



DeepL

Subscribe to DeepL Pro to translate larger documents.

Visit www.DeepL.com/pro for more information.



HC32F460_F45x_A460 Series

32-bit ARM® Cortex®-M4

microcontrollers

**Referenc
e Manual**

Rev1.5 September 2022



Sound Ming

(“XHSC”) reserves the right to change, correct, enhance, or modify XHSC products and/or this document at any time without notice. XHSC products are sold under the terms and conditions of sale set forth in the basic purchase and sale contract.

- ★ Customer shall select the appropriate XHSC product for your application and design, validate and test your application to ensure that your application meets the appropriate standards and any safety, security or other requirements. The customer shall be solely responsible for this.
- ★ XHSC hereby acknowledges that no license to any intellectual property is granted, either expressly or impliedly.
- ★ Resale of XHSC products on terms different from those specified herein shall void any warranty commitment by XHSC with respect to such products.
- ★ Any graphic or word with the “®” or “™” logo is a trademark of XHSC. All other product or service names displayed on XHSC products are the property of their respective owners.
- ★ The information in this notice supersedes and replaces the information in the previous version.

©2022 Xiaohua Semiconductor Co. All rights reserved.

Table of Contents

Sound	clear
2	
Table of Contents	Table of Contents
3	
Table Index.....	32
Figure Index	36
Introduction (Overview)	44
1 Memory Mapping	45
1.1 Memory Mapping	45
1.2 External Space Mapping.....	49
1.3 Bit space.....	49
1.4 Address Remapping.....	49
1.5 Remap Register	51
1.5.1 Access Protection Register (MMF_REMPRT)	52
1.5.2 Remap control register (MMF_REMCRx)(x=0, 1).....	53
2 Bus Architecture (BUS)	54
2.1 Overview	54
2.2 Bus Architecture.....	55
2.3 Bus functions.....	56
3 Reset Control (RMU)	57
3.1 Brief Introduction.....	57
3.2 Reset mode and reset flag bit	58
3.3 Reset Timing.....	60
3.3.1 Power on reset.....	60
3.3.2 NRST pin reset.....	60
3.3.3 Undervoltage reset.....	61
3.3.4 Programmable voltage detection 1 reset, programmable voltage detection 2 reset.	62
3.3.5 Watchdog reset, dedicated watchdog reset.....	64
3.3.6 Power-down wake-up reset.....	64
3.3.7 Software Reset.....	65
3.3.8 MPU Error Reset.....	65
3.3.9 RAM Parity Reset.....	65
3.3.10 RAMECC Reset.....	66

3.3.11	Clock frequency abnormal reset	66
3.3.12	External high-speed oscillator abnormal stop reset	67
3.3.13	Determination of reset mode	67
3.3.14	Reset conditions for each module.....	68
3.4	Register Description	69
3.4.1	Reset flag register 0 (RMU_RSTF0).....	69
4	Clock Controller (CMU)	71
4.1	Brief Introduction.....	71
4.2	System Block Diagram.....	72
4.2.1	System Block Diagram	72
4.2.2	Clock frequency measurement block diagram	73
4.3	Clock Source Specifications.....	74
4.4	Operating clock specifications.....	76
4.5	Crystal Circuit.....	78
4.5.1	External high-speed oscillator.....	78
4.5.2	External high-speed oscillator fault detection	80
4.5.3	External low-speed oscillator	82
4.6	Internal RC Clock	83
4.6.1	HRC Clock	83
4.6.2	MRC Clock.....	83
4.6.3	LRC Clock	83
4.6.4	SWDTRC Clock.....	84
4.7	PLL Clock	84
4.8	Clock Switching Procedure	84
4.8.1	Clock source switching	85
4.8.2	Clock division switching	86
4.9	Clock output function.....	87
4.10	Clock frequency measurement.....	88
4.10.1	Clock frequency measurement	88
4.10.2	Digital filtering function.....	89
4.10.3	Interrupt/reset function.....	89
4.11	Register Description	90
4.11.1	CMU XTAL Configuration Register (CMU_XTALCFGREG)	92
4.11.2	CMU XTAL Settling Configuration Register (CMU_XTALSTBCR)	93
4.11.3	CMU XTAL Control Register (CMU_XTALCR)	93

4.11.4	CMU XTAL Oscillation Fault Control Register (CMU_XTALSTDCR)	94
4.11.5	CMU XTAL Oscillation Fault Status Register (CMU_XTALSTDSR)	95
4.11.6	CMU XTAL32 Configuration Register (CMU_XTAL32CFG)	95
4.11.7	CMU XTAL32 Filter Register (CMU_XTAL32NFR)	96
4.11.8	CMU XTAL32 control register (CMU_XTAL32CR)	96
4.11.9	CMU HRC Calibration Register (CMU_HRCTRM)	97
4.11.10	CMU HRC Control Register (CMU_HRCCR)	97
4.11.11	CMU MRC Calibration Register (CMU_MRCTRM)	98
4.11.12	CMU MRC Control Register (CMU_MRCCR)	99
4.11.13	CMU LRC Calibration Register (CMU_LRCTRM)	100
4.11.14	CMU LRC Control Register (CMU_LRCCR)	101
4.11.15	CMU MPLL Configuration Register (CMU_PLLCFG)	102
4.11.16	CMU MPLL control memory (CMU_PLLCR)	104
4.11.17	CMU UPLL Configuration Register (CMU_UPLLCFG)	105
4.11.18	CMU UPLL control memory (CMU_UPLLCR)	107
4.11.19	CMU Clock Source Stabilization Stateer (CMU_OSCSTBSR)	108
4.11.20	CMU System Clock Source Switching Register (CMU_CKSWR)	109
4.11.21	CMU clock division configuration register (CMU_SCFGR)	110
4.11.22	CMU USBFS clock configuration memory (CMU_UFSCKCFG)	112
4.11.23	CMU AD/TRNG clock configuration memory (CMU_PERICKSEL)	113
4.11.24	CMU I ₂ S clock configuration memory (CMU_I2SCKSEL)	114
4.11.25	CMU Debug Clock Configuration Memory (CMU_TPIUCKCFG)	115
4.11.26	CMU MCO1 Configuration Memory (CMU_MCO1CFG)	116
4.11.27	CMU MCO2 Configuration Memory (CMU_MCO2CFG)	117
4.11.28	FCM lower limit comparison value register (FCM_LVR)	118
4.11.29	FCM upper limit comparison value register (FCM_UVR)	118
4.11.30	FCM Counter Value Register (FCM_CNTR)	118
4.11.31	FCM Start Stop Register (FCM_STR)	119
4.11.32	FCM Measurement Object Control Register (FCM_MCCR)	119
4.11.33	FCM Measurement Reference Control Register (FCM_RCCR)	120
4.11.34	FCM Interrupt Reset Control Register (FCM_RIER)	121
4.11.35	FCM flag register (FCM_SR)	121
4.11.36	FCM Flag Bit Clear Register (FCM_CLR)	122
5	Power Control (PWC)	123
5.1	Introduction	123

5.2	Power Distribution.....	123
5.3	Description of Power Supply Voltage Detection Unit (PVD)	125
5.3.1	Power-on reset/power-down reset action description	125
5.3.2	Description of undervoltage reset (BOR)	126
5.3.3	Programmable voltage detection 1 (PVD1), programmable voltage detection 2 (PVD2) 127	
5.3.4	PVD1, PVD2 interrupt/reset block diagram.....	128
5.3.5	Input/output pins.....	128
5.3.6	PVD1 Interrupt and Reset.....	129
5.3.7	PVD2 Interrupt and Reset.....	130
5.3.8	Internal voltage sampling and detection function.....	131
5.4	Action Mode and Low Power Mode.....	132
5.4.1	Operation mode	135
5.4.2	Sleep mode.....	137
5.4.3	Stop mode.....	137
5.4.4	Power down mode	138
5.5	Ways to reduce power consumption	142
5.5.1	Reduce system clock speed	142
5.5.2	Turn off clock sources not in use	142
5.5.3	Function clock stop	142
5.5.4	Turning off RAM that is not in use	143
5.6	Register protection function.....	144
5.7	Register Description.....	145
5.7.1	Power mode control register 0 (PWC_PWRC0)	147
5.7.2	Power mode control register 1 (PWC_PWRC1)	148
5.7.3	Power mode control register 2 (PWC_PWRC2)	149
5.7.4	Power mode control register 3 (PWC_PWRC3)	149
5.7.5	Power-down wake-up enable register 0 (PWC_PDWKE0)	150
5.7.6	Power-down wake-up enable register 1 (PWC_PDWKE1)	151
5.7.7	Power-down wake-up enable register 2 (PWC_PDWKE2)	152
5.7.8	Power-down wake-up event edge selection register (PWC_PDWKES)	153
5.7.9	Power-down wake-up flag register 0 (PWC_PDWKF0)	154
5.7.10	Power-down wake-up flag register 1 (PWC_PDWKF1)	155
5.7.11	Power Supply Monitoring Control Register (PWC_PWCMR).....	155
5.7.12	Mode Switching Control Register (PWC_MDSWCR)	156

5.7.13	Function Clock Control 0 (PWC_FCG0)	157
5.7.14	Function Clock Control 1 (PWC_FCG1)	159
5.7.15	Function Clock Control 2 (PWC_FCG2)	161
5.7.16	Function Clock Control 3 (PWC_FCG3)	163
5.7.17	PWC_FCG0 Protection Control (PWC_FCG0PC)	165
5.7.18	Function Protection Control Register (PWC_FPRC)	165
5.7.19	STOP mode control register (PWC_STPMCR)	166
5.7.20	RAM power control register O (PWC_RAMPC0)	167
5.7.21	RAM operation condition register (PWC_RAMOPM)	169
5.7.22	XTAL32 Controlled with Current Source (PWC_XTAL32CS)	169
5.7.23	Wakeup Timer Control Register (PWC_WKTCR)	170
5.7.24	PVD control register O (PWC_PVDCR0)	170
5.7.25	PVD control register 1 (PWC_PVDCR1)	171
5.7.26	PVD Filter Control Register (PWC_PVDFCR)	172
5.7.27	PVD Level Control Register (PWC_PVDLCSR)	173
5.7.28	PVD Interrupt Control Register (PWC_PVDICR)	174
5.7.29	PVD Detection Status Register (PWC_PVDDSR)	175
6	Initialization Configuration (ICG)	176
6.1	Introduction	176
6.2	Register Description	177
6.2.1	Initialization configuration register O (ICG0)	177
6.2.2	Initialization configuration register 1 (ICG1)	179
6.2.3	Initialize configuration register n (ICGn)n=2~3	180
6.2.4	Initialization configuration register 4 (ICG4)	180
6.2.5	Initialization Configuration Register 5 (ICG5)	180
6.2.6	Initialize configuration register n (ICGn)n=6~7	181
7	Embedded FLASH (EFM)	182
7.1	Brief Introduction	182
7.2	Main Features	182
7.3	Embedded FLASH	183
7.4	Read Interface	185
7.4.1	The relationship between CPU clock and FLASH read time	185
7.4.2	FLASH Low Power Read	185
7.5	FLASH Read Acceleration Cache	188
7.6	FLASH Programming and Erase Operations	189

7.6.1	Single programming without readback mode.....	189
7.6.2	Single Programming Readback Mode	189
7.6.3	Continuous programming operation.....	189
7.6.4	Erase operation.....	190
7.6.5	Data Security Protection	191
7.6.6	Bus hold function	191
7.6.7	FLASH erase, programming window protection.....	192
7.6.8	Interruptions.....	192
7.7	One-time programmable bytes	193
7.8	Guided exchange	194
7.9	Register Description.....	195
7.9.1	Access Protection Register (EFM_FAPRT)	196
7.9.2	FLASH Stop Register (EFM_FSTP)	196
7.9.3	Read mode register (EFM_FRMC)	197
7.9.4	Erase Write Mode Register (EFM_FWMC).....	198
7.9.5	Status register (EFM_FSR).....	199
7.9.6	Status Clear Register (EFM_FSCLR).....	200
7.9.7	Interrupt License Register (EFM_FITE)	201
7.9.8	Bootstrap Switching Status Register (EFM_FSWP)	201
7.9.9	FLASH Window Protection Start Address Register (EFM_FPMTSW).....	202
7.9.10	FLASH End of Window Protection Address Register (EFM_FPMTEW).....	202
7.9.11	UNIQUE ®ister (EFM_UQID1)	202
7.9.12	UNIQUE ®ister (EFM_UQID2)	203
7.9.13	UNIQUE ®ister (EFM_UQID3)	203
7.10	Cautions	204
8	Internal SRAM (SRAM)	205
8.1	Introduction.....	205
8.2	Register Description.....	207
8.2.1	SRAM wait control register (SRAM_WTCR)	207
8.2.2	SRAM wait protection register (SRAM_WTPR).....	209
8.2.3	SRAM checksum control register (SRAM_CKCR).....	210
8.2.4	SRAM checksum protection register (SRAM_CKPR).....	211
8.2.5	SRAM Check Status Register (SRAM_CKSR).....	212
9	General Purpose IO (GPIO)	213
9.1	Introduction.....	213

9.2	Port Function Summary	214
9.3	Action Description.....	215
9.3.1	General Purpose Input/Output GPIO Function.....	215
9.3.2	Peripheral Features	216
9.3.3	Dual Peripheral Function	216
9.3.4	Event Port input and output functions.....	216
9.3.5	External Interrupt EIRQ Input Function.....	217
9.3.6	Simulation function	217
9.3.7	Universal control.....	218
9.4	Register Description.....	219
9.4.1	General Input Register (PIDRx).....	220
9.4.2	General Output Data Register (PODRx)	220
9.4.3	General Output License Register (POERx).....	220
9.4.4	General Purpose Output Reset Register (POSRx)	221
9.4.5	General Output Reset Register (PORRx).....	221
9.4.6	General Output Rollover Register (POTRx)	221
9.4.7	Special Control Register (PSPCR)	222
9.4.8	Public Control Register (PCCR)	222
9.4.9	Input Control Register (PINAER).....	223
9.4.10	Write Protect Register (PWPR)	223
9.4.11	General Control Register (PCRxy)	224
9.4.12	Function Selection Register (PFSRxy)	226
9.4.13	Event Port Direction Selection Register (PEVNTDIRM)	227
9.4.14	Event Port Input Data Register (PEVNTIDRM).....	227
9.4.15	Event Port Output Data Register (PEVNTODRM).....	228
9.4.16	Event Port Output Data Reset Register (PEVNTORRM).....	228
9.4.17	Event Port Output Data Reset Register (PEVNTOSRM).....	229
9.4.18	Event Port Rising Edge Input Permit Register (PEVNTRISRM).....	229
9.4.19	Event Port Falling Edge Input License Register (PEVNTFALRM)	230
9.4.20	Event Port Input Filter Control Register (PEVNTNFCR).....	231
9.4.21	32bit Access.....	232
9.5	Cautions	232
10	Interrupt Controller (INTC)	233
10.1	Introduction.....	233
10.2	INTC System Block Diagram	235

10.2.1 System Block Diagram	235
10.3 vector table.....	236
10.3.1 Interrupt vector table	236
10.3.2 Interrupt Event Request Serial Number	241
10.4 Function Description	255
10.4.1 Non-maskable interrupts.....	255
10.4.2 External Pin Interrupt Event Request	256
10.4.3 Interrupt source selection.....	256
10.4.4 Software Interrupts.....	256
10.4.5 Interrupt/event selection	257
10.4.6 WFE Wakeup Event Management	257
10.4.7 Digital Filters.....	258
10.4.8 Low power mode return.....	259
10.4.9 Internal trigger events.....	259
10.5 Register Description	260
10.5.1 NMI Pin Non-Maskable Interrupt Control Register (INT_NMICR).....	265
10.5.2 Non-maskable interrupt enable register (INT_NMIENR)	266
10.5.3 Non-maskable interrupt flag register (INT_NMIFR)	267
10.5.4 Non-maskable interrupt flag clear register (INT_NMICFR)	268
10.5.5 External pin interrupt control register (INT_EIRQCRx) (x=0~15)	269
10.5.6 External Pin Interrupt Flag Register (INT_EIFR).....	270
10.5.7 External Pin Interrupt Flag Clear Register (EICFR).....	270
10.5.8 Interrupt/event selection registers (INT_SEL0~31)	271
10.5.9 Interrupt selection registers (INT_SEL32~127).....	272
10.5.10 Vector shared interrupt selection registers (INT_VSSEL128~143).....	273
10.5.11 Stop Mode Wakeup Event Enable Register (INT_WUPEN).....	274
10.5.12 Software interrupt/event register (INT_SWIER).....	275
10.5.13 Event enable register (INT_EVTER).....	275
10.5.14 Interrupt Enable Register (INT_IER).....	276
10.6 Notes on use	277
11 Automatic Operating System (AOS).....	278
11.1 Brief Introduction	278
11.1.1 Function Overview	278
11.1.2 Module Schematic	279
11.2 Function Description.....	280

11.2.1 AOS Source Event List	280
11.2.2 AOS Target List	280
11.3 Action Description.....	281
11.3.1 Dedicated trigger source	281
11.3.2 Common trigger source.....	281
11.4 Register Description.....	282
11.4.1 Peripheral trigger event register (INTSFTTRG)	283
11.4.2 DCU trigger source selection register (DCU_TRGSELx) (x=1~4)	283
11.4.3 DMA1 Transmit Start Trigger Source Select Register (DMA1_TRGSELx) (x=0~3)	
284	
11.4.4 DMA2 transmit start trigger source selection register (DMA2_TRGSELx) (x=0~3)	284
11.4.5 DMA channel reset trigger source selection register (DMA_TRGSELRC)	285
11.4.6 Timer6 Hardware Trigger Event Selection Register (TMR6_HTSSRx) (x=0~1)	286
11.4.7 Timer0 Hardware Trigger Event Select Register (TMR0_HTSRR)	287
11.4.8 Event Port Trigger Source Select Register (PEVNTTRGSR12,	
PEVNTTRGSR34)	288
11.4.9 TimerA Internal trigger event selection register 0 (TMRA_HTSRR0)	289
11.4.10 TimerA Internal trigger event selection register 1 (TMRA_HTSRR1)	290
11.4.11 OTS Trigger Source Select Register (OTS_TRG)	290
11.4.12 A/D1 conversion start on-chip trigger source selection register ADC1_ITRGSELRx(x=0,1)	
291	
11.4.13 A/D2 conversion start on-chip trigger source selection register ADC2_ITRGSELRx(x=0,1)	
291	
11.4.14 Common trigger source selection register 1 (AOS_COMTRG1)	292
11.4.15 Common trigger source selection register 2 (AOS_COMTRG2)	292
12 Keypad Scan Control Module (KEYSCAN)	293
12.1 Introduction.....	293
12.2 KEYSAN system block diagram.....	293
12.3 Pin Description.....	294
12.4 Function Description	294
12.4.1 Key recognition function	294
12.4.2 Keyboard scanning function	294
12.4.3 Precautions on use	295
12.5 Register Description.....	296
12.5.1 KEYSAN Scan Control Register (KEYSCAN_SCR)	297

12.5.2 KEYS defense Scan Enable Register (KEYSCAN_SER)	299
12.5.3 KEYS defense Scan Status Register (KEYSCAN_SSR)	299
13 Storage Protection Unit (MPU).....	300
13.1 Brief Introduction	300
13.2 Function Description	301
13.2.1 Regional Range Setting.....	301
13.2.2 Permission setting	301
13.2.3 MPU Action Selection.....	301
13.2.4 Start MPU	301
13.3 Application examples	302
13.3.1 Allow only partial space access.....	302
13.3.2 Block access to only some spaces	302
13.4 Register Description.....	303
13.4.1 Region Range Description Register MPU_RGD _n (n=0 to 15)	304
13.4.2 Area control register MPU_RGCR _n (n=0 to 15).....	305
13.4.3 Control register MPU_CR	306
13.4.4 Status flag register MPU_SR.....	308
13.4.5 Flag Clear Register MPU_ECLR	308
13.4.6 Write protect register MPU_WP	309
13.4.7 IP Access Protection Register MPU_IPPR	310
14 DMA Controller (DMA).....	312
14.1 Introduction.....	312
14.2 Module schematic.....	313
14.3 Function Description	314
14.3.1 Enabling the DMA Controller	314
14.3.2 Channel selection and channel priority.....	314
14.3.3 Start DMA	314
14.3.4 Data Block	314
14.3.5 Transmission address control.....	314
14.3.6 Number of transmissions	315
14.3.7 Interrupt and event signal output.....	315
14.3.8 Chain transmission.....	315
14.3.9 Discontinuous address transmission.....	318
14.3.10 Channel Reset	319
14.3.11 Transmission terminated early	321

14.4 Application examples	322
14.4.1 Memory-to-memory transfer	322
14.4.2 Memory to peripheral circuit transfer	323
14.4.3 Memory-to-memory chain transfer.....	325
14.5 Register Description.....	327
14.5.1 DMA enable register (DMA_EN)	328
14.5.2 Interrupt status register 0(DMA_INTSTAT0).....	328
14.5.3 Interrupt status register 0(DMA_INTSTAT1)	329
14.5.4 Interrupt Mask Register (DMA_INTMASK0).....	329
14.5.5 Interrupt Mask Register (DMA_INTMASK1).....	330
14.5.6 Interrupt reset register (DMA_INTCLR0).....	330
14.5.7 Interrupt reset register (DMA_INTCLR1)	331
14.5.8 Channel enable register (DMA_CHEN)	331
14.5.9 Channel reset control register (DMA_RCFGCTL)	332
14.5.10 Transfer request status register (DMA_REQSTAT)	334
14.5.11 Channel status observation register (DMA_CHSTAT)	335
14.5.12 Transfer source address register (DMA_SARx) (x=0~3).....	335
14.5.13 Transfer destination address register (DMA_DARx) (x=0~3)	336
14.5.14 Data control register (DMA_DTCTLx) (x=0~3)	336
14.5.15 Repeat Region Size Register (DMA_RPTx) (x=0~3).....	337
14.5.16 Repeat area size register B(DMA_RPTBx) (x=0~3)	338
14.5.17 Source device discontinuous address transfer control register (DMA_SNSEQCTLx) (x=0~3)	339
14.5.18 Source device discontinuous address transfer control register B(DMA_SNSEQCTLBx) (x=0~3)	340
14.5.19 Destination device discontinuous address transfer control register (DMA_DNSEQCTLx) (x=0~3).....	341
14.5.20 Destination device discontinuous address transfer control register B(DMA_DNSEQCTLBx) (x=0~3)	342
14.5.21 Chain pointer register (DMA_LLPx) (x=0~3).....	343
14.5.22 Channel control register (DMA_CHxCTL) (x=0~3).....	344
14.5.23 Channel Monitor Register(DMA_MONSARx, DMA_MONDARx, DMA_MONDTCTLx, DMA_MONRPTx, DMA_MONSSEQCTLx, DMA_MONDNSEQCTLx) (x=0~3) 345	
14.6 Notes on use.....	345

15 Voltage Comparator (CMP)	346
15.1 Brief Introduction.....	346
15.2 Function Description.....	348
15.2.1 Voltage Comparison	348
15.2.2 Digital filtering.....	349
15.2.3 Interrupts and events.....	349
15.2.4 Scan comparison mode.....	350
15.2.5 8bit-DAC setting	351
15.3 Cautions.....	351
15.3.1 Module stop function.....	351
15.3.2 Action when the module is stopped	351
15.3.3 Stopping the action in low-power mode.....	351
15.3.4 Action in power-down low-power mode.....	352
15.4 Register Description	353
15.4.1 CMP control register (CMP_CTRL)	354
15.4.2 CMP Voltage Select Register (CMP_VLTSEL)	356
15.4.3 CMP result monitoring register (CMP_OUTMON)	358
15.4.4 CMP Stabilization Time Register (CMP_CVSSTB).....	359
15.4.5 CMP Compare Voltage Sweep Period Register (CMP_CVSPRD)	359
15.4.6 CMP with 8bit-DAC control register (CMP_DACR)	360
15.4.7 CMP with 8bit-DAC data registers (CMP_DADR1,CMP_DADR2)	360
15.4.8 CMP Internal Reference Voltage AD Conversion Register (CMP_RVADC)	361
16 Analog to Digital Conversion Modules (ADCs)	362
16.1 Introduction.....	362
16.2 ADC System Block Diagram	364
16.3 Function Description	366
16.3.1 ADC Clocks.....	366
16.3.2 Channel selection	366
16.3.3 Trigger source selection.....	367
16.3.4 Sequence A Single Scan Mode	368
16.3.5 Sequence A Continuous Scan Mode	369
16.3.6 Dual sequence scanning mode.....	370
16.3.7 Simulating Watchdog Functionality	372
16.3.8 Sampling time and conversion time of analog inputs	373
16.3.9 A/D data register auto-clear function	374

16.3.10 Convert data averaging functions	374
16.3.11 Programmable Gain Amplifier PGA.....	375
16.3.12 Multi ADC co-working mode	375
16.3.13 Interrupt and event signal output.....	380
16.4 Register Description.....	381
16.4.1 A/D start register ADC_STR	383
16.4.2 A/D control register 0 ADC_CR0	384
16.4.3 A/D control register 1 ADC_CR1	385
16.4.4 A/D conversion start trigger register ADC_TRGSR	386
16.4.5 A/D channel selection register A ADC_CHSELRA0	387
16.4.6 A/D channel selection register A 1 ADC_CHSELRA1	387
16.4.7 A/D Channel Select Register B ADC_CHSELRB0	388
16.4.8 A/D Channel Select Register B1 ADC_CHSELRB1	388
16.4.9 A/D Average Channel Select Register ADC_AVCHSELRO.....	389
16.4.10 A/D Average Channel Select Register 1 ADC_AVCHSELR1.....	389
16.4.11 A/D Sample Status Register ADC_SSTR.....	390
16.4.12 A/D Channel Mapping Control Register ADC_CHMUXR	391
16.4.13 A/D Interrupt Status Register ADC_ISR	392
16.4.14 A/D Interrupt Permit Register ADC_ICR	392
16.4.15 A/D Cooperative Mode Control Register ADC_SYNCCR.....	393
16.4.16 A/D Data Register ADC_DR	394
16.4.17 Analog Watchdog Control Register ADC_AWDCR	395
16.4.18 Analog Watchdog Threshold Register ADC_AWDDR0, ADC_AWDDR1.....	396
16.4.19 Analog Watchdog Compare Channel Select Register ADC_AWDCHSR0.....	397
16.4.20 Analog Watchdog Compare Channel Select Register 1 ADC_AWDCHSR1.....	397
16.4.21 Analog Watchdog Status Register ADC_AWDSR0	398
16.4.22 Analog Watchdog Status Register 1 ADC_AWDSR1	398
16.4.23 A/D Programmable Gain Amplifier Control Register ADC_PGACR	399
16.4.24 A/D Programmable Gain Multiplier Register ADC_PGAGSR	399
16.4.25 A/D Programmable Gain Amplifier Input Select Register ADC_PGAINSR0.....	400
16.4.26 A/D Programmable Gain Amplifier Input Select Register 1 ADC_PGAINSR1.....	400
16.5 Notes on use	401
16.5.1 Precautions for data register reading	401
16.5.2 Notes on Scan Completion Interrupt Handling.....	401
16.5.3 Notes on Module Stop and Low Power Setting.....	401

16.5.4	Pin setting for A/D conversion analog channel input	401
16.5.5	Noise Control.....	401
17	Temperature Sensor (OTS)	402
17.1	Brief Introduction	402
17.2	Instructions for use.....	403
17.3	Register Description.....	405
17.3.1	OTS control register (OTS_CTL).....	405
17.3.2	OTS Data Register 1 (OTS_DR1).....	406
17.3.3	OTS Data Register 2 (OTS_DR2).....	406
17.3.4	OTS Error Compensation Register (OTS_ECR).....	406
18	Advanced Control Timer (Timer6)	407
18.1	Brief Introduction	407
18.2	Basic Block Diagram.....	407
18.3	Function Description	409
18.3.1	Basic movements.....	409
18.3.2	Clock source selection.....	412
18.3.3	Counting direction	412
18.3.4	Digital filtering	413
18.3.5	Software synchronization.....	413
18.3.6	Hardware Synchronization.....	415
18.3.7	Pulse Width Measurement.....	417
18.3.8	Periodic measurement.....	417
18.3.9	Cache function.....	417
18.3.10	Universal PWM Output.....	424
18.3.11	Orthogonal coding count.....	429
18.3.12	Cycle interval response	434
18.3.13	EMB Control.....	434
18.3.14	Typical application examples.....	435
18.3.15	Function Summary Table	441
18.4	Interrupts and Event Descriptions.....	443
18.4.1	Interrupt output	443
18.4.2	Event Output	443
18.5	Register Description.....	445
18.5.1	General purpose count value register (TMR6_CNTER)	447
18.5.2	General Periodic Reference Value Register (TMR6_PERAR-PERCR)	447

18.5.3	General Comparison Reference Value Register (TMR6_GCMAR-GCMFR).....	447
18.5.4	Dedicated Comparison Reference Value Register (TMR6_SCMAR-SCMFR)	448
18.5.5	Dead Time Reference Register (TMR6_DTU<D>AR).....	448
18.5.6	General Control Register (TMR6_GCONR).....	449
18.5.7	Interrupt Control Register (TMR6_ICONR).....	450
18.5.8	Port Control Register (TMR6_PCONR)	452
18.5.9	Cache Control Register (TMR6_BCONR)	454
18.5.10	Deadband Control Register (TMR6_DCONR)	456
18.5.11	Filter Control Register (TMR6_FCONR)	457
18.5.12	Valid Period Register (TMR6_VPERR).....	458
18.5.13	Status Flag Register (TMR6_STFLR).....	459
18.5.14	Hardware Boot Event Select Register (TMR6_HSTAR)	460
18.5.15	Hardware Stop Event Select Register (TMR6_HSTPR)	461
18.5.16	Hardware Clear Event Select Register (TMR6_HCLRR).....	462
18.5.17	Hardware Capture Event Selection Register (TMR6_HCPAR)	463
18.5.18	Hardware Capture Event Selection Register (TMR6_HCPBR)	464
18.5.19	Hardware recursive event selection register (TMR6_HCUPR).....	465
18.5.20	Hardware decrement event selection register (TMR6_HCDOR)	467
18.5.21	Software Synchronization Start Control Register (TMR6_SSTAR)	469
18.5.22	Software Synchronization Stop Control Register (TMR6_SSPPR).....	469
18.5.23	Software Synchronous Clear Control Register (TMR6_SCLRR).....	470
19	Universal control timer (Timer4).....	471
19.1	Brief Introduction	471
19.2	Basic Block Diagram.....	471
19.3	Function Description	473
19.3.1	Basic movements.....	473
19.3.2	Cache function.....	476
19.3.3	Universal PWM Output.....	481
19.3.4	Cycle interval response	487
19.3.5	EMB Control.....	489
19.4	Interrupts and event descriptions	489
19.4.1	Counting comparison match interrupt.....	489
19.4.2	Counting cycle matching interrupt.....	489
19.4.3	Reload Count Match Interrupt.....	489
19.4.4	Dedicated comparison match events	490

19.5 Register Description	491
19.5.1 Count value register (TMR4_CNTR).....	493
19.5.2 Periodic Reference Register (TMR4_CPSR).....	493
19.5.3 Control Status Register (TMR4_CCSR)	494
19.5.4 Valid Period Register (TMR4_CVPR)	496
19.5.5 General Comparison Reference Register (TMR4_OCCRm)	496
19.5.6 General Control Status Register (TMR4_OCSRn)	497
19.5.7 General-purpose extended control register (TMR4_OCERn).....	498
19.5.8 General Purpose Mode Control Register (TMR4_OCMRm).....	500
19.5.9 Dedicated comparison reference register (TMR4_SCCRm).....	505
19.5.10 Dedicated Control Status Register (TMR4_SCSRm)	506
19.5.11 Dedicated Mode Control Register (TMR4_SCMRm)	508
19.5.12 PWM Basic Control Register (TMR4_POCRn)	509
19.5.13 PWM Filter Control Register (TMR4_PFSRn).....	510
19.5.14 PWM Deadband Control Register (TMR4_PDARn)	510
19.5.15 Reload Control Status Register (TMR4_RCSR)	511
19.5.16 EMB Control Status Register (TMR4_ECSR)	512
19.5.17 EMB Extended Control Register (TMR4_ECER)	512
20 Emergency Brake Module (EMB).....	513
20.1 Introduction.....	513
20.2 Function Description	514
20.2.1 Overview.....	514
20.2.2 Stop PWM signal output when external port input level changes	514
20.2.3 Stop PWM signal output when the PWM output port level is in phase (same high or same low)	
515	
20.2.4 Stop PWM signal output according to voltage comparator comparison result.....	516
20.2.5 Stop PWM signal output when external oscillator stops oscillating	516
20.2.6 Write register software control PWM signal output	516
20.3 Register Description.....	517
20.3.1 EMB control register 0 (EMB_CTL0).....	518
20.3.2 EMB control registers 1~3 (EMB_CTL1~3).....	519
20.3.3 EMB Feedback Level Select Register 0 (EMB_PWMLV0)	520
20.3.4 EMB feedback level selection registers 1~3 (EMB_PWMLV1~3)	521
20.3.5 EMB software output enable control register (EMB_SOEx) (x=0~3).....	521
20.3.6 EMB Status Register (EMB_STATx) (x=0~3)	522

20.3.7 EMB Status Reset Register (EMB_STATCLR x) ($x=0\sim 3$)	523
20.3.8 EMB interrupt permit register (EMB_INTEN x) ($x=0\sim 3$)	524
21 General purpose timer (TimerA).....	525
21.1 Introduction.....	525
21.2 Basic block diagram.....	525
21.3 Function Description	527
21.3.1 Basic movements.....	527
21.3.2 Clock source selection.....	530
21.3.3 Synchronous start	531
21.3.4 Digital filtering	532
21.3.5 Cache function.....	533
21.3.6 Cascade Count	534
21.3.7 PWM Output	535
21.3.8 Orthogonal coding count.....	536
21.4 Interrupts and Event Descriptions.....	541
21.4.1 Compare Matching Interrupts and Events.....	541
21.4.2 Cycle Matching Breaks and Events.....	541
21.5 Register Description	542
21.5.1 General purpose count value register (TMRA_CNTER).....	544
21.5.2 Periodic Reference Value Register (TMRA_PERAR).....	544
21.5.3 Compare reference value registers (TMRA_CMPAR1~8)	544
21.5.4 Control Status Register (TMRA_BCSTR)	545
21.5.5 Interrupt Control Register (TMRA_ICONR)	546
21.5.6 Event Control Register (TMRA_ECONR)	547
21.5.7 Filter Control Register (TMRA_FCONR).....	548
21.5.8 Status Flag Register (TMRA_STFLR)	549
21.5.9 Cache control registers (TMRA_BCONR1~4)	550
21.5.10 Capture control registers (TMRA_CCONR1~8).....	551
21.5.11 Port control registers (TMRA_PCONR1~8)	552
21.5.12 Hardware Trigger Event Selection Register (TMRA_HCONR)	554
21.5.13 Hardware Recursive Event Select Register (TMRA_HCUPR)	556
21.5.14 Hardware decrement event selection register (TMRA_HCDOR).....	558
22 General purpose timer (Timer0).....	560
22.1 Introduction.....	560
22.2 Basic Block Diagram.....	560

22.3 Function Description	561
22.3.1 Clock Source Selection.....	561
22.3.2 Basic counting action	562
22.3.3 Hardware Trigger Action	562
22.4 Interrupts and Event Descriptions.....	563
22.4.1 Interrupt output.....	563
22.4.2 Event Output	563
22.5 Register Description.....	564
22.5.1 Count value register (TMR0_CNTAR).....	564
22.5.2 Base value register (TMR0_CMPAR).....	564
22.5.3 Basic Control Register (TMR0_BCONR).....	565
22.5.4 Status Flag Register (TMR0_STFLR).....	568
22.6 Notes on use	568
23 Real Time Clock (RTC)	569
23.1 Introduction.....	569
23.2 Basic block diagram.....	570
23.3 Function Description	571
23.3.1 Power-up setting.....	571
23.3.2 RTC Counting Start Setting	571
23.3.3 System low power mode switching.....	571
23.3.4 Read out count register	571
23.3.5 Write Count Register	572
23.3.6 Alarm clock setting	572
23.3.7 Clock error compensation	572
23.3.8 1Hz output	572
23.4 Interrupt Description.....	574
23.4.1 Alarm clock interruption.....	574
23.4.2 Fixed-cycle interrupts.....	574
23.5 Register Description.....	575
23.5.1 Control register 0 (RTC_CR0)	576
23.5.2 Control register 1 (RTC_CR1)	577
23.5.3 Control register 2 (RTC_CR2)	578
23.5.4 Control register 3 (RTC_CR3)	579
23.5.5 Second Count Register (RTC_SEC).....	579
23.5.6 Sub-count register (RTC_MIN)	580

23.5.7 Time Count Register (RTC_HOUR).....	581
23.5.8 Day Count Register (RTC_DAY)	583
23.5.9 Week Count Register (RTC_WEEK)	584
23.5.10 Monthly Count Register (RTC_MON)	585
23.5.11 Year Count Register (RTC_YEAR).....	585
23.5.12 Minute Alarm Register (RTC_ALMMIN).....	586
23.5.13 Time Alarm Clock Register (RTC_ALMHOUR)	586
23.5.14 Weekly alarm clock register (RTC_ALMWEEK)	587
23.5.15 Clock error compensation registers (RTC_ERRCRH, RTC_ERRCRL).....	588
24 Watchdog Counter (WDT/SWDT).....	591
24.1 Introduction.....	591
24.2 Function Description	592
24.2.1 Start Watchdog	592
24.2.2 Hardware boot method.....	592
24.2.3 Software startup method	593
24.2.4 Refresh action	594
24.2.5 Logo position.....	594
24.2.6 Interrupt Reset	594
24.2.7 Counting down overflow	595
24.2.8 Refresh error	596
24.3 Register Description.....	597
24.3.1 Control register (WDT_CR).....	598
24.3.2 Status registers (SWDT_SR, WDT_SR).....	599
24.3.3 Refresh registers (SWDT_RR, WDT_RR)	599
24.4 Notes on use	599
25 Universal Synchronous Asynchronous Transceiver (USART).....	600
25.1 Brief Introduction	600
25.2 USART System Block Diagram	601
25.3 Pin Description.....	601
25.4 Function Description	602
25.4.1 UART	602
25.4.2 Multi-processor communication.....	611
25.4.3 Smart Card	615
25.4.4 Clock synchronization mode	619
25.4.5 Digital filtering function.....	625

25.5 Register Description.....	626
25.5.1 Status register (USART_SR)	627
25.5.2 Data register (USART_DR).....	630
25.5.3 Baud Rate Register (USART_BRR)	631
25.5.4 Control register 1 (USART_CR1).....	632
25.5.5 Control register 2 (USART_CR2).....	635
25.5.6 Control register 3 (USART_CR3).....	636
25.5.7 Prescaler register (USART_PR)	637
25.6 Notes on use	638
25.6.1 UART Caution.....	638
25.6.2 Clock synchronization mode notes	638
25.6.3 Other Notes.....	638
26 Integrated Circuit Bus (I2C)	639
26.1 Brief Introduction	639
26.2 I2C System Block Diagram.....	640
26.2.1 System block diagram	640
26.2.2 Structure Diagram.....	641
26.3 Action Description.....	642
26.3.1 I2C Protocol.....	642
26.3.2 Address Matching	651
26.3.3 SMBus Action.....	657
26.3.4 Reset	658
26.3.5 Interrupt and event signal output.....	659
26.3.6 Programmable digital filtering	660
26.4 Application Software Setting I2C Initialization Process	660
26.5 Register Description.....	661
26.5.1 I2C control register 1 (I2C_CR1).....	662
26.5.2 I2C control register 1 (I2C_CR2)	664
26.5.3 I2C control register 1 (I2C_CR3)	666
26.5.4 I2C control register 1 (I2C_CR4)	667
26.5.5 I2C slave address register 0 (I2C_SLR0)	668
26.5.6 I2C slave address register 1 (I2C_SLR1)	669
26.5.7 I2C SCL Level Timeout Control Register (I2C_SLTR)	670
26.5.8 I2C Status Register (I2C_SR)	671
26.5.9 I2C Status Clear Register (I2C_CLR)	675

