Register 4(ACMP_CR4)

Table 9-6 ACMP_CR4 register

ACMP_CR4 **Configuration register 4** **Reset:00000000**

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

Field	Description
8: 0 PLSEQ	**Polling channel sequence setting** Bit0~8 represent external input channel 0~7 and DAC channel respectively. 0: disable corresponding channel 1: enable corresponding channel

9.5.6 Data Output Register(ACMP_DR)

Table 9-7 ACMP_DR register

ACMP_DR **Data output register** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name							O	O8	O7	O6	O5	O4	O3	O2	O1	O0
Type							RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset							0	0	0	0	0	0	0	0	0	0

Field	Description
9 O	**Normal mode output** 0: comparison result is 0 in normal mode 1: comparison result is 1 in normal mode

Field	Description
8 O8	**Polling mode channel 8 output** 0: comparison result of DAC channel is 0 1: comparison result of DAC channel is 1
7 O7	**Polling mode channel 7 output** 0: comparison result of external channel 7 is 0 1: comparison result of external channel 7 is 1
6 O6	**Polling mode channel 6 output** 0: comparison result of external channel 6 is 0 1: comparison result of external channel 6 is 1
5 O5	**Polling mode channel 5 output** 0: comparison result of external channel 5 is 0 1: comparison result of external channel 5 is 1
4 O4	**Polling mode channel 4 output** 0: comparison result of external channel 4 is 0 1: comparison result of external channel 4 is 1
3 O3	**Polling mode channel 3 output** 0: comparison result of external channel 3 is 0 1: comparison result of external channel 3 is 1
2 O2	**Polling mode channel 2 output** 0: comparison result of external channel 2 is 0 1: comparison result of external channel 2 is 1
1 O1	**Polling mode channel 1 output** 0: comparison result of external channel 1 is 0 1: comparison result of external channel 1 is 1
0 O0	**Polling mode channel 0 output** 0: comparison result of external channel 0 is 0 1: comparison result of external channel 0 is 1

9.5.7 Status register(ACMP_SR)

Table 9-8 ACMP_SR register

ACMP_SR **Status register** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name						WPF	F	F8	F7	F6	F5	F4	F3	F2	F1	F0
Type						R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
Reset						0	0	0	0	0	0	0	0	0	0	0

Field	Description
10 WPF	**Low power mode wakeup interrupt flag** 0: low power mode wakeup flag is not triggered 1: low power mode wakeup flag is triggered Write 1 clear the flag
9 F	**Normal mode interrupt flag** 0: normal mode edge flag is not triggered 1: normal mode edge flag is triggered Write 1 clear the flag
8 F8	**Polling mode channel 8 interrupt flag** 0: DAC channel edge flag is not triggered 1: DAC channel edge flag is triggered Write 1 clear the flag
7 F7	**Polling mode channel 7 interrupt flag** 0: external channel 7 edge flag is not triggered 1: external channel 7 edge flag is triggered Write 1 clear the flag
6 F6	**Polling mode channel 6 interrupt flag** 0: external channel 6 edge flag is not triggered 1: external channel 6 edge flag is triggered Write 1 clear the flag
5 F5	**Polling mode channel 5 interrupt flag** 0: external channel 5 edge flag is not triggered 1: external channel 5 edge flag is triggered Write 1 clear the flag
4 F4	**Polling mode channel 4 interrupt flag**

Field	Description
	0: external channel 4 edge flag is not triggered 1: external channel 4 edge flag is triggered Write 1 clear the flag
3 F3	**Polling mode channel 3 interrupt flag** 0: external channel 3 edge flag is not triggered 1: external channel 3 edge flag is triggered Write 1 clear the flag
2 F2	**Polling mode channel 2 interrupt flag** 0: external channel 2 edge flag is not triggered 1: external channel 2 edge flag is triggered Write 1 clear the flag
1 F1	**Polling mode channel 1 interrupt flag** 0: external channel 1 edge flag is not triggered 1: external channel 1 edge flag is triggered Write 1 clear the flag
0 F0	**Polling mode channel 0 interrupt flag** 0: external channel 0 edge flag is not triggered 1: external channel 0 edge flag is triggered Write 1 clear the flag

9.5.8 Polling Frequency Divider Register(ACMP_FD)

Table 9-9 ACMP_FD register

ACMP_FD **Polling frequency divider register** **Reset:00000000**

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

Field	Description
1: 0 PLFD	**Polling mode frequency divider** This divider controls the switch frequency of polling channels 00b: source_clk/171 01b: source_clk/67 10b: source_clk/47 11b: source_clk/34

9.5.9 Hall A Output Setting Register(ACMP_OPA)

Table 9-10 ACMP_OPA register

ACMP_OPA **Hall A output setting register** **Reset:00000000**

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

Field	Description
3: 0 OPASEL	**Hall A output setting** 0000b: polling channel 0 0001b: polling channel 1 0010b: polling channel 2 0011b: polling channel 3 0100b: polling channel 4 0101b: polling channel 5 0110b: polling channel 6 0111b: polling channel 7 1000b: polling DAC Other values are invalid.

9.5.10 Hall B Output Setting Register(ACMP_OPB)

Table 9-11 ACMP_OPB register

ACMP_OPB **Hall B output setting register** **Reset:00000000**

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

Field	Description
3: 0 OPBSEL	**Hall B output setting** 0000b: polling channel 0 0001b: polling channel 1

Field	Description
	0010b: polling channel 2 0011b: polling channel 3 0100b: polling channel 4 0101b: polling channel 5 0110b: polling channel 6 0111b: polling channel 7 1000b: polling DAC Other values are invalid.

9.5.11 Hall C Output Setting Register(ACMP_OPC)

Table 9-12 ACMP_OPC register

ACMP_OPC **Hall C output setting register** **Reset:00000000**

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

Field	Description
3: 0 OPCSEL	**Hall C output set** 0000b: polling channel 0 0001b: polling channel 1 0010b: polling channel 2 0011b: polling channel 3 0100b: polling channel 4 0101b: polling channel 5 0110b: polling channel 6 0111b: polling channel 7 1000b: polling DAC Other values are invalid.

9.5.12 DAC Reference Source Select Register(ACMP_DACSR)

Table 9-13 ACMP_DACSR register

ACMP_DACSR **DAC reference source select register** **Reset:00000000**

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

Field	Descriptions
0 DACREF	**DAC reference select** 0: The DAC selects bandgap as the reference source 1: The DAC selects Vdd as the reference source

9.5.13 Analog Configuration Register(ACMP_CFG)

Table 9-14 ACMP_ANACFG register

ACMP_CFG **ACMP analog configuration register** **Reset:01400000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name									HYS_SEL		LPF_FREQ_SEL				DAC_BUF_EN	DAC_BUFBYPASS_EN
Type									RW	RW	RW	RW			RW	RW
Reset									0	1	0	0			0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name																
Type																
Reset																

Field	Description
23:22 HYS_SEL	**Analog comparator 0 hysteresis voltage selection** 00b: no hysteresis 01b: 10mV 10b: 20mV 11b: 40mV
21: 20 LPF_FREQ_SEL	**LPF Select** 00b: 500kHz

Field	Description
	01b: 1MHz 10b: 2MHz 11b: no LPF
17 DAC_BUF_EN	**DAC internal buffer enable** **0: power off internal buffer** **1: power on internal buffer**
16 DAC_BUFBYPASS_EN	**Bypass internal DAC buffer** **0: DAC will go through internal buffer** **1: DAC will not go through internal buffer, but will output directly**

10 Pulse Width Modulation(PWM)

10.1 Introduction

The PWM module is a multi-functional timer that supports input capture, output compare, quadrature decoder mode and the generation of PWM signals to control electric motor and power management applications. The PWM time reference is a 16-bit counter. The device contains three PWM modules, the PWM0 and PWM1 module supports two channels, the PWM2 module supports four channels.

10.2 Features

The PWM features include:

- PWM source clock is the bus clock.
- 16-bit Prescaler divide-by 1 to 65536.
- 16-bit counter.
  - It can be a free-running counter with no limitation or a counter with initial and final value.
  - The counting can be up or up-down.
- Each channel can be configured for input capture, output compare, or edge-aligned PWM mode, center-aligned PWM mode.
- In input capture mode, Capture can occur on rising edge, falling edge, or rising/falling edge.
- In input capture mode, an input filter can be selected for all channels.
- In output compare mode, the output signal can be set, cleared, or toggled on match.
- Each pair of channels can be combined to generate a PWM signal with independent control of rising/falling edges of PWM signal.
- The PWM channels can operate as pairs with equal outputs, pairs with complementary outputs.
- The dead-time insertion is available for each pair of combined channel, the dead-time insertion can be done in complementary/non-complementary mode.
- Generation of match triggers.
- Software control of PWM outputs.
- Output mask can set the channel to invalid state.
- Up to 3 fault inputs for global fault control of each PWM module.
- The polarity of each channel is configurable.

- The generation of an interrupt per channel.
- The generation of an interrupt when the counter overflows.
- The generation of an interrupt when the fault condition is detected.
- Synchronized loading of write buffered PWM registers.
- Write protection for critical registers.
- Dual edge capture for pulse and period width measurement.
- Supports Quadrature decoder (The input pins of phase A and B are mapped to CH0 and CH1 of each PWM module).
- Supports global time base.
- Supports PWM output waveform phase shift.

10.3 Block diagram

The PWM uses one input/output (I/O) pin per channel, CHn (PWM channel (n)), where n is the channel number (PWM0 and PWM1 are two channel modules (n = 0/1), PWM2 are four channel module(n = 0/1/2/3)).

The following figure shows the PWM structure. The central component of the PWM is the 16-bit counter with programmable initial and final values and its counting can be up or up-down.

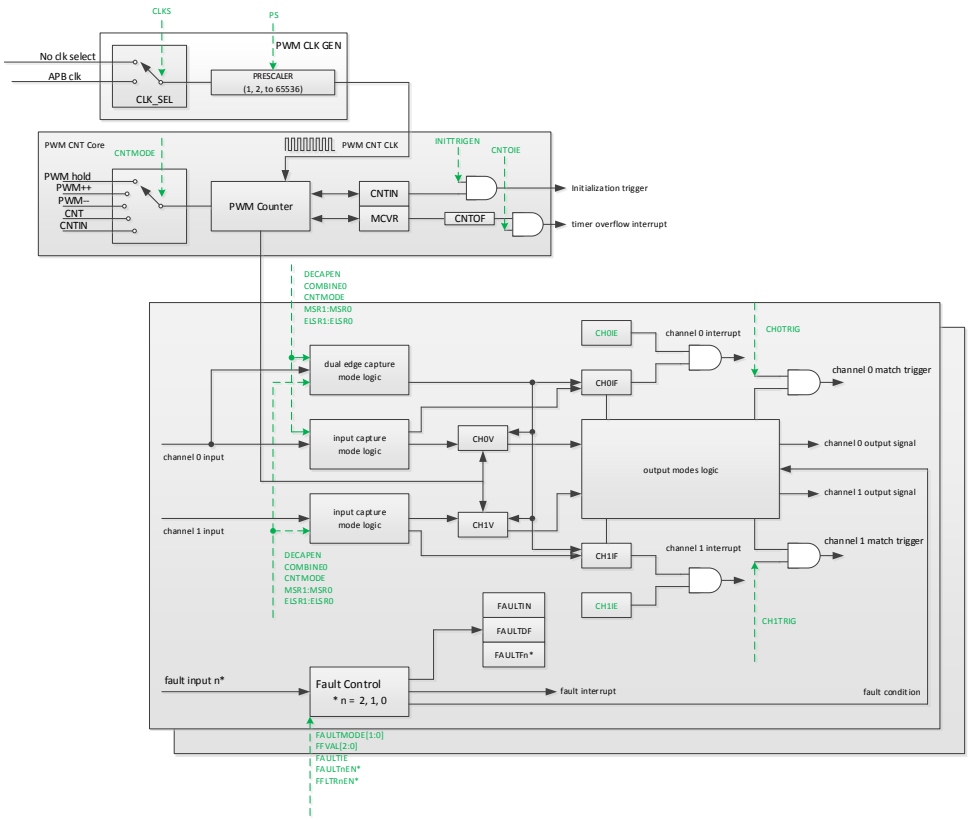

Figure 10-1 PWM block diagram

10.4 Functional description

10.4.1 Clock source

The CLKSRC bit in the PWM_INIT register select the clock source of the PWM counter as the APB clock or disable the PWM counter. After any MCU reset, CLKSRC = 0, so no clock source is selected. Disabling the PWM counter by writing 0 to the CLKSRC bit does not affect the PWM counter value or other registers.

10.4.2 Counter

The PWM has a 16-bit counter that is used by the channels either for input or output modes. The PWM counter clock is the selected clock divided by the prescaler. The PWM counter has the following modes of operation:

- Up counting
- Up-down counting
- Quadrature decoder mode

10.4.2.1 Up counting

Up counting is selected when (QDIEN=0) and (CNTMODE = 0). CNTIN defines the starting value of the count and MCVR defines the final value of the count, see the following figure. The value of CNTIN is loaded into the PWM counter, and the counter increments until the value of MCVR is reached, at which point the counter is reloaded with the value of CNTIN. The PWM period when using up counting is (MCVR – CNTIN + 0x0001) × period of the PWM counter clock. The CNTOF bit is set when the PWM counter changes from MCVR to CNTIN.

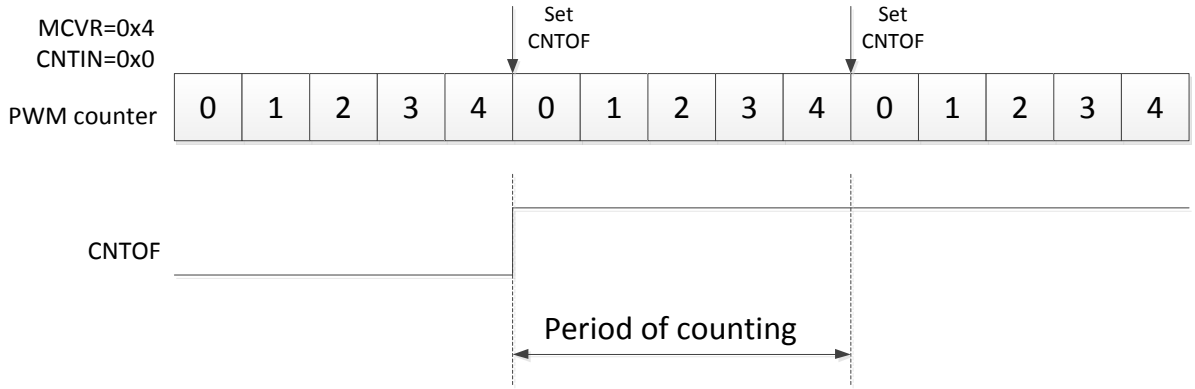

Figure 10-2 Up counting

10.4.2.2 Up-Down counting

Up-down counting is selected when (QDIEN=0) and (CNTMODE = 0). CNTIN defines the starting value of the count and MCVR defines the final value of the count. The value of CNTIN is loaded into the PWM counter, and the counter increments until the value of MCVR is reached, at which point the counter is decremented until it returns to the value of CNTIN and the up-down counting restarts.

The PWM period when using up-down counting is 2 × (MCVR – CNTIN) × period of the PWM counter clock. The CNTOF bit is set when the PWM counter changes from MCVR to (MCVR – 1). See the following figure.

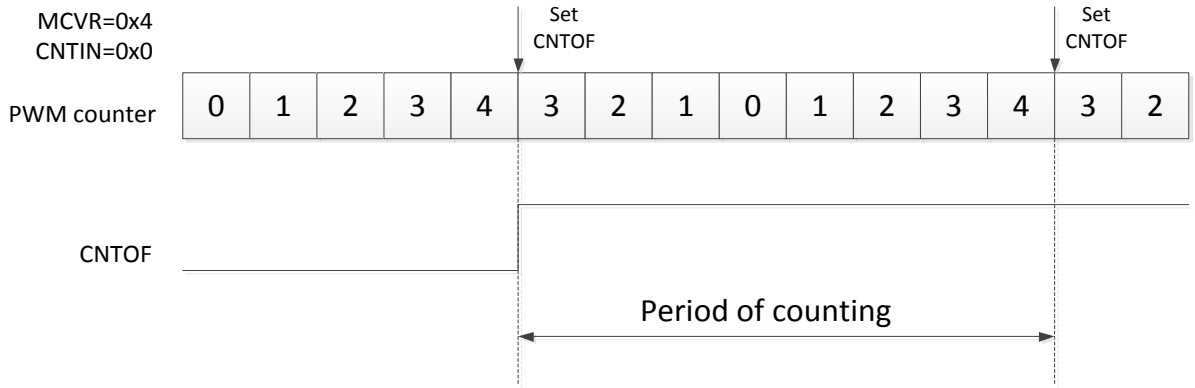

Figure 10-3 Up-Down counting

10.4.3 Operation mode

PWM can be configured for input capture, output compare, or edge-aligned PWM mode, center-aligned PWM mode, combination mode, Quadrature decoder mode. Please refer to the following table in detail.

Table 10-1 Operation mode configuration

QDIEN	DECAPEN	COMBINE	CNTMODE	MSR1:MSR0		ELSR1:ELSR0		Mode	Configuration
0	0	0	0	0	0	0	1	Input capture	Capture on rising edge
0	0	0	0	0	0	1	0		Capture on falling edge
0	0	0	0	0	0	1	1		Capture on rising or falling edge
0	0	0	0	0	1	0	1	Output compare	Toggle output on match
0	0	0	0	0	1	1	0		Clear output on match
0	0	0	0	0	1	1	1		Set output on match
0	0	0	0	1	X	1	0	Edge-aligned PWM	High-true pulses(clear output on match)
0	0	0	0	1	X	X	1		Low-true pulses(set output on match)
0	0	0	1	X	X	1	0	Center-aligned PWM	High-true pulses(clear output on match)
0	0	0	1	X	X	X	1		Low-true pulses(set output on match)
0	0	1	0	X	X	1	0	Combine mode+Up counting	Set output on channel(n) match Clear output on channel(n+1) match
0	0	1	0	X	X	X	1		Set output on channel(n) match Clear output on channel(n+1) match
0	0	1	1	X	X	1	0	Combine mode+Up-down counting	Set output on channel(n) match Clear output on channel(n+1) match
0	0	1	1	X	X	X	1		Set output on channel(n) match Clear output on channel(n+1) match
0	1	0	0	X	0	0	1	Dual edge single capture	Rising edge
0	1	0	0	X	0	1	0		Falling edge
0	1	0	0	X	0	1	1		Rising or fallling edge
0	1	0	0	X	1	0	1	Dual edge continuous capture	Rising edge
0	1	0	0	X	1	1	0		Falling edge
0	1	0	0	X	1	1	1		Rising or fallling edge
1	X	X	X	X	X	X	X	Quadrature decoder	QDIEN=1, enable Quadrature docoder,which is prior to the other modes

10.4.4 Input capture mode

When a selected edge occurs on the channel input, the current value of the PWM counter is captured into the PWM_CHnV register. At the same time, the CHnIF bit is set and the channel interrupt is generated if enabled by_CHnIE = 1. When a channel is configured for input capture, the PWMx_CHn pin is an edge-sensitive input. ELSnR1:ELSnR0 control bits determine which edge, falling or rising, triggers input capture event. Note that the maximum frequency for the channel input signal to be detected correctly is system clock divided by 4, which is required to meet Nyquist criteria for signal sampling. Writes to the CHnV register is ignored in input capture mode.

The following figure shows the channel rising edge capture.

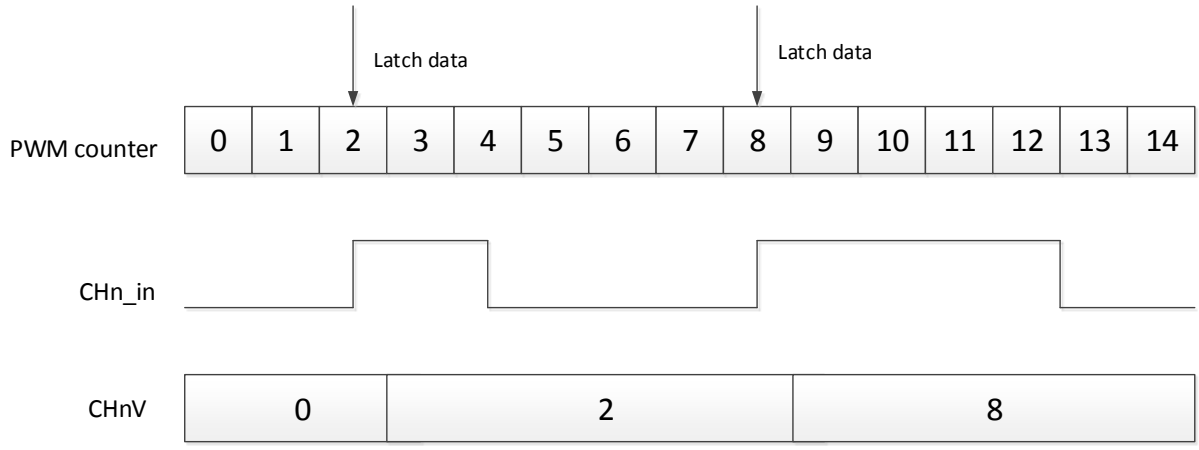

Figure 10-4 Input capture mode

For all PWM channels, there exist additional capture filter to make input signal clean. The filter is 5-bit counter and can be configured through the register CHnCAPFVAL(n = 0,1,2,3). When the CHnCAPFVAL[4:0] = 0, the filter function is disabled, if CHnCAPFVAL[4:0] ≠00000, the input signal will be delayed CHnCAPFVAL[4:0] × 4 system Clock, and then be transported to the input capture function.

The counter clock in the channel input filter is divided by 4 of the bus clock.

10.4.5 Output Compare mode

In output compare mode, the PWM can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter matches the value in the CHnV register of an output compare channel, the channel (n) output can be set, cleared, or toggled. When a channel is initially configured to Toggle mode, the previous value of the channel output is held until the first output compare event occurs. The CHnIF bit is set and the channel (n) interrupt is generated if CHnIE = 1 at the channel(n) match (PWM counter = CHnV).

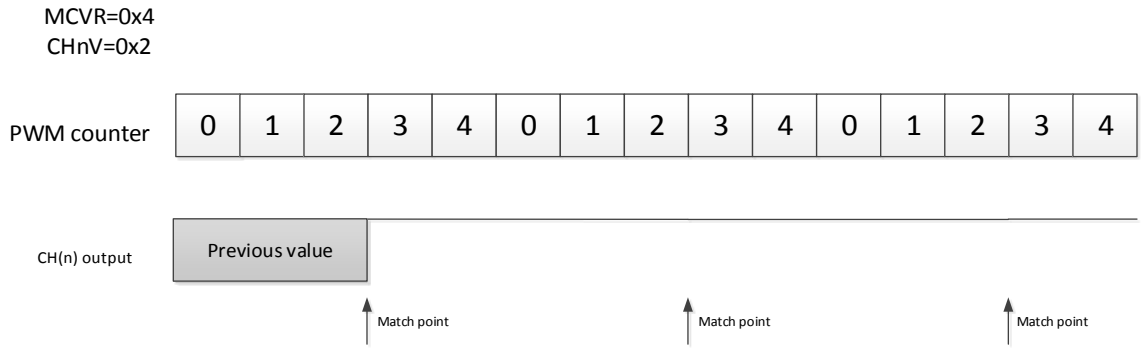

Figure 10-5 Match setting output compare mode

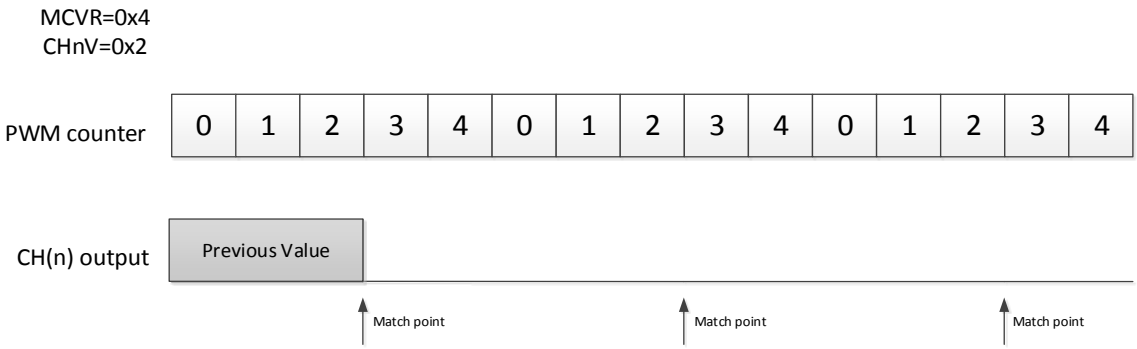

Figure 10-6 Match clear output compare mode

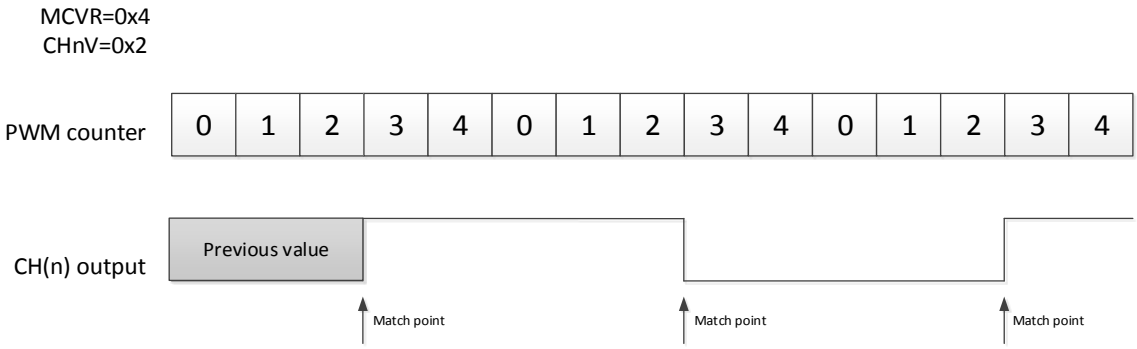

Figure 10-7 Match toggle output compare mode

10.4.6 Edge-Aligned PWM(EPWM) mode

The EPWM period = (MCVR - CNTIN + 0x0001) × PWM counter clock cycle.

The pulse width = (CHnV + 0x0001 - CNTIN) × PWM counter clock cycle.

ELSnR1: ELSnR0 = 1:0, the channel (n) output is forced high when the CNTIN value is loaded into the PWM counter, and forced low when channel (n) matches (PWM counter = CHnV).

ELSnR1: ELSnR0 = X:1, the channel (n) output is forced low when the CNTIN value is loaded into the PWM counter, and forced high when channel (n) matches (PWM counter = CHnV).

CHnIE = 1, when the channel(n) matches (PWM counter = CHnV), the CHnIF bit is set and the channel (n) interrupt is generated.

This type of PWM signal is called edge-aligned because the starting edges of all PWM signals are aligned with the beginning of the period, which is the same for all channels within an PWM.

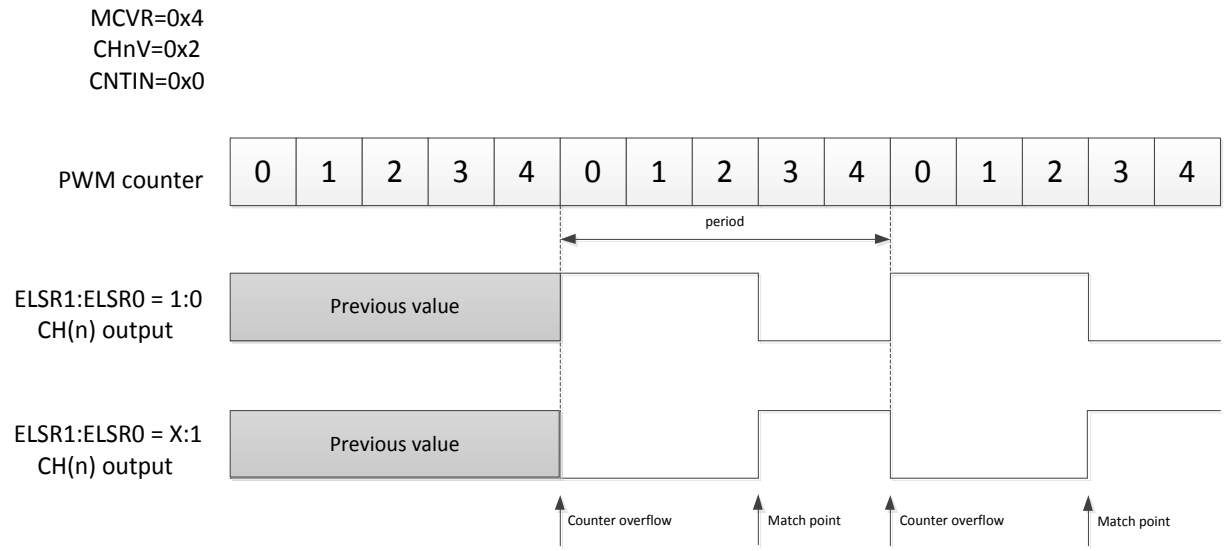

Figure 10-8 Edge-aligned PWM mode

ELSnR1: ELSnR0= 1:0, if (CHnV = 0x0000), the channel(n) output is 0% duty cycle of the EPWM signal; If (CHnV >= MCVR), the channel(n) output is 100% duty cycle of the EPWM signal. ELSnR1: ELSnR0=X: 1, the reverse happens.

10.4.7 Center-Aligned PWM(CPWM) mode

The CPWM period = 2 × (MCVR - CNTIN) × PWM counter clock cycle.

The CPWM pulse width = 2 × (CHnV - CNTIN) × PWM counter clock cycle.

In CPWM mode, the PWM counter counts up until MCVR is reached, and then counts down until CNTIN is reached.

ELSnR1: ELSnR0 = 1:0, the channel(n) output is forced to high level when it matches channel (n) while down counting (PWM counter = CHnV), and forced low when matching with channel(n) while counting up (PWM counter = CHnV).

ELSnR1: ELSnR0 = X:1, the channel (n) output is forced low when it matches channel (n) while down counting (PWM counter = CHnV), and forced high when matching with channel (n) while up counting (PWM counter = CHnV).

The CHnIF bit is set and channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (PWM counter = CHnV) when the PWM counting is down or when the PWM counting is up.

This type of PWM signal is called center-aligned because the pulse width centers for all channels are aligned with the value of CNTIN.

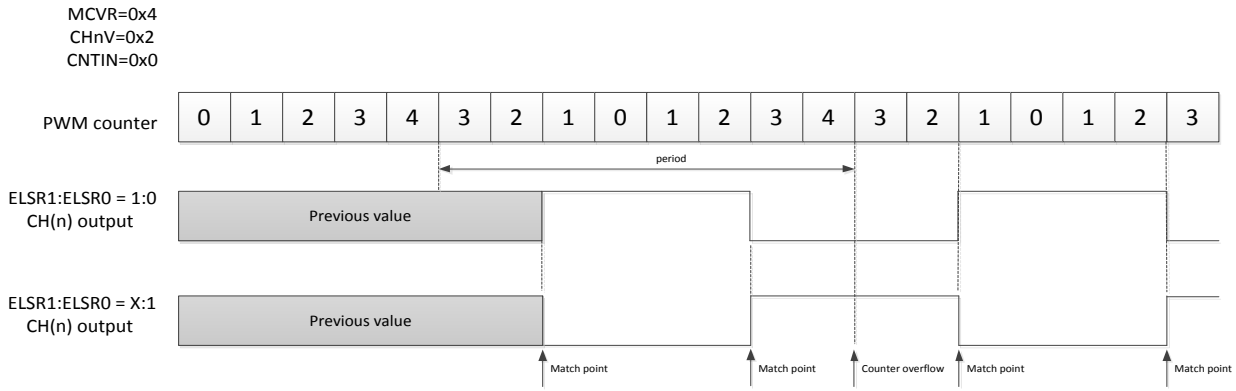

Figure 10-9 CPWM waveform

ELSnR1: ELSnR0= 1:0, if (CHnV = 0x0000), the channel (n) output is 0% duty cycle; If (CHnV >= MCVR), the channel (n) output is 100% duty cycle. ELSnR1: ELSnR0=X: 1, the reverse happens.

10.4.8 Combine mode

In Combine mode, an even channel (n) and adjacent odd channel (n+1) are combined to generate a PWM signal in the channel (n) output.

According to the counting methods, it can be divided into up counting combination mode and up-down counting combination mode.

If (ELSnR1:ELSnR0 = 1:0), then the channel (n) output is forced high at the channel(n) matches(PWM counter = CH(n)V). It is forced low at the channel(n+1) match.

If (ELSnR1:ELSnR0 = X:1), then the channel (n) output is forced low at the channel(n) matches (PWM counter = CH(n)V). It is forced high at the channel(n+1) match.

If CH(n)IE = 1, when the channel (n) matches (PWM counter = CH(n)V), the CH(n)IF bit is set and the channel (n) interrupt is generated.

If CH(n+1)IE = 1, when the channel (n+1) matches (PWM counter = CH(n+1)V), the CH(n+1)IF bit is set and the channel (n+1) interrupt is generated.

Note:
ELS(n+1)R1 and ELS(n+1)R0 cannot be used to control the output of the channels(n) and (n+1).

10.4.8.1 Up counting combine mode

Period = (MCVR - CNTIN + 0x0001) × PWM counter clock period.

Pulse width = |CH(n+1)V - CH(n)V| × PWM counter clock period.

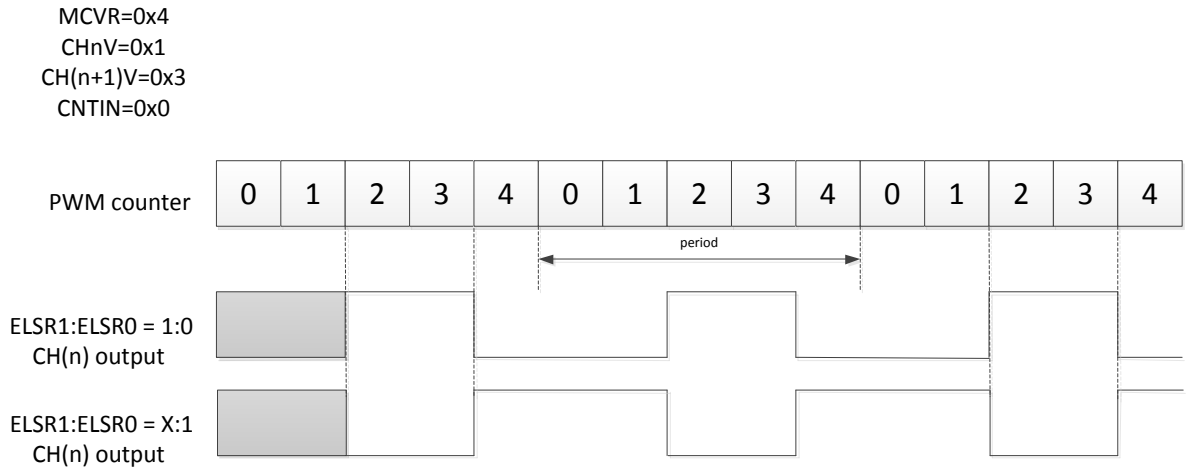

Figure 10-10 CH(n) output waveform in up counting mode

The following figures show the PWM signal output waveforms under various conditions of the up counting combine mode.

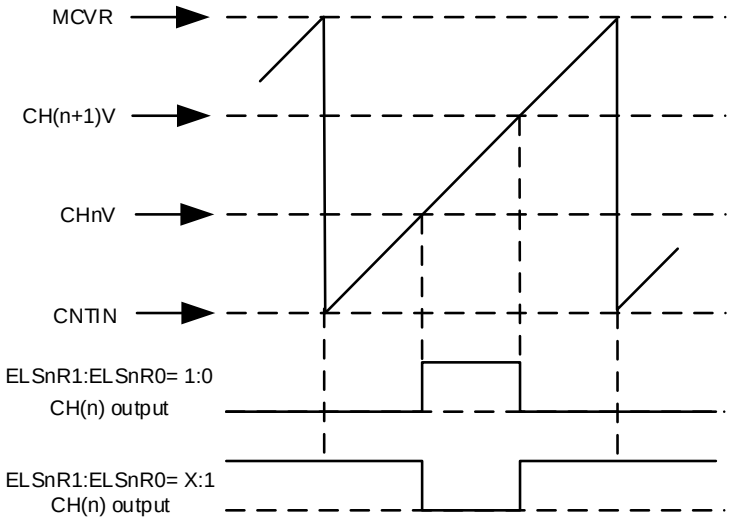

Figure 10-11 CH(n) output waveform if (CNTIN < CHnV/CH(n+1)V < MCVR) &(CHnV < CH(n+1)V)

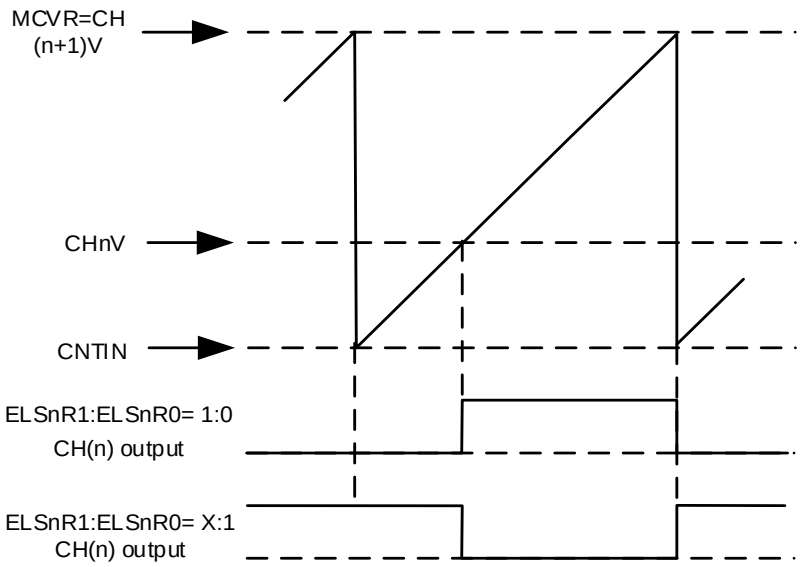

Figure 10-12 CH(n) output waveform if (CNTIN < CHnV < MCVR)&(CH(n+1)V = MCVR)

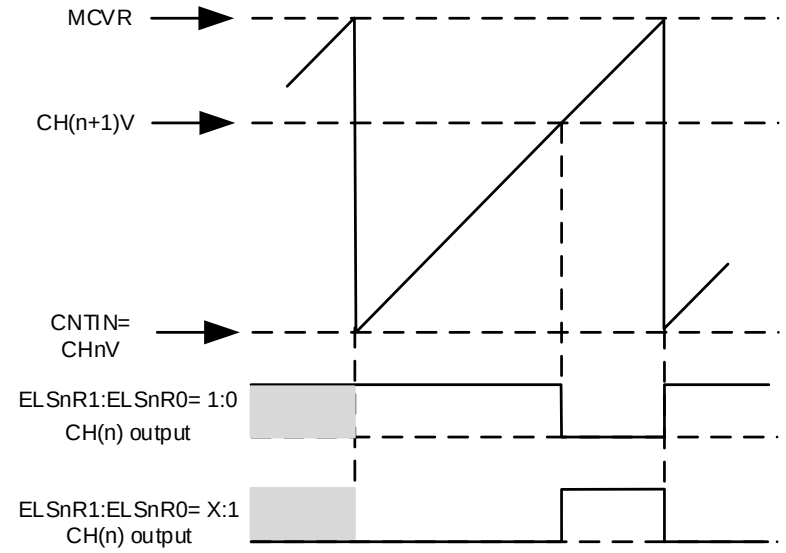

Figure 10-13 CH(n) output waveform if (CHnV = CNTIN)&(CNTIN < CH(n+1)V < MCVR)

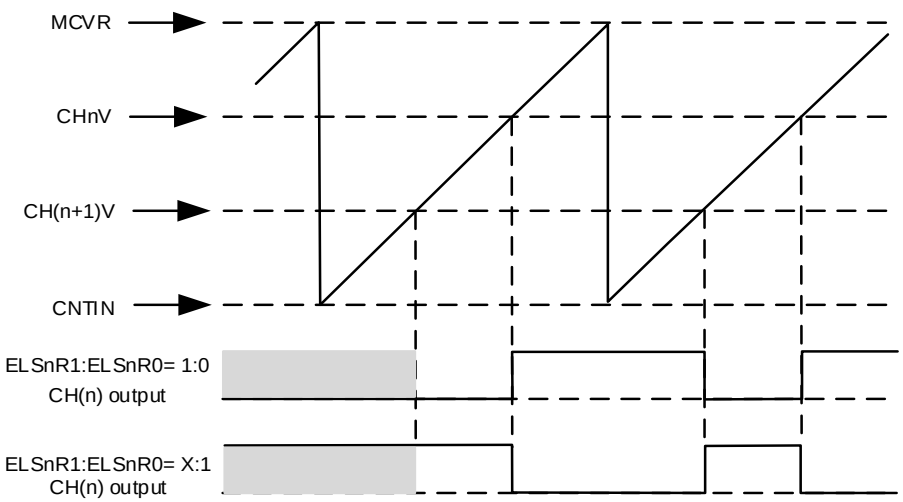

Figure 10-14 CH(n) output if (CNTIN < CHnV/ CH(n+1)V < MCVR) & (CHnV > CH(n+1)V)

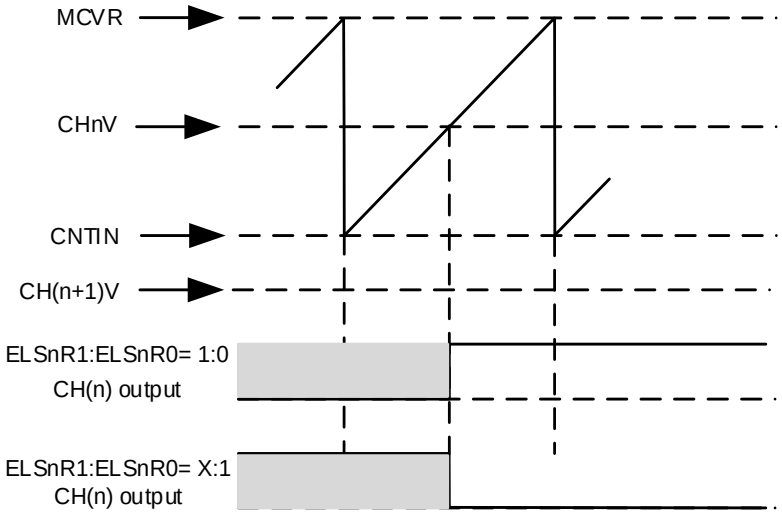

Figure 10-15 CH(n) output if (CH(n+1)V < CNTIN)&(CNTIN < CHnV < MCVR)

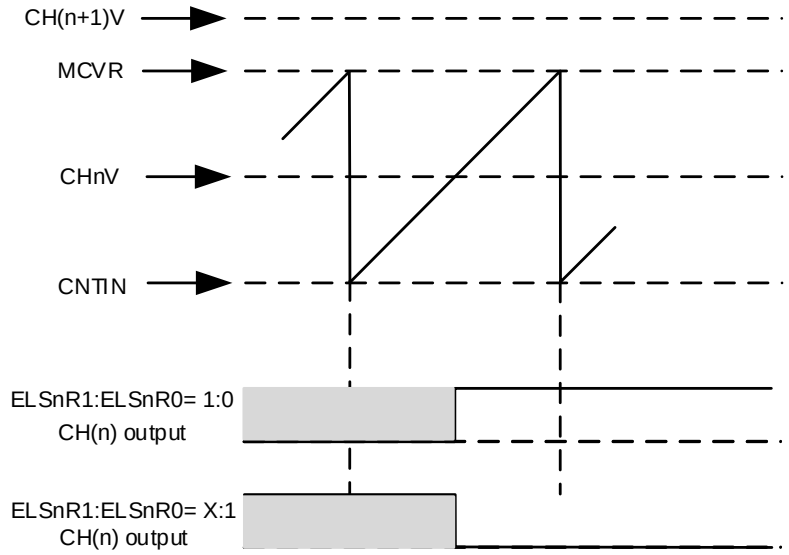

Figure 10-16 CH(n) output if (CH(n+1)V > MCVR) & (CNTIN < CHnV < MCVR)

10.4.8.2 Up-down counting combine mode

Period = 2 × (MCVR - CNTIN) × PWM timer clock cycle

During the up-down counting, the channel generates a match while up counting, and the down counting also generates a match. In order to facilitate the control of the channel output, the matching effective point setting function is provided, and the setting of the matching effective point CHSCR[DIR] can be used in the process of up counting or down counting. The matching effective point depends on the counting direction, it is necessary to clearly define the range of the up counting and down counting interval, as is shown in the following chart.

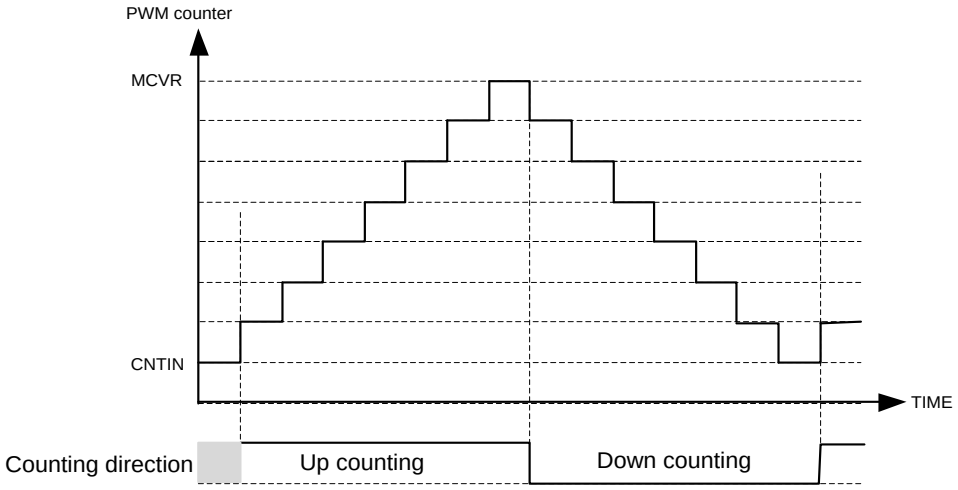

Figure 10-17 Up-down counting range define

Note:
The CNTIN value in the first counting period is the upward counting interval, and the CNTIN value in the subsequent periods is the downward counting interval.

Two matching points can be combined into the following 4 situations:

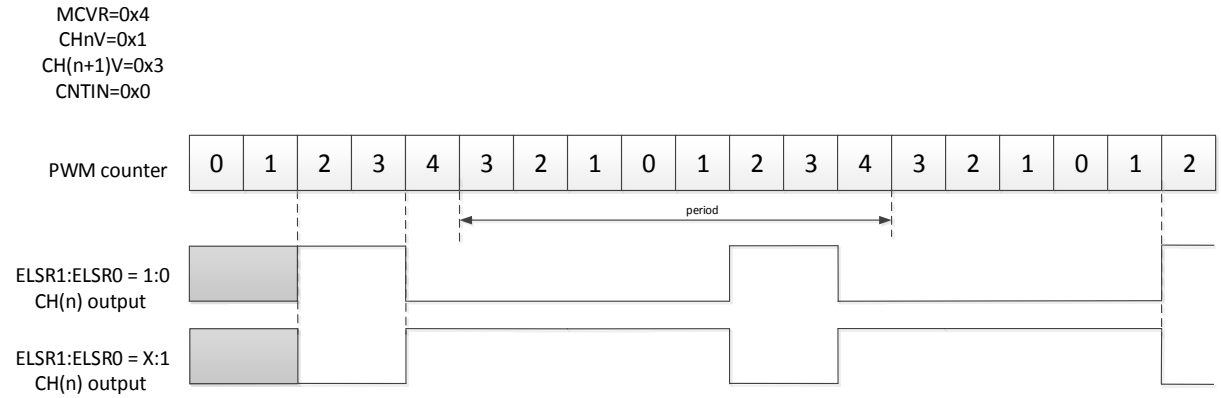

Figure 10-18 CHn matching point DIR=1(Up), CH(n+1) matching point DIR=1(Up)

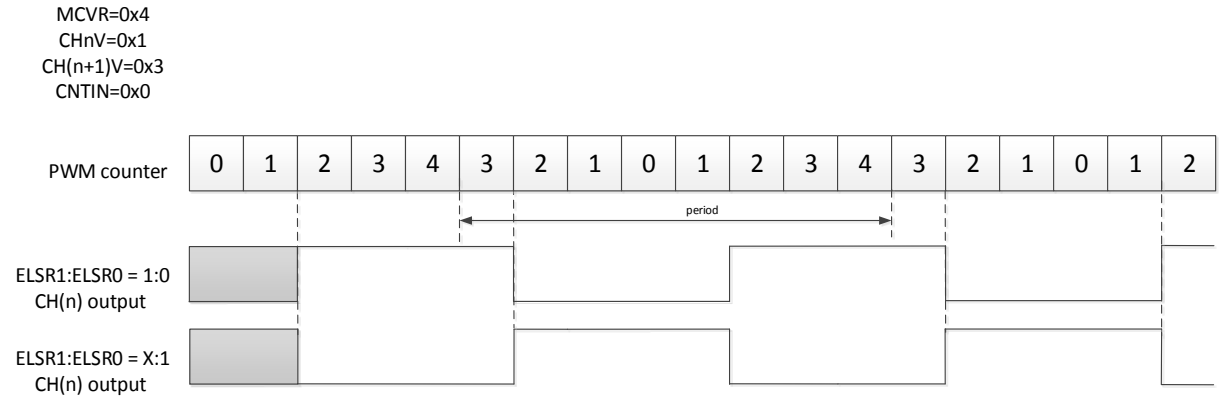

Figure 10-19 CHn matching point DIR=1(Up), CH(n+1) matching point DIR=0(Down)

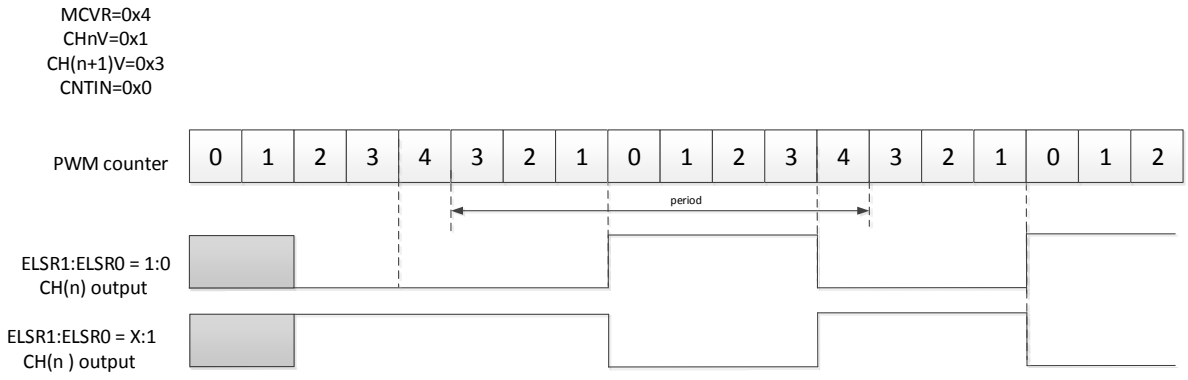

Figure 10-20 CHn matching point DIR=0(Down), CH(n+1) matching point DIR=1(Up)

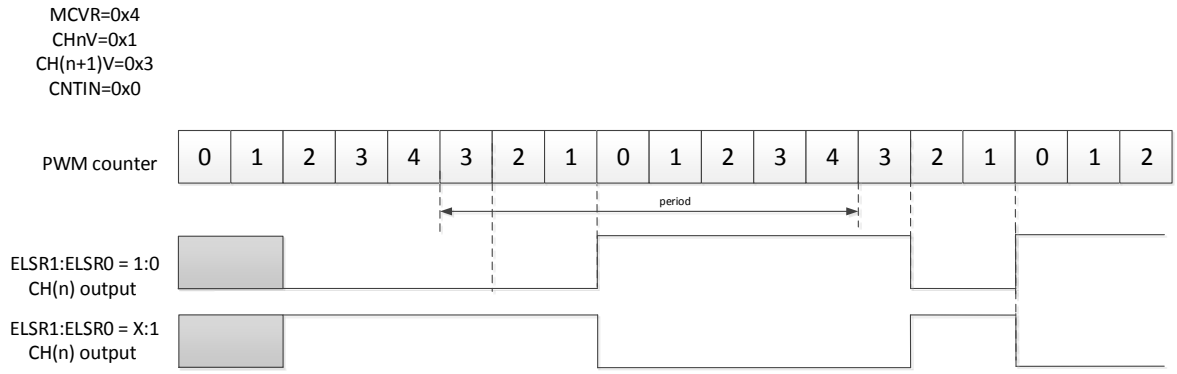

Figure 10-21 CHnV matching point DIR=0(Down), CH(n+1)V matching point DIR=0(Down)

10.4.8.3 Complementary mode

The Complementary mode is supported in the combine mode. By enabling PAIRnCOMPEN=1 in Complementary mode, the channel (n+1) output is the inverse of the channel (n) output.

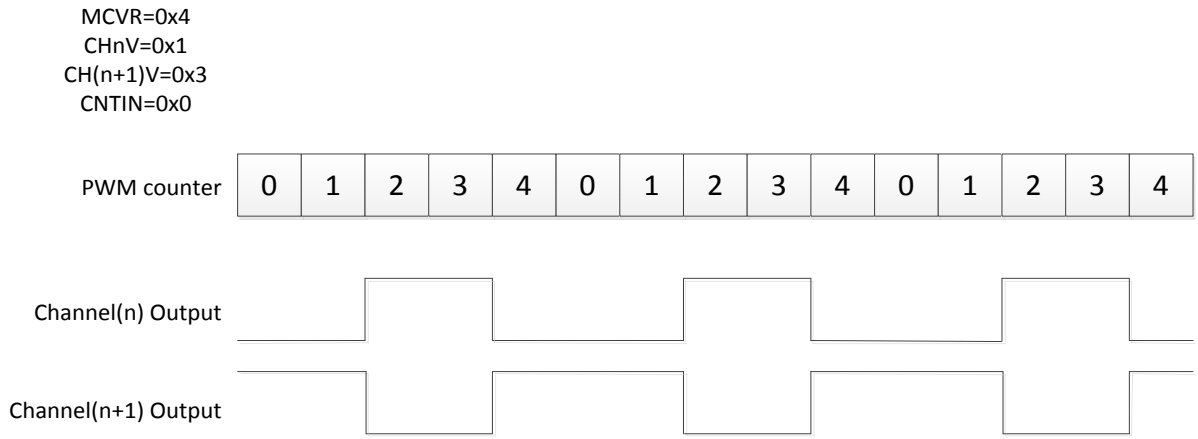

Figure 10-22 Output in complementary mode

10.4.8.4 Dead-time insertion

The dead-time insertion is enabled when (DTEN = 1) and (DTVAL[5:0] is non- zero). The PWM_DTSET register defines the dead-time delay that can be used for all PWM channels. The DTPSC[1:0] bits define the prescaler for the system clock and the DTVAL[5:0] bits define the dead-time modulo, that is, the number of the dead-time prescaler clocks. The dead-time delay insertion ensures that no two complementary signals (channels (n)and (n+1)) drive the active state at the same time.

If CH(n)POL = 0, CH(n+1)POL = 0, and the dead-time is enabled, then when the channel (n) match (PWM counter = CH(n)V) occurs, the channel (n) output remains at the low value until the end of the dead-time delay when the channel (n) output is set. Similarly, when the channel (n+1) matches

(PWM counter = CH(n+1)V) occurs, the channel (n+1) output remains at the low value until the end of the dead-time delay when the channel (n+1) output is set.

If CH(n)POL = 1, CH(n+1)POL= 1, and the dead-time is enabled, then when the channel (n)matches (PWM counter = CH(n)V) occurs, the channel (n) output remains at the high value until the end of the dead-time delay when the channel (n) output is cleared. Similarly, when the channel (n+1) matches (PWM counter = CH(n+1)V) occurs, the channel (n+1) output remains at the high value until the end of the dead-time delay when the channel (n+1) output is cleared.

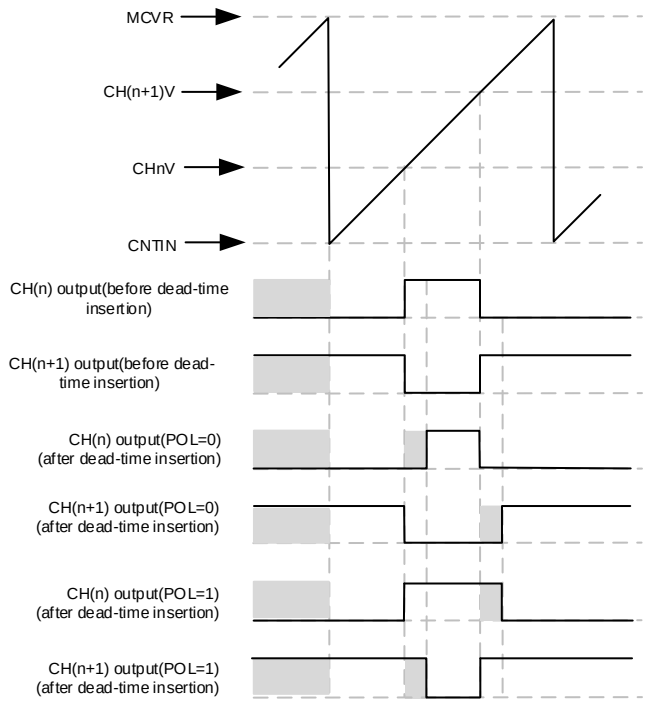

Figure 10-23 Dead-time insertion

10.4.8.5 Inverting

The Invert functionality swaps the signals between channel(n) and channel(n+1) outputs. The inverting operation is selected when PAIR(n)INVEN = 1, n=0,1,2,3.

10.4.8.6 Phase shift

In some cases, channel (n+1) matching will occur in the next PWM cycle. The phase shift occurs when (CNTIN < CHnV < MCVR) & (CNTIN < CH(n+1)V < MCVR) & (CHnV > CH(n+1)V).

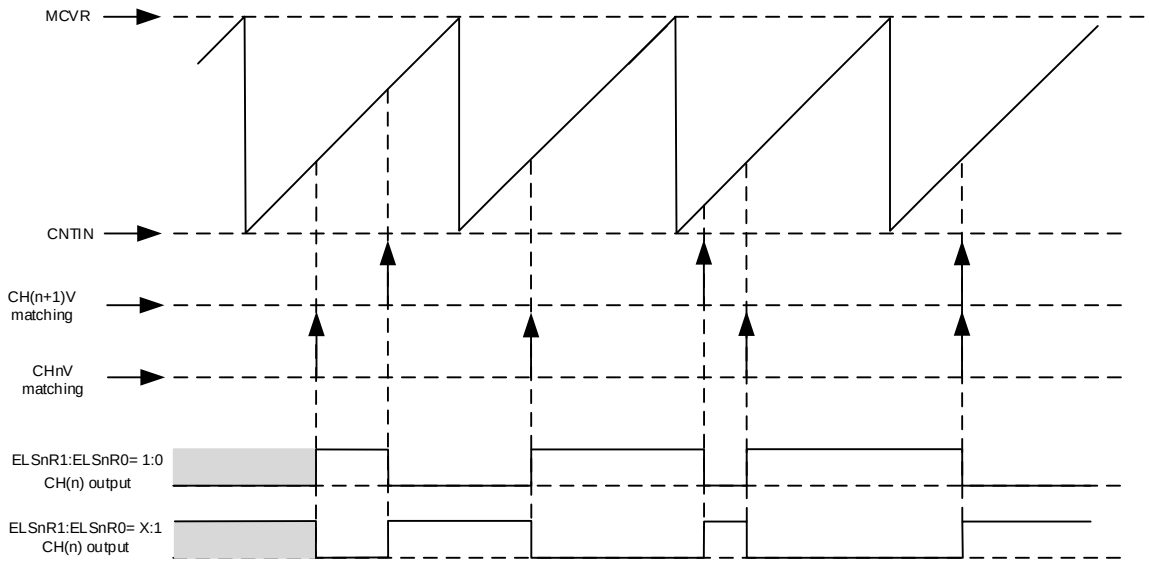

Figure 10-24 CH(n+1)V output in the next period matching

Based on the above features, when multiple channels of the PWM module output in combine mode, the phase shift between different channel pairs of the PWM module can be set. This behavior is useful in the generation of lighting PWM control signals where it is desirable that edges are not coincident with each other pair to help eliminate noise generation.

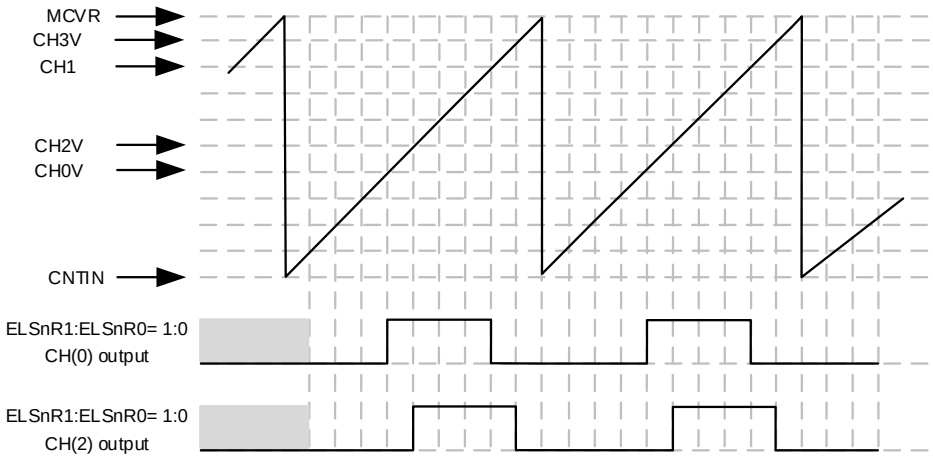

Figure 10-25 Multi-channel phase shift output waveform

10.4.9 Dual Edge Capture mode

The dual edge capture mode is selected if DECAPEN = 1. This mode allows to measure a pulse width or signal period on the input of channel (n). In this mode, only CH(n) input is used and CH(n+1) input is ignored. The CH(n) filter can be active in this mode when n is 0 or 2. The ELS(n)R1:ELS(n)R0 bits select the edge that is captured by CH(n), and ELS(n+1)R1:ELS(n+1)R0 bits select the edge that is captured by CH(n+1). If both ELS(n)R1:ELS(n)R0 and ELS(n+1)R1:ELS(n+1)R0 bits select the same edge, then it is the period measurement. If these bits select different edges, then it is a pulse width measurement.

If the selected edge by CH(n) bits is detected at CH(n) input, then CH(n)IF bit is set and the CH(n) interrupt is generated (if CH(n)IE = 1). If the selected edge by CH(n+1) bits is detected at CH(n) input and (CH(n)IF = 1), then CH(n+1)IF bit is set and the CH(n+1) interrupt is generated (if CH(n+1)IE = 1). The Dual edge capture does not support enabling CH(n) and CH(n+1) interrupts simultaneously.

The PWM_CH(n)V register stores the value of PWM counter when the selected edge by channel(n) is detected at channel (n) input. The PWM_CH(n+1)V register stores the value of PWM counter when the selected edge by channel (n+1) is detected at channel (n) input. In this mode, a coherency mechanism ensures coherent data when the PWM_CH(n)V and PWM_CH(n+1)V registers are read. The only requirement is that CH(n)V must be read before CH(n+1)V.

As shown in Figure 10-26, channel (n) selects rising edge capture and channel (n+1) selects falling edge capture for pulse width measurement.

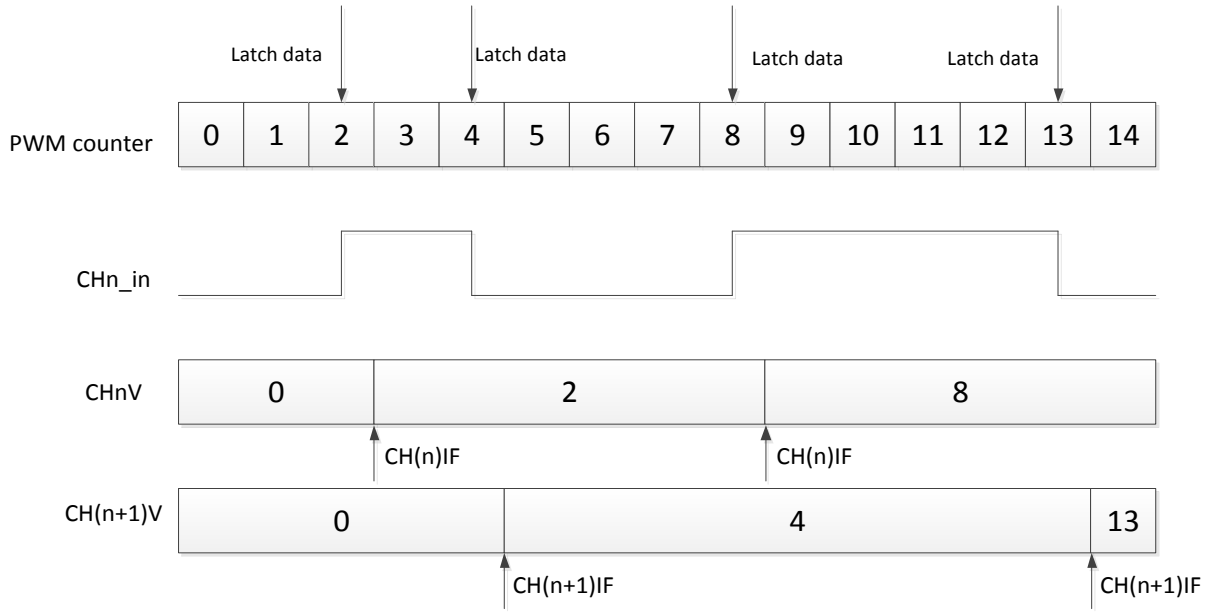

Figure 10-26 Dual edge capture mode

10.4.10 Quadrature decoder mode

The quadrature decoder mode is enabled if QDIEN = 1. The quadrature decoder mode uses the input signals phase A and B to control the PWM counter increment and decrement, the signal phase A inputs through the CH0 channel of the PWM module, and the signal phase B inputs through the CH1 channel of the PWM module. Each of input signals phase A and B has a filter that is equivalent to the channel input filter. The phase A input filter value is defined by CH0CAPFVAL[4:0] bits. The phase B input filter value is defined by CH1CAPFVAL[4:0] bits. The filter is disabled when its value is 0(the bit CH(n)CAPFVAL[4:0] is in the PWM_CAPFILTER register).

The PHAPOL bit selects the polarity of the phase A input, and the PHBPOL bit selects the polarity of the phase B input. The QUADMODE selects the encoding mode used in the quadrature decoder

mode. If QUADMODE= 1, then the count and direction encoding mode is enabled, see the following figure. In this mode, the phase B input value indicates the counting direction, and the phase A input defines the counting rate. The PWM counter is updated when there is a rising edge at phase A input signal.

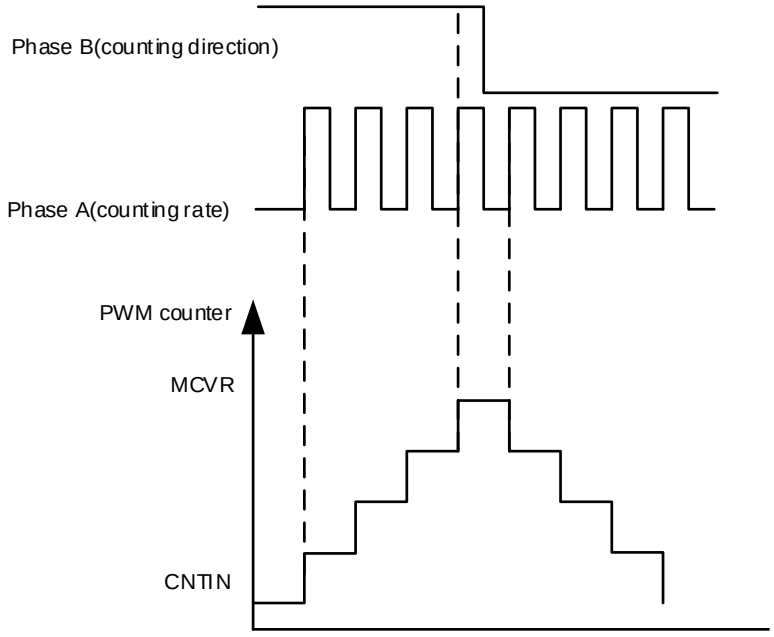

Figure 10-27 Quadrature Decoder—Count and Direction Encoding mode

If QUADMODE = 0, then the phase A and phase B encoding mode is enabled, see the following figure. In this mode, the relationship between phase A and B signals indicates the counting direction, and phase A and B signals define the counting rate. The PWM counter is updated when there is an edge either at the phase A or phase B signals.

If PHAPOL = 0 and PHBPOL = 0, then the PWM counter increment happens when:

- there is a rising edge at phase A signal and phase B signal is at logic zero;
- there is a rising edge at phase B signal and phase A signal is at logic one;
- there is a falling edge at phase B signal and phase A signal is at logic zero;
- there is a falling edge at phase A signal and phase B signal is at logic one.

and the PWM counter decrement happens when:

- there is a falling edge at phase A signal and phase B signal is at logic zero;
- there is a falling edge at phase B signal and phase A signal is at logic one;
- there is a rising edge at phase B signal and phase A signal is at logic zero;
- there is a rising edge at phase A signal and phase B signal is at logic one.

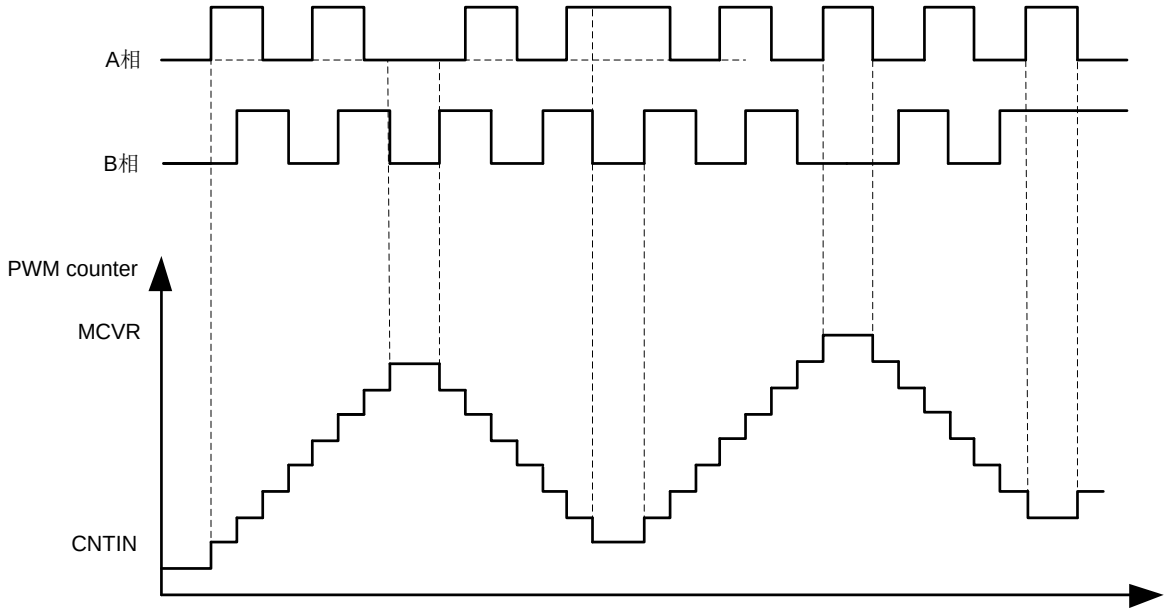

Figure 10-28 Quadrature Decoder—Phase A and Phase B Encoding Mode

The following figure shows the PWM counter overflow in up counting. In this case, when the PWM counter changes from MCVR to CNTIN, CNTOF and CNTOFDIR bits are set. CNTOF bit indicates the PWM counter overflow occurred. CNTOFDIR indicates the counting was up when the PWM counter overflow occurred.

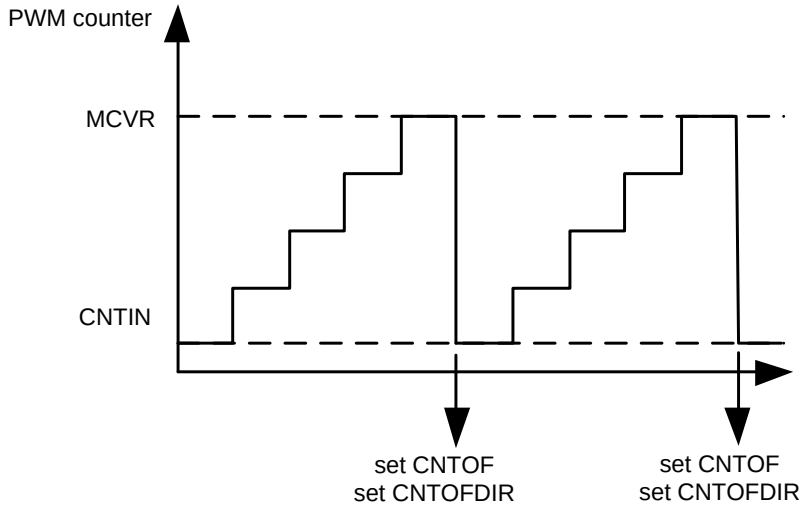

Figure 10-29 PWM Counter Overflow in Up Counting for Quadrature Decoder Mode

The following figure shows the PWM counter overflow in down counting. In this case, when the PWM counter changes from CNTIN to MCVR, CNTOF bit is set and CNTOFDIR bit is cleared. CNTOF bit indicates the PWM counter overflow occurred. CNTOFDIR indicates the counting was down when the PWM counter overflow occurred.

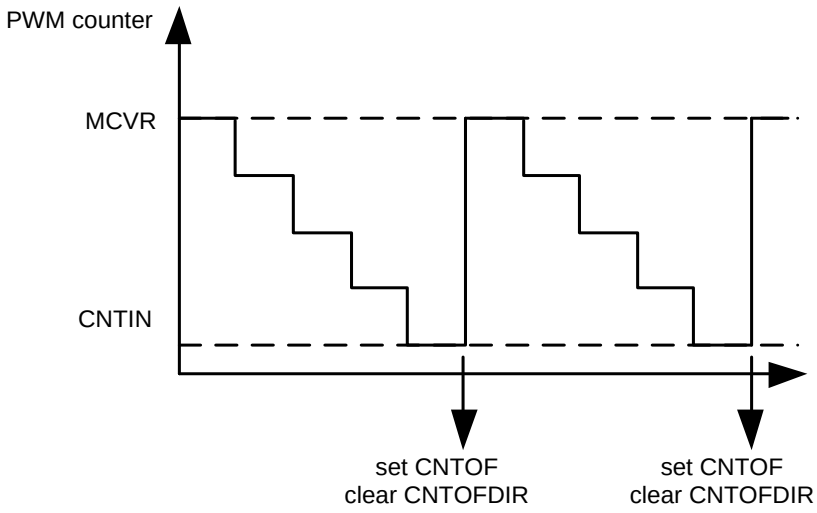

Figure 10-30 PWM Counter Overflow in Down Counting for Quadrature Decoder Mode

10.4.11 Write protection

In some application scenarios with high security level requirements (such as motor control), incorrectly modifying the register configuration can cause system abnormalities, which can easily lead to motor damage. The write protection function can provide write protection for certain key register bits. After completing the initial configuration, user can activate the write protection function by setting PWM_FDSR[WPEN] = 1, to avoid disoperation of key register bits caused by human or other abnormalities. Set PWM_FUNCSEL[WPDIS] = 1 to disable the write protection function and write again. The bits that support the write protection are described in chapter 10.5.

10.4.12 Initialization

The initialization forces the PWM_OUTINIT[CHnOIV] bit value to the channel (n) output when a one is written to the PWM_FUNCSEL[INIT] bit. The initialization feature can only be used when the PWM counter is disabled. When the PWM counter starts to work, the PWM_FUNCSEL[INIT] bit is automatically cleared, and the initialization output remains until the PWM channel starts to take over control.

10.4.13 Polarity control

The CHnPOL bit selects the channel(n) output polarity:

- If PWM_CHOPOLCR[CHnPOL] = 0, the channel(n) output polarity is high, so the logical one is the active state and logical zero is the inactive state.
- If PWM_CHOPOLCR[CHnPOL] = 1, the channel(n) output polarity is low, so the logical zero is the active state and logical one is the inactive state.

10.4.14 Output mask

The output mask can be used to force the channel output to its respective invalid state through software. If CHnOMEN = 1, the channel(n) output is forced to the inactive state of the channel (PWM_CHOPOLCR[CHnPOL] bit value).

10.4.15 Software output control

The software output control forces the channel output according to software defined values in the PWM generation. The CH(n)SWEN bit enables the software output control, and the CH(n)SWCV selects the value that is forced to this channel output. Independent mode, software output function can be set separately for each channel, software output function in combine mode is shown in the following table.

Table 10-2 Software output control behavior in combine mode

CH(n)SWEN	CH(n+1)SWEN	CH(n)SWCV	CH(n+1)SWCV	Channel(n) Output	Channel(n+1) Output
0	X(1)	X(1)	X(1)	Software control disable	Software control disable
1	X(1)	0	0	0	0
1	X(1)	0	1	0	1
1	X(1)	1	0	1	0
1	X(1)	1	1	1	0 or 1(2)

(1): X is either 0 or 1.

(2): If the PAIRnCOMPEN bit is 0, the output is 1. If the PAIRnCOMPEN bit is 1, the output is 0.

10.4.16 Initialization trigger

If INITTRIGEN = 1, PWM generates a trigger when the PWM counter is updated with the PWM_CNTIN register value in the following cases.

- The PWM counter is automatically updated with the PWM_CNTIN register value by the selected counting mode.
- When there is a write to PWM_CNT register.
- When there is the PWM counter synchronization.

10.4.17 Channel match trigger

If CHnTRIG = 1, PWM generates a trigger when the channel (n) match occurs (PWM counter = CH(n)V). The channel trigger output provides a trigger signal that is used for on-chip modules.

In the up-down combination mode, the channel value generates a match while up counting, and it also generates a match while down counting. In order to facilitate the control channel matching

trigger, the matching effective point PWM_CHnSCR[DIR] can be set to take effect while up counting or down counting.

10.4.18 Fault control

PWM provides 3 fault input sources: one is from chip inside, two is from external pin input.

FERnEN bit enables fault input n, FFnEN bit enables fault input n filter. The FFVAL bit selects the value of each fault input filter that has been enabled.

If the fault control and fault input n are enabled and effective edge at the fault input n signal is detected, a fault condition has occurred and the FAULTDFn bit is set. The FAULTDF bit is the logical OR of the FAULTDFn[2:0] bit.

If the fault control is enabled (FAULTMODE[1:0] ≠ 0:0), a fault condition has occurred and (FAULTEN = 1), then outputs are forced to their safe values:

- Channel (n) output takes the value of CHnPOL
- Channel (n+1) takes the value of CH(n+1)POL

The fault interrupt is generated when (FAULTDF = 1) and (FAULTIE = 1).

Table 10-3 Fault Source and Number Table

Fault Input Number	Fault Source	Description
FAULT0	ACMP0_OUT	Internal Fault Input
FAULT1	PWM_FLT0	External Pin Fault Input
FAULT2	PWM_FLT1	External Pin Fault Input

10.4.18.1 Automatic fault clearing

If the automatic fault clearing is selected (FAULTMODE[1:0] = 1:1), then the channels output disabled by fault control is again enabled when the fault input signal returns to zero and a new PWM cycle begins.

10.4.18.2 Manual fault clearing

If the manual fault clearing is selected (FAULTMODE[1:0]=0:1 or 1:0), then the channels output disabled by fault control is again enabled when the FAULTDF bit is cleared and a new PWM cycle begins.

10.4.18.3 Fault inputs polarity control

The FLTnPOL bit selects the fault input n polarity:

- If FLTnPOL = 0, the fault n input polarity is high, so the logical one at the fault input n indicates a fault.

- If FLTnPOL = 1, the fault n input polarity is low, so the logical zero at the fault input n indicates a fault.

10.4.19 Registers updated from write buffers

10.4.19.1 PWM_CNTIN register update buffer

Table 10-4 PWM_CNTIN register update buffer

Condition	Register update opportunity
CLKSRC = 0	When PWM_CNTIN register is written
CLKSRC≠ 0 & PWMSYNCEN = 0	At the next system clock after PWM_CNTIN was written
CLKSRC≠ 0 & PWMSYNCEN = 1	Refer to Chapter 10.4.20.6 PWM_CNTIN register synchronization

10.4.19.2 PWM_CH(n)V register update buffer

Table 10-5 PWM_CH(n)V register update buffer

Condition	Register update opportunity
CLKSRC=0	When PWM_CH(n)V register is written
CLKSRC≠ 0 & PWMSYNCEN = 0	• If the selected mode is EPWM, then register is updated after the PWM counter changes from MCVR to CNTIN. • If the selected mode is CPWM, then register is updated after the PWM counter changes from MCVR to (MCVR – 0x0001).
CLKSRC ≠ 0 & PWMSYNCEN = 1	Refer to Chapter 10.4.20.7 PWM_CH(n)V and PWM_CH(n+1)V register synchronization

10.4.19.3 PWM_MCVR register update buffer

Table 10-6 PWM_MCVR register update buffer

Condition	Register update opportunity
CLKSRC=0	When PWM_MCVR register is written
CLKSRC ≠ 0 & PWMSYNCEN = 0	• If the selected mode is EPWM, then register is updated after the PWM counter changes from MCVR to CNTIN. • If the selected mode is CPWM, then register is updated after the PWM counter changes from MCVR to (MCVR – 0x0001).
CLKSRC0 ≠ 0 & PWMSYNCEN = 1	Refer to Chapter 10.4.20.4 PWM_MCVR register synchronization

10.4.20 PWM synchronization

The PWM synchronization provides a mechanism to update the PWM_MCVR, PWM_CNTIN, PWM_CHnV, PWM_OMCR, PWM_INVCR and PWM_CHOSWCR registers with their buffered value and set the PWM counter to the PWM_CNTIN register value.

10.4.20.1 Hardware trigger

Three hardware trigger signal inputs of the PWM module are enabled when TRIGn = 1, where n = 0, 1 or 2 corresponding to each one of the input signals, respectively. The hardware trigger input n is synchronized by the system clock.

- Hardware trigger 0: the ACMP_OUT signal is used as PWM synchronous trigger source 0;
- Hardware trigger 1: the PWMSYNC control bit of the CTU module is used as the PWM synchronous hardware trigger source 1;
- Hardware trigger 2: PWM channel 0 output as a synchronous hardware trigger source 2(the trigger source of the PWM0 & PWM2 module is the PWM1_CH0_OUT signal, the trigger source of the PWM1 module is the PWM0_CH0_OUT signal).

The PWM synchronization with hardware trigger is initiated when a rising edge is detected at the enabled hardware trigger inputs if (HWTRIGMODESEL = 0) , then the TRIGn bit is cleared when 0 is written to it or when the trigger n event is detected.

In this case, if two or more hardware triggers are enabled (for example, TRIG0 and TRIG1 = 1) and only trigger 1 event occurs, then only TRIG1 bit is cleared. If a trigger event occurs together with a write setting TRIGn bit, then the synchronization is initiated, but TRIGn bit remains set due to the write operation.

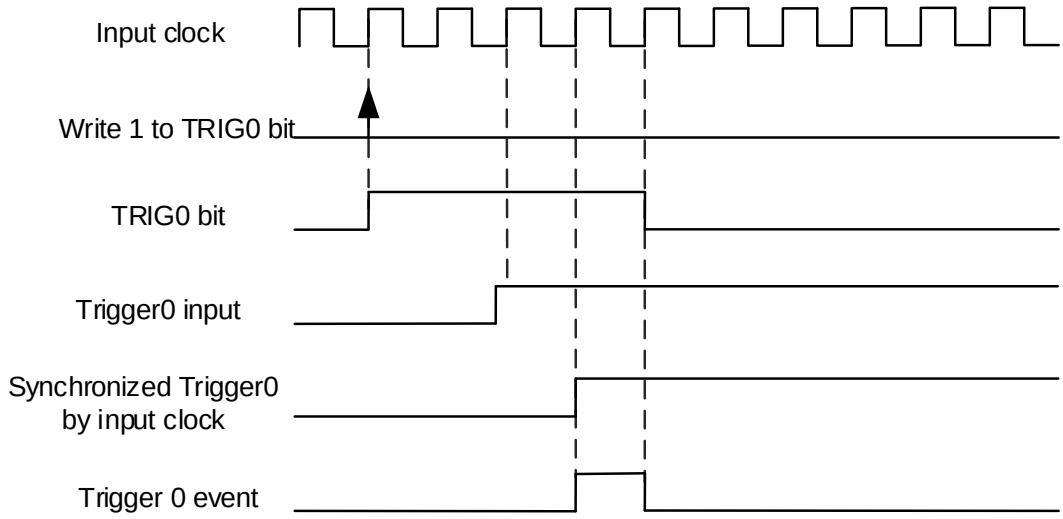

Figure 10-31 Hardware trigger event with HWTRIGMODESEL = 0

10.4.20.2 Software trigger

A software trigger event occurs when 1 is written to the PWM_SYNC[SWSYNC] bit. The SWSYNC bit is cleared when 0 is written to it or when the PWM synchronization, initiated by the software event, is completed.

If another software trigger event occurs (by writing another 1 to the SWSYNC bit) at the same time, the PWM synchronization initiated by the previous software trigger event is ending, a new PWM synchronization is started and the SWSYNC bit remains equal to 1.

If SYNCMODE = 1, then the SWSYNC bit is also cleared by PWM according to the CNTVSWSYNC bit. If CNTVSWSYNC=0, then the SWSYNC bit is cleared at the next selected loading point after that the software trigger event occurred. If CNTVSWSYNC = 1, then SWSYNC bit is cleared when the software trigger event occurs.

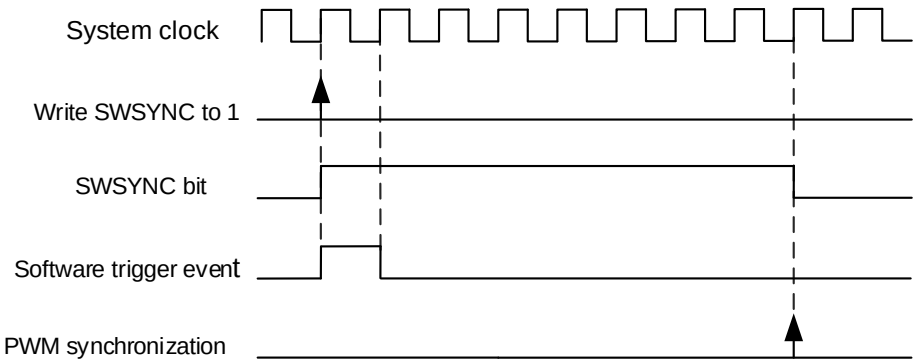

Figure 10-32 Software trigger event

10.4.20.3 Boundary period and synchronization points

In Up counting mode, the boundary period is defined as when the counter becomes its initial value (CNTIN). In Up-down counting, the boundary period is defined as when the counter changes from down counting to up counting and from up counting to down counting. The following figure shows the boundary period and synchronization points of the register.

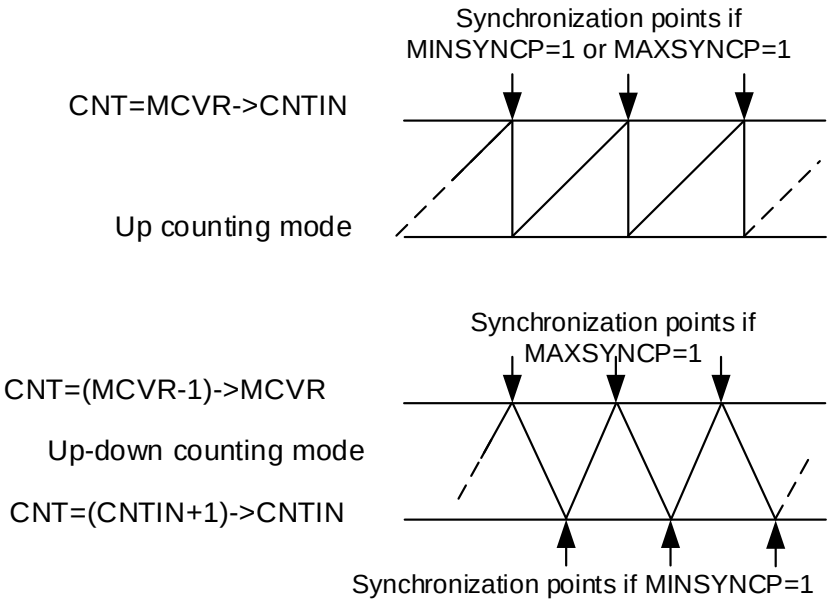

Figure 10-33 Synchronization points

In up counting mode, the MINSYNCP or MAXSYNCP is 1, then enable the synchronization point. In up-down counting mode, it is the MINSYNCP & MAXSYNCP to select the synchronization point. In the two counting modes, if both MINSYNCP and MAXSYNCP are not 1, the boundary period is not used as a synchronization point for register updates, even if the trigger signal is received, a synchronization event will not be generated (CNTVSWSYNC=0). For details, see the description of register synchronization in the following sections.

10.4.20.4 PWM_MCVR register synchronization

The PWM_MCVR register synchronization updates the PWM_MCVR register with its buffer value. This synchronization is enabled if (PWMSYNCEN = 1).

The flow chart of the PWM_MCVR register synchronization is shown below.

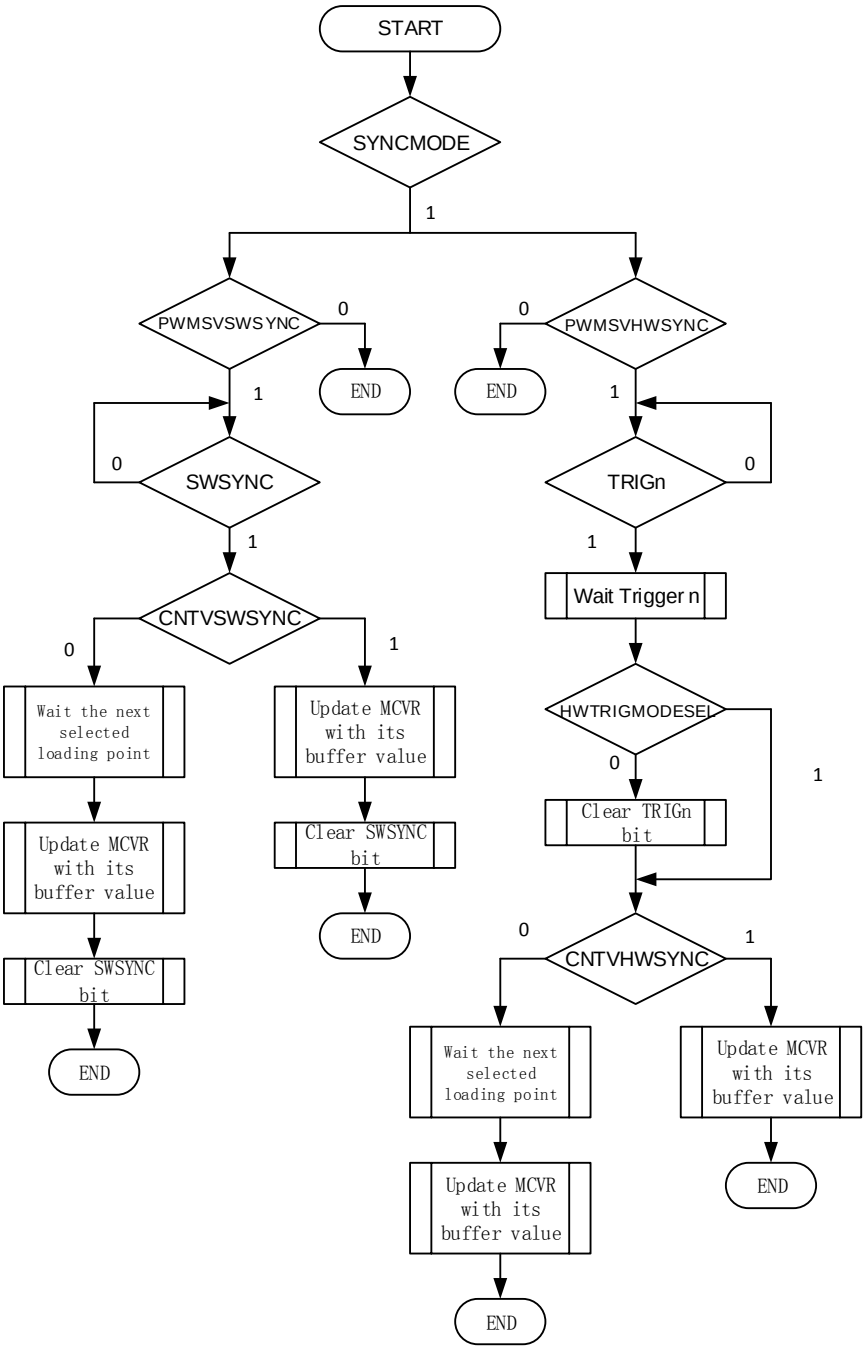

Figure 10-34 PWM_MCVR register synchronization flowchart

10.4.20.5 PWM_CNT register synchronization

The PWM_CNT register synchronization function can restart to generate PWM at a specific point in the PWM cycle. The channel output is forced to its initial value, and the PWM_CNT register is forced to the initial count value defined by the PWM_CNTIN register. The flow chart of the PWM_CNT register synchronization is given below.