26.5.10 I2C Data Transmit Register (I2C_DTR)	676
26.5.11 I2C Data Receive Register (I2C_DR).....	676
26.5.12 I2C Data Shift Register (I2C_DSR)	676
26.5.13 I2C clock control register (I2C_CCR).....	677
26.5.14 I2C Filter Control Register (I2C_FLTR).....	679
27 Serial Peripheral Interface (SPI).....	680
27.1 Introduction.....	680
27.2 SPI System Block Diagram	681
27.3 Pin Description.....	681
27.4 SPI Action System Description.....	682
27.4.1 Host Mode Pin Status.....	682
27.4.2 Slave Mode Pin Status.....	682
27.4.3 SPI System Connection Example.....	683
27.5 Data Communication Description	684
27.5.1 Baud rate.....	684
27.5.2 Data Format	685
27.5.3 Transmission format.....	686
27.5.4 Communication method	688
27.6 Running Instructions	690
27.6.1 Outline of operation mode	690
27.6.2 Host action during SPI operation mode	691
27.6.3 Slave operation during SPI operation mode	691
27.6.4 Host action in clock synchronous operation mode	693
27.6.5 Slave operation in clock synchronous operation mode.....	694
27.6.6 Processing flow of several SPI actions	695
27.7 Parity bit self-diagnosis	696
27.8 Error Detection	697
27.8.1 Underload error.....	697
27.8.2 Mode error.....	698
27.8.3 Overload error.....	698
27.8.4 Parity error	700
27.9 Initialization of SPI.....	701
27.9.1 Clear SPE bit for initialization	701
27.9.2 System reset initialization	701
27.10 Interrupt source	702

27.11 Available event trigger sources	702
27.12 Register Description.....	703
27.12.1 SPI Data Register (SPI_DR).....	703
27.12.2 SPI control register (SPI_CR1).....	704
27.12.3 SPI communication configuration register 1 (SPI_CFG1).....	705
27.12.4 SPI Status Register (SPI_SR)	707
27.12.5 SPI communication configuration register 2 (SPI_CFG2).....	708
28 Quad Wire Serial Peripheral Interface (QSPI).....	710
28.1 Introduction.....	710
28.2 Memory Mapping	712
28.2.1 Internal bus space.....	712
28.2.2 ROM space and address width of the bus	713
28.3 QSPI Bus.....	714
28.3.1 SPI Protocol	714
28.3.2 SPI Mode.....	716
28.4 Timing Adjustment of QSPI Bus.....	717
28.4.1 QSPI Bus Reference Clock	717
28.4.2 SPI Bus Reference Clock	718
28.4.3 QSSN signal minimum high level width.....	718
28.4.4 QSSN build time.....	719
28.4.5 QSSN hold time	719
28.4.6 Serial data reception delay.....	720
28.5 Introduction of SPI Instructions for ROM Access.....	721
28.5.1 Existing QSPI-ROM Instruction Reference.....	721
28.5.2 Standard Read Instructions.....	722
28.5.3 Quick read command	722
28.5.4 2-wire output quick read command	724
28.5.5 2-wire input/output quick read command.....	725
28.5.6 Four-wire output quick read command.....	726
28.5.7 Four-wire input/output quick read command	727
28.5.8 Enter 4-Byte mode command	728
28.5.9 Exit 4-Byte mode command	729
28.5.10 Write permission command	729
28.6 QSPI Bus Cycle Scheduling.....	730
28.6.1 Single flash read with independent conversion.....	730

28.6.2 Flash memory reading using the pre-read function	730
28.6.3 Pre-read termination.....	731
28.6.4 Pre-reading status monitoring.....	731
28.6.5 Flash Reads with QSPI Bus Cycle Extension.....	731
28.7 XIP Control	732
28.7.1 XIP mode settings	733
28.7.2 XIP mode exit.....	733
28.8 QSIO2 and QSIO3 pin status.....	734
28.9 Direct communication mode	735
28.9.1 About direct communication mode.....	735
28.9.2 Setting of direct communication mode	735
28.9.3 Generation of QSPI Bus Cycles in Direct Communication Mode.....	735
28.10 Interruption	736
28.11 Notes on use	736
28.11.1 QSPI Register Setting Order.....	736
28.11.2 Setting of the module stop signal	736
28.12 Register Description.....	737
28.12.1 QSPI Control Register (QSCR)	738
28.12.2 QSPI Chip Select Control Register (QSCSCR)	741
28.12.3 QSPI Format Control Register (QSFCR).....	742
28.12.4 QSPI Status Register (QSSR).....	744
28.12.5 QSPI Command Code Register (QSCCMD)	746
28.12.6 QSPI Direct Communication Command Register (QSDCOM)	746
28.12.7 QSPI XIP Mode Code Register (QSXCMD)	746
28.12.8 QSPI System Configuration Register (QSSR2).....	747
28.12.9 QSPI External Extended Address Register (QSEXAR)	747
29 Integrated circuit built-in audio bus module (I2S).....	748
29.1 Introduction.....	748
29.2 I2S System Block Diagram	749
29.3 Pin Description.....	750
29.4 Function Description	750
29.4.1 I2S General Description.....	750
29.4.2 Communication method	750
29.4.3 Supported audio protocols.....	751
29.4.4 Clock generators	757

29.4.5 I2S Master Mode.....	759
29.4.6 I2S Slave Mode.....	760
29.4.7 I2S Interrupt	761
29.4.8 Notes on use	761
29.5 Register Description.....	764
29.5.1 I2S control register (I2S_CTRL)	765
29.5.2 I2S Status Register (I2S_SR)	767
29.5.3 I2S Error Status Register (I2S_ER).....	768
29.5.4 I2S Configuration Register (I2S_CFGR).....	769
29.5.5 I2S transmit buffer FIFO data register (I2S_TXBUF)	770
29.5.6 I2S Receive Buffer FIFO Data Register (I2S_RXBUF).....	770
29.5.7 I2S divider register (I2S_PR)	771
30 Controller Area Network (CAN)	772
30.1 Brief Introduction	772
30.2 CAN System Block Diagram	773
30.3 Pin Description.....	773
30.4 Function Description.....	774
30.4.1 Baud rate setting	774
30.4.2 Send buffer.....	775
30.4.3 Receiving Buffer	775
30.4.4 Receive Screening Register Set.....	776
30.4.5 Data sending	777
30.4.6 Single data transmission	777
30.4.7 Cancel data sending.....	777
30.4.8 Data reception.....	778
30.4.9 Error Handling.....	778
30.4.10 Node closure	779
30.4.11 Arbitration Failure Location Capture.....	779
30.4.12 Loopback mode.....	779
30.4.13 Silent mode	780
30.4.14 Software reset function.....	781
30.4.15 Upward compatible with CAN-FD function	783
30.4.16 Time triggered TTCAN	783
30.4.17 Interruptions	786
30.5 Register Description.....	787

30.5.1 CAN Receive BUF Register (CAN_RBUF)	790
30.5.2 CAN Transmit BUF register (CAN_TBUF)	792
30.5.3 CAN Configuration and Status Register (CAN_CFG_STAT)	794
30.5.4 CAN Command Register (CAN_TCMD).....	795
30.5.5 CAN Transmit Control Register (CAN_TCTRL).....	797
30.5.6 CAN Receive Control Register (CAN_RCTRL)	799
30.5.7 CAN Receive and Transmit Interrupt Enable Register (CAN_RTIE).....	800
30.5.8 CAN Receive and Transmit Interrupt Status Register (CAN_RTIF)	801
30.5.9 CAN Error Interrupt Enable and Flag Register (CAN_ERRINT).....	803
30.5.10 CAN Bit Timing Register (CAN_BT)	804
30.5.11 CAN Error and Arbitration Failure Capture Register (CAN_EALCAP)	805
30.5.12 CAN Warning Qualification Register (CAN_LIMIT)	806
30.5.13 CAN Receive Error Counter Register (CAN_RECNT)	806
30.5.14 CAN Transmit Error Counter Register (CAN_TECNT)	807
30.5.15 CAN Filter Group Control Register (CAN_ACFCTRL)	808
30.5.16 CAN Filter Group Enable Register (CAN_ACFEN).....	809
30.5.17 CAN filter group code a n d mask registers (CAN_ACF)	810
30.5.18 TTCAN TB slot pointer register (CAN_TBSLOT).....	811
30.5.19 TTCAN Time-Triggered Configuration Register (CAN_TTCFG).....	812
30.5.20 TTCAN Reference Message Register (CAN_REF_MSG)	813
30.5.21 TTCAN Trigger Configuration Register (CAN_TRG_CFG).....	814
30.5.22 TTCAN Trigger Time Register (CAN_TT_TRIG)	815
30.5.23 TTCAN Trigger Watchdog Time Register (CAN_TT_WTRIG)	815
30.6 Notes on use	816
30.6.1 CAN bus anti-interference measures	816
30.6.2 CAN controller noise constraints	816
31 USB2.0 Full Speed Module (USBFS).....	817
31.1 USBFS Introduction	817
31.2 USBFS Key Features.....	817
31.2.1 General Features.....	817
31.2.2 Host Mode Features	818
31.2.3 Device mode characteristics.....	818
31.3 USBFS System Block Diagram.....	819
31.4 USBFS Pin Description.....	819
31.5 USBFS Feature Description.....	820

31.5.1	USBFS Clocks and Operating Modes.....	820
31.5.2	USBFS Mode Decision.....	820
31.5.3	USBFS Host Features	821
31.5.4	USBFS Device Features.....	825
31.5.5	USBFS SOF Pulse Pin Output Function.....	828
31.5.6	USBFS power control.....	829
31.5.7	USBFS dynamic update USBFS_HFIR register	830
31.5.8	USBFS Data FIFO	830
31.5.9	USBFS Host FIFO Architecture.....	831
31.5.10	USBFS Device FIFO Architecture.....	833
31.5.11	USBFS FIFO RAM allocation.....	834
31.5.12	USBFS System Performance.....	835
31.5.13	USBFS Interrupts and Events	836
31.6	USBFS Programming Model	837
31.6.1	USBFS module initialization	837
31.6.2	USBFS Host Initialization.....	837
31.6.3	USBFS device initialization.....	838
31.6.4	USBFS DMA mode	838
31.6.5	USBFS Host Programming Model	838
31.6.6	USBFS Device Programming Model.....	840
31.6.7	USBFS Operation Model.....	842
31.7	Register Description	858
31.7.1	USBFS system control register.....	862
31.7.2	USBFS Global Register.....	863
31.7.3	USBFS Host Mode Register.....	889
31.7.4	USBFS Device Mode Register	905
31.7.5	USBFS clock gating control register.....	935
32	Cryptographic Coprocessing Module (CPM)	936
32.1	Introduction.....	936
32.2	Encryption and decryption algorithm processor (AES)	936
32.2.1	Algorithm Introduction	936
32.2.2	Encryption operation process.....	937
32.2.3	Decrypt the operation process.....	937
32.2.4	Encryption and decryption time.....	938
32.2.5	Operation Notes	938

32.2.6 Register Description.....	939
32.3 Secure Hash Algorithm (HASH).....	943
32.3.1 Algorithm Introduction	943
32.3.2 Operation flow.....	944
32.3.3 Message Filling.....	944
32.3.4 Register Description.....	946
32.4 True Random Number Generator (TRNG).....	950
32.4.1 Module Block Diagram	950
32.4.2 Operation Process.....	950
32.4.3 Interrupt and event output	950
32.4.4 Operation Notes	950
32.4.5 Register Description.....	951
33 Data Computing Unit (DCU)	953
33.1 Introduction.....	953
33.2 Function Description.....	954
33.2.1 Additive mode	954
33.2.2 Subtractive mode	954
33.2.3 Hardware Triggered Boot Mode.....	954
33.2.4 Compare modes.....	954
33.2.5 Interrupt and event signal output.....	955
33.3 Application Examples.....	956
33.3.1 Additive mode	956
33.3.2 Trigger plus mode	956
33.3.3 Compare modes.....	956
33.4 Register Description.....	957
33.4.1 DCU control register (DCU_CTL).....	959
33.4.2 DCU flag register (DCU_FLAG)	960
33.4.3 DCU data register (DCU_DATAx) (x=0,1,2)	961
33.4.4 DCU flag reset register (DCU_FLAGCLR).....	962
33.4.5 DCU interrupt condition selection register (DCU_INTEVTSEL)	963
34 CRC operation (CRC)	964
34.1 Introduction.....	964
34.2 Functional Block Diagram	964
34.3 Function Description	965
34.3.1 CRC code generation	965

34.3.2	CRC checksum	965
34.3.3	CRC checks for XOROUT,REFOUT,REFIN not all 1	966
34.4	Register Description.....	967
34.4.1	Control register (CRC_CR).....	967
34.4.2	Result register (CRC_RESLT).....	968
34.4.3	Flag register (CRC_FLG).....	969
34.4.4	Data register (CRC_DAT).....	969
35	SDIO Controller (SDIOC)	970
35.1	Introduction.....	970
35.2	Function Description.....	971
35.2.1	Port assignment.....	971
35.2.2	Basic access method	971
35.2.3	Data Transfer.....	972
35.2.4	SD Clock	972
35.2.5	Interrupt and DMA Start Request.....	972
35.2.6	Host and device initialization	973
35.2.7	SD/MMC single block read/write	975
35.2.8	SD/MMC multi block read/write	975
35.2.9	Transfer termination (abort), suspend (suspend) and resume (resume) 977	
35.2.10	read wait	977
35.2.11	Wakeup	978
35.3	Register Description.....	980
35.3.1	Data block length register (BLKSIZE)	981
35.3.2	Data Block Counting Memory (BLKCNT)	981
35.3.3	Parameter register 0(ARG0)	981
35.3.4	Parameter register 1(ARG1)	982
35.3.5	Transmission mode register (TRANSMODE)	982
35.3.6	Command Register (CMD)	983
35.3.7	Response register 0 (RESP0).....	983
35.3.8	Response register 1 (RESP1).....	984
35.3.9	Response register 2 (RESP2).....	984
35.3.10	Response register 3 (RESP3).....	984
35.3.11	Response register 4 (RESP4).....	985
35.3.12	Response register 5 (RESP5).....	985
35.3.13	Response register 6 (RESP6).....	985

35.3.14 Response register 7 (RESP7).....	986
35.3.15 Data buffer register 0(BUF0).....	986
35.3.16 Data buffer register 1(BUF1).....	986
35.3.17 Host Status Register (PSTAT).....	987
35.3.18 Host control register (HOSTCON)	988
35.3.19 Power Control Register (PWRCON).....	988
35.3.20 Data block gap control register (BLKGPCON)	989
35.3.21 Clock control register (CLKCON)	989
35.3.22 Timeout control register (TOUTCON).....	990
35.3.23 Software Reset Register (SFTRST)	990
35.3.24 General interrupt status register (NORINTST)	991
35.3.25 Error interrupt status register (ERRINTST).....	991
35.3.26 General interrupt status enable register (NORINTSTEN).....	992
35.3.27 Error interrupt status enable register (ERRINTSTEN).....	993
35.3.28 General interrupt signal enable register (NORINTSGEN)	994
35.3.29 Error interrupt signal enable register (ERRINTSGEN)	995
35.3.30 Automatic Command Error Status Register (ATCERRST)	996
35.3.31 Forced Automatic Command Error Status Control Register (FEA)	997
35.3.32 Forced Error Status Control Register (FEE)	998
35.3.33 MMC Mode Enable Register (MMCER).....	999
36 Debugging Controller (DBGC).....	1000
36.1 Brief Introduction	1000
36.2 DBGC System Block Diagram.....	1001
36.3 SWJ-DP Debug Port (SWD and JTAG)	1002
36.3.1 Switching mechanism f o r JTAG-DP or SW-DP	1003
36.4 Pinouts and Debug Port Pins	1004
36.4.1 SWJ Debug Port Pins	1004
36.4.2 Flexible SWJ-DP Pin Assignment	1004
36.4.3 Internal pull-up on JTAG pins	1005
36.4.4 Using the serial interface and releasing unused debug pins for GPIO	1005
36.5 Register	1006
36.5.1 DBG Status Register (MCUDBGSTAT).....	1006
36.5.2 Peripheral debug pause register (MCUSTPCTL)	1007
36.5.3 Debug Component Configuration Register (MCUTRACECTL)	1009
36.6 SW Debug Port.....	1010

36.6.1	Introduction to SW Protocol.....	1010
36.7	TPIU (Tracking Port Interface Unit).....	1010
36.7.1	Introduction.....	1010
36.7.2	TRACE Pin Assignment	1011
36.7.3	MCU Internal TRACECLKIN Connection.....	1012
36.7.4	TPIU Register.....	1012
36.7.5	TPIU Configuration Example.....	1012
Version revision record.	1014

Table Index

Table 1-1	
Memory Mapping	45
Table 1-2	
QSPI Address Space Allocation	49
Table 1-3 Example of Destination Address Configuration	50
Table 1-4	
Register List.....	51
Table 3-1 Reset mode and generation conditions	58
Table 3-2 Reset method and reset flag.....	59
Table 3-3 RMU Register List	69
Table 4-1 Specifications of each internal clock.....	76
Table 5-1	BOR
Configuration	126
Table 5-2	
PVD1/PVD2 Characteristics.....	127
Table 5-3	
Operation Mode.....	132
Table 5-4	Low
Power Mode.....	132
Table 5-5 Operating conditions of low-power mode and the state of each module in low-power mode ...	133
Table 5-6	
Operation Mode Description.....	135
Table 5-7 Power-down mode sub-mode.....	139
Table 5-8 RAM Module and RAM Power Down Control Bits.....	143
Table 5-9 Register Protection List	144
Table 5-10	Register
List 145	
Table 6-1	List of
registers 176	
Table 7-1	CPU
clock frequency and FLASH read wait cycle comparison table	186
Table 7-2FLASH Actual Read Cycles.....	188
Table 7-3	OTP
Address Composition.....	193
Table 7-4	Register

List	195
Table 8-1 Relationship between the wait cycle setting of SRAM read/write access and CPU clock frequency	205
Table 8-2 SRAM Space Allocation.....	206
Table 9-1 PORT Register List 1.....	219
Table 9-2 PORT Register List 2.....	219
Table 9-3 List of PORT registers at 32bit access.....	232
Table 10-1	Interrupt vector table 236
Table 10-2	Interrupt Event Request Sequence Number and Selection 241
Table 12-1 KEYS CAN Pin Descriptions.....	294
Table 12-2 List o f KEYS CAN registers.....	296
Table 14-1	Channel Reset Description 319

Table 15-1 CMP Detailed Specifications.....	346
Table 15-2 List of CMP registers	353
Table 16-1 Specifications of each ADC unit	365
Table 16-2Register setting method when converting internal channels	367
Table 16-3 Various competitions forsequences A a n d B	370
Table 16-4 AD Conversion Time.....	373
Table 16-5ADC Register List 1/2	381
Table 17-1 List o f OTS registers.....	405
Table 18-1..... Basic Functions and Features o f Timer6	407
Table 18-2	Timer6 Port List
	408
Table 18-3	Comparison table of functions in different modes.....
	441
Table 18-4	Register List
	445
Table 19-1..... Basic Functions and Features o f Timer4	471
Table 19-2	Timer4 Port List
	472
Table 19-3	Register List
	491
Table 21-1	Basic Functions and Features o f TimerA
	525
Table 21-2.....	TimerA Port List
	526
Table 21-3	Register List
	542
Table 22-1	Register List
	564
Table 23-1 Basic Specifications of the RTC	569
Table 23-2	Register List
	575
Table 24-1	Basic characteristics of the

watchdog counter	591
Table 24-2	Register List
	597
Table 25-1 USART Pin Descriptions	601
Table 25-2.....	UART receiver tolerance w h e n DIV_Fraction is 0
	607
Table 25-3.....	Tolerance of UART Receiver when DIV_Fraction is not 0
	607
Table 25-4 UART interrupt/event table.....	611
Table 25-5	Multiprocessor Mode Interrupt/Event Table
	615
Table 25-6	Smart Card Mode Interrupt/Event Table
	619
Table 25-7	Clock synchronization mode interrupt/event table
	624
Table 25-8 List o f USART registers	626
Table 25-9 Baud rate calculation formula (fractional baud rate invalid FBME=0)	631
Table 25-10 Baud rate calculation formula (fractional baud rate valid FBME=1)	631
Table 26-1.....	Input/Output Pins
	641
Table 26-2	Register List
	661

Table 27-1	Highlights of SPI Characteristics
680	
Table 27-2	Pin Descriptions
681	
Table 27-3 SPI Pin Status Descriptions in	Host Mode
682	
Table 27-4 SPI Pin Status Descriptions in	Slave Mode
682	
Table 27-5	Partial setpoint lower speed
684	
Table 27-6	SPI Mode and Register Setting Relationships
690	
Table 27-7	Error Detection Correspondence Table
697	
Table 27-8	SPI Interrupt Source Descriptions
702	
Table 28-1	QSPI Main Specifications
710	
Table 28-2	QSPI Pins
711	
Table 28-3	QSPI Bus Reference Clock Selection List
717	
Table 28-4 List of	Reference Instructions
721	
Table 28-5.....	Pin Status of QIO2 and QIO3
734	
Table 28-6	QSPI Register List
737	
Table 29-1.....	I2S Main Characteristics
748	
Table 29-2.....	I2S Pin Descriptions
750	
Table 29-3 Audio Frequency Accuracy (for VCO input frequency = 1MHz)	758
Table 29-4.....	I2S Interrupt Requests
761	
Table 29-5.....	List of I2S registers
764	

Table 30-1 CAN Pin Descriptions.....	773
Table 30-2 CAN Bit Time Setting Rules	775
Table 30-3	Software Reset Range Table
	781
Table 30-4 CAN Interrupt Table.....	786
Table 30-5 List of CAN registers.....	787
Table 30-6 CAN Register BYTE/HALFWORD/WORD Access Layout	787
Table 30-7 Standard Format CAN Receive Mailbox Format.....	790
Table 30-8 Extended Format CAN Receive Mailbox Format	791
Table 30-9 Standard Format CAN Send Mailbox Format	792
Table 30-10 Extended Format CAN Send Mailbox Format	793
Table 31-1	USBFS Pin Descriptions
	819
Table 31-2 USBFS Interrupt Events Table	836
Table 31-3 USBFS System Control Register List.....	858
Table 31-4 USBFS System Control Register List.....	859
Table 34-1 CRC Register List.....	967
Table 36-1	SWJ Debug Port Pins
	1004

Table 36-2 Flexible SWJ-DP Pin Assignments 1004

Figure Index

Figure 2-1	Bus Architecture.....	55
Figure 3-1	Power-on reset	60
Figure 3-2	Reset Timing.....	NRST 60
Figure 3-3	Undervoltage reset.....	61
Figure 3-4	Programmable Voltage Detect 1 Reset.....	62
Figure 3-5	Programmable Voltage Detect 2 Reset.....	63
Figure 3-6	Watchdog and Dedicated Watchdog Reset.....	64
Figure 3-7	Power-down wake-up reset.....	64
Figure 3-8	Software Reset.....	65
Figure 3-9 MPU Error Reset	65	
Figure 3-10	RAM Parity Reset	65
Figure 3-11	RAMECC Reset.....	66
Figure 3-12	Frequency Abnormal Reset.....	Clock 66
Figure 3-13	External high-speed oscillation abnormal reset	67
Figure 4-1	Block diagram of clock system	72
Figure 4-2	Block diagram of clock frequency measurement	73
Figure 4-3	Example of external high-speed oscillator connection.....	78
Figure 4-4	Example diagram of the connection of the external clock input.....	79
Figure 4-5		

Example of external high-speed oscillator fault detection	80
Figure 4-6	
System clock selection XTAL, XTAL oscillation fault detected Example	81
Figure 4-7	
Example of external low-speed oscillator connection	82
Figure 4-8	Clock
Frequency Measurement Timing Diagram	88
Figure 5-1	Power
supply composition diagram.....	124
Figure 5-2	Power-
on reset and power-down reset waveforms.....	125
Figure 5-3	
Undervoltage reset waveform	126
Figure 5-4	PVD1
interrupt/reset block diagram.....	128
Figure 5-5	Block
diagram of interrupt/reset.....	128
Figure 5-6	Power
Monitor 1 Interrupt Timing Diagram.....	129
Figure 5-7	Power
Monitor 1 Reset Timing Diagram.....	129
Figure 5-8	Power
Monitor 2 Interrupt Operation Timing Diagram.....	130
Figure 5-9	Power
Monitor 2 Reset Operation Timing Diagram.....	130
Figure 5-10	Internal
voltage sampling diagram	131
Figure 5-11.....	PTWK _n
structure block diagram.....	141

Figure 7-1FLASH address structure (512KB product)	183
Figure 7-2FLASH Address Structure (256KB product)	184
Figure 7-3	Startup Sector Swap Function 1
194	
Figure 7-4	Launching the switching function 2
194	
Figure 9-1	Schematic diagram of the basic structure of the port
214	
Figure 10-1	Interrupt System Block Diagram
235	
Figure 10-2	Interrupt Event Selection
257	
Figure 10-3	Digital filter operating diagram
258	
Figure 12-1	KEYSCAN system block diagram
293	
Figure 12-2	Schematic diagram of
keyboard scanning function.....	294
Figure 14-1	DMA structure diagram
313	
Figure 14-2	Chain transfer diagram
317	
Figure 14-3	Discontinuous Address Transfer Schematic
318	
Figure 14-4	Discontinuous Reset Schematic
320	
Figure 14-5	Application Example 1: Memory-to-Memory Transfer
323	
Figure 14-6	Application Example 2: Memory to Peripheral Circuit Transfer
324	
Figure 15-1	CMP, 8bitDAC Function Connection Diagram
347	
Figure 15-2	CMP working schematic
348	
Figure 15-3	Scan Mode Action Schematic
350	
Figure 16-1	ADC Block Diagram

364

Figure 16-2	Internal Analog Channel Selection
	366
Figure 16-3	Sequence A Single Scan Mode
	368
Figure 16-4	Continuous Scan
	369
Figure 16-5	Dual Sequence Scan Mode (Sequence A Restarted from Interrupted Channel)
	371
Figure 16-6	Dual Sequence Scan Mode (Sequence A restarted from the first channel)
	371
Figure 16-7	Simulated Watchdog Protection Area (Compare Conditions)
	372
Figure 16-8.....	A/D Conversion Time
	373
Figure 16-9	Switching action when the averaging function is active.....
	374
Figure 16-10.....	Schematic diagram of PGA operation
	375
Figure 16-11 Single Parallel Trigger Mode (Triple ADC).....	376
Figure 16-12 Single Delayed Trigger Mode (Triple ADC).....	377
Figure 16-13 Cyclic Parallel Trigger Mode (Triple ADC)	378
Figure 16-14 Cyclic delayed trigger mode (two ADCs)	379
Figure 16-15 Cyclic Delay Trigger Mode (Triple ADC)	379
Figure 16-16	ADC Interrupt and Event Output Timing
	380

Figure 17-1	OTS Function Block Diagram
	402
Figure 18-1	Timer6 Basic Block Diagram
	408
Figure 18-2	Sawtooth waveform (recursive counting)
	409
Figure 18-3	Triangular waveform
	409
Figure 18-4	Compare Output Actions
	410
Figure 18-5	Capturing Input Actions
	411
Figure 18-6	Capturing the filtering function of the input port.....
	413
Figure 18-7	Software synchronization action
	414
Figure 18-8	Hardware synchronization action
	416
Figure 18-9	Single-cache method comparison output timing
	418
Figure 18-10	Dual Cache Method Capture Input Timing
	419
Figure 18-11	Counting cache action during sawtooth wave mode
	420
Figure 18-12 Counting cache action during	delta wave A mode
	421
Figure 18-13 Counting cache action during	delta wave B mode
	422
Figure 18-14TIM6_<t>_PWMA O u t p u t	PWM Wave
	424
Figure 18-15	Software Setting GCMBR Complementary PWM Output in Triangle A Mode
	425
Figure 18-16 Hardware Setting GCMBR Complementary PWM Output in Triangle B Mode (Symmetric Deadband)	426
Figure 18-17	6-Phase PWM Wave
	427
Figure 18-18	Three-phase complementary PWM output with dead time in triangle A mode
	428

Figure 18-19	Position Mode - Basic Counting
	429
Figure 18-20	Position Counting Mode - Phase Difference Counting (1X Count)
	430
Figure 18-21	Position Counting Mode - Phase Difference Counting (2X Counting)
	430
Figure 18-22	Position Counting Mode - Phase Difference Counting (4x Counting)
	430
Figure 18-23	Position Count Mode - Direction Count
	431
Figure 18-24	Rotation counting mode - Z-phase counting
	431
Figure 18-25	Rotation Counting Mode - Position Overflow Counting
	432
Figure 18-26	Rotation Counting Mode-Mixed Counting
	432
Figure 18-27	Rotation counting mode - mixed counting Z-phase shield action example 1
	433
Figure 18-28	Rotation counting mode - mixed counting Z-phase shield action example 2
	433
Figure 18-29	Cycle interval valid request signal action
	434
Figure 18-30 Example of	interrupt & event output during sawtooth wave mode
	444
Figure 19-1	Timer4 Basic Block Diagram
	472
Figure 19-2.....	Timer4 sawtooth waveform
	473
Figure 19-3	Timer4 Triangle Waveform
	473
Figure 19-4	Timer4 sawtooth wave mode counting action
	473

Figure 19-5.....	Timer4 Triangle Wave Mode Counting Action
	474
Figure 19-6	Sawtooth mode waveform output example
	475
Figure 19-7 Example of	triangle waveform output
	475
Figure 19-8	Modifying the Ramp Count Period
when the Cache is Invalid.....	476
Figure 19-9	Modifying the Sawtooth Count Period
when Cache Enabled	477
Figure 19-10	Modifying the Triangle Wave Count Period
when Cache Enabled	477
Figure 19-11 OCCR Buffered Data Transfer (When Cycle Interval Response Link is Disabled)	478
Figure 19-12 OCCR Buffered Data Transfer (Cycle Interval Response Link Enable)	479
Figure 19-13 Output Compare Buffer Data Transfer (OCMR Buffer Enable).....	479
Figure 19-14 SCCR Buffered Transfer Operation (Periodic Interval Response When Link Transfer is Disallowed)	
480	
Figure 19-15 SCCR Buffered Transfer Operation (When Link Transfer Enable is Responded to at Periodic Intervals)	481
Figure 19-16.....	Example of Sawtooth Independent PWM Output
	482
Figure 19-17.....	Example of Triangle Independent PWM Output
	482
Figure 19-18.....	Sawtooth Wave Extended PWM Output
	483
Figure 19-19.....	Software Implementation of Complementary PWM Outputs
	484
Figures 19-20	Complementary PWM Output in Dead Timer Mode
	485
Figure 19-21	Waveform output
in dead timer mode during pulse width exception	485
Figures 19-22.....	Complementary PWM Output in Dead Timer Filter Mode
	486
Figure 19-23	Cycle Interval Response Timing Diagram
	487
Figure 19-24	Dedicated Event Output Signal Cycle Interval Response Output
	488

Figure 19-25	Output Timing of Dedicated Event Output Signal in Delayed Start Mode	490
Figure 21-1.....	TimerA Basic Block Diagram	
	526	
Figure 21-2	Sawtooth waveform (recursive counting)	
	527	
Figure 21-3	Triangular waveform	
	527	
Figure 21-4	Comparison Output Action	
	528	
Figure 21-5	Capturing Input Actions	
	529	
Figure 21-6	Software synchronization action	
	531	
Figure 21-7	Filtering Function of Clock Input Port.....	
	532	
Figure 21-8	Cache action during sawtooth wave mode.....	
	533	
Figure 21-9	32 Bit Cascade Counting Action.....	534
Figure 21-10	General Purpose PWM Output Example	
	535	
Figure 21-11	Position Mode - Basic Counting	
	537	
Figure 21-12	Position Counting Mode - Phase Difference Counting (1X Count)	
	537	
Figure 21-13	Position Counting Mode - Phase Difference Counting (2X Counting)	
	538	
Figure 21-14	Position Counting Mode - Phase Difference Counting (4x Counting)	
	538	

Figure 21-15	Position Count Mode - Direction Count
	538
Figure 21-16	Rotation counting mode - Z-phase counting
	539
Figure 21-17	Rotation Counting Mode - Position Overflow Counting
	539
Figure 21-18	Rotation counting mode - mixed counting
	540
Figure 22-1	Timer0 Basic Block Diagram
	560
Figure 22-2.....	Timer0 Counting Timing Diagram
	562
Figure 23-1.....	Basic block diagram of the RTC
	570
Figure 24-1	Hardware Boot Example
	592
Figure 24-2	Software startup example
	593
Figure 24-3 Example of various refresh action timings (action confirmation, falling edge of refresh request signal, etc.)	594
Figure 24-4 Example of	Counter Overflow Action
	595
Figure 24-5 Example of	counter refresh action
	596
Figure 25-1	USART System Block Diagram
	601
Figure 25-2.....	UART Data Format
	603
Figure 25-3	UART Transmit Data Legend 1
	605
Figure 25-4	UART Transmit Data Legend 2
	605
Figure 25-5.....	UART Internal Synchronization and Sampling Timing
	606
Figure 25-6	UART Receive Data Legend 1
	608
Figure 25-7	UART Receive Data Legend 2
	608

Figure 25-8	Multiprocessor Communication Legend
611	
Figure 25-9	Multiprocessor Mode Data Format
612	
Figure 25-10	Multiprocessor Mode Sending Data Example
613	
Figure 25-11	Multiprocessor Mode Receive Data Legend 1
614	
Figure 25-12	Multiprocessor Mode Receive Data Legend 2
614	
Figure 25-13	Smart Card Connection Schematic
615	
Figure 25-14	Smart Card Mode Synchronization Timing and Sampling Timing Diagram
616	
Figure 25-15	Smart Card Mode Synchronization Timing and Sampling Timing Diagram
617	
Figure 25-16	Smart Card Mode Sending Data Example
618	
Figure 25-17	Smart Card Mode Receive Data Legend
619	
Figure 25-18	Clock Synchronization Mode Data Format
620	
Figure 25-19	Clock Synchronization Mode Sending Data Legend 1
622	
Figure 25-20	Clock synchronization mode sending data example 2
622	
Figure 25-21	Clock Synchronization Mode Receive Data Legend 1
623	
Figure 25-22	Clock Synchronization Mode Received Data Legend 2
624	
Figure 26-1.....	I ² C system block diagram
640	

Figure 26-2	Example of the structure of the I ² C bus
	641
Figure 26-3	Timing diagram of the I ² C bus
	642
Figure 26-4	Data Format of I ² C Bus
	643
Figure 26-5	Timing diagram of host sending data in bit address format (example).....
	644
Figure 26-6	Timing diagram for host receiving data in bit address format (example).....
	645
Figure 26-7	Slave transmit mode timing diagram for bit address format (example).....
	646
Figure 26-8	Bit Address Format Slave Receive Mode Timing Diagram (Example).....
	647
Figure 26-9	SCL Synchronization Timing.....
	648
Figure 26-10	Slave Transmit Timing Diagram (1)
	649
Figure 26-11	Slave Transmit Timing Diagram (2)
	650
Figure 26-12	Timing when 7-bit address format is selected
	652
Figure 26-13	Timing when 10-bit address format is selected
	653
Figure 26-14	Block diagram of digital filtering circuit.....
	660
Figure 27-1	System Block Diagram
	681
Figure 27-2	Host Mode Structure
	683
Figure 27-3	Three-Wire Clock Synchronization Operation
	683
Figure 27-4	Data Format
	685
Figure 27-5	Data transfer format diagram (CPHA=0)
	686
Figure 27-6	Data Transfer Format (CPHA=1)
	687
Figure 27-7	Full Duplex Synchronous Serial Communication
	688
Figure 27-8	Transmit-only communication
	689

Figure 27-9	Parity Check Flow
696	
Figure 27-10	Overload Error Handling
698	
Figure 27-11	Diagram of the action when the clock auto-stop function is enabled (CPHA=1)
699	
Figure 27-12	Diagram of the action when the clock auto-stop function is enabled (CPHA=0)
699	
Figure 27-13	Parity error
700	
Figure 28-1 Module Composition of	QSPI
711	
Figure 28-2	Default area setting and AHB bus space memory mapping relationship
712	
Figure 28-3	QSPI-ROM Space Memory Image Map
713	
Figure 28-4.....	Extended SPI Protocol Action Schematic 1 (Fast Read Mode)
714	
Figure 28-5.....	Extended SPI Protocol Action Schematic 2 (4-Wire Input/Output Fast Read Mode)
714	
Figure 28-6	Schematic diagram of 2-wire SPI protocol action (fast read mode)
715	
Figure 28-7	Four-wire SPI Protocol Action Schematic (Fast Read Mode)
715	
Figure 28-8	Basic timing diagram of the serial interface
	716
Figure 28-9 Output clock duty cycle correction diagram when HCLK triplet frequency is selected for the reference clock	718

Figure 28-10	QSSL Build Time Configuration Diagram
719	
Figure 28-11	QSSN Hold Time Configuration Diagram
719	
Figure 28-12	Data reception delay diagram
720	
Figure 28-13	Standard Read Bus Cycle Diagram
722	
Figure 28-14	Fast Read Bus Cycle Diagram
723	
Figure 28-15 Diagram of Fast Read Bus Cycle with XIP Mode	Selected
723	
Figure 28-16	2-wire output fast read bus cycle diagram
724	
Figure 28-17 Diagram of fast read bus cycle with 2-wire output for XIP mode	selection
724	
Figure 28-18	2-wire input/output fast read bus cycle diagram
725	
Figure 28-19 Schematic diagram of a 2-wire input-output fast read bus cycle with XIP mode	selected
725	
Figure 28-20	Four-wire output fast read bus cycle diagram
726	
Figure 28-21 Schematic diagram of a four-wire output fast read bus cycle with XIP mode	selected
726	
Figure 28-22	Four-wire input/output fast read bus cycle diagram
727	
Figure 28-23 Schematic diagram of a four-wire input-output fast read bus cycle with XIP mode	selected
728	
Figure 28-24	Diagram of entering 4-Byte mode command bus cycle
728	
Figure 28-25.....	Exit 4-Byte Mode Instruction Bus Cycle Diagram
729	
Figure 28-26	Schematic diagram of the
write permission instruction bus cycle.....	729
Figure 28-27 Schematic	of a single flash data read operation with independent conversion
730	
Figure 28-28	Schematic diagram of
data reading operation when the pre-reading function is active	731

Figure 28-29 Schematic of Data Read Operation	Using QSPI Bus Cycle Extension Function
732	
Figure 28-30.....	XIP mode control schematic
732	
Figure 29-1	I2S system block diagram
749	
Figure 29-2	I2S Philips protocol waveform (16/32-bit full precision)
751	
Figure 29-3.....	I2S Philips protocol waveform (16-bit data encapsulated in a 32-bit frame)
752	
Figure 29-4	I2S Philips protocol waveform (24-bit data encapsulated in a 32-bit frame)
752	
Figure 29-5.....	I2S MSB protocol waveform (16/32-bit full precision)
752	
Figure 29-6	I2S MSB protocol waveform (16-bit data encapsulated in a 32-bit frame)
753	
Figure 29-7	I2S MSB protocol waveform (24-bit data encapsulated in a 32-bit frame)
753	
Figure 29-8.....	I2S LSB protocol waveform (16/32-bit full precision)
753	
Figure 29-9	I2S LSB protocol waveform (16-bit data encapsulated in a 32-bit frame)
754	
Figure 29-10	I2S LSB protocol waveform (24-bit data encapsulated in a 32-bit frame)
754	
Figure 29-11.....	I2S PCM protocol waveform (16/32-bit full precision)
755	
Figure 29-12.....	I2S PCM protocol waveform (16-bit data encapsulated in a 32-bit frame)
755	
Figure 29-13.....	I2S PCM protocol waveform (24-bit data encapsulated in a 32-bit frame)
756	
Figure 29-14	Audio Sampling Frequency Definition
757	

Figure 29-15	Clock Generator Architecture
757	
Figure 29-16	Host only receives temporary stop reception
762	
Figure 29-17 PCM Short Frame Host Sending Pause and Resend Method I	763
Figure 29-18 PCM Short Frame Host Sending Pause and Resend Method II	763
Figure 30-1	CAN System Block Diagram
773	
Figure 30-2	CAN Bit Time Definition Diagram
774	
Figure 30-3CAN TBUF register write send buffer and schematic	775
Figure 30-4CAN RBUF Register Read Receive Buffer Schematic	776
Figure 30-5CAN ACF Register Access Filter Group Schematic	776
Figure 30-6CAN LBMI and LBME Schematic	780
Figure 31-1	USBFS system block diagram
819	
Figure 31-2	USBFS host mode system build diagram
821	
Figure 31-3	USBFS device mode system build diagram
825	
Figure 31-4	USBFS Dynamic Update USBFS_HFIR Register Schematic
830	
Figure 31-5 Schematic of FIFO architecture in	USBFS host mode
831	
Figure 31-6 Schematic of FIFO architecture in	USBFS device mode
833	
Figure 32-1AES encryption flow diagram	936
Figure 32-2.....	HASH algorithm flow chart
943	
Figure 32-3.....	TRNG System Block Diagram
950	
Figure 34-1	CRC Module Block Diagram
964	
Figure 36-1	Commissioning Control System
1001	
Figure 36-2	Commissioning the control system
1002	

Figure 36-3 JTAG-DP to SW-DP Switching Timing
1003

Figure 36-4 TPIU Block Diagram
1010

Introduction (Overview)

This series is a high-performance MCU based on the ARM® Cortex®-M4 **32-bit** RISC CPU, operating at up to 200MHz. The Cortex-M4 core integrates a floating-point unit (FPU) and DSP for single-precision floating-point arithmetic operations, supports all ARM single-precision data processing instructions and data types, and supports the full DSP instruction set. The core integrates an MPU unit and overlays a DMAC dedicated MPU unit for secure system operation.