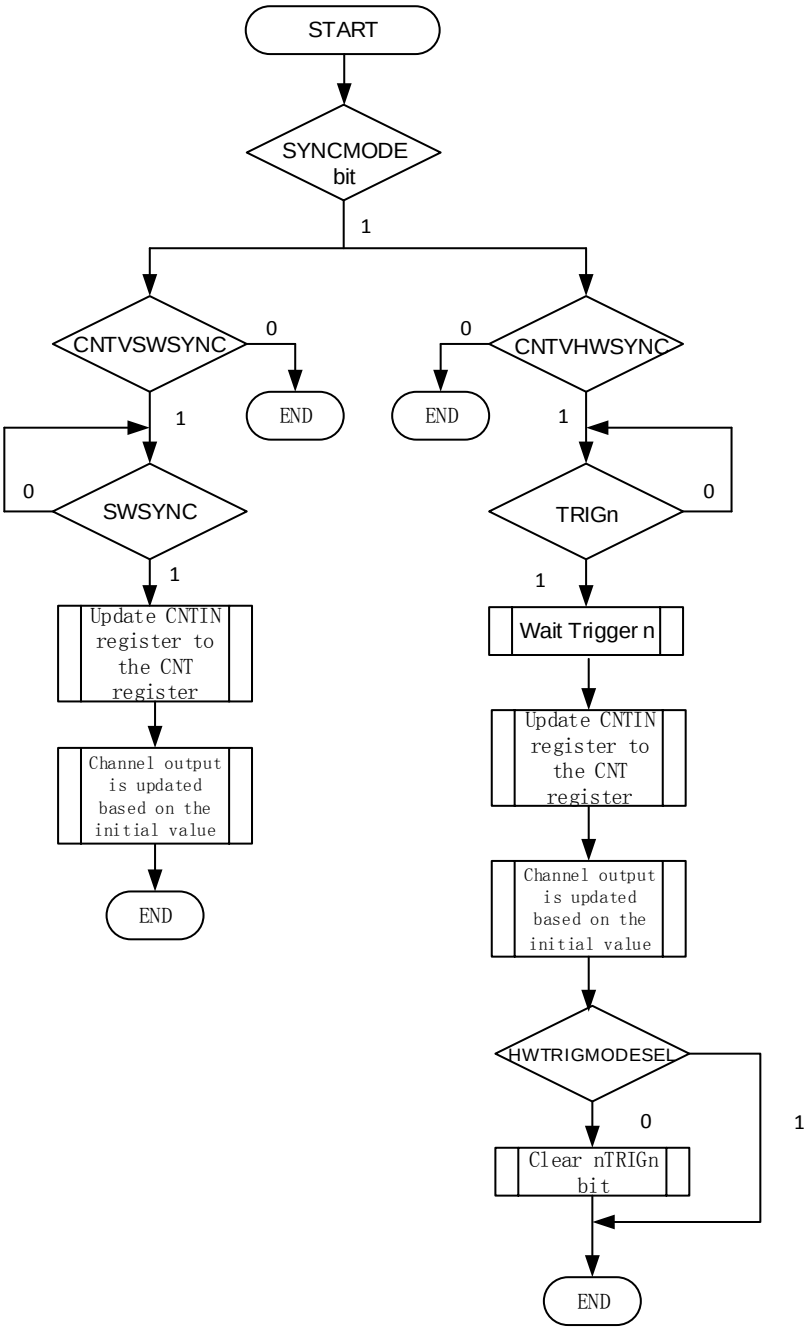

Figure 10-35 PWM_CNT register synchronization flow

10.4.20.6 PWM_CNTIN register synchronization

This synchronization is enabled if (PWMSYNCEN=1) , (SYNCMODE = 1) and (CNTINC = 1). The synchronization mechanism is the same as the PWM_MCVR register synchronization.

10.4.20.7 PWM_CH(n)V and PWM_CH(n+1)V register synchronization

This synchronization is enabled if (PWMSYNCEN=1) and (PAIR(n)SYNCEN=1). The synchronization mechanism is the same as the PWM_MCVR register synchronization.

10.4.20.8 PWM_OMCR register synchronization

The PWM_OMCR register synchronization updates the PWM_OMCR register with its buffer value.

For the flow chart, refer to Figure 10-36.

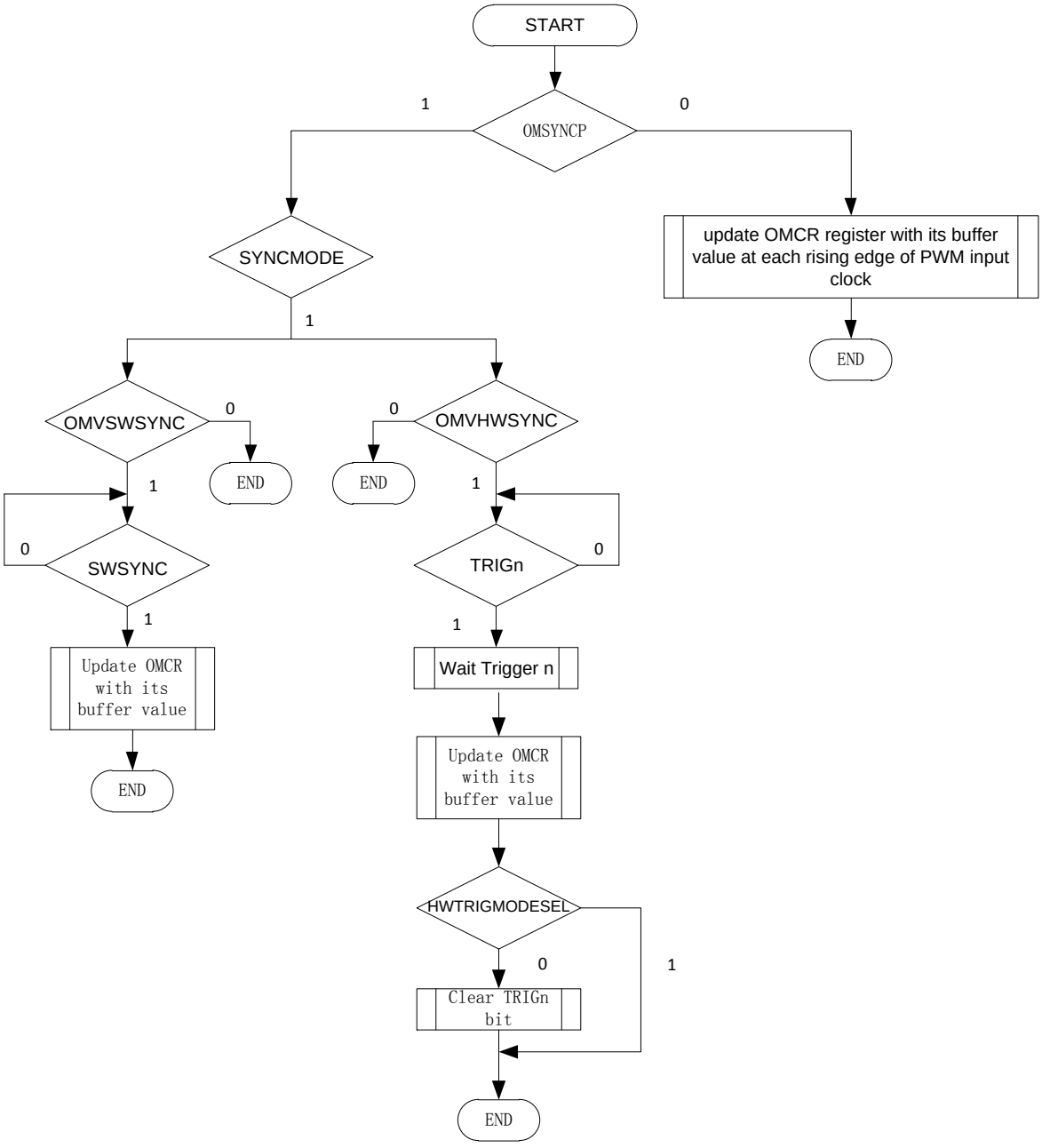

Figure 10-36 PWM_OMCR register synchronization flow chart

10.4.20.9 PWM_INVCR register synchronization

The PWM_INVCR register synchronization updates the PWM_INVCR register with its buffer value. For the flow chart, refer to Figure 10-37.

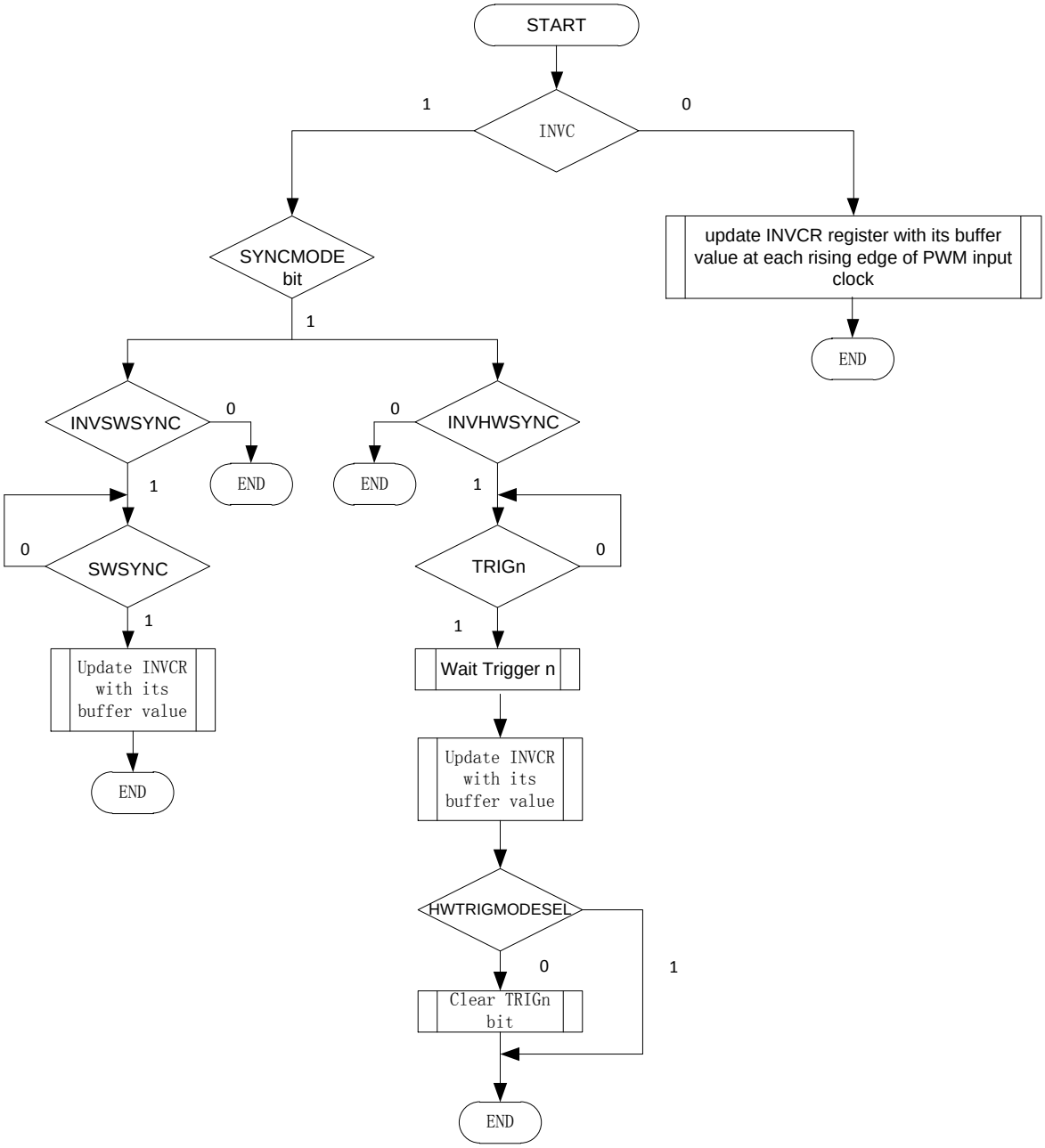

Figure 10-37 PWM_INVCR register synchronization flowchart

10.4.20.10 PWM_CHOSWCR register synchronization

The PWM_CHOSWCR register synchronization updates the PWM_CHOSWCR register with its buffer value. For the flow chart, refer to Figure 10-38.

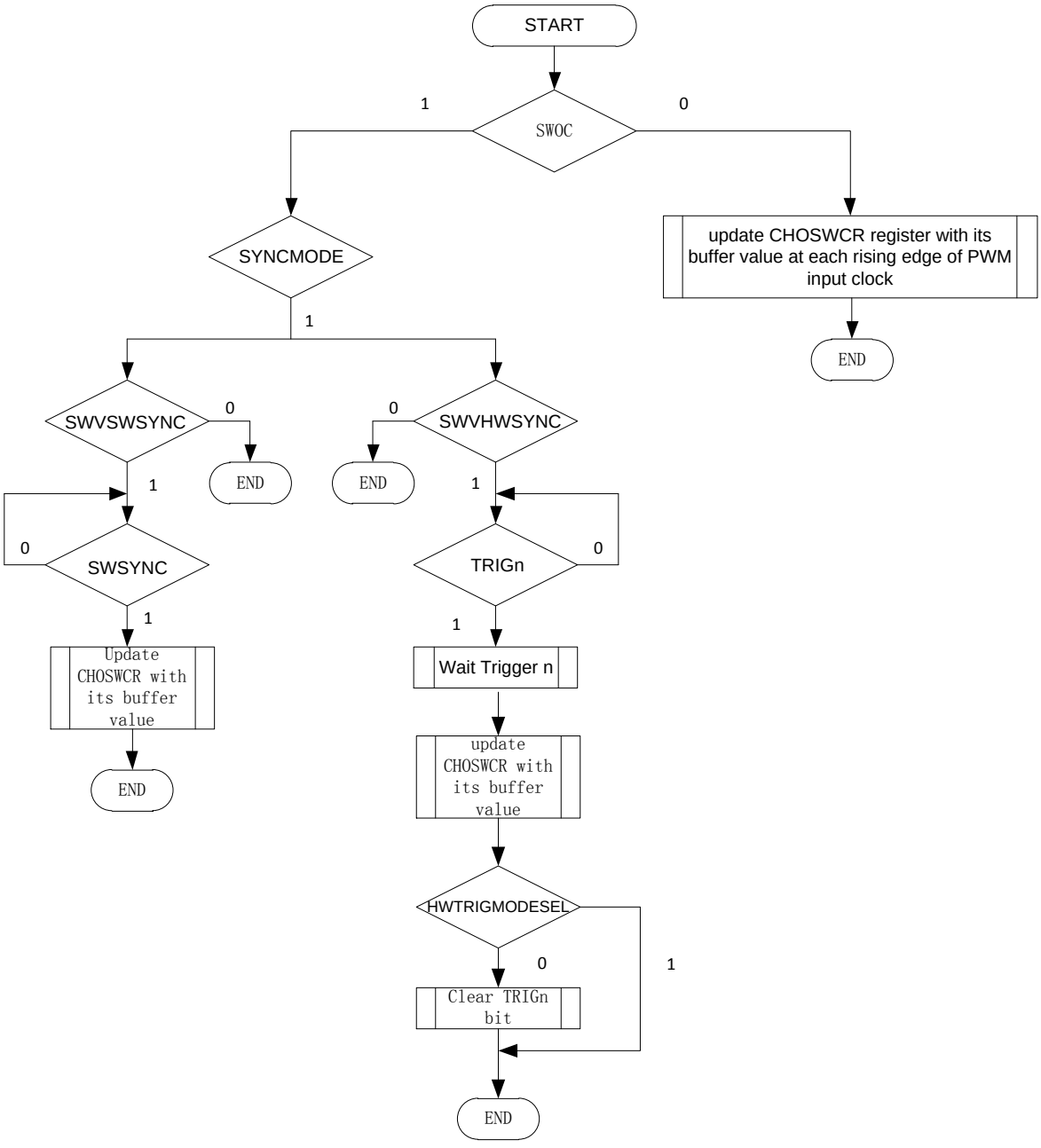

Figure 10-38 PWM_CHOSWCR register synchronization flowchart

10.4.20.11 PWM_CHOPOLCR register synchronization

The PWM_CHOPOLCR register synchronization updates the PWM_CHOPOLCR register with its buffer value. For the flow chart, refer to Figure 10-39.

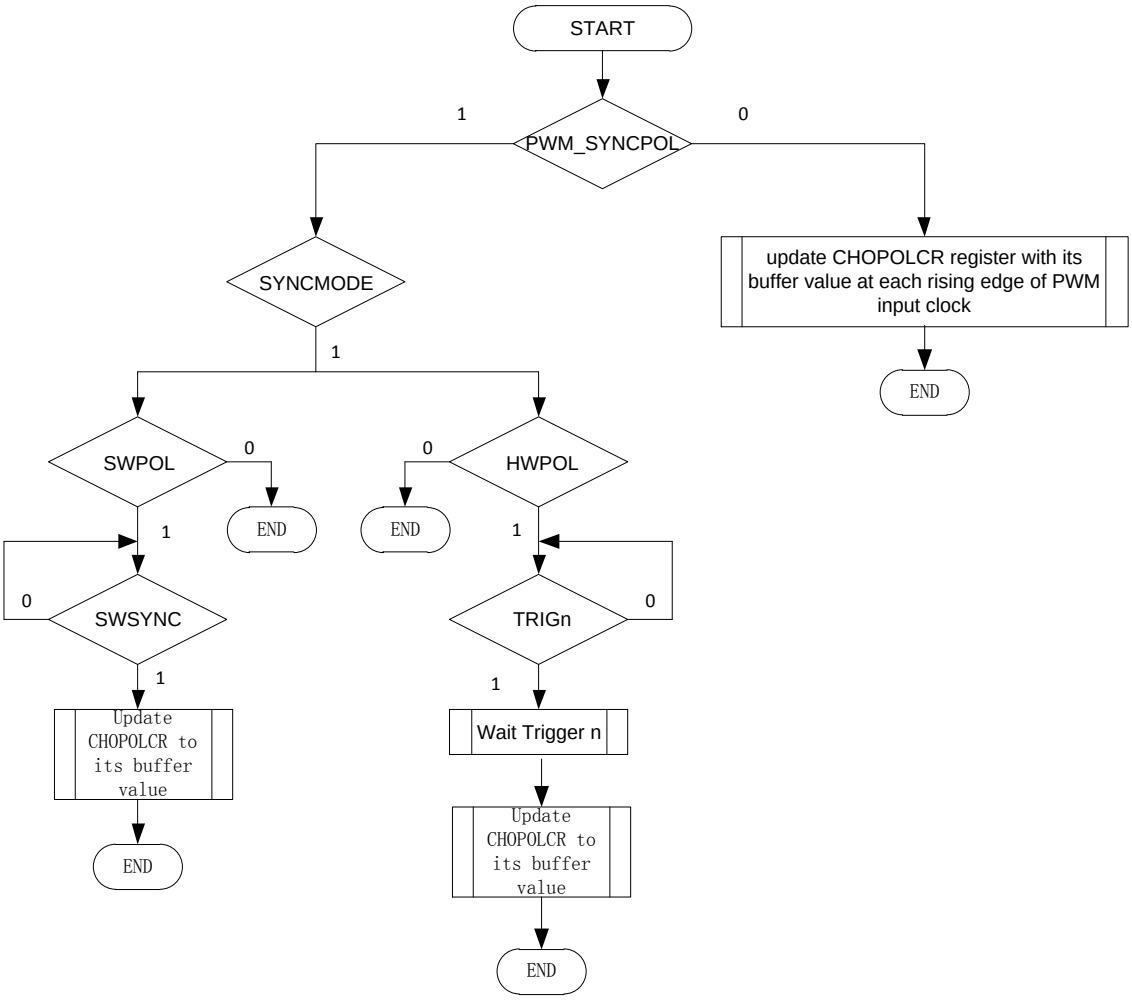

Figure 10-39 PWM_CHOPOLCR register synchronization flowchart

10.4.21 Features priority

The following figure shows the priority of the features used at the generation of CH(n) and CH(n+1) outputs signals.

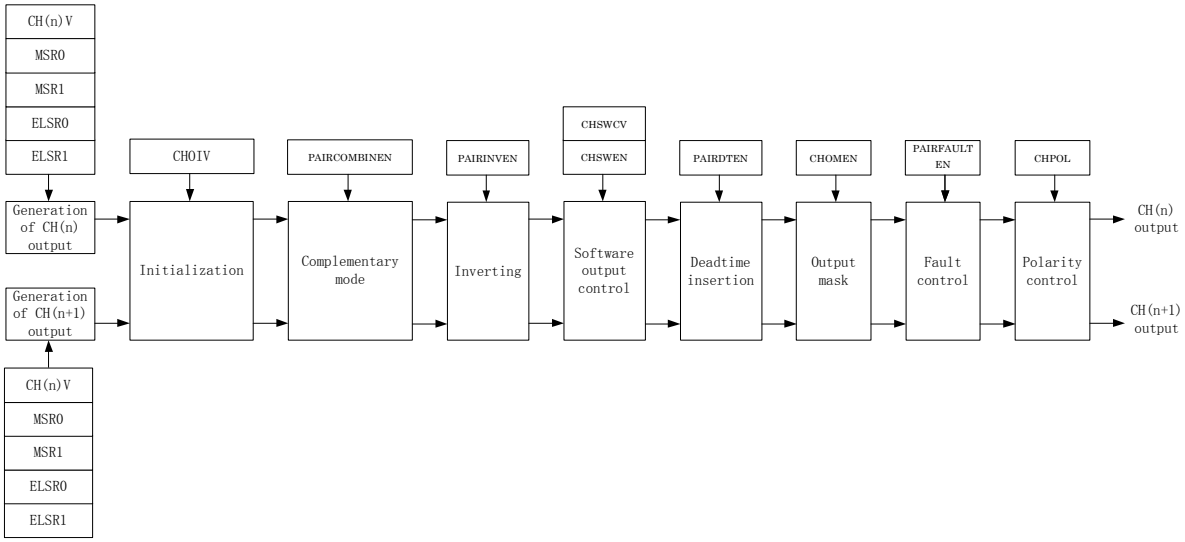

Figure 10-40 Features priority

10.4.22 Global time base (GTB)

The global time base (GTB) is a PWM function that allows the synchronization of multiple PWM modules on a chip.

The GTB functionality is implemented by the GTBEEN and GTBEOUT bits in the PWM_CONF register.

To enable the GTB feature, follow these steps for each participating PWM module:

1. Reset the PWM module;
2. Configure the PWM counter, but the PWM clock is not enabled, that is, write 00b to PWM_INIT[CLKSRC];
3. Write 1 to PWM_CONF [GTBEEN] and write 0 to PWM_CONF[GTBEOUT] at the same time;
4. Select the intended PWM counter clock source in PWM_INIT[CLKSRC];
5. Reset the PWM counter by writing any value to the PWM_CNT register;
6. Write 1 to PWM_CONF [GTBEOUT] in the PWM module used as the time base, starting multiple PWM modules to work at the same time.

10.4.23 PWM interrupts

- The Count Overflow interrupt is generated when (CNTOIE=1) and (CNTOF=1).
- The Channel(n) interrupt is generated when (CHnIE=1) and (CHnIF = 1).
- The Fault interrupt is generated when (FAULTIE =1) and (FAULTDF = 1).

10.4.24 Low power mode

Table 10-7 PWM low power mode

Mode	Wakeup source	Description
Sleep mode	interrupt	The module works normally, waking MCU by interrupts
Stop mode	----	The module is disabled

10.5 Register definition

Table 10-8 PWM register map

PWM0 base address = 0x40013000
PWM1 base address = 0x40014000
PWM2 base address = 0x40015000

Address	Register name	Width	Description
PWMx base address +0x00	PWM_INIT	32	Initialization Register
PWMx base address +0x04	PWM_CNT	32	Counter Value Register
PWMx base address +0x08	PWM_MCVR	32	Maximum Count Value Register
PWMx base address +0x0C	PWM_CH0SCR	32	Channel (0) Status And Control Register
PWMx base address +0x10	PWM_CH0V	32	Channel (0) Value
PWMx base address +0x14	PWM_CH1SCR	32	Channel (1) Status And Control Register
PWMx base address +0x18	PWM_CH1V	32	Channel (1) Value
PWMx base address +0x1C	PWM_CH2SCR	32	Channel (2) Status And Control Register
PWMx base address +0x20	PWM_CH2V	32	Channel (2) Value
PWMx base address +0x24	PWM_CH3SCR	32	Channel (3) Status And Control Register
PWMx base address +0x28	PWM_CH3V	32	Channel (3) Value
PWMx base address +0x4C	PWM_CNTIN	32	Counter Initial Value Register
PWMx base address +0x50	PWM_STR	32	Capture and Compare Status Register
PWMx base address +0x54	PWM_FUNCSEL	32	Function Mode Selection Register
PWMx base address +0x58	PWM_SYNC	32	Synchronization Register
PWMx base address +0x5C	PWM_OUTINIT	32	Initial State For Channels Output
PWMx base address +0x60	PWM_OMCR	32	Output Mask Control Register
PWMx base address +0x64	PWM_MODESEL	32	Mode Selection Register
PWMx base address +0x68	PWM_DTSET	32	Dead-time Setting Register
PWMx base address +0x6C	PWM_EXTTRIG	32	External Trigger Register
PWMx base address +0x70	PWM_CHOPOLCR	32	Channel Output Polarity Control Register
PWMx base address +0x74	PWM_FDSR	32	Fault Detect Status Register
PWMx base address +0x78	PWM_CAPFILTER	32	Input Capture Mode Filter Control
PWMx base address +0x7C	PWM_FFAFER	32	Fault Filter And Fault Enable Register
PWMx base address +0x80	PWM_QDI	32	Quadrature Decoder Control And Status
PWMx base address +0x84	PWM_CONF	32	Configuration register
PWMx base address +0x88	PWM_FLTPOL	32	Fault Input Polarity Register
PWMx base address +0x8C	PWM_SYNCONF	32	Synchronization Configuration Register
PWMx base address +0x90	PWM_INVCR	32	Inverting Control Register
PWMx base address +0x94	PWM_CHOSWCR	32	Channel software output control register

Note: x=0,1,2 in the above table.

10.5.1 Initialization Register(PWM_INIT)

Table 10-9 PWM_INIT register

PWM_INIT **PWM Initialization Register** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name									CLKPSC[15：8]							
Type									RW							
Reset									0							
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	CLKPSC[7：0]								CNTOF	CNTOIE	CNTMODE		CLKSRC			
Type	RW								R/W0C	RW	RW		RW			
Reset	0								0				0			

Field	Description
23: 8 CLKPSC	**PWM CLK prescaler** **CLK prescaler = CLKPSC + 1** The new prescaler factor affects the clock source on the next system clock cycle after the new value is updated into the register bits. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
7 CNTOF	**Timer Overflow Flag** 0: PWM counter has not overflowed. 1: PWM counter has overflowed. Set by hardware when the PWM counter passes the value in the MCVR register. The CNTOF bit is cleared by reading the INIT register while CNTOF is set and then writing a 0 to CNTOF bit. If another PWM overflow occurs between the read and write operations, the write operation has no effect. Therefore, CNTOF remains set indicating an overflow has occurred.
6 CNTOIE	**Timer Overflow Interrupt Enable** 0: Disable CNTOF interrupts. Use software polling. 1: Enable CNTOF interrupts. An interrupt is generated when CNTOF is set. Enables PWM overflow interrupts.
5 CNTMODE	**Counter Mode Select** 0: PWM counter operates in Up Counting mode. 1: PWM counter operates in Up-Down Counting mode. Select counter mode. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.

Field	Description
3 CLKSRC	**Clock Source Selection** 0: No clock selected. This in effect disables the PWM counter. 1: Bus clock Select PWM counter clock sources. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.

10.5.2 Counter Value Register(PWM_CNT)

Table 10-10 PWM_CNT register

PWM_CNT **PWM counter value** **Reset:00000000**

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

Field	Description
15: 0 COUNT	**PWM counter value.** The CNT register contains the PWM counter value. Reset clears the CNT register. Writing any value to COUNT updates the counter with its initial value, CNTIN.

10.5.3 Maximum Count Value Register(PWM_MCVR)

Table 10-11 PWM_MCVR register

PWM_MCVR **Max Count Value Register** **Reset:00000000**

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

Field	Description
15: 0	**Max Count Value Register**

Field	Description
MCVR	The MCVR register contains the modulo value for the PWM counter. After the PWM counter reaches the MCVR value, the overflow flag (CNTOF) becomes set at the next clock, and the next value of PWM counter depends on the selected counting method. Writing to the MCVR register latches the value into a buffer. The MCVR register is updated with the value of its write buffer according to registers updated from write buffers. Initialize the PWM counter, by writing to CNT before writing to the MCVR register to avoid confusion about when the first counter overflow will occur.

10.5.4 Channel Status and Control Register(PWM_CHnSCR)

Table 10-12 PWM_CHnSCR register

PWM_CHnSCR **Channel (n) Status And Control Register** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									CHIF	CHIE	MSR1	MSR0	ELSR1	ELSR0	DIR	
Type									R/W0C	RW	RW	RW	RW	RW	RW	
Reset									0	0	0	0	0	0	0	

Field	Description
7 CHIF	**Channel Interrupt Flag** 0: No channel event has occurred. 1: A channel event has occurred. Set by hardware when an event occurs on the channel. By reading the CHSCR register, the CHIF bit is cleared by writing a 0 to this bit. Writing a 1 to CHIF has no effect. If another event occurs between the read and write operations, the write operation has no effect. Therefore, CHIF remains set indicating an event has occurred. In this case, a CHIF interrupt request is not lost due to the write clear operation.
6 CHIE	**Channel Interrupt Enable** 0: Disable channel interrupts. 1: Enable channel interrupts. Enables channel interrupts.
5 MSR1	**Channel Mode Select Register 1** Used for further selections in the channel logic. Its functionality is dependent on the channel mode. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
4 MSR0	**Channel Mode Select Register 0**

Field	Description
	Used for further selections in the channel logic. Its functionality is dependent on the channel mode. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
3 ELSR1	**Edge or Level Select Register 1** The functionality of ELSR1 and ELSR0 depends on the channel mode. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
2 ELSR0	**Edge or Level Select Register 0** The functionality of ELSR1 and ELSR0 depends on the channel mode. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
1 DIR	**Match point direction** 0: Match points take effect during down counting. 1: Match points take effect during up counting. Only used in up-down counting combination mode, and used to select the direction of matching effective point. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.

10.5.5 Channel Value(PWM_CHnV)

Table 10-13 PWM_CHnV register

PWM_CHnV **Channel (n) value** **Reset:00000000**

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

Field	Description
15:0 CHCVAL	**Channel Count Value** These registers contain the captured PWM counter value for the input modes or the match value for the output modes. In Input Capture and Dual Edge Capture modes, any write to a CHnV register is ignored. In output modes, writing to a CHnV register latches the value into a buffer. A CHnV register is updated with the value of its write buffer according to registers updated from write buffers.

10.5.6 Counter Initial Value Register(PWM_CNTIN)

Table 10-14 PWM_CNTIN register

PWM_CNTIN **Counter Initial Value** **Reset:00000000**

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

Field	Description
15: 0 CNTINIT	**Counter Initial Value** The Counter Initial Value register contains the initial value for the PWM counter. Writing to the CNTIN register latches the value into a buffer. The CNTIN register is updated with the value of its write buffer according to registers updated from write buffers. When the PWM clock is initially selected, by writing a non-zero value to the CLKSRC bits, the PWM counter starts with the value 0x0000. To avoid this behavior, before the first write to select the PWM clock, write the new value to the CNTIN register and then initialize the PWM counter by writing any value to the CNT register.

10.5.7 Capture and Compare Status Register(PWM_STR)

Table 10-15 PWM_STR register

PWM_STR **Capture And Compare Status Register** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name													CH3SF	CH2SF	CH1SF	CH0SF
Type													R/W0C	R/W0C	R/W0C	R/W0C
Reset													0	0	0	0

Field	Description
3 CH3SF	**Channel 3 Status Flag** 0: No channel event has occurred. 1: A channel event has occurred.

Field	Description
2 CH2SF	**Channel 2 Status Flag** 0: No channel event has occurred. 1: A channel event has occurred.
1 CH1SF	**Channel 1 Status Flag** 0: No channel event has occurred. 1: A channel event has occurred.
0 CH0SF	**Channel 0 Status Flag** 0: No channel event has occurred. 1: A channel event has occurred.

10.5.8 Function Mode Selection Register(PWM_FUNCSEL)

Table 10-16 PWM_FUNCSEL register

PWM_FUNCSEL **Function Mode Selection** **Reset:00000004**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									FAULTIE	FAULTMODE			PWMSYNC	WPDIS	INIT	PWMSYNCEN
Type									RW	RW			RW	RW	RW	RW
Reset									0	0			0	1	0	0

Field	Description
7 FAULTIE	**Fault Interrupt Enable** 0: Fault control interrupt is disabled. 1: Fault control interrupt is enabled. Enables the generation of an interrupt when a fault is detected by PWM and the PWM fault control is enabled.
6: 5 FAULTMODE	**Fault Control Mode** 00: Fault control is disabled for all channels. 01: Fault control is enabled for even channels only (channels 0, 2), and the selected mode is the manual fault clearing. 10: Fault control is enabled for all channels, and the selected mode is the manual fault clearing.

Field	Description
	11: Fault control is enabled for all channels, and the selected mode is the automatic fault clearing. Defines the PWM fault control mode. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
3 PWMSYNC	**PWM Synchronization Mode** 0: No restrictions. Software and hardware triggers can be used by MCVR, CHnV, OMCR and PWM counter synchronization. 1: Software trigger can only be used by MCVR and CHnV synchronization, and hardware triggers can only be used by OMCR and PWM counter synchronization. Selects which triggers can be used by MCVR, CHnV, OMCR, and PWM counter synchronization.
2 WPDIS	Write Protection Disable 0: Write protection is enabled. 1: Write protection is disabled. Only 1 is allowed to be written, and 0 is invalid. When write protection is enabled (WPDIS = 0), write protected bits cannot be written. When write protection is disabled (WPDIS = 1), write protected bits can be written. The WPDIS bit is the negation of the WPEN bit. WPDIS is cleared when 1 is written to WPEN, that is, enable write protection. WPEN is cleared when WPDIS bit is written as a 1, that is, disable write protection.
1 INIT	**Initialize the Channels Output** When a 1 is written to INIT bit, the channels output is initialized according to the state of their corresponding bit in the OUTINIT register. Writing a 0 to INIT bit has no effect. The INIT bit is always read as 0.
0 PWMSYNCEN	**PWM Synchronization Enable** 0: Disable PWM synchronization 1: Enable PWM synchronization This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.

10.5.9 Synchronization Register(PWM_SYNC)

Table 10-17 PWM_SYNC register

PWM_SYNC **Synchronization register** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name					PWM_SYNCPOL				SWSYNC	TRIG2	TRIG1	TRIG0	OMSYNCP		MAXSYNCP	MINSYNCP
Type					RW				RW	RW	RW	RW	RW		RW	RW
Reset					0				0	0	0	0	0		0	0

Field	Description
11 PWM_SYNCPOL	**PWM_SYNCPOL** 0: POL register is updated with the value of its buffer in all rising edges of the system clock. 1: POL register is updated with the value of its buffer only by the PWM synchronization. Selects when the CHOPOLCR register is updated with the value of its buffer.
7 SWSYNC	**PWM Synchronization Software Trigger** 0: Software trigger is not selected. 1: Software trigger is selected. Selects the software trigger as the PWM synchronization trigger. The software trigger happens when a 1 is written to SWSYNC bit.
6 TRIG2	**PWM Synchronization Hardware Trigger 2** 0: Trigger is disabled. 1: Trigger is enabled. Enables PWM channel 0 output to the PWM synchronization hardware trigger source. Hardware trigger happens when a rising edge is detected at the trigger input signal.
5 TRIG1	**PWM Synchronization Hardware Trigger 1** 0: Trigger is disabled. 1: Trigger is enabled.

Field	Description
	Enables CTU control bit PWMTRIG output to the PWM synchronization hardware trigger source. Hardware trigger 1 happens when a rising edge is detected at the trigger 1 input signal.
4 TRIG0	**PWM Synchronization Hardware Trigger 0** 0: Trigger is disabled. 1: Trigger is enabled. Enables ACMP0 output to the PWM synchronization hardware trigger source. Hardware trigger 0 happens when a rising edge is detected at the trigger 0 input signal.
3 OMSYNCP	**Output Mask Synchronization** 0: OMCR register is updated with the value of its buffer in all rising edges of the system clock. 1: OMCR register is updated with the value of its buffer only by the PWM synchronization. Selects when the OMCR register is updated with the value of its buffer.
1 MAXSYNCP	**Maximum Loading Point Enable** 0: The maximum loading point is disabled. 1: The maximum loading point is enabled. Selects the maximum loading point to PWM synchronization. If MAXSYNCP is 1, the selected loading point is when the PWM counter reaches its maximum value (MCVR register).
0 MINSYNCP	**Minimum Loading Point Enable** 0: The minimum loading point is disabled. 1: The minimum loading point is enabled. Selects the minimum loading point to PWM synchronization. If MINSYNCP is 1, the selected loading point is when the PWM counter reaches its minimum value (CNTIN register).

10.5.10 Initial State for Channels Output(PWM_OUTINIT)

Table 10-18 PWM_OUTINIT register

PWM_OUTINIT **Initial State For Channels Output** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name													CH3OIV	CH2OIV	CH1OIV	CH0OIV
Type													RW	RW	RW	RW
Reset													0	0	0	0

Field	Description
3 CH3OIV	**Channel 3 Output Initialization Value** 0: The initialization value is 0. 1: The initialization value is 1 Selects the value that is forced into the channel output when the initialization occurs.
2 CH2OIV	**Channel 2 Output Initialization Value** 0: The initialization value is 0. 1: The initialization value is 1 Selects the value that is forced into the channel output when the initialization occurs.
1 CH1OIV	**Channel 1 Output Initialization Value** 0: The initialization value is 0. 1: The initialization value is 1 Selects the value that is forced into the channel output when the initialization occurs.
0 CH0OIV	**Channel 0 Output Initialization Value** 0: The initialization value is 0. 1: The initialization value is 1 Selects the value that is forced into the channel output when the initialization occurs.

10.5.11 Output Mask Control Register(PWM_OMCR)

Table 10-19 PWM_OMCR register

PWM_OMCR **Output Mask Control Register** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name													CH3OMEN	CH2OMEN	CH1OMEN	CH0OMEN
Type													RW	RW	RW	RW
Reset													0	0	0	0

Field	Description
3 CH3OMEN	**Channel 3 Output Mask** 0: Channel output is not masked. It continues to operate normally. 1: Channel output is masked. It is forced to its inactive state. Defines if the channel output is masked or unmasked.
2 CH2OMEN	**Channel 2 Output Mask** 0: Channel output is not masked. It continues to operate normally. 1: Channel output is masked. It is forced to its inactive state. Defines if the channel output is masked or unmasked.
1 CH1OMEN	**Channel 1 Output Mask** 0: Channel output is not masked. It continues to operate normally. 1: Channel output is masked. It is forced to its inactive state. Defines if the channel output is masked or unmasked.
0 CH0OMEN	**Channel 0 Output Mask** 0: Channel output is not masked. It continues to operate normally. 1: Channel output is masked. It is forced to its inactive state. Defines if the channel output is masked or unmasked.

10.5.12 Mode Selection Register(PWM_MODESEL)

Table 10-20 PWM_MODESEL register

PWM_MODESEL **PWM Function Mode Selection** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name		PAIR1FAULTEN	PAIR1SYNCEN	PAIR1DTEN	PAIR1DECAP	PAIR1DECAPEN	PAIR1COMPEN	PAIR1COMBINEN		PAIR0FAULTEN	PAIR0SYNCEN	PAIR0DTEN	PAIR0DECAP	PAIR0DECAPEN	PAIR0COMPEN	PAIR0COMBINEN
Type		RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW
Reset		0	0	0	0	0	0	0		0	0	0	0	0	0	0

Field	Description
14 PAIR1FAULTEN	**Fault Control Enable for n = 2** 0: The fault control in this pair of channels is disabled. 1: The fault control in this pair of channels is enabled. Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
13 PAIR1SYNCEN	**Synchronization Enable for n = 2** 0: The PWM synchronization in this pair of channels is disabled. 1: The PWM synchronization in this pair of channels is enabled. Enables PWM synchronization of registers CH(n)V and CH(n+1)V.
12 PAIR1DTEN	**Dead-time Enable for n = 2** 0: The dead-time insertion in this pair of channels is disabled. 1: The dead-time insertion in this pair of channels is enabled. Enables the dead-time insertion in the channels (n) and (n+1). This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
11 PAIR1DECAP	**Dual Edge Capture Mode Captures for n = 2** 0: The dual edge captures are inactive. 1: The dual edge captures are active.

Field	Description
	Enables the capture of the PWM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1. DECAP bit is cleared automatically by hardware if dual edge capture one-shot mode is selected and when the capture of channel (n+1) event is made.
10 PAIR1DECAPEN	**Dual Edge Capture Mode Enable for n = 2** .0: The Dual Edge Capture mode in this pair of channels is disabled. 1: The Dual Edge Capture mode in this pair of channels is enabled. Enables the Dual Edge Capture mode in the channels (n) and (n+1). This bit reconfigures the function of MSnR0, ELSnR1:ELSnR0 and ELS(n+1)R1:ELS(n+1)R0 bits in Dual Edge Capture mode. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
9 PAIR1COMPEN	**Complement of Channel (n) for n = 2** 0: The channel (n+1) output is the same as the channel (n) output. 1: The channel (n+1) output is the complement of the channel (n) output. Enables Complementary mode for the combined channels. In Complementary mode the channel (n+1) output is the inverse of the channel(n) output. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
8 PAIR1COMBINEN	**Combine Channels for n = 2** 0: Channels (n) and (n+1) are independent. 1: Channels (n) and (n+1) are combined. Enables the combine feature for channels (n) and (n+1). This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
6 PAIR0FAULTEN	**Fault Control Enable for n = 0** 0: The fault control in this pair of channels is disabled. 1: The fault control in this pair of channels is enabled. Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
5 PAIR0SYNCEN	**Synchronization Enable for n = 0** 0: The PWM synchronization in this pair of channels is disabled. 1: The PWM synchronization in this pair of channels is enabled. Enables PWM synchronization of registers CH(n)V and CH(n+1)V.
4 PAIR0DTEN	**Dead-time Enable for n = 0** 0: The dead-time insertion in this pair of channels is disabled. 1: The dead-time insertion in this pair of channels is enabled.

Field	Description
	Enables the dead-time insertion in the channels (n) and (n+1). This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
3 PAIR0DECAP	**Dual Edge Capture Mode Captures for n = 0** 0: The dual edge captures are inactive. 1: The dual edge captures are active. Enables the capture of the PWM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1. DECAP bit is cleared automatically by hardware if dual edge capture one-shot mode is selected and when the capture of channel (n+1) event is made.
2 PAIR0DECAPEN	**Dual Edge Capture Mode Enable for n = 0** 0: The Dual Edge Capture mode in this pair of channels is disabled. 1: The Dual Edge Capture mode in this pair of channels is enabled. Enables the Dual Edge Capture mode in the channels (n) and (n+1). This bit reconfigures the function of MSnR0, ELSnR1:ELSnR0 and ELS(n+1)R1:ELS(n+1)R0 bits in Dual Edge Capture mode. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
1 PAIR0COMPEN	**Complement of Channel (n) for n = 0** 0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output. Enables Complementary mode for the combined channels. In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
0 PAIR0COMBINEN	**Combine Channels for n = 0** 0: Channels (n) and (n+1) are independent. 1: Channels (n) and (n+1) are combined. Enables the combine feature for channels (n) and (n+1). This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.

10.5.13 Dead-time Setting Register(PWM_DTSET)

Table 10-21 PWM_DTSET register

PWM_DTSET **Dead-time Insertion Control** **Reset:00000000**

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

PWM_DTSET **Dead-time Insertion Control** **Reset:00000000**

Type		RW	RW
Reset		0	0

Field	Description
7: 6 DTPSC	**Dead-time Prescaler Value** 0x: Divide the system clock by 1. 10: Divide the system clock by 4. 11: Divide the system clock by 16. Selects the division factor of the system clock. This prescaled clock is used by the dead-time counter. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
5: 0 DTVAL	**Dead-time Value** Selects the dead-time insertion value for the dead-time counter. The dead-time counter is clocked by a scaled version of the system clock. See the description of DTPS. Dead-time insert value = (DTPSC x DTVAL). DTVAL selects the number of dead-time counts inserted as follows: When DTVAL is 0, no counts are inserted. When DTVAL is 1, 1 count is inserted. When DTVAL is 2, 2 counts are inserted. This pattern continues up to a possible 63 counts. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.

10.5.14 External Trigger Register(PWM_EXTTRIG)

Table 10-22 PWM_EXTTRIG register

PWM_EXTTRIG **PWM External Trigger** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name							TRIGF	INITTRIGEN					CH3TRIG	CH2TRIG	CH1TRIG	CH0TRIG
Type							R/W0C	RW					RW	RW	RW	RW
Reset							0	0					0	0	0	0

Field	Description
9 TRIGF	**Channel Trigger Flag**

Field	Description
	0: No channel trigger was generated. 1: A channel trigger was generated. Set by hardware when a channel trigger is generated. Clear TRIGF by reading EXTTRIG while TRIGF is set and then writing a 0 to TRIGF. If another channel trigger is generated before the clearing sequence is completed, the sequence is reset so TRIGF remains set after the clear sequence is completed for the earlier TRIGF.
8 INITTRIGEN	**Initialization Trigger Enable** 0: The generation of initialization trigger is disabled. 1: The generation of initialization trigger is enabled. Enables the generation of the trigger when the PWM counter is equal to the CNTIN register.
3 CH3TRIG	**Channel 3 Trigger Enable** 0: The generation of the channel trigger is disabled. 1: The generation of the channel trigger is enabled. Enable the generation of the channel trigger when the PWM counter is equal to the CHnV register.
2 CH2TRIG	**Channel 2 Trigger Enable** 0: The generation of the channel trigger is disabled. 1: The generation of the channel trigger is enabled. Enables the generation of the channel trigger when the PWM counter is equal to the CHnV register.
1 CH1TRIG	**Channel 1 Trigger Enable** 0: The generation of the channel trigger is disabled. 1: The generation of the channel trigger is enabled. Enables the generation of the channel trigger when the PWM counter is equal to the CHnV register.
0 CH0TRIG	**Channel 0 Trigger Enable** 0: The generation of the channel trigger is disabled. 1: The generation of the channel trigger is enabled. Enables the generation of the channel trigger when the PWM counter is equal to the CHnV register.

10.5.15 Channel Output Polarity Control Register(PWM_CHOPOLCR)

Table 10-23 PWM_CHOPOLCR register

PWM_CHOPOLCR **Channels Output Polarity Control Register** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name													CH3POL	CH2POL	CH1POL	CH0POL
Type													RW	RW	RW	RW
Reset													0	0	0	0

Field	Description
3 CH3POL	**Channel 3 Polarity** 0: The channel polarity is active high. 1: The channel polarity is active low. Defines the polarity of the channel output. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
2 CH2POL	**Channel 2 Polarity** 0: The channel polarity is active high. 1: The channel polarity is active low. Defines the polarity of the channel output. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
1 CH1POL	**Channel 1 Polarity** 0: The channel polarity is active high. 1: The channel polarity is active low. Defines the polarity of the channel output. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
0 CH0POL	**Channel 0 Polarity** 0: The channel polarity is active high. 1: The channel polarity is active low. Defines the polarity of the channel output. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.