The HC32F460_F45x_A460 series integrates high-speed on-chip memory, including up to 512KB of Flash and up to 192KB of SRAM, and integrates a Flash access acceleration unit to enable single-cycle program execution on Flash by the CPU. The polled bus matrix supports multiple bus hosts to access memory and peripherals simultaneously to improve operational performance. Bus hosts include CPU, DMA, USB dedicated DMA, etc. In addition to the bus matrix, it supports inter-peripheral data transfer, basic arithmetic operations and event triggering, which can significantly reduce the CPU transaction processing load.

This series integrates a rich set of peripheral functions. It includes two independent **12bit** 2.5MSPS ADCs, one gain adjustable PGA, three voltage comparators (CMP) three multi-function **16bit** PWM timers (Timer6) supporting 6 complementary PWM outputs, three motor PWM timers (Timer4) supporting 18 complementary PWM outputs, six **16bit** general-purpose Timer (TimerA) supports 3 3-phase quadrature coded inputs and 48 Duty independent configurable PWM outputs, 11 serial communication interfaces (I2C/UART/SPI)1 QSPI interface, 1 CAN, 4 I2S with audio PLL support, 2 SDIO, 1 USB FS **Controller** with on-chip FS PHY support **Device/Host**.

This series supports wide voltage range (1.8-3.6V) wide temperature range (-40-105°C)and various low power modes. ultra-high speed mode ($\leq 200\text{MHz}$) high speed mode ($\leq 168\text{MHz}$) and ultra-low speed mode ($\leq 8\text{MHz}$)can be switched in Run mode and Sleep mode. Supports fast wake-up in low-power mode, up to 2us for STOP mode and up to 20us for Power Down mode.

Typical Applications

This series is available in 48pin,64pin,100pin LQFP packages, 32pin,48pin,60pin QFN packages, and 100pin VFBGA packages for automotive electronics, high performance motor inverter control, smart hardware, IoT connectivity modules, etc.

About this brochure

This manual introduces the functions, operations, and usage of the chip. For the specifications of the chip, please

refer to the corresponding "Data Sheet".

1 Memory Mapping (Mapping)

1.1 Memory Mapping

This MCU supports 4GB of linear address space with addresses from 0x0000_0000 to 0xFFFF_FFFF. See Table 1-1 for more information on memory mapping.

Table 1-1 Memory Mapping

Memory Classification		Start Address	End Address	Space size	Module*3	Protection*4	Description
System	System	0xE010_0000	0xFFFF_FFFF	511MB	Reserved		Custom Space
	Private	0xE00F_F000	0xE00F_FFFF	4KB	ROMTABLE		Debug control register area
	Peripheral	0xE004_2400	0xE00F_EFFF	755KB			
	External	0xE004_2000	0xE004_23FF	1KB			
	Bus	0xE004_1000	0xE004_1FFF	4KB			The MCU without ETM
	Private	0xE000_F000	0xE003_FFFF	196KB	SCS		System control space NVIC/MPU, etc.
	Peripheral	0xE000_E000	0xE000_EFFF	4KB			
	Internal	0xE000_3000	0xE000_DFFF	44KB			
	Bus	0xE000_2000	0xE000_2FFF	4KB			
	Private	0xE000_1000	0xE000_1FFF	4KB			
	Peripheral	0xE000_0000	0xE000_0FFF	4KB			
External Equipment	–	0xA000_0000	0xDFFF_FFFF	1024MB	Reserved		
External RAM	AHB5	0x9800_0000	0x9FFF_FFFF	128MB	QSPI		
	Clock: HCLK	0x6000_0000	0x97FF_FFFF	896MB	Reserved		
Peripherals	–	0x4400_0000	0x5FFF_FFFF	448MB	Reserved		Hosts other than CPU Reserved
		0x4200_0000	0x43FF_FFFF	32MB	PeriBitBand		
		0x4010_0000	0x41FF_FFFF	31MB	Reserved		
	AHB3 Clock: PCLK1	0x400C_0000	0x400F_FFFF	256KB	USBFS		

	AHB2	0x4007_0800	0x400B_FFFF	318KB	BLANK		
	Clocks:	0x4007_0400	0x4007_07FF	1KB	CAN		
	PCLK1	0x4007_0000	0x4007_03FF	1KB	SDIOC_2		

Memory Classification		Start Address	End Address	Space size	Module*3	Protection*4	Des crip tion
Peripherals	AHB4	0x4006_FC00	0x4006_FFFF	1KB	SDIOC_1		
	Clocks:						
	PCLK1	0x4006_0000	0x4006_FBFF	63KB	BLANK		
	AHB1	0x4005_5800	0x4005_FFFF	42KB	BLANK		
	Clock:	0x4005_5400	0x4005_57FF	1KB	PERIC		
	HCLK	0x4005_4800	0x4005_53FF	3KB	BLANK		
		0x4005_4000	0x4005_47FF	2KB	SYSC	With protection	
		0x4005_3800	0x4005_3FFF	2KB	GPIO		
		0x4005_3400	0x4005_37FF	1KB	DMA_2		
		0x4005_3000	0x4005_33FF	1KB	DMA_1		
		0x4005_2C00	0x4005_2FFF	1KB	DCU_4		
		0x4005_2800	0x4005_2BFF	1KB	DCU_3		
		0x4005_2400	0x4005_27FF	1KB	DCU_2		
		0x4005_2000	0x4005_23FF	1KB	DCU_1		
		0x4005_1000	0x4005_1FFF	4KB	INTC	With protection	
		0x4005_0C00	0x4005_0FFF	1KB	KEYSCAN		
		0x4005_0800	0x4005_0BFF	1KB	RAMIF	With protection	
		0x4005_0400	0x4005_07FF	1KB	BLANK		
		0x4005_0000	0x4005_03FF	1KB	DMPU	With protection	
	APB4	0x4004_EC00	0x4004_FFFF	5KB	BLANK		
	Clock:	0x4004_E800	0x4004_EBFF	1KB	I2C_3		
	PCLK3	0x4004_E400	0x4004_E7FF	1KB	I2C_2		
		0x4004_E000	0x4004_E3FF	1KB	I2C_1		
		0x4004_C400	0x4004_DFFF	7KB	BLANK		
		0x4004_C000	0x4004_C3FF	1KB	RTC	With protection	
		0x4004_A800	0x4004_BFFF	6KB	BLANK		
		0x4004_A400	0x4004_A7FF	1KB	OTS		
		0x4004_A000	0x4004_A3FF	1KB	CMP		
		0x4004_9800	0x4004_9FFF	2KB	BLANK		
		0x4004_9400	0x4004_97FF	1KB	SWDT	With protection	
		0x4004_9000	0x4004_93FF	1KB	WDT	With protection	
		0x4004_8800	0x4004_8FFF	2KB	BLANK		

		0x4004_8400 0x4004_8000	0x4004_87FF 0x4004_83FF	1KB 1KB	FCM MSTP	With protection	
APB3		0x4004_1400	0x4004_7FFF	27KB	BLANK		
Clock:		0x4004_1000	0x4004_13FF	1KB	TRNG	With protection	
PCLK4		0x4004_0800	0x4004_0FFF	2KB	BLANK		

Memory Classification	Start Address	End Address	Space size	Module*3	Protection*4	Des crip tion
	0x4004_0400	0x4004_07FF	1KB	ADC_2		
	0x4004_0000	0x4004_03FF	1KB	ADC_1		
APB2	0x4002_5000	0x4003_FFFF	108KB	BLANK		
Clock:	0x4002_4C00	0x4002_4FFF	1KB	Timer4_3		
PCLK1	0x4002_4800	0x4002_4BFF	1KB	Timer4_2		
	0x4002_4400	0x4002_47FF	1KB	Timer0_2		
	0x4002_4000	0x4002_43FF	1KB	Timer0_1		
	0x4002_2800	0x4002_3FFF	6KB	BLANK		
	0x4002_2400	0x4002_27FF	1KB	I2S_4		
	0x4002_2000	0x4002_23FF	1KB	I2S_3		
	0x4002_1800	0x4002_1FFF	2KB	BLANK		
	0x4002_1400	0x4002_17FF	1KB	USART_4		
	0x4002_1000	0x4002_13FF	1KB	USART_3		
	0x4002_0800	0x4002_0FFF	2KB	BLANK		
	0x4002_0400	0x4002_07FF	1KB	SPI_4		
	0x4002_0000	0x4002_03FF	1KB	SPI_3		
APB1	0x4001_E800	0x4001_FFFF	6KB	BLANK		
Clock:	0x4001_E400	0x4001_E7FF	1KB	I2S_2		
PCLK1	0x4001_E000	0x4001_E3FF	1KB	I2S_1		
	0x4001_D800	0x4001_DFFF	2KB	BLANK		
	0x4001_D400	0x4001_D7FF	1KB	USART_2		
	0x4001_D000	0x4001_D3FF	1KB	USART_1		
	0x4001_C800	0x4001_CFFF	2KB	BLANK		
	0x4001_C400	0x4001_C43F	1KB	SPI_2		
	0x4001_C000	0x4001_C3FF	1KB	SPI_1		
	0x4001_8000	0x4001_BFFF	16KB	Timer6		Counting clock :P CLK0
	0x4001_7C00	0x4001_7FFF	1KB	EMB		
	0x4001_7400	0x4001_7BFF	2KB	BLANK		
	0x4001_7000	0x4001_73FF	1KB	Timer4_1		
	0x4001_6800	0x4001_6FFF	2KB	BLANK		
	0x4001_6400	0x4001_67FF	1KB	TimerA_6		
	0x4001_6000	0x4001_63FF	1KB	TimerA_5		
	0x4001_5C00	0x4001_5FFF	1KB	TimerA_4		
	0x4001_5800	0x4001_5BFF	1KB	TimerA_3		
	0x4001_5400	0x4001_57FF	1KB	TimerA_2		
	0x4001_5000	0x4001_53FF	1KB	TimerA_1		

Memory Classification		Start Address	End Address	Space size	Module*3	Protection*4	Des crip tion
SRAM	AHB3 Clock: PCLK1	0x4001_0C00	0x4001_4FFF	17KB	BLANK		
		0x4001_0800	0x4001_0BFF	1KB	AOS		Internal trigger event register area
		0x4001_0400	0x4001_07FF	1KB	EFM	With protection	
		0x4001_0000	0x4001_03FF	1KB	BLANK		
		0x4000_9000	0x4000_FFFF	28KB	BLANK		
		0x4000_8C00	0x4000_8FFF	1KB	CRC	With protection	
		0x4000_8800	0x4000_8BFF	1KB	BLANK		
		0x4000_8400	0x4000_87FF	1KB	HASH256	With protection	
		0x4000_8000	0x4000_83FF	1KB	AES128	With protection	
	-	0x4000_0000	0x4000_7FFF	32KB	Reserved		
	SRAM Clock: HCLK	0x2400_0000	0x3FFF_FFFF	448MB	Reserved		
		0x2200_0000	0x23FF_FFFF	32MB	SRAMBitBand		Hosts other than CPU Reserved
		0x2010_0000	0x21FF_FFFF	31MB	Reserved		
		0x200F_1000	0x200F_FFFF	60KB	BLANK		
		0x200F_0000	0x200F_0FFF	4KB	Ret_SRAM		
		0x2002_7000	0x200E_FFFF	804KB	BLANK		
	CODE	0x2002_0000	0x2002_6FFF	28KB	SRAM3		ECCRAM
		0x2001_0000	0x2001_FFFF	64KB	SRAM2		
		0x2000_0000	0x2000_FFFF	64KB	SRAM1		
		0x1FFF_8000	0x1FFF_FFFF	32KB	SRAMH		
CODE	SRAM Clock: HCLK	0x0318_0000	0x1FFF_7FFF	462.4MB	BLANK		
		0x0317_FFE0	0x0317_FFFF	32B	Data Security Protection		For configuring data security protection
	Flash clock: HCLK	0x0210_0000	0x0317_FFDF	16.5MB	BLANK		
		0x0208_0000	0x020F_FFFF	512KB	REMAP1		Address Remapping Area 1
		0x0200_0000	0x0207_FFFF	512KB	REMAP0		Address Remapping Area

							0
-	0x0008_0000	0x01FF_FFFF	31.5MB	BLANK			
Flash Clock : HCLK	0x0000_0000	0x0007_FFFF	512KB	Embedded Flash*5			

*1 Please refer to the ARM Cortex-M4 instruction manual "Memory System".

*2 Please refer to the Bus chapter for bus descriptions.

*3 Reserved: Access to the bus causes a bus error; BLANK: Write access is invalid, read access reads 0.

*4 Modules with protection support only CPU privileged mode access when the protection is active.
Refer to the DMPU chapter for specific registers and descriptions.

*5 In the 256KB product, the Flash address is 0x0000_0000~0x0003_FFFF.

1.2 External Space Mapping

QSPI space is divided into 2 segments, QSPI Refer to Table 1-2 for the allocation relationship.

Table 1-2 QSPI Address Space Allocation

QSPI	0x9800_0000	0x9FFF_FFFF	128MB	QSPI I/O Registers	0x9C00_0000	0x9FFF_FFFF	64MB
				External QSPI Devices	0x9800_0000	0x9BFF_FFFF	64MB

1.3 Bitspace

The Cortex™-M4F memory mapping consists of two bit segment regions. These regions map each word in the memory alias region to the corresponding bit in the memory bit segment region. Writing a word in the alias region is equivalent to performing a read-modify-write operation on the target bit in the bit segment region.

In this MCU, peripheral registers and SRAM are mapped to a bit-segment area, which allows for single bit-segment read and write operations. These operations are only available for Cortex™-M4F accesses and are not available for other bus master interfaces such as DMA.

1.4 Address Remapping

The MCU provides two remapped addresses, and the memory address remapping function can be configured, and the source address can be set to the main flash FLASH address and the high-speed SRAM address.

Remap address 0:

0x0200_0000~0x0208_0000 (depending on the remapping space
MMF_REMCRO/1.RMSIZE[4:0])

Remap Address 1:

0x0208_0000~0x0210_0000 (depending on the remapping space)

MMF_REMCR0/1.RMSIZE[4:0]

When the remapping function is active, the addresses correspond to those shown in Table 1-3.

Table 1-3 Example of destination address configuration

Register Setting	Remap Address (CPU address - CPUADDR[31:0])	Source Addresses		
		High 3 Address	Medium Address	Low Address
RMSIZE[4:0]=01110 situation (Remapping space: 16K)	0x0200_0000~0x0200_3FFF	All 0	RMTADDR[16:2]	CPUADDR[13:0]
RMSIZE[4:0]=01111 case (Remapping space: 32K)	0x0200_0000~0x0200_7FFF	All 0	RMTADDR[16:3]	CPUADDR[14:0]
RMSIZE[4:0]=10000 cases (Remapping space: 64K)	0x0208_0000~0x0208_FFFF	All 0	RMTADDR[16:4]	CPUADDR[15:0]
RMSIZE[4:0]=10001 situation (Remapping space: 128K)	0x0208_0000~0x0209_FFFF	All 0	RMTADDR[16:5]	CPUADDR[16:0]

For example, using the Remap Address 0 function, set the source address as the main flash FLASH address 0x0000_8000, the remap space 32K, and the register MMF_REMCR0 should be set to 0x8000_800F.

Using the remapping address 1 function, set the source address to the high-speed SRAM address 0x1FFF_8000, the remapping space to 16K, and the register MMF_REMCR1 to 0x9FFF_800E.

Note: The starting address of the source address should be set to an integer multiple of the remap space.

1.5 Remap Register

There are three registers in the remapping

module. The address space is as follows:

Register base address: 0x4001_0500

Table 1-4 Register List

Register Name	Symbols	Offset Address	Bit width	Reset value
Access protection register	MMF_REMPRT	0x0000	32	0x0000_0000
Remap control register 0	MMF_REMCR0	0x0004	32	0x0000_0000
Remap control register 1	MMF_REMCR1	0x0008	32	0x0000_0000

1.5.1 Access protection register (MMF_REMPRT)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
MMF_REMPRT[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~16	Reserved	-	Read "0", write "0" when writing	R
b15~0	MMF_REMPRT[15:0]	Protection register	Registers MMF_REMCR0 and MMF_REMCR1 are write-protected: Write 0x0123 to MMF_REMPRT[15:0] followed by 0x3210 to unprotect; When the registers MMF_REMCR0 and MMF_REMCR1 are write-protected, the read register is 0 When registers MMF_REMCR0 and MMF_REMCR1 are unprotected from writing, the read send The memory is 1	R/W

1.5.2 Remap control register (MMF_REMCRx|x=0, 1)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EN	-														RMTADDR[16:4]
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RMTADDR[3:0]				-						RMSIZE[4:0]					

position	Marker	Place Name	Function	Reading and writing
b31	EN	Remap valid bits	0: Remapping is invalid 1: Remapping is valid	R/W
b30~29	Reserved	-	Read "0", write "0" when writing	R
b28~12	RMTADDR[16:0]	Source Address	The number of valid bits is related to the RMSIZE[4:0] setting. The settings can be found in Table 1-3.	R/W
b11~b5	Reserved	-	Read "0", write "0" when writing	R
b4~b0	RMSIZE[4:0]	Remap space	Set remapping space 00000~01011: Reserved, set forbidden 01100: 4KB 01101: 8KB 01110: 16KB 01111: 32KB 10000: 64KB 10001: 128KB 10010: 256KB 10011: 512KB (256KB product setting prohibited) 10100 ~ 11111 : Reserve, set forbidden	R/W

2 Bus Architecture (BUS)

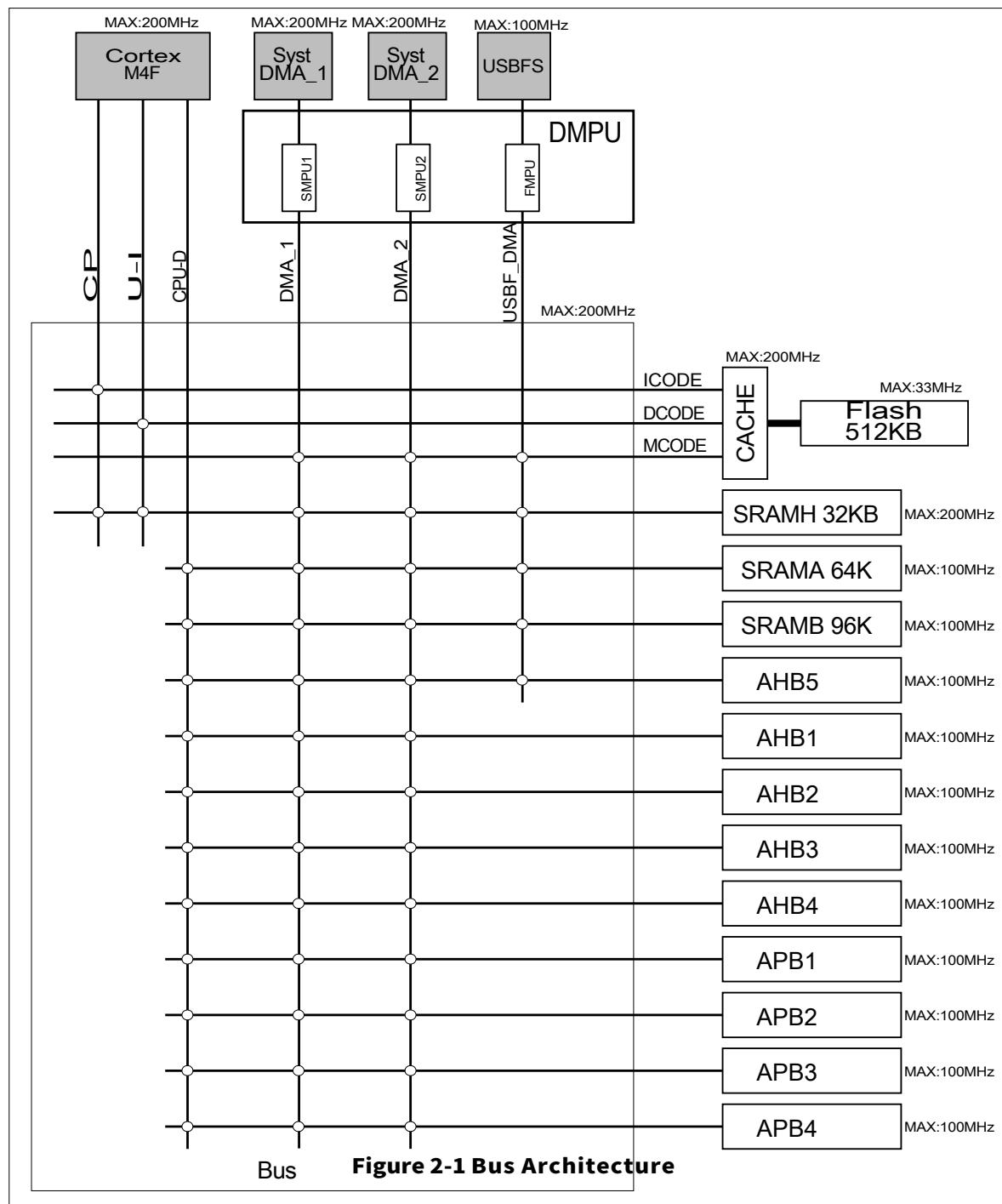
2.1 Overview

The master system consists of a 32-bit multi-layer AHB bus matrix that allows interconnection of the following host and slave buses:

- Host Bus
 - Cortex-M4F Core CPU-I Bus, CPU-D Bus, CPU-S Bus
 - System DMA_1 bus, System DMA_2 bus
 - USBFS_DMA Bus
- Slave Bus
 - Flash ICODE Bus
 - Flash DCODE Bus
 - Flash MCODE bus (bus for hosts other than the CPU to access Flash)
 - High-speed SRAMH (SRAMH 32KB) bus
 - System SRAMA (SRAM1 64KB) Bus
 - System SRAMB (SRAM2 64KB, SRAM3 28KB, Ret_SRAM 4KB) Bus
 - APB1 Peripheral Bus (EMB/Timers/SPI/USART/I2S)
 - APB2 Peripheral Bus (Timers/SPI/USART/I2S)
 - APB3 Peripheral Bus (ADC/PGA/TRNG)
 - APB4 Peripheral Bus (FCM/WDT/CMP/OTS/RTC/WKTM/I2C)
 - AHB1 Peripheral bus (KEYSCAN/INTC/DCU(GPIO/SYSC)
 - AHB2 Peripheral Bus (CAN/SDIOC)
 - AHB3 Peripheral Bus (AES/HASH/CRC/USBFS)
 - AHB4 Peripheral Bus (SDIOC)
 - AHB5 Peripheral Bus (QSPI)

With the help of the bus matrix, efficient concurrent access from the host bus to the slave bus is possible.

2.2 Bus Architecture



The bus matrix is used for access arbitration management between the host buses. Arbitration is performed using a round-robin scheduling algorithm.

- **CPU-I Bus**

The instruction bus of the M4F kernel, through which the CPU gets instructions. The accesses are to the Flash and SRAMH containing the code.

- **CPU-D Bus**

The M4F core has a data bus through which the CPU performs immediate number loading and debug accesses. The accesses are to Flash and SRAMH containing code or data.

- **CPU-S Bus**

The system bus of the M4F kernel, through which the CPU accesses peripherals or the system SRAM, and through which instructions and immediate numbers can be fetched (less efficiently than through **the CPU-I** and CPU-D buses). The accesses are to SRAMA/SRAMB/all peripherals and the AHB5 external expansion space.

- **DMA_1 bus, DMA_2 bus**

System DMA_1/System DMA_2 Dedicated bus, DMA_1/DMA_2 accesses data memory and peripherals via this bus for Flash/SRAMH/SRAMA/SRAMB/all peripherals and AHB5 external expansion space.

- **USBFS_DMA Bus**

USBFS has a dedicated DMA bus, through which USBFS accesses all memory spaces. The accesses are to Flash/SRAMH/SRAMA/SRAMB/AHB5 external expansion space.

2.3 Bus functions

The bus is responsible for implementing read and write accesses from the master to the slave. When the host module operates at a higher frequency than the slave module (e.g., CPU-S accesses RTC), the bus automatically performs downsampling and synchronization. When the host module operates at a lower frequency than the slave module (e.g. USBFS_DMA accesses SRAMH), the bus is automatically up-coded and synchronized.

With the bus matrix, each access can be performed simultaneously when the access targets of different host buses do not conflict. For example, **CPU-I** accesses Flash, CPU-D accesses SRAMH, CPU-S accesses APB peripherals, DMA_1 accesses SRAMA, DMA_2 accesses SRAMB, and USBFS_DMA accesses the external expansion space of AHB5, and these accesses can be performed simultaneously.

3 Reset control (RMU)

3.1 Introduction

The chip is configured with 14 reset methods.

- Power-On Reset (POR)
- NRST Pin Reset (NRST)
- Undervoltage Reset (BOR)
- Programmable Voltage Detect 1 Reset (PVD1R)
- Programmable voltage detection 2 reset (PVD2R)
- Watchdog Reset (WDTR)
- Dedicated watchdog reset (SWDTR)
- Power-down wake-up reset (PDRST)
- Software Reset (SRST)
- MPU Error Reset (MPUR)
- RAM Parity Reset (RAMPR)
- RAMECC reset (RAMECCR)
- Clock abnormal reset (CKFER)
- External high-speed oscillator abnormal stop reset (XTALER)

3.2 Reset mode and reset flag bit

The reset method and the generation conditions are shown in Table 3-1.

Table 3-1 Reset mode and generation conditions

Reset method	Generation conditions
Power-on reset	VCC Power-up
NRST pin reset	NRST pin input low
Undervoltage reset	VCC voltage drops below VBOR voltage
Programmable voltage detection 1 reset	VCC voltage drops below PVD1 voltage
Programmable voltage detection 2 reset	VCC voltage drops below PVD2 voltage
Watchdog Reset	Watchdog timer generates refresh error or overflow error
Dedicated watchdog reset	Dedicated watchdog has a refresh error or overflow error
Power-down wake-up reset	By setting the reset generated by power-down mode, the kernel wakes up from the reset state after a power-down wake-up event
Software Reset	Set the reset register bit (ARM register AIRCR.SYSRESETREQ bit)
MPU Error Reset	Reset generated by MPU access error
RAM Parity Reset	RAM Reset generated when a parity error occurs
RAM ECC error reset	Reset generated when an ECC error occurs in RAM
Clock frequency abnormal reset	When the clock frequency monitoring function (FCM) detects a clock period error
External high-speed oscillator abnormal stop reset	Reset generated when external high-speed oscillator stops abnormally

When a reset occurs, the chip sets the corresponding reset flag bit according to the reset mode, and the reset flag bits are shown in Table 3-2. For example, if a pin reset occurs, the pin reset flag bit PINRF is set to 1. After PINRF is set, PINRF can be cleared by writing CLRF.

Table 3-2 Reset mode and reset flag

Reset flag	Reset method							
	RAM ECC	RAM	MPU	Power-down wake-up reset	Dedicated watchdog reset	Watchdog Reset	Voltage detection reset	Undervoltage reset
Power-on reset flag (RMU_RSTF0.PORF)	✓	-	-	-	-	-	-	-
Pin reset flag (RMU_RSTF0.PINRF)	-	✓	-	-	-	-	-	-
Undervoltage reset flag (RMU_RSTF0.BORF)	-	-	✓	-	-	-	-	-
Programmable Voltage Detect 1 Reset Flag (RMU_RSTF0.PVD1RF)	-	-	-	✓	-	-	-	-
Programmable Voltage Detect 2 Reset Flag (RMU_RSTF0.PVD2RF)	-	-	-	-	✓	-	-	-
Watchdog reset flag (RMU_RSTF0.WDRF)	-	-	-	-	-	✓	-	-
Dedicated watchdog reset flag (RMU_RSTF0.SWDRF)	-	-	-	-	-	-	✓	-
Power-down wake-up reset flag (RMU_RSTF0.PDRF)	X	-	-	-	-	-	✓	-
Software reset flag (RMU_RSTF0.SWRF)	X	X	X	-	-	-	X	✓
MPU error reset (RMU_RSTF0.MPUERF)	X	X	X	-	-	-	X	-
RAM Parity Error Reset (RMU_RSTF0.RAPERF)	X	X	X	-	-	-	X	-
RAM ECC reset (RMU_RSTF0.RAECRF)	X	X	X	-	-	-	X	-
Clock frequency abnormal reset (RMU_RSTF0.CKFERF)	X	X	X	-	-	-	X	-
External high-speed oscillator abnormal stop reset (RMU_RSTF0.XTALERF)	X	X	X	-	-	-	X	-

✓: Set X: Clear -: No change

3.3 Reset Timing

3.3.1 Power-on reset

A power-on reset is an internal reset caused by the power-on reset circuit, and the timing is as shown in Figure 3-1. A power-on reset is generated when the NRST pin is set high and the power is turned on. The VCC voltage is higher than the power-on reset voltage V_{POR} and after a certain time (T_{RSTPOR}) the internal reset of the chip is released and the CPU starts to execute the code. When a power-on reset is generated, the power-on reset flag RMU_RSTF0.PORF is set. Please refer to [5.3.1 Power-on Reset / Power-down Reset Action Description] for details of the power-on reset.

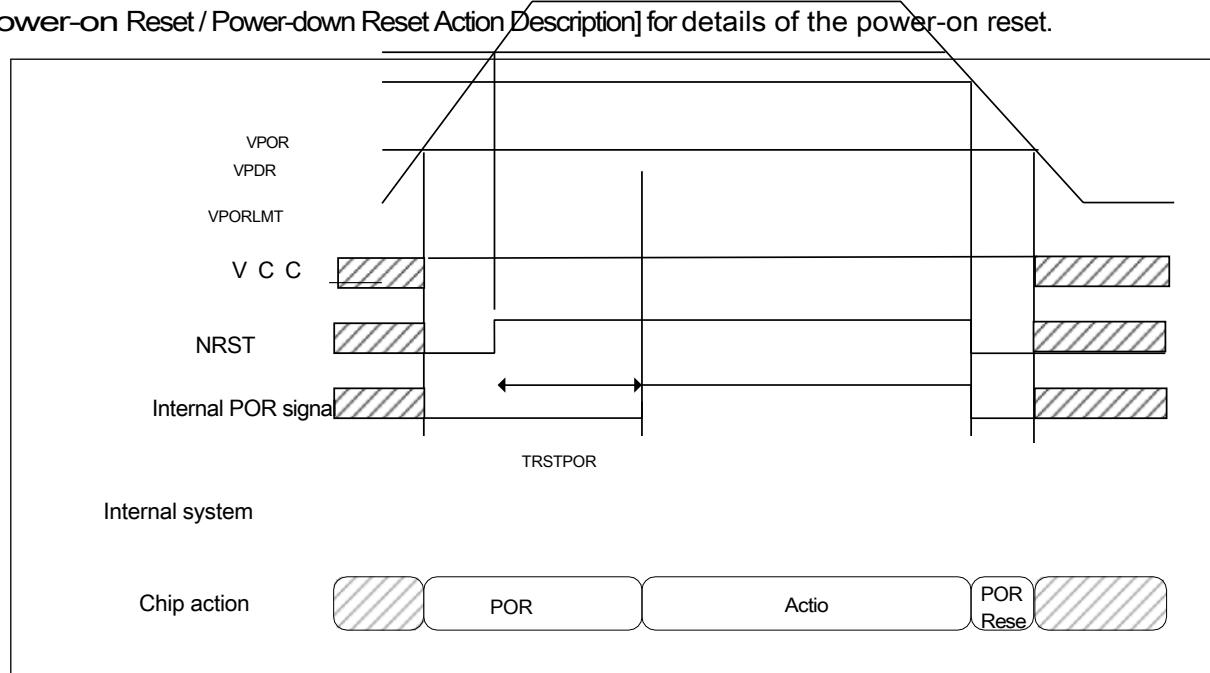


Figure 3-1 Power-on Reset

3.3.2 NRST pin

reset

A pin reset is a reset caused by the NRST pin being driven low, and the reset timing is shown in Figure 3-2. After the NRST pin maintains a low level above the width of T_{NRST} , the internal reset is released after a certain internal reset time (T_{INRST})

When NRST pin reset is generated, the pin reset flag RMU_RSTF0._PINRF is set.

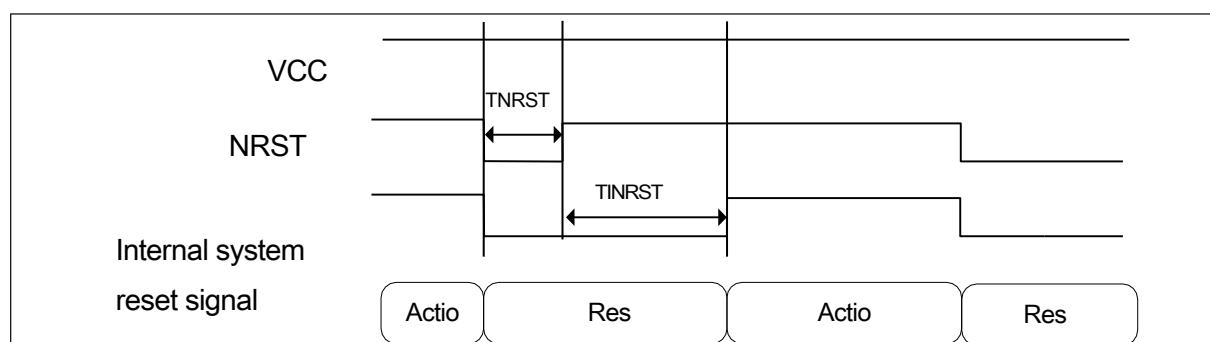


Figure 3-2 NRST Reset Timing

3.3.3 Undervoltage reset

Undervoltage reset is an internal reset caused by the voltage monitoring circuit, and the timing is as shown in Figure 3-3. After the undervoltage is set to reset enable by the ICG register, RMU_RSTF0.VBORF is set if the VCC voltage is lower than the monitoring voltage V_{BOR} . When the VCC voltage is higher than the monitoring voltage V_{BOR} , the reset time (T_{RSTBOR}) of V_{BOR} will be released.

For the reset setting of undervoltage, please refer to [5.3.2 Undervoltage Reset (BOR) Description].

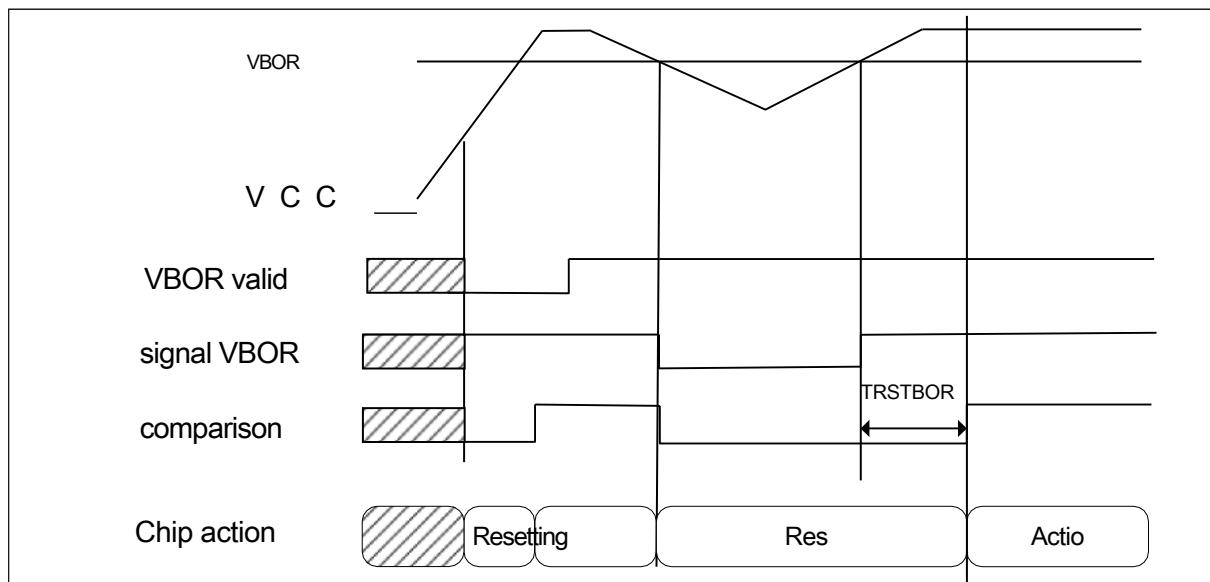


Figure 3-3 Undervoltage Reset

3.3.4 Programmable voltage detection 1 reset, programmable voltage detection 2 reset

Reset caused by the voltage monitoring circuit when Programmable Voltage Detect 1 and Programmable Voltage Detect 2 are reset.