10.5.16 Fault Detect Status Register(PWM_FDSR)

Table 10-24 PWM_FDSR register

PWM_FDSR **Fault Detect Status Register** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									FAULTDF	WPEN	FAULTIN			FAULTDF2	FAULTDF1	FAULTDF0
Type									R/W0C	RW	RO			R/W0C	R/W0C	R/W0C
Reset									0	0	0			0	0	0

Field	Description
7 FAULTDF	**Fault Detection Flag** 0: No fault condition was detected. 1 : A fault condition was detected. Represents the logic OR of the individual FAULTDF j bits where j = 2, 1, 0. Clear FAULTDF by reading the PWM_FDSR register while FAULTDF is set and then writing a 0 to FAULTDF while there is no existing fault condition at the enabled fault inputs. If another fault condition is detected in an enabled fault input before the clearing sequence is completed, the sequence is reset so FAULTDF remains set after the clearing sequence is completed for the earlier fault condition. FAULTDF is also cleared when FAULTDF j bits are cleared individually.
6 WPEN	**Write Protection Enable** 0: Write protection is disabled. Write protected bits can be written. 1: Write protection is enabled. Write protected bits cannot be written. Only 1 is allowed to be written, and 0 is invalid. When write protection is enabled, write protected bits cannot be written. When write protection is disabled, write protected bits can be written. The WPEN bit is the negation of the WPDIS bit. WPEN is cleared when WPDIS bit is written as a 1, that is, disable write protection. WPDIS is cleared when 1 is written to WPEN, that is, enable write protection.
5 FAULTIN	**Fault Inputs** 0 : The logic OR of the enabled fault inputs is 0. 1 : The logic OR of the enabled fault inputs is 1. Represents the logic OR of the enabled fault inputs after their filter (if their filter is enabled) when fault control is enabled.

Field	Description
2 FAULTDF2	**Fault Detection Flag 2** 0 : No fault condition was detected at the fault input. 1 : A fault condition was detected at the fault input. Clear FAULTDF2 by reading the PWM_FDSR register while FAULTDF2 is set and then writing a 0 to FAULTDF2. Writing a 1 to FAULTDF2 has no effect. FAULTDF2 bit is also cleared when FAULTDF bit is cleared. If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the clear operation is invalid and the FAULTDF2 remains set.
1 FAULTDF1	**Fault Detection Flag 1** 0 : No fault condition was detected at the fault input. 1: A fault condition was detected at the fault input. Clear FAULTDF1 by reading the PWM_FDSR register while FAULTDF1 is set and then writing a 0 to FAULTDF1. Writing a 1 to FAULTDF1 has no effect. FAULTDF1 bit is also cleared when FAULTDF bit is cleared. If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the clear operation is invalid and the FAULTDF1 remains set.
0 FAULTDF0	**Fault Detection Flag 0** 0 : No fault condition was detected at the fault input. 1 : A fault condition was detected at the fault input. Clear FAULTDF0 by reading the PWM_FDSR register while FAULTDF0 is set and then writing a 0 to FAULTDF0. Writing a 1 to FAULTDF0 has no effect. FAULTDF0 bit is also cleared when FAULTDF bit is cleared. If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the clear operation is invalid and the FAULTDF0 remains set.

10.5.17 Input Capture Mode Filter Control(PWM_CAPFILTER)

Table 10-25 PWM_CAPFILTER register

PWM_CAPFILTER **Input Capture Filter Control** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name													CH3CAPFVAL[4: 1]			
Type													RW			
Reset													0			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	CH3CAPFVAL[0: 0]	CH2CAPFVAL					CH1CAPFVAL						CH0CAPFVAL			

PWM_CAPFILTER **Input Capture Filter Control** **Reset:00000000**

Type	RW	RW	RW	RW
Reset	0	0	0	0

Field	Description
19: 15 CH3CAPFVAL	**Channel 3 Input Filter** Selects the filter value for the channel input. The filter is disabled when the value is 0.
14: 10 CH2CAPFVAL	**Channel 2 Input Filter** Selects the filter value for the channel input. The filter is disabled when the value is 0.
9: 5 CH1CAPFVAL	**Channel 1 Input Filter** Selects the filter value for the channel input. The filter is disabled when the value is 0.
4: 0 CH0CAPFVAL	**Channel 0 Input Filter** Selects the filter value for the channel input. The filter is disabled when the value is 0.

10.5.18 Fault Filter and Fault Enable Register(PWM_FFAFER)

Table 10-26 PWM_FFAFER register

PWM_FFAFER **Fault Filter and Fault Enable Register** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	FFVAL									FF2EN	FF1EN	FF0EN		FER2EN	FER1EN	FER0EN
Type	RW									RW	RW	RW		RW	RW	RW
Reset	0									0	0	0		0	0	0

Field	Description
15: 8 FFVAL	**Fault Input Filter** Selects the filter value for the fault inputs. The fault filter is disabled when the value is zero.
6 FF2EN	**Fault Input 2 Filter Enable** 0: Fault input filter is disabled. 1: Fault input filter is enabled.

Field	Description
	Enables the filter for the fault input. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
5 FF1EN	**Fault Input 1 Filter Enable** 0: Fault input filter is disabled. 1: Fault input filter is enabled. Enables the filter for the fault input. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
4 FF0EN	**Fault Input 0 Filter Enable** 0: Fault input filter is disabled. 1: Fault input filter is enabled. Enables the filter for the fault input. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
2 FER2EN	**Fault Input 2 Enable** 0: Fault input is disabled. 1: Fault input is enabled. Enables the fault input. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
1 FER1EN	**Fault Input 1 Enable** 0: Fault input is disabled. 1: Fault input is enabled. Enables the fault input. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
0 FER0EN	**Fault Input 0 Enable** 0: Fault input is disabled. 1: Fault input is enabled. Enables the fault input. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.

10.5.19 Quadrature Decoder Control and Status(PWM_QDI)

Table 10-27 PWM_QDI register

PWM_QDI **Quadrature Decoder Interface Configuration Register** **Reset:00000000**

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

PWM_QDI **Quadrature Decoder Interface Configuration Register** **Reset:00000000**

Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name											PHAPOL	PHBPOL	QUADMODE	QUADIR	CNTOFDIR	QDIEN
Type											RW	RW	RW	RO	RO	RW
Reset											0	0	0	1	0	0

Field	Description
5 PHAPOL	**Phase A Input Polarity** 0: Normal polarity. Phase A input signal is not inverted before identifying the rising and falling edges of this signal. 1: Inverted polarity. Phase A input signal is inverted before identifying the rising and falling edges of this signal. Selects the polarity for the quadrature decoder phase A input.
4 PHBPOL	**Phase B Input Polarity** 0: Normal polarity. Phase B input signal is not inverted before identifying the rising and falling edges of this signal. 1: Inverted polarity. Phase B input signal is inverted before identifying the rising and falling edges of this signal. Selects the polarity for the quadrature decoder phase B input.
3 QUADMODE	**Quadrature Decoder Mode** 0: Phase A and phase B encoding mode. 1: Count and direction encoding mode. Selects the encoding mode used in the Quadrature Decoder mode.
2 QUADIR	**PWM Counter Direction in Quadrature Decoder Mode** 0: Counting direction is decreasing (PWM counter decrement). 1: Counting direction is increasing (PWM counter increment). Indicates the counting direction.
1 CNTOFDIR	**Timer Overflow Direction in Quadrature Decoder Mode** 0: CNTOF bit was set on the bottom of counting. There was an PWM counter decrement and PWM counter changes from its minimum value (CNTIN register) to its maximum value (MCVR register). 1: CNTOF bit was set on the top of counting. There was an PWM counter increment and PWM counter changes from its maximum value (MCVR register) to its minimum value (CNTIN register).

Field	Description
	Indicates if the TOF bit was set on the top or the bottom of counting.
0 QDIEN	**Quadrature Decoder Mode Enable** 0: Quadrature Decoder mode is disabled. 1: Quadrature Decoder mode is enabled. Enables the Quadrature Decoder mode. In this mode, the phase A and B input signals control the PWM counter direction. The Quadrature Decoder mode has precedence over the other modes. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.

10.5.20 Configuration Register(PWM_CONF)

Table 10-28 PWM_CONF register

PWM_CONF **Configuration register** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name									EVENT3PSC		EVENT2PSC		EVENT1PSC		EVENT0PSC	
Type									RW		RW		RW		RW	
Reset									0		0		0		0	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name						GTBEOUT	GTBEEN		CNTOFNUM							
Type						RW	RW		RW							
Reset						0	0		0							

Field	Description
23: 22 EVENT3PSC	**Channel 3 input event prescaler** 00: 1 input event triggers a capture 01: 2 input event triggers a capture 10: 4 input event triggers a capture 11: 8 input event triggers a capture
21: 20 EVENT2PSC	**Channel 2 input event prescaler** 00: 1 input event triggers a capture 01: 2 input event triggers a capture 10: 4 input event triggers a capture 11: 8 input event triggers a capture
19: 18 EVENT1PSC	**Channel 1 input event prescaler**

Field	Description
	00: 1 input event triggers a capture 01: 2 input event triggers a capture 10: 4 input event triggers a capture 11: 8 input event triggers a capture
17: 16 EVENT0PSC	**Channel 0 input event prescaler** 00: 1 input event triggers a capture 01: 2 input event triggers a capture 10: 4 input event triggers a capture 11: 8 input event triggers a capture
10 GTBEOUT	**Global time base output** 0: disable GTB signal generation 1: enable GTB signal generation Enables the global time base signal generation of other PWM.
9 GTBEEN	**Enable global time base** 0: disable external global time base 1: enable external global time base
6: 0 CNTOFNUM	**CNTOF Frequency** Selects the ratio between the number of counter overflows to the number of times the CNTOF bit is set. CNTOFNUM = 0: The CNTOF bit is set for each counter overflow. CNTOFNUM = 1: The CNTOF bit is set for the first counter overflow but not for the next overflow. CNTOFNUM = 2: The CNTOF bit is set for the first counter overflow but not for the next 2 overflows. CNTOFNUM = 3: The CNTOF bit is set for the first counter overflow but not for the next 3 overflows.

10.5.21 Fault Input Polarity Register(PWM_FLTPOL)

Table 10-29 PWM_FLTPOL register

PWM_FLTPOL **PWM Fault Input Polarity** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name														FLT2POL	FLT1POL	FLT0POL
Type														RW	RW	RW
Reset														0	0	0

Field	Description
2 FLT2POL	**Fault Input 2 Polarity** 0: The fault input polarity is active high. A one at the fault input indicates a fault. 1: The fault input polarity is active low. A zero at the fault input indicates a fault. Defines the polarity of the fault input. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
1 FLT1POL	**Fault Input 1 Polarity** 0: The fault input polarity is active high. A one at the fault input indicates a fault. 1: The fault input polarity is active low. A zero at the fault input indicates a fault. Defines the polarity of the fault input. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.
0 FLT0POL	**Fault Input 0 Polarity** 0: The fault input polarity is active high. A one at the fault input indicates a fault. 1: The fault input polarity is active low. A zero at the fault input indicates a fault. Defines the polarity of the fault input. This field is write protected. It can be written only when FUNCSEL[WPDIS] = 1.

10.5.22 Synchronization Configuration Register(PWM_SYNCONF)

Table 10-30 PWM_SYNCONF register

PWM_SYNCONF **Synchronization Configuration** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name										HWPOL	SWPOL	SWVHWSYNC	INVHWSYNC	OMVHWSYNC	PWMSVHWSYNC	CNTVHWSYNC
Type										RW	RW	RW	RW	RW	RW	RW
Reset										0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name				SWVSWSYNC	INVSWSYNC	OMVSWSYNC	PWMSVSWSYNC	CNTVSWSYNC	SYNCMODE		SWOC	INVC		CNTINC		HWTRIGMODESEL
Type				RW	RW	RW	RW	RW	RW		RW	RW		RW		RW
Reset				0	0	0	0	0	0		0	0		0		0

Field	Description
22 HWPOL	**Channel CHPOLCR synchronization is activated by a hardware trigger.** 0: A hardware trigger does not activate the CHPOLCR register synchronization. 1: A hardware trigger activates CHPOLCR register synchronization.
21 SWPOL	**Channel CHPOLCR synchronization is activated by the software trigger.** 0: The software trigger does not activate the CHPOLCR register synchronization. 1: The software trigger activates CHPOLCR register synchronization.
20 SWVHWSYNC	**Software output control synchronization is activated by a hardware trigger.** 0: A hardware trigger does not activate the CHOSWCR register synchronization. 1 : A hardware trigger activates the CHOSWCR register synchronization.
19 INVHWSYNC	**Inverting control synchronization is activated by a hardware trigger.**

Field	Description
	0: A hardware trigger does not activate the INVCR register synchronization. 1: A hardware trigger activates the INVCR register synchronization.
18 OMVHWSYNC	**Output mask synchronization is activated by a hardware trigger.** 0 : A hardware trigger does not activate the OMCR register synchronization. 1: A hardware trigger activates the OMCR register synchronization.
17 PWMSVHWSYNC	**MCVR, CNTIN, and CHV registers synchronization is activated by a hardware trigger.** 0 : A hardware trigger does not activate MCVR, CNTIN, and CHV registers synchronization. 1: A hardware trigger activates MCVR, CNTIN, and CHV registers synchronization.
16 CNTVHWSYNC	**PWM counter synchronization is activated by a hardware trigger.** 0 : A hardware trigger does not activate the PWM counter synchronization. 1 : A hardware trigger activates the PWM counter synchronization.
12 SWVSWSYNC	**Software output control synchronization is activated by the software trigger.** 0: The software trigger does not activate the CHOSWCR register synchronization. 1: The software trigger activates the CHOSWCR register synchronization.
11 INVSWSYNC	**Inverting control synchronization is activated by the software trigger.** 0 : The software trigger does not activate the INVCR register synchronization. 1 : The software trigger activates the INVCR register synchronization.
10 OMVSWSYNC	**Output mask synchronization is activated by the software trigger.** 0 : The software trigger does not activate the OMCR register synchronization. 1: The software trigger activates the OMCR register synchronization.
9 PWMSVSWSYNC	**MCVR, CNTIN, and CHV registers synchronization is activated by the software trigger.** 0 : The software trigger does not activate MCVR, CNTIN, and CHV registers synchronization. 1: The software trigger activates MCVR, CNTIN, and CHV registers synchronization.
8 CNTVSWSYNC	**PWM counter synchronization is activated by the software trigger.** 0 : The software trigger does not activate the PWM counter synchronization. 1 : The software trigger activates the PWM counter synchronization.
7 SYNCMODE	**Synchronization Mode** 0: Legacy PWM synchronization is selected. 1 : Enhanced PWM synchronization is selected.

Field	Description
	Selects the PWM Synchronization mode.
5 SWOC	**CHOSWCR Register Synchronization** 0 : CHOSWCR register is updated with its buffer value at all rising edges of system clock. 1 : CHOSWCR register is updated with its buffer value by the PWM synchronization.
4 INVC	**INVCR Register Synchronization** 0: INVCR register is updated with its buffer value at all rising edges of system clock. 1: INVCR register is updated with its buffer value by the PWM synchronization.
2 CNTINC	**CNTIN Register Synchronization** 0 : CNTIN register is updated with its buffer value at all rising edges of system clock. 1: CNTIN register is updated with its buffer value by the PWM synchronization.
0 HWTRIGMODESEL	**Hardware Trigger Mode** 0: PWM clears the TRIG j bit when the hardware trigger j is detected, where j = 0, 1, 2. 1: PWM does not clear the TRIG j bit when the hardware trigger j is detected, where j = 0, 1, 2.

10.5.23 Inverting Control Register(PWM_INVCR)

Table 10-31 PWM_INVCR register

PWM_INVCR **PWM Inverting Control Register** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name															PAIR1INVEN	PAIR0INVEN
Type															RW	RW
Reset															0	0

Field	Description
1 PAIR1INVEN	**Pair Channels1 Inverting Enable**

Field	Description
	0: Inverting is disabled. 1: Inverting is enabled.
0 PAIR0INVEN	**Pair Channels 0 Inverting Enable** 0: Inverting is disabled. 1: Inverting is enabled.

10.5.24 Channel Software Output Control Register(PWM_CHOSWCR)

Table 10-32 PWM_CHOSWCR register

PWM_CHOSWCR **PWM Channel Software Output Control Register** **Reset:00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name					CH3SWCV	CH2SWCV	CH1SWCV	CH0SWCV					CH3SWEN	CH2SWEN	CH1SWEN	CH0SWEN
Type					RW	RW	RW	RW					RW	RW	RW	RW
Reset					0	0	0	0					0	0	0	0

Field	Description
11 CH3SWCV	**Channel 3 Software Output Control Value** 0: The software output control forces 0 to the channel output. 1: The software output control forces 1 to the channel output.
10 CH2SWCV	**Channel 2 Software Output Control Value** 0: The software output control forces 0 to the channel output. 1: The software output control forces 1 to the channel output.
9 CH1SWCV	**Channel 1 Software Output Control Value** 0: The software output control forces 0 to the channel output. 1: The software output control forces 1 to the channel output.
8 CH0SWCV	**Channel 0 Software Output Control Value** 0: The software output control forces 0 to the channel output. 1: The software output control forces 1 to the channel output.
3 CH3SWEN	**Channel 3 Software Output Control Enable** 0: The channel output is not affected by software output control. 1 : The channel output is affected by software output control.
2	**Channel 2 Software Output Control Enable**

Field	Description
CH2SWEN	
	0: The channel output is not affected by software output control.
	1 : The channel output is affected by software output control.
1	**Channel 1 Software Output Control Enable**
CH1SWEN	
	0: The channel output is not affected by software output control.
	1 : The channel output is affected by software output control.
0	**Channel 0 Software Output Control Enable**
CH0SWEN	
	0: The channel output is not affected by software output control.
	1 : The channel output is affected by software output control.

11 Pulse Width Detect Timer(PWDT)

11.1 Introduction

Pulse Width Detect Timer (PWDT) is used to measure the pulse width or as a 16-bit timer. The MCU contains one PWDT module, which supports 3 external channels and 1 internal channel input.

11.2 Features

- 4 selectable pulse clock input.
- Support 2 functions: pulse width measurement function and timer function.
  - For pulse width measurement function:
    - Programmable start trigger edge
    - 4 programmable measurement modes
    - Support 3 hall sensors signal input measurement
    - Support 3 inputs from analog comparator.
  - For timer function
- 16-bit counter used for pulse width measurement or timer function.
- Interrupts:
  - OVF: counter overflow interrupt
  - RDYF: pulse width measurement value ready interrupt

11.3 Block diagram

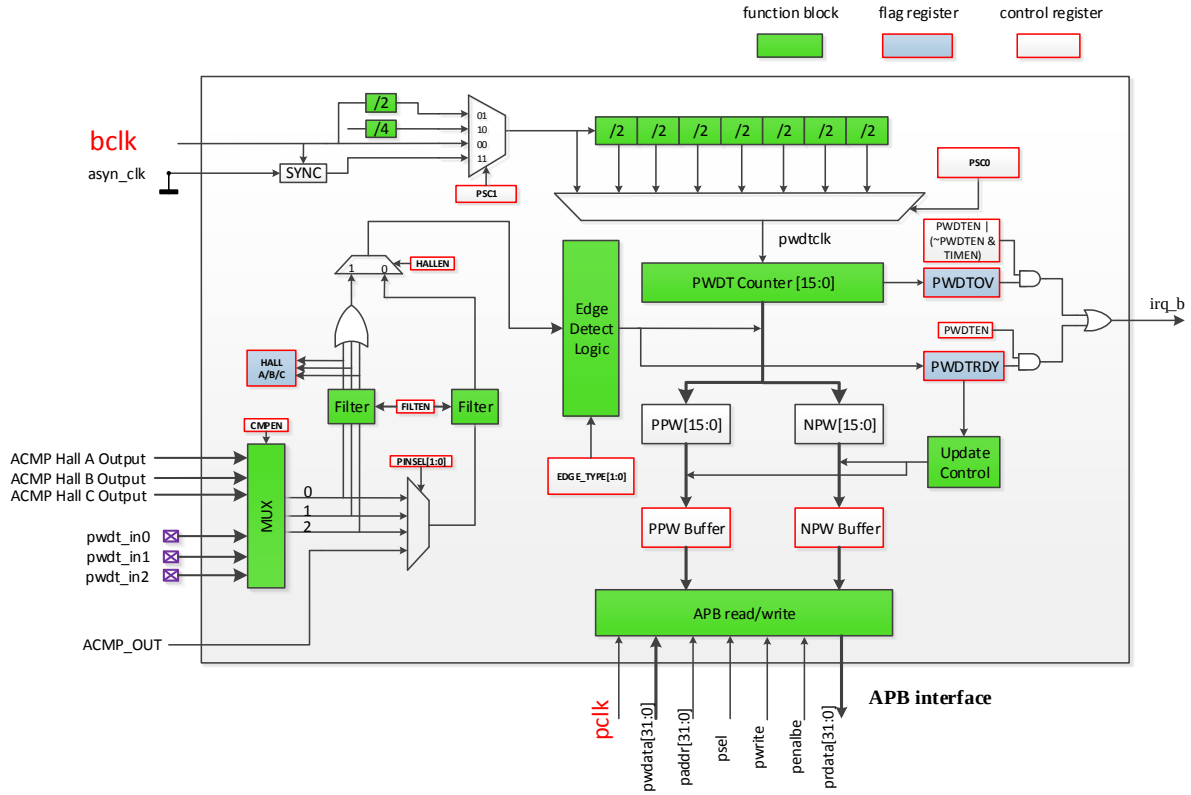

Figure 11-1 PWDT block diagram

11.4 Functional description

11.4.1 Pulse Width Measurement function

To get the pulse width measurement value, the PWDTC counter runs based on the pwdtclk described in block diagram after PWDTEN=1. And pwdtclk is deriving from the bus clock division by the PSC0[2:0] and PSC1[1:0], PSC1 is the pre-divider, PSC0 is the post-divider, the two values are multiplied and combined to form the total prescaler PSC. In order to get more accurate measurement value, user had better take the smaller value of PSC for narrower pulse input and larger value of PSC for wider pulse input.

For pulse width measurement, user must clearly know the application scenarios, which mainly include two conditions: one is just single channel input for general measurement and the other is 3 channel inputs for hall measurement.

11.4.1.1 General measurement mode

For general measurement, user choose to measure specific channel input by setting PINSEL[1:0] and can choose one of the 4 measurement modes by setting EDGE[1:0] based on the practical

applications, as illustrated in Figure 11-2. Among them, PPW is positive pulse width value and NPW is negative pulse width value respectively.

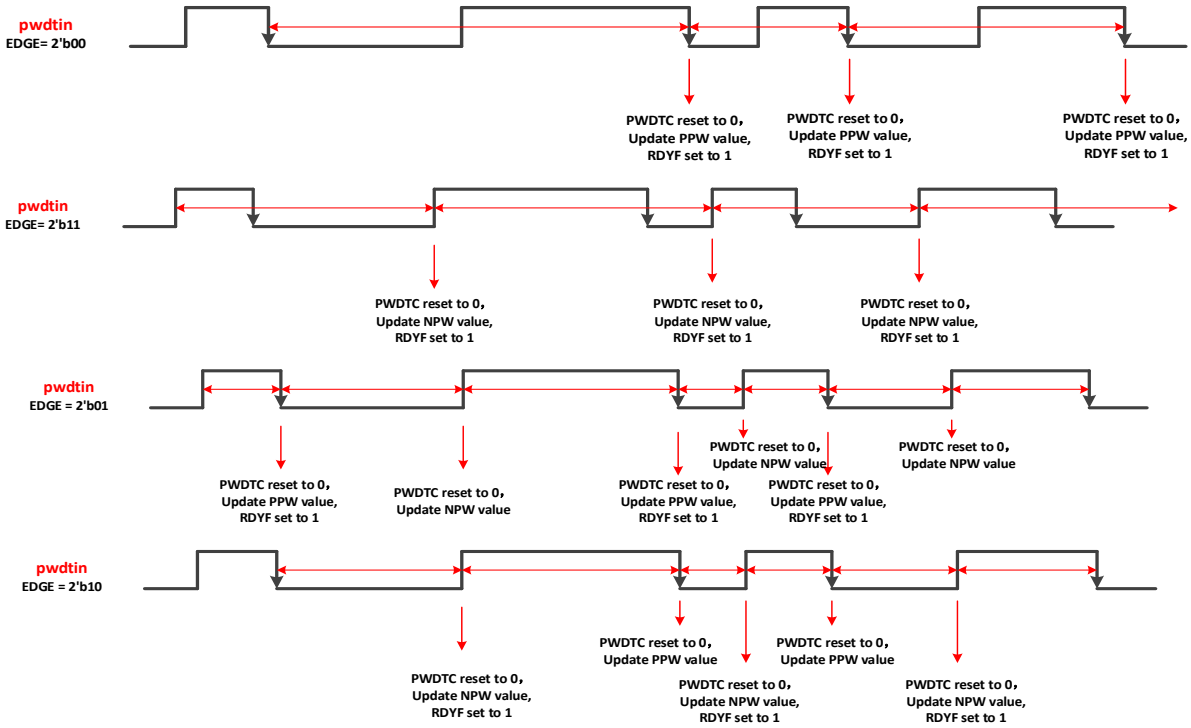

Figure 11-2 Four basic measurement modes(HALLEN=0)

11.4.1.2 Hall measurement mode

For hall measurement, user should set the EDGE[1:0] = 2'b01, HALLEN = 1'b1. When CMPEN = 1'b0, the module measures the pulse input obtained after XOR of input signals from 3 channels (pwdt_in0, pwdt_in1, pwdt_in2). When CMPEN = 1'b1, the module measures the pulse input obtained after XOR of the input signal from the ACMP HALL A/B/C Outpout channel.

For hall measurement, select this mode for motor speed calculation or commutation and is illustrated in Figure 11-3. Compared to the general measurement mode, the RDYF flag is set as 1 on both the rising edge and the falling edge.

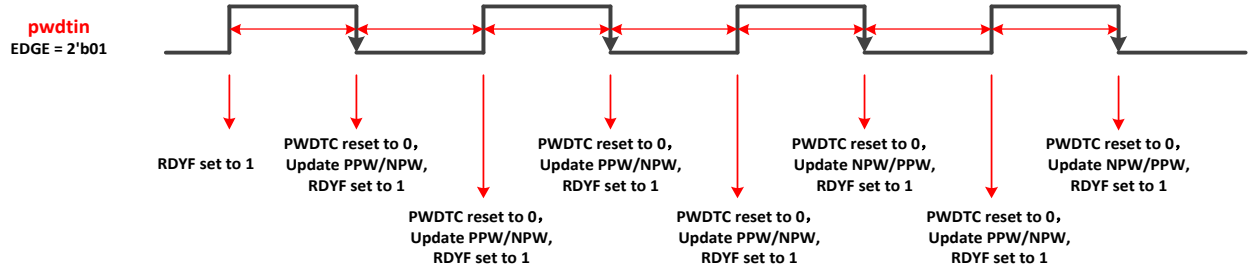

Figure 11-3 Hall measurement modes(HALLEN=1)

In the motors, installation of hall devices is used for detecting location of the rotor to commutate properly. And there usually two installations as Figure 11-4. One is 120 electrical degree interval and the other is 60 electrical degree interval.

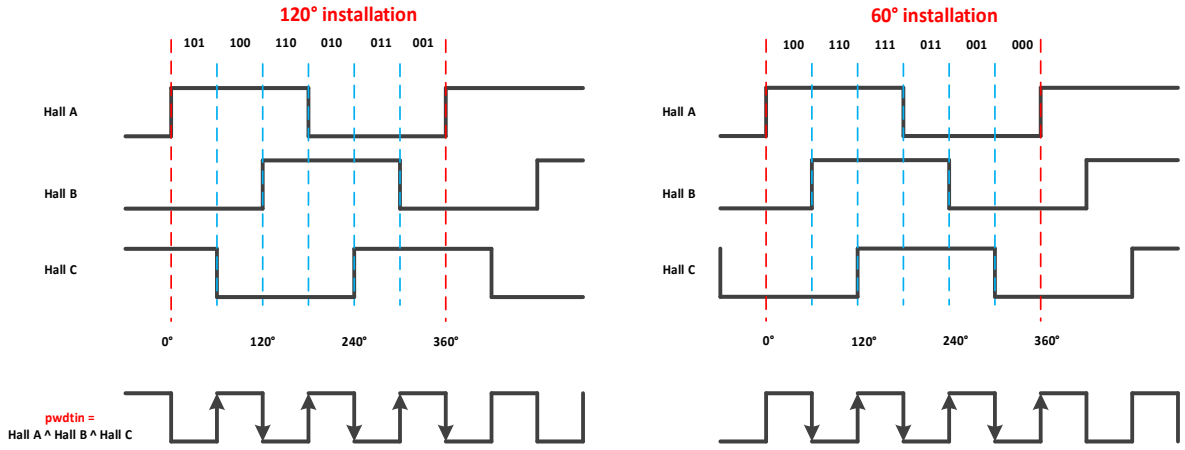

Figure 11-4 Two common installation ways

11.4.1.3 Input filter

Input filter is designed for filter the noise signal which high/low level is less than specific width described in the paragraph. The FILT_PSC[3:0] and FILTVAL[3:0] settings determine the maximum and minimum noise pulse width. The Figure 11-5 and Figure 11-6 introduce the noise width settings, judgement and filter. When user configures the FILTVAL=15 and FILT_PSC=2, the filter pulse width is 60 bclk and pulses less than 60 bclk are judged as noise pulse and will be filtered. The filterable pulse width is listed in Table 11-1.

Note: Enabling the filter function will cause signal delay (the delay time is the filter width). Therefore, if the PWDT module is running in the Hall measurement mode of the motor application, it may be necessary to consider the commutation time offset caused by the filter delay, and user can make corresponding software compensation based on the filter delay.

Table 11-1 Filterable pulse width range

Item	Clock	Time
Filterable pulse width range	2 bclk ~ 15*4096 bclk	0.04μs ~ 1.229ms

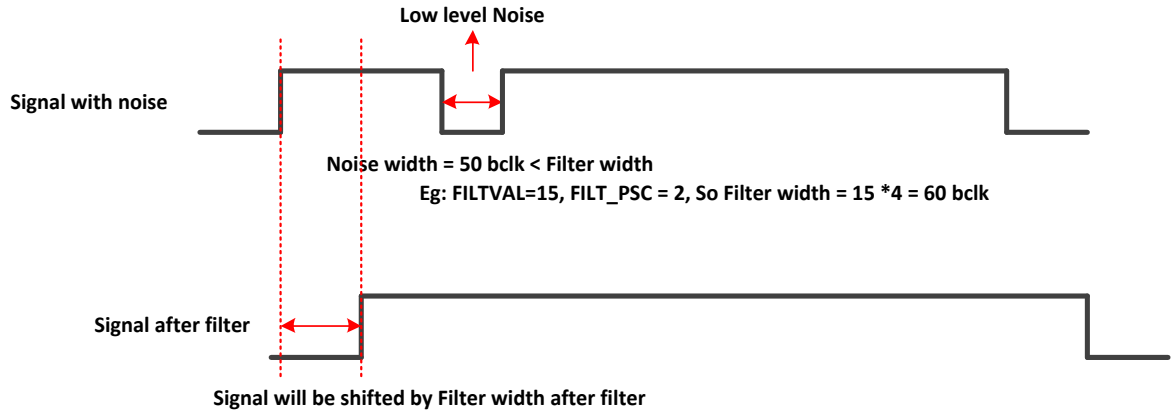

Figure 11-5 Example for low level noise and filter

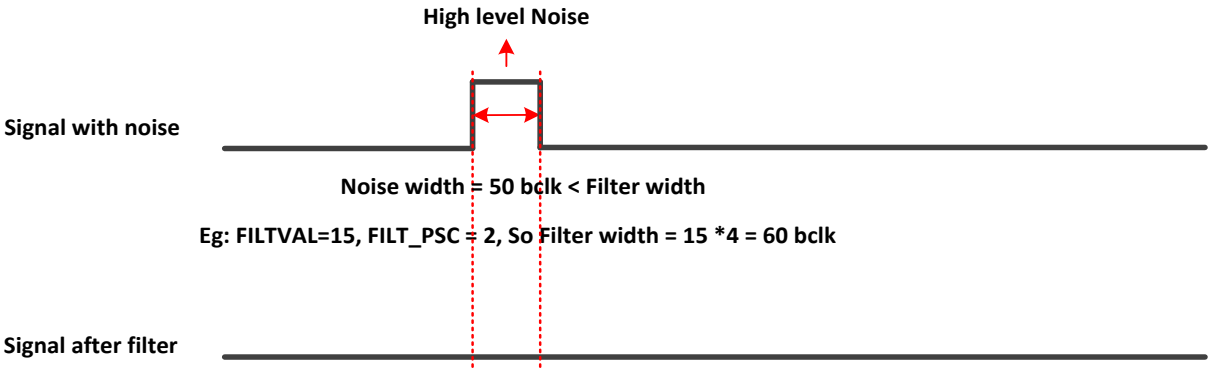

Figure 11-6 Example for high level noise and filter

11.4.1.4 Measurement error

User should understand the measurement accuracy for pulse width measurement to configure proper value of PSC to reach a more accurate measurement value. A basic principle is that more accurate measurement value can be gotten with smaller PSC. Obviously, more narrow input pulse, more relative measurement error. The Figure 11-7 describes the error when it operates for pulse width measurement function. In the Figure 11-7, the PWDTC counter and the pwdtclk divisor counter reset to 0 simultaneously when the pwdtin pulse changes from high to low or from low to high. And exactly here the counting error occurs which is deriving from the last counting value illustrated in Figure 11-7. The practical width value is less than measurement value by less than a pwdtclk period.

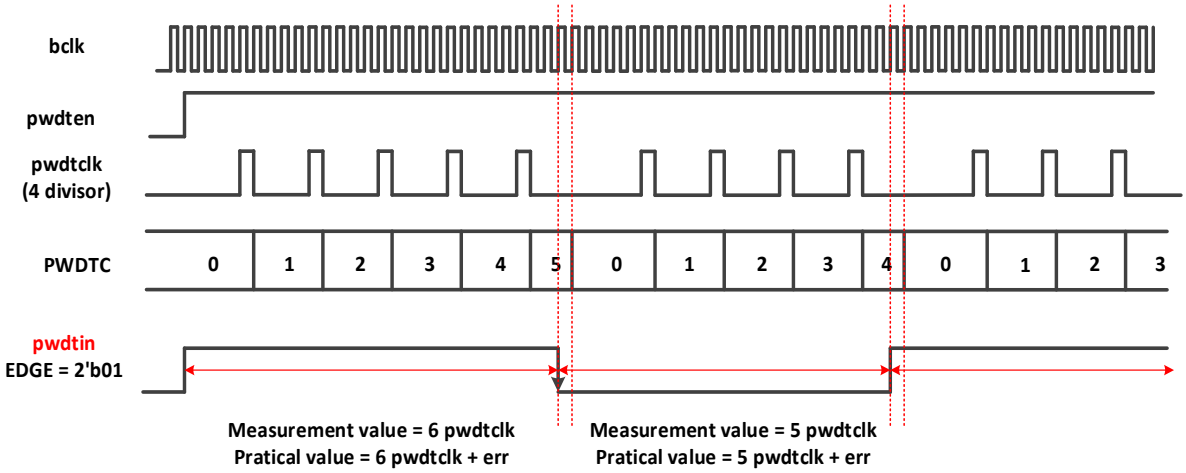

Figure 11-7 PWDTC counter and counting error

11.4.2 Timer function

For timer function, only OVF status is valid and occurs when PWDTC counter overflows. The counter load value TIMLDTVAL[15:0] can be modified all the time, but different timing of modifying the value leads to different operation in Figure 11-8 and Figure 11-9.

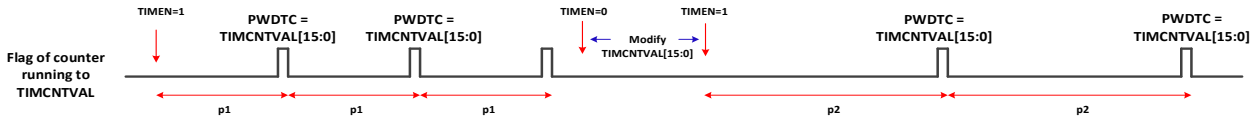

Figure 11-8 Modify the TIMLDTVAL during TIMEN=0

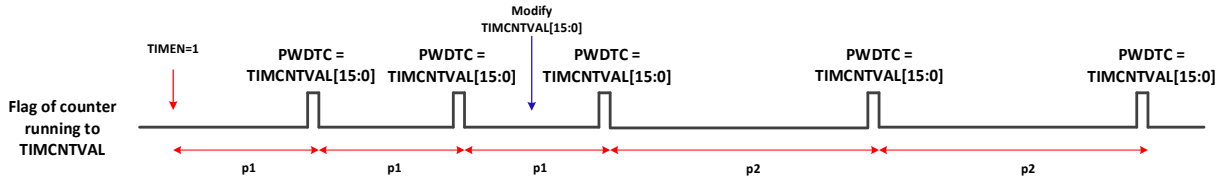

Figure 11-9 Modify the TIMLDTVAL during TIMEN=1

11.4.3 Interrupt request

Table 11-2 PWDT interrupt summary

Interrupt source	Flag	Local enable	Global enable
PWDT interrupt	OVF/RDYF	OVIE/PRDYIE	IE

11.4.4 Low power mode

Table 11-3 PWDT low power mode

Mode	Wakeup source	Description
Sleep mode	interrupt	The module works normally, and the MCU can be woken up through an interrupt
Stop mode	---	The module is disabled

11.5 Application note

11.5.1 Pulse Width Measurement function program guide

User must keep in mind that PWDTEN should be configured to 1 after all other controlling bits. Otherwise, abnormal condition may occur. Especially, HALLEN and CMPEN should be configured to 1 for the internal 3 comparator inputs.

11.5.2 Timer function program guide

Timer function is used easily with just configuration of the TIMLDVAL, PRESCALE and TIMEN and so on. And user should set TIMEN to 1 at last and must not set the PWDTEN to 1, because the pulse width measurement function is prior to timer function.

11.6 Register definition

Table 11-4 PWDT register map

PWDT0 Base address = 0x40017000

Address	Register name	Width	Description
PWDT0 base address +0x0	PWDT_INIT0	32	General control, status bits and positive pulse width contents
PWDT0 base address +0x04	PWDT_NPW	32	Negative pulse width content and 16-bit free running counter
PWDT0 base address +0x08	PWDT_INIT1	32	Hall function control and timer function control

11.6.1 Initialization Register 0(PWDT_INIT0)

Table 11-5 PWDT_INIT0 register

PWDT_INIT0 **PWDT Initialization Register 0** **Reset:00000000**

Bit	31~16	15~14	13~12	11~10	9~7	6	5	4	3	2	1	0
Name	PPW	PSC1	PINSEL	EDGE	PSC0	PWDTEN	IE	PRDYIE	OVIE		RDYF	OVF
Type	RO	RW	RW	RW	RW	RW	RW	RW	RW		R/W0C	R/W0C
Reset	0	0	0	0	0	0	0	0	0		0	0

Field	Description
31: 16 PPW	**Positive Pulse Width** Indicate the positive pulse width value.
15: 14 PSC1	**PWDT counter pre-prescaler** 00 ~ 11: respectively indicate 1/2/4/reserved.
13: 12 PINSEL	**Pin select** 00/01/10/11: respectively select pwdt_in0/ pwdt_in1/ pwdt_in2/ACMP_OUT.
11: 10 EDGE	**Selects the input edge trigger type** 00: the first falling edge starts, and all the subsequent falling edge trigger the pulse width to be captured. 01: the first rising edge starts, and all the subsequent rising edge and falling edge trigger width to be captured. 10: the first falling edge starts, and all the subsequent rising edge and falling edge trigger width to be captured. 11: the first rising edge starts, and all the subsequent rising edge trigger the pulse width to be captured.
9: 7 PSC0	**PWDT counter post-prescaler** 000 ~ 111: indicates 1/2/4/8/…/128
6 PWDTEN	**PWDT pulse width measurement mode enable** 0: disable 1: enable **Note: PWDTEN (pwdt function) enable prior to TIMEN (timer function), so when timer function is going to be enabled, PWDTEN must be disabled .**
5 IE	**PWDT interrupt enable** 0: disable 1: enable
4 PRDYIE	**PWDT pulse width data ready interrupt enable** 0: disable 1: enable
3 OVIE	**PWDT counter overflow interrupt enable** 0: disable 1: enable
1 RDYF	**PWDT pulse width data ready** 0: pwdt pulse width register is not up to date. 1: pwdt pulse width register has been updated, write 0 to clear.
0	**PWDT counter overflow**

Field	Description
OVF	
	0: no overflow
	1: pwdt counter overflow, write 0 to clear.

11.6.2 NPW count value(PWDT_NPW)

Table 11-6 PWDT_NPW register

PWDT_NPW **PWDT NPW count value** **Reset: 00000000**

Bit	31~16	15~0
Name	PWDTC	NPW
Type	RO	RO
Reset	0	0

Field	Description
31: 16 PWDTC	**Pulse Width Counter** Used to count for pulse width measurement or for timer.
15: 0 NPW	**Negative Pulse Width Count Value** Indicate the negative pulse width value.

11.6.3 Initialization Register 1(PWDT_INIT1)

Table 11-7 PWDT_INIT1 register

PWDT_INIT1 **PWDT Initialization Register 1** **Reset:00000000**

Bit	31	30	29	28	27~12	11	10	9	8	7~4	3~0
Name		HALLA	HALLB	HALLC	TIMLDVAL	CMPEN	TIMEN	HALLEN	FILTEN	FILTPSC	FILTVAL
Type		RO	RO	RO	RW	RW	RW	RW	RW	RW	RW
Reset		0	0	0	0	0	0	0	0	0	0

Field	Description
30: 28 **HALLA/B/C**	**HALLA/HALLB/HALLC status value** If 3 hall sensors installed spacing 60 electrical degree: 100 → 110 → 111 → 011 → 001 → 000 else 3 hall sensors installed spacing 120 electrical degree: 101 → 100 → 110 → 010 → 011 → 001
27: 12 **TIMLDVAL**	**Timer counter load value** Counter runs from 0x0000 to TIMLDVAL
11 **CMPEN**	**Comparator input enable**

Field	Description
	0: enable pwdt_in0 ~ pwdt_in2 derived from pad PWDT_IN0 ~ PWDT_IN2 externally. 1: enable pwdt_in0 ~ pwdt_in2 derived from acmp0_0 ~ acmp0_2 internally. **Note: When CMPEN=1, pwdt_in0 ~ pwdt_in2 are derived from acmp0_0 ~ acmp0_2 internally. And then acmp0_0 ~ acmp0_2 signals will be measured by the behavior of HALL sensor if HALLEN=1, otherwise just one of them will be measured based on the PINSEL .**
10 **TIMEN**	**Enable the timer function** 0: disable. 1: enable timer function. **Note: PWDTEN (pwdt pulse width measurement mode) enable is prior to TIMEN enable(timer function), so when timer function is going to be enabled, PWDTEN must be disabled .**
9 **HALLEN**	**Enable hall sensor signal detect function** 0: disable the hall sensor signal detect function. 1: enable the hall sensor signal detect function.
8 **FILTEN**	**Enable the pwdt input filter function** 0: disable. 1: enable the filter function.
7 ~ 4 **FILTPSC**	**Prescaler for filter** 1~ 12: respectively represent 2/4/8…/4096 divisor. 0, 13 ~ 15: not divide the filter clock.
3 ~ 0 **FILTVAL**	**Filter value** 0: disable filter 1 ~ 15 : for filter different width pulse.

12 TIMER

12.1 Introduction

The TIMER module is an array of timers that can be used to raise interrupts and triggers.

12.2 Features

- Ability of timers to generate interrupts.
- Ability of timers to generate trigger pulses.
- Independent timeout periods for each timer.
- Supports 4 * 32bit timer.
- Support Link mode.

12.3 Block diagram

The following is the block diagram of the TIMER module.

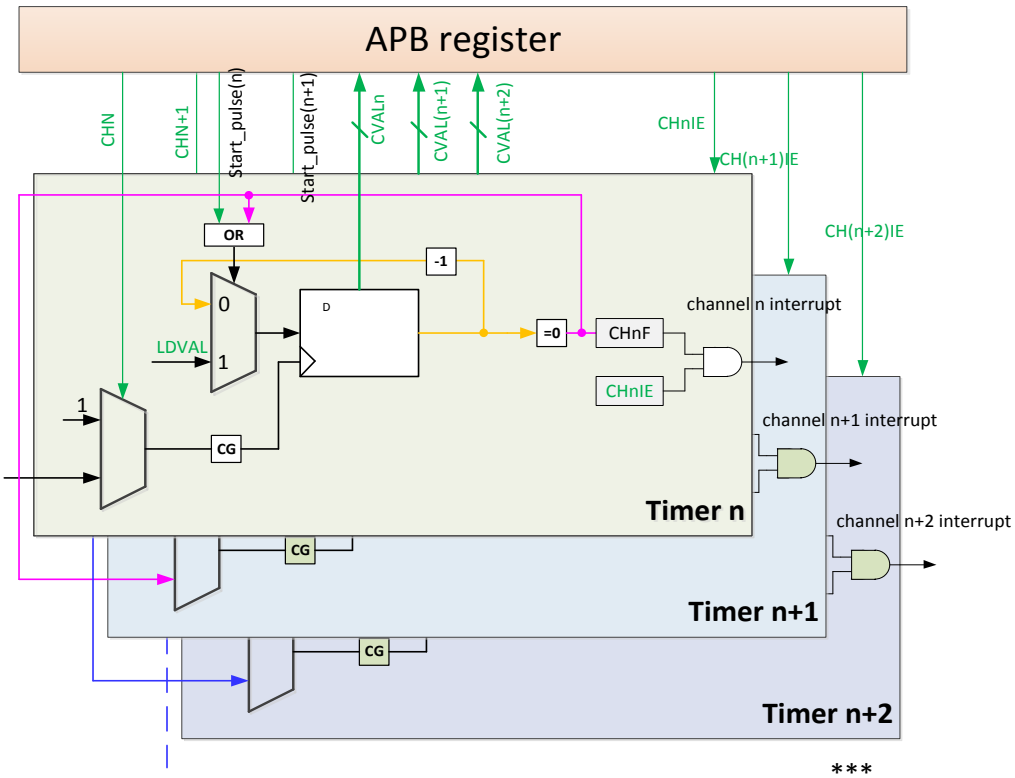

Figure 12-1 TIMER block diagram

12.4 Functional description

12.4.1 General mode

The timers generate triggers at periodic intervals, when enabled. The timers load the start values as specified in their TIMER_LDVAL registers, count down to 0 and then load the respective start value again. Each time a timer reaches 0, it will generate a trigger pulse and set the interrupt flag.

If desired, the current counter value of the timer can be read via the TIMER_CVAL registers. The counter period can be restarted, by first disabling, and then enabling the timer with TIMER_INIT[TEN].

12.4.2 Link mode

When a timer has link mode enabled, it will only count after the previous timer has expired. So if timer n-1 has counted down to 0, counter n will decrement the value by one. This allows linking some of the timers together to form a longer timer. The first timer (timer 0) cannot be linked to any other timer.

12.4.3 Interrupts

Timer interrupts can be enabled by setting TIE. TFLG flag is set to 1 when a timeout occurs on the associated timer, and is cleared to 0 by writing 1 to the corresponding TFLG.

When using the link function, generally only the timer n interrupt is enabled, and the timer interrupts linked to timer n are all off.

12.5 Register definition

Table 12-1 TIMER register map

TIMER base address = 0x40011000
TIMER_CH0 base address =0x40011100
TIMER_CH1 base address =0x40011110
TIMER_CH2 base address =0x40011120
TIMER_CH3 base address =0x40011130

Address	Register name	Width	Description
TIMER base address+0x00	TIMER_MCR	32	Module controller register
TIMER_CHx base address +0x00	TIMER_LDVAL	32	Initial value register
TIMER_CHx base address +0x04	TIMER_CVAL	32	Current value register
TIMER_CHx base address +0x08	TIMER_INIT	32	Initial register
TIMER_CHx base address +0x0C	TIMER_TF	32	Interrupt flag register

Note: In the above table, x=0~3.