After programmable voltage detection 1 is active and set to reset enable, if VCC is below the monitoring voltage of programmable voltage detection 1, programmable voltage detection 1 reset is generated and RMU_RSTF0.PVD1F is set. After the VCC voltage is higher than the monitoring voltage of programmable voltage detection 1, the reset is released after the reset time of PVD1 (T_{IPVD1}).

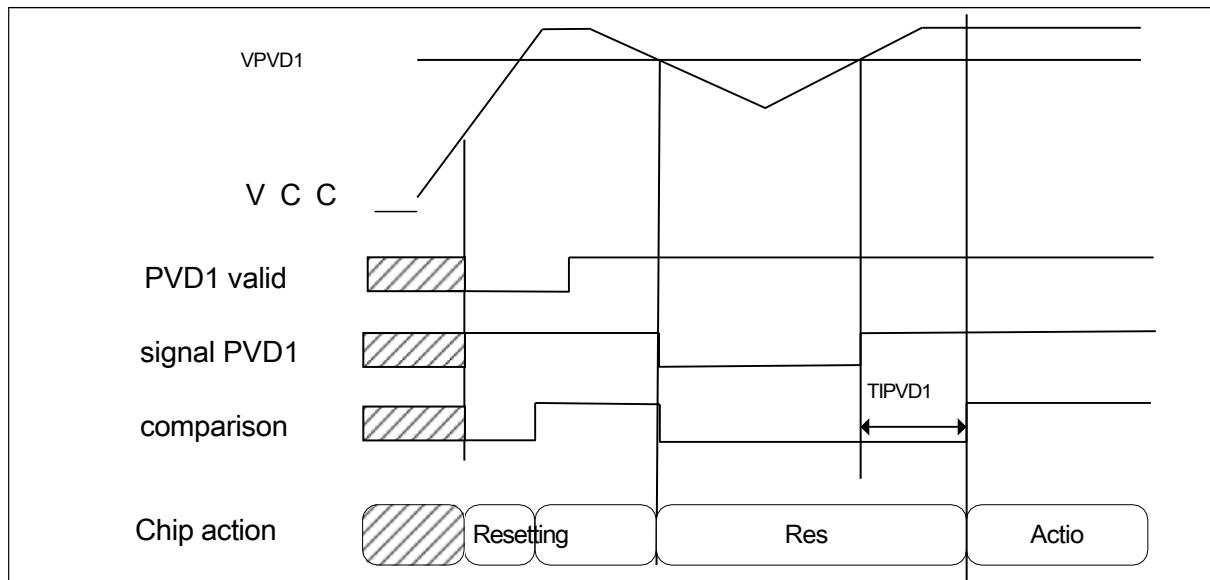


Figure 3-4 Programmable voltage detection 1 reset

After Programmable Voltage Detect 2 is active and set to reset enable, if VCC is lower than the monitoring voltage of Programmable Voltage Detect 2, a Programmable Voltage Detect 2 reset is generated and RMU_RSTF0.PVD2F is set. When VCC voltage is higher than the monitoring voltage of programmable voltage detection 2, the reset is released after the reset time of PVD2 (T_{IPVD2}).

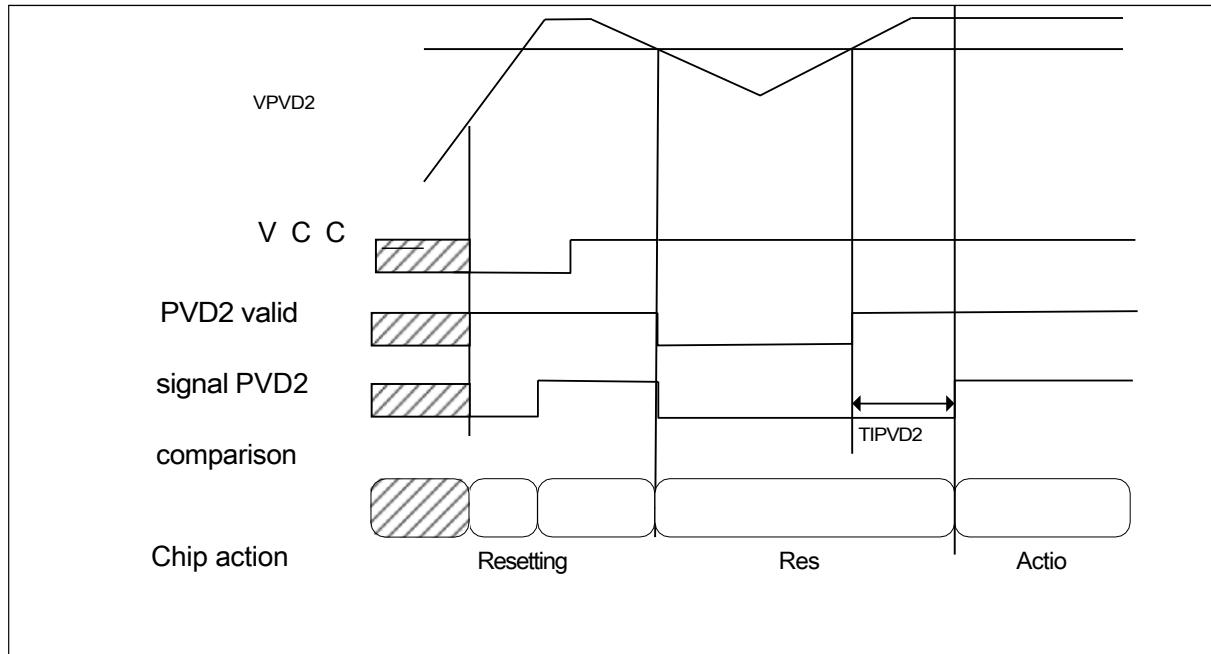


Figure 3-5 Programmable Voltage Detect 2 Reset

Refer to [5.3 Power supply voltage detection unit (PVD)] ~~chip~~ for the reset setting of programmable voltage detection 1 and programmable voltage detection 2.

3.3.5 Watchdog reset, dedicated watchdog reset

The watchdog reset is an internal reset caused by the watchdog timer, and the dedicated watchdog reset is an internal reset caused by the dedicated watchdog timer, and the reset timing is shown in Figure 3-6.

After setting the watchdog reset to be valid, a watchdog reset is generated when the watchdog timer generates an underflow or when a write operation is not performed during the refresh allowance period. The watchdog reset sets RMU_RSTF0.WDRF to bit. After a watchdog reset is generated, the chip is released from reset after the internal reset time T_{RIPT} .

After setting the dedicated watchdog reset to be valid, a watchdog reset is generated when the dedicated watchdog timer generates an underflow or when a write operation is not performed during the refresh permission period. The dedicated watchdog reset sets RMU_RSTF0_SWDRF to bit. After a dedicated watchdog reset is generated, the chip is released from reset after the internal reset time T_{RIPT} .

For details of watchdog reset and dedicated watchdog reset, please refer to [Watchdog Counter (WDT/SWDT)]

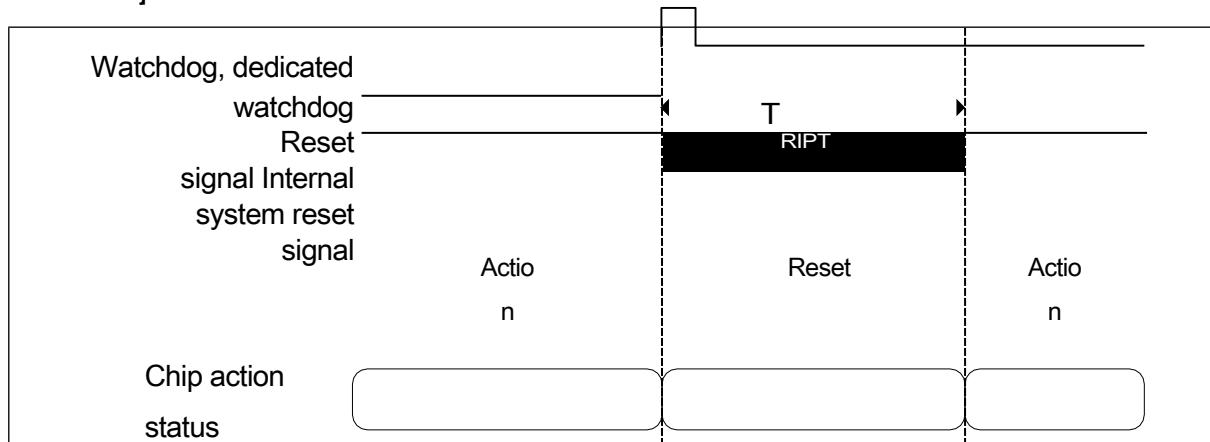


Figure 3-6 Watchdog and Dedicated Watchdog Reset

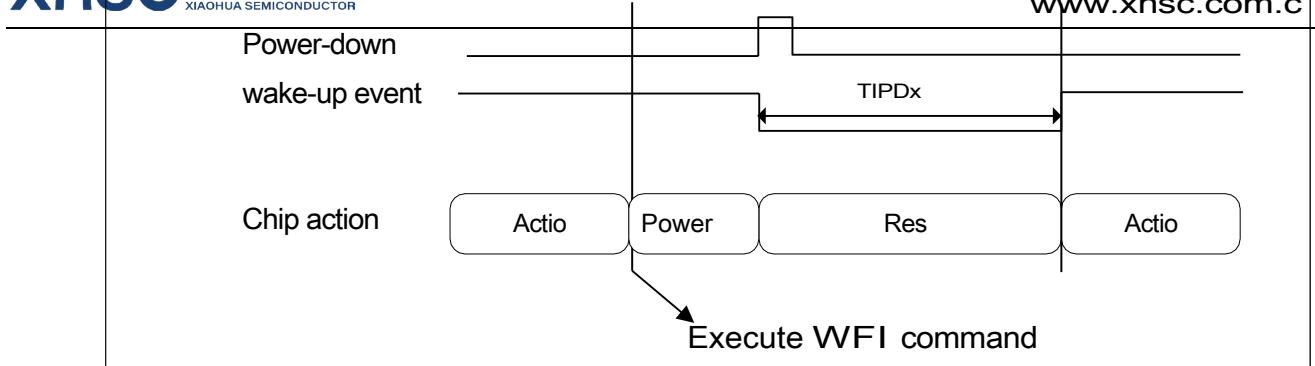
3.3.6 Power-

down wake-

up reset

Power-down wake-up reset is an internal reset generated when the chip executes the WFI command when PWC_PWRC0.PWDN is set to 1 and enters the power-down mode, and is released from the power-down mode by the power-down mode wake-up event, as shown in Figure 3-7. The power-down wake-up reset is released after the power-down mode is released and the return time (T_{IPDX} , $x=1,2,3,4$) is elapsed. The return time varies depending on the specific power-down mode set, with a minimum in power-down mode 1 and a maximum in power-down mode 3.

For details of power-down wake-up reset, please refer to [5.4.4 Powerdownmode].

**Figure 3-7 Power-down wake-up reset**

3.3.7 Software Reset

A software reset is generated by writing the SYSRESETREQ bit of the ARM register AIRCR. When a software reset is generated, the RMU_RSTF0.SWRF bit is set. After the internal reset time T_{RIPT} , the chip is released from reset.

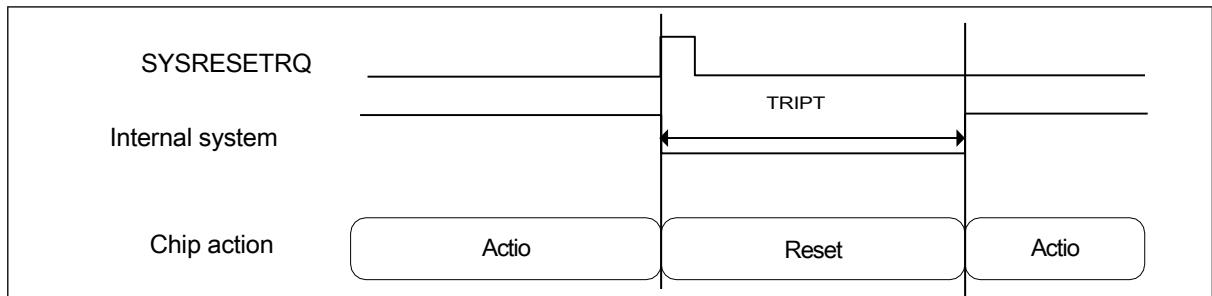


Figure 3-8 Software Reset

3.3.8 MPU Error

Reset

MPU error reset sets RMU_RSTF0.MPUERF, the timing is as shown in Figure 3-9. After the internal reset time T_{RIPT} , the chip is released from reset.

For the setting of MPU error reset, please refer to [13 Memory Protection Unit (MPU)]

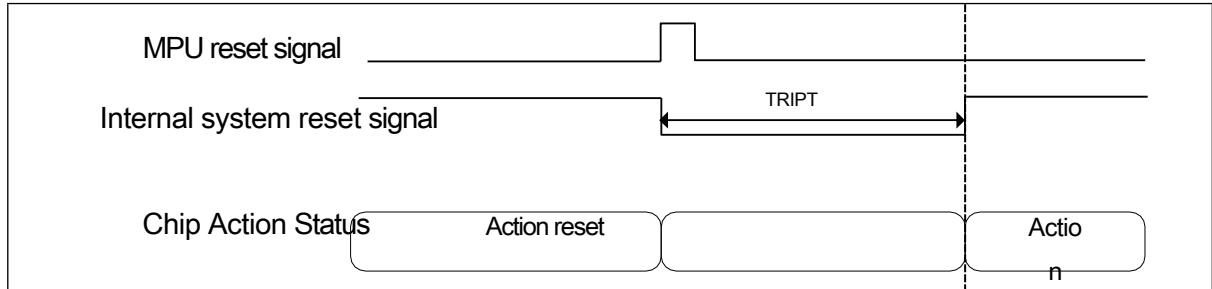


Figure 3-9 MPU Error Reset

3.3.9 RAM Parity Reset

When a RAM parity error occurs, a RAM parity reset is generated, with the timing sequence shown in Figure 3-10. A RAM parity error sets RMU_RSTF0.RAPERF. After the internal reset time T_{RIPT} , the chip is released from reset.

For the setting of RAM parity error reset, please refer to [Built-in SRAM (SRAM)]

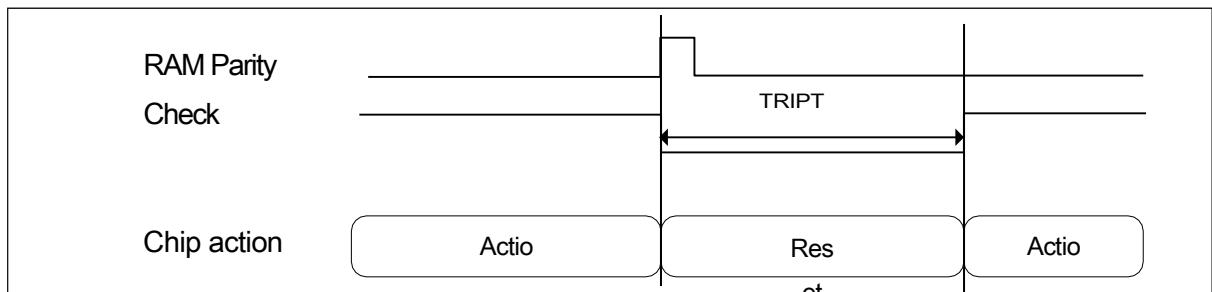


Figure 3-10 RAM Parity Reset

3.3.10 RAMECC reset

A RAMECC reset is generated when a RAMECC checksum error occurs, as shown in the timing sequence at 3-11. The RAMECC reset sets the RMU_RSTF0.RAECRF. After the internal reset time T_{RIPT} , the chip is released from reset.

For the setting of RAMECC reset, please refer to [Built-in SRAM (SRAM)]

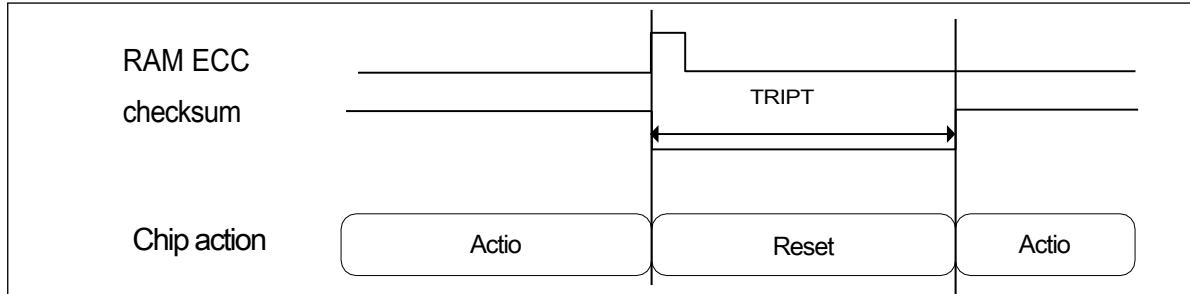


Figure 3-11 RAMECC Reset

3.3.11 Clock frequency abnormal reset

The chip's built-in FCM module will generate a clock frequency abnormal reset if it is set to reset when an abnormal clock frequency is detected, as shown in Figure 3-12. The clock frequency abnormal reset will set RMU_RSTF0. After the internal reset time T_{RIPT} , the chip is released from reset.

For the setting of clock frequency abnormal reset, refer to [4.10 Clock Frequency Measurement].

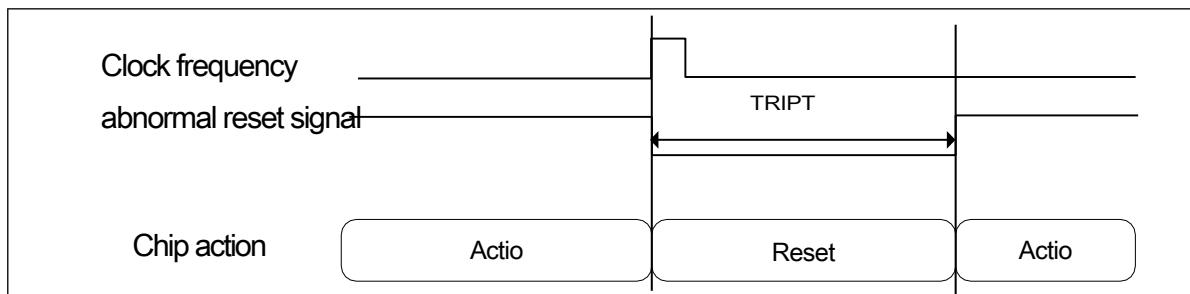


Figure 3-12 Clock Frequency Abnormal Reset

3.3.12 External high-speed oscillator abnormal stop reset

When the chip's oscillation stop detection module is active and reset is enabled, if an abnormal external high-speed oscillator stop occurs, an abnormal external high-speed oscillator stop reset is generated and RMU_RSTF0.XTALERF is set. After the internal reset time T_{RIPT} , the chip is released from reset.

For the setting of the external high-speed oscillator abnormal stop reset, refer to [4.5.2 External high-speed oscillator fault detection]

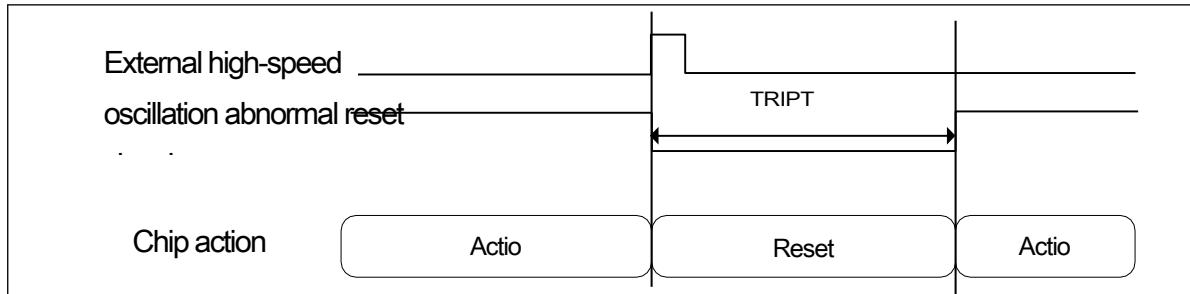


Figure 3-13 External high-speed oscillation abnormal reset

3.3.13 Determinatio

n of reset

mode

The reset mode can be determined by the reset flag of RMU_RSTF0. When two or more resets are generated at the same time, multiple reset flags may be generated. a MULTIRF bit of 1 in RMU_RSTF0 indicates that multiple resets have occurred. After reading RMU_RSTF0, all reset flags can be cleared to 0 by setting the CLRF bit. after setting RMU_RSTF0 to zero, wait at least 6 CPU clock cycles before reading the RMU_RSTF0 register again.

3.3.14 Reset conditions for each module

Mod ules	Regist er	Reset Sourc e
Debugging Controller (DBG)	MCUSTPCTL MCUTRACECTL MCUDBGSTAT	1. Power-on reset 2. Power-down wake-up reset
Real Time Clock (RTC)	RTC internal registers	Module software reset control bit: RTC_CR0.RESET
Power Control (PWC) Clock Control (CMU)	pwc_pwrc0 pwc_pwrc1 pwc_pdwke0 pwc_pdwke1 pwc_pdwke2 pwc_pdwkes CMU_XTALCFG	All reset sources other than down-marked reset: power- down mode 1 wake-up reset Power down mode 2 Wake-up reset Power down mode 4 Wake-up reset
	pwc_pdwkf0 pwc_pdwkf1	All reset sources other than power-down wake-up reset
	PWC_PVDLCSR PWC_PVDCR1 PWC_PVDFCR PWC_PVDCR0 PWC_PWRC2 PWC_PWCMR	1. Power-on reset 2. Pin Reset 3. Undervoltage reset 4. Watchdog Reset 5. Dedicated watchdog reset 6. Power down mode 3 Wake-up reset
	pwc_pvdcir[0] pwc_pvddsr.pvd1detflg pwc_pvdcir[4] pwc_pvddsr.pvd2detflg	1. Power-on reset 2. Pin Reset 3. Undervoltage reset 4. Watchdog Reset 5. Dedicated watchdog reset 6. Power-down wake-up reset
	Other than the above note	All reset sources
	Registers other than the above modules	All reset sources

All reset sources in the table refer to the 14 reset sources described in the introduction of this chapter.

3.4 Register Description

The register list is shown in

Table 3-3. BASE ADDR:

0x400540C0

Table 3-3 List of RMU registers

Register Name	Symbols	Offset Address	Bit width	Reset value
Reset Status Register	RMU_RSTF0	0x00	16	Different reset values according to different reset methods

3.4.1 Reset flag register 0 (RMU_RSTF0)

Reset value: 0xFFFF (depending on the reset mode, the reset value is different)

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CLRF	MULTIRF	XTALERF	CKFERF	RAECRF	RAPERF	MPUERF	SWRF	PDRF	SWDRF	WDRF	PVD2RF	PVD1RF	BORF	PINRF	PORF

position	Marker	Place Name	Function	Reading and writing
b15	CLRF	Clear the reset flag	Software set to 1 to clear the reset flag bit. 0 on read. the reset action must be performed after reading RMU_RSTF0. 0: No operation 1: Zero reset flag	R/W
b14	MULTIRF	More than 2 reset occurrence flag bits	Set by hardware when two or more resets occur. Cleared by setting CLRF to zero. 0: Two and more resets have not occurred 1: When two or more resets occur	R/W
b13	XTALERF	External high-speed oscillator abnormal stop reset flag	Set by hardware when an abnormal external high-speed oscillator deactivation reset occurs. Clear by setting CLRF to zero. 0: No abnormal external high-speed oscillator stop reset occurred 1: The occurrence of external high-speed oscillator abnormal stop reset	R/W
b12	CKFERF	Clock frequency exception reset flag	Set by hardware when an abnormal clock frequency reset occurs. Clear by setting CLRF to zero. 0: Clock frequency abnormal reset did not occur 1: Clock frequency abnormal reset occurs	R/W
b11	RAECRF	RAMECC Reset Logo	Set by hardware when a RAMECC reset occurs. Cleared by setting CLRF to zero. 0: RAMECC reset did not occur 1: RAMECC reset occurs	R/W
b10	RAPERF	RAM parity error reset flag	Set by hardware when a RAM parity error reset occurs. Cleared by setting CLRF to zero. 0: No RAM parity error reset occurred 1: RAM parity error reset occurred	R/W

b9	MPUERF	MPU error reset flag	Set by hardware when an MPU error reset occurs. Cleared by setting CLRF to zero. 0: MPU error reset did not occur 1: MPU error reset occurred	R/W
----	--------	----------------------	---	-----

b8	SWRF	Software reset flag	Set by hardware when a software reset occurs. Cleared by setting CLRF to zero. 0: Software reset did not occur 1: Software reset occurs	R/W
b7	PDRF	Power down mode reset	Set by hardware when a power-down mode reset occurs. Cleared by writing to CLRF. 0: No power-down mode reset has occurred 1: Power-down mode reset occurs	R/W
b6	SWDRF	Dedicated watchdog reset flag	Set by hardware when a dedicated watchdog reset occurs. Cleared by writing to CLRF. 0: No dedicated watchdog reset has occurred 1: A dedicated watchdog reset occurs	R/W
b5	WDRF	Watchdog reset flag	Set by hardware when a watchdog reset occurs. Cleared by writing to CLRF. 0: Watchdog reset did not occur 1: Watchdog reset occurs	R/W
b4	PVD2RF	Programmable voltage detection 2 reset flag	Set by hardware when a programmable voltage detect 2 reset occurs. Clear by writing to CLRF. 0: No programmable voltage detect 2 reset occurred 1: Programmable voltage detection 2 reset occurs	R/W
b3	PVD1RF	Programmable voltage detection 1 reset flag	Set by hardware when a programmable voltage detect 1 reset occurs. Clear by writing to CLRF. 0: No programmable voltage detect 1 reset occurred 1: Programmable voltage detection 1 reset occurs	R/W
b2	BORF	Undervoltage Reset flag	Set by hardware when an undervoltage reset occurs. Cleared by writing to CLRF. 0: No undervoltage reset has occurred 1: Under-voltage reset occurs	R/W
b1	PINRF	NRST pin reset flag	Set by hardware when a pin reset occurs. Cleared by writing to CLRF. 0: NRST reset did not occur 1: NRST reset occurs	R/W
b0	PORF	Power-on reset flag	Set by hardware when a power-on reset occurs. Cleared by writing to CLRF. 0: No power-on reset occurred 1: Power-on reset occurs	R/W

4 Clock Controller (CMU)

4.1 Introduction

The clock control unit provides clock functions for a range of frequencies, including: an external high-speed oscillator, an external low-speed oscillator, two PLL clocks, an internal high-speed oscillator, an internal medium-speed oscillator, an internal low-speed oscillator, a SWDT dedicated internal low-speed oscillator, clock prescaler, clock multiplexing, and clock gating circuitry.

The Clock Control Unit also provides a clock frequency measurement function. The clock frequency measurement circuit (FCM) uses the measurement reference clock to monitor and measure the measurement object clock. An interrupt or reset occurs when the set range is exceeded.

The AHB, APB and Cortex-M4 clocks are all derived from the system clock, which can be sourced from a choice of six clock sources:

- 1) External high-speed oscillator (XTAL)
- 2) External low-speed oscillator (XTAL32)
- 3) MPLL Clock (MPLL)
- 4) Internal high-speed oscillator (HRC)
- 5) Internal medium speed oscillator (MRC)
- 6) Internal low speed oscillator (LRC)

The system clock can run at a maximum clock frequency of 200MHz. The SWDT has a separate clock source: the SWDT dedicated internal low-speed oscillator (SWDTLRC). The Real Time Clock (RTC) uses either an external low-speed oscillator or an internal low-speed oscillator as the clock source. 48MHz clock of USB-FS can choose System Clock, MPLL, UPLL as the clock source.

For each clock source, it can be turned on and off individually when not in use to reduce power consumption.

4.2 System Block Diagram

4.2.1 System Block Diagram

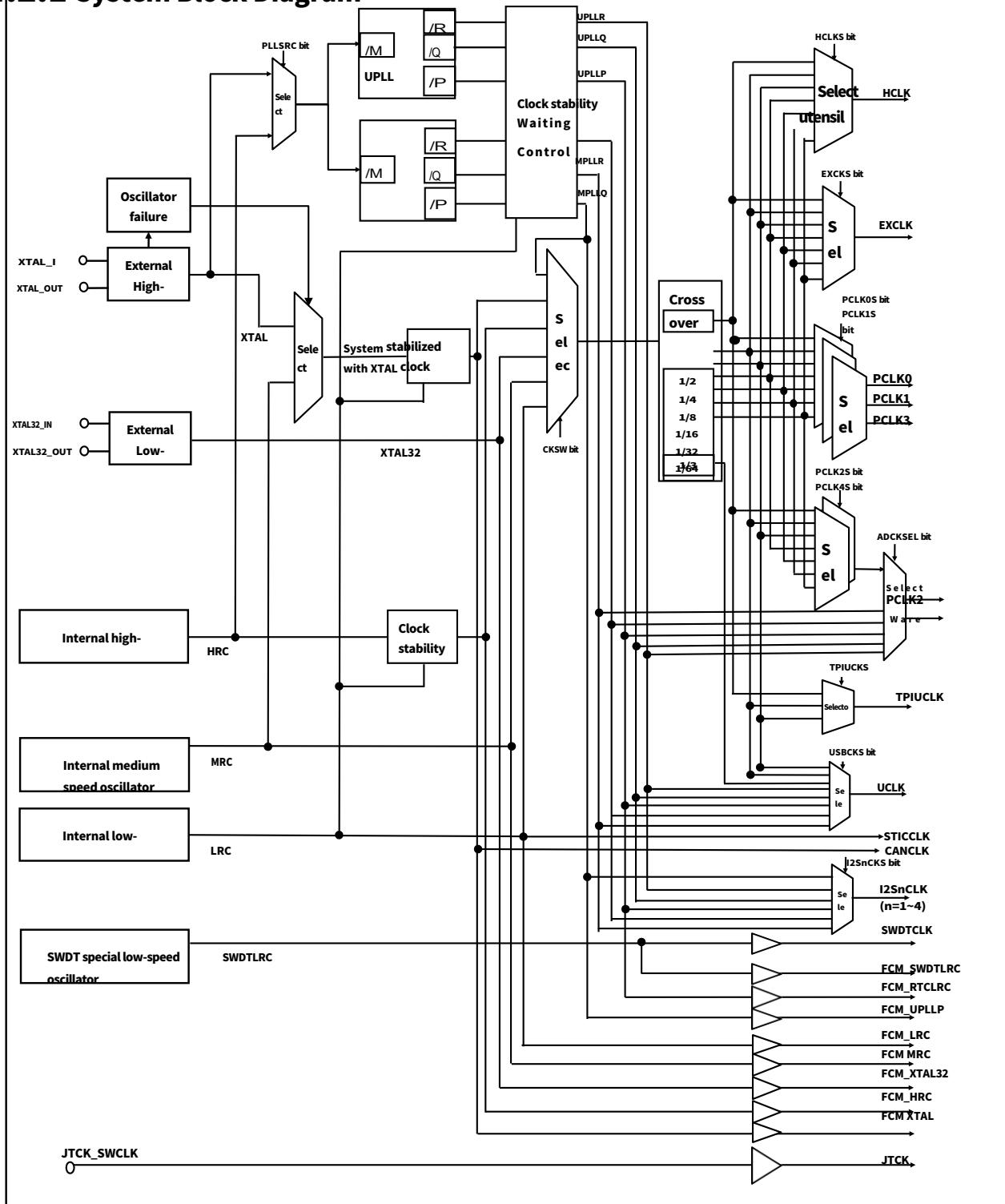


Figure 4-1 Block diagram of clock system

4.2.2 Clock frequency measurement block diagram

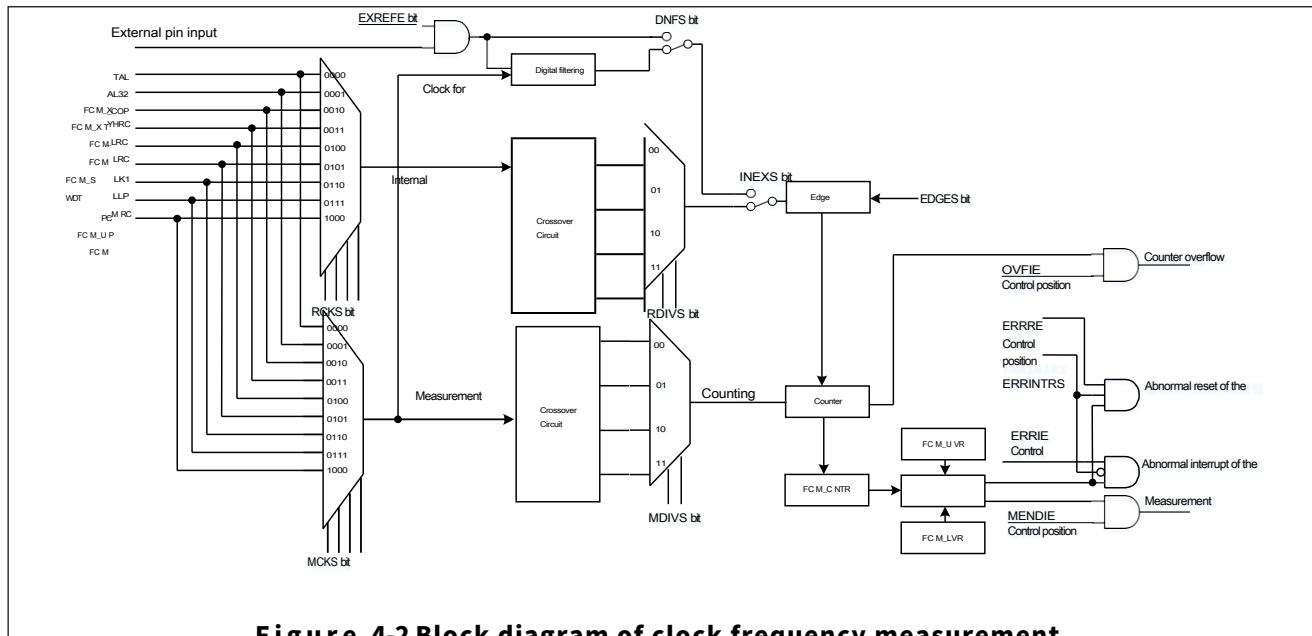


Figure 4-2 Block diagram of clock frequency measurement

4.3 Clock Source Specifications

The main characteristics of each clock source are shown in the following table.

Clock source	Specification
External high-speed oscillator (XTAL)	Frequency range of crystal oscillator: 4~25MHz External clock input: up to 25MHz oscillator fault detection function
External low-speed oscillator (XTAL32)	Frequency of crystal: 32.768KHz
MPLL Clock (MPLL)	Input clock: External high-speed oscillator or internal high-speed oscillator MPLL Input clock division: 1~24 arbitrary division selectable PFD input frequency = input clock/MPLL input clock division, frequency range 1MHz~25MHz MPLL multiplication factor: 20~480 times VCO oscillation frequency: 240MHz~480MHz MPLLQ output crossover ratio: 2~16 arbitrary crossover MPLLQ output crossover ratio: 2~16 arbitrary crossover MPLLQ output crossover ratio: 2~16 arbitrary crossover MPLLQ output frequency = (input clock/MPLL input clock division) * MPLL multiplication factor /MPLLQ Output Frequency Division Ratio MPLLQ output frequency = (input clock/MPLL input clock division) * MPLL multiplication factor /MPLLQ Output Frequency Division Ratio MPLLQ output frequency = (input clock/MPLL input clock division) * MPLL multiplication factor /MPLLQ Output Frequency Division Ratio

UPLL Clock (UPLL)	<p>Input clock: External high-speed oscillator or internal high-speed oscillator UPLL Input clock</p> <p>division: 1~24 arbitrary division selectable</p> <p>PFD input frequency = input clock/UPLL input clock division, frequency range 1MHz~25MHz UPLL multiplication factor: 20~480 times</p> <p>VCO oscillation frequency:</p> <p>240MHz~480MHz UPLLP output</p> <p>crossover ratio: 2~16 arbitrary</p> <p>crossover UPLLQ output crossover ratio: 2~16 arbitrary crossover</p> <p>UPLL output crossover ratio: 2~16 arbitrary crossover</p> <p>UPLLP output frequency = (input clock/UPLL input clock division) * UPLL multiplier /UPLLP Output Frequency Division Ratio</p> <p>UPLLQ output frequency = (input clock/UPLL input clock division) * UPLL multiplier /UPLLQ Output Frequency Division Ratio</p> <p>UPLL output frequency = (input clock/UPLL input clock division) * UPLL multiplier /UPLL Output Frequency Division Ratio</p>
Internal high-speed oscillator (HRC)	<p>Frequency: 16MHz or 20MHz user writable</p> <p>registers for fine-tuning the frequency</p>
Internal medium speed oscillator (MRC)	<p>Frequency: 8MHz</p> <p>User writable registers for fine-tuning frequency</p>

Clock source	Specification
Internal low speed oscillator (LRC)	Frequency: 32.768KHz User writable registers for fine-tuning frequency Can be used as count clock for RTC, count clock for wake-up timer WKTM, backup clock for XTAL32
SWDT dedicated internal low-speed oscillator (SWDTRC)	Frequency: 10KHz

4.4 Working

Clock

Table 4-1 Specifications of each internal clock

Specification

ns

Clock	Scope of action	Specification
HCLK	CPU, DMA(n=1, 2), EFM (main flash), SRAM1, SRAM2, SRAM3, SRAMH, Ret-SRAM, MPU, GPIO, DCU, INTC, QSPI	Maximum frequency 200MHz Dividing frequency of optional clock source: 1, 2, 4, 8, 16, 32, 64
PCLK0	Timer6 Clock for Counters	Maximum frequency 200MHz Dividing frequency of optional clock source: 1, 2, 4, 8, 16, 32, 64
PCLK1	USARTn (n=1~4), SPI(n=1~4), USBFS (control logic) TimerOn (n=1, 2), TimerAn (n=1~6), Timer4n (n=1~3), Timer6 (control logic) EMB, CRC, HASH, AES, I2Sn (n=1~4) control System logic	Maximum frequency 100MHz Dividing frequency of optional clock source: 1, 2, 4, 8, 16, 32, 64
PCLK2	ADC Transformation Clock	Maximum frequency 60MHz Selectable clock source divisions: 1, 2, 4, 8, 16, 32, 64 Independently selectable clock sources: UPLL, UPLLQ, UPLL, MPLL, MPLLQ, MPLLR
PCLK3	RTC (control logic) I2Cn (n=1, 2, 3), CMP, WDT, SWDT (control logic)	Maximum frequency 50MHz Dividing frequency of optional clock source: 1, 2, 4, 8, 16, 32, 64
PCLK4	ADC (control logic) TRNG	Maximum frequency 100MHz Selectable clock source divisions: 1, 2, 4, 8, 16, 32, 64 Independently selectable clock sources: UPLL, UPLLQ, UPLL, MPLL, MPLLQ, MPLLR
EXCLK	SDIO(n=1, 2), CAN	Maximum frequency 100MHz Dividing frequency of optional clock source: 1, 2, 4, 8, 16, 32, 64

UCLK	xUSBFS Pass Through Clock	Frequency 48MHz www.xhsc.com.c
		Clock source selectable system clock division 2, 3, 4. Clock source can be selected independently. UPLL_P, UPLL_Q, UPLL_R, MPLL_P, MPLL_Q. MPLL_R
CANCLK	CAN communication clock	Frequency range 4~25MHz
STICCLK	Clock for CPU's SysTickTimer counter, clock source is LRC	Configurable as clock source LRC or system clock
SWDTCLK	SWDT Clock for Counters	Frequency 10KHz
JTCK	JTAG with clock	Maximum frequency 25MHz
TPIUCLK	Cortex-M4 Debug Clock for Tracker	Maximum frequency 50MHz

Clock	Scope of action	Specification
		Dividing frequency of optional clock source: 1, 2, 4
I2SnCLK (n=1~4)	I2Sn(n=1~4)	Maximum frequency 200MHz with independently selectable clock source. UPLL, UPLLQ, UPLL, MPLLP, MPLLQ.