12.5.1 Module Controller Register(TIMER_MCR)

Table 12-2 TIMER_MCR register

TIMER_MCR **TIMER module controller register** **Reset: 0x00000002**

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

Field	Description
1 MDIS	**Module Disable - (TIMER section)** 0: Timers module is enabled. 1: Timers module is disabled Disables the timer module. This field must be enabled before any other setup is done. **Note:** MDIS can pause and restart counting of 4 timers at the same time.

12.5.2 Initial Value Register(TIMER_LDVAL)

Table 12-3 TIMER_LDVAL register

TIMER_LDVAL **TIMER LDVAL register** **Reset: 0x00000000**

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

Field	Description
31: 0 LDVAL	**Load Value Register** Sets the timer start value. The timer will count down until it reaches 0, then it will generate an interrupt and load this register value again. Writing a new value to this register will not restart the timer. Instead the value will be loaded after the timer expires. To abort the current cycle and start a timer period with the new value, the timer must be disabled and enabled again. The conversion formula from TVAL to time is as follows:

Field	Description
	the timing period(Unit: second) = (TVAL + 1) / timing clock frequency(Unit: Hz).

12.5.3 Current Value Register(TIMER_CVAL)

Table 12-4 TIMER_CVAL register

TIMER_CVAL **TIMER CVAL register** **Reset: 0x00000000**

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

Field	Description
31: 0 CVAL	**Current Timer Value** It represents a real-time count value. Automatically reload the start value after counting down to 0.

12.5.4 Initial Register(TIMER_INIT)

Table 12-5 TIMER_INIT register

TIMER_INIT **TIMER INIT register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name														LINKEN	TIE	TEN
Type														RW	RW	RW
Reset														0	0	0

Field	Description
2 LINKEN	**Link Mode** 0: Timer is not linked. 1: Timer is linked to previous timer. For example, for Timer 2, if this field is set, Timer 2 is linked to Timer 1. When activated, Timer n-1 needs to expire before timer n can decrement by 1. Timer 0 cannot be linked.

Field	Description
1 TIE	**Timer Interrupt Enable** 0: Interrupt requests from Timer n are disabled. 1: Interrupt will be requested whenever TFLG is set. When an interrupt is pending, or, TFLG is set, enabling the interrupt will immediately cause an interrupt event. To avoid this, the associated TFLG must be cleared first.
0 TEN	**Timer Enable** 0: Timer n is disabled. 1: Timer n is enabled. Enables or disables the timer n.

12.5.5 Interrupt Flag Register(TIMER_TF)

Table 12-6 TIMER_TF register

TIMER_TF **TIMER interrupt flag register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name																TFLG
Type																R/W1C
Reset																0

Field	Description
0 TFLG	Timer timeout flag 0: Timeout has not yet occurred. 1: Timeout has occurred. Sets to 1 at the end of the timer period. Writing 1 to this flag clears it. Writing 0 has no effect. When TIE = 1, TFLG causes an interrupt request.

13 Connection Transmission Unit(CTU)

13.1 Introduction

The CTU module defines module-to-module interconnections and signal transmission between different modules on the chip.

13.2 Features

- ACMP output capture
- RTC output capture
- ADC hardware trigger
- PWM software synchronization

13.3 Block diagram

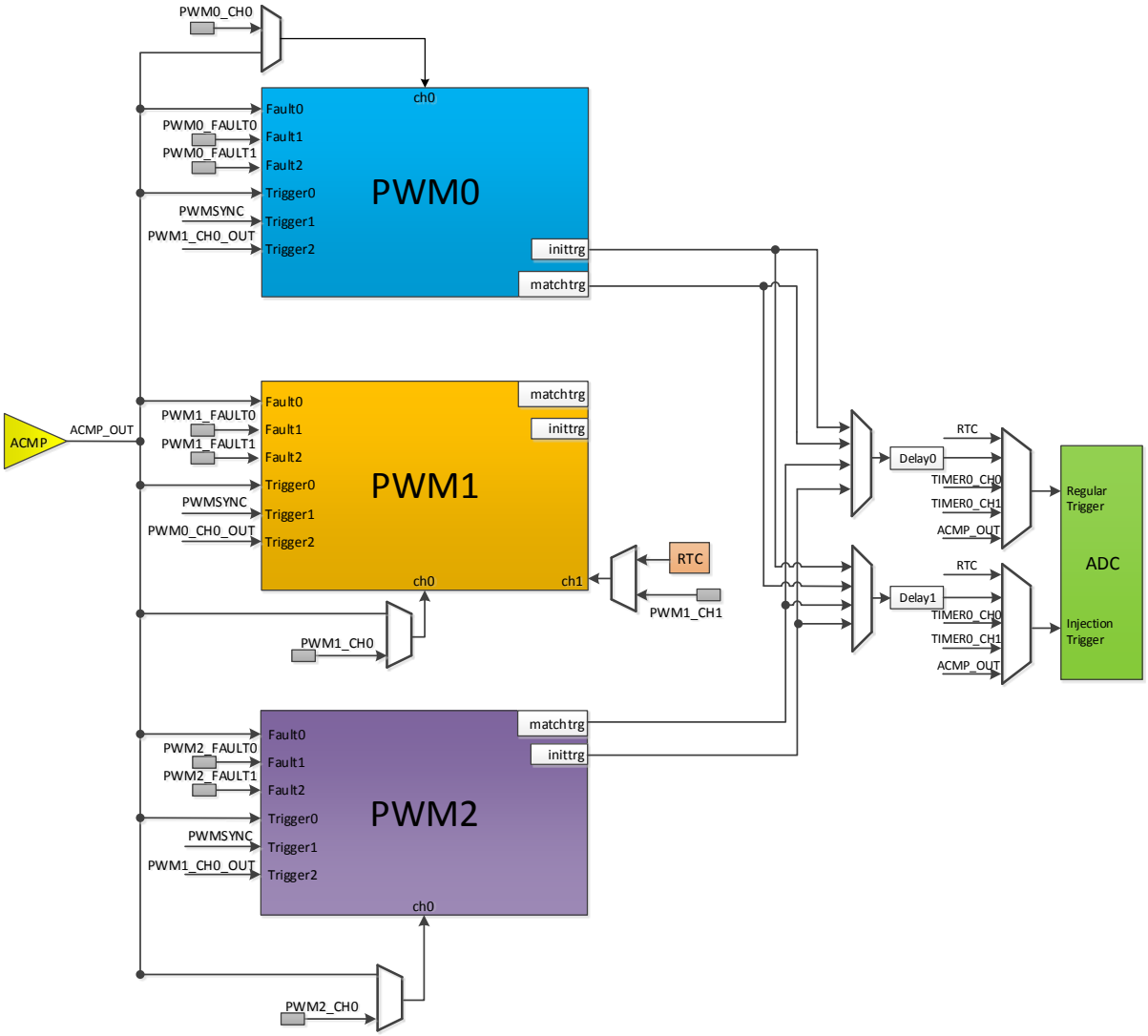

Figure 13-1 CTU block diagram

13.4 Function description

13.4.1 ACMP output capture

The CTU_CONFIG0[PWMx_ACIC] bit enables the output of ACMP0 to connect to the PWMx_CH0, the PWMx_CH0 pin is released to other multi-functions, refer to Table 14-2 GPIO multi-function, where x = 0/1/2. ACMP0 output is also connected to PWDT input (for Brushless Direct Current Motor using), or can be used as PWM trigger/fault input and ADC hardware trigger.

13.4.2 RTC capture

RTC overflow signal can be captured by PWM1_CH1 by setting CTU_CONFIG0[RTCC] bit. When this bit is set, the RTC overflow is connected to PWM1_CH1 for capture, the PWM1_CH1 pin is released to other multi-functions. Refer to Table 14-2 GPIO multi-function.

13.4.3 ADC hardware trigger

ADC module can be initialized to a conversion via a hardware trigger. The available ADC hardware regular trigger sources by setting CTU_CONFIG0[ADHWT0], ADC hardware injected trigger sources by setting CTU_CONFIG1[ADHWT1]. When ADC hardware trigger selects the output of PWM triggers, an 8-bit delay block will be enabled. This logic delays any trigger from PWM with an 8-bit counter whose value is specified by DELAY. The reference clock to this module is the bus clock with selectable pre-divider specified by CTU_CONFIG0[PSC].

13.4.4 PWM software synchronization

PWM contains three synchronization input triggers, one of which is a software trigger by writing 1 to CTU_CONFIG0[PWMSYNC]. Writing 0 to this field takes no effect. This field is always read 0.

13.4.5 Low power mode

Table 13-1 CTU low power mode

Mode	Wake-up source	Description
Sleep mode	---	The module works normally.
Stop mode	---	The module is disabled

13.5 Register definition

Table 13-2 CTU register map

CTU base address = 0x40016000

Address	Register name	Width	Description
CTU base address +0x00	CTU_CONFIG0	32	CTU Configuration register 0
CTU base address +0x04	CTU_CONFIG1	32	CTU Configuration register 1

13.5.1 CTU Configuration Register 0(CTU_CONFIG0)

Table 13-3 CTU_CONFIG0 register

CTU_CONFIG0	CTU configuration register 0	Reset: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DELAY0								DLYACT0	ADHWT0				PSC		
Type	RW								RO	RW				RW		
Reset	0								0	0				0		
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name		PWMSYNC		PWM0_ACIC	PWM1_ACIC	PWM2_ACIC	RTCC									
Type		RW		RW	RW	RW	RW									
Reset		0		0	0	0	0									

Field	Description
31: 24 DELAY0	**Regular Trigger Delay** Specifies the delay from PWM initial or match trigger to ADC hardware trigger. The 8-bit modulo value allows the delay from 0 to 255 upon the PSC settings. This is a one-shot counter that starts ticking when the trigger arrives and stops ticking when the counter value reaches the modulo value that is defined.
23 DLYACT0	**Trigger DELAY0 Active** 0: The delay is inactive. 1: The delay is active. This read-only field specifies the status if the PWM initial or match delay is active. This field is set when an PWM trigger arrives and the delay counter is ticking. Otherwise, this field will be cleard.
22: 20 ADHWT0	**ADC Regular Group Hardware Trigger Source** 000: RTC overflow as the ADC hardware trigger 001: PWM0 initial trigger with 8-bit programmable counter delay 010: PWM0 match trigger with 8-bit programmable counter delay 011: PWM2 initial trigger with 8-bit programmable counter delay 100: PWM2 match trigger with 8-bit programmable counter delay 101: TIMER channel0 overflow as the ADC hardware trigger 110 : TIMER channel1 overflow as the ADC hardware trigger 111 : ACMP0 out as the ADC hardware trigger. Selects the ADC hardware trigger source. All trigger sources start ADC conversion on rising-edge.
18: 16 PSC	**BUS Clock Output select** 000: Bus divided by 1 001: Bus divided by 2 010: Bus divided by 4

Field	Description
	011: Bus divided by 8 100: Bus divided by 16 101: Bus divided by 32 110: Bus divided by 64 111: Bus divided by 128 Enables bus clock output via an optional prescaler.
14 PWMSYNC	**PWM Synchronization Select** 0: No synchronization triggered. 1: Generates a PWM synchronization trigger to the PWM modules. Generates a PWM synchronization trigger TRIG1 to the PWM module if 1 is written to this field. Note that when set PWMTRIG = 1 to generate a trigger behavior, the PWMTRIG bit should be write to 0 manually.
12 PWM0_ACIC	**Analog comparator input capture enable** 0: ACMP0 output is not connected to the input channel 0 of the PWM0 module. 1: ACMP0 output is connected to the input channel 0 of the PWM0 module. Connect the ACMP0 output to the input channel 0 of the PWM0 module.
11 PWM1_ACIC	**Analog comparator input capture enable** 0: ACMP0 output is not connected to the input channel 0 of the PWM1 module. 1: ACMP0 output is connected to the input channel 0 of the PWM1 module. Connect the ACMP0 output to the input channel 0 of the PWM1 module.
10 PWM2_ACIC	**Analog comparator input capture enable** 0: ACMP0 output is not connected to the input channel 0 of the PWM2 module. 1: ACMP0 output is connected to the input channel 0 of the PWM2 module. Connect the ACMP0 output to the input channel 0 of the PWM2 module.
10 RTCC	**Real-Time Counter Capture** 0: RTC overflow is not connected to PWM1 input channel 1. 1: RTC overflow is connected to PWM1 input channel 1. Allows the Real-time Counter (RTC) overflow to be captured by PWM1 channel 1.

13.5.2 CTU Configuration Register 1(CTU_CONFIG1)

Table 13-4 CTU_CONFIG1 register

CTU_CONFIG1 **CTU Configuration 1 register** **Reset:0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name		DELAY1								DLYACT1						

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Type		RW								RW						
Reset		0								0						
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name								ADHWT1								
Type								RW								
Reset								0								

Field	Description
30:23 DELAY1	**Injected Trigger Delay** Specifies the delay from PWM initial or match trigger to ADC hardware trigger. The 8-bit modulo value allows the delay from 0 to 255 upon the PSC clock settings. This is a one-shot counter that starts ticking when the trigger arrives and stops ticking when the counter value reaches the modulo value that is defined.
22 DLYACT1	**Trigger DELAY1 Active** 0: The delay is inactive. 1: The delay is active. This read-only field specifies the status if the PWM initial or match delay is active. This field is set when an PWM trigger arrives and the delay counter is ticking. Otherwise, this field will be clear.
8:6 ADHWT1	**ADC Injected Group Hardware Trigger Source** 000: RTC overflow as the ADC hardware trigger 001: PWM0 initial trigger with 8-bit programmable counter delay 010: PWM0 match trigger with 8-bit programmable counter delay 011: PWM2 initial trigger with 8-bit programmable counter delay 100: PWM2 match trigger with 8-bit programmable counter delay 101: TIMER channel0 overflow as the ADC hardware trigger 110: TIMER channel1 overflow as the ADC hardware trigger 111: ACMP0 out as the ADC hardware trigger. Selects the ADC hardware trigger source. All trigger sources start ADC conversion on rising-edge.

14 General Purpose Input/Output(GPIO)

14.1 Introduction

The general-purpose input and output (GPIO) module is accessible via AHB for maximum pin performance.

When the pin is configured for the GPIO function, the Port Configuration register(GPIO_CR) controls the direction of each pin. The Port Output Data register(GPIO_ODR) controls output data of each pin. Also the GPIO output level is controlled by the Bit Set/Reset register(GPIO_BSRR), and the Bit Reset register(GPIO_BRR).

When the pins are configured for input functions, the GPIO input data register shows the high and low levels on each pin (1 for high level, 0 for low level).

Also, The MCU I/O pins are connected to onboard peripherals/modules through a multiplexer that allows only one peripheral's alternate function (AF) connected to an I/O pin at a time. In this way, there can be no conflict between peripherals sharing the same I/O pin. Each I/O pin has a multiplexer with alternate function that can be configured through the GPIO_PINMUX registers. When the function of a certain peripheral/module needs to be transferred from the current IO to another IO, in addition to the new IO that needs to connect the multiplexing function to the peripheral/module, the multiplexing configuration of the original IO also needs to be closed, otherwise it will cause Peripherals/modules work abnormally on new IO pins.

14.2 Features

GPIO pins support the following features:

- Up to 27 I/Os under control.
- Output states: push-pull or open drain (be related to I2C).
- Output data from output data register GPIO_ODR or peripheral (alternate function output).
- Driving capability selection for each I/O.
- Input states: floating, pull-up/down, analog (be related to ADC/ACMP).
- Input data to input data register GPIO_IDR or peripheral (alternate function input).
- Bit set and reset register GPIO_BSRR for bitwise write access to GPIO_ODR.
- Highly flexible pin multiplexing allows the use of I/O pins as GPIOs or as one of several peripheral functions.
- Configurable rising/falling/dual edge interrupt.
- Low power mode wakes up interrupt.

14.3 Block diagram

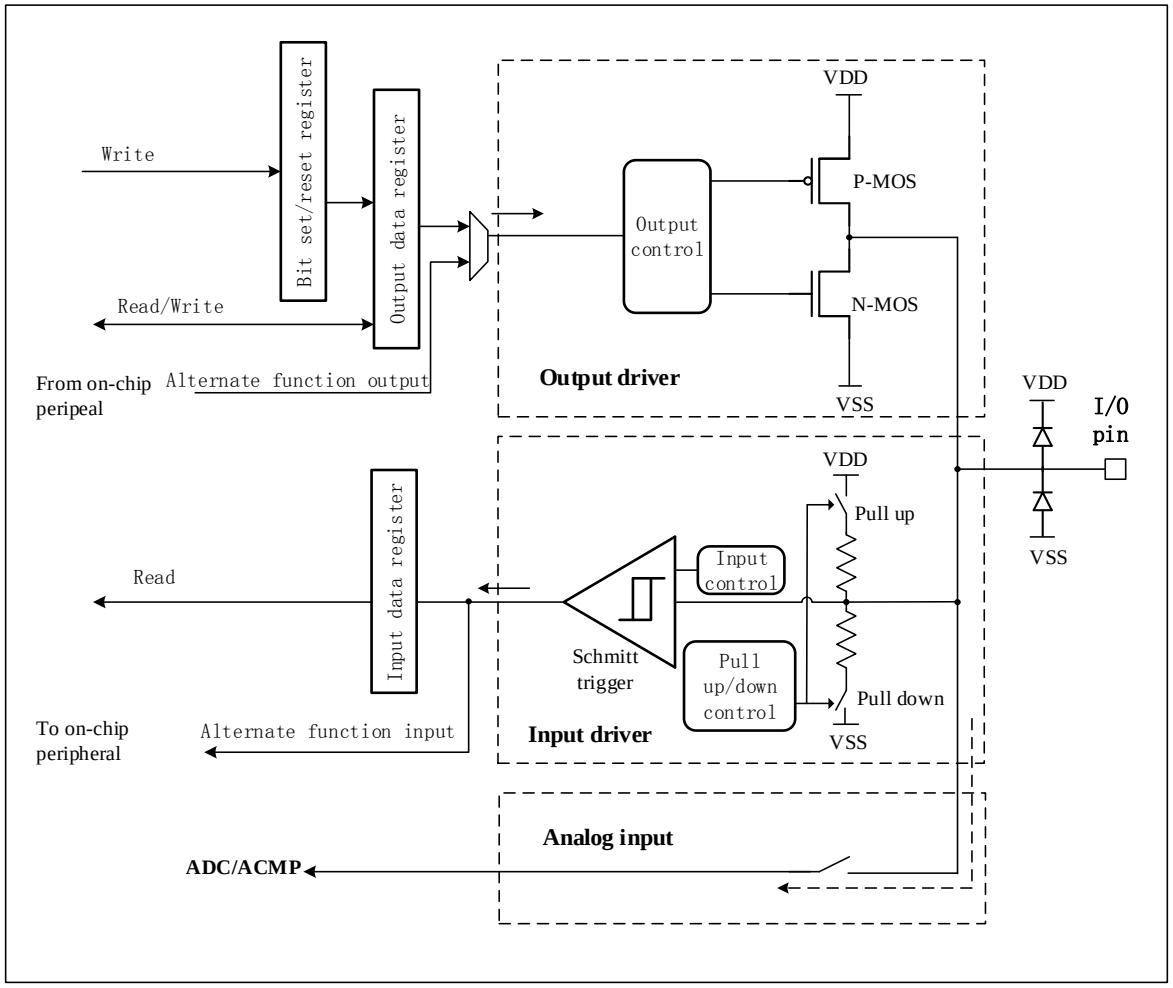

Figure 14-1 GPIO block diagram

14.4 Functional description

14.4.1 External interrupt

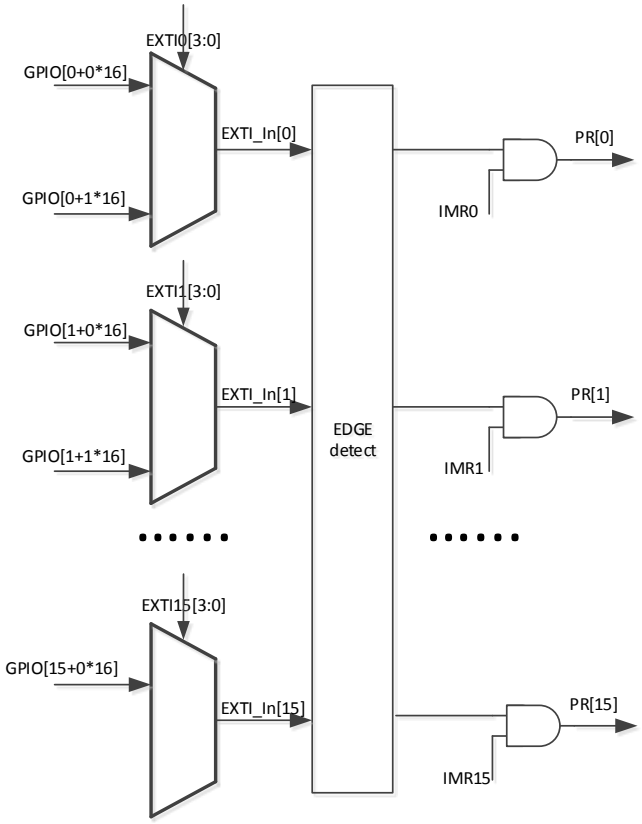

Figure 14-2 GPIO external interrupt

GPIO is divided into groups, and every 16 IOs form a group. Taking GPIO[y+x*16] as an example, 'y' represents the y-th PIN in a certain group of IOs, and x*16 represents the x-th group of GPIOs. For example, GPIO[1+0*16] represents GPIOA_PIN_1, GPIO[15+1*16] represents GPIOB_PIN_15.

The correspondence relationship between external interrupt line and interrupt vector:

- When m≤2, EXTI_In[m] corresponds to the interrupt vector EXTIm_IRQn
- When 3≤m≤8, EXTI_In[m] corresponds to the interrupt vector EXTI3_8_IRQn
- When 9≤m≤15, EXTI_In[m] corresponds to the interrupt vector EXTI9_15_IRQn

The corresponding relationship between GPIO external interrupts and ISR is shown below.

Table 14-1 Corresponding relationship between GPIO external interrupt and ISR

GPIO pin	Interrupt flag	ISR
PA0~PB0	EXTI0	EXTI0_IRQHandler
PA1~PB1	EXTI1	EXTI1_IRQHandler
PA2~PB2	EXTI2	EXTI2_IRQHandler
PA3~PB3	EXTI3	EXTI3_8_IRQHandler
PA4~PB4	EXTI4	
PA5~PB5	EXTI5	
PA6~PB6	EXTI6	
PA7~PB7	EXTI7	
PA8~PB8	EXTI8	
PA9~PB9	EXTI9	EXTI9_15_IRQHandler
PA10~PB10	EXTI10	
PA11	EXTI11	
PA12	EXTI12	
PA13	EXTI13	
PA14	EXTI14	
PA15	EXTI15	

14.4.2 Multi-Function

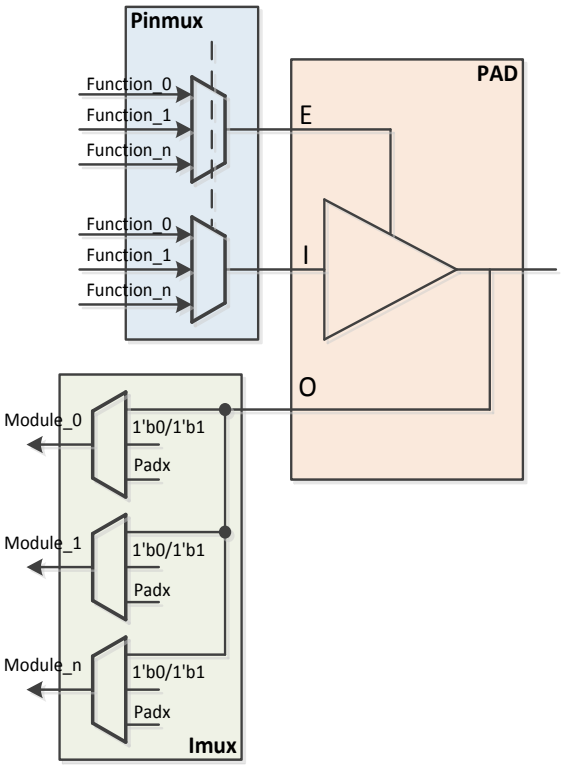

Figure 14-3 GPIO multi-function

Each IO have multi-functions, if we want to begin a peripheral communication, we should configure GPIO multi-function first. The corresponding multi-function of each GPIO is as below.

Table 14-2 GPIO multi-function

32 Pin	20 Pin	Pin name	Function0	Function1	Function2	Function3	PINMUX	GPIO (NUM)
1		PB0	GPIO	ADC_IN18	PWM1_CH1	PWM2_FLT1	PINMUX1[20:18]	16
2		PB1	GPIO		PWM1_CH0	DAC_OUT	PINMUX1[23:21]	17
3	4	VDD1	VDD1					
4	5	VSS1	VSS1					
5	6	PA12	GPIO	I2C0_SCL	OSC_OUT[1]	PWM0_FLT0	PINMUX1[8:6]	12
6	7	PA15	GPIO	I2C0_SDA	OSC_IN[1]	PWDT0_IN0	PINMUX1[17:15]	15
7	8	PA0	GPIO	PWM2_CH3	VREF-/ADC_IN10	I2C0_SCL	PINMUX0[2:0]	0
8	9	PA1	GPIO	PWM2_CH2	VREF+/ADC_IN9	I2C0_SDA	PINMUX0[5:3]	1
9		PB3	GPIO	PWM2_CH0	ADC_IN13	SPI0_MOSI	PINMUX1[29:27]	19
10	10	PA2	GPIO	PWM2_CH1	ADC_IN8	SPI0_MISO	PINMUX0[8:6]	2
11	11	PA3	GPIO	PWM2_CH0	ADC_IN7	SPI0_SCK	PINMUX0[11:9]	3
12	12	PA4	GPIO	PWM0_CH1	ADC_IN6/ACMP_IN6	UART1_TX	PINMUX0[14:12]	4
13	13	PA5	GPIO	PWM0_CH0	ADC_IN5/ACMP_IN5	UART1_RX	PINMUX0[17:15]	5

32 Pin	20 Pin	Pin name	Function0	Function1	Function2	Function3	PINMUX	GPIO (NUM)
14	14	PA6	GPIO	BOOT[1]	GPIO	SPI0_NSS	PINMUX0[20:18]	6
15		PB4	GPIO	PWM2_CH1	ADC_IN12	SPI0_MISO	PINMUX2[2:0]	20
16		PB5	GPIO	PWM0_CH0	ADC_IN11	SPI0_SCK	PINMUX2[5:3]	21
17	15	PA7	GPIO	UART0_TX	ADC_IN4/ACMP_IN4	SPI0_MOSI	PINMUX0[23:21]	7
18	16	PA8	GPIO	UART0_RX	ADC_IN3/ACMP_IN3	SPI0_NSS	PINMUX0[26:24]	8
19	17	PA9	GPIO	PWM2_FLT0	ADC_IN2/ACMP_IN2	RTC_CLKIN	PINMUX0[29:27]	9
20		VSS2	VSS2					
21	18	VDD2	VDD2					
22		PB6	GPIO	ADC_IN16	PWM1_FLT0	PWM0_FLT1	PINMUX2[8:6]	22
23		PB7	GPIO	ADC_IN15	ACMP_IN7	I2C0_SCL	PINMUX2[11:9]	23
24		PB8	GPIO	ADC_IN14	PWDT0_IN2	I2C0_SDA	PINMUX2[14:12]	24
25	19	PA10	GPIO	PWM1_CH1	ADC_IN1/ACMP_IN1	PWDT0_IN2	PINMUX1[2:0]	10
26	20	PA11	GPIO	PWM1_CH0	ADC_IN0/ACMP_IN0	PWDT0_IN1	PINMUX1[5:3]	11
27		PB9	GPIO	PWM2_CH3	I2C0_SCL	UART0_TX/UARTLIN_TX	PINMUX2[17:15]	25
28		PB10	GPIO	PWM2_CH2	I2C0_SDA	UART0_RX/UARTLIN_RX	PINMUX2[20:18]	26
29		PB2	GPIO	NMI_B	PWM1_FLT1	PWDT0_IN0	PINMUX1[26:24]	18
30	1	PA13	GPIO	SWD_CLK[1]		RTC_CLKOUT	PINMUX1[11:9]	13
31	2	RESET_B	RESET_B					
32	3	PA14	GPIO	SWD_DIO[1]	ACMP_OUT	PWM0_CH1	PINMUX1[14:12]	14

Note:

1. **All the pins are default as function0 on the first time power on except some dedicated pins (marked as 1 in the above table). In the GPIO control register, except GPIO_PINMUX, others respectively represent PA, PB, such as GPIO_CR, GPIO_IDR, etc., x=0,1, the lower 16 bits of each register respectively represent Px0~Px15.**
2. **Input/output configuration example: set PB1 as output mode, GPIO1_CR[1] = 1.**
3. **Multi-function configuration (take 32Pin as example): if we want to configure PB0 as PWM0_CH1, we should set PMUX1[20:18] to 2.**
4. **The suffix _B of NMI_B, RESET_B represents low level effective.**

14.4.3 Low power mode

When the MCU is in Stop mode, any IO can generate an interrupt and wake up the MCU.

14.5 Application note

14.5.1 External input

The external interrupt/event controller consists of up to 16 edge detectors for generating event/interrupt requests. Each input line can be independently configured to select the type (event or interrupt) and the corresponding trigger event (rising edge, falling edge or both). Each line can also be masked independently. A pending register GPIO_PR maintains the status line of the interrupt requests.

14.5.2 Multi-Function

To optimize functionality in small packages, pins have several functions available via signal multiplexing. The GPIO_PINMUX Register control which signal is present on the external pin. Refer to that register to find the detailed control operation of a specific multiplexed pin. For details, please refer to the Multiplexing Function Selection Register GPIO_PINMUX and the chapter 14.4.2 Multi-Function.

14.5.3 Open-drain output

GPIO PIN is not separately configured as an open-drain output register, but it can be configured through a combination of registers to achieve the function of open-drain output.

Analog open-drain output high (external pull-up resistor is required):

1. Configure the corresponding BIT of the GPIO_CR register of the PIN to 0, input mode;
2. Configure the corresponding BIT of the GPIO_ODR register to 0;
3. Configure the corresponding BIT of the GPIO_PU and GPIO_PD register to 0, no pull-up and pull-down.

Analog open-drain output low:

1. Configure the corresponding BIT of the GPIO_CR register of the PIN to 1, output mode;
2. Configure the corresponding BIT of the GPIO_ODR register to 0;
3. Configure the corresponding BIT of the GPIO_PU and GPIO_PD register to 0, no pull-up and pull-down.

14.5.4 HIGH_Z mode

The flow of configuring GPIO PIN as HIGH_Z mode is as follows:

1. Configure the corresponding BIT of the GPIO_PU and GPIO_PD register to 0, no pull-up and pull-down.

2. Configure the corresponding BIT of the GPIO_CR register of the PIN to 0, disable the output function of this PIN;
3. Configure the corresponding BIT of the GPIO_IES register of the PIN to 0, disable the input function of this PIN.

14.5.5 GPIO function

During and just after reset, the GPIO functions are active and the I/O ports are configured in input mode without pull-up or pull-down, except RST and Arm debug interface. Before entering Stop mode, if the I/O is set to input state without pull-up or pull-down, you need to set a high or low stable level externally to avoid power leakage caused by unstable I/O level.

User can program GPIO_PINMUX Register to change I/O ports from other function to GPIO function.

When the pin is configured as output, the value written to the output data register is output on the I/O pin. It is possible to use the output driver in push-pull mode or open-drain mode (only the N-MOS is activated when 0 is output).The input data register (GPIO_IDR) captures the data present on the I/O pin at every AHB clock cycle.

All GPIO pins have weak internal pull-up and pull-down resistors, which can be activated or not depending on the value in the GPIO_PU and GPIO_PD register.

14.5.6 Programming guide

Firstly, after reset, all I/O ports are in GPIO input mode, except RST and Arm debug interface.

Secondly, Software can program GPIO_PINMUXto map/remap I/O function.

As I/O in GPIO function, Software can program external interrupt. When MCU in low power mode, external interrupt uses internal 32K LPOSC.

14.6 Register description

Table 14-3 GPIO register map

GPIOA base address: 0x20080000
GPIOB base address: 0x20080030

Address	Register name	Width	Description
GPIOx base address + 0x00	GPIO_CR	32	Port configuration
GPIOx base address + 0x04	GPIO_IDR	32	Port input data
GPIOx base address + 0x08	GPIO_ODR	32	Port output data
GPIOx base address + 0x0C	GPIO_BSRR	32	Port set/reset
GPIOx base address + 0x10	GPIO_BRR	32	Port reset
GPIOx base address + 0x18	GPIO_PD	32	Pull-down enable
GPIOx base address + 0x1C	GPIO_PU	32	Pull-up enable

Address	Register name	Width	Description
GPIOx base address + 0x20	GPIO_E4_E2	32	Driving Capability selection
GPIOx base address + 0x24	GPIO_IES	32	Input enable
GPIOA base address + 0x140 GPIOA base address + 0x144 GPIOA base address + 0x148	GPIO_PINMUX	32	Multi-function selection: total 3 registers
GPIOA base address + 0x160	GPIO_PR	32	External interrupt flag pending
GPIOA base address + 0x164	GPIO_IMR	32	Interrupt mask
GPIOA base address + 0x168	GPIO_RTSR	32	Rising edge trigger event configuration
GPIOA base address + 0x16C	GPIO_FTSR	32	Falling edge trigger event configuration
GPIOA base address + 0x170 GPIOA base address + 0x174 GPIOA base address + 0x178 GPIOA base address + 0x17C	GPIO_EXTICR	32	External interrupt: total 4 registers

Note: in the above table, x=A, B.

14.6.1 Port Configuration Register(GPIO_CR)

Table 14-4 GPIO_CR register

GPIO_CR **Port configuration register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	MODE(16*x+15)	MODE(16*x+14)	MODE(16*x+13)	MODE(16*x+12)	MODE(16*x+11)	MODE(16*x+10)	MODE(16*x+9)	MODE(16*x+8)	MODE(16*x+7)	MODE(16*x+6)	MODE(16*x+5)	MODE(16*x+4)	MODE(16*x+3)	MODE(16*x+2)	MODE(16*x+1)	MODE(16*x+0)
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Field	Description
15: 0 MODE	**Mode(y):** Port y configuration bits These bits are written by software to configure the I/O direction mode. 0: Input (reset state) 1: output mode

14.6.2 Port Input Data Register(GPIO_IDR)

Table 14-5 GPIO_IDR register

GPIO_IDR **Port input data register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	IDR(16*x+15)	IDR(16*x+14)	IDR(16*x+13)	IDR(16*x+12)	IDR(16*x+11)	IDR(16*x+10)	IDR(16*x+9)	IDR(16*x+8)	IDR(16*x+7)	IDR(16*x+6)	IDR(16*x+5)	IDR(16*x+4)	IDR(16*x+3)	IDR(16*x+2)	IDR(16*x+1)	IDR(16*x+0)
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Field	Description
15: 0 IDR	**IDR(y): Port y input data** These bits are read-only. They contain the input value of the corresponding I/O port.

14.6.3 Port Output Data Register(GPIO_ODR)

Table 14-6 GPIO_ODR register

GPIO_ODR **Port output data register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	ODR(16*x+15)	ODR(16*x+14)	ODR(16*x+13)	ODR(16*x+12)	ODR(16*x+11)	ODR(16*x+10)	ODR(16*x+9)	ODR(16*x+8)	ODR(16*x+7)	ODR(16*x+6)	ODR(16*x+5)	ODR(16*x+4)	ODR(16*x+3)	ODR(16*x+2)	ODR(16*x+1)	ODR(16*x+0)
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Field	Description
15: 0 ODR	**ODR(y): Port(y) output data** These bits can be read and written by software. **Note: For PIN set/reset, the ODR bits can be individually set and reset by writing to the GPIOx_BSRR register (x = A, B) and GPIOx_BRR register (x = A, B).**

14.6.4 Port Set/Reset Register (GPIO_BSRR)

Table 14-7 GPIO_BSRR register

GPIO_BSRR **Port set/reset register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	BR(16*x+15)	BR(16*x+14)	BR(16*x+13)	BR(16*x+12)	BR(16*x+11)	BR(16*x+10)	BR(16*x+9)	BR(16*x+8)	BR(16*x+7)	BR(16*x+6)	BR(16*x+5)	BR(16*x+4)	BR(16*x+3)	BR(16*x+2)	BR(16*x+1)	BR(16*x+0)
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	BS(16*x+15)	BS(16*x+14)	BS(16*x+13)	BS(16*x+12)	BS(16*x+11)	BS(16*x+10)	BS(16*x+9)	BS(16*x+8)	BS(16*x+7)	BS(16*x+6)	BS(16*x+5)	BS(16*x+4)	BS(16*x+3)	BS(16*x+2)	BS(16*x+1)	BS(16*x+0)
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Field	Description
31:16 BR	**BR(y): Port(y) Reset bit y** 0: No action on the corresponding ODRy bit 1: Reset the corresponding ODRy bit These bits are write-only. A read to these bits returns the value 0x0000.
15: 0 BS	**BS(y): Port(y) set bit y** 0: No action on the corresponding ODRy bit 1: Sets the corresponding ODRy bit These bits are write-only and can be accessed in word, half-word or byte mode. A read to these bits returns the value 0x0000. **Note: if BS and BR both configured, BS has higher priority**.

14.6.5 Port reset register(GPIO_BRR)

Table 14-8 GPIO_BRR register

GPIO_BRR **Port reset register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	BR(16*x+15)	BR(16*x+14)	BR(16*x+13)	BR(16*x+12)	BR(16*x+11)	BR(16*x+10)	BR(16*x+9)	BR(16*x+8)	BR(16*x+7)	BR(16*x+6)	BR(16*x+5)	BR(16*x+4)	BR(16*x+3)	BR(16*x+2)	BR(16*x+1)	BR(16*x+0)
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Field	Description
15: 0 BR	**BR(y): Port(y) Reset bit y** 0: No action on the corresponding ODRy bit 1: Reset the corresponding ODRy bit These bits are write-only. A read to these bits returns the value 0x0000.

14.6.6 Pull-down Enable Register(GPIO_PD)

Table 14-9 GPIO_PD register

GPIO_PD **Pull-down enable register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	PD(16*x+15)	PD(16*x+14)	PD(16*x+13)	PD(16*x+12)	PD(16*x+11)	PD(16*x+10)	PD(16*x+9)	PD(16*x+8)	PD(16*x+7)	PD(16*x+6)	PD(16*x+5)	PD(16*x+4)	PD(16*x+3)	PD(16*x+2)	PD(16*x+1)	PD(16*x+0)
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Field	Description
15:0 PD	**PD (y): Pull-down enable** 0: disable pull-down 1: enable pull-down These bits can be read and written by software. **Note: Pull-up and Pull-down are not supported enable at the same time.**

14.6.7 Pull-Up Enable Register(GPIO_PU)

Table 14-10 GPIO_PU register

GPIO_PU **Pull-Up enable register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	PU(16*x+15)	PU(16*x+14)	PU(16*x+13)	PU(16*x+12)	PU(16*x+11)	PU(16*x+10)	PU(16*x+9)	PU(16*x+8)	PU(16*x+7)	PU(16*x+6)	PU(16*x+5)	PU(16*x+4)	PU(16*x+3)	PU(16*x+2)	PU(16*x+1)	PU(16*x+0)
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Field	Description
15:0 PU	**PU (y): Pull-up enable** 0: disable pull-up 1: enable pull-up These bits can be read and written by software. **Note: Pull-up and Pull-down are not supported enable at the same time.**

14.6.8 Driving Capability Selection Register(GPIO_E4_E2)

Table 14-11 GPIO_E4_E2 register

GPIO_E4_E2 **Driving capability selection register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	E4_E2(16*x+15)		E4_E2(16*x+14)		E4_E2(16*x+13)		E4_E2(16*x+12)		E4_E2(16*x+11)		E4_E2(16*x+10)		E4_E2(16*x+9)		E4_E2(16*x+8)	
Type	RW		RW		RW		RW		RW		RW		RW		RW	
Reset	0		0		0		0		0		0		0		0	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	E4_E2(16*x+7)		E4_E2(16*x+6)		E4_E2(16*x+5)		E4_E2(16*x+4)		E4_E2(16*x+3)		E4_E2(16*x+2)		E4_E2(16*x+1)		E4_E2(16*x+0)	
Type	RW		RW		RW		RW		RW		RW		RW		RW	
Reset	0		0		0		0		0		0		0		0	

Field	Description
1: 0 3: 2 5: 4 …… E4_E2	**E4_E2 (y): Driving capability selection** 00: 4mA 01: 8mA 10: 12mA 11: 16mA

Field	Description
	These bits can be read and written by software.

14.6.9 Input Enable Register(GPIO_IES)

Table 14-12 GPIO_IES register

GPIO_IES **Input enable register** **Reset：0x0000FFFF**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	IES(16*x+15)	IES (16*x+14)	IES (16*x+13)	IES (16*x+12)	IES (16*x+11)	IES (16*x+10)	IES (16*x+9)	IES (16*x+8)	IES (16*x+7)	IES (16*x+6)	IES (16*x+5)	IES (16*x+4)	IES (16*x+3)	IES (16*x+2)	IES (16*x+1)	IES (16*x+0)
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Field	Description
15:0 IES	**IES (y): input enable** 0: disable input 1: enable input These bits can be read and written by software. **Note：bit11 of the GPIOB IES register needs to be remained as "1", clearing it will cause a system exception.**

14.6.10 Multi-function Selection Register(GPIO_PINMUX)

Table 14-13 GPIO_PINMUX register

GPIO_PINMUX **Multi-function selection register** **Reset: 0x00000000**

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

Field	Description
2: 0 5: 3 8: 6 …… PINMUXx	**PINMUXx (y): Multi-function selection** 000: function 0 001: function 1 010: function 2 011: function 3 These bits are written by software to configure alternate function I/Os Note: GPIO_PINMUX0 register default value is 0x00040000; GPIO_PINMUX1 register default value is 0x00011280; GPIO_PINMUX2 register default value is 0x00000000.

14.6.11 External interrupt flag pending(GPIO_PR)

Table 14-14 GPIO_PR register

GPIO_PR **External interrupt flag pending** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	PR(15)	PR(14)	PR(13)	PR(12)	PR(11)	PR(10)	PR(9)	PR(8)	PR(7)	PR(6)	PR(5)	PR(4)	PR(3)	PR(2)	PR(1)	PR(0)
Type	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Field	Description
15:0 PR	**PR (y): External interrupt flag pending bit** 0: No trigger request occurred 1: The selected trigger request occurred This bit is set when the selected edge event arrives on the external interrupt line. This bit is cleared by writing a '1' to the bit.

14.6.12 Interrupt mask register(GPIO_IMR)

Table 14-15 GPIO_IMR register

GPIO_IMR **Interrupt mask register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	IMR(15)	IMR(14)	IMR(13)	IMR(12)	IMR(11)	IMR(10)	IMR(9)	IMR(8)	IMR(7)	IMR(6)	IMR(5)	IMR(4)	IMR(3)	IMR(2)	IMR(1)	IMR(0)
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Field	Description
15:0 IMR	**IMR (y): Interrupt mask of line y** 0: Interrupt request from Line y is masked 1: Interrupt request from Line y is not masked

14.6.13 Rising Edge Trigger Event Configuration(GPIO_RTSR)

Table 14-16 GPIO_RTSR register

GPIO_RTSR **Rising edge trigger event configuration** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RTSR(15)	RTSR(14)	RTSR(13)	RTSR(12)	RTSR(11)	RTSR(10)	RTSR(9)	RTSR(8)	RTSR(7)	RTSR(6)	RTSR(5)	RTSR(4)	RTSR(3)	RTSR(2)	RTSR(1)	RTSR(0)
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Field	Description
15:0 RTSR	**RTSR (y): Rising edge trigger event configuration bit of line y** 0: Rising edge trigger disabled (for Event and Interrupt) for input line y 1: Rising edge trigger enabled (for Event and Interrupt) for input line y

14.6.14 Falling Edge Trigger Event Configuration(GPIO_FTSR)

Table 14-17 GPIO_FTSR register

GPIO_FTSR Falling edge trigger event configuration register **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	FTSR(15)	FTSR(14)	FTSR(13)	FTSR(12)	FTSR(11)	FTSR(10)	FTSR(9)	FTSR(8)	FTSR(7)	FTSR(6)	FTSR(5)	FTSR(4)	FTSR(3)	FTSR(2)	FTSR(1)	FTSR(0)
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Field	Description
15:0 FTSR	**FTSR (y): Falling edge trigger event configuration bit of line y** 0: Falling edge trigger disabled (for Event and Interrupt) for input line y 1: Falling edge trigger enabled (for Event and Interrupt) for input line y

14.6.15 External interrupt register(GPIO_EXTICR)

Table 14-18 GPIO_EXTICR register

GPIO_EXTICR External interrupt register **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	EXTI(4*x+3)				EXTI(4*x+2)				EXTI(4*x+1)				EXTI(4*x+0)			
Type	RW				RW				RW				RW			
Reset	0				0				0				0			

Field	Description
3: 0 7: 4 11: 8 …… EXTI	**EXTI (y): EXTI y configuration** 0000: PA[x] pin 0001: PB[x] pin These bits are written by software to select the source input for the EXTI(y) external interrupt.

15 Inter-IC(I2C)

15.1 Introduction

I2C(Inter-IC) is a simple, bi-directional synchronous two-wire serial interface. Data transmission is done by SCL and SDA. SCL is a clock signal that is driven by the master. SDA is bi-directional data signal that can be driven by either the master or the slave.

15.2 Features

- Supports master and slave mode operation.
- Supports I2C standard mode 100kHz, fast mode 400kHz and fast mode plus mode 1MHz.
- Supports 7-bit address range.
- Slave 10-bit address extension.
- Supports slave stretch.
- Supports slave low power mode wakeup.
- Supports slave monitor function.
- Supports multi-master arbitration.
- Master switch to slave mode automatically while arbitration lost.
- Programmable input glitch filter.
- Bus START/STOP signal detection.
- Software-controlled acknowledge bit.
- Address match interrupt.
- Byte transfer interrupt.
- No ack interrupt.
- Transmit underflow interrupt and receive overflow interrupt
- SDA/SCL low timeout interrupt.
- Supports clock synchronization.
- Supports SMBus 2.0 – Alert Response Address.
- Supports SMBus 2.0 – SDA/SCL low timeout detection.