Caution:

The following rules are to be observed between the clocks:

- HCLK frequency \geq PCLK1 frequency, HCLK frequency \geq PCLK3 frequency, HCLK frequency \geq PCLK4 frequency
- **HCLK** HCLK frequency: EXCLK frequency = 2:1, 4:1, 8:1, 16:1, 32:1
- PCLK0 frequency \geq PCLK1 frequency, PCLK0 frequency \geq PCLK3 frequency
- HCLK Frequency : PCLK0 Frequency = N:1, 1:N
- **PCLK2** PCLK2 Frequency: PCK4 Frequency = 1:4, 1:2, 1:1, 2:1, 4:1, 8:1

4.5 Crystal Circuit

4.5.1 External high-speed oscillator

4.5.1.1 Oscillator mode

An external high-speed oscillator provides a more accurate clock source for the system clock. The frequency range is 4 to 25 MHz. XTAL is turned on and off by the XTALSTP bit of CMU_XTALCR.

The XTALSTBF flag bit of CMU_OSCSTBSR indicates whether the external high-speed oscillator is stable or not, and the stability time is configured through the register CMU_XTALSTBCR. The stability time set by CMU_XTALSTBCR must be greater than or equal to the stability time required by the crystal manufacturer.

The circuit constants of the crystal oscillator vary depending on the crystal and the parasitic capacitance of the mounting circuit, so it must be decided after careful discussion with the crystal manufacturer. The various characteristics of the oscillator are closely related to the user's board design. The crystal and load capacitance must be as close as possible to the oscillator pins to minimize output distortion and start-up stability time. The load capacitance value must be adjusted appropriately according to the selected oscillator. In the vicinity of the oscillation circuit can not pass the signal line, otherwise it may not oscillate properly due to inductance.

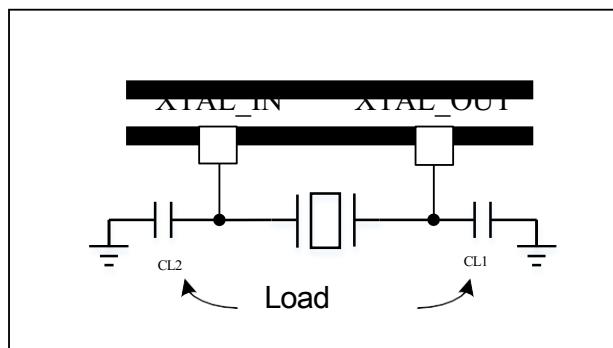


Figure 4-3 Example of external high-speed oscillator connection

4.5.1.2 Clock input mode

In clock input mode, an external clock source must be provided. This mode is selected by the XTALMS position "1" of CMU_XTALCFGR and the XTALSTP position "0" of CMU_XTALCR. An external clock signal with a duty cycle of approximately 50% must be used to drive the XTAL_IN pin. In this case, the XTAL_OUT pin can be configured as GPIO according to the register setting.

An example of the connection of the external clock input is shown in the following figure.

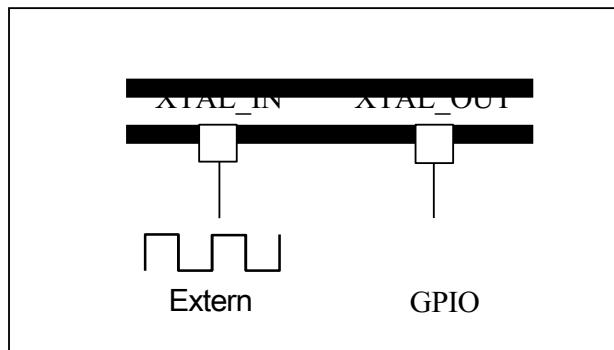


Figure 4-4 Connection example diagram of external clock input

4.5.2 External high-speed oscillator fault detection

Oscillator fault detection is to detect if the external high-speed oscillator (XTAL) is oscillating properly. It is turned on or off by the XTALSTDE bit of the CMU_XTALSTDCCR register.

When the reset is released, the external high-speed oscillator stops oscillating and the external high-speed oscillator fault detection function is disabled. To set the external high-speed oscillator fault detection function to be active, the external high-speed oscillator must be oscillated and turned on by the XTALSTDE bit of the CMU_XTALSTDCCR register until the external high-speed oscillator is stable, i.e., CMU_OSCSTBSR_XTALSTBF is 1.

When MPLL, UPLL selects XTAL clock as input source, only XTAL oscillation fault generation reset function can be selected.

Because oscillator fault detection is to detect abnormal oscillation of the oscillator caused by external factors, the oscillator oscillation fault detection function is to be disabled when the external high-speed oscillator stops oscillating or is transferred to stop mode and power-down mode by software.

If the external high-speed oscillator fails, the action waveform is shown in the figure below. Refer to [XTAL Fault Detection Action Detected] for the operation procedure.

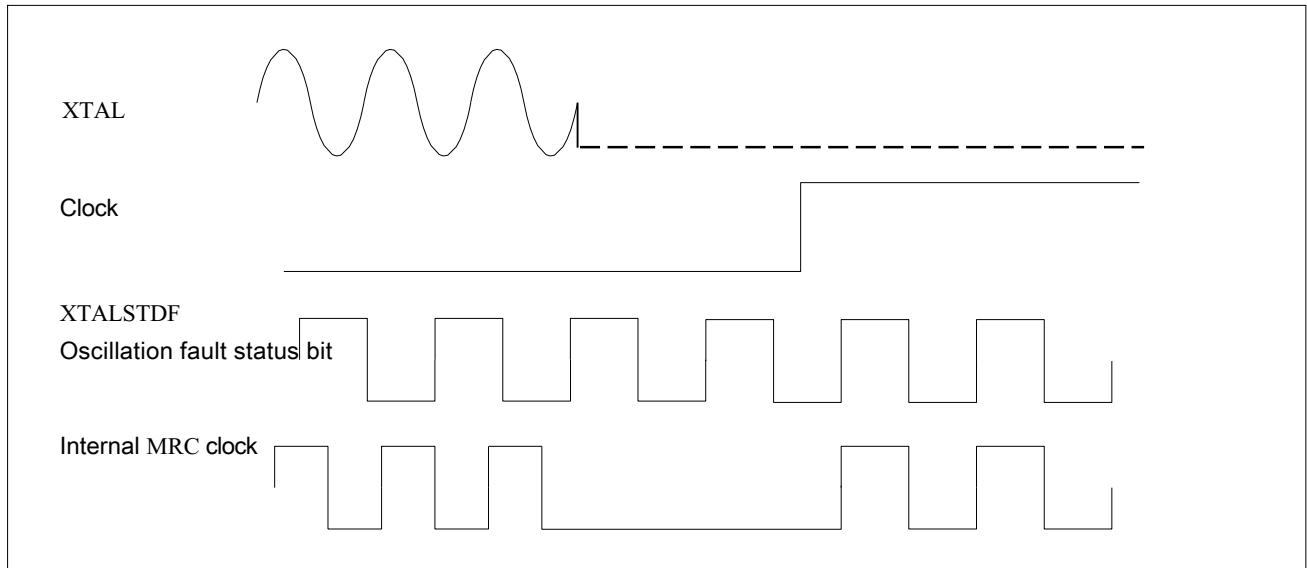


Figure 4-5 Example of external high-speed oscillator fault detection

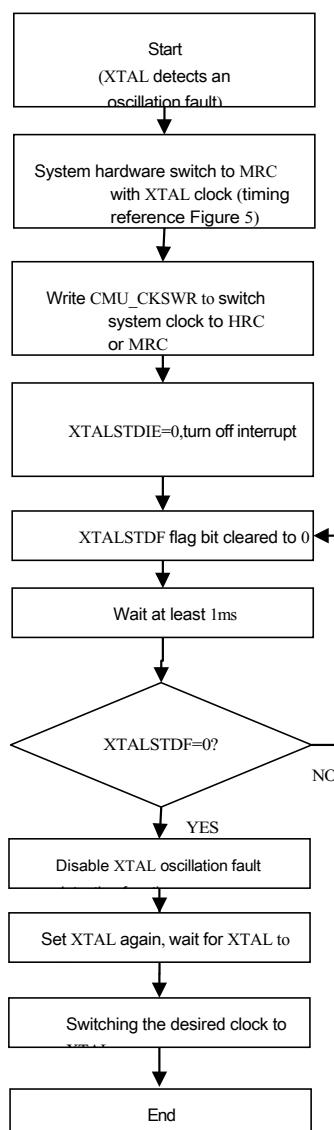


4.5.2.1 XTAL fault detection action detected

When an external high-speed oscillator oscillation fault is detected, the system clock automatically switches to MRC if the system clock selects the external high-speed oscillator as the system clock.

When an external high-speed oscillator oscillation fault is detected, the EMB can be triggered to set the PWM output of Timer6/Timer4 to Hiz output. Refer to [Emergency Brake Module (EMB)] chapter.

The system clock is selected as XTAL, and when an XTAL fault is detected, the action example is shown in the following figure.



**Figure 4-6 System clock selection XTAL, XTAL
oscillation fault detected Example**

4.5.2.2 XTAL oscillation fault detected to generate interrupt reset

XTAL oscillation fault interrupts can be configured as maskable or non-maskable interrupts, refer to the [Interrupt Controller (INTC) chapter.

When XTAL oscillation fault is configured as reset, XTAL oscillation fault is detected and the chip generates a reset, refer to [Reset Control (RMU) chapter for reset action.

4.5.3 External low-speed oscillator

The 32.768KHz external low-speed oscillator provides a more accurate clock source for system clocks and real-time clock circuits (RTCs). It has the advantages of low power consumption and high accuracy.

XTAL32 is turned on and off by the XTAL32STP bit of CMU_XTAL32CR.

The circuit constants of the crystal oscillator vary depending on the crystal and the parasitic capacitance of the mounting circuit, so it must be decided after careful discussion with the crystal manufacturer. The various characteristics of the oscillator are closely related to the user's board design. The crystal and load capacitor must be as close as possible to the oscillator pins to minimize output distortion and start-up stability time. The load capacitance value must be adjusted appropriately according to the selected drive capability. In the vicinity of the oscillation circuit can not pass the signal line, otherwise it may not oscillate properly due to inductance.

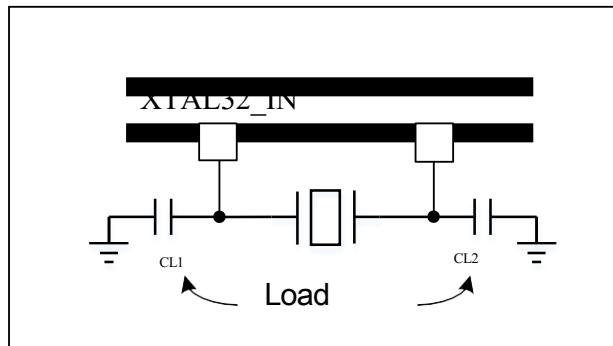


Figure 4-7 Example of external low-speed oscillator connection

The initialization flow for the initial power-up of XTAL32 is shown below:

1. CMU_XTAL32CR.XTAL32STP bit write 1, stop XTAL32
2. Set the matching XTAL32 drive capability via CMU_XTAL32FGR
3. Set the filter function via CMU_XTAL32FGR
4. CMU_XTAL32CR.XTAL32STP bit write 0, XTAL32 oscillation
5. The software waits for XTAL32 to stabilize, refer to the Electrical Characteristics chapter for

stabilization times.

If the external low-speed oscillator is not used, set the XTAL32STP bit of CMU_XTAL32CR to 1 to turn off the external low-speed oscillator.

4.6 Internal RC clock

4.6.1 HRC Clock

The HRC clock signal is generated by the internal high-speed oscillator and can be used directly as a system clock or as an MPLL/UPLL input. The frequency of the HRC can be configured to 16MHz or 20MHz by ICG1.HRCFREQSEL.

The advantage of the HRC oscillator is its lower cost (no external components are required) It also starts up faster than the XTAL crystal block, but even after calibration, it is not as accurate as an external crystal.

Frequency calibration

Because the RC oscillator frequency varies from chip to chip due to different production processes, each device will be factory calibrated to ensure accuracy refer to the **internal high speed (HRC) oscillator** chapter in the **electrical characteristics of the datasheet**.

This may also affect the speed of the RC oscillator if the application is affected by temperature variations. The HRC frequency can be fine-tuned by the user via the CMU_HRCTRM register.

The HRCSTBF flag in CMU_OSCSTBSR indicates whether the HRC is stable or not. The HRC is available only after the hardware sets this position 1 at boot time.

HRC can be turned on or off by the HRCSTP bit in the CMU_HRCCR control register.

4.6.2 MRC Clock

The MRC clock signal is generated by the internal 8MHz medium speed oscillator and can be used directly as the system clock. The advantage of the MRC oscillator is that it is fast to start up and can be used without waiting for stability after startup. **Frequency Calibration**

Because the RC oscillator frequency varies from chip to chip due to different production processes, each device will be factory calibrated to ensure accuracy refer to the **internal medium speed (MRC) oscillator** chapter in the **electrical characteristics of the datasheet**.

If the application is affected by temperature variations, this may also affect the speed of the RC oscillator. The MRC frequency can be fine-tuned by the user via the CMU_MRCTR register.

MRC can be turned on or off by the MRCSTP bit in the CMU_MRCCR control register.

The MRC clock can also be used as a backup clock source in case the XTAL crystal fails. See Detect XTAL Failure Detection Action.

4.6.3 LRC Clock

The LRC clock signal is generated by the internal 32.768KHz low-speed oscillator and can be used directly as a system clock. IRC can be used as a low-power clock source to keep running in power-down mode and stop mode for RTC/Timer0/KEYSCAN/WKTM.

The LRC oscillator has a fast start-up speed and can be used without waiting for stability after start-up.

Frequency calibration

Because the RC oscillator frequency varies from chip to chip due to different production processes, each device will be factory calibrated to ensure accuracy refer to the **internal low speed (LRC) oscillator** chapter in the **electrical characteristics of the datasheet**.

If the application is affected by voltage or temperature variations, this may also affect the speed of the RC oscillator. The LRC frequency can be fine-tuned by the user via the CMU_LRCTR register.

LRC can be turned on or off by the LRCSTP bit in the CMU_LRCCR control register.

4.6.4 SWDTRC Clock

The SWDTRC clock signal is generated by the internal 10KHz low-speed oscillator, which is dedicated to the SWDT. if the SWDT has been started by the ICG setting, the SWDT dedicated internal low-speed oscillator will be forced on and cannot be disabled.

Because the RC oscillator frequency varies from chip to chip due to different manufacturing processes, each device is factory calibrated to ensure accuracy refer to the **SWDT dedicated internal low speed (SWDTLRC) oscillator** section in the **Electrical Characteristics of the datasheet**.

4.7 PLL Clock

The HC32F46xx devices have two PLLs:

- The MPLL is clocked by an XTAL or HRC oscillator and has three different output clocks:
 - P divider output for system clock generation (up to 200 MHz)
 - All three outputs can be used to generate USBFS, TRNG, ADC, and I2S clocks.
- The three UPLL outputs can also be used to generate USBFS, TRNG, ADC and I2S clocks.

The UPLL uses the same input clock source as the MPLL, either XTAL oscillator as the clock source, as

configured by the CMU_PLLCFGR.PLLSRC bit. The PLL is configured after the HRC or XTAL oscillator has stabilized.

The MPLL/UPLL crossover coefficients M, N, P, Q, and R can be configured independently. Since the PLL configuration parameters cannot be changed after the PLL is enabled, it is recommended to configure the PLL first and then enable it.

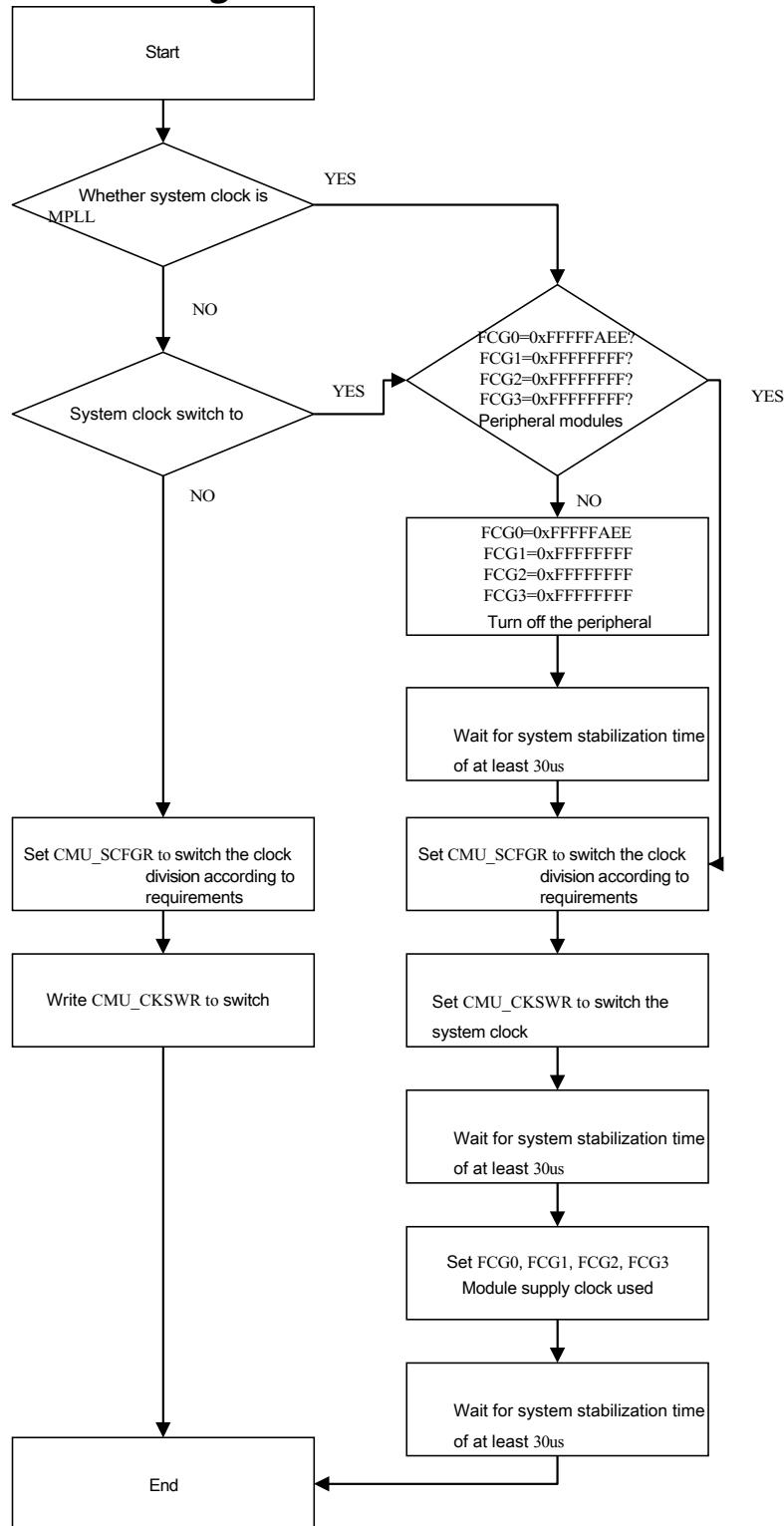
Both PLLs will be disabled by hardware when power-down and stop modes are entered.

4.8 Clock Switching Procedure

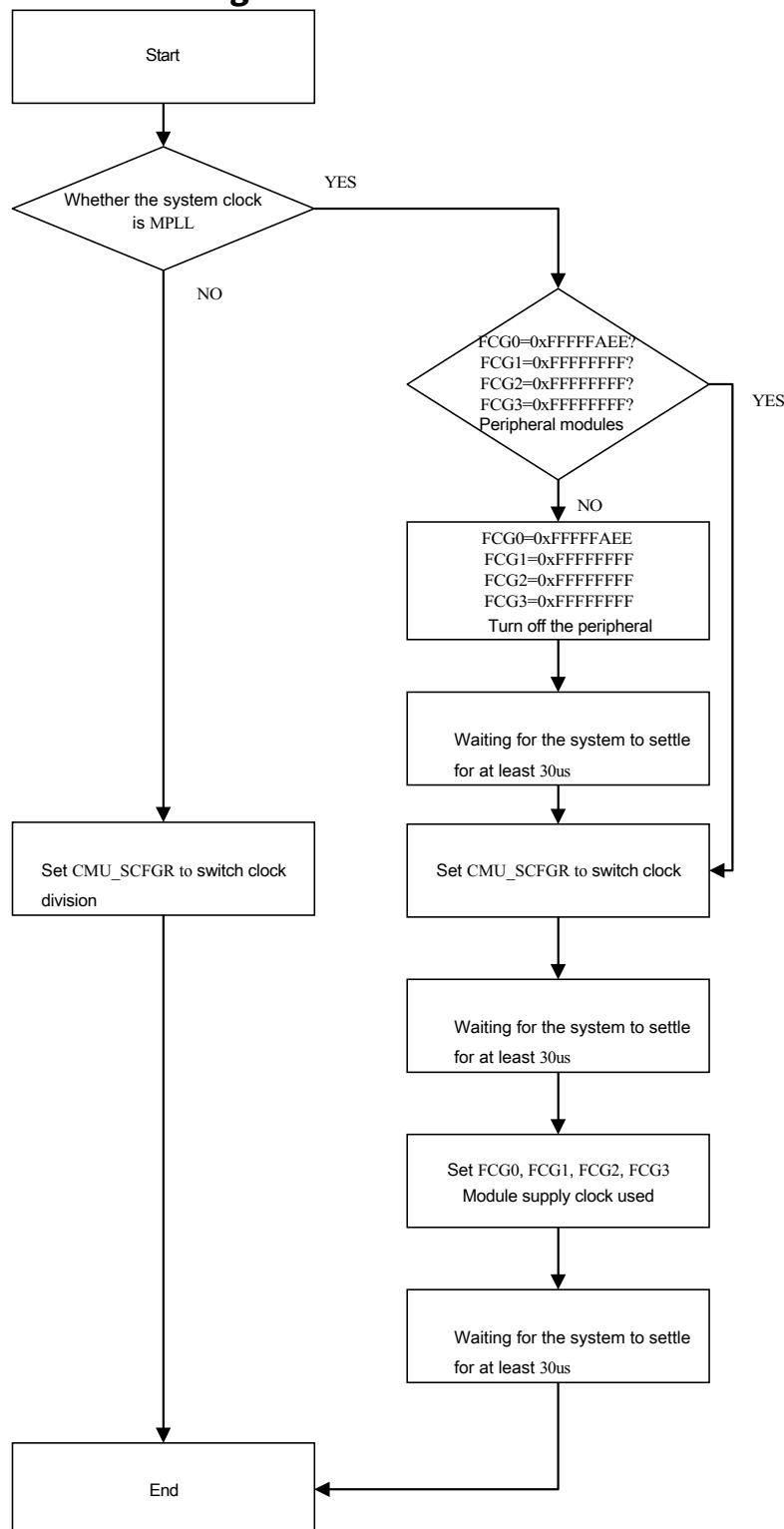
After system reset, the default system clock is MRC. switch the clock source by setting register CMU_CKSW, and the switching procedure is referred to as clock source switching. You can switch from one clock source to another only when the target clock source has been stabilized.

The wait period of FLASH/SRAM needs to be configured correctly during clock switching to prevent the system clock frequency from being greater than the maximum action frequency of FLASH/SRAM. Refer to [Relationship between CPU clock and FLASH read time] and [Built-in SRAM (SRAM) configuration].

4.8.1 Clock source switching



4.8.2 Clock division switching



4.9 Clock output function

There are two clock outputs:

- MCO_1

The user can output different clock sources to the MCO_1 pin via configurable pre-distributors (from 1 to 128):

- HRC Clock
- MRC Clock
- LRC Clock
- XTAL Clock
- XTAL32 Clock
- MPLLP/MPLLQ Clock
- UPLLQ/UPLLQ Clock
- System Clock

The desired clock source is selected via the CMU_MCO1CFGR.MCO1SEL bit.

- MCO_2

The user can output different clock sources to the MCO_2 pin via configurable pre-distributors (from 1 to 128):

- HRC Clock
- MRC Clock
- LRC Clock
- XTAL Clock
- XTAL32 Clock
- MPLLP/MPLLQ Clock
- UPLLQ/UPLLQ Clock
- System Clock

The desired clock source is selected via the CMU_MCO2CFGR.MCO2SEL bit. The MCO_1/MCO_2 output clock must not exceed 100 MHz (maximum I/O speed)

4.10 Clock frequency measurement

4.10.1 Clock frequency measurement

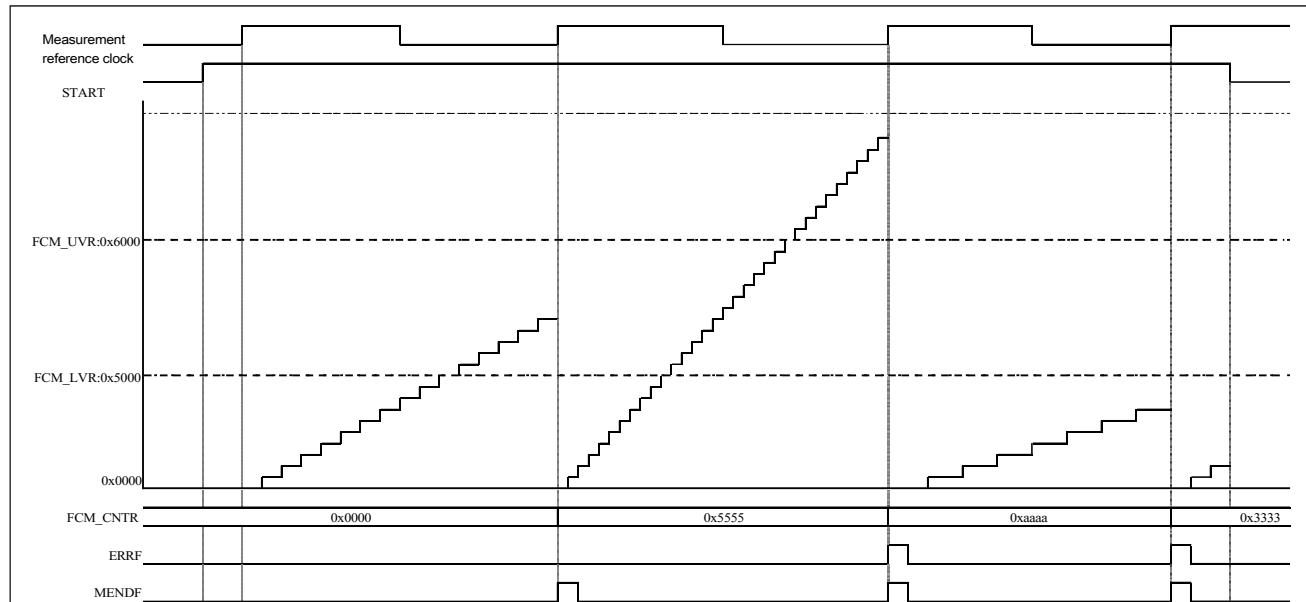


Figure 4-8 Clock Frequency Measurement

Timing Diagram

1. Use FCM_MCCR/FCM_RCCR to select the reference clock to be measured, the dividing frequency of the clock and the effective edge of the selected reference clock.
2. After the START bit of FCM_STR is written to 1, the counter starts counting incrementally when a valid edge of the EDGES bit selection is detected.
3. When a valid edge of the next EDGES bit selection of the reference clock is detected, the value of the counter is saved to the FCM_CNTR register and compared to the FCM_LVR/FCM_UVR setting. When $FCM_LVR \leq FCM_CNTR \leq FCM_UVR$, the clock frequency under test is measured normally. When $FCM_LVR > FCM_CNTR$ or $FCM_CNTR > FCM_UVR$, the measured clock frequency is abnormal and an interrupt or reset can occur according to the ERRINTRS/ERRRE/ERRIE setting.
4. After writing 0 to the START bit of FCM_STR, the counter count stops and clears.

4.10.2 Digital filtering function

The external pin input reference clock FCMREF has a digital filter function. The digital filter function performs 3 samples based on the sample clock selected by the DNFS bit, and sends this level internally when the levels of the 3 samples are the same.

The digital filtering function allows you to set the digital filtering function to be active or inactive and the sampling clock.

4.10.3 Interrupt/reset function

The clock frequency measurement circuit has three kinds of interrupt requests. They are:

- 1) Abnormal frequency interruption
- 2) Frequency measurement end interrupt
- 3) Counter overflow interrupt

Clock frequency measurement

circuits have one type of reset

request: 1) Frequency

abnormal reset

4.11 Register Description

Base address 1: 0x4004_8400

Register Name	Sym bols	Offset Address	Bit width	Reset value
FCM lower limit comparison value register	FCM_LVR	0x00	32	0x0000_0000
FCM upper limit comparison value register	FCM_UVR	0x04	32	0x0000_0000
FCM counter value register	FCM_CNTR	0x08	32	0x0000_0000
FCM Start Stop Register	FCM_STR	0x0C	32	0x0000_0000
FCM measurement object control register	FCM_MCCR	0x10	32	0x0000_0000
FCM measurement reference control register	FCM_RCCR	0x14	32	0x0000_0000
FCM Interrupt Reset Control Register	FCM_RIER	0x18	32	0x0000_0000
FCM flag register	FCM_SR	0x1C	32	0x0000_0000
FCM flag bit clear register	FCM_CLR	0x20	32	0x0000_0000

Base address 2: 0x40054000

Register Name	Sym bols	Offset Address	Bit width	Reset value
CMU_XTAL Configuration Register	CMU_XTALCFG	0x410	8	0x80
CMU_XTAL Valium Configuration Register	CMU_XTALSTBCR	0x0A2	8	0x05
CMU_XTAL control register	CMU_XTALCR	0x032	8	0x01
CMU_XTAL Oscillation Fault Control Register	CMU_XTALSTDRCR	0x040	8	0x00
CMU_XTAL Oscillation Fault Status Register	CMU_XTALSTDSCR	0x041	8	0x00
CMU_HRC Calibration Register	CMU_HRCTRM	0x062	8	0x00
CMU_HRC Control Register	CMU_HRCCR	0x036	8	Determined by ICG1.HRCSTP value
CMU_MRC Calibration Register	CMU_MRCTRM	0x061	8	0x00
CMU_MRC control register	CMU_MRCCR	0x038	8	0x80
CMU_MPLL Configuration Register	CMU_PLLCFG	0x100	32	0x1110_1300
CMU_MPLL Control Register	CMU_PLLCR	0x02A	8	0x01
CMU_UPLL Configuration Register	CMU_UPLLCFG	0x104	32	0x1110_1300
CMU_UPLL control register	CMU_UPLLCR	0x02E	8	0x01
CMU_Clock Source Stability Status Register	CMU_OSCSTBSR	0x03C	8	0x00
CMU_system clock source switching register	CMU_CKSWR	0x026	8	0x01
CMU_Clock Divider Configuration Register	CMU_SCFGR	0x020	32	0x0000_0000
CMU_USBFS Clock Configuration Register	CMU_UFSCKCFG	0x024	8	0x40
CMU_AD/TRNG Clock Configuration Register	CMU_PERICKSEL	0x010	16	0x0000
CMU_I2S Clock Configuration Register	CMU_I2SCKSEL	0x012	16	0xB BBBB
CMU_Debug Clock Configuration Register	CMU_TPIUCKCFG	0x03F	8	0x00
CMU_MCO1 Clock Output Configuration Register	CMU_MCO1CFG	0x03D	8	0x00
CMU_MCO2 Clock Output Configuration Register	CMU_MCO2CFG	0x03E	8	0x00
CMU_XTAL32 control register	CMU_XTAL32CR	0x420	8	0x00
CMU_XTAL32 Configuration Register	CMU_XTAL32CFG	0x421	8	0x00
CMU_XTAL32 Filter Register	CMU_XTAL32NFR	0x425	8	0x00
CMU_LRC control register	CMU_LRCCR	0x427	8	0x00
CMU_LRC Calibration Register	CMU_LRCTRM	0x429	8	0x00

4.11.1 CMU XTAL Configuration Register (CMU_XTALCFG)R

Reset value: 0x80

b7	b6	b5	b4	b3	b2	b1	b0
SUPDRV	XTALMS	XTALDRV[1:0]	-	-	-	-	-
<hr/>							
position	Marker	Place Name	Function			Reading and writing	
b7	SUPDRV	XTAL ultra-high speed drive allows	0: Disable ultra-high speed drive 1: Allows ultra-high speed drives When super speed drive is allowed, XTAL is set to ignore this bit setting and disable super speed drive to reduce power consumption after stabilization.			R/W	
b6	XTALMS	XTAL mode selection bit	0: Oscillator mode 1: External clock input mode			R/W	
b5~b4	XTALDRV[1:0]	XTAL drive capability selection	00: High drive capability (recommended $20 \leq f_{XTAL_IN} \leq 25MHz$ crystal) 01: Medium drive capability (recommended $16 \leq f_{XTAL_IN} \leq 20MHz$ crystal) 10: Small drive capability (recommended $8 \leq f_{XTAL_IN} \leq 16MHz$ crystal) 11: Ultra-small drive capability (recommended $4 \leq f_{XTAL_IN} \leq 8MHz$ crystal)			R/W	
b3~b0	Reserved	-	Read "0" for read write "0" for write			R/W	

4.11.2 CMU XTAL Settling Configuration Register (CMU_XTALSTBCR)

Reset value: 0x05

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-				XTALSTB[3:0]

position	Marker	Place Name	Function	Reading and writing
b7~b4	Reserved	-	Read "0" for read write "1" for write	R/W
b3~b0	XTALSTB[3:0]	XTAL stabilization time selection	0001: Stabilization counter 35 cycles 0010: Stabilization counter 67 cycles 0011: Stabilization counter 131 cycles 0100: Stabilization counter 259 cycles 0101: Stabilization counter 547 cycles 0110: Stabilization counter 1059 cycles 0111: Stabilization counter 2147 cycles 1000: Stable counter 4291 cycles 1001: Stabilization counter 8163 cycles One count period of the stabilization counter = LRC period/8 CMU_XTALCR_XTALSTP bit 1 and CMU_OSCSTBSR.XTALSTBF bit is 0 to configure this register.	R/W

4.11.3 CMU XTAL Control Register (CMU_XTALCR)

Reset value: 0x01

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	XTALSTP

position	Marker	Place Name	Function	Reading and writing
b7~b1	Reserved	-	Read "0" for read write "1" for write	R/W
b0	XTALSTP	XTAL oscillator on stop bit	0: XTAL oscillator oscillation 1: XTAL oscillator stopped	R/W

Caution:

- When XTAL is selected as the system clock or MPLL/UPLL clock source, disable XTALSTP to write "1" to stop XTAL oscillator.
- The software sets the XTAL oscillator to oscillate, and the XTALSTBF bit confirms that the XTAL oscillator is stable before you can enter the stop mode, power-down mode, or software set the XTAL oscillator to stop.
- The software sets the XTAL oscillator to stop and the XTALSTBF bit confirms that the XTAL oscillator is stopped before the XTAL oscillator can be entered into stop mode, power down mode or started again.

4.11.4 CMU XTAL Oscillation Fault Control Register (CMU_XTALSTDCR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
XTALSTDE	-	-	-	-	XTALSTDRI	XTALSTDRE	XTALSTDIE

position	Marker	Place Name	Function	Reading and writing
b7	XTALSTDE	XTAL oscillation fault detection function Can allow	0: XTAL oscillation fault detection is disabled 1: Allow XTAL oscillation fault detection Notes: Oscillator fault detection is the detection of abnormal oscillation of the oscillator caused by external factors, after entering Before stop mode or power-down mode, please disable the oscillator oscillation fault detection function.	R/W
b6~b3	Reserved	-	Read "0" for read write "1" for write	R/W
b2	XTALSTDRI	XTAL oscillation fault reset in progress Select	0: XTAL oscillation fault generates interrupt 1: XTAL oscillation fault generates reset Notes: When MPLL and UPLL select XTAL clock as input source, only XTAL oscillation can be selected. Fault generation reset function.	R/W
b1	XTALSTDRE	XTAL oscillation fault reset permit	0: XTAL oscillation fault reset is disabled 1: Allow XTAL oscillation fault reset	R/W
b0	XTALSTDIE	XTAL oscillation fault interrupter permit	0: XTAL oscillation fault interrupt is disabled 1: Allow XTAL oscillation fault interruption The PWM output of Timer6/Timer4 is set to Hiz output via EMB. The XTALSTDIE bit needs to be set to 1.	R/W

Caution:

- **XTAL** When XTAL is selected as the system clock or MPLL/UPLL clock source, disable XTALSTP to write "1" to stop XTAL oscillator.

4.11.5 CMU XTAL Oscillation Fault Status Register (CMU_XTALSTDSR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	XTALSTDF

position	Marker	Place Name	Function	Reading and writing
b7~b1	Reserved	-	Read "0" for read write "1" for write	R/W
b0	XTALSTDF	XTAL oscillation fault status bit	0: XTAL oscillation fault not detected 1: XTAL oscillation fault setting condition is detected: XTAL oscillation fault under XTALSTDE=1 Clear condition: Read 1 Write 0 when system clock is selected other than XTAL clock.	R/W

4.11.6 CMU XTAL32 Configuration Register (CMU_XTAL32CFGR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	XTAL32DRV[2:0]

position	Marker	Place Name	Function	Reading and writing
b7~b3	Reserved	-	Read "0" for read write "1" for write	R/W
b2~b0	XTAL32DRV[2:0]	XTAL32 drive capability selection	000: Medium drive capability 001: Large drive capability Other: Prohibit setting Note: Refer to the Electrical Characteristics chapter [Crystal/Ceramic Resonator Generated Low Speed external clock]	R/W

4.11.7 CMU XTAL32 Filter Register (CMU_XTAL32NFR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	XTAL32NF[1:0]

position	Marker	Place Name	Function	Reading and writing
b7~b2	Reserved	-	Read "0" for read write "0" for write	R/W
b1~b0	XTAL32NF[1:0]	XTAL32 Oscillator Filter Selection	00: RUN mode/stop mode/power-down mode, XTAL32's 3us filtering is valid 01: RUN mode XTAL32's 3us filter is valid, stop mode or power-down mode XTAL32's 3us filter is invalid 10: Set the ban 11: RUN mode/stop mode/power-down mode, XTAL32's 3us filtering is invalid	R/W

4.11.8 CMU XTAL32 control register (CMU_XTAL32CR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	XTAL32STP

position	Marker	Place Name	Function	Reading and writing
b7~b1	Reserved	-	Read "0" for read write "0" for write	R/W
b0	XTAL32STP	XTAL32 oscillator on stop bit	0: XTAL32 oscillator oscillation 1: XTAL32 oscillator stopped	R/W

Caution:

- When XTAL32 is selected as the system clock source, disable XTAL32STP write "1" to stop XTAL32 oscillator.
- The software sets the XTAL32 action to start and waits for 5 XTAL32 cycles before stopping the XTAL32 again.
- The software sets XTAL32 to stop and waits for 5 XTAL32 cycles before starting XTAL32 again.