15.3 Block diagram

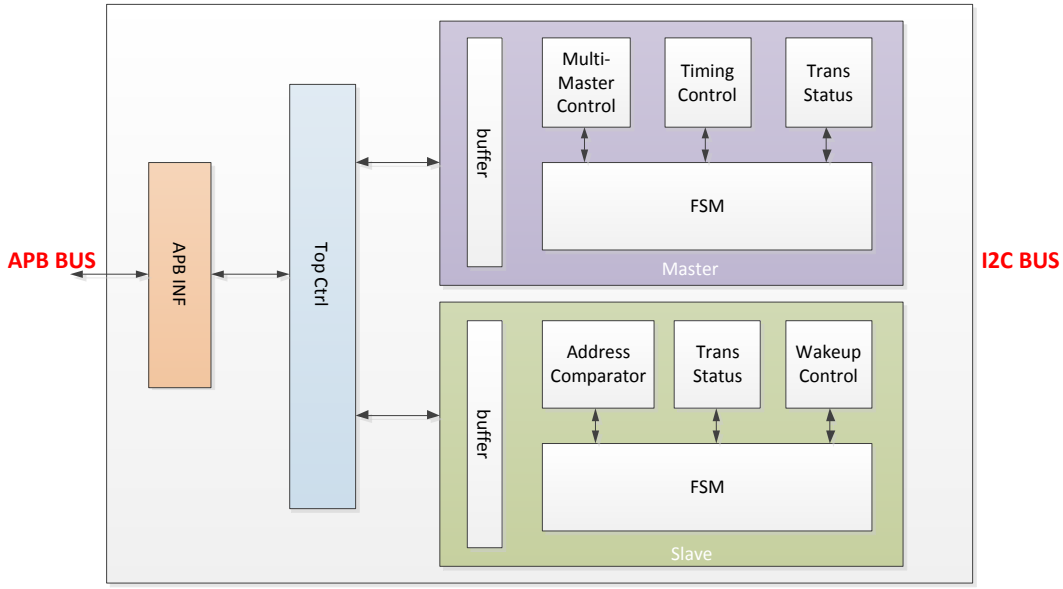

Figure 15-1 I2C block diagram

15.3.1 I2C signal

All transactions begin with a START (S) and terminated by a STOP (P). A high to low transition on the SDA line while SCL is high defines a START condition. A low to high transition on the SDA line while SCL is high defines a STOP condition (see Figure 15-2).

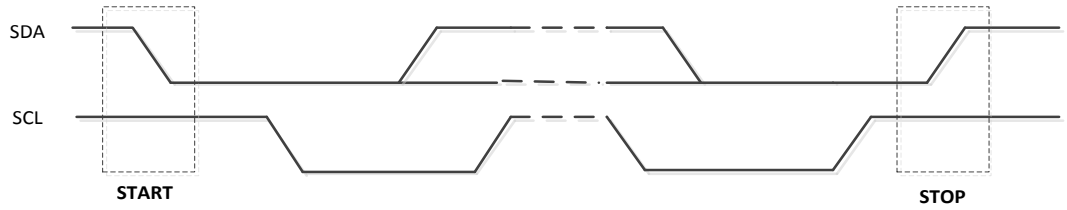

Figure 15-2 START and STOP conditions

Each frame of data consists of 9 bits, 8 bits of data (MSB first) and 1 bit of ack signal, and the transmission times are not limited. (see Figure 15-3).

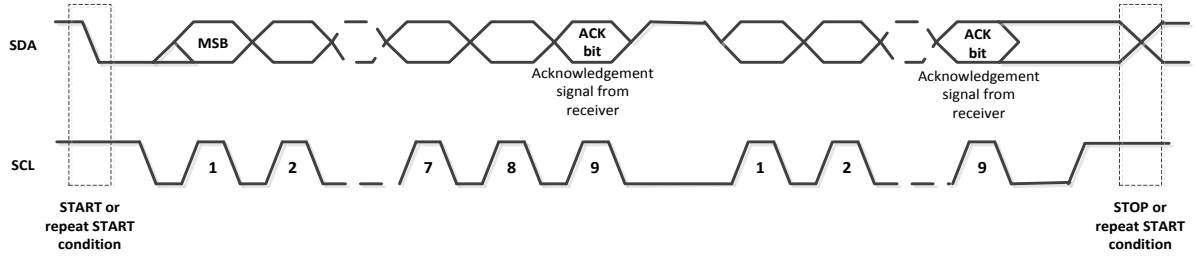

Figure 15-3 Data transmission format

The first data after the start signal is the address byte, and the data that continues to be transmitted after the address byte is the data byte.

In the 7-bit address mode, the I2C protocol stipulates that the high 7 bits of the address byte are the address of the slave. If the lowest bit is 0, it means to write to the slave, and if the lowest bit is 1, it means to read from the slave.

In the 10-bit address mode, the address is composed of two bytes.

When writing to the slave, the I2C protocol stipulates that: the first byte is sent to 11110XX0, the high five bits are fixed to 11110, bit2, bit1 is the high two bits in the 10-bit address, bit0 is the direction bit, and the value is 0, indicating that the direction is write; The second byte is the lower 8 bits of the 10-bit address.

When reading from the slave, I2C stipulates that the address to be written (two bytes) should be sent first, and then the address to be read (the address to be read only needs to send one byte). The specific process is as follows: for the address to be written first, the first byte is sent to 11110XX0, the high five bits are fixed to 11110, bit2 and bit1 are the high two bits of the 10-bit address, bit0 is the direction bit, and the value is 0, indicating that the direction is write, and the second byte is the low eight bits of the 10-bit address. For the address to be read again, 11110XX1 is sent. bit2 and bit1 are the upper two bits of the 10-bit address. bit0 is the direction bit, and the value is 1, indicating that the direction is read.

15.3.2 Baud rate component

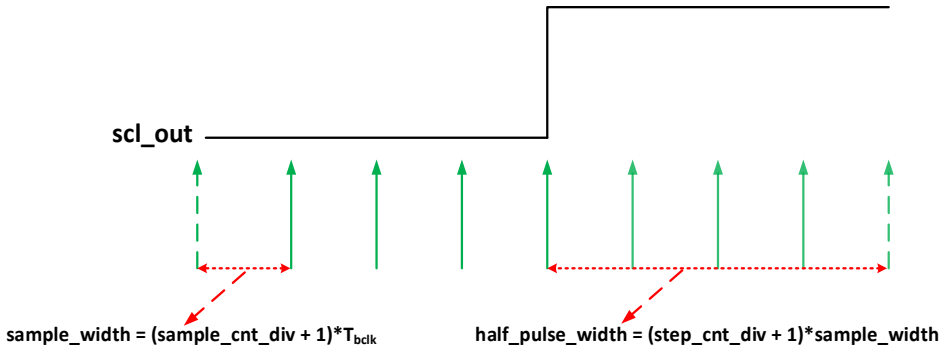

Figure 15-4 Baud rate generation

Baud rate: $f_{SCL}=f_{bclk}/(((SAMPLE\_CNT\_DIV+1) * (STEP\_CNT\_DIV+1))*2)$
f_{bclk} is the APB bus clock frequency.

15.3.3 Data flow

For transmitter, data is written to an 8-bit TX buffer, then load to TX shift refer to baud rate control logic. The output data is most significant bit(MSB) first. The transmitter sample the ack bit every 9th SCL per byte.

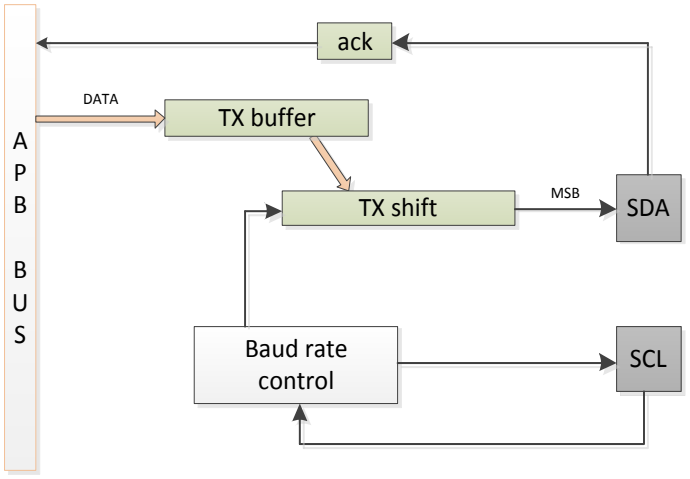

Figure 15-5 Data flow of transmitter

For receiver, RX shift sample and shift in the SDA line input data. Data received store into an 8-bit RX buffer after every 8th SCL per byte with the most significant bit(MSB) first. The ack bit is put on the SDA line at 9th SCL pulse.

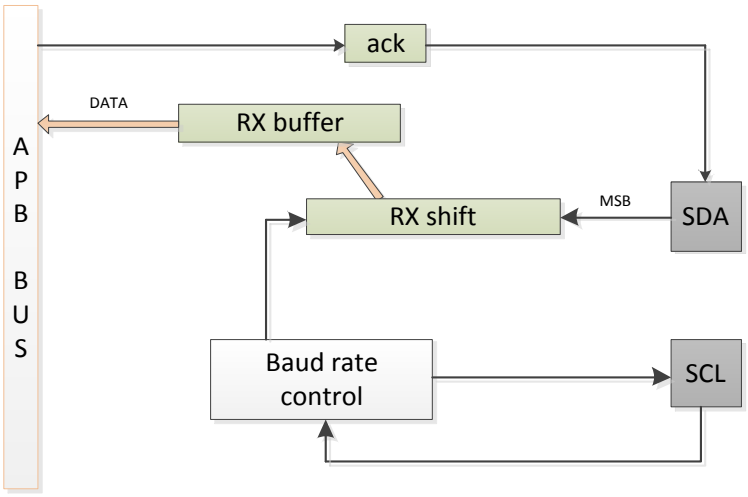

Figure 15-6 Data flow of Receiver

15.4 Functional description

The I2C module supports standard, fast, and fast plus communication modes. I2C peripherals can only be used as a master or slave mode at the same time. Once configured as slave mode, it will remain in slave mode unless the module is reset.

15.4.1 Master mode

In master mode, each time the master sends a byte of data, the BND flag is set to generate an interrupt request after the master sends the start signal.

The master can send repeated start signals, and re-address the slave and write or read data without sending the STOP signal, as shown in Figure 15-7.

The master supports clock synchronization. When multi-master/slave communication or slave stretch, clock synchronization can synchronize with the clock of other master slave: when the output of SCL of the master is high, high-level time count will not be started, but the count will start after the SCL of the bus is pulled up by all the masters. In this way, the clock synchronization between multiple masters is realized. At this time, the actual SCL clock frequency will be slow, because the SCL pull-up is charged by the external pull-up resistance, and the frequency depends on the value of the external bus capacitance and pull-up resistance.

Support master arbitration. In the case of multi-master communication, the I2C module compares the data transmitted on the bus bit by bit. When the master sends a 1 and the bus data is detected to be 0, the arbitration is lost, and the arbitration lost flag is set to generate an interrupt request and switch itself to the slave mode, because other masters may be addressing themselves at this time. The precondition of this function is that the clock synchronization function is enabled first.

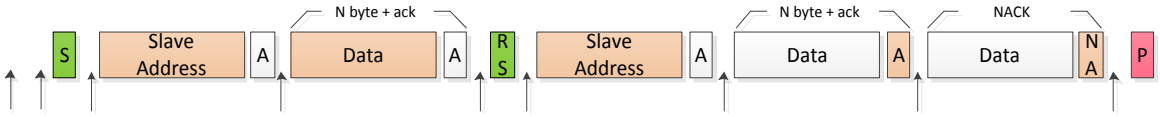

Figure 15-7 Master combined mode

15.4.2 Slave mode

In the slave mode, the slave supports four address matching modes: 7-bit address, 10-bit address, 7-bit range address, and broadcast address.

After the address of the slave is matched in the above four modes, the SAMF flag is set and the SRW flag is set to 0/1 according to the write/read signal of the master. Only after the address matches, subsequently once the master sends each byte, the BND flag is set. If the master sends a NACK signal while reading the slave, if the master continues to transmit later, the slave will not generate a BND flag, and needs to be re-addressed in order to communicate with the slave.

The slave has stretch function. When the slave receives a byte of data but the software does not read it or the slave has not yet prepared the data when the master reads, if this function is enabled, the slave will actively pull down the SCL bus to prevent the master from sending the data clock signal of the next frame. Of course, the master needs to enable the clock synchronization function to wait for the slave to release the SCL bus.

The slave has four status flags:

- **Transmit buffer empty flag(TXEF):** when the slave sending register is not loaded with sending data or one byte sending is completed, this flag is set and an interrupt request can be generated.
- **Transmit buffer underflow flag(TXUF):** when the slave has sent one byte of data, but has not written data of the next byte. And at this time, after the master has sent a byte of clock signal again, this flag is set and an interrupt request can be generated.

- **Receive buffer full flag(RXFF):** when the slave data register receives a byte of data, this flag is set and an interrupt request can be generated.
- **Receive buffer overflow flag(RXOF):** After the slave receives a byte of data, if the software does not read the data in time, and the master sends a byte of clock signal again at this time, this flag is set and an interrupt request can be generated.

15.4.3 SMBus

The I2C module complies with the System Management Bus (SMBus) Specification, version 2.0. The SMBus is a single-ended simple two-wire bus, which is typically used for low bandwidth communications.

15.4.3.1 SMBus – SCL/SDA low timeout detection

SCL/SDA low timeout detection is supported when I2C is used as master or slave. The timeout period is software configurable. SCL is detected by default, and SDA can be selected whether to detect. If I2C_CTRL2[PLTIE] = 1, an interrupt request will be generated when PLTF is set.

15.4.3.2 SMBus – alert address response

As a slave, I2C can respond to SMBus alert address 0001100X. After the address is matched by the slave, the I2C_STATUS1[SARF] flag is set at the same time when the I2C_STATUS0[SAMF] flag is set.

15.4.4 Interrupt request

I2C total has 11 interrupts.

Table 15-1 I2C Interrupt summary

Interrupt	Flag	Local enable	Global enable
One Byte Transfer End	BND		IICIE
Slave Address Match	SAMF		IICIE
Arbitration Lost	ARBLOST		IICIE
Bus START Detected	START	SSIE	IICIE
Bus STOP Detected	STOP	SSIE	IICIE
RX Buffer Overflow	RXOF	RXOFIE	IICIE
TX Buffer Overflow	TXUF	TXUFIE	IICIE
RX Buffer Full	RXFF	RXFIE	IICIE
TX Buffer Empty	TXEF	TXEIE	IICIE
Ack Get	RACK	NACKIE	IICIE
SDA/SCL low timeout	PLTF	PLTIE	IICIE

15.4.5 Slave low power wakeup

If the wake-up function is enabled in the low-power mode of the MCU, when the address matches, an ACK low signal will be generated at the 9th SCL, and a wake-up signal will be generated to wake up the MCU. The data followed by address byte won't be ACKed, I2C need to be initialed or received a new start signal to handle data.

15.5 Application note

15.5.1 Data transmission

Write STARTSTOP[START] as '1' to send a START signal to the I2C bus. After the transmission, write STARTSTOP[STOP] as '1' to send a STOP signal to the I2C bus.

For the master transmitter, set the I2C_CTRL0[TX] to 1. After writing the data register, the baud rate controller automatically starts to transmit data to the I2C bus. After the one-byte transmission is completed, the BND flag is set, indicating that data can be written again, write data or write 1 to the BND bit to clear the BND flag.

For the master receiver, set the I2C_CTRL0[TX] to 0, and the read operation of the data register will trigger the baud rate controller to automatically send a clock signal to the I2C bus, and ack signal is sent according to the I2C_CTRL0[TACK] control response, received data is loaded into the data register. The BND flag is set. Read the data register or write 1 to the BND bit to clear the BND flag.

For slave transmitter, write data to data register first. After receiving the entire clock signal read by the master, the data is sent out, the TXEF flag is set, and the BND flag is set. The write operation to the data register can clear the TXEF and BND flags, or write 1 to the BND bit to clear the BND flag too.

For the slave receiver, after the master sends one byte of entire data, the data is loaded into the slave data register, the RXFF flag is set, and the BND flag is set. Reading the data register can clear the RXFF and BND flags, or write 1 to the I2C_STATUS0[BND], the BND flag can also be cleared.

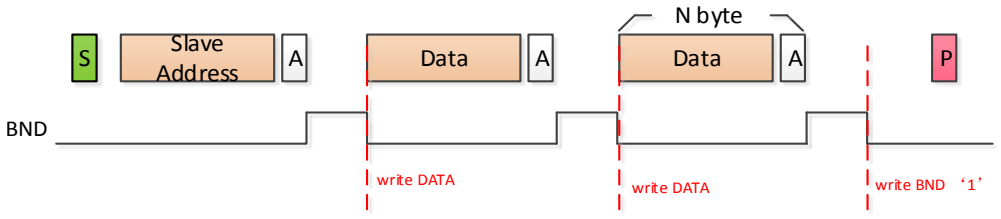

Figure 15-8 BND sequence of master write slave mode

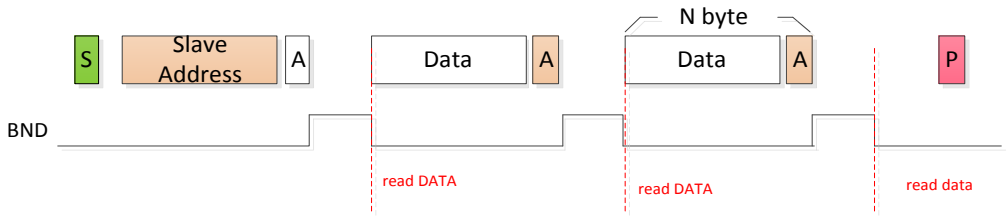

Figure 15-9 BND sequence of master read slave mode

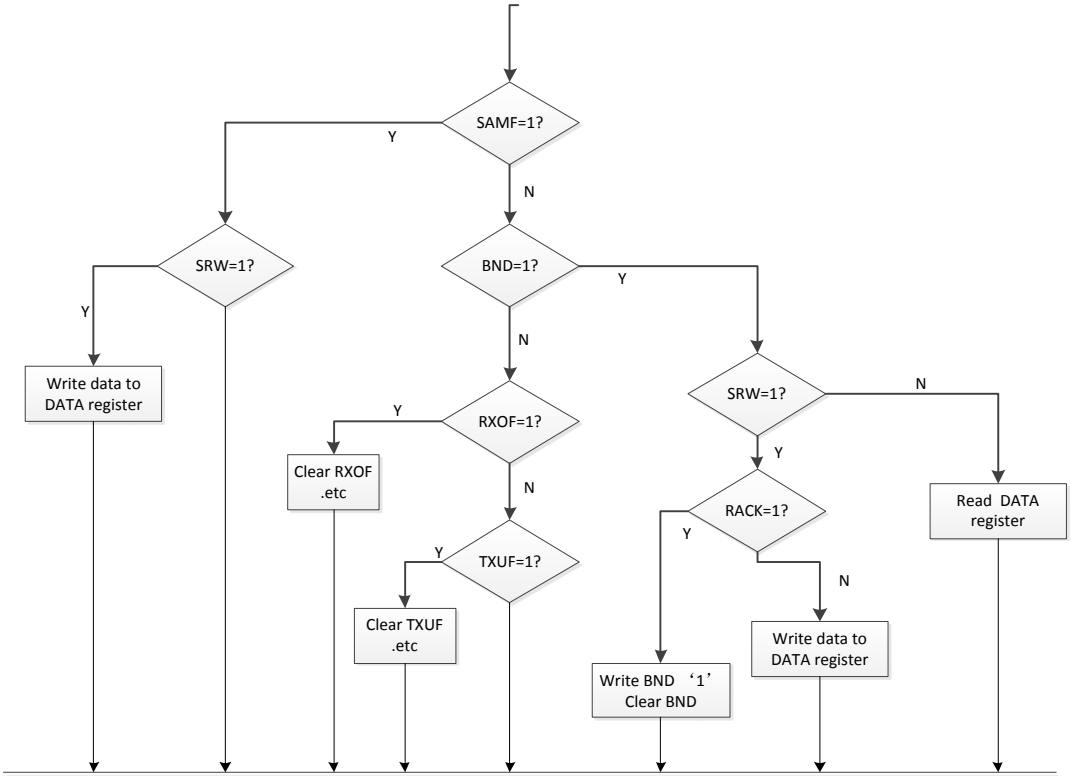

Figure 15-10 Typical I2C slave interrupt routine

15.5.2 ACK control

The I2C module controls the ack signal through software.

When the master receives data, the value of the TACK bit determines the response value on the 9th clock signal after the next byte is read.

When the slave receives data, the value of the TACK bit determines the response value on the 9th clock signal after the next byte is read.

15.6 Register description

Table 15-2 I2C register map

I2C0 base address:0x4000E000

Address	Register name	Width	Description
I2Cx base address + 0x00	I2C_ADDR0	32	Address register 0
I2Cx base address + 0x04	I2C_ADDR1	32	Address register 1
I2Cx base address + 0x08	I2C_SAMPLE_CNT	32	Baud rate configuration register 0
I2Cx base address + 0x0C	I2C_STEP_CNT	32	Baud rate configuration register 1
I2Cx base address + 0x10	I2C_CTRL0	32	Control register 0
I2Cx base address + 0x14	I2C_CTRL1	32	Control register 1
I2Cx base address + 0x18	I2C_CTRL2	32	Control register 2
I2Cx base address + 0x1C	I2C_CTRL3	32	Control register 3
I2Cx base address + 0x20	I2C_STATUS0	32	Status register 0
I2Cx base address + 0x24	I2C_STATUS1	32	Status register 1
I2Cx base address + 0x28	I2C_DGLCFG	32	Deglitch configuration register
I2Cx base address + 0x2C	I2C_DATA	32	Data port register
I2Cx base address + 0x30	I2C_STARTSTOP	32	Master START STOP signal control register

Note: in the above table, x=0.

15.6.1 Address Register 0(I2C_ADDR0)

Table 15-3 I2C_ADDR0 register

I2C_ADDR0 **Address register 0** **Reset: 0x000000FE**

Bit	31: 8	7	6	5	4	3	2	1	0
Name		AD[6: 0]							
Type		RW							
Reset		0x7F							

Field	Description
7: 1 AD[6: 0]	**Specifies the 7Bit address when I2C is slave mode** Set 7-bit address and the lower seven bits in the 10-bit address in slave mode.

15.6.2 Address Register 1(I2C_ADDR1)

Table 15-4 I2C_ADDR1 register

I2C_ADDR1 **Address register 1** **Reset: 0x000007F7**

Bit	31: 13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name		RMEN		RAD								AD[9: 7]		
Type		RW		RW								RW		
Reset		0		0x7F								0x07		

Field	Description
12 RMEN	**Enable range address in slave mode** 1: enable 0: disable Slave range address enable bit
10: 4 RAD	**Range address in slave mode** 7-bit slave range address value, when the range address is enabled, if the address received by the slave is larger than AD[6:0] and less than or equal to RAD[6:0], salve will get matched.
2: 0 AD[9: 7]	**Specifies the 10-bit address when I2C is slave mode** Contains the upper threes bits of the address in the 10-bit address scheme.

15.6.3 Baud Rate Configuration Register 0(I2C_SAMPLE_CNT)

Table 15-5 I2C_SAMPLE_CNT register

I2C_SAMPLE_CNT **Baud Rate Configuration Register 0** **Reset: 0x00000004**

Bit	31: 8	7	6	5	4	3	2	1	0
Name		SAMPLE_CNT							
Type		RW							
Reset		0x04							

Field	Description
7: 0 SAMPLE_CNT	**This adjusts the width of each sampling point.** Sample width = (SAMPLE_CNT +1) *T_{bclk}

15.6.4 Baud Rate Configuration Register 1(I2C_STEP_CNT)

Table 15-6 I2C_STEP_CNT register

I2C_STEP_CNT **Baud Rate Configuration Register 1** **Reset: 0x00000004**

Bit	31: 8	7	6	5	4	3	2	1	0
Name		STEP_CNT							
Type		RW							
Reset		0x04							

Field	Description
7: 0 STEP_CNT	**Specifies the number of samples per half pulse width, i.e. each high or low pulse.** half_baudrate width= (STEP_CNT +1) * sample width **Note: the value of STEP_CNT is 3 at minimum.**

15.6.5 Control Register 0(I2C_CTRL0)

Table 15-7 I2C_CRTL0 register

I2C_CTRL0 **CTRL0** **Reset: 0x00000000**

Bit	31: 8	7	6	5	4	3	2	1	0
Name		IICEN	IICIE	MSTR	TX	TACK	WUEN		
Type		RW	RW	RW	RW	RW	RW		
Reset		0	0	0	0	0	0		

Field	Description
7 IICEN	**I2C module enable** 1: enable 0: disable
6 IICIE	**I2C global interrupt enable** 1: enable the I2C module global interrupt 0: disable
5 MSTR	**I2C operation mode select** 1: master 0: slave **Note:** if ARB_LOST is set, this bit auto clear.
4 TX	**Transmission direction select** 1: transmit(TX) 0: receive(RX)

Field	Description
	In master mode, if TX=0, reading DATA register will trigger reading clock; TX = 1, reading DATA register will not trigger reading clock.
3 TACK	**Acknowledge control** 1: NACK will sent to the bus on the following(next) receiving byte 0: ACK will sent to the bus on the following(next) receiving byte Specifies the value driven onto the SDA during data acknowledge cycles for both master and slave receivers.
2 WUEN	**Wakeup function enable** 1: enable the wakeup function in low power mode. 0: disable the wakeup function. The I2C slave mode can wake the MCU up from low power mode when slave address matching occurs. **Note: When the I2C module is in slave mode and enters low power mode and 7bit, 10bit (ADEXT=1), or general broadcast address (GCAEN=1) match occurs, or range address match(RMEN=1), the I2C module will wake up the MCU.**

15.6.6 Control Register 1(I2C_CTRL1)

Table 15-8 I2C_CRTL1 register

I2C_CTRL1 **CTRL1** **Reset: 0x00000000**

Bit	31: 8	7	6	5	4	3	2	1	0
Name		GCAEN	ADEXT		SYNCEN	ARBEN		SAEN	STREN
Type		RW	RW		RW	RW		RW	RW
Reset		0	0		0	0		0	0

Field	Description
7 GCAEN	**Slave General Broadcast Address Enable** 1: enable 0: disable
6 ADEXT	**Slave Address Extension enable** 1: 10-bit address mode 0: 7-bit address mode
4 SYNCEN	**Master SCL Sync Enable** 1: enable 0: disable Enables the SCL synchronization function of master

Field	Description
3 ARBEN	**Master Arbitration Enable** 1: enable 0: disable Enables the master's arbitration function. **Note:** If I2C used in multi-master system, multi-master function must set both SYNC_EN and ARB_EN. If I2C used in single master system, and slave have SCL stretch function, then it can set SYNC_EN only.
1 SAEN	**Slave SMBus Alert Address Enable** 0: disable 1: enable
0 STREN	**Slave SCL stretch enable** 1: enable 0: disable Enable this bit, slave hardware will stretch SCL low after the 9th SCL falling per byte. **Note:** after SAMF=1, if SRW=1, TXEF=1 will cause SCL stretch. Otherwise SRW=0, RXFF=1 will cause SCL stretch. So when enable STREN and slave is transmitter(TX), after 9th SCL falling per byte (include address byte), slave will stretch SCL, until write DATA register. Similarly, when slave is receiver(RX), after 9th SCL falling per byte (without address byte) , slave will stretch SCL, until read DATA register. In slave mode, when enable GCAEN or MNTEN, slave cannot stretch SCL.

15.6.7 Control Register 2(I2C_CTRL2)

Table 15-9 I2C_CRTL2 register

I2C_CTRL2 **CTRL2** **Reset: 0x00000000**

Bit	31: 8	7	6	5	4	3	2	1	0
Name		RXOFIE	TXUFIE	RXFIE	TXEMIE		PLTIE	NACKIE	MNTEN
Type		RW	RW	RW	RW		RW	RW	RW
Reset		0	0	0	0		0	0	0

Field	Description
7 RXOFIE	**Slave RX Buffer Overflow Error Interrupt Enable** 1: enable 0: disable
6 TXUFIE	**Slave TX Buffer Underflow Error Interrupt Enable** 1: enable 0: disable
5	**Slave RX Buffer Full Interrupt Enable**

Field	Description
RXFIE	
	1: enable 0: disable
4 TXEMIE	**Slave TX Buffer Empty Interrupt Enable** 1: enable 0: disable
2 PLTIE	**SDA/SCL Low Timeout Interrupt Enable** 1: enable 0: disable
1 NACKIE	**NACK Get Interrupt Enable** 1: enable 0: disable
0 MNTEN	**Slave Monitor Function Enable** 1: enable 0: disable

15.6.8 Control Register 3(I2C_CTRL3)

Table 15-10 I2C_CRTL3 register

I2C_CTRL3 **CTRL3** **Reset: 0x00000000**

Bit	31: 16	15: 3	2	1	0
Name		PINLOW	TIMECFG		
Type		RW	RW		
Reset		0	0		

Field	Description
15:3 PINLOW	**SDA/SCL Low Timeout** Set SDA or SCL low level detection timeout. PLTF is set when SCL or SDA low level time more than PINLOW *256 clock cycles. When the PINLOW value is 0, the SDA/SCL low timeout detection function is disabled.
2 TIMECFG	**Timeout Detect Type** 1: SDA/SCL 0: SCL

15.6.9 Status register 0(I2C_STATUS0)

Table 15-11 I2C_STATUS0 register

I2C_STATUS0 **STATUS0** **Reset: 0x00000008**

Bit	31: 8	7	6	5	4	3	2	1	0
Name		BND	SAMF	BUSY	ARBLOST	READY	SRW		RACK
Type		R/W1C	R/W1C	RO	R/W1C	RO	RO		R/W1C
Reset		0	0	0	0	**1**	0		0

Field	Description
7 BND	**Byte End Flag** 1: one byte transfer finish (include ACK bit, total 9 SCL) 0: transfer in progress, one byte transfer not finish After reset, BND is '0'. BND is set only during data transmission period which between START and STOP signal on the bus. BND will set after per 9th SCL falling edge. In master mode, when sending data, the software writes DATA register to clear this bit and send DATA. When reading data, one byte transmission is completed, the software will clear this bit when reading DATA register, and send out the next byte data clock. In slave mode, SAMF is set after address matching. At this time, the BND flag will not be set. BND is set every time data transmission after address matching. Specifically, after address matching, the master writes data and the slave sets BND. When the slave takes the initiative to NACK, if the master continues to write data, the slave does not set BND. The master reads data and the slave sets BND. If NACK is received from the master, the master continues to read data and the slave does not set BND. **Note:** when I2C is slave and MNTEN=1, when BND is '1', then reading the DATA register will clear this bit. writing '1' can clear this bit too.
6 SAMF	**Slave Address Match Flag** 1: address match 0: address not match **Note:** this bit set by one of the following conditions. a. 7-bit address, address match b. 10-bit address, both 1st byte and 2nd byte match. And set after 2nd byte match. c. general call address matching. d. range address matching e. alert address matching **Note:** write '1' clear this bit.
5 BUSY	**Bus Busy** 1: bus is busy. 0: bus is idle.

Field	Description
	Indicates the status of the bus, and valid for both slave and master mode. This bit is set when a START signal is detected and cleared when a STOP signal is detected on the bus by hardware.
4 ARBLOST	**Arbitration Lost Flag** 1: arbitration lost 0: arbitration not lost **Note:** Write '1' clear this bit. When arbitration losts , I2C master will switch to slave mode, and MSTR is cleared by hardware.
3 READY	**Internal Hardware Core Is Ready for New Command** 1: internal hardware is ready for software new command 0: internal hardware is not ready This bit indicates the internal hardware status, and only valid for master mode. **Note:** this bit is valid for master, and maybe used when master generates a START/STOP signal. When I2C module is master mode, writing to the STARTSTOP register must wait for the READY bit to be 1 after generating the START/STOP signal.
2 SRW	**Slave Read/Write Direction** 1: slave is transmitter(TX), master reads from slave 0: slave is receiver(RX), master writes to slave
0 RACK	**Ack Received** 1: No acknowledge signal detected 0: Acknowledge signal was received after one byte of data This field is valid for transmitter(TX), master or slave sends.. **Note:** if NACKIE=1, RACK=1 will set interrupt, write 1 clear this bit.

15.6.10 Status register 1(I2C_STATUS1)

Table 15-12 I2C_STATUS1 register

I2C_STATUS1 **STATUS1** **Reset: 0x00000081**

Bit	31: 8	7	6	5	4	3	2	1	0
Name		IDLE	SARF	PLTF	GCMF	RXOF	TXUF	RXFF	TXEF
Type		RO	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
Reset		1	0	0	0	0	0	0	1

Field	Description
7 IDLE	**I2C hardware status** 1:idle

Field	Description
	0: not idle
6 SARF	**Slave SMBus Alert Address Match Flag** 1: Enable alert address and match 0: Disable alert address or no match **Write '1' clear this bit.**
5 PLTF	**SDA/SCL Low Timeout Flag** 1: SDA/SCL low timeout 0: No timeout **Note:** This flag will be set when SDA/SCL is kept low for longer than the set PINLOW. This flag cannot be cleared as long as SDA/SCL is always low. This flag must be cleared before I2C sends a START signal. Write '1' to clear this bit.
4 GCMF	**General Call Address Matching Flag** 1: general call address matching occurs 0: general call address matching does not occur **Note:** The I2C START or STOP signal will automatically clear the flag. Write '1' to clear this bit.
3 RXOF	**Slave RX Buffer Overflow Flag** 1: RX buffer overflow 0: Not overflow **Note:** When RX buffer overflow, new received data does not store to RX buffer. Write '1' clear this bit.
2 TXUF	**Slave TX Buffer Underflow Flag** 1: TX buffer underflow 0: No underflow **Note:** When TX buffer underflows, send the last DATA in TX buffer again. Write '1' clear this bit.
1 RXFF	**Slave RX Buffer Full Flag** 1: RX buffer full 0: Not full **Note:** Read DATA register will clear this bit. Write '1' clear this bit.
0	**Slave TX Buffer Empty Flag**

Field	Description
TXEF	1: TX buffer empty 0: Not empty **Note:** Write DATA register will clear this bit. Write '1' clear this bit.

15.6.11 Deglitch configuration register(I2C_DGLCFG)

Table 15-13 I2C_DGLCFG register

I2C_DGLCFG **DGLCFG** **Reset: 0x00000000**

Bit	31: 8	7	6	5	4	3	2	1	0
Name		DGLEN	STOPF	SSIE	STARTF	DGL_CNT			
Type		RW	R/W1C	RW	R/W1C	RW			
Reset		0	0	0	0	0			

Field	Description
7 DGLEN	**Deglitch Filter Enable** 1: enable 0: disable
6 STOPF	**Bus STOP Flag** 1: STOP detected on I2C bus 0: No STOP detected on I2C bus Hardware sets this bit when the STOP signal is detected on the I2C bus. **Note:** write "1" clear this bit.
5 SSIE	**Bus STOP or START Interrupt Enable** 1: enable STRAT or STOP detection interrupt 0: disable **Note:** write '1 ' clear STARTF or STOPF flag.
4 STARTF	**Bus START Flag** 1: START detected on I2C bus 0: No START detected Hardware set this bit when the START signal is detected on the I2C bus. **Note:** write '1' clear.
3: 0 DGL_CNT	**Deglitch Counter** 0h: No filter/bypass

Field	Description
	1-Fh: Filter glitches up to width of 1-15 clock cycles. Controls the width of the glitch. For any glitch whose size is less or equal to this width setting, the filter does not allow the glitch to pass.

15.6.12 Data Register(I2C_DATA)

Table 15-14 I2C_DATA register

I2C_DATA **DATA** **Reset: 0x000001FF**

Bit	31: 9	8	7	6	5	4	3	2	1	0
Name		MAK	DATA							
Type		RO	RW							
Reset		1	0xFF							

Field	Description
8 MAK	**Slave Monitor Function ACK Bit** For slave monitor, this field is the ACK bit from I2C bus. **Note:** MAK= '1', NACK MAK= '0', ACK For monitor, DATA[7:0] is the data transfer on I2C bus, and DATA[8] is the ACK bit.
7: 0 DATA	**Data** For master transmit(TX) mode, when data is written to this register, a data transfer is initiated. The most significant bit is sent first. For master receive(RX) mode, reading this register initiates a receiving sequence of the next byte. **Note**: when making the transition out of master receive mode, switch the I2C mode before reading the DATA register to prevent an inadvertent initiation of a master receive data transfer.

15.6.13 Master START STOP Signal Control Register(I2C_STARTSTOP)

Table 15-15 I2C_STARTSTOP register

I2C_STARTSTOP **Master START STOP signal control register** **Reset: 0x00000000**

Bit	31: 2	1	0
Name		STOP	START
Type		RW	RW
Reset		0	0

Field	Description
1 STOP	**Master Sends STOP signal** Write “1”, master will send STOP signal. Read this bit always return “0”.
0 START	**Master Sends START signal** Write “1”, master will send START or RESTART signal. Read this bit always return “0”.

16 Serial Peripheral Interface(SPI)

16.1 Introduction

Serial Peripheral Interface SPI (Serial Peripheral Interface) bus system is a synchronous serial peripheral interface that supports serial, synchronous, and full-duplex protocols. This SPI module is a 4-wire interface that includes master and slave both.

Figure 16-1 gives an example of the connection between SPI master and SPI slave.

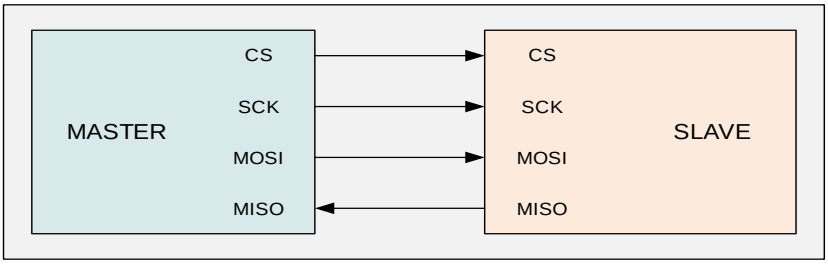

Figure 16-1 SPI system connection

16.2 Features

- Master mode or slave mode operation.
  - As master, the highest baud rate supports up to 8M
  - As slave, the highest baud rate supports up to 8M
- Full-duplex mode.
- Master programmable baud rate.
- Serial clock phase and polarity options.
- Configurable continuous or discontinuous CS (slave select) output.
- Mode error flag with CPU interrupt capability.
- Selectable MSB-first or LSB-first shifting.
- Configurable CS setup time, hold time and idle time.
- Configurable SCK high and low period.
- 4-16 bits transfer frame format selection.
- TX buffer underflow and RX buffer overflow flag with interrupt capability.
- Slave supports wake up function in Stop mode.

16.3 Block diagram

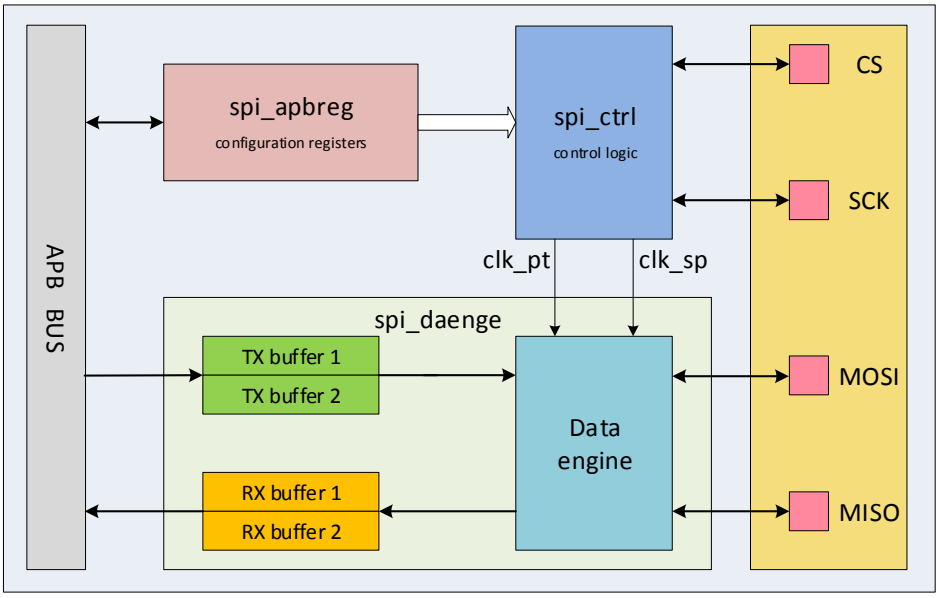

Figure 16-2 SPI block diagram

16.4 Functional description

16.4.1 Data flow & Algorithm

For master mode, data is written to a 16-bit TX buffer, then loaded to shift register by baud rate control unit. The output data most significant first or least significant first is controlled by TMSBF. After the numbers SCK periods specified by FRMSIZE, shift register shifts in data from MISO pin. Data received is stored into a 16-bit RX buffer.

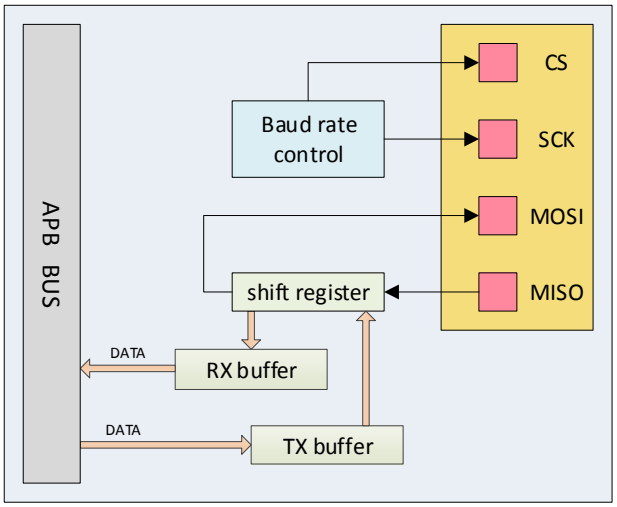

Figure 16-3 Data flow of master

For slave mode, data flow is similar to master mode. But the CS pin is the slave select input, and SCK is the SPI clock input from the master. Before a data transmission occurs, the CS pin of the slave SPI must be low. MOSI is the slave data input pin, and MISO is the data output pin.

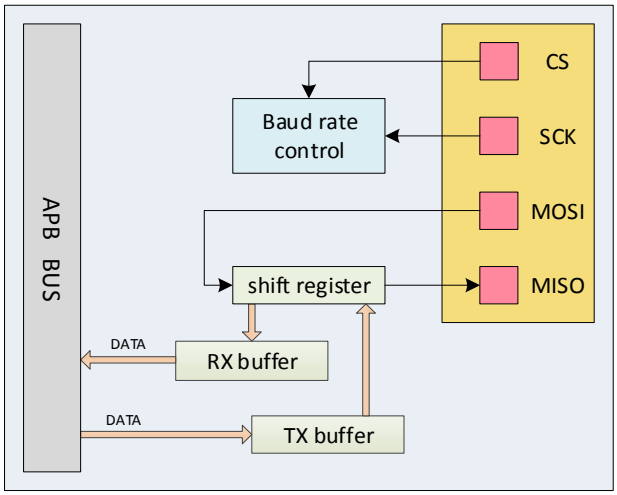

Figure 16-4 Data flow of slave

16.4.2 Input & Output timing

16.4.2.1 CPHA = 0 transfer format

The first edge on the SCK line is used to clock the first data bit of the slave into the master and the first data bit of the master into the slave. In some peripherals, the first bit of the slave's data is available at the slave's data out pin as soon as the slave is selected. In this format, the first SCK edge is issued a half cycle after CS has become low.

A half SCK cycle later, the second edge appears on the SCK line. When this second edge occurs, the value previously latched from the serial data input pin is shifted into the shift register.

After this second edge, the next bit of the SPI master data is transmitted out of the serial data output pin of the master to the serial input pin on the slave. This process continues for a total 16 edges on the SCK line, with data being latched on odd numbered edges and shifted on even numbered edges.

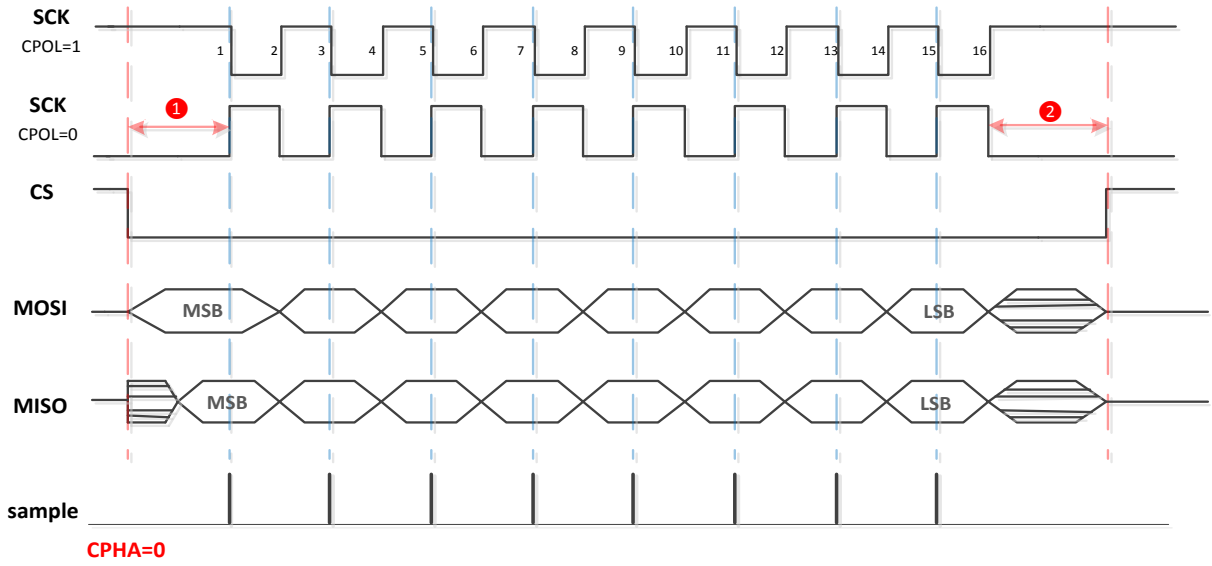

Figure 16-5 CPHA=0 transmission format

16.4.2.2 CPHA = 1 transfer format

Some peripherals require the first SCK edge before the first data bit becomes available at the data output pin, the second edge clocks data into the system.

The first edge of SCK occurs immediately after the half SCK clock cycle synchronization delay. This first edge commands the slave to transfer its first data bit to the serial data input pin of the master. A half SCK cycle later, the second edge appears on the SCK pin. This is the latching edge for both the master and slave.

When the third edge occurs, the value previously latched from the serial data input pin is shifted into the shift register. After this edge, the next bit of the master data is coupled out of the serial data output pin of the master to the serial input pin on the slave.

This process continues for a total 16 edges on the SCK line with data being latched on even numbered edges and shifting taking place on odd numbered edges.