4.11.9 CMU HRC Calibration Register (CMU_HRCTRM)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
HRCTRIM[7:0]							

position	Marker	Place Name	Function	Reading and writing
b7~b0	HRCTRIM[7:0]	HRC frequency calibration bits	Frequency calibration needs to be within the HRC frequency guarantee. 10000000: -128 10000001: -127 11111111: -1 00000000: Center Code 00000001: +1 01111110: +126 01111111: +127	R/W

4.11.10 CMU HRC Control Register (CMU_HRCCR)

Reset value: determined by ICG1.HRCSTP value

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	HRCSTP

position	Marker	Place Name	Function	Reading and writing
b31~b1	Reserved	-	Read "0" for read write "0" for write	R/W
b0	HRCSTP	HRC oscillator on stop bit	0: HRC oscillator oscillation 1: HRC oscillator stops According to ICG1.HRCSTOP configuration, HRC starts to stop after reset.	R/W

Caution:

- When HRC is selected as the system clock source or MPLL/UPLL clock source, disable CMU_HRCCR.HRCSTP to write "1" to stop the HRC clock.
- The software sets the HRC oscillation and confirms that the HRC is stable by the HRCSTBF bit before entering the stop mode, power-down mode or stopping the HRC.
- The software sets the HRC to stop, and the HRCSTBF bit confirms that the MPLL is stopped before it can enter stop mode, power-down mode, or start the HRC again.

4.11.11 CMU MRC Calibration Register (CMU_MRCTRIM)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
MRCTRIM[7:0]							

position	Marker	Place Name	Function	Reading and writing
b7~b0	MRCTRIM[7:0]	MRC frequency calibration bits	10000000: -128 10000001: -127 11111111: -1 00000000: Center Code 00000001: +1 01111110: +126 01111111: +127	R/W

Caution:

-Frequency calibration The frequency calibration needs to be within the MRC frequency guarantee.

4.11.12 CMU MRC Control Register (CMU_MRCCR)

Reset value: 0x80

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	MRCSTP

position	Marker	Place Name	Function	Reading and writing
b7	-	-	Read "1" for read, write "1" for write	R/W
b6~b1	Reserved	-	Read "0" for read, write "0" for write	R/W
b0	MRCSTP	MRC oscillator on stop bit	0: MRC oscillator oscillation 1: MRC oscillator stop Note: 1) When the XTAL oscillation fault function is active, this bit is cleared to zero at the same time and the MRC oscillates. 2) Stop mode wakeup action when PWC_STPMCR.CKSMRC bit is 1, set when MRC oscillator is in oscillation.	R/W

Caution:

- When MRC is selected as the system clock source, MRCSTP writing "1" is disabled to stop the MRC clock.
- The software sets the MRC oscillation and waits for 5 MRC cycles before entering Stop Mode, Power Down Mode, or Stop MRC.
- The software sets the MRC to stop and waits for 5 MRC cycles before it can enter stop mode, power down mode or start the MRC again.
- MRC is used as RTC calibration clock. possibility of MRC oscillation when RTC is not initialized. possibility of MRC oscillation when RTC calibration function is active, ignoring MRCSTP bit setting.

4.11.13 CMU LRC Calibration Register (CMU_LRCTRM)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
LRCTRM[7:0]							

position	Marker	Place Name	Function	Reading and writing
b7~b0	LRCTRM[7:0]	LRC frequency calibration bits	10000000: -128 10000001: -127 11111111: -1 00000000: Center Code 00000001: +1 01111110: +126 01111111: +127	R/W

Caution:

-Frequency calibration The frequency calibration needs to be within the LRC frequency guarantee.

4.11.14 CMU LRC Control Register (CMU_LRCCR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	LCRSTP

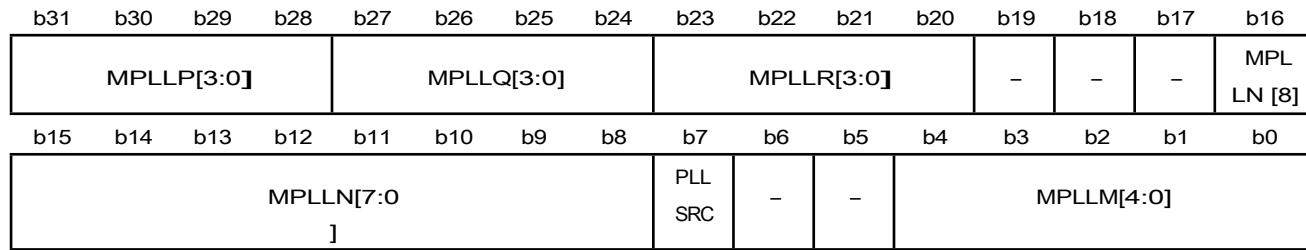
position	Marker	Place Name	Function	Reading and writing
b31~b1	Reserved	-	Read "0" for read write "0" for write	R/W
b0	LCRSTP	LRC oscillator on stop bit	0: LRC oscillator oscillation 1: LRC oscillator stop	R/W

Caution:

- When LRC is selected as the system clock source, disable LRCSTP to write "1" to stop the LRC clock.
- The software sets the LRC action to start and waits for 5 LRC cycles before going into stop mode, power down mode, or stopping the LRC.
- The software sets the LRC to stop and waits for 5 LRC cycles before it can enter stop mode, power down mode, or start the LRC again.
- Wait for XTAL oscillator, HRC, MPLL, and UPLL clocks to stabilize when the LRCSTP bit is set to ignore and the LRC forces oscillation.
- When the RTC selects LRC as the clock source, LRC ignores this register bit and LRC oscillates. when the RTC is not initialized, LRC has the possibility of oscillating.

4.11.15 CMU MPLL Configuration Register (CMU_PLLCFGR)

Reset value: 0x1110_1300



position	Marker	Place Name	Function	Reading and writing
b31-b28	MPLL P[3:0]	MPLL dividing factor for system clock	Frequency used for MPLLP clock to write MPLLP in MPLL stop condition MPLL output clock frequency = VCO frequency of MPLL/MPLL 0000: Forbidden to set 0001: 2-way frequency 0010: 3-way frequency 0011: 4-way frequency 1101: 14 crossover 1110:15 Crossover 1111: 16 crossover frequencies	R/W
b27-b24	MPLL Q[3:0]	MPLL dividing factor for system clock	Frequency used for MPLLQ clock to write MPLLQ in MPLL stop condition MPLL output clock frequency = VCO frequency of MPLL/MPLLQ 0000: Forbidden to set 0001: 2-way frequency 0010: 3-way frequency 0011: 4-way frequency 1101: 14 crossover 1110:15 Crossover 1111: 16 crossover frequencies	R/W
b23-b20	MPLL R[3:0]	MPLL dividing factor for system clock	Frequency used for MPLLR clock to write MPLLR under MPLL stop condition MPLL output clock frequency = VCO frequency of MPLL/MPLL R 0000: Forbidden to set 0001: 2-way frequency 0010: 3-way frequency 0011: 4-way frequency 1101: 14 crossover 1110:15 Crossover 1111: 16 crossover frequencies	R/W
b19-b17	-	-	Read "0" for read write "0" for write	R/W
b16-b8	MPLL N[8:0]	MPLL frequency multiplication factor	Used to control the multiplication factor of the VCO of the MPLL, write MPLLN under the MPLL stop condition. ensure that the VCO frequency of the MPLL is between 240MHz and 480MHz. VCO frequency of MPLL = PFD input frequency of MPLL * MPLLN 000010011: 20 000010100: 21	R/W

000010101: 22
 000010110: 23
 11011101: 478
 111011110: 479
 111011111: 480

b7	PLLSRC	MPLL/UPLL input clock source selection	0: Select external high-speed oscillator as input clock for MPLL/UPLL 1: Select the internal high-speed oscillator as the input clock for MPLL/UPLL	R/W
b6-b5	-	-	Read "0" for read write "0" for write	R/W
b4-b0	MPLL M[4:0]	MPLL input clock division factor	Used to divide the MPLL input clock prior to the VCO of the MPLL. Write MPLLM under MPLL stop conditions. ensure that the PFD input frequency of MPLL is between 1MHz and 25MHz. 00000: 1 crossover frequency 00001: 2-way frequency 00010: 3-way frequency 10111: 24 crossover other prohibitions	R/W

4.11.16 CMU MPPLL control memory (CMU_PLLCR)

Reset value: 0x01

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	MPLLOFF

position	Marker	Place Name	Function	Reading and writing
b7~b1	Reserved	-	Read "0" for read write "0" for write	R/W
b0	MPLLOFF	MPLL Enable	Do not set this bit to 1 if the MPLL clock is used as the system clock. 0: MPLL action starts 1: MPLL stop	R/W

Caution:

- When MPLL is selected as the system clock source, disable MPLLOFF from writing "1" to stop the MPLL clock.
- The software sets the MPLL action to start, and the MPLL can only enter the stop mode after confirming the MPLL is stable by the MPLLSTBF bit, power-down mode or software setting to stop the MPLL.
- The software sets the MPLL to stop and confirms the MPLL is stopped by the MPLLSTBF bit before it can enter stop mode, power down mode or start the MPLL again.
- When the MPLL selects the XTAL oscillator as the clock source, the MPLL action can be set to start only after the XTAL oscillator is confirmed to be stable by the XTALSTBF bit. when the MPLL selects the HRC as the clock source, the MPLL action can be set to start only after the HRC is confirmed to be stable by the HRCSTBF bit.

4.11.17 CMU UPLL Configuration Register (CMU_UPLLCFGR)

Reset value: 0x1110_1300

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
UPLL[3:0]				UPLLQ[3:0]				UPLL[3:0]				-	-	-	UPL LN [8]
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
UPLL[7:0]								-	-	-	UPLL[4:0]				
position	Marker	Place Name	Function	R/W											
b31-b28	UPLL[3:0]	System clock with UPLL dividing factor	Used to control the frequency of UPLL clock and write UPLL in UPLL stop condition. UPLL output clock frequency = VCO frequency of UPLL/UPLL 0000: Forbidden to set 0001: 2-way frequency 0010: 3-way frequency 0011: 4-way frequency 1101: 14 crossover 1110:15 Crossover 1111: 16 crossover frequencies	R/W											
b27-b24	UPLLQ[3:0]	System clock with UPLL dividing factor	Used to control the frequency of UPLLQ clock and write UPLLQ in UPLL stop condition. UPLL output clock frequency = VCO frequency of UPLL/UPLLQ 0000: Forbidden to set 0001: 2-way frequency 0010: 3-way frequency 0011: 4-way frequency 1101: 14 crossover 1110:15 Crossover 1111: 16 crossover frequencies	R/W											
b23-b20	UPLL[3:0]	System clock with UPLL dividing factor	Used to control the frequency of UPLL[3:0] clock and write UPLL[3:0] under UPLL stop condition. UPLL output clock frequency = VCO frequency of UPLL/UPLL[3:0] 0000: Forbidden to set 0001: 2-way frequency 0010: 3-way frequency 0011: 4-way frequency 1101: 14 crossover 1110:15 Crossover 1111: 16 crossover frequencies	R/W											
b19-b17	-	-	Read "0" for read, write "0" for write	R/W											
b16-b8	UPLL[8:0]	UPLL frequency multiplication factor	The multiplication factor used to control the VCO of the UPLL. Write UPLL[8:0] in the UPLL stop condition. Ensure that the VCO frequency of the UPLL is between 240MHz and 480MHz. VCO frequency of the UPLL = PFD input frequency of the UPLL * UPLL[8:0] 000010011: 20 000010100: 21 000010101: 22	R/W											

000010110: 23
 111011101: 478
 111011110: 479
 111011111: 480
 Other prohibited settings

b7-b5	-	-	Read "0" for read write "0" for write	R/W
b4-b0	UPLL[4:0]	UPLL input clock division factor	Used to divide the UPLL input clock prior to the VCO of the UPLL. Write UPLLM under UPLL stop conditions. ensure that the PFD input frequency of UPLL is between 1MHz and 25MHz. 00000: Prohibit setting 00001: 2-way frequency 00010: 3-way frequency 10111: 24 crossover	R/W

4.11.18 CMU UPLL control memory (CMU_UPLLCSR)

Reset value: 0x01

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	UPLLOFF

position	Marker	Place Name	Function	Reading and writing
b7~b1	Reserved	-	Read "0" for read write "0" for write	R/W
b0	UPLLOFF	UPLL Enable	Used to start and stop UPLL. 0: UPLL action starts 1: UPLL Stop	R/W

Caution:

- When UPLL is selected as the clock source for I2S/TRNG/ADC/USBFS, disable UPLLOFF to write "1" to stop the UPLL clock.
- The software sets the UPLL action to start and confirms the UPLL is stable by the UPLLSTBF bit before entering the stop mode, power-down mode or software setting to stop the UPLL.
- The software sets the UPLL to stop and confirms the UPLL is stopped by the UPLLSTBF bit before entering the stop mode, power-down mode or starting the UPLL again.
- When the XTAL oscillator is selected as the clock source for the UPLL, the XTAL oscillator can be set to start the UPLL operation only after the XTALSTBF bit is used to confirm that the XTAL oscillator is stable, and when the HRC is selected as the clock source for the UPLL, the HRCSTBF bit is used to confirm that the HRC is stable before the UPLL operation can be set to start.

4.11.19 CMU Clock Source Stabilization Stateer (CMU_OSCSTBSR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	UPLLSTBF	MPLLSTBF	-	XTALSTBF	-	-	HRCSTBF

position	Marker	Place Name	Function	Reading and writing
b7	Reserved	-	Read "0" for read write "0" for write	R
b6	UPLLSTBF	UPLL stability flag bit	0: UPLL stopped or not stable 1: UPLL stable	R
b5	MPLLSTBF	MPLL stability flag bit	0: MPLL stopped or not stable 1: MPLL stable	R
b4	Reserved	-	Read "0" for read write "0" for write	R
b3	XTALSTBF	XTAL stability flag bit	0: XTAL stopped or not stabilized 1: XTAL stable	R
b2~b1	Reserved	-	Read "0" for read write "0" for write	R
b0	HRCSTBF	HRC stability flag bit	0: HRC stopped or not stabilized 1: HRC stable	R

4.11.20 CMU System Clock Source Switching Register (CMU_CKSWR)

Reset value: 0x01

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-		CKSW[2:0]

position	Marker	Place Name	Function	Reading and writing
b31~b3	Reserved	-	Read "0" for read write "0" for write	R/W
b2-b0	CKSW[2:0]	System clock source switching	000: Select HRC clock as system clock 001: Select MRC clock as system clock 010: Select LRC clock as system clock 011: Select XTAL clock as system clock 100: Select XTAL32 clock as system clock 101: Select MPLL as system clock 110: Prohibit setting 111: Prohibition of setting note: 1, switch the target clock source, need to ensure that in the clock stable state. 2, process refer to [clock source switching] chapter 3, When PWC_STPMCR.CKSMRC bit is 1, this register is initialized after stop mode wake-up, and MRC clock is selected as the system clock source.	R/W

4.11.21 CMU clock division configuration register (CMU_SCFGR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	HCLKS[2:0]			-	EXCKS[2:0]			-	PCLK4S[2:0]		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	PCLK3S[2:0]			-	PCLK2S[2:0]			-	PCLK1S[2:0]			-	PCLK0S[2:0]		

position	Marker	Place Name	Function	Reading and writing
b31~b27	Reserved	-	Read "0" for read write "0" for write	R/W
b26~24	HCLKS[2:0]	HCLK clock division selection bit	000: 1 division of the system clock 001: 2 divisions of the system clock 010: 4 divisions of the system clock 011: 8 divisions of the system clock 100: 16 divisions of the system clock 101: 32 divisions of the system clock 110: 64 divisions of the system clock 111: Prohibit setting Note: When PWC_STPMCR.CKSMRC bit is 1, this register is initialized after stop mode wake-up, and HCLK is 1 division of the system clock.	R/W
b23	Reserved	-	Read "0" for read write "0" for write	R/W
b22~20	EXCKS[2:0]	ExMC clock division selection bit	000: 1 division of the system clock 001: 2 divisions of the system clock 010: 4 divisions of the system clock 011: 8 divisions of the system clock 100: 16 divisions of the system clock 101: 32 divisions of the system clock 110: 64 divisions of the system clock 111: Prohibit setting Note: When PWC_STPMCR.CKSMRC bit is 1, this register is initialized after stop mode wake-up, and EXCLK is 1 division of the system clock.	R/W
b19	Reserved	-	Read "0" for read write "0" for write	R/W
b18~16	PCLK4S[2:0]	PCLK4 clock division selection bit	000: 1 division of the system clock 001: 2 divisions of the system clock 010: 4 divisions of the system clock 011: 8 divisions of the system clock 100: 16 divisions of the system clock 101: 32 divisions of the system clock 110: 64 divisions of the system clock 111: Prohibit setting Note: When PWC_STPMCR.CKSMRC bit is 1, this register is initialized after stop mode wake-up, and PCLK4 is 1 division of the system clock.	R/W
b15	Reserved	-	Read "0" for read write "0" for write	R/W

b14~12	PCLK3S[2:0]	PCLK3 clock division selection bit	000: 1 division of the system clock 001: 2 divisions of the system clock 010: 4 divisions of the system clock 011: 8 divisions of the system clock	R/W
--------	-------------	------------------------------------	---	-----

100: 16 divisions of the system clock
 101: 32 divisions of the system clock
 110: 64 divisions of the system clock
 111: Prohibit setting
 Note: When PWC_STPMCR.CKSMRC bit is 1, this register is initialized after stop mode wake-up, and PCLK3 is 1 division of the system clock.

b11	Reserved	-	Read "0" for read write "0" for write	R/W
b10~8	PCLK2S[2:0]	PCLK2 clock division selection bit	000: 1 division of the system clock 001: 2 divisions of the system clock 010: 4 divisions of the system clock 011: 8 divisions of the system clock 100: 16 divisions of the system clock 101: 32 divisions of the system clock 110: 64 divisions of the system clock 111: Prohibit setting Note: When PWC_STPMCR.CKSMRC bit is 1, this register is initialized after stop mode wake-up, and PCLK2 is 1 division of the system clock.	R/W
b7	Reserved	-	Read "0" for read write "0" for write	R/W
b6~4	PCLK1S[2:0]	PCLK1 clock division selection bit	000: 1 division of the system clock 001: 2 divisions of the system clock 010: 4 divisions of the system clock 011: 8 divisions of the system clock 100: 16 divisions of the system clock 101: 32 divisions of the system clock 110: 64 divisions of the system clock 111: Prohibit setting Note: When PWC_STPMCR.CKSMRC bit is 1, this register is initialized after stop mode wake-up, and PCLK1 is 1 division of the system clock.	R/W
b3	Reserved	-	Read "0" for read write "0" for write	R/W
b2~0	PCLK0S[2:0]	PCLK0 clock division selection bit	000: 1 division of the system clock 001: 2 divisions of the system clock 010: 4 divisions of the system clock 011: 8 divisions of the system clock 100: 16 divisions of the system clock 101: 32 divisions of the system clock 110: 64 divisions of the system clock 111: Prohibit setting Note: When PWC_STPMCR.CKSMRC bit is 1, this register is initialized after stop mode wake-up, and PCLK0 is 1 division of the system clock.	R/W

4.11.22 CMU USBFS clock configuration memory (CMU_USBCKCFGR)

Reset value: 0x40

b7	b6	b5	b4	b3	b2	b1	b0		
USBCKS[3:0]		-		-		-			
position	Marker	Place Name	Function						
b7~b4		USBCKS[3:0]		USB-FS at 48MHz	0010: System clock 2 division				
				Zhong source selection	0011: System clock 3 division				
					0100: System clock 4 divisions				
					1000: MPLL/P				
					1001: MPLL/Q				
					1010: MPLL/R				
					1011: UPLL/P				
					1100: UPLL/Q				
					1101: UPLL/R				
					Other				
					prohibited				
					settingsNote:				
					1. When the target clock source for switching is MPLL/UPLL, it is necessary to ensure that the MPLL/UPLL clock is in a stable state.				
					2. When the system clock selects MPLL, you need to set USB, CAN, QSPI, SPI, Universal Timer, FCM, ADC, DAC to module stop state, and then write CMU_SCFGR register to switch the clock division frequency. After writing the CMU_USBCKCFGR register, the software waits for the system stabilization time of 30us.				
					3. When the PWC_STPMCR.CKSMRC bit is 1, after the stop mode wakes up, this send The memory is initialized with USBCLK as a 4-minute frequency of the system clock.				
b3~b0	Reserved	-	Read "0" for read write "0" for write						

4.11.23 CMU AD/TRNG clock configuration memory (CMU_PERICKSEL)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PERICKSEL[3:0]

position	Marker	Place Name	Function	Reading and writing
b15~b4	Reserved	-	Read "0" for read write "1" for write	R/W
b3~b0	PERICKSEL[3:0]	AD/TRNG clock source selection	0000: PCLK2/PCLK4 set by CMU_SCFG 1000: MPLL/P 1001: MPLL/Q 1010: MPLL/R 1011: UPLL/P 1100: UPLL/Q 1101: UPLL/R Setting other than this is prohibited.	R/W

Caution:

- If the target clock source for switching is MPLL/UPLL, the MPLL/UPLL clock steady state must be ensured.

4.11.24 CMU I2S clock configuration memory (CMU_I2SCKSEL)

Reset value: 0xBBB

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
I2S4CKSELc				I2S3CKSEL[3:0]				I2S2CKSEL[3:0]				I2S1CKSEL[3:0]			
position		Marker		Place Name				Function				Reading and writing			
b15~b12		I2S4CKSEL		I2S clock source selection				0000: PCLK3 set by CMU_SCFG 0001: MPPLL/P 0010: MPPLL/Q 0011: MPPLL/R 0100: UPLL/P 0101: UPLL/Q 0110: UPLL/R 0111: UPLL/R Other than this setting is prohibited. Note: No clock for settings other than the above note				R/W			
b11~b8		I2S3CKSEL		I2S clock source selection				0000: PCLK3 set by CMU_SCFG 0001: MPPLL/P 0010: MPPLL/Q 0011: MPPLL/R 0100: UPLL/P 0101: UPLL/Q 0110: UPLL/R 0111: UPLL/R Other than this setting is prohibited. Note: No clock for settings other than the above note				R/W			
b7~b4		I2S2CKSEL		I2S clock source selection				0000: PCLK3 set by CMU_SCFG 0001: MPPLL/P 0010: MPPLL/Q 0011: MPPLL/R 0100: UPLL/P 0101: UPLL/Q 0110: UPLL/R 0111: UPLL/R Other than this setting is prohibited. Note: No clock for settings other than the above note				R/W			
b3~b0		I2S1CKSEL		I2S clock source selection				0000: PCLK3 set by CMU_SCFG 0001: MPPLL/P 0010: MPPLL/Q 0011: MPPLL/R 0100: UPLL/P 0101: UPLL/Q 0110: UPLL/R 0111: UPLL/R Other than this setting is prohibited. Note: No clock for settings other than the above note				R/W			

Caution:

- When the target clock source for switching is MPPLL/UPLL, it is necessary to ensure that the MPPLL/UPLL oscillates in a stable state.
- When MPPLL/UPLL is selected as the target clock source, refer to the Clock Controller

(CMU) Details on how to configure MPLL/UPLL.

4.11.25 CMU Debug Clock Configuration Memory (CMU_TPIUCKCFGR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
TPIUCKOE	-	-	-	-	-	-	TPIUCKS[1:0]

position	Marker	Place Name	Function	Reading and writing
b7	TPIUCKOE	TPIU clock supply allow bit	0: Forbidden 1: Allowed	R/W
b6~b2	-	-	Read "0" for read write "0" for write	R/W
b1~0	TPIUCKS[1:0]	TPIU clock division selection bit	00:1 Crossover 01: 2-way frequency 10:4 crossover Other prohibited settings	R/W

4.11.26 CMU MCO1 configuration memory (CMU_MCO1CFGR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
MCO1EN		MCO1DIV[2:0]			MCO1SEL[3:0]		

position	Marker	Place Name	Function	Reading and writing
b7	MCO1EN	MCO_1 output license	0: MCO_1 output is disabled 1: Allow MCO_1 output	R/W
b6~b4	MCO1DIV[2:0]	MCO_1 crossover frequency selection	000: 1 crossover 001: 2-way 010: 4-way 011: 8-way frequency 100: 16 crossover frequencies 101: 32 crossover 110: 64 crossover 111: 128 crossover	R/W
b3~b0	MCO1SEL[3:0]	MCO_1 clock source selection	0000: HRC Clock 0001: MRC clock 0010: LRC clock 0011: XTAL clock 0100: XTAL32 clock 0110: MPLLP 0111: UPLL 1000 : MPLLQ 1001: UPLLQ 1011: System clock other prohibited settings.	R/W

4.11.27 CMU MCO2 Configuration Memory (CMU_MCO2CFGR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
MCO2EN		MCO2DIV[2:0]			MCO2SEL[3: 0]		

position	Marker	Place Name	Function	Reading and writing
b7	MCO2EN	MCO_2 output license	0: MCO_2 output is disabled 1: Allow MCO_2 output	R/W
b6~b4	MCO2DIV[2:0]	MCO_2 crossover selection	000: 1 crossover 001: 2-way 010: 4-way 011: 8-way frequency 100: 16 crossover frequencies 101: 32 crossover 110: 64 crossover 111: 128 crossover	R/W
b3~b0	MCO2SEL[3: 0]	MCO_2 Clock Source Selection	0000: HRC Clock 0001: MRC clock 0010: LRC clock 0011: XTAL clock 0100: XTAL32 clock 0110: MPLLP 0111: UPLL 1000 : MPLLQ 1001: UPLLQ 1011: System clock other prohibited settings.	R/W

4.11.28 FCM lower limit comparison value register (FCM_LVR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LVR[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Read "0" for read write "0" for write	R/W
b15~b0	LVR[15:0]	Lower limit comparison value	This register is configured when the START bit is 0.	R/W

4.11.29 FCM upper limit comparison value register (FCM_UVR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
UVR[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Read "0" for read write "0" for write	R/W
b15~b0	UVR[15:0]	Upper limit comparison value	This register is configured when the START bit is 0.	R/W

4.11.30 FCM Counter Value Register (FCM_CNTR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CNTR[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Read "0" for read write "0" for write	R/W
b15~b0	CNTR[15:0]	Counter value	When a valid edge selected by the EDGES bit of the reference clock is detected, the counter value is saved to this register (except for the first valid edge after START=1)	R

4.11.31 FCM Start Stop Register (FCM_STR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	STA RT

position	Marker	Place Name	Function	Reading and writing
b31~b1	Reserved	-	Read "0" for read write "0" for write	R/W
b0	START	Frequency measurement start position	0: Frequency measurement stops 1: Start of frequency measurement	R/W

4.11.32 FCM Measurement Object Control Register (FCM_MCCR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	MCKS[3:0]	-	-	-	-	-	MDIVS[1:0]

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Read "0" for read write "0" for write	R/W
b7~b4	MCKS[3:0]	Measurement object clock selection bit	0000: XTAL 0001: XTAL32 0010: HRC 0011: LRC 0100: SWDTRC 0101: PCLK1 0110: UPLL 0111: MRC 1000: MPLLP 1001: RTCLRC Others: Set the ban	R/W
b3~b2	Reserved	-	Read "0" for read write "0" for write	R/W
b1~b0	MDIVS[1:0]	Measurement object crossover frequency selection	00: No crossover 01: 4-way frequency 10: 8 Crossover 11: 32 Crossover	R/W

4.11.33 FCM Measurement Reference Control Register (FCM_RCCR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EXR EFE	-	EDGES [1:0] -	-	-	DNFS[1:0]	INE XS	RCKS[3:0]	-	RDIVS[1:0]						

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Read "0" for read write "0" for write	R/W
b15	EXREFE	External pin input reference clock reference clock FCMREF allow bit	0: Disable external pin input reference clock FCMREF 1: Allow external pin input reference clock FCMREF	R/W
b14	Reserved	-	Read "0" for read write "0" for write	R/W
b13~b12	EDGES [1:0] -	Measurement reference edge selection bit	00: Rising edge 01: Falling edge 10: Rising and falling edges 11: Prohibit setting	R/W
b11~b10	Reserved	-	Read "0" for read write "0" for write	R/W
b9~b8	DNFS[1:0]	Digital filter function selection bit	00: No filter function 01: MCKS bit selected clock as filter clock 10: 4 divisions of the MCKS bit-selected clock as the filter clock 11: 16 divisions of the MCKS bit-selected clock as the filter clock	R/W
b7	INEXS	Measurement reference, internal clock and terminal selection bits	0: External pin input reference clock FCMREF 1: Clock selected by the RCKS selection bit	R/W
b6~b3	RCKS[3:0]	Measurement reference clock selection bit	0000: XTAL 0001: XTAL32 0010: HRC 0011: LRC 0100: SWDTRC 0101: PCLK1 0110: UPLL 0111: MRC 1000: MPLLP 1001: RTCLRC Others: Set the ban	R/W
b2	Reserved	-	Read "0" for read write "0" for write	R/W
b1~b0	RDIVS[1:0]	Measurement reference crossover frequency selection	00:32 Crossover 01: 128 crossover frequency 10: 1024 crossover frequencies 11: 8192 crossover frequency	R/W

4.11.34 FCM Interrupt Reset Control Register (FCM_RIER)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	ERR E	-	-	ERRI NTRS	-	OVF IE	MEN DIE	ERR IE

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Read "0" for read write "0" for write	R/W
b7	ERRE	Frequency abnormal reset allowed bit	0: Forbidden 1: Allowed	R/W
b6~b5	Reserved	-	Read "0" for read write "0" for write	R/W
b4	ERRINTRS	Frequency abnormal interrupt select bit	0: Frequency abnormal interruption occurs 1: Frequency abnormal occurrence reset	R/W
b3	Reserved	-	Read "0" for read write "0" for write	R/W
b2	OVFIE	Counter overflow interrupt allow bit	0: Disable counter overflow interrupt 1: Allow counter overflow interrupts	R/W
b1	MENDIE	Measurement end interrupt allow bit	0: Interruption at the end of measurement is prohibited 1: Allow interruptions to occur at the end of the measurement	R/W
b0	ERRIE	Frequency exception interrupt allow bit	0: Interruption of frequency abnormalities is prohibited 1: Allow frequency abnormalities to occur interruptions	R/W

4.11.35 FCM flag register (FCM_SR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	OVF	MEN DF	ERR F

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Read "0" for read write "0" for write	R/W
b2	OVF	Counter overflow flag bit	0: Counter not overflowed 1: Counter overflow	R
b1	MENDF	Measurement end flag bit	0: Measurement in progress 1: End of measurement	R
b0	ERRF	Frequency exception flag bit	0: No frequency anomaly occurs 1: Abnormal frequency of occurrence	R

4.11.36 FCM flag bit clear register (FCM_CLR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	OVF CLR	MEN DFC LR	ERR FCLR

position	Marker	Place Name	Function	Reading and writing
b31~b3	Reserved	-	Read "0" for read write "0" for write	R/W
b2	OVFCLR	Counter overflow flag clear bit	Write "" to clear the counter overflow flag bit	W
b1	MENDFCLR	Measurement end marker zero position	Write "1" to clear the end-of-measurement flag bit	W
b0	ERRFCLR	Frequency exception flag clear bit	Write "1" to clear the end-of-measurement flag bit	W

5 Power Control (PWC)

5.1 Introduction

The power controller is used to control the power supply, switching, and detection of multiple power domains of the chip in multiple operation modes and low power modes. The power controller consists of a power consumption control logic (PWCL) and a power supply voltage detection unit (PWD).

The chip operates from 1.8 V to 3.6 V. The voltage regulator (LDO) supplies power to the VDD and VDDR domains, and the VDDR voltage regulator (RLDO) supplies power to the VDDR domain during power-down mode. The chip provides three modes of operation, including ultra-high speed, high speed, and ultra-low speed, and three low-power modes, including sleep, stop, and power-down, through the power control logic (PWCL).

The power supply voltage detection unit (PWD) provides power-on reset (POR), power-down reset (PDR), under-voltage reset (BOR), programmable voltage detection 1 (PWD1), programmable voltage detection 2 (PWD2), etc. Among them, POR, PDR, BOR control the chip reset action by detecting the VCC voltage. PWD2 generates reset or interrupt by detecting VCC voltage or external input detection voltage, and generates reset or interrupt by register selection.

The VDDR area can maintain power through the RLDO after the chip goes into power-down mode, ensuring that the real-time clock module (RTC), wake-up timer (WKT), and can continue to operate and keep data in the 4KB low-power SRAM (Ret-SRAM). The analog module is equipped with dedicated power supply pins to improve analog performance.

5.2 Power Distribution

Figure 5-1 shows the power distribution of the chip. The chip is composed of VCC domain, VDD power domain, AVCC power domain, and VDDR domain.

The VCC domain is powered through the VCC/VSS pin and consists of power consumption control logic (PWCL), power supply voltage detection unit (PWD), IO level hold circuit, voltage regulator (LDO), VDDR domain regulator (RLDO), and oscillator circuit. The oscillator circuit includes an external high-speed oscillator (XTAL), an external low-speed oscillator (XTAL32), and an internal low-speed oscillator (LRC).

The VDD domain consists of the CPU, digital logic such as digital peripherals, RAM, FLASH, etc., and is powered by the VDD generated by the LDO. In the VDD domain RAM is divided into 4 groups and can be powered off independently by register control.

The VDDR domain consists of 4KB hold RAM (Ret-SRAM), real-time clock (RTC), and wake-up timer

(WKTM). It is powered by RLDO in power-down mode and by LDO in modes other than power-down mode. In power-down mode, **the** Ret-SRAM can hold data, the real-time clock RTC and the wake-up timer WKTM can continue to operate. When the function of the VDDR domain is not needed, the power consumption can be further reduced by setting PWC_PWRC0.VVDRSD to disconnect the VDDR domain in the power-down mode.

The analog power domain mainly consists of the digital-to-analog converter (ADC), comparator (CMP), programmable gain amplifier (PGA), and input and output pins of the analog system, powered by the AVCC/AVSS pins. In order to provide high precision analog performance, the analog area is equipped with a separate power supply. To ensure higher accuracy of the ADC, a dedicated pin is used for the ADC's reference voltage VREFH.

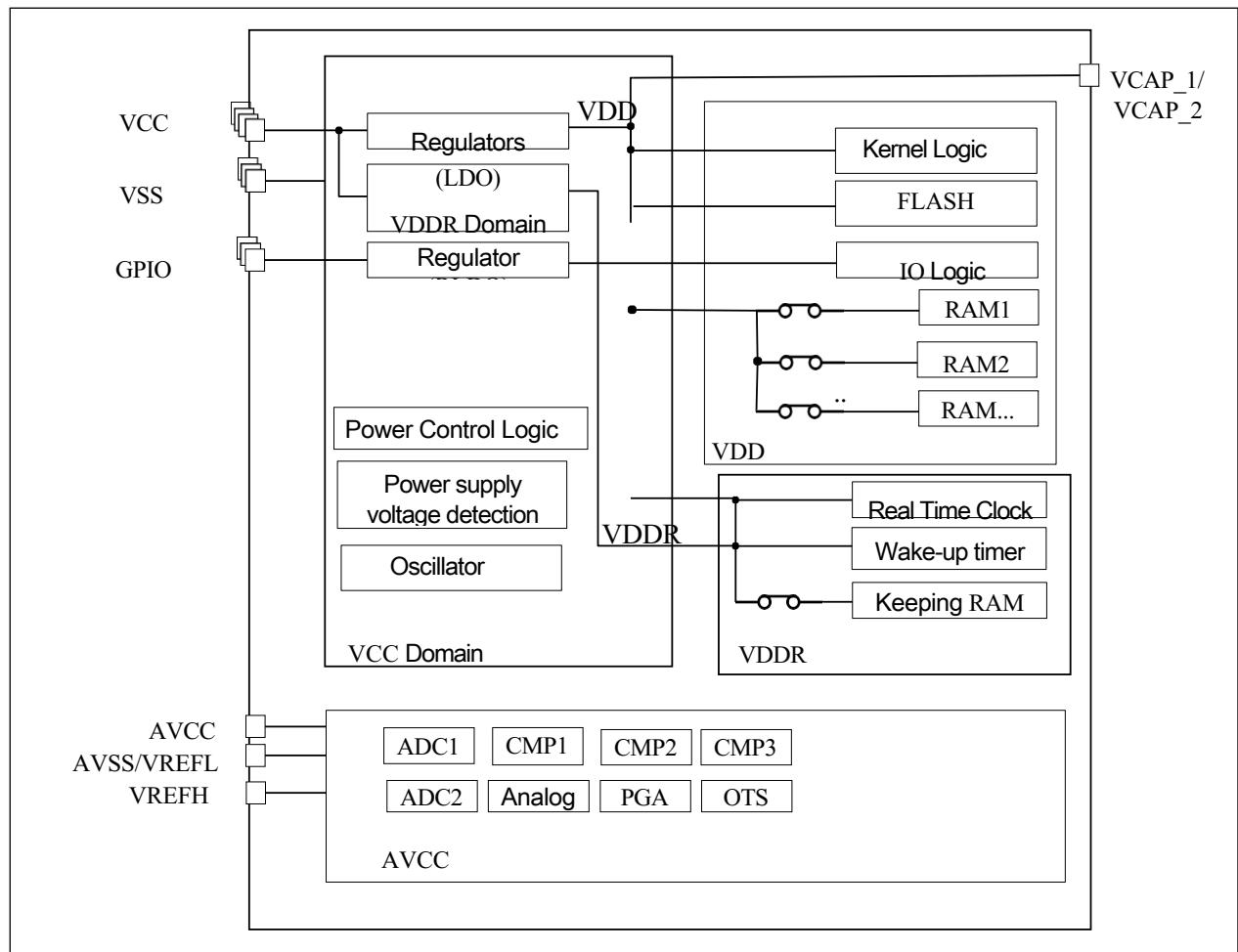


Figure 5-1 Power supply composition diagram

5.3 Description of the power supply voltage detection unit (PVD)

The power supply voltage detection unit (PVD) includes power-on reset (POR), power-down reset (PDR), under-voltage reset (BOR), programmable voltage detection 1 (PVD1), and programmable voltage detection 2 (PVD2).

5.3.1 Power-on reset/power-down reset action description

The chip has integrated power-on reset and power-down reset circuits. The waveform of power-on reset and power-down reset is shown in Figure 5-2. When VCC is higher than the specified threshold V_{POR} , the chip will release the power-on reset state after $TRSTPOR$ time, and the CPU starts to execute the code. When VCC is lower than V_{PDR} , the chip maintains the reset state. When using power-on reset, the reset pin NRST must be 1. If the reset pin is pulled down, the chip will be reset and started by pin reset.

For more information on parameters such as V_{POR} , V_{PDR} , $TRSTPOR$, etc., please refer to **the electrical characteristics in the data sheet**.