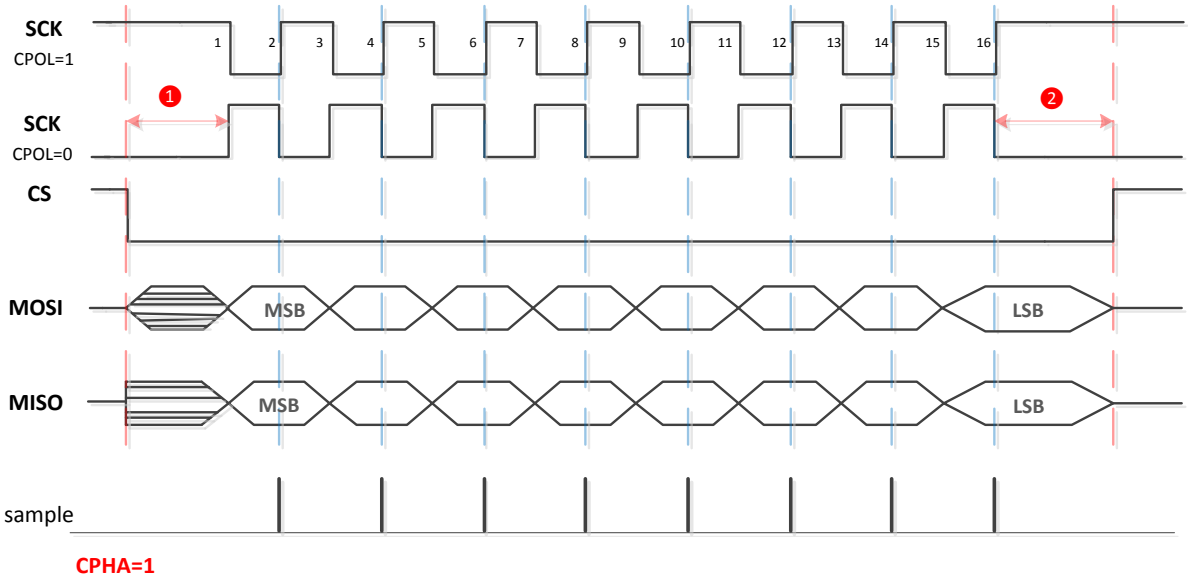

Figure 16-6 CPHA=1 transmission format

16.4.3 Master SCK output timing setting

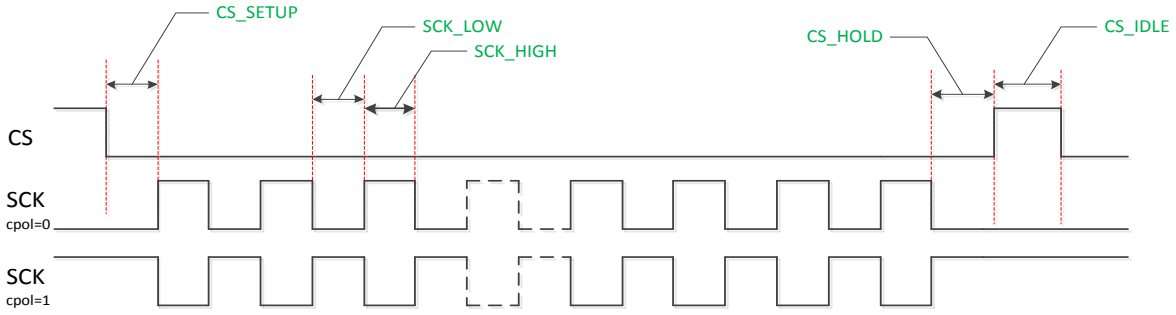

Figure 16-7 Baud rate generation

Baud rate: $f_{SCL}=f_{bclk}/(SCK\_LOW+1+SCK\_HIGH+1)$, f_{bclk} is the APB bus clock frequency.

16.4.4 Master mode fault detect

MODEF is set if the CS pin has been driven to low before SPI master initializes a transmission. Then master mode fault detect function is valid only when MSTR=1, MODFEN=1, CSOE=1.

The red part in Figure 16-8 is the time period for the master to detect the mode fault.

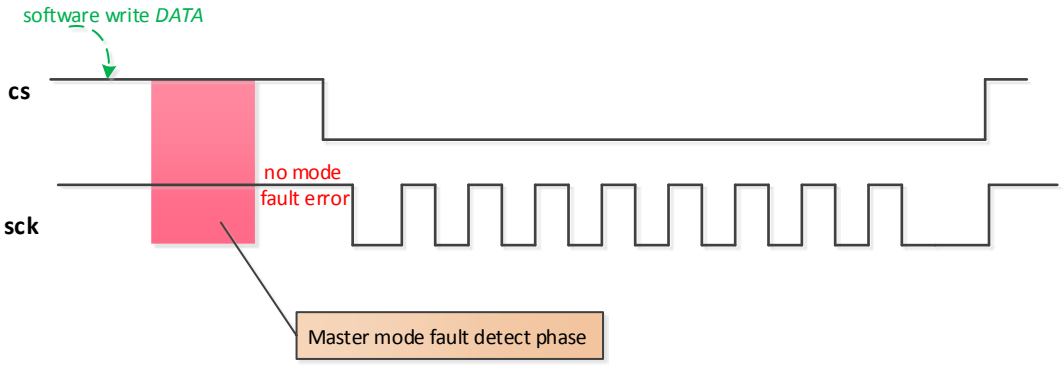

Figure 16-8 SCK output timing with mode fault detect enable

In some special cases, this mode fault detect function may can't detect the mode fault by master 1. If master 2 drive CS low during (1) period in Figure 16-9, master 1 will not set MODEF. Only master 2 drives CS low before (1) period, master 1 can set MODEF normally.

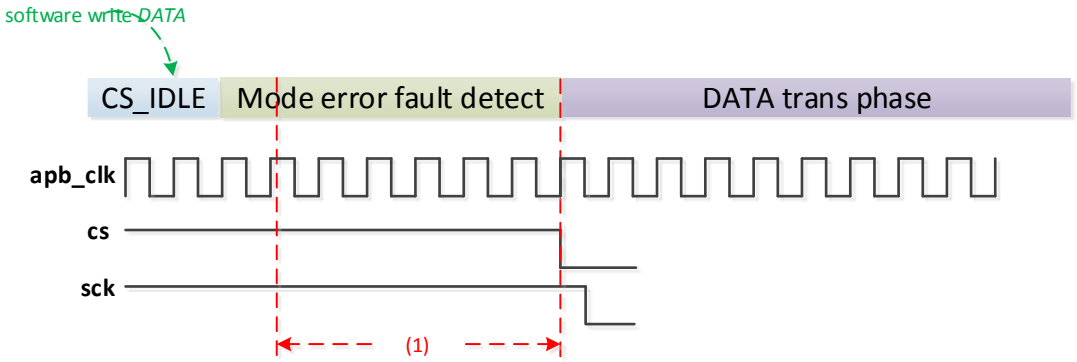

Figure 16-9 Limitation of mode fault detect

16.4.5 Slave low power wakeup

SPI slave in stop mode can generate an asynchronous interrupt to wake up the CPU from the low power mode when receives a transmission. To ensure the low power wakeup function of the SPI module is correct, system must follow some regulations. Before CPU enters low power mode, system must confirm that SPI module is in IDLE state. Software can check SPI Status Register(SPI_STATUS[8] IDLEF bit state. For master mode, tx buffers are empty, rx buffers are empty, and internal hardware is idle, IDLEF can be '1'. For slave mode, tx buffers are empty, rx buffers are empty, and CS is deassert(CS is high), IDLEF can be '1'. SPI module can't ensure data valid or wakeup function correct if CPU enter low power mode when SPI module is busy.

The slave generates asynchronous wakeup interrupt only when all of the following conditions apply:

a. SPI module is in slave mode.

b. SPI slave is in IDLE states.

c. WUEN bit is '1'.

d. The one byte wakeup sequence transmission ends.

CS high to low initializes the wakeup phase, and slave generates the asynchronous wakeup request after the numbers SCK cycle that FRMSIZE specified. SPI slave can receive the data of wakeup phase byte. The RXFF flag will be set after chip wakeup finish (clock recover), and reading data register will return the data master sent in wakeup phase byte. Only one byte can be transmitted during wakeup phase. During the wakeup phase, a continuous transmission from a master would destroy the data received, and may lead to RXFF flag can't be set rightly.

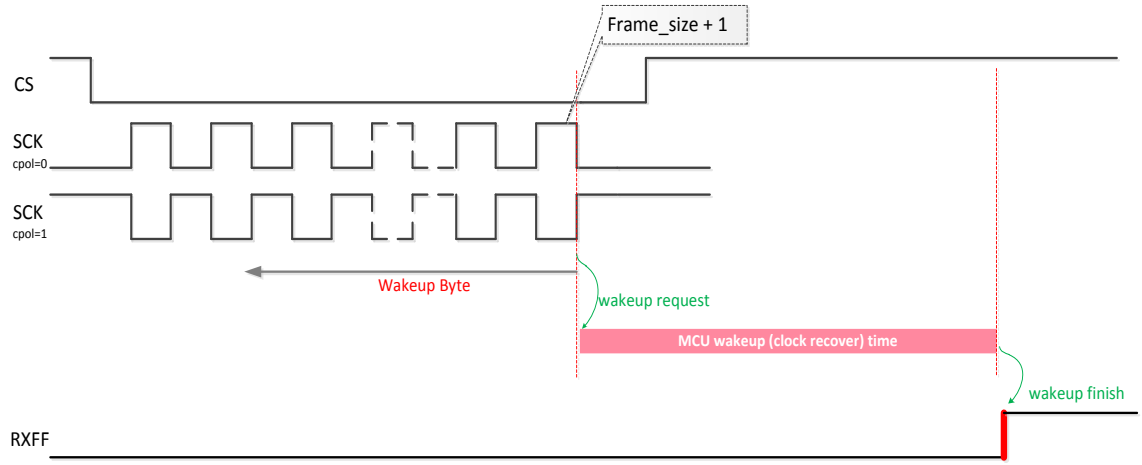

Figure 16-10 Wakeup sequence

16.4.6 Interrupt

SPI totally has 5 interrupts.

Table 16-1 Interrupt summary

Flag	Local enable
TXEF(TX Buffer Empty Flag)	TXEIE
RXFF(RX Buffer Full Flag)	RXFIE
TXUF(TX Underflow Flag)	TXUIE
RXOF(RX Overflow Flag)	RXOIE
MODFF(Mode Fault Error Flag)	MODFIE

16.5 Application note

16.5.1 Master CS continuous mode

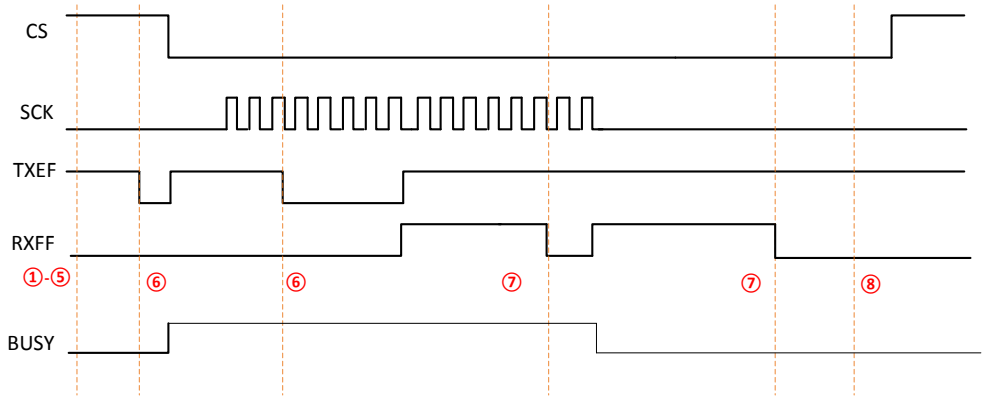

Figure 16-11 CS continuous mode

CS continuous output

1. Configure SPI configuration register 0(SPI_CFG0: CS_SETUP, CS_HOLD,SCK_LOW, SCK_HIGH.
2. Configure SPI Configuration Register 1(SPI_CFG1: CS_IDLE
3. Configure FRMSIZE, CPHA, CPOL, RMSBF, TMSBF, etc
4. Configure CSOE, CONT_CS, MSTR
5. SPIEN=1
6. TXEF=1, write data to DATA
7. RXFF=1, read data from DATA
8. Write CSRLS'1', release CS, then hardware goes to idle. That is, when CSOE = 1, CONT_CS = 1, CS is automatically pulled low by hardware. However, after the data is transmitted, it is needed for user software to write CSRLS'1' to pull CS high.

16.5.2 Master CS discontinuous output

1. Configure SPI configuration register 0(SPI_CFG0: CS_SETUP,CS_HOLD,SCK_LOW,SCK_HIGH.
2. Configure SPI Configuration Register 1(SPI_CFG1: CS_IDLE.
3. Configure FRMSIZE, CPHA, CPOL, RMSBF, TMSBF, etc.

4. Configure CSOE, CONT_CS, MSTR.
5. SPIEN=1
6. TXEF=1, write data to DATA.
7. RXFF=1, read data from DATA.

CS change to low or high by hardware when CS discontinuous mode. So software does not care the CSRLS.

16.5.3 Slave mode

1. Configure FRMSIZE, CPHA, CPOL, RMSBF, TMSBF, etc.
2. Configure MSTR.
3. SPIEN=1.
4. TXEF=1, write data to DATA.
5. RXFF=1, read data from DATA.

16.6 Register definition

Table 16-2 SPI register map

SPI0 base address: 0x4000c000

Address	Register name	Width	Description
SPIx base address +0x00	SPI_CFG0	32	SPI configuration register 0
SPIx base address +0x04	SPI_CFG1	32	SPI configuration register 1
SPIx base address +0x08	SPI_CMD	32	SPI command register
SPIx base address +0x0c	SPI_STATUS	32	SPI status register
SPIx base address +0x10	SPI_DATA	32	SPI data register
SPIx base address +0x14	SPI_CFG2	32	SPI configuration register 2

Note: in the above table, x=0.

16.6.1 SPI configuration register 0(SPI_CFG0)

Table 16-3 SPI_CFG0 register

SPI_CFG0 **SPI configuration register 0** **Reset: 0x05050505**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	CS_SETUP								CS_HOLD							
Type	RW								RW							
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	SCK_LOW								SCK_HIGH							
Type	RW								RW							
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	1

Bits	Description
31: 24 CS_SETUP	**CS Setup Time Configuration** The chip select setup time = (CS_SETUP +1)*CLK_PERIOD, where CLK_PERIOD is the cycle time of the clock the SPI engine adopts.
23: 16 CS_HOLD	**CS Hold Time Configuration** The chip select hold time = (CS_HOLD +1)*CLK_PERIOD.
15:8 SCK_LOW	**SCK Low Time Configuration** The SCK clock low time = (SCK_LOW +1) * CLK_PERIOD. **Note: When CPOL is 0, this bit configures the time of SCK_LOW. When CPOL is 1, this bit configures the time of SCK_HIGH.**
7: 0 SCK_HIGH	**SCK High Time Configuration** The SCK clock high time = (SCK_HIGH +1) * CLK_PERIOD. **Note: When CPOL is 0, this bit configures the time of SCK_HIGH. When CPOL is 1, this bit configures the time of SCK_LOW.**

16.6.2 SPI Configuration Register 1(SPI_CFG1)

Table 16-4 SPI_CFG1 register

SPI_CFG1 **SPI configuration register 1** **Reset: 0x027C0005**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name		WKUEN		CONT_CS		MODFEN	CSOE		FRMSIZE				RMSBF	MSBF	CPHA	CPOL
Type		RW		RW		RW	RW		RW				RW	RW	RW	RW
Reset		0		0		0	1		0	1	1	1	1	1	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name			MODFIE	MSTR	RXOIE	TXUIE	RXFIE	TXEIE	CS_IDLE							
Type			RW	RW	RW	RW	RW	RW	RW							
Reset			0	0	0	0	0	0	0	0	0	0	0	1	0	1

Bits	Description
30 WKUEN	**Slave Wakeup Function Enable** 0: slave wake up function disable 1: slave wake up function enable
28 CONT_CS	**CS Continuous Output Enable** 0: CS output non-continuous 1: CS output continuous
26 MODFEN	**Master Mode Fault Detect Enable** 0: disable the master mode fault detect function 1: enable the master mode fault detect function
25 CSOE	**CS Hardware Output Enable** 0: disable the CS hardware output 1: enable the CS hardware output
23: 20 FRMSIZE	**Frame Size** 0000: 4bit 0001: 4bit 0010: 4bit 0011: 4bit 0100: 5bit ``` 1110: 15bit 1111: 16bit
19 RMSBF	**RX MSB First** 0: the first bit of shifter shift in is the LSB of the input data 1: the first bit of shifter shift in is the MSB of the input data
18 MSBF	**TX MSB First** 1: TX MSB first (MSB first shift out) 0: TX LSB first (LSB first shift out)
17 CPHA	**Clock Phase** 1: the first SCK transition edge is the data capture edge 0: the first SCK transition edge is the data shift out edge
16 CPOL	**Clock Polarity**

Bits	Description
	0: SCK is 0 when idle 1: SCK is 1 when idle
13 MODFIE	**Mode Fault Interrupt Enable** 0: disable 1: enable
12 MSTR	**Master or Slave Mode Selection** 0: slave mode 1: master mode
11 RXOIE	**RX Buffer Overflow Interrupt Enable** 0: disable 1: enable
10 TXUIE	**TX Buffer Underflow Interrupt Enable** 0: disable 1: enable
9 RXFIE	**RX Buffer Full Interrupt Enable** 0: disable 1: enable **Note:** RX buffers Not Empty will generate an interrupt.
8 TXEIE	**TX Buffer Empty Interrupt Enable** 0: disable 1: enable **Note:** TX buffers Not Full will generate an interrupt.
7: 0 CS_IDLE	**CS Idle Time** CS IDLE time = (CS_IDLEA_COUNT+1)*CLK_PERIOD

16.6.3 SPI Command Register(SPI_CMD)

Table 16-5 SPI_CMD register

SPI_CMD **SPI command register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name										RO TRIG	CS RLS	SW RST				SPI EN
Type										RW	RW	RW				RW
Reset										0	0	0				0

Bits	Description
6 ROTRIG	**Master RX only mode trigger** **Note:** write this bit '1' will trigger a sequence of read, while ROEN=1 in CFG2. Read this bit will return '0' always.
5 CSRLS	**CS Release** 0: no effect 1: release CS Software writes this bit '1', then CS will go to high. Read this bit always return "0". This bit is valid for CS continuous output(CONT_CS=1, CSOE=1).
4 SWRST	**Software reset** 0: no effect 1: reset **Note:** this reset only resets master engine/buffer/flag logic, slave buffer/flag logic. CFG0/CFG1/CFG2/CMD control bits will not reset.
0 SPIEN	**SPI Enable** 0: disable 1: enable

16.6.4 SPI Status Register(SPI_STATUS)

Table 16-6 SPI_STATUS register

SPI_STATUS **SPI status register** **Reset: 0x00000101**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name								IDLEF	MEBY			MODEF	RXOF	TXUF	RXFF	TXEF
Type								RO	RO			RO	R/W1C	R/W1C	RO	RO
Reset								1	0			0	0	0	0	1

Bits	Description
8 IDLEF	**SPI IDLE Flag** 0: SPI module hardware not IDLE 1: SPI module hardware IDLE **Note:** For master, TX buffer is empty, RX buffer is empty, and internal hardware is idle, this bit can be '1'. For slave, TX buffer is empty, RX buffer is empty, and CS deassert(not be selected) , this bit can be '1'.
7 MEBY	**SPI Master Engine Busy Flag** 0: SPI master hardware's one frame transmission finish 1: SPI master hardware's one frame transmission not finish
4 MODEF	**Mode Fault Error Flag** 0: no master error detected 1: master mode error detected When SPI configured as a master, it will detect the CS line state before drive CS low. If CS have been low, indicating another master have initial a transmission, MODEF flag will be set. **Note:** Write "SWRST=1" clear this bit, SPI module has to be reset by software.

Bits	Description
3 RXOF	**RX Buffer Overflow Flag** 0: no overflow 1: RX buffer overflow There are two FIFOs in the receive buffer. If the RX buffer data is not read out by user in time after receiving two frames, the next received data will come and generate RX overflow. **Note:** Write "1" clear this bit. When an overflow occurs, it needs to be cleared in time to avoid affecting the RXFF flag. If overflow, then only 1 frame valid data in DATA register can be read out.
2 TXUF	**TX Buffer Underflow Flag** 0: not underflow 1: TX buffer underflow In slave mode, if no data is written to the TX data register, the master will start to communicate and TX down flow will occur. **Note**: write "1" clear this bit. When an overflow occurs, it needs to be cleared in time.
1 RXFF	**RX Buffer Full Flag** 0: not full 1: full **Note:** As long as rx buffers have valid data received (1 byte or 2 bytes), this bit will be '1'. So this bit '1' does not indicate that 2 frame data received in rx buffers. Reading DATA register will clear this bit hardware automatically. This bit named RX Buffers Not Empty is more reasonable maybe.
0 TXEF	**TX Buffer Empty Flag** 0: not empty 1: empty **Note:** As long as tx buffers is not full (2 frames data), this bit will be '1'. Writing DATA register will clear this bit hardware automatically. This bit named TX buffers Not Full is more reasonable maybe.

16.6.5 SPI Data Register(SPI_DATA)

Table 16-7 SPI_DATA register

SPI_DATA **SPI data register** **Reset: 0x00000000**

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

Bits	Description
15:0 DATA	**SPI Data Port Register** Read: read will return DATA received Write: write the DATA to be sent

16.6.6 SPI Configuration Register 2(SPI_CFG2)

Table 16-8 SPI_CFG2 register

SPI_CFG2 **SPI configuration register 2** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	CS_CHECK															
Type	RW															
Reset	0															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name													ROEN	TOEN	MNDC	
Type													RW	RW	RW	
Reset													0	0	0	

Bits	Description
31:24 CS_CHECK	**CS Check Count Period** 0: 1 functional clock check period 1: 2 functional clock check periods ``` 255: 256 functional clock check periods

Bits	Description
3 ROEN	**RX Only Mode Enable** 0: disable 1: enable **Note:** RX only mode, TXEF will keep 0. Not trigger RXEF IRQ even TXEIE=1. When the master uses RX only mode, it is recommended to use poll and interrupt transmission
2 TOEN	**TX Only Mode Enable** 0: disable 1: enable **Note:** TX only mode, RXFF does not set after one frame transfer finished. Not trigger RXFF IRQ even if RXFIE=1.
1 MNOV	**Master No Overflow Mode** 0: disable 1: enable If rxbuff is full, then write to txbuff will not trigger new sending.

17 Watchdog Timer(WDG)

17.1 Introduction

The Watchdog Timer (WDG) module is an independent timer that is available for system use. It provides a safety feature to ensure that software is executing as planned and that the CPU is not stuck in an infinite loop or executing unintended code. If the WDG module is not serviced (refreshed) within a certain period, it resets the MCU. This mechanism is often used in occasions with high security requirements.

17.2 Features

- Four clock source inputs.
- Optional fixed 256 clock prescaler
- Programmable 32 bit timeout period.
- Window mode option for the refresh mechanism.
- Timeout interrupt to allow post-processing diagnostics.

17.3 Block diagram

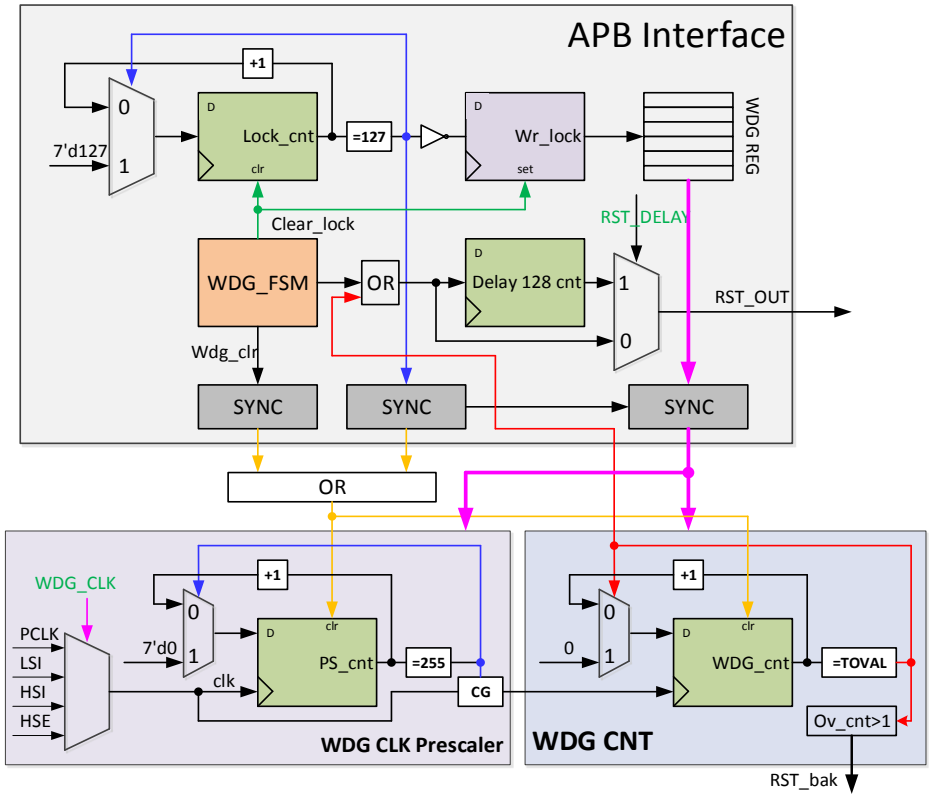

Figure 17-1 WDG block diagram

17.4 Functional description

17.4.1 Basic Watchdog

The watchdog has four clock sources: bus clock, internal 32 kHz RC oscillator, internal 32 MHz RC oscillator and external XOSC clock source. The watchdog timer uses a 32-bit programmable up counter and an optional fixed 256 clock prescaler.

After the watchdog is enabled, it starts counting. When the counting reaches the TOVAL value, a system reset will occur. Before the counting reaches the TOVAL value, refresh the watchdog to reset the counter and restart counting.

17.4.2 Watchdog default timeout behavior

After the MCU is powered on, the watchdog counter counts using the bus clock(16MHz) by default and times out after 0x271000 clock cycles. This will cause the MCU to reset about 160ms after power on. In order to avoid this situation, user needs to ensure that the watchdog is configured or refreshed within about 160ms after the MCU is powered on.

17.4.3 Window Watchdog

The watchdog has a window mode. In this mode, refreshing the watchdog before the counter reaches the window value WIN or not refreshing the watchdog before the calculator reaches the TOVAL value will cause the system to reset. When the counter is higher than the WIN value and less than TOVAL, refreshing the watchdog resets the counter and restarts counting.

17.4.4 Low-power behavior

In the Stop mode, the watchdog can keep running, but it needs to use the internal 32 kHz RC oscillator as the clock source, and the WDG needs timeout twice to cause system reset.

17.4.5 Debug mode

In debug mode, the watchdog cannot be enabled.

17.5 Application note

17.5.1 Configuring the Watchdog

The condition for configuring all registers of the watchdog is that the update bit WDG_CS0[UPDATE] is 1 and the watchdog is unlocked. After unlocking, any register of the watchdog can only be configured within 128 bus clocks, and then all registers are automatically locked, and for reconfiguration, it is needed to meet the above two conditions again.

Once user configures WDG_CS0[UPDATE] to 0, the watchdog register configuration cannot be modified unless a reset is forced. When WDG_CS0[UPDATE] is 0, unlocking will cause the system reset.

Unlock sequence: write 0xE064D987 and 0x868A8478 to WDG_CNT register successively. If the written value is incorrect or the order is reversed, it will result in system reset.

17.5.2 Watchdog refresh mechanism

In basic watchdog or window watchdog mode, in order to ensure that the watchdog does not reset the system, it is needed for the software to refresh the watchdog within the specified time. After the watchdog is refreshed, the watchdog counter starts counting from 0 again, and the software needs to refresh the watchdog again. This mechanism makes the software must refresh the watchdog regularly, which reflects the normal operation of the program to a certain extent. When the program runs away unexpectedly, the watchdog will reset the system.

Refresh sequence: write 0x7908AD15 and 0x5AD5A879 to WDG_CNT register successively. If the written value is incorrect or the order is reversed, it will result in system reset.

17.5.3 Watchdog interrupt

The watchdog has an interrupt function. After the interrupt is enabled, if the watchdog counter times out, the watchdog will not reset the system immediately, but it will reset the system after a delay of 128 bus clocks. The watchdog delays forcing a reset for 128 bus clocks to allow the interrupt service routine (ISR) to perform tasks. The user software can perform simple program processing in the ISR, but it is not recommended to refresh the watchdog at this time.

17.6 Register definition

Table 17-1 WDG register map

WDG base address: 0x4000B000

Address	Register name	Width	Description
WDG base address +0x00	WDG_CS0	32	Watchdog Control and Status Register(CS0)
WDG base address +0x04	WDG_CS1	32	Watchdog Control and Status Register(CS1)
WDG base address +0x08	WDG_CNT	32	Watchdog Counter Register
WDG base address +0x0C	WDG_TOVAL	32	Watchdog Timeout Value Register
WDG base address +0x10	WDG_WIN	32	Watchdog Window Value Register

17.6.1 Watchdog Control and Status Register 0(WDG_CS0)

Table 17-2 WDG_CS0 register

WDG_CS0 **Control and Status Register 0** **Reset: 0x000000A0**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									EN	INT	UPDATE					
Type									RW	RW	RW					
Reset									1	0	1					

Bits	Description
7 EN	**Watchdog Enable** 0: Watchdog disabled. 1: Watchdog enabled. This bit enables the watchdog counter to start counting.
6 INT	**Watchdog Interrupt** 0: Watchdog interrupts are disabled. Watchdog resets are not delayed. 1: Watchdog interrupts are enabled. Watchdog resets are delayed by 128 bus clocks.

Bits	Description
	This bit configures the watchdog to generate an interrupt request upon a reset-triggering event (timeout or illegal write to the watchdog), prior to forcing a reset. After the interrupt vector fetch, the reset occurs after a delay of 128 bus clocks.
5 UPDATE	**Watchdog configuration update bit** 0: Updates not allowed. After the initial configuration, the watchdog cannot be later modified without forcing a reset. 1: Updates allowed. Software can modify the watchdog configuration registers within 128 bus clocks after performing the unlock write sequence.

17.6.2 Watchdog Control and Status Register 1(WDG_CS1)

Table 17-3 WDG_CS1 register

WDG_CS1 **Control and Status Register 1** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name									WIN	FLG		PRES			CLK	
Type									RW	R/W1C		RW			RW	
Reset									0	0		0			0	

Bits	Description
7 WIN	**Watchdog Window** 0: Window mode disabled. 1: Window mode enabled. This bit enables window mode.
6 FLG	**Watchdog Interrupt Flag** 0: No interrupt occurred. 1: An interrupt occurred. This bit is an interrupt indicator when INT is set.
4 PRES	**Watchdog Prescalar** 0: 256 prescalar disabled. 1: 256 prescalar enabled. This bit enables a fixed 256 pre-scaling of watchdog counter reference clock.
1: 0 CLK	**Watchdog Clock** 00: Bus clock.

Bits	Description
	01: 32 kHz internal RC oscillator. 10: 32 MHz internal RC oscillator. 11: External clock source(XOSC). Select the WDG counting clock source

17.6.3 Watchdog Counter Register(WDG_CNT)

Table 17-4 WDG_CNT register

WDG_CNT **Counter register** **Reset: 0x00000000**

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

Bits	Description
31: 16 CNT	**Watchdog Counter Register** The watchdog counter registers provide access to the value of the free running watchdog counter. Software can read the counter registers at any time. Software cannot modify directly to the watchdog counter. However, two write sequences to these registers have special functions: the refresh sequence and the unlock sequence.

17.6.4 Watchdog Timeout Value Register(WDG_TOVAL)

Table 17-5 WDG_TOVAL register

WDG_TOVAL **Timeout value register** **Reset: 0x00271000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	TOVAL[31: 16]															
Type	RW															
Reset	0x27															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	TOVAL[15: 0]															
Type	RW															
Reset	0x1000															

Bits	Description
31: 0 TOVAL	**Watchdog Timeout Value Register** The watchdog counter is continuously compared with the timeout value. If the counter reaches the timeout value, the watchdog forces a reset. The counter length is equivalent to WDG_TOVAL+1, the default value is 0x271000.

17.6.5 Watchdog Window Value Register(WDG_WIN)

Table 17-6 WDG_WIN register

WDG_WIN **Window value register** **Reset: 0x00000000**

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

Bits	Description
31: 0 WIN	**Watchdog Window Register** When window mode is enabled (WDG_CS1[WIN] is set), WDG_WIN determines the earliest time that a refresh sequence is considered valid. Refreshing earlier than the window value will force a system reset.

18 Real Time Counter(RTC)

18.1 Introduction

Real time counter module (RTC), the main function is real-time counting. In Stop low power mode, RTC can keep running and wake up the MCU.

18.2 Features

Features of the RTC module include:

- 32-bit up-counter.
- Programmable 20 bit prescaler.
- Count overflow to flip GPIO.

18.3 Block diagram

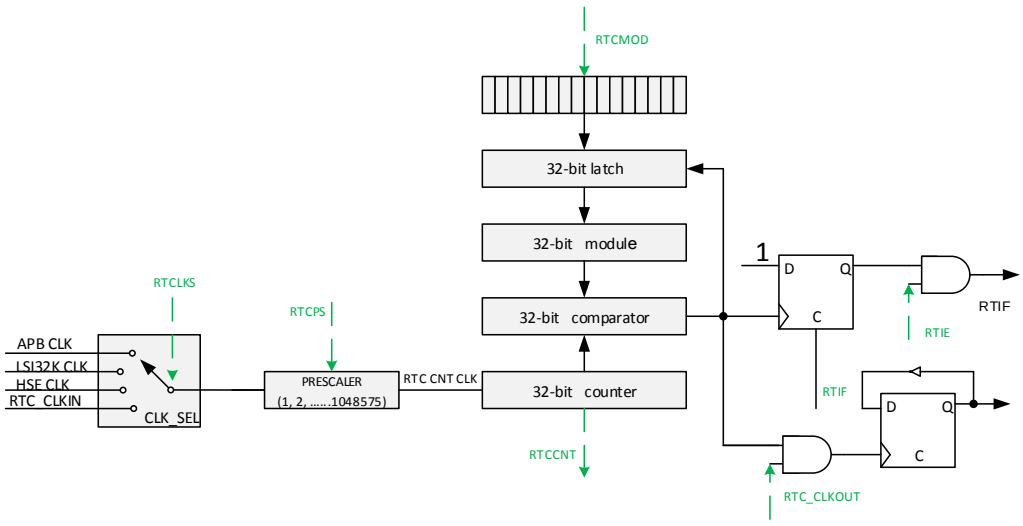

Figure 18-1 RTC block diagram

18.4 Functional description

18.4.1 Clock source selection

The RTC modules has four clock sources: bus clock, internal 32 kHz RC oscillator, HSE clock and external RTCIN pin input clock source.

18.4.2 Counting

RTC is an upwardly increasing counter. After the count reaches the preset modulus value, it will generate the RTC overflow flag. If the RTC overflow interrupt is enabled, RTC interrupt request is generated, and the RTC counter will start a new count from 0 after overflow. The RTC also has a built-in prescaler that upwardly counts. When it reaches the preset prescaler value, the prescaler overflow flag is generated. If the prescaler interrupt is enabled, prescaler interrupt request is generated. After the prescaler count overflows, it will start a new count from 0.

18.4.3 RTC timing signal output

PA13 can be used as RTC_CLKOUT function, which can flip PA13 when RTC overflows.

18.4.4 Low power wake up

In the low-power Stop mode of the MCU, RTC can be used as a wake-up source to wake up the MCU. It is necessary to select the internal 32K as the clock source and start the module before entering the low-power mode. In Stop mode, an RTC interrupt is generated after the RTC overflows to wake up the MCU. The prescaler interrupt cannot wake up the MCU. After waking up from the low power mode, the RTC counter will continue counting.

18.5 Application note

18.5.1 Basic use of RTC

Before using the RTC module, configure the_RTC_SC register, initialize the clock, modulus, interrupt, etc. of the RTC module. When the prescaler is configured to a non-zero value finally, the RTC counter and the prescaler counter start counting.

When the RTC prescaler counter overflows, the RTC counter increase and the flag bit RPIF is set, write 1 to clear the flag. When the RTC counter overflows, the flag bit RTIF is set, and write 1 to clear the flag. When the counter is running, modification to RTC_CLK or RTC_PS will clear the counter.

When the RTCO bit is 1, enable the RTC counter overflow to flip the designated GPIO, and the GPIO needs to be multiplexed as the RTC_CLKOUT function.

18.5.2 RTC low power wake up

To use the RTC to wake up the MCU in the low-power Stop state, it is needed to enable the RTC wake-up in the SPM module, and select the internal 32K clock or RTC_CLKIN clock as the RTC clock source and start timing.

18.6 Register definition

Table 18-1 RTC register map

RTC base address: 0x40008400

Address	Register name	Width	Description
RTC base address + 0x00	RTC_SC	32	RTC Control and Status Register
RTC base address + 0x04	RTC_MOD	32	RTC Modulo Register
RTC base address + 0x08	RTC_CNT	32	RTC Counter Register
RTC base address + 0x0C	RTC_PS	32	RTC Clock Prescaler Register
RTC base address + 0x10	RTC_PSCNT	32	RTC Prescaler Counter Register

18.6.1 RTC Control and Status Register(RTC_SC)

Table 18-2 RTC_SC register

RTC_SC **Control and Status register** **Reset: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name															RPIF	RPIE
Type															R/W1C	RW
Reset															0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RTCLKS								RTIF	RTIE		RTCO				
Type	RW								R/W1C	RW		RW				
Reset	0								0	0		0				

Bits	Description
17 RPIF	**Real-Time Prescaler Interrupt Flag** 0: RTC Prescaler counter has not reached the value in the RTC prescaler divider register. 1: RTC Prescaler counter has reached the value in the RTC prescaler divider register. This status bit indicates the RTC Prescaler counter register reached the value in the RTC prescaler divider register. Writing a logic 0 has no effect. Writing a logic 1 clears the bit and the real-time interrupt request. Reset clears RPIF to 0.
16 RPIE	**Real-Time Prescaler Interrupt Enable** 0: Real-time prescaler interrupt requests are disabled. 1: Real-time prescaler interrupt requests are enabled. This read/write bit enables real-time prescaler interrupts. If RPIE is set, then an interrupt is generated when RPIF is set.

Bits	Description
15: 14 RTCLKS	**Real-Time Clock Source Select** 00: Bus clock. 01: Internal 32kHz oscillator. 10: HSE. 11: External RTCIN pin input clock. This read/write field selects the clock source input to the RTC prescaler. Switching the clock source clears the prescaler and RTCCNT counters. Reset clears RTCLKS to 0. RTC overflow time = (((MOD +1) * (RTCPS+1)) / RTCLKS).
7 RTIF	**Real-Time Interrupt Flag** 0: RTC counter has not reached the value in the RTC modulo register. 1: RTC counter has reached the value in the RTC modulo register. This status bit indicates the RTC counter register reached the value in the RTC modulo register. Writing a logic 0 has no effect. Writing a logic 1 clears the bit and the real-time interrupt request.
6 RTIE	**Real-Time Interrupt Enable** 0: Real-time interrupt requests are disabled. 1: Real-time interrupt requests are enabled. This read/write bit enables real-time interrupts. If RTIE is set, then an interrupt is generated when RTIF is set.
4 RTCO	**Real-Time Counter Output** 0: Real-time counter output are disabled. 1: Real-time counter output are enabled. The read/write bit enables counter to toggle output on pinout. If this bit is set, the RTC_CLKOUT pinout will be toggled when RTC counter overflows.

18.6.2 RTC Modulo Register(RTC_MOD)

Table 18-3 RTC_MOD register

RTC_MOD **RTC Modulo** **Reset: 0x00000000**

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

Bits	Description
31: 0 MOD	**RTC Modulo** The modulo value is used to compare with the current count value (RTC_CNT). When the count value is equal to the modulo, the count will be reset to 0x0 and set SC[RTIF].

18.6.3 RTC Counter Register(RTC_CNT)

Table 18-4 RTC_CNT register

RTC_CNT **RTC count register** **Reset: 0x00000000**

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

Bits	Description
31: 0 CNT	**RTC counter** This read-only field contains the current value of the 32-bit counter. Writes have no effect to this register. Reset or writing different values to SC[RTCLKS] and PS[RTCPS] clear the counter to 0x0.

18.6.4 RTC Clock Prescaler Register(RTC_PS)

Table 18-5 RTC_PS register

RTC_PS **Prescaler** **Reset: 0x00000000**

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

Bits	Description
19: 0 RTCPS	**RTC Prescaler Select** Changing the prescaler value clears the prescaler and RTCCNT counters. When RTCPS equals 0, RTC counter stops counting. When RTCPS does not equal 0, RTC counter on and starts counting.

18.6.5 RTC Prescaler Counter Register(RTC_PSCNT)

Table 18-6 RTC_PSCNT register

RTC_PSCNT **Prescaler counter register** **Reset: 0x00000000**

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

Bits	Description
19: 0 PSCNT	**RTC Prescaler Counter** This field contains the current value of the 20-bit counter. Writes have no effect to this register. Reset or writing different values to SC[RTCLKS] and RTC_PS[RTCPS] clear the count to 0x0.

19 Embedded Flash

19.1 Introduction

The embedded flash controller acts as a bridge between the Cortex™-M0+ and the embedded flash memory. In practical application, and the user code is stored in embedded flash, embedded flash boot mode is the main boot mode.

The embedded flash is hereinafter referred to as eflash, which includes P-Flash, D-Flash and information region.

19.2 Features

- P-flash memory size: 32KB, page capacity = 512B
- D-flash memory size: 2KB, page capacity = 8B, block capacity = 16B
- Information memory size: 512B, page capacity = 128B
- Data retention: >100years@25℃, >10years@125℃
- Endurance: P-Flash > 10K, D-Flash > 100K
- P-Flash operation list
  - Read, Page Program, Mass Erase, Page Erase, Page Erase Verify, Mass Erase Verify
- D-Flash operation list
  - Read, Page Program, Mass Erase, Page Erase, Block Erase, Mass Erase Verify
- Information memory operation list
  - Read, Page Program, Page Erase
- Read/Write protection
  - Write protection prevents accidental programming or erasing of memory data
  - Read protection prevents illegal data access
- Supports AEC-Q100 grade 1

19.3 Block diagram

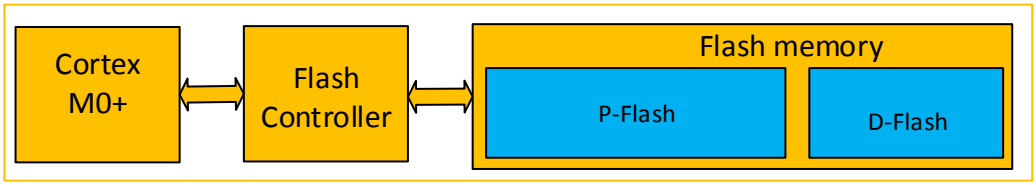

Figure 19-1 Block diagram for eflash and eflash controller

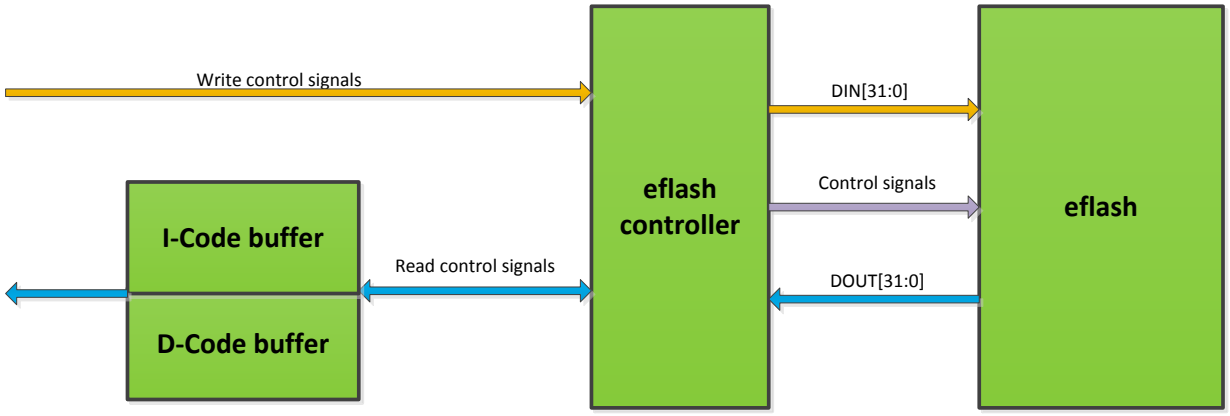

Figure 19-2 Data flow for eflash and eflash controller

19.4 Functional description

19.4.1 Embedded flash memory organization

Before introducing the commands, user should learn about the embedded flash memory organization in Table 19-1. The whole embedded flash memory is composed of three parts: P-Flash memory, D-Flash memory and information memory.

- P-Flash memory is used for storing the user code and data, and user code can put erase, program, verify and read command operations on P-Flash.
- The information memory is also called Option byte section, which is used to store the reading and writing protection information of the whole P-Flash and D-Flash. For the first 8 bytes of the 2nd page, it is used to save user special data(user data in section 19.4.2.2). For Option byte section, user can erase, program and read.

Table 19-1 Embedded Flash memory organization

Flash memory	Name	Address	Size (bytes)	User permission
P-Flash (32KB)	Page 0	0x0800 0000 ~ 0x0800 01FF	512	
	Page 1	0x0800 0200 ~ 0x0800 03FF	512	

	Page 2	0x0800 0400 ~ 0x0800 05FF	512	Page erase/Mass erase Page program/ Read/ Page erase verify/Mass erase verify
	Page 3	0x0800 0600 ~ 0x0800 07FF	512	
	…	…	…	
	Page 63	0x0800 7E00 ~ 0x0800 7FFF	512	
D-Flash (2KB)	Page 0	0x0802 0000 ~ 0x0802 0007	8	Page erase/Block erase/ Mass erase Page program/ Read/ Mass erase verify
	Page 1	0x0802 0008 ~ 0x0802 000F	8	
	Page 2	0x0802 0010 ~ 0x0802 0017	8	
	Page 3	0x0802 0018 ~ 0x0802 001F	8	
	…	…	…	
	Page 254	0x0802 07F0 ~ 0x0802 07F7	8	
	Page 255	0x0802 07F8 ~ 0x0802 07FF	8	
Option byte (512B)	Page 0	0x0804 0000 ~ 0x0804 007F	128	Option byte erase Option byte program Read
	Page 1	0x0804 0080 ~ 0x0804 00FF	128	
	Page 2	0x0804 0100 ~ 0x0804 017F	128	
	Page 3	0x0804 0180 ~ 0x0804 01FF	128	

19.4.2 Embedded flash protection

The main contents saved in the option byte page are read protection, write protection, etc. In order to avoid illegal access to eflash, the controller has the function of protecting the writing and reading of the P-Flash and the D-Flash memory. The related information is stored in the following option bytes and writing protection information is also loaded to the register EFLASH_PWPR0 ~ EFLASH_PWPR1_and EFLASH_DWPR**.** After modifying the content in the option byte, it will take effect after reset or power-on again. Specially, the complement code(such as nRDP / nP_WPRT_EN / nD_WPRT_EN / nDATAx) are implemented by the hardware automatically.

When the write protection is effective, the erase operations(page erase/mass erase) and the program operations on eflash sections with the write protection page are not supported, but the normal reading of eflash data is not affected.

When the disable write protection takes effect, the corresponding eflash section can be erased and programmed correctly.

After the read protection takes effect, the eflash main memory section data(0x0 while reading) cannot be correctly read through JTAG / SWD. When the disable read protection takes effect, data in eflash main memory section data will be erased.

Note:
Both enable and disable read-write protection take effect through the programming option byte address and after reset. Refer to the following for detailed settings.

Table 19-2 Content list in option byte page

Page	Address	[31：24]	[23：16]	[15：8]	[7：0]	Default value	Description

Page0	0x0804 0000	0xFF	nRDP	0xFF	RDP	0xFF53FFAC	P-Flash and D-Flash read protection
Page1	0x0804 0084	nP_WPRT_EN[15：0]		P_WPRT_EN[15：0]		0xFFFFFFFF	P-Flash write protection information
	0x0804 0088	nP_WPRT_EN[31：16]		P_WPRT_EN[31：16]		0xFFFFFFFF	
	0x0804 008C	nP_WPRT_EN[47：32]		P_WPRT_EN[47：32]		0xFFFFFFFF	
	0x0804 0090	nP_WPRT_EN[63：48]		P_WPRT_EN[63：48]		0xFFFFFFFF	
Page2	0x0804 0100	nDATA0		DATA0		0xFFFFFFFF	user data
	0x0804 0104	nDATA1		DATA1		0xFFFFFFFF	
Page3	0x0804 0184	nD_WPRT_EN[15：0]		D_WPRT_EN[15：0]		0xFFFFFFFF	D-Flash write protection information
	0x0804 0188	nD_WPRT_EN[31：16]		D_WPRT_EN[31：16]		0xFFFFFFFF	

19.4.2.1 Read and write protection

As shown in Table 19-2, it can be seen that the option byte address for read protection of P-Flash memory and D-Flash memory ranges from 0x0804 0000 to 0x0804 0003. The option byte address for write protection of P-Flash memory ranges from 0x0804 0084 to 0x0804 0093. The option byte address for write protection of D-Flash memory ranges from 0x0804 0184 to 0x0804 018B. The read/write protection settings are shown in Table 19-3 and Table 19-4.

Table 19-3 Read protection setting

Conditions	RDP	nRDP	Read protection status
Case1	0xFF	0xFF	Protected
Case2	0xAC	0x53	Not protected(at default)
Other cases	Values except case1 and case2		Protected

Table 19-4 Write protection setting

Conditions	P_WPRT_EN[x]/ D_WPRT_EN[x]	nP_WPRT_EN[x]/ nD_WPRT_EN[x]	Write protection status
Case1	1	0	Protected
Other cases	Except case1		Not protected(at default)

Note:
In the P-Flash write protection value, one bit corresponds to one page. In the D-Flash write protection value, one bit corresponds to 8 pages, that is, 64 bytes.

19.4.2.2 User data

As can be seen in Table 19-2, the option byte address of user data DATAx / nDATAx ranges 0x0804 0100 ~ 0x0804 0107. Among them, the lower 16 bits DATAx is used to store data freely, and the higher 16 bits nDATAx is the user data complement code, which is automatically calculated by hardware.

19.4.3 Flash command ID

Table 19-5 Flash command ID

Operation	Command	Command ID	Description
Erase	Page erase	0x1	Page erase P-Flash or D-Flash
	Option byte erase	0x2	Page erase option byte section
	Mass erase	0x3	Erase main memory of P-Flash or D–Flash
	Block erase	0x4	Block erase main memory of D-Flash
Program	Page program	0x5	Program P-Flash or D-Flash based on 32-bit aligned
	Option byte program	0x6	Program option byte section based on 32-bit aligned
Verify	Page erase verify	0x7	Verify whether the P-Flash main memory section based on 32-bit aligned is erased
	Mass erase verify	0x8	Verify whether the entire P-Flash or D-Flash main memory section is erased

19.5 Application note

In this section, the contents about Page erase, Mass erase, Page program, Option byte page erase, Option byte page program, Page erase verify and Mass erase verify will be described by flow chart. All the command operation mainly refers to the following registers: EFLASH_GCR, EFLASH_CCR, EFLASH_CAR , EFLASH_CDR , EFLASH_CSR . For read operation, user can directly read the desired address needed for eflash according to the 8 bit/16 bit/32 bit access mode, so the related description will be ignored in the document.

In particular, the following flow charts just illustrate the single command operation. You only need to unlock once before multiple command operations. For embedded flash memory, there are 64 pages P-Flash memory, 256 pages D-Flash memory, and 4 pages Option byte.

The following Part focuses on the flow of page erase, page erase verify and page program. For other command operations, refer to these flows.

19.5.1 Page erase

Page erase operation just acts on P-Flash memory and D-Flash memory in eflash. Page erase operation can't act on the option byte section. In the following paragraphs, detailed descriptions are for page erase flow is introduced in detail, and for other command operations, refer to it.

1. Before configuring the EFLASH_GCR and EFLASH_CCR,, we must check whether the controller is locked by reading the status of LOCK. If the controller is in lock state, we must sequentially write 0xac7802 and 0x01234567 to EFlASH_UKR_register to unlock. If the controller is not in lock state, go to the next step directly.
2. After unlocking, check whether there are some command operations in process by reading out the status of CMDBSY. CMDBSY equal to 1 represents some command operations is not finished and we must wait until CMDBSY is equal to 0. In fact, the unlock process can be done before or after checking CMDBSY and the following chart just illustrate the "before" case.
3. When CMDBSY turns to 0, we can configure the two registers: EFLASH_GCR and EFLASH_CCR. For all the erase and program command operations, user must adjust the FREQ value in EFLASH_GCR, the configuration formula is: FREQ=eflash controller clock frequency/1. For example, if the eflash controller clock frequency is 32MHZ, FREQ=32/1=32=0x20, but it is recommended that the value is configured larger than 0x20, such as 0x21.
4. Configure the page erase start address in EFLASH_CAR, which value is the start address of the desired erase page. The address value is used to distinguish whether to erase the P-Flash memory or the D-Flash memory, and the address value must be 4-byte aligned.
5. After the basic configuration above, user can launch the command and trigger it by controlling the EFLASH_CCR. It should be guaranteed that bit START must turn from 0 to 1 after other bits have been configured in EFLASH_CCR in order to get a valid trigger. CMD is configured to 0x1 to determine the incoming command operation is page erase. Other bits should be 0 and will be used in other command operations.
6. After a valid trigger, the command operation starts. User should check whether the command operation is finished by reading out the bit CMDBSY. When CMDBSY is equal to 0, it represents that the command operation has been finished.
7. Clear EFLASH_CCR.
8. Then user can read out other status to check whether there are some errors based on user's requirement, such as PGMERR, especially for write protection case, user can not erase the page with write protection.

The page erase flow is shown as the following flow chart. To point out, the process of other command operations is similar to the page erase, just with some difference, which will be described later.

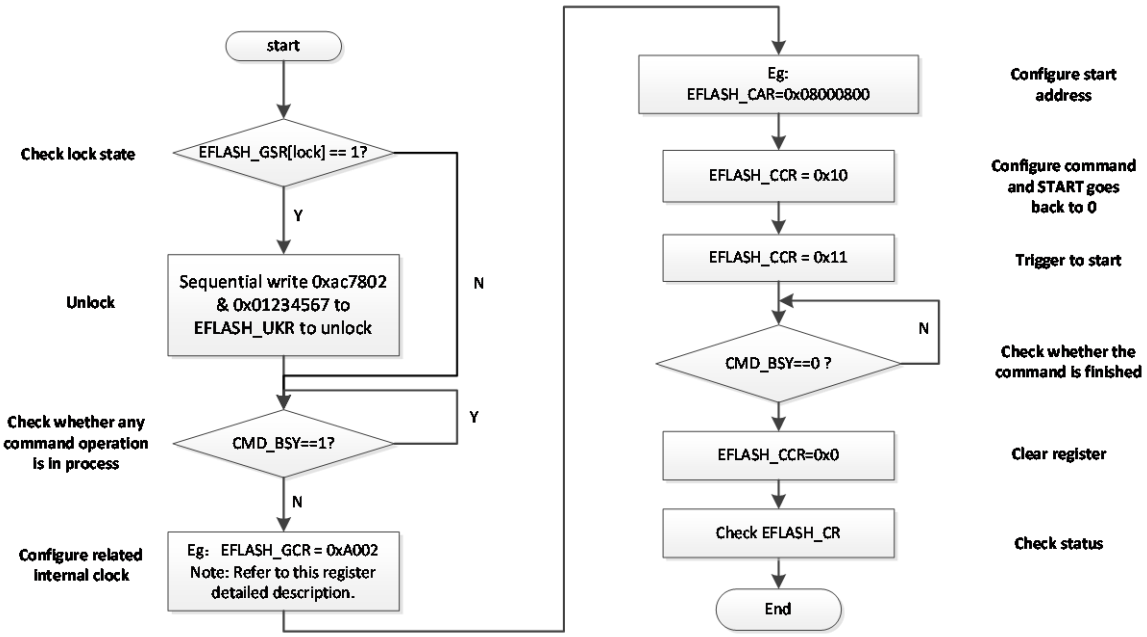

Figure 19-3 Page erase command operation flow

19.5.2 Block erase

Block erase can be used for D-Flash memory. The flow is illustrated in Figure 19-4.

Compared with the page erase command flow, the block erase process has two differences: one is that the CMD & SEL bit of the EFLASH_CCR register is set to a different value, and the other is that the mass erase can only be used for D-Flash memory.

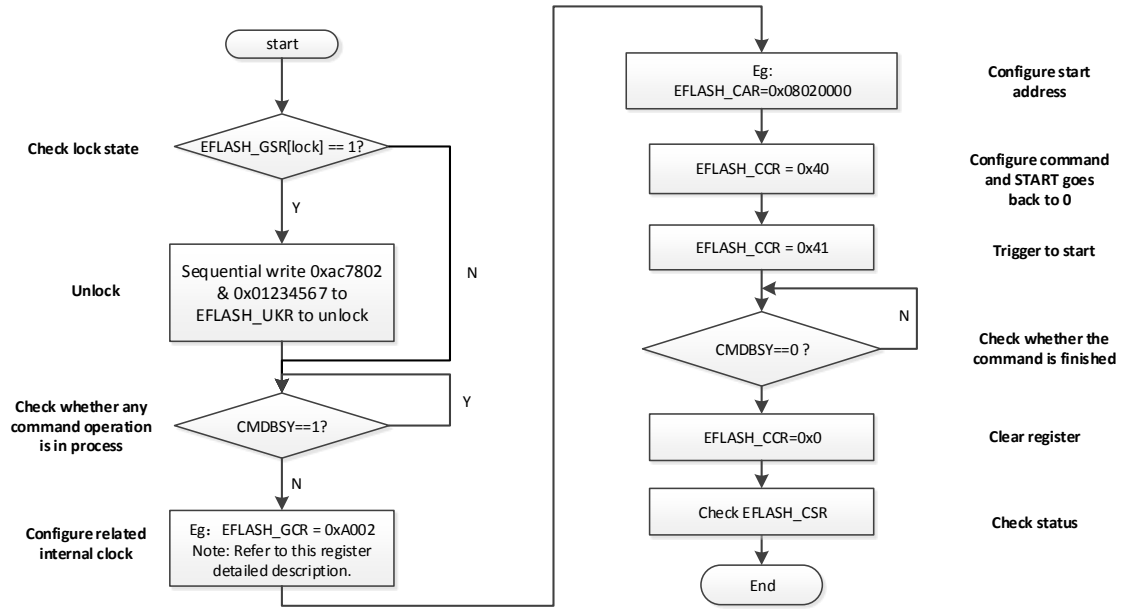

Figure 19-4 Block erase command operation flow

19.5.3 Mass erase

Mass erase can erase the whole P-Flash memory or D-Flash memory. The flow is illustrated in Figure 19-5. Compared with the page erase command flow, the mass erase process has two differences: one is that the CMD & SEL bit of the EFLASH_CCR register is set to a different value, SEL is used to distinguish to erase the P-Flash memory or D-Flash memory. and the other is that the erase address does not need to be specified in the EFLASH_CAR register.

Specially, when the main memory attribute is changed from read-protect to non-read-protect, a mass erase will be performed automatically in order to protect the user code from illegal reading. For example, when user program the RDP in option byte page from the default 0xFF to 0xAC, a mass erase performs automatically.

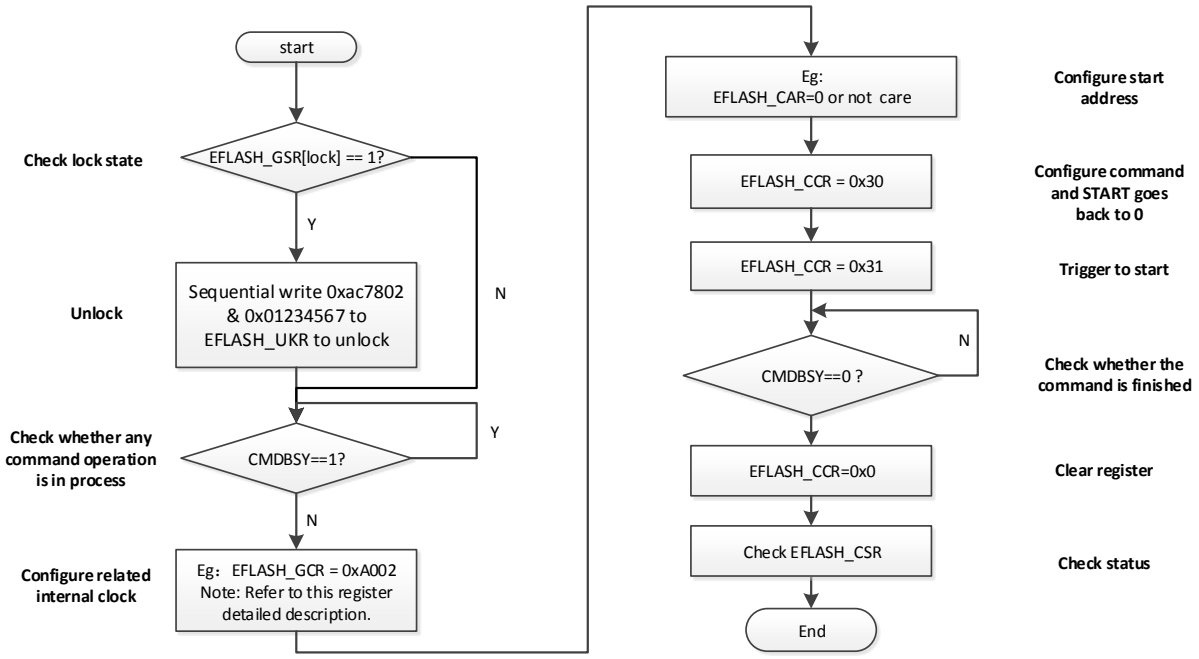

Figure 19-5 Mass erase command operation flow

19.5.4 Page program

Page program command can be used in the P-Flash memory and the D-Flash memory in eflash. The flow is illustrated in Figure 19-6. The page program command flow is also similar to the page erase, but there are two differences: one is CMD bits settings are different. and the other is that programming length and programming data need to be configured. The detailed steps are listed as follows.

1. Configure the programming address value to the EFLASH_CAR register;
2. Configure the LEN [9:0] bits in the EFLASH_CCR register to configure the word length written by the page program command;
3. Configure EFLASH_CCR[CMD] bits to 0x5.

4. Configure EFLASH_CCR[START] bit as 1, that is, start the program command.
5. When the software checks that the EFLASH_CSR[DBUFRDY] is equal to 1, write the data to be programmed into the EFLASH_CDR register until the length specified by LEN [9:0] is written.

Note:

1. The order of step 1, 2, and 3 can be changed.
2. The bits LEN[9:0] are configured to a specified value to determine the programming length by word. For example, if LEN[9:0] are configured to 150, user should write not more than 150 words. If the user write more than 150 words, such as 180 words, the last 30 words will not be written into eflash in fact. Meanwhile, if user writes less than 150 words, such as 110 words, the write operation will be validated normally, but user should keep in mind that program process will be not finished until the data number equal to length or you should use abort command to force program be terminated.

About page program command, there are some auto pre-check mechanisms to do content protection, such as write protection check, blank content check and address boundary check. So user should check the EFLASH_CSR register carefully after each program command to confirm whether the write operation is successful.

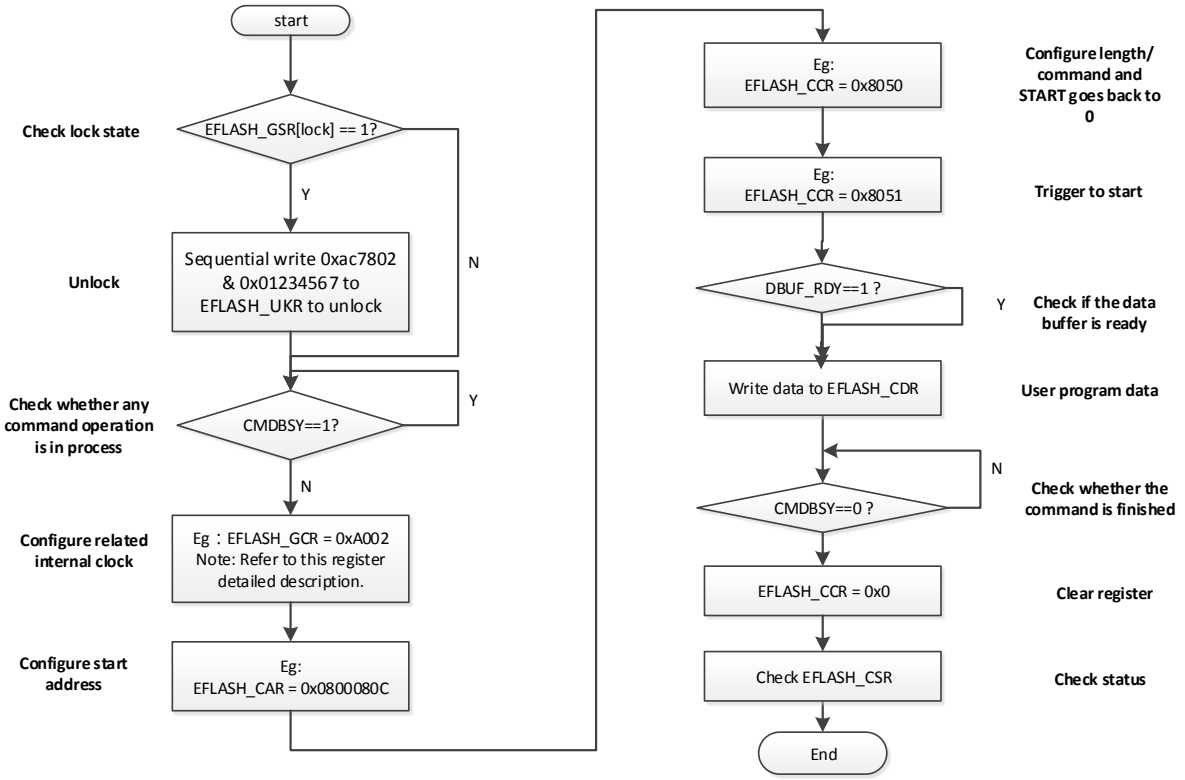

Figure 19-6 Page program command operation flow

19.5.5 Page erase verify

Page erase verify command can be used in the P-Flash memory of the eflash. This operation is usually executed after erase operation in order to verify whether the erase operation is performed successfully. The flow is illustrated in Figure 19-7. Compared with the page erase command flow, the page erase verify flow has two differences: one is that the CMD bits settings in the EFLASH_CCR register are different, and the other is that page erase can be used for D-flash memory, but page erase verify cannot.

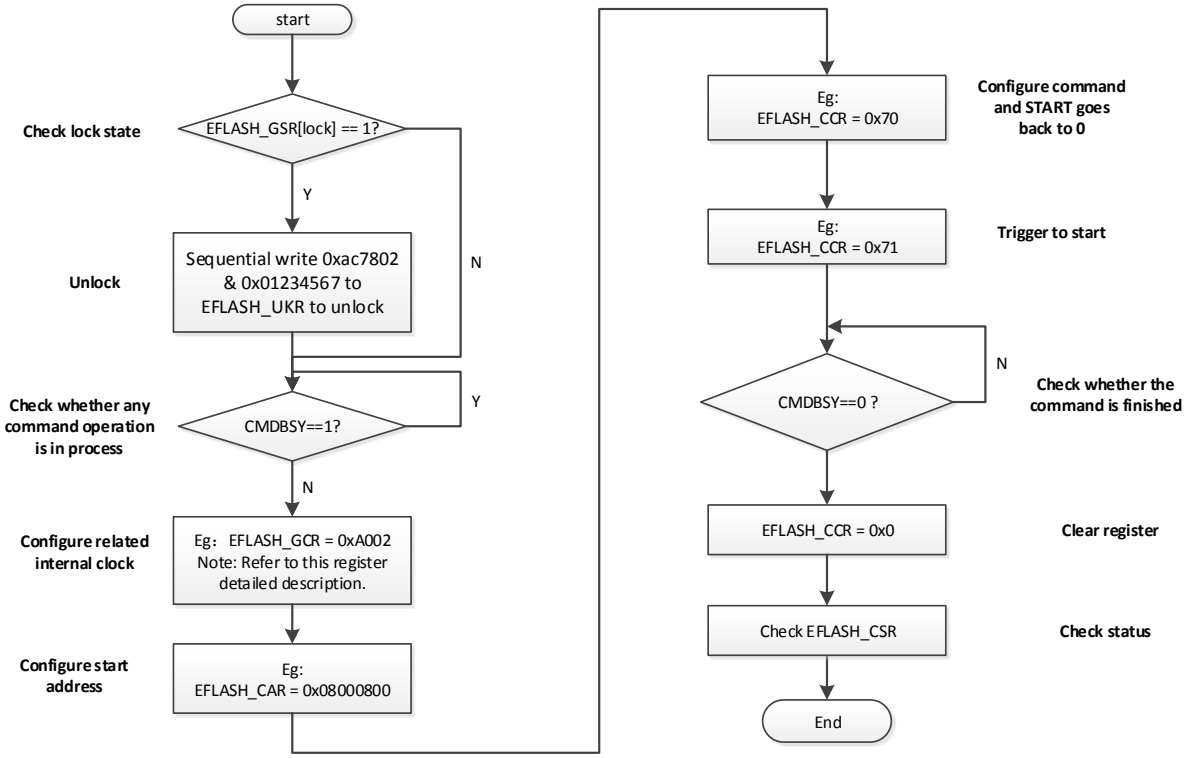

Figure 19-7 Page erase verify command operation flow

19.5.6 Mass erase verify

Mass erase verify command can be used in the P-Flash memory and the D-Flash memory of the eflash. This operation is usually executed after mass erase operation in order to verify whether the mass erase operation is finished successfully. The flow is illustrated in Figure 19-8. Compared with the page erase command flow, the mass erase verify flow has two differences: one is that the CMD bits settings in the EFLASH_CCR register are different, and the other is that it does not need to specifies the erase address in the EFLASH_CAR register. Since the entire memory pages have been reached, there is no need to specify erase address in the EFLASH_CAR register.

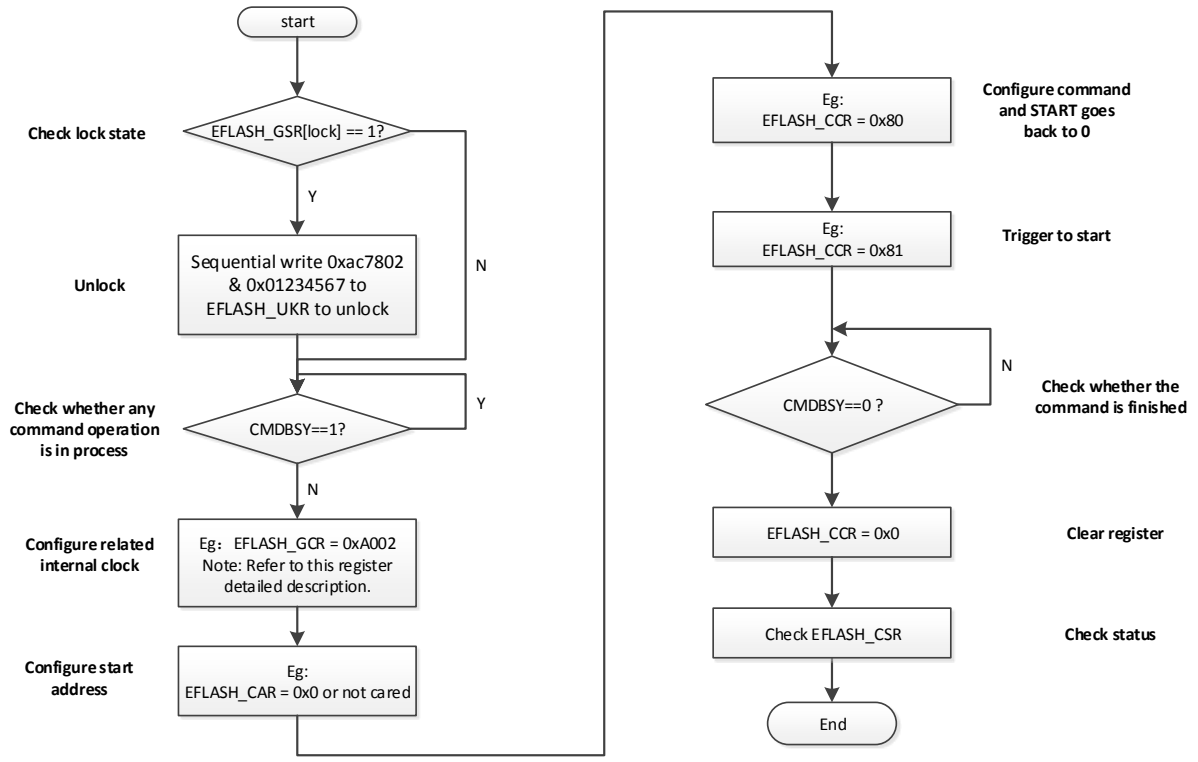

Figure 19-8 Mass erase verify command operation flow

19.5.7 Option byte erase

Option byte erase command can be used for the whole Option byte section. The flow is illustrated in Figure 19-9. Compared with the page erase command flow, there are two differences in the option byte erase flow: one is that the CMD & OPTEN bits settings in the EFLASH_CCR register are different, the other is that the address which specified in the EFLASH_CAR register is the option byte address.

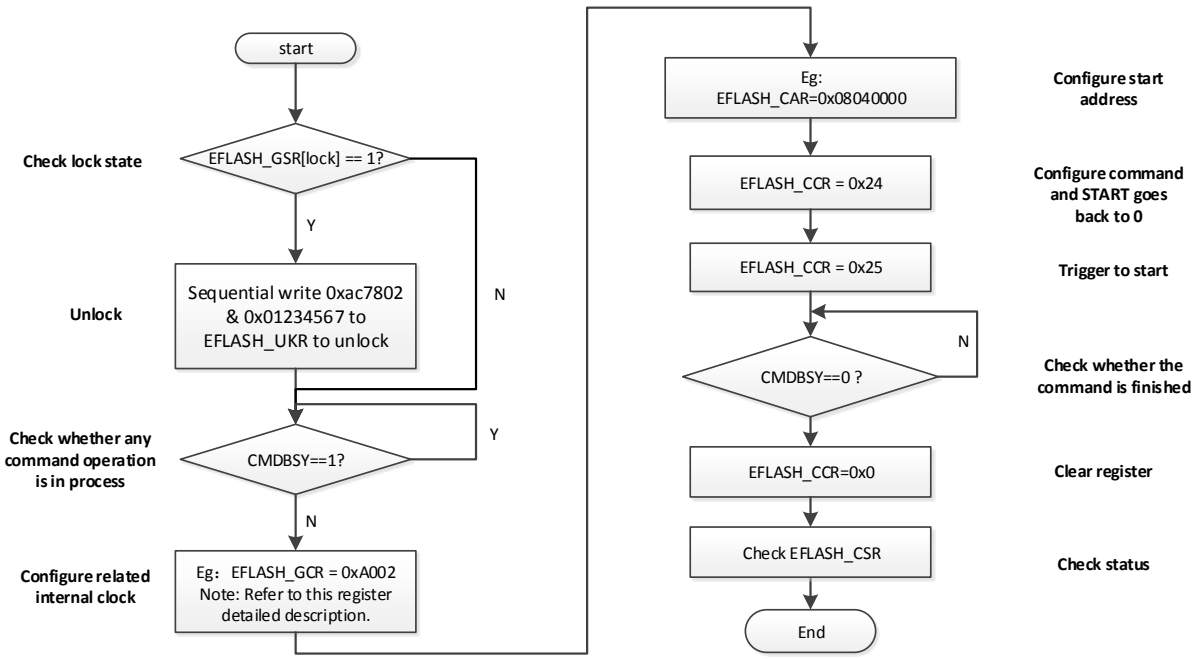

Figure 19-9 Option byte erase command operation flow

19.5.8 Option byte program

Option byte program command can be used for the whole Option byte section. The flow is illustrated in Figure 19-10. Compared with page program command flow, there are two differences in option byte program flow: one is the CMD bits settings in the EFLASH_CCR register are different, and the other is the address which specified in the EFLASH_CAR register is the option byte address.

User should pay high attention to read protection character change while executing option byte program. If you change read protect character from protect to un-protection, eflash controller will execute mass erase command automatically to erase the content of whole user pages.

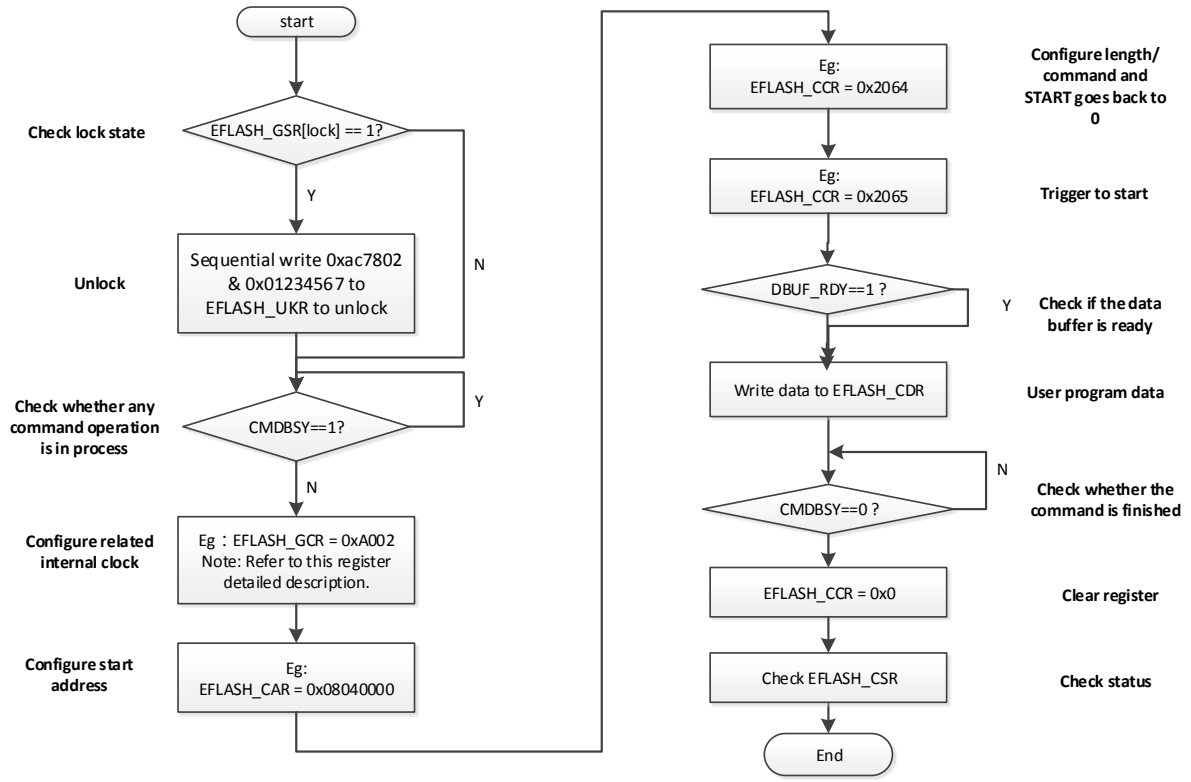

Figure 19-10 Option byte program command operation flow

19.6 Register definition

Table 19-6 Embedded flash register map

EFLASH base address: 0x40002000

Address	Register name	Width	Description
EFLASH base address +0x0	EFLASH_UKR	32	Unlock sequence register
EFLASH base address +0x4	EFLASH_GSR	32	Global status register
EFLASH base address +0x8	EFLASH_GCR	32	Global control register
EFLASH base address +0xC	EFLASH_CSR	32	Command status register
EFLASH base address +0x10	EFLASH_CCR	32	Command control register
EFLASH base address +0x14	EFLASH_CAR	32	Command address register
EFLASH base address +0x18	EFLASH_CDR	32	Command data register
EFLASH base address +0x40	EFLASH_PWPR0	32	P-Flash write protection register 0
EFLASH base address +0x44	EFLASH_PWPR1	32	P-Flash write protection register1
EFLASH base address +0x48	EFLASH_DWPR	32	D-Flash write protection register

19.6.1 Unlock Sequence Register(EFLASH_UKR)

Table 19-7 EFLASH_UKR register

EFLASH_UKR **Unlock Sequence Register** **RESET: 0x00000000**

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

Field	Description
31:0 KEY	**Unlock the write protection of the eflash control registers** The eflash control registers are indicated to be locked when EFLASH_GSR[LOCK] bit is 1. Sequential write 0xac7802 and 0x01234567 to unlock the protection.

19.6.2 Global Status Register(EFLASH_GSR)

Table 19-8 EFLASH_GSR register

EFLASH_GSR **Global Status Register** **RESET: 0x00000001**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name								RPRT				BUSY				LOCK
Type								RO				RO				RW
Reset								0				0				1

Field	Description
8 RPRT	**P-Flash and D-Flash read protection status** 0: read protection disabled. 1: read protection enabled. **Note: the reset value of this bit depends on the current read protection enable state.**
4 BUSY	**Indicate whether the EFLASH controller is in busy state** 0: not busy 1: busy

Field	Description
0 LOCK	**Embedded Flash controller lock register** 0: Embedded Flash controller register does not lock, then write 1 to lock Embedded Flash controller directly. 1: Embedded Flash controller register locks, it cannot be unlocked by writing 0 and can only be unlocked by operating the EFLASH_UKR register.

19.6.3 Global Control Register(EFLASH_GCR)

Table 19-9 EFLASH_GCR register

EFLASH_GCR **Global Control Register** **RESET: 0x00006002**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	FREQ_LOCK	RD_MODE	FREQ													HDFEN
Type	RW	RW	RW													RW
Reset	0	0	0x20													0

Field	Description
15 FREQ_LOCK	**Lock the configuration for FREQ and RD_MOD** 0: not lock 1: lock **Note: if this bit is written to 1 after power on, it can't be configured again until the next power down/power on.**
14 RD_MOD	**Configure whether the read/write speed is divided by 2** 0: eflash controller frequency is not divided by 2 1: eflash controller frequency is divided by 2
13:8 FREQ	**Clock divisor for generating 1us pulse** Configure a reasonable value based on the speed of eflash controller to get 1us period before the following operation: Page program, Page erase, Mass erase based on the command operation flow. For example, if eflash controller clock frequency is 32MHz, CKDIV= 32/1 = 32 = 0x20, but it is recommended that the value is configured larger than 0x20, such as 0x21.
0 HDFEN	**Generate hardfault control or not when the read protection rule is violated** 0: disable hardfault generation 1: enable hardfault generation

19.6.4 Command Status Register(EFLASH_CSR)

Table 19-10 EFLASH_CSR register

EFLASH_CSR **Command Status Register** **RESET: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name				OPTDATERR	OPTDWERR		OPTPWERR	OPTRERR								
Type				RO	RO		RO	RO								
Reset				0x0	0x0		0x0	0x0								
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name							VRFERR	ERSERR	PGMERR	DRVIO	DWVIO	PRVIO	PWVIO	DBUFLST	DBUFRDY	CMDBUSY
Type							RO	RO	RO	RW	RW	RW	RW	RO	RO	RO
Reset							0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0

Field	Description
28 OPTDATERR	**Option byte DATAx verify error** 0: no error 1: error exists
27 OPTDWERR	**Option byte D_WPRT_EN verify error** 0: no error 1: error exists
25 OPTPWERR	**Option byte P_WPRT_EN verify error** 0: no error 1: error exists
24 OPTRERR	**Option byte RDP verify error** 0: no error 1: error exists
9 VRFERR	**Indicate whether there exists error in the verify command operation.** 0: data verify is ok 1: data verify is not ok
8 ERSERR	**Indicate whether there exists error in the erase command operation.** 0: no error 1: error occurs, because program address in EFLASH_CAR is illegal
7 PGMERR	**Indicate whether there exists error in the program command operation.** 0: no error

Field	Description
	1: error occurs, because program address is illegal or verify operation before program command operation is not OK.
6 DRVIO	**Indicate whether the D-Flash operation violates the read protection rules** 0: no violation for the read protection rules 1: violation for the read protection rules Write 1 to clear DRVIO to 0.
5 DWVIO	**Indicate whether the D-Flash operation violates the write protection rules** 0: no violation for the write protection rules 1: violation for the write protection rules Write 1 to clear DWVIO to 0.
4 PRVIO	**Indicate whether the P-Flash operation violates the read protection rules** 0: no violation for the read protection rules 1: violation for the read protection rules Write 1 to clear PRVIO to 0.
3 PWVIO	**Indicate whether the P-Flash operation violates the write protection rules** 0: no violation for the write protection rules 1: violation for the write protection rules Write 1 to clear PWVIO to 0.
2 DBUFLST	**Indicate whether the command operation reaches the end of the data buffer** 0: no 1: yes
1 DBUFRDY	**Indicate whether data buffer is ready** 0: DAT_BUF write is not ready 1: DAT_BUF write is ready
0 CMDBSY	**Indicate whether any of the command operation is in process** 0: all operations are not in process 1: at least one operation is in process

19.6.5 Command Control Register(EFLASH_CCR)

Table 19-11 EFLASH_CCR register

EFLASH_CCR **Command Control Register** **RESET: 0x00000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name									LEN							
Type									RW							
Reset									0x0							
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	LEN								CMD				SEL	OPTEN	ABORT	START
Type	RW								RW				RW	RW	RW	RW
Reset	0x0								0x0				0x0	0x0	0x0	0x0

Field	Description
23: 8 LEN	**Length for Program operation** Program length of the page program, option byte program and page erase verify command. **Note: the unit is word(4 bytes), and it cannot exceed the memory boundary at the maximum.**
7:4 CMD	**FLASH command type selection** 0x0: Idle 0x1: Page Erase 0x2: Option byte Erase 0x3: Mass Erase 0x4: Block Erase 0x5: Page Program 0x6: Option byte Program 0x7: Page Erase Verify 0x8: Mass Erase Verify Others: Reserved **Note: Please refer to the configuration of OPTEN for the validity of the CMD value.**
3 SEL	**Command operation section selection** 0: P-Flash memory 1: D-Flash memory **Note: SEL is valid only for mass erase and mass erase verify.**
2 OPTEN	**Enable the command operation related with the option byte section** 0: Register CMD values 0x6 or 0x2 are invalid, and the other values are valid. 1: Register CMD values 0x6 or 0x2 are valid, and the other values are invalid. **Note: refer to CMD.**
1 ABORT	**Abort current command control**

Field	Description
	0: Not abort current command 1: Abort current command **Note: Write 1 to abort the current command, and the hardware will be automatically cleared to 0.**
0 START	**Control the start command operation** Write 1 to trigger the start command and clear the error status bits [9:7] of the EFLASH_CSR register.

19.6.6 Command Address Register(EFLASH_CAR)

Table 19-12 EFLASH_CAR register

EFLASH_CAR **Command Address Register** **RESET: 0x00000000**

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

Field	Description
31:0 ADDR	**Start address for commands** (1) Page erase ADDR[31:0] is the absolute address of the eflash address, which is just among this page. For example, if user want to erase the 63th page which ranges from0x0800 7E00 to 0x0800 7FFF, so, user can configure ADDR[31:0] as any value from 0x0800 7E00 to 0x0800 7FFF. (2) Program ADDR[31:0] is the program start address. ADDR[31:0] and LEN[15:0] are together to determine the program address range. (3) Page verify ADDR[31:0] is the first eflash address of this page. (4) Mass erase/verify ADDR[31:0] is not cared. **Note: Before the erase/program/verify command is used, the start address must be set.**

19.6.7 Command Data Register(EFLASH_CDR)

Table 19-13 EFLASH_CDR register

EFLASH_CDR **Command Data Register** **RESET: 0x00000000**

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

Field	Description
31: 0 DATA	**Program command data**

19.6.8 P-Flash Write Protection Register 0(EFLASH_PWPR0)

Table 19-14 EFLASH_PWPR0 register

EFLASH_PWPR0 **P-Flash Write Protection Register 0** **RESET:**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	PWPR0															
Type	RO															
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	PWPR0															
Type	RO															
Reset																

Field	Description
31:0 PWPR0	**Bit31:0 indicates whether there exists the write protection for the corresponding P-Flash page 31~0.** 0: no write protection 1: write protection **Note: The Reset value in the above table will be changed due to the option byte register configuration.**

19.6.9 P-Flash Write Protection Register 1(EFLASH_PWPR1)

Table 19-15 EFLASH_PWPR1 register

EFLASH_PWPR1 **P-flash Write Protection Register 1** **RESET:**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	PWPR1															
Type	RO															
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	PWPR1															
Type	RO															
Reset																

Field	Description
31:0 PWPR1	**Bit31:0 indicates whether there exists the write protection for the corresponding P-Flash page 63~32.** 0: no write protection 1: write protection **Note: The Reset value in the above table will be changed due to the option byte register configuration.**

19.6.10 D-Flash Write Protection Register(EFLASH_DWPR)

Table 19-16 EFLASH_DWPR register

EFLASH_DWPR **D-Flash Write Protection Register** **RESET:**

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

Field	Description
31: 0 DWPR	**Bit31:0 indicates whether there exists the write protection for the corresponding D-Flash 32~0 region, one region contains 8 pages (64 bytes).** 0: no write protection 1: write protection **Note: The Reset value in the above table will be changed due to the option byte register configuration.** **This register divides D-Flash into 32 regions, and D-flash is 2KB, so each region size is 64B , that is , 8 pages, and each bit represents the write protection status of one region.**

20 SRAM ECC

20.1 Introduction

The full name of ECC_SRAM is SRAM Error Correcting Code, which is used for SRAM error detection and correction. When an SRAM error occurs, it will generally not cause the entire SRAM to be unreadable or all errors, but only one or a few bits' error in the entire SRAM. ECC_SRAM uses Hamming code ECC single-bit error correction and two-bit detection algorithms. The calculation speed is very fast. Errors above 1 bit cannot be corrected, and errors above 2 bits are not guaranteed to be detected.

20.2 Features

- Support error detection of 1bit and 2 bits.
  - 1 bit and 2 bits' error status can be detected.
  - Software can read the error address and the error status through the register.
- Support error status correction of 1bit.
  - When a 1bit' error state is detected, the hardware will automatically correct the error, interrupt and reset will not be generated.
  - For 2 bits' error status, the hardware cannot correct errors.
- Support 2 bits' error status interrupt.
  - The software can be configured to generate an interrupt when ECC_SRAM detects a 2 bits' error.
  - Software can configure whether to enable system reset when ECC_SRAM detects a 2 bits' error.
- Error status of more than 2 bits cannot be guaranteed to be detected.

20.3 Functional description

1. ECC_SRAM is enabled by default and cannot be disabled.
2. ECC_SRAM will generate an interrupt if enabled ERR2_IRQEN =1 and a 2 bits' error is detected. When ECC_SRAM interrupt occurs,
   - The interrupt flag register ERR2_STATUS is set to 1.
   - The ECC_SRAM status register ERR_STATUS is set to 1.
   - The ECC_SRAM error address register ERR2_ADDR will fill with the error address automatically.

- It is needed for the software to write 1 to interrupt flag register ERR2_STATUS to clear the flag.

3. The software can query the value of the register ERR_STATUS to check the current ECC_SRAM error status.
   - If ERR_STATUS=2'b00, it indicates that ECC_SRAM has not detected an error.
   - If ERR_STATUS=2'b01, it indicates that ECC_SRAM has detected a 2 bits' error. At this time, the software obtains the address of the current 2 bits' error by reading the register ERR2_ADDR.
   - If ERR_STATUS=2'b10 or 2b'11, it indicates that ECC_SRAM has detected a 1 bit error. At this time, the software obtains the address of the current 1 bit error by reading the register ERR1_ADDR.
   - The ECC_SRAM error status register ERR_STATUS and the error address register ERRx_ADDR (x=1,2) will retain the error status and error address of the last ECC_SRAM check when the system is not reset. It will clear this error status register ERR_STATUS and the error address register ERRx_ADDR (x=1,2) by writing 2'b11 to ERR_STATUS.
   - The system will generate a reset if programed the 23th bit (ECC2_RST_EN) of the register RESET_CRTL (address is 0x4000000C) to be 1 and the ECC_SRAM detects a 2 bits' error.

20.4 Register definition

Table 20-1 SRAM ECC register map

ECC_SRAM base address: 0x40000020

Address	Register name	Width	Description
ECC_SRAM base address +0x0	ECC_SRAM_CTRL	32	ECC control and status register
ECC_SRAM base address +0x4	ECC_SRAM_ERR1_ADDR	32	ECC 1 bit error address register
ECC_SRAM base address+0x8	ECC_SRAM_ERR2_ADDR	32	ECC 2 bits error address register

20.4.1 ECC Control and Status Register(ECC_SRAM_CTRL)

Table 20-2 ECC_SRAM_CTRL register

ECC_SRAM_CTRL **ECC_SRAM ctrl and status register** **RESET: 0x00000001**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name											ERR_STATUS			ERR2_STATUS	ERR2_IRQEN	
Type											R/W1C			R/W1C	RW	
Reset											0			0	0	

Bits	**Description**
5: 4 ERR_STATUS	**ECC_SRAM Error Status** 00: no error 01: 2 bits error and cannot be corrected 10/11: 1 bit error and can be corrected **Note: write 2'b11 to clear these two status bits and ERRx_ADDR(x=1,2)**
2 ERR2_STATUS	**ECC_SRAM 2 bits' error status** **Note: write 1 to clear this bit.** **If ERR2_IRQEN is enabled and 2 bits' error is generated, it must write 1 to clear this bit, otherwise this module will always generate 2 bits' error interrupt.**
1 ERR2_IRQEN	**ECC 2 bits' error interrupt enable** 0: disable 1: enable

20.4.2 ECC 1 Bit Error Address Register(ECC_SRAM_ERR1_ADDR)

Table 20-3 ECC_SRAM_ERR1_ADDR register

ECC_SRAM_ ERR1_ADDR ECC_SRAM 1bit error address register RESET: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name				ERR1_ADDR [12: 0]												
Type				RO												
Reset				0												

Bits	Description
12: 0 ERR1_ADDR	**1 bit error address** **Note: multiply ERR1_ADDR by 4 and then plus 0x20000000 to get the SRAM address of occurring 1 bit error.**

20.4.3 ECC 2 Bits Error Address Register(ECC_SRAM_ERR2_ADDR)

Table 20-4 ECC_SRAM_ERR2_ADDR register

ECC_SRAM_ERR2_ADDR ECC_SRAM 2 bits error address register RESET: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name																
Type																
Reset																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name				ERR2_ADDR [12: 0]												
Type				RO												
Reset				0												

Bits	Description
12: 0 ERR2_ADDR	**2 bits' error address** **Note: multiply ERR2_ADDR by 4 and then plus 0x20000000 to get the SRAM address of occurring 2 bits' error.**

21 Debug

21.1 Introduction

This debug of the device is based on the ARM CoreSight architecture. External debugger accesses registers and memory, control program run/stop/reset operations through the debug interface.

This device supports only one debug interface, that is, Serial Wire Debug (SWD).

21.2 Features

- 4 hardware breakpoints.
- 2 data watchpoints.
- SWD interface accessibility.