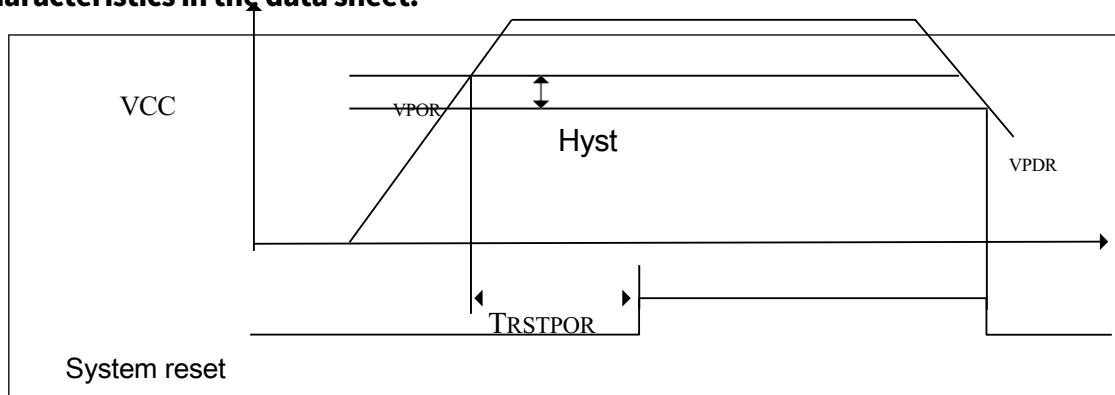


Figure 5-2 Power-on reset and power-down reset
waveforms

5.3.2 Undervoltage Reset (BOR) Description

During power-up, until VCC is above V_{BOR} , an undervoltage reset (BOR) will put the chip in reset.

The V_{BOR} threshold is configured via the BOR_lev, BORDIS of the initialization configuration bit (ICG). when BORDIS=0, the BOR detection voltage can be selected from 4 thresholds. when BORDIS is configured to 1, the chip is reset controlled via power-on reset, power-down reset.

BORDIS	BOR_lev	Description
1	XX	BOR is not valid
0	00	BOR valid, BOR threshold 0 selected (VBOR0)
0	01	BOR valid, BOR threshold 1 selected (VBOR1)
0	10	BOR valid, select BOR threshold 2 (VBOR2)
0	11	BOR valid, select BOR threshold 3 (VBOR3)

Table 5-1 BOR Configuration

For the electrical characteristics of the BOR threshold, refer to Electrical Characteristics.

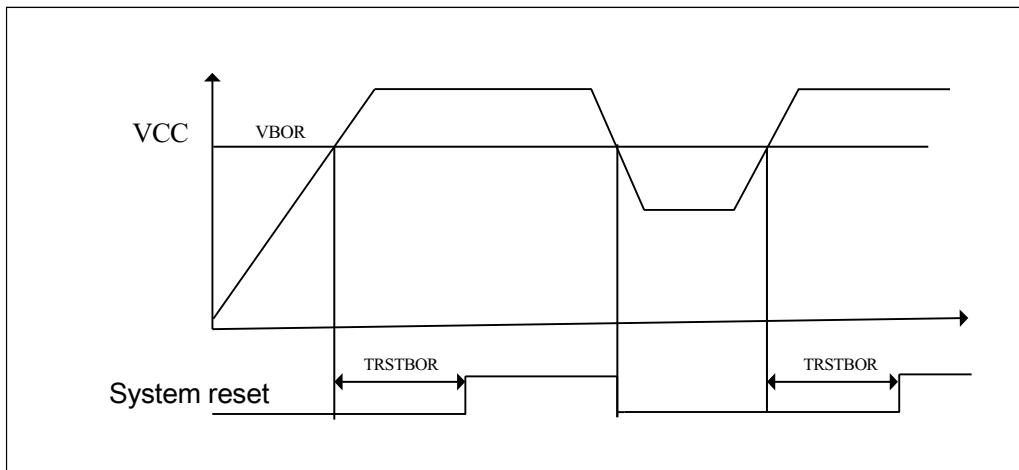


Figure 5-3 Undervoltage reset waveform

5.3.3 Programmable voltage detection 1 (PVD1), programmable voltage detection 2 (PVD2)

Programmable Voltage Detect 1 and Programmable Voltage Detect 2 trigger a reset or interrupt by detecting whether the VCC supply voltage passes the detection threshold. Each detection circuit is programmable.

The event can be programmed and configured as a reset/interrupt (maskable/non-maskable)/peripheral circuit trigger function when the power supply voltage passes the threshold voltage point of each detection circuit.

The main characteristics of the programmable voltage detection are shown in Table 5-2.

Table 5-2 PVD1/PVD2 Characteristics

Proj ects	PVD1	PVD2
Test Object	Whether the threshold voltage point (VPVD1) is passed during VCC fall/rise	<ol style="list-style-type: none">1. PWC_PVDLCR.PVD2LVL[2:0] set Whether the threshold voltage point (VPVD2) is passed during VCC fall/rise when the value is other than 1112. PWC_PVDLCR.PVD2LVL[2:0]=111 Is the drop/rise process of the external input voltage After the threshold voltage point (VPVD2)
Detection voltage point	Configured by PVD1LVL[2:0]	Configured by PVD2LVL[2:0]
Reset	Reset: VCC<VPVD1; Reset release: VCC> VPVD1 after a certain Reset processing time.	Reset: VCC<VPVD2; Reset release: VCC> VPVD2 after a certain reset processing time.
Interruptions	Configured as voltage detect 1 interrupt or non-maskable interrupt VCC drops past the threshold voltage point (VPVD1)	Configured as voltage detect2 interrupt or non-maskable interrupt VCC drops past the threshold voltage point (VPVD2)
Filter function	Digital filtering	Digital filtering
Peripheral circuit trigger function	VCC drops past the threshold voltage point (VPVD1)	VCC drops past the threshold voltage point (VPVD2)

5.3.4 PVD1, PVD2 interrupt/reset block diagram

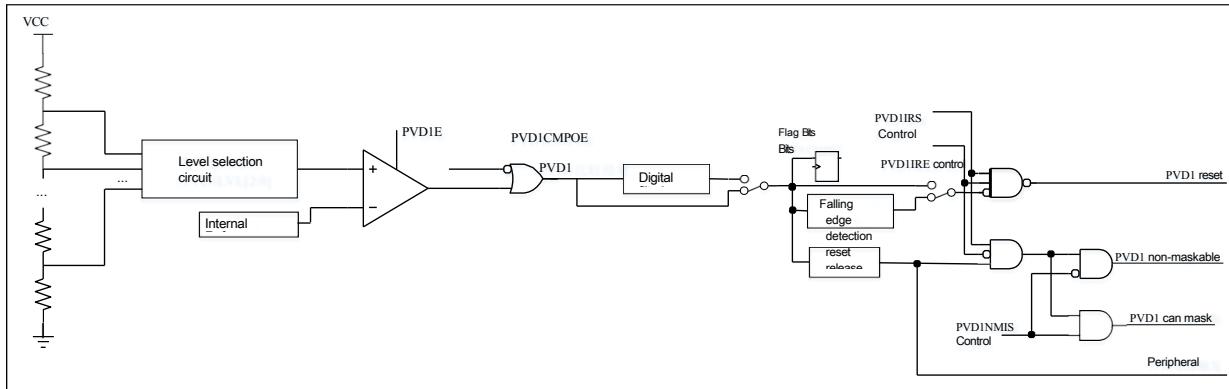


Figure 5-4 PVD1 interrupt/reset block diagram

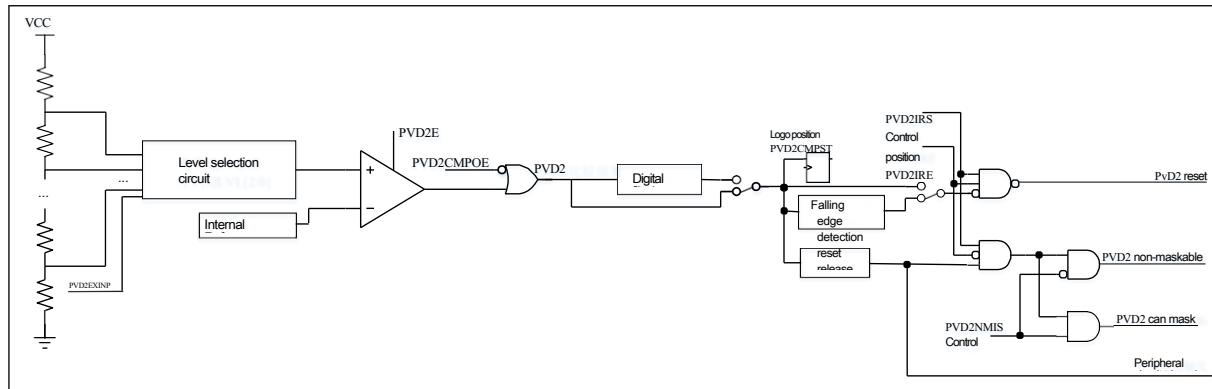


Figure 5-5 Interrupt/reset block diagram

5.3.5 Input/output

pins

Pin Name	Input/output	Function
PVD2EXINP	Input	External input PVD2 comparison voltage

5.3.6 PVD1 Interrupt and Reset

When using the PVD1 circuit in stop mode or power-down mode, observe the following precautions.

1. Stop Mode

- 1) The digital filter must be invalidated.

2. Power down mode

- 1) The digital filter must be invalidated.

2) PVD1IRS is set to 0 and PVD1 is selected to generate interrupt; when reset function is selected, power-down mode cannot be entered. The following diagram shows the operation timing of the voltage monitoring 1 interrupt, PVD1DETFLG needs to be cleared to zero before the interrupt can occur again.

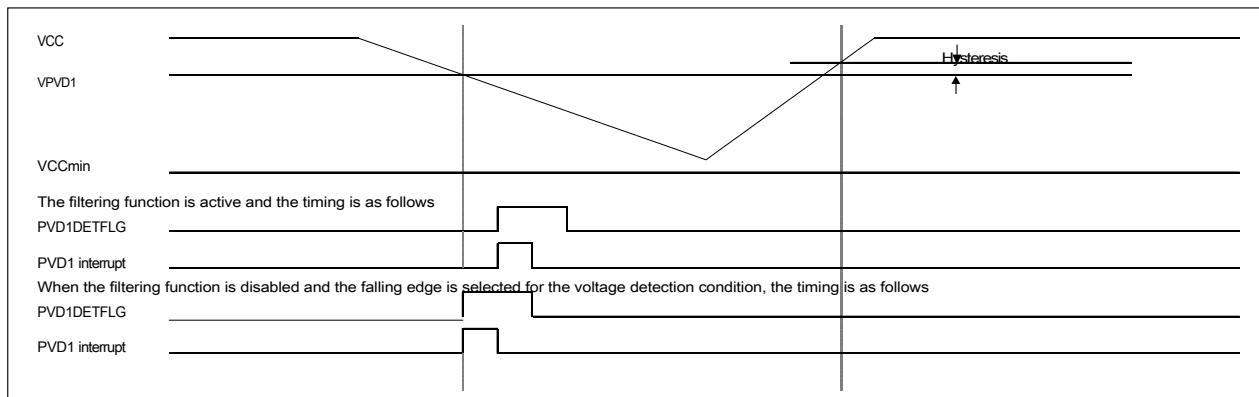


Figure 5-6 Power Monitor 1

Interrupt Timing Diagram

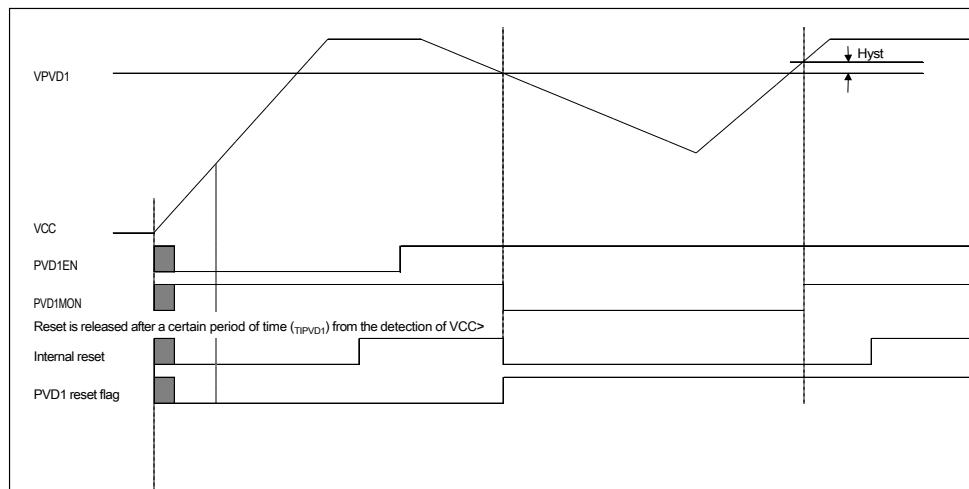


Figure 5-7 Power Monitor 1

Reset Timing Diagram

5.3.7 PVD2 Interrupt and Reset

When using the PVD2 circuit in stop mode or power-down mode, observe the following precautions:

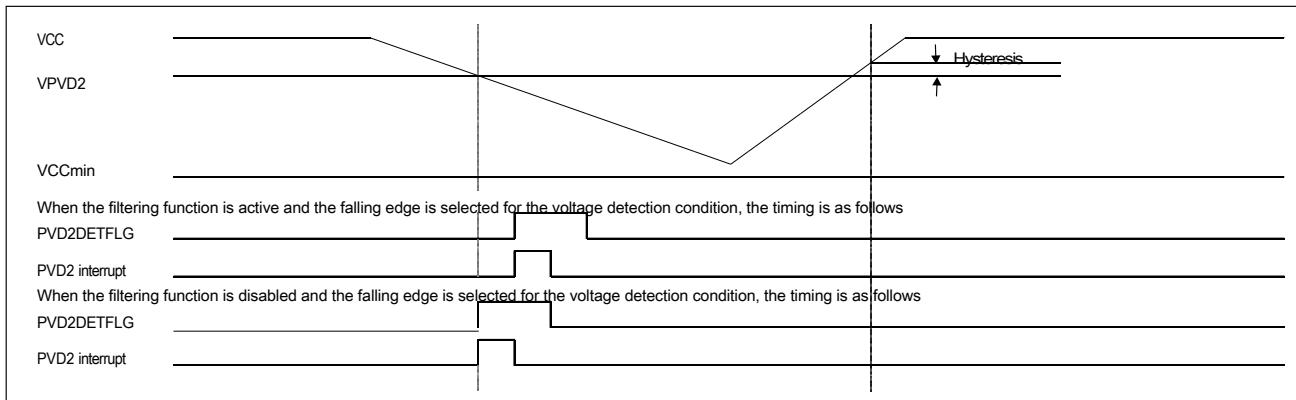
1. Stop Mode

- 1) The digital filter must be invalidated.

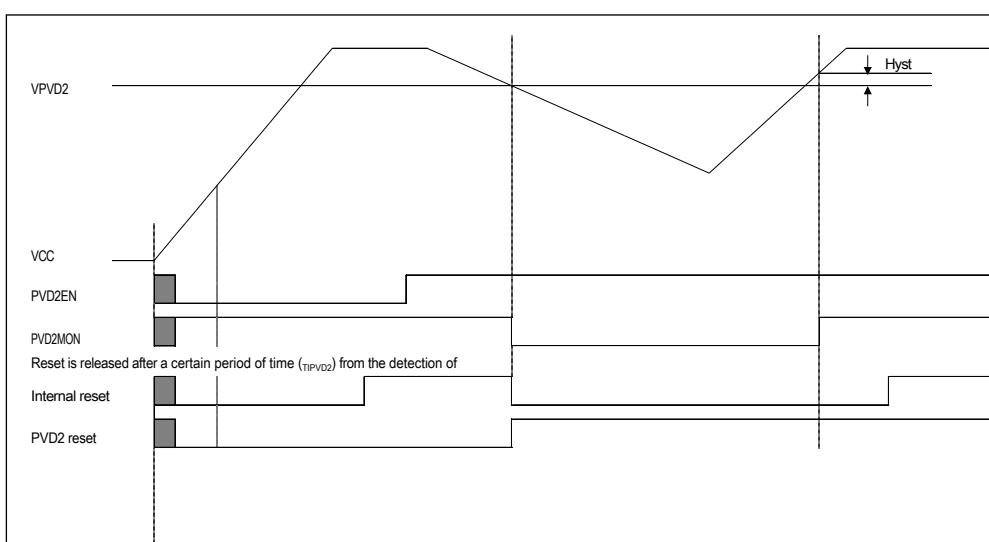
2. Power down mode

- 1) The digital filter must be invalidated.

2) PVD2IRS is set to 0 to select PVD2 to generate an interrupt; when the reset function is selected, the power-down mode cannot be entered. The following diagram shows the operation timing of the voltage monitoring 2 interrupt. PVD2DETFLG needs to be cleared before the interrupt can occur again.



**Figure 5-8 Power Monitor 2
Interrupt Operation Timing
Diagram**



**Figure 5-9 Power Monitor 2
Reset Operation Timing
Diagram**

5.3.8 Internal voltage sampling and detection function

The internal voltage sampling and detection function of the chip refers to the reference voltage measurement function. The reference voltage measurement path, through the ADC, measures the reference voltage. The internal reference voltage is approximately 1.15V.

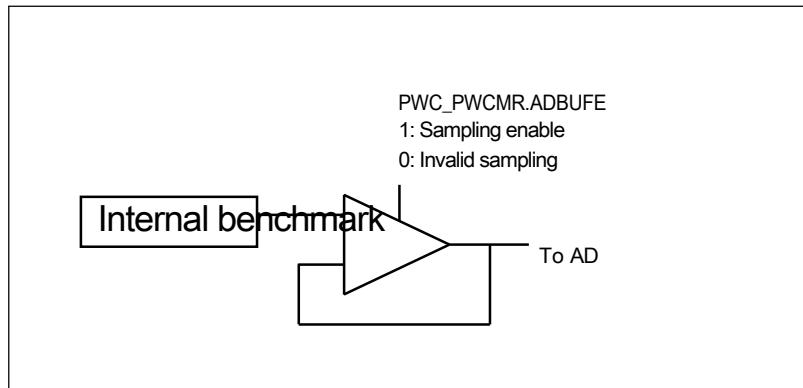


Figure 5-10 Internal voltage sampling diagram

- Baseline voltage measurement path

To use the reference voltage measurement path, you need to select the reference voltage measurement path according to the following procedure.

1. PWC_PWCMR.ADBUFE=1, enable internal voltage measurement function
2. Configure the registers according to Table 16-2 in the ADC section to enable the ADC to select the internal voltage measurement channel
3. Wait 50uS and use the ADC to measure the internal reference voltage

5.4 Motion mode and low power mode

After a system reset or power-on reset, all power domains of the chip are powered and the chip enters high-speed operation mode. In the run mode, the CPU provides the clock via HCLK and executes the program code. The chip provides three operation modes, such as ultra-high-speed operation mode, high-speed operation mode, and ultra-low-speed operation mode. The operation modes that can be configured are shown in Table 5-3.

In order to save the power consumption when the CPU does not need to run, the system provides three low-power modes, such as sleep mode, stop mode, and power-down mode. The low-power modes that the chip can be configured are shown in Table 5-4. In sleep mode, the chip's CortexTM-M4F core stops and the peripherals remain running; in stop mode, both the peripherals and the CPU stop; in power-down mode, the power in the VDD domain is turned off and the peripherals in the VDD domain stop. The real-time clock and wake-up timer located in the VDDR domain can operate in the low-power mode, and the hold SRAM can hold data; when the real-time clock, wake-up timer and hold SRAM in the VDDR domain are not needed, the regulator RLDO in the VDDR domain can be set to be turned off, and the power consumption can be further reduced after entering the power-down mode.

Users can choose between run mode and low-power mode depending on the application to find the best balance between low power consumption, short boot time, wake-able sources, and system execution efficiency.

The operating conditions of the low-power mode and the status of each module in the low-power mode are shown in Table 5-5.

Table 5-3 Operation Mode

Operation mode	Description
Ultra-high speed operation mode (Ultra high Speed Run Mode)	Main frequency below 200MHz
High-speed operation mode (High Speed Run Mode)	Main frequency below 168MHz
Ultra low speed operation mode (Ultra low Speed Run Mode)	Main frequency below 8MHz

Table 5-4 Low Power Mode

Mode	Description
Sleep Mode	CPU clock stops, peripherals stay running
Stop Mode	Both chip peripherals and CPU clocks stop
Power Down	Power-down mode 1 VDD domain power down

Mode	(PDMD1)	
	Power down mode 2 (PDMD2)	The voltage detection unit is invalid outside the VDD domain power down
	Power down mode 3 (PDMD3)	In addition to the VDD domain power-down, the VDDR domain power-down, the power-on reset circuit enters the low-power mode, and the voltage detection unit (PVD) is invalid. Compared with the power-down mode 4, the power-down wake-up except PWC_PDWKF0/PWC_PDWKF1/RSTF0 external chip is completely reset.
	Power down mode 4 (PDMD4)	In addition to VDD domain power down, VDDR domain power down, power-on reset circuit into low power mode, voltage Detection unit (PVD) is invalid

**Table 5-5 Operating conditions of low-power mode
and the state of each module in low-power mode**

Proj ects	Sleep mode	Stop Mode	Power down mode
Enter	PWC_STPMCR.STOP=0 PWC_PWRC0.PWDN=0, WFI	PWC_STPMCR.STOP=1 PWC_PWRC0.PWDN=0, WFI	PWC_STPMCR.STOP=1 PWC_PWRC0.PWDN=1, WFI
Release	Arbitrary interrupt or reset	Interrupts or resets that can be used in stop mode	Wake-up events or resets that can be used in power-down mode
External high-speed oscillator	Work can be set	Stop	Stop
External low-speed oscillator	Work can be set	Work can be set	Work can be set
Internal high-speed oscillator	Work can be set	Stop	Power down
Internal medium speed oscillator	Work can be set	Stop	Power down
Internal low-speed oscillator	Work can be set	Work can be set	Work can be set
WDT dedicated clock oscillator	Work can be set	Work can be set	Power down
PLLU	Work can be set	Stop	Power down
PLLM	Work can be set	Stop	Power down
CPU	Stop (Hold)	Stop (Hold)	Power down
RAM	Work can be set Can be set to work, power down	Stop (Hold) Power down or sleep can be maintained depending on the setting before entering standby	Power down
FLASH	Work can be set	Stop (Hold)	Power down, content retention
DMA	Work can be set	Stop (Hold)	Power down
Regulators	Jobs The driver can be adjusted	Jobs The driver can be adjusted	Stop
Power-on reset circuit	Jobs	Jobs	Jobs Power-down mode 1, power-down mode 2 reset circuit accuracy can be guaranteed, power-down mode 3 and power-up reset power in power-down mode 4 Pressure is not guaranteed

Undervoltage reset BOR	Work can be set	Work can be set	Power down mode 1 working can be set Power down mode 2/3/4 in stop
Voltage detection module PVD	Work can be set	Work can be set	Power down mode 1 working can be set Power down mode 2/3/4 in stop
WDT	Work can be set	Stop (Hold)	Power down
SWDT	Work can be set	Work can be set	Power down
RTC	Work can be set	Work can be set	Power down mode 1/2 work can set power down mode 3/4 under power down
USB-FS	Work can be set	Stop (Hold)	Power down
Timer0	Work can be set	Work can be set	Power down
Ret-SRAM	Work can be set	Stop (Hold)	Power down mode 1/2 under stop (hold)

Proj ects	Sleep mode	Stop Mode	Power down mode
	Can be set to work, power down, sleep	Power down, sleep can be set	Power down can be set, sleep power down mode 3/4 power down
WKTM	Work can be set	Work can be set	Power down mode 1/2 work can set power down mode 3/4 under power down
Other Peripheral Modules	Work can be set	Stop (Hold)	Power down
AD	Work can be set	Stop	Power down
DA	Work can be set	Work can be set	Power down
PGA	Work can be set	Work can be set	Power down
CMP	Work can be set	Work can be set	Power down
pa0-pa10, pb0-pb2, pb5-pb10, pb12-pb15, pc0-pc13, pd0-pd15, pe0-PE15, PH2	Work can be set	Maintain	Hold or high resistance
PC14-PC15	Work can be set	When used as an external low-speed oscillator pin, keep the oscillator operating; when set to GPIO or other peripheral functions, set to keep both pins as the same Leveling	When set to GPIO or other peripheral function, the state of PC14 and PC15 can be set to hold or high resistance, please set to keep both pins at the same level
PH0-PH1	Work can be set	When used as an external high-speed oscillator, the oscillator stops oscillating and the pin state remains the same before entering STOP mode; set to GPIO or other peripheral functions When the state before STOP is maintained	When used as an external high-speed oscillator, the oscillator stops oscillating and the pin state remains the same before entering STOP mode; set to GPIO or other peripheral functions When the state before STOP is maintained
NRST reset pin	The outside of the chip is pulled up to VCC via a circuit	Outside the chip is pulled up to VCC through a resistor	Outside the chip is pulled up to VCC through a resistor

PA11-PA12	Work can be set	Maintenance; As the level of this pin is pulled high a redundant current is generated into the STOP module The pull-up is prohibited when the style is	Holding or high resistance; Due to the redundant current generated when the level of this pin is pulled high, it enters power-down mode Pull-up is prohibited when pulling up.
PB11/MD	Work can be set	Maintenance;	Maintenance; Outside the chip is pulled up to VCC through a resistor
PA13-PA15, PB3,PB4	Work can be set. The built-in pull-up circuit is effective when used as a JTAG function	Maintenance; The built-in pull-up circuit is effective when used as a JTAG function	Maintenance; The built-in pull-up circuit is effective when used as a JTAG function

5.4.1 Operation mode

The chip has three operation modes, such as ultra-high speed, high speed and ultra-low speed. According to the system speed requirement, the best operation mode is set to adapt the core voltage and drive capability to the system clock frequency, so as to achieve the purpose of reducing power consumption. As shown in Table 5-6, according to the DVS and DDAS bits of PWC_PWRC2, the chip can be made to operate in the corresponding operation mode. When the chip is selected to operate in ultra-low speed mode, FLASH and RAM also need to be set to operate at low core voltage, so the register bits EFM_FRMC.LVM=1 and PWC_RAMOPM=0x9062 need to be set.

Table 5-6 Operation Mode Description

Operati on mode	Frequen cy range	Register Setting	
		PWC_PWRC2.DVS	PWC_PWRC2.DDAS
Ultra-high speed operation mode	≤200MHz	00	1111
High-speed operation mode	≤168MHz or less	11	1111
Ultra-low speed operation mode*	≤8MHz or less	10	1000

* : Ultra-low speed mode only supports Ta = -40°C~85°C

To switch between ultra-high speed operation mode, high speed operation mode and ultra-low speed operation mode, the following flow 1~flow 6 should be followed.

1. High-speed mode to ultra-low-speed mode switching

- 1) Set the clock source to be used in ultra-low speed mode to ensure that the clock source meets the frequency requirements in ultra-low speed mode
- 2) Turn off clock sources and modules that are not needed in ultra-low speed mode and make sure that the FLASH is not programmed or erased
- 3) LVM=1, RAM action mode register PWC_RAMOPM is set to 0x9062
- 4) Confirm FRMC.LVM=1, PWC_RAMOPM=0x9062
- 5) Set PWC_PWRC2.DDAS[3:0] to 1000; PWC_PWRC2.DVS[1:0] to 10
- 6) Write PWC_MDSWCR=0x10
- 7) Waiting for TSWMD1(30uS)
- 8) Chip action in ultra-low speed mode

2. Switching from ultra-low speed mode to high speed mode

-
- 1) Set PWC_PWRC2. DDAS[3:0] to 1111; PWC_PWRC2_. DVS[1:0] according to the system frequency to be
Request a setting of 11
 - 2) Write PWC_MDSWCR=0x10
 - 3) Waiting for TSWMD2(30uS)
 - 4) LVM=0, RAM action mode register PWC_RAMOPM is set to 0x8043
 - 5) Confirm FRMC.LVM=0, PWC_RAMOPM=0x8043
 - 6) Chip action in high speed mode

3. High speed mode to super speed mode switching

- 1) Set PWC_PWRC2.DDAS[3:0] to 1111; PWC_PWRC2_.DVS[1:0] to 00
- 2) Write PWC_MDSWCR =0x10
- 3) Wait T SWMD2 (30uS)
- 4) Chip action in ultra-high speed mode

4. Switching from super speed mode to high speed mode

- 1) Set the clock source to be used in high speed mode to ensure that the clock source meets the frequency requirements for high speed mode
- 2) PWC_PWRC2.DDAS[3:0] is set to 1111; PWC_PWRC2_.DVS[1:0] is set to 11
- 3) Write PWC_MDSWCR =0x10
- 4) Waiting for TSWMD2(30uS)
- 5) Chip action in high speed mode

5. Switching from ultra-low speed mode to ultra-high speed mode

- 1) Set PWC_PWRC2.DDAS[3:0] to 1111; PWC_PWRC2_.DVS[1:0] to 00
- 2) Write PWC_MDSWCR =0x10
- 3) Waiting for TSWMD2(30uS)
- 4) LVM=0, RAM action mode register PWC_RAMOPM is set to 0x8043
- 5) Confirm FRMC_LVM=0, PWC_RAMOPM=0x8043
- 6) Chip action in ultra-high speed mode

6. Switching from ultra-high speed mode to ultra-low speed mode

- 1) Set the clock source to be used in ultra-low speed mode to ensure that the clock source meets the frequency requirements in ultra-low speed mode
- 2) Turn off clock sources and modules that are not needed in ultra-low speed mode and make sure that the FLASH is not programmed or erased
- 3) LVM=1, RAM action mode register PWC_RAMOPM is set to 0x9062
- 4) Confirm FRMC_LVM=1, RAMOPT=0x9062
- 5) Set PWC_PWRC2.DDAS[3:0] to 1000; PWC_PWRC2.DVS[1:0] to 10
- 6) Write PWC_MDSWCR =0x10
- 7) Wait for TSWMD1(30uS)
- 8) Chip action in ultra-low speed mode

5.4.2 Sleep mode

In sleep mode, the CPU stops running and its internal registers remain in the same state as before it entered sleep mode. The action state of peripherals and other system modules other than watchdogs and dedicated watchdogs is not changed.

If the ICG is set to auto-start, the watchdog stops counting in sleep mode if the WDTSLPOFF bit of the ICG is 1. If the WDTSLPOFF bit is 0, the watchdog continues counting in sleep mode. If the ICG is not set to auto-start and the watchdog is started by software start, the watchdog stops counting in sleep mode if the WDT_CR.SLPOFF bit is 1; if the WDT_CR.SLPOFF bit is 0, the watchdog does not stop counting in sleep mode.

When set to auto-start via ICG, the dedicated watchdog stops counting in sleep mode if the SWDTSLPOFF bit of ICG is 1. If the SWDTSLPOFF bit is 0, the dedicated watchdog continues counting in sleep mode.

- Enter sleep mode
 - Execute the **VVF1** command at PWC_STPMCR.STOP=0 to enter sleep mode.
- Exit sleep mode
 - Any interrupt or reset can wake up the chip from sleep mode. When waking up by interrupt, the chip enters the interrupt handler; when exiting the sleep mode by reset, the chip enters the reset state.

5.4.3 Stop Mode

In stop mode, the CPU, most peripherals and clock sources stop acting. The chip maintains the CPU internal registers and SRAM data, peripheral states and pin states. In stop mode, the chip power consumption is significantly reduced because most of the clock sources stop operating and the regulator has reduced driving capability.

When set to auto-start via ICG, the dedicated watchdog stops counting in stop mode if the SWDTSLPOFF bit of ICG is 1. If the SWDTSLPOFF bit is 0, the dedicated watchdog continues counting in stop mode.

Before executing the **VVF1** command to enter the stop mode, you need to make sure that the FLASH is not programmed or erased, and the oscillation stop monitoring function is disabled, otherwise the chip will enter the sleep mode instead of the stop mode.

In stop mode, the ADC and DAC will also consume power unless they are disabled before entering stop mode. DAE, DAOE0, DAOE1 need to be cleared to "0" to disable DAC. To disable the ADC, clear the ADC_STR.START bit to "0" and set the register to "0".

Write 1 to the ADC corresponding bit in PWC_FCG3 to put the ADC into the module stop state, and then execute the WFI instruction to enter the stop mode.

When waking up in STOP mode, the bits CKSMRC and FLNWT in the PWC_STPMCR register are used to select the clock after waking up and whether or not to wait for the FLASH to stabilize. CKSMRC is used to control the clock source after waking up.

When CKSMRC =1, the system clock source after wake-up is MRC; when CKSMRC=0, the system clock after wake-up remains the same as the clock source before entering STOP. FLNWT is used to control whether to wait for FLASH to stabilize after wake-up, when FLNWT=0, it is necessary to wait for FLASH to stabilize after wake-up; when FLNWT=1, it is not necessary to wait for FLASH to stabilize after wake-up.

Otherwise, the action of the chip after waking up from STOP is not guaranteed. Selecting CKSMRC =1 and FLNWT=1 will wake up the system in the shortest possible time when the program is running on RAM and entering STOP mode.

Before executing the **WFI** command to enter the stop mode, you need to make sure that the DMA is in the stop state, otherwise the chip may have an unguaranteed action.

The leakage current in STOP mode is different at different voltage temperatures, and the drive capability must be set to meet the leakage needs of the chip.

Before executing the **WFI** instruction to enter the stop mode, the digital filtering of EIRQ needs to be set to invalid, otherwise the interrupt cannot be used for STOP wake-up.

STPDAS=11 before entering STOP mode; if PWC_PWRC1_STPDAS=00 when entering STOP mode, the chip will consume more current in STOP mode.

To release the stop mode by non-maskable interrupt, set the corresponding bit of INT_NMIER to enable the interrupt; to release the stop mode by maskable interrupt, set the corresponding bit of INT_WUPENR register to enable the wake-up permission of the interrupt. Before executing the **WFI** or WFE command, make sure all interrupts that are not used for wake-up in stop mode have been turned off.

- Enter stop mode

Execute the WFI instruction when PWC_STPMCR.STOP=1 and PWC_PWRC0_PWDN=0 to enter the stop mode. Table 5-5 gives the states of the peripherals and clock sources of the chip in the stop mode.

- Release stop mode

Stop mode can be de-activated by reset and interrupt. The resets that can be used to release the stop mode are pin reset, power-on reset, undervoltage reset (BOR), programmable voltage monitoring 1/2 reset, and dedicated watchdog reset. The interrupt events that can be used to disengage the stop

mode are as follows:
■ NMI interrupt, interrupt EIRQ0-15, voltage monitoring 1 interrupt,
■ Voltage monitor2 interrupt, dedicated watchdog underflow interrupt, cycle
interrupt for real-time clock, alarm clock interrupt, wakeup timer interrupt,

When the chip is released from stop mode by interrupt, it first starts the clock sources used before entering stop mode. After all clock sources are stabilized, the chip is released from stop mode.

5.4.4 Power down mode

In the power-down mode, the power to all modules in the VDD domain is cut off and power consumption can be minimized.

When set to auto-start via ICG, if the SWDTSLPOFF bit in ICG is 1, the dedicated watchdog will be

powered off and will no longer count, as with other modules in the VDD domain. If the SWDTSLPOFF bit is 0, the chip will enter stop mode instead of power-down mode, and the dedicated watchdog's oscillator and dedicated watchdog will continue to operate if set to auto-start in the ICG.

When the reset of voltage monitor 1 and voltage monitor 2 is enabled, the chip will enter the stop mode instead of the power-down mode.

Before executing the **VVF1** command to enter the power-down mode, you need to make sure that the FLASH is not programmed or erased and the oscillation stop monitoring function is disabled, otherwise the chip will enter the sleep mode instead of the power-down mode.

The capacitors used in the VCAP_1/VCAP_2 pins of the chip are as follows: 1) For chips with both VCAP_1 and VCAP_2 pins, each pin can use 0.047uF or 0.1uF capacitors (total capacity of 0.094uF or

0.2uF)2)For chips with only VCAP_1 pin, either 0.1uF or 0.22uF capacitor can be used. When waking up from power-down mode, VCAP_1/VCAP_2 needs to be charged during the core voltage build-up. On the one hand, a smaller total VCAP_1/VCAP_2 capacity reduces the charge time and brings fast response time to the application; on the other hand, a larger total VCAP_1/VCAP_2 capacity extends the charge time, but also provides better electromagnetic compatibility (EMC). Users can choose smaller or larger capacitors depending on the EMC and system response speed requirements. The total capacity of VCAP_1/VCAP_2 must match the assignment of the PWC_PWRC3.PDTS bit. If the total capacity of VCAP_1/VCAP_2 is 0.094uF or 0.1uF, you need to make sure the PWC_PWRC3.PDTS bit is cleared before entering power-down mode.

The power consumption in power-down mode can be further reduced by setting PWC_PWRC0.PDMDS[1:0]. The sub-modes of the power-down mode are shown in Table 5-7. In power-down mode 1, the voltage monitoring circuit can be used and the power-on reset detection circuit is in action. Since the wake-up does not need to wait for the stability of the VCC domain reference voltage circuit, the voltage monitoring circuit and the power-on reset detection circuit, the wake-up time is the shortest while achieving low power consumption. In power-down mode 2, the VCC domain reference voltage circuit and voltage monitoring circuit stop working, and the power-on reset detection circuit is in the active state. In power-down mode 3, the VCC domain reference voltage circuit, voltage monitoring circuit, and power-on reset detection circuit stop working, and it is necessary to wait for the stabilization of these circuits when waking up. Therefore, while achieving the lowest power consumption, the wake-up time is longer than that of power-down mode 2 and power-down mode 1. Power-down mode 4 is the same as stopping in power-down mode 3. The same circuitry is used to operate, so power-down mode 4 has the same power consumption as power-down mode 3. For specific power consumption values and wake-up time, please refer to [Electrical Characteristics - Low Power Mode].

The VDDR domain works in power down mode 1 and power down mode 2, so the real time clock module, wake-up timer can continue to operate and can be used to wake up in power down mode. ret-SRAM can still hold data in power down mode. If the real-time clock, wake-up timer, and Ret-SRAM are not needed in power-down mode, the low-power regulator can be turned off by setting PWC_PWRC0.VVDRSD to further reduce power consumption. In power-down mode 3 and power-

down mode 4, the VDDR domain is also powered down.

Table 5-7 Power down mode sub-mode

Power down mode	PDMD[1:0]	Power consumption	Wake-up time	Description
Power down mode 1	00	IPD1	TPD1	VCC domain power supply voltage detection unit is valid
Power down mode 2	10	IPD2	TPD2	VCC domain POR, PDR detection circuit valid, BOR, PVD1, PVD2 invalid
Power down mode 3	01	IPD3	TPD3	VCC domain POR, PDR, BOR, PVD1, PVD2 invalid VDDR domain power down
Power down mode 4	11	IPD4	TPD4	VCC domain POR, PDR, BOR, PVD1, PVD2 invalid VDDR domain power down

The relationship between power consumption and wake-up time: $I_{pd1} > I_{pd2} > I_{pd3} = I_{pd4}$,
 $T_{pd1} < T_{pd2} < T_{pd4} < T_{pd3}$

- Enter power-down mode

Execute **VVF** command when PWC_STPMCR.STOP=1,PWC_PWRC0.PWDN=1 to enter power down mode.

- Disengage power-down mode

Power-down mode can be released by a power-down mode wake-up event or a reset. The resets that can be used to wake up power-down mode are pin reset, power-on reset, and voltage monitor 0

Reset Events that can be used to Wake up Power-down mode include: alarm and timing event of real-time clock, voltage monitoring 1 wake-up event, voltage monitoring 2 wake-up

After waking up from power-down mode 1 or power-down mode 2, the chip resets and re-executes the program. The wake-up event can be queried by the power-down wake-up flag bit, and the reset flag bit can be queried by RSTF0.PDRF.

In power-down mode 3, POR, PDR, BOR, PVD1, PVD2 circuits are invalid. After waking up from power-down mode 3, all registers except PWC_PDWKF0/PWC_PDWKF1/RSTF0 are reset and the chip works in a similar way as power-on reset; the reset flag bit can be queried by RSTF0.PDRF. After waking up from power-down mode 3, the RTC/WKTM of VDDR domain is reset and the data of Ret-SRAM is not guaranteed.

In power-down mode 4, POR, PDR, BOR, PVD1, and PVD2 circuits are invalid, and the chip resets and re-executes the program; the reset flag bit can be queried by RSTF0.PDRF. When waking up from power-down mode 4, the RTC/WKTM of VDDR domain is reset, **and the** data of Ret-SRAM is not guaranteed.

The power-down wake-up event is controlled by the power-down wake-up enable register (PWC_PDWKE0-PDWKE3) and the power-down wake-up event edge select register (PWC_PDWKES). **When a power-down wake-up event occurs, the corresponding power-down wake-up flag (PWC_PDWKF0-PWC_PDWKF1) is set.** After power-down wake-up, if the power-down wake-up flag is not cleared, the chip cannot enter the power-down mode again. The edge of the power-down wake-up event can be selected by PWC_PDWKES.

When waking up in power-down mode, the VDD domain will be re-powered and the system performs a power-down wake-up reset with the operating clock internal medium speed oscillator.

Power down mode	Registers that are not reset
Power down mode 1	pwc_pwrc0 pwc_pwrc1 pwc_pwrc3 pwc_pdwke0 pwc_pdwke1 pwc_pdwke2 pwc_pdwkes pwc_pdwkf0 pwc_pdwkf1 pwc_pwmr pwc_xtal32cs pwc_pvdcr0 PWC_PVDCR1 PWC_PVDFCR PWC_PVDLCR PWC_PVDICR
Power down mode 2	PWC_PVDDSR

- Pin status after unplugging power-down mode

In power-down mode, the chip pins will keep the state before entering power-down mode or set to high resistance state by register. If PWC_PWRC0.IORTN[1:0]=10 or 11, the pin state is high resistance in power down mode and is initialized after power down mode is released. If PWC_PWRC0.IORTN[1:0]=00, the pin state remains in the power-down mode before the power-down mode, and the pin is initialized to the high-resistance state after waking up. If PWC_PWRC0.IORTN[1:0]=01, the chip pin will keep the state before entering power-down mode, and the state of the chip pin will not change after waking up even if the register of the peripheral or pin is set. After PWC_PWRC0.IORTN is cleared to zero by software, the pin state will be controlled by the register setting of peripheral or pin.

- WKTM pin for power-down wake-up

The chip has built-in counter WKTM for power-down wake-up, which can select internal low-speed oscillator, external low-speed oscillator, and 64Hz internal clock signal as clock source, where 64Hz internal clock signal is valid when the RTC uses external low-speed oscillator as clock source and the RTC is active. WKTCE is set, the counter starts to count and when the count value is equal to the WKT CMP[11:0] setting, the count stops and a wake-up event is generated to wake up the chip from power-down mode. WKTC0.WKTCE needs to be reset and set again when WKTM is used again.

- PTWK Power-down mode wake-up event

PTWK is used to wake up the chip in power-down mode, which is set by software. Each PTWK event can be selected from 4 pins, with either rising or falling trigger edge, with independent flag bits. the structure block diagram of PTWK_n is shown in Figure 5-11 PTWK_n structure block diagram.

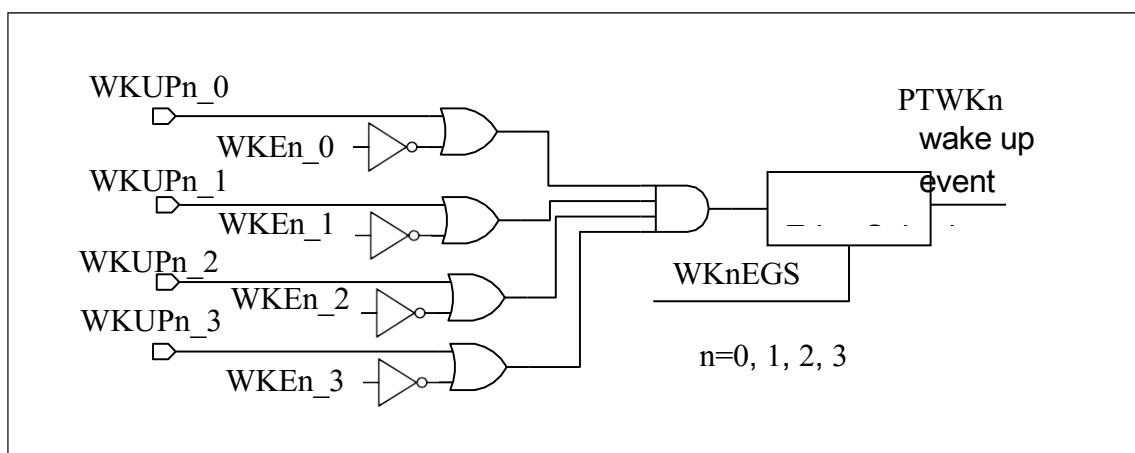


Figure 5-11 PTWK_n structure block diagram

- Description of the action of the VDDR domain in power-down mode

The VDDR domain continues to be powered via RLDO after the chip enters power-down mode

1 or power-down mode 2, so the RTC/WKTM/Ret-SRAM can continue to operate or hold data. The VDDR domain is powered by the LDO after power-down wake-up. In power-down mode 3 or power-down mode 4 the VDDR domain is not powered.

5.5 Ways to reduce power consumption

Power consumption in operating mode can be optimized by the following methods.

1. Set the optimal operation mode
2. Reduced system clock speed
3. Turn off clock sources that are not in use
4. Set function clock control register PWC_FCGn (n=0/1/2/3) to turn off the function that is not needed to be used
5. Powering off the RAM

5.5.1 Reduced system clock speed

The system clock (HCLK), external bus clock (EXCLK), and peripheral clock PCLK0/PCLK1/PCLK2/PCLK3/PCLK4 can be slowed down by programming the prescaler registers in Run mode. These prescalers can also be used to reduce the peripheral speed before entering sleep mode. For more information, refer to [Clock Controller (CMU)]

5.5.2 Turn off clock sources that are not in use

The chip's system clock has six clock sources:

- External high-speed oscillator (XTAL)
- External low-speed oscillator (XTAL32)
- MPLL Clock (MPLL)
- Internal high-speed oscillator (HRC)
- Internal medium speed oscillator (MRC)
- Internal low speed oscillator (LRC)

SWDT has a separate dedicated internal low-speed oscillator (SWDTLRC); RTC can select either an external low-speed oscillator or an internal low-speed oscillator as the clock source. In addition, the chip is equipped with a UPLL clock source for I2S.

Each clock source can be turned off individually when not in use, reducing system power consumption; both the HRC and PLL are equipped with separate power supply circuits that can be used to further reduce power consumption by disconnecting power to the HRC after the HRC is turned off by setting the PWC_PWRC1 . VHRCSD bit; and by disconnecting power to the UPLL and MPLL after both are turned off by setting the PWC_PWRC1 . VPLLSD bit to disconnect power to the UPLL and MPLL to further reduce power consumption.

For more information, please refer to [Clock Controller (CMU)]

5.5.3 Function clock stop

The peripheral module of the chip is equipped with a functional clock stop function. By placing the register corresponding to the position bit, the module that does not need to be used can be stopped from running, and the clock of the corresponding module is also stopped from supplying, reducing power consumption. In the module stop state, the internal registers of the module will maintain the state before stopping.

5.5.4 Turn off RAM that is not in use

Each RAM module in the chip is configured with a functional clock stop bit and a power-down control bit, which reduces power consumption by setting the module stop bit to stop clocking RAMs that do not need to be used. The power-down control bit of the module can be set to allow the corresponding RAM module to be powered down, thus reducing power consumption. Table 5-8 shows the correspondence between the RAM module and the power-down control bits. The corresponding RAM can be powered down by setting the corresponding RAMPDCn (n=0-8), PW_PWRC0.RETRAMSD bits in the PWC_RAMPC0 register.

Table 5-8 RAM Module and RAM Power Down Control

RAM module	Des crip tion	Bits
SRAM1	0x2000_0000~0x2000_FFFF RAM for address space	PWC_RAMPC0.RAMPDC0
SRAM2	0x2001_0000~0x2001_FFFF RAM for address space	PWC_RAMPC0.RAMPDC1
SRAM3	0x2002_0000~0x2002_6FFF RAM for address space	PWC_RAMPC0.RAMPDC2
SRAMH	0x1FFF_8000~0x1FFF_FFFF RAM for address space	PWC_RAMPC0.RAMPDC3
USBFS	RAM for USBFS FIFO	PWC_RAMPC0.RAMPDC4
SDIO0RAM	SDIO0 with RAM	PWC_RAMPC0.RAMPDC5
SDIO1RAM	SDIO1 with RAM	PWC_RAMPC0.RAMPDC6
CANRAM	RAM for CAN	PWC_RAMPC0.RAMPDC7
CACHERAM	RAM for Cache	PWC_RAMPC0.RAMPDC8
Ret-SRAM	0x200F_0000~0x200F_0FFF RAM for address space	PWC_PWRC0.RETRAMSD

5.6 Register protection function

The register protection function is used to invalidate the write operation of a register to protect the register from accidental rewriting. Table 5-9 shows the list of register protection bits and protected registers.

Table 5-9 Register Protection List

Protection register bits	Protected registers
PWC_FPRC.fprcb0	CMU_XTALCFGGR, CMU_XTALSTBCR, CMU_XTALCR, CMU_XTALSTDGR, CMU_XTALSTDGR, CMU_HRCTRM, CMU_HRCCR, CMU_MRCTR, CMU_MRCCR, CMU_PLLCFGGR, CMU_PLLCR, CMU_UPLLCFGGR, CMU_UPLLCR, CMU_OSCSTBSR, CMU_CKSWR, CMU_SCFGR, CMU_UFSCKCFGGR, CMU_TPIUCKCFGGR, CMU_MCO1CFGGR, CMU_MCO2CFGGR, CMU_XTAL32CR, CMU_XTALC32CFGGR, CMU_XTALC32CFGGR, CMU_XTAL32NFR. CMU_LRCCR, CMU_LRCTR, PWC_XTAL32CS
PWC_FPRC.FPRCB1	pwc_pwrc0, pwc_pwrc1, pwc_pwrc2, pwc_pwrc3, pwc_pdwke0, pwc_pdwke1, pwc_pdwke2, pwc_pdwkes, pwc_pdwkf0, pwc_pdwkf1, pwc_pwmr, cmu_pericksel. CMU_I2SCKSEL. pwc_mdswcr, pwc_stpmcr, pwc_rampc0, pwc_ramopm, rmu_rstf0
PWC_FPRC.FPRCB3	pwc_pvdcr0, pwc_pvdcr1, pwc_pvdocr, pwc_pvdicr, pwc_pvddsr

Protection register bits	Protected registers
PWC_FCG0PRC.B0	PWC_FCG0

5.7 Register Description

The list of registers is shown in

Table 5-10. BASE

ADDR:0x4005_4400

Table 5-10 Register List

Register Name	Sym bols	Offset Address	Bit width	Reset value
Power mode control register 0	PWC_PWRC0	0x00	8	0x00
Power mode control register 1	PWC_PWRC1	0x01	8	0x00
Power mode control register 2	PWC_PWRC2	0x02	8	0xFF
Power mode control register 3	PWC_PWRC3	0x03	8	0x07
Power-down wake-up enable register 0	PWC_PDWKE0	0x04	8	0x00
Power-down wake-up enable register 1	PWC_PDWKE1	0x05	8	0x00
Power-down wake-up enable register 2	PWC_PDWKE2	0x06	8	0x00
Power-down wake-up event edge selection register	PWC_PDWKES	0x07	8	0x00
Power-down wake-up flag register 0	PWC_PDWKF0	0x08	8	0x00
Power-down wake-up flag register 1	PWC_PDWKF1	0x09	8	0x00
Power monitoring register	PWC_PWCMR	0x0A	8	0x00
Mode switching control register	PWC_MDSWCR	0x0F	8	0x00
PVD control register 0	PWC_PVDCR0	0x12	8	0x00
PVD control register 1	PWC_PVDCR1	0x13	8	0x00
PVD filter control register	PWC_PVDFCR	0x14	8	0x11
PVD level control register	PWC_PVDLCR	0x15	8	0x00
XTAL32 Current Control Register	PWC_XTAL32CS	0x2B	8	0x02

BASE ADDR:0x4005_4000

Register Name	Sym bols	Offset Address	Bit width	Reset value
STOP mode wake-up control register	PWC_STPMCR	0x0C	16	0x4000
RAM power consumption control register 0	PWC_RAMPC0	0x14	32	0x0000_0000
RAM operating condition register	PWC_RAMOPM	0x18	16	0x8043
PVD interrupt control register	PWC_PVDICR	0xE0	8	0x00
PVD detection status register	PWC_PVDDSR	0xE1	8	0x11
Function protection control register	PWC_FPRC	0x3FE	16	0x0000

BASE ADDR:0x4004_C400

Register Name	Sym bols	Offset Address	Bit width	Reset value
Wake-up timer control register	PWC_WKTCR	0x00	16	0x0000

BASE ADDR:0x4004_8000

Register Name	Sym bols	Offset Address	Bit width	Reset value
Function Clock Control 0	PWC_FCG0	0x00	32	0xFFFF_FAEE
Function Clock Control 1	PWC_FCG1	0x04	32	0xFFFF_FFFF

Function Clock Control 2	PWC_FCG2	0x08	32	0xFFFF_FFFF
Function clock control3	PWC_FCG3	0x0C	32	0xFFFF_FFFF
PWC_FCG0 protection control	PWC_FCG0PC	0x10	32	0x0000_0000

5.7.1 Power mode control register 0 (PWC_PWRC0)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
PWDN	-	IORTN[1:0]		RETRAMSD	VVDRSD		PDMDS[1:0]
<hr/>							
position	Marker	Place Name	Function				Reading and writing
b7	PWDN	Power down mode control bit	0: Power-down mode is invalid 1: Power down mode enable				R/W
b6	Reserved	-	Read <code>0</code> , write <code>0</code> when writing				R/W
b5-b4	IORTN[1:0]	IO hold control in power-down mode	00: IO hold state in power-down mode, hardware release 01: IO hold state after power-down wake-up 10: IO is high resistance in power-down mode and after power-down wake-up 11: IORTN[1:0] after power-down wake-up	00b, release IO hold state	IO hold state after power-down wake-up		R/W
b3	RETRAMSD	Maintain RAM power-down control	0: Ret-SRAM is not powered down 1: Ret-SRAM power down				R/W
b2	VVDRSD	VDR LDO control	0: Use RLDO 1: Close RLDO When RLDO is turned off and the chip goes into power-down mode the RTC/Wakeup Timer/Hold RAM power is turned off and therefore cannot be used.				R/W
b1-b0	PDMDS[1:0]	Power down mode selection control	PDMDS[1:0] 00: Power down mode 1 01: Power down mode 2 10: Power down mode 3 11: Power down mode 4				R/W

5.7.2 Power mode control register 1 (PWC_PWRC1)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
STPDAS[1:0]		-	-	-	-	VHRCSD	VPLLSD

position	Marker	Place Name	Function	Reading and writing
b7-b6	STPDAS[1:0]	STOP mode LDO driver selection	00: Super high speed mode, high speed mode into the STOP mode when the drive capacity set 11: Drive capacity set when entering STOP mode in ultra-low speed mode 00\01: Forbidden to set.	R/W
b5-b2	Reserved	-	Read "0000**", write "0000**" when writing	R/W
b1	VHRCSD	HRC power off	0: HRC power enable 1: HRC power off When HRC is not in use, set VHRCSD and turn off the power for HRC to further reduce power consumption; after VHRCSD is cleared, you need to wait 25uS before starting again HRC Module	R/W
b0	VPLLSD	PLL power off	0: PLL power enable 1: PLL power off After UPLL and MPLL are both off and wait 50us, after setting VPLLSD Turn off the power supply for PLL to further reduce power consumption. after VPLLSD is cleared, you need to wait 25uS before starting the PLL module.	R/W

5.7.3 Power mode control register 2 (PWC_PWRC2)

Reset value:0xFF

b7	b6	b5	b4	b3	b2	b1	b0
-	-	DVS[1:0]			DDAS[3:0]		

position	Marker	Place Name	Function	Reading and writing
b7	Reserved	-	Read "1", write "1" when writing	R/W
b6	Reserved	-	Read "1", write "1" when writing	R/W
b5-b4	DVS[1:0]	Voltage selection in action mode	00: Select ultra-high speed action mode voltage 01 : Setting the ban 10: Select ultra-low speed action voltage 11: Select high-speed action mode voltage	R/W
b3-b0	DDAS[3:0]	Power driver selection	1111: Super high speed operation mode, high speed operation mode drive capability selection 1110: Set Prohibition 1001: Set Prohibition 1000: Ultra-low speed operation mode drive capability selection Other: Setting disable	R/W

5.7.4 Power mode control register 3 (PWC_PWRC3)

Reset value: 0x07

b7	b6	b5	b4	b3	b2	b1	b0
		-		PDTs	-	-	

position	Marker	Place Name	Function	Reading and writing
b7-b3	Reserved	-	Reserved bit, write "000000"when writing.	R/W
b2	PDTs	Power-down wake-up time control bit	0: When the total capacitance of VCAP_1/VCAP_2 is two 0.1uF or one At 0.22uF 1: VCAP_1/VCAP_2 total capacitance is two 0.047uF or one At 0.1uF	R/W
b1	Reserved	-	Read "1", write "1" when writing	R/W
b0	Reserved	-	Read "1", write "1" when writing	R/W

5.7.5 Power-down wake-up enable register 0 (PWC_PDWKE0)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
WKE13	WKE12	WKE11	WKE10	WKE03	WKE02	WKE01	WKE00

position	Marker	Place Name	Function	Reading and writing
b7	WKE13	WKUP1_3 wake-up event enable	0: WKUP1_3 wake-up event is invalid 1: WKUP1_3 wake-up event enable	R/W
b6	WKE12	WKUP1_2 wake-up event enable	0: WKUP1_2 wake-up event is invalid 1: WKUP1_2 wake-up event enable	R/W
b5	WKE11	WKUP1_1 wake-up event enable	0: WKUP1_1 wake-up event is invalid 1: WKUP1_1 wake-up event enable	R/W
b4	WKE10	WKUP1_0 wake-up event enable	0: WKUP1_0 wake-up event is invalid 1: WKUP1_0 wake-up event enable	R/W
b3	WKE03	WKUP0_3 wake-up event enable	0: WKUP0_3 wake-up event is invalid 1: WKUP0_3 wake-up event enable	R/W
b2	WKE02	WKUP0_2 wake-up event enable	0: WKUP0_2 wake-up event is invalid 1: WKUP0_2 wake-up event enable	R/W
b1	WKE01	WKUP0_1 wake-up event enable	0: WKUP0_1 wake-up event is invalid 1: WKUP0_1 wake-up event enable	R/W
b0	WKE00	WKUP0_0 wake-up event enable	0: WKUP0_0 wake-up event is invalid 1: WKUP00 wake-up event enable	R/W

5.7.6 Power-down wake-up enable register 1 (PWC_PDWKE1)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
WKE33	WKE32	WKE31	WKE30	WKE23	WKE22	WKE21	WKE20

position	Marker	Place Name	Function	Reading and writing
b7	WKE33	WKUP3_3 wake-up event enable	0: WKUP3_3 wake-up event is invalid 1: WKUP3_3 wake-up event enable	R/W
b6	WKE32	WKUP3_2 wake-up event enable	0: WKUP3_2 wake-up event is invalid 1: WKUP3_2 wake-up event enable	R/W
b5	WKE31	WKUP3_1 wake-up event enable	0: WKUP3_1 wake-up event is invalid 1: WKUP3_1 wake-up event enable	R/W
b4	WKE30	WKUP3_0 wake-up event enable	0: WKUP3_0 wake-up event is invalid 1: WKUP3_0 wake-up event enable	R/W
b3	WKE23	WKUP2_3 wake-up event enable	0: WKUP2_3 wake-up event is invalid 1: WKUP2_3 wake-up event enable	R/W
b2	WKE22	WKUP2_2 wake-up event enable	0: WKUP2_2 wake-up event is invalid 1: WKUP2_2 wake-up event enable	R/W
b1	WKE21	WKUP2_1 wake-up event enable	0: WKUP2_1 wake-up event is invalid 1: WKUP2_1 wake-up event enable	R/W
b0	WKE20	WKUP2_0 wake-up event enable	0: WKUP2_0 wake-up event is invalid 1: WKUP2_0 wake-up event enable	R/W

5.7.7 Power-down wake-up enable register 2(PWC_PDWKE2)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
WKTMWKE	-	RTCALMWKE	RTCPRDWKE	-	NMIWKE	PVD2WKE	PVD1WKE

position	Marker	Place Name	Function	Reading and writing
b7	WKTMWKE	WKT M wakeup event enable	0: WKT M wake-up event is invalid 1: WKT M wake-up event enable	R/W
b6	Reserved	-	Read "0", write "0" when writing	R/W
b5	RTCALMWKE	RTC alarm wake-up event enable	0: RTC alarm wake-up event is invalid 1: RTC alarm clock wakeup event enable	R/W
b4	RTCPRDWKE	RTC cycle wake-up event enable	0: RTC cycle wake-up event is invalid 1: RTC cycle wake-up event enable	R/W
b3	Reserved	-	Reserved bit, write "0" when writing.	R/W
b2	NMIWKE	NMI wake-up event enable	0: NMI wake-up event is invalid 1: NMI wake-up event enable	R/W
b1	PVD2WKE	PVD2 wake-up event enable	0: PVD2 wake-up event is invalid 1: PVD2 wake-up event enable	R/W
b0	PVD1WKE	PVD1 wake-up event enable	0: PVD1 wake-up event is invalid 1: PVD1 wake-up event enable	R/W

5.7.8 Power-down wake-up event edge selection register (PWC_PDWKES)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	NMIEGS	VD2EGS	VD1EGS	WK3EGS	WK2EGS	WK1EGS	WK0EGS

position	Marker	Place Name	Function	Reading and writing
b7	Reserved	-	Read "0", write "0" when writing	R/W
b6	NMIEGS	NMI wake-up event edge selection	0: falling edge 1: Rising edge	R/W
b5	VD2EGS	VD2 edge selection	0: VCC<VPVD2 1: VCC> VPVD2	R/W
b4	VD1EGS	VD1 edge selection	0: VCC<VPVD2 1: VCC> VPVD2	R/W
b3	WK3EGS	PTWK3 Edge Selection	0: falling edge 1: Rising edge	R/W
b2	WK2EGS	PTWK2 Edge Selection	0: falling edge 1: Rising edge	R/W
b1	WK1EGS	PTWK1 Edge Selection	0: falling edge 1: Rising edge	R/W
b0	WK0EGS	PTWK0 edge selection	0: falling edge 1: Rising edge	R/W

5.7.9 Power-down wake-up flag register 0 (PWC_PDWKF0)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	NMIWKF	PVD2WKF	PVD1WKF	PTWK3F	PTWK2F	PTWK1F	PTWK0F

position	Marker	Place Name	Function	Reading and writing
b7	Reserved	-	Read "1", write "1" when writing	R/W
b6	NMIWKF	NMI wake-up flag bit	0: No NMI pin wake-up event occurred 1: NMI pin wake-up event occurs After power-down wake-up, you need to write zero to clear the home position.	R/W
b5	PVD2WKF	PVD2 wake-up flag bit	0: No PVD2 wake-up event occurred 1: PVD2 wake-up event occurs After power-down wake-up, you need to write zero to clear the home position.	R/W
b4	PVD1WKF	PVD1 wake-up flag bit	0: No PVD1 wake-up event occurred 1: PVD1 wake-up event occurs After power-down wake-up, you need to write zero to clear the home position.	R/W
b3	PTWK3F	PTWK3 Wake-up flag bit	0: No PTWK3 wake-up event occurred 1: PTWK3 wake-up event occurs After power-down wake-up, you need to write zero to clear the home position.	R/W
b2	PTWK2F	PTWK2 wake-up flag bit	0: No PTWK2 wake-up event occurred 1: PTWK2 wake-up event occurs After power-down wake-up, you need to write zero to clear the home position.	R/W
b1	PTWK1F	PTWK1 wake-up flag bit	0: No PTWK1 wake-up event occurred 1: PTWK1 wake-up event occurs After power-down wake-up, you need to write zero to clear the home position.	R/W
b0	PTWK0F	PTWK0 wake-up flag bit	0: No PTWK0 wake-up event occurred 1: PTWK0 wake-up event occurs After power-down wake-up, you need to write zero to clear the home position.	R/W

5.7.10 Power-down wake-up flag register 1 (PWC_PDWKF1)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
WKTMWKF	-	RTCALMWKF	RTCPRDWKF	-	-	-	

position	Marker	Place Name	Function	Reading and writing
b7	WKTMWKF	WKTM wake-up flag bit	0: No WKTM wake-up event occurred 1: WKTM wake-up event occurs After power-down wake-up, you need to write zero to clear the home position.	R/W
b6	Reserved	-	Reserved bit, write "0" when writing.	R/W
b5	RTCALMWKF	RTC alarm wake-up flag bit	0: No RTC alarm wake-up event occurred 1: RTC alarm wake-up event occurs After power-down wake-up, you need to write zero to clear the home position.	R/W
b4	RTCPRDWKF	RTC cycle wake-up flag bit	0: No RTC cycle wake-up event occurred 1: RTC cycle wake-up event occurs After power-down wake-up, you need to write zero to clear the home position.	R/W
b3	Reserved	-	Reserved bit, write "0" when writing.	R/W
b2	Reserved	-	Reserved bit, write "0" when writing.	R/W
b1	Reserved	-	Reserved bit, write "0" when writing.	R/W
b0	Reserved	-	Reserved bit, write "0" when writing.	R/W

5.7.11 Power Supply Monitoring Control Register (PWC_PWCMR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
ADBUFE	-	-	-	-	-	-	-

position	Marker	Place Name	Function	Reading and writing
b7	ADBUFE	ADBUF enable	When using AD to measure the internal voltage of the chip, you need to set this bit to 1: Invalid 0: Effective	R/W
b6	Reserved	-	Read "0", write "0" when writing	R/W
b5	Reserved	-	Read "0", write "0" when writing	R/W
b4	Reserved	-	Read "0", write "0" when writing	R/W
b2	Reserved	-	Read "0", write "0" when writing	R/W
b1	Reserved	-	Read "0", write "0" when writing	R/W
b0	Reserved	-	Read "0", write "0" when writing	R/W

5.7.12 Mode Switching Control Register (PWC_MDSWCR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
MDSWC [7:0]							

position	Marker	Place Name	Function	Reading and writing
b7-b0	MDSWC [7:0]	Mode Switching Enable	When performing action mode switching, after setting PWRC2, MDSWC[7:0] needs to be set to 0x10 before it can take effect. Only 0x10 can be written, other values are forbidden to be written and read out as 0x00.	R/W

5.7.13 Function Clock Control 0 (PWC_FCG0)

Reset value : 0xFFFF_FAEE

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
KEY	-	-	-	DCU4	DCU3	DCU2	DCU1	CRC	TRNG	HASH	AES	-	-	AOS	FCM
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DMA2	DMA1	-	-	-	SRAMRET	-	SRAM3	-	-	-	SRAM12	-	-	-	SRAMH

position	Marker	Place Name	Function	Reading and writing
b31	KEY	KEYSCAN function control	0:Keyboard scan control module KEYSCAN function enable 1: Keypad scan control module KEYSCAN function is invalid	R/W
b30	Reserved	-	Read "1", write "1" when writing	R/W
b29	Reserved	-	Read "1", write "1" when writing	R/W
b28	Reserved	-	Read "1", write "1" when writing	R/W
b27	DCU4	DCU4 Function Control	0: Digital Computing Unit DCU3 function enable 1: Digital Computing Unit DCU3 function is invalid	R/W
b26	DCU3	DCU3 Function Control	0: Digital Computing Unit DCU2 function enable 1: Digital Computing Unit DCU2 function is invalid	R/W
b25	DCU2	DCU2 Function Control	0: Digital Computing Unit DCU1 function enable 1: Digital Computing Unit DCU1 function is invalid	R/W
b24	DCU1	DCU1 function control	0: Digital Computing Unit DCU0 function is enabled 1: Invalid function of Digital Computing Unit DCU0	R/W
b23	CRC	CRC function control	0: CRC function is enabled 1: CRC function is invalid	R/W
b22	TRNG	TRNG Function Control	0: True Random Generator TRNG function in the CPM of the encryption coprocessing module is enabled 1: The TRNG function of the True Random Generator in the CPM encryption coprocessing module is invalid.	R/W
b21	HASH	HASH Function Control	0: Secure hash algorithm module HASH function in the cryptographic coprocessing module CPM is enabled 1: The secure hash algorithm module function in the cryptographic coprocessing module CPM is invalid	R/W
b20	AES	AES function control	0: The AES function of the encryption algorithm processor in the CPM is enabled. 1: The AES function of the encryption and decryption algorithm processor in the CPM is invalid.	R/W
b19	Reserved	-	Read "1", write "1" when writing	R/W
b18	Reserved	-	Read "1", write "1" when writing	R/W
b17	AOS	Peripheral circuit trigger function control	0: Peripheral circuit trigger function enable 1: Peripheral circuit trigger function is invalid	R/W
b16	FCM	FCM Function Control	0: Clock frequency measurement module FCM function in the clock controller CMU is enabled 1: The clock frequency measurement module FCM function in the clock controller CMU is invalid	R/W
b15	DMA2	DMA2 function control	0: DMA controller DMA2 function enable 1: DMA controller DMA2 function is invalid	R/W
b14	DMA1	DMA1 function control	0: DMA controller DMA1 function is enabled 1: DMA controller DMA1 function is invalid	R/W

b13	Reserved	-	Read "1", write "1" when writing	R/W
b12	Reserved	-	Read "1", write "1" when writing	R/W
b11	Reserved	-	Read "1", write "1" when writing	R/W
b10	SRAMRET	Ret_SRAM Function Control	O: Ret_SRAM function in the built-in SRAM is enabled 1: The Ret_SRAM function in the built-in SRAM is invalid	R/W

b9	Reserved	-	Read "1", write "1" when writing	R/W
b8	SRAM3	ECCRAM Function Control	0: SRAM3 function in the built-in SRAM is enabled 1: The SRAM3 function in the built-in SRAM is invalid	R/W
b7	Reserved	-	Read "1", write "1" when writing	R/W
b6	Reserved	-	Read "1", write "1" when writing	R/W
b5	Reserved	-	Read "1", write "1" when writing	R/W
b4	SRAM12	SRAM1/SRAM2 function Control	0: SRAM1 and SRAM2 functions in the built-in SRAM are enabled 1: The SRAM1 and SRAM2 functions in the built-in SRAM are invalid	R/W
b3	Reserved	-	Read "1", write "1" when writing	R/W
b2	Reserved	-	Read "1", write "1" when writing	R/W
b1	Reserved	-	Read "1", write "1" when writing	R/W
b0	SRAMH	RAMHS function control	0: SRAMH function in the built-in SRAM is enabled 1: The SRAMH function in the built-in SRAM is invalid	R/W

5.7.14 Function Clock Control 1(PWC_FCG1)

Reset value: 0xFFFF_FFFF

b31	b30	b29	b28	b27	b26	b25	b24
–	–	–	–	USART4	USART3	USART2	USART1
b23	b22	b21	b20	b19	b18	b17	b16
–	–	–	–	SPI4	SPI3	SPI2	SPI1
b15	b14	b13	b12	b11	b10	b9	b8
I2S4	I2S3	I2S2	I2S1	SDIOC2	SDIOC1	–	USBFS
b7	b6	b5	b4	b3	b2	b1	b0
–	I2C3	I2C2	I2C1	QSPI	–	–	CAN

position	Marker	Place Name	Function	Reading and writing
b31	Reserved	–	Read "1", write "1" when writing	R/W
b30	Reserved	–	Read "1", write "1" when writing	R/W
b29	Reserved	–	Read "1", write "1" when writing	R/W
b28	Reserved	–	Read "1", write "1" when writing	R/W
b27	USART4	USART4 function control	0:Universal synchronous asynchronous transceiver USART4 function enable	R/W
b26	USART3	USART3 function control	1: Universal synchronous asynchronous transceiver USART4 function is invalid 0: Universal synchronous asynchronous transceiver USART3 function enable 1: Universal synchronous asynchronous transceiver USART3 function is invalid	R/W
b25	USART2	USART2 function control	0: Universal synchronous asynchronous transceiver USART2 function enable 1: Universal synchronous asynchronous transceiver USART2 function is invalid	R/W
b24	USART1	USART1 function control	0: Universal synchronous asynchronous transceiver USART1 function enable 1: Universal synchronous asynchronous transceiver USART1 function is invalid	R/W
b23	Reserved	–	Read "1", write "1" when writing	R/W
b22	Reserved	–	Read "1", write "1" when writing	R/W
b21	Reserved	–	Read "1", write "1" when writing	R/W
b20	Reserved	–	Read "1", write "1" when writing	R/W
b19	SPI4	SPI4 Function Control	0: Serial Peripheral Interface SPI4 function enable 1: Serial peripheral interface SPI4 function is invalid	R/W
b18	SPI3	SPI3 Function Control	0: Serial Peripheral Interface SPI3 function enable 1: Serial peripheral interface SPI3 function is invalid	R/W
b17	SPI2	SPI2 Function Control	0: Serial Peripheral Interface SPI2 function enable 1: Serial peripheral interface SPI2 function is invalid	R/W

b16	SPI1	SPI1 Function Control	0: Serial Peripheral Interface SPI1 function R/W enable 1: Serial peripheral interface SPI1 function is invalid
b15	I2S4	I2S4 function control	0:IC built-in audio bus module I2S4 function R/W enable 1: Integrated circuit built-in audio bus module I2S4 function is invalid
b14	I2S3	I2S3 Function Control	0: Integrated circuit built-in audio bus module I2S3 function enable R/W 1: Integrated circuit built-in audio bus module I2S3 function is invalid
b13	I2S2	I2S2 function control	0: Integrated circuit built-in audio bus module I2S2 function enable R/W 1: Integrated circuit built-in audio bus module I2S2 function is invalid
b12	I2S1	I2S1 function control	0: Integrated circuit built-in audio bus module I2S1 function enable R/W 1: Integrated circuit built-in audio bus module I2S1 function is invalid
b11	SDIOC2	SDIOC2 Function Control	0:SDIO controller SDIOC2 function enable R/W

			1: SDIO controller SDIOC2 function is invalid	
b10	SDIOC1	SDIOC1 Function Control	0: SDIO controller SDIOC1 function enable 1: SDIO controller SDIOC1 function is invalid	R/W
b9	Reserved	-	Read "1", write "1" when writing	R/W
b8	USBFS	USBFS function control	0: USB2.0 full speed module USBFS function enable 1: USB2.0 full speed module USBFS function is invalid	R/W
b7	Reserved	-	Read "1", write "1" when writing	R/W
b6	I2C3	I2C3 Function Control	0: Integrated circuit bus I2C3 function enable 1: Integrated circuit bus I2C3 function is invalid	R/W
b5	I2C2	I2C2 Function Control	0: Integrated circuit bus I2C2 function enable 1: Integrated circuit bus I2C2 function is invalid	R/W
b4	I2C1	I2C1 Function Control	0: Integrated circuit bus I2C1 function enable 1: Integrated circuit bus I2C1 function is invalid	R/W
b3	QSPI	QSPI Function Control	0: Four-wire serial peripheral interface QSPI function enable 1: The QSPI function of the four-wire serial peripheral interface is disabled	R/W
b2	Reserved	-	Read "1", write "1" when writing	R/W
b1	Reserved	-	Read "1", write "1" when writing	R/W
b0	CAN	CAN function control	0: Controller LAN CAN function is enabled 1: Controller LAN CAN function is invalid	R/W



5.7.15 Function Clock Control 2 (PWC_FCG2)

Reset value: 0xFFFF_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	TIME R6_3	TIME R6_2	TIME R6_1
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EMB	-	-	-	-	TIME R4_3	TIME R4_2	TIME R4_1	TIME RA_6	TIME RA_5	TIME RA_4	TIME RA_3	TIME RA_2	TIME RA_1	TIME R0_2	TIME R0_1

position	Marker	Place Name	Function	Reading and writing
b31	Reserved	-	Read "1", write "1" when writing	R/W
b30	Reserved	-	Read "1", write "1" when writing	R/W
b29	Reserved	-	Read "1", write "1" when writing	R/W
b28	Reserved	-	Read "1", write "1" when writing	R/W
b27	Reserved	-	Read "1", write "1" when writing	R/W
b26	Reserved	-	Read "1", write "1" when writing	R/W
b25	Reserved	-	Read "1", write "1" when writing	R/W
b24	Reserved	-	Read "1", write "1" when writing	R/W
b23	Reserved	-	Read "1", write "1" when writing	R/W
b22	Reserved	-	Read "1", write "1" when writing	R/W
b21	Reserved	-	Read "1", write "1" when writing	R/W
b20	Reserved	-	Read "1", write "1" when writing	R/W
b19	Reserved	-	Read "1", write "1" when writing	R/W
b18	TIMER6_3	TIMER6_3 Function Control	0:TIMER6_3 function enable 1:TIMER6_3 function is invalid	R/W
b17	TIMER6_2	TIMER6_2 Function Control	0:TIMER6_2 function enable 1:TIMER6_2 function is invalid	R/W
b16	TIMER6_1	TIMER6_1 function control	0:TIMER6_1 function enable 1:TIMER6_1 function is invalid	R/W
b15	EMB	EMB Function Control	0: Emergency brake module EMB function is enabled 1: Emergency brake module EMB function is invalid	R/W
b14	Reserved	-	Read "1", write "1" when writing	R/W
b13	Reserved	-	Read "1", write "1" when	R/W

writing				
b12	Reserved	-	Read "1", write "1" when writing	R/W
b11	Reserved	-	Read "1", write "1" when writing	R/W
b10	TIMER4_3	TIMER4_3 function control	0:TIMER4_3 function enable 1:TIMER4_3 function is invalid	R/W
b9	TIMER4_2	TIMER4_2 function control	0:TIMER4_2 function enable 1:TIMER4_2 function is invalid	R/W
b8	TIMER4_1	TIMER4_1 function control	0:TIMER4_1 function enable 1:TIMER4_1 function is invalid	R/W
b7	TIMERA_6	TIMERA_6 Function Control	0:TIMERA_6 function enable 1:TIMERA_6 function is invalid	R/W
b6	TIMERA_5	TIMERA_5 Function Control	0:TIMERA_5 function enable 1:TIMERA_5 function is invalid	R/W

b5	TIMERA_4	TIMERA_4 Function Control	0:TIMERA_4 function enable 1:TIMERA_4 function is invalid	R/W
b4	TIMERA_3	TIMERA_3 Function Control	0:TIMERA_3 function is enabled 1:TIMERA_3 function is invalid	R/W
b3	TIMERA_2	TIMERA_2 Function Control	0:TIMERA_2 function is enabled 1:TIMERA_2 function is invalid	R/W
b2	TIMERA_1	TIMERA_1 function control	0:TIMERA_1 function is enabled 1:TIMERA_1 function is invalid	R/W
b1	TIMER0_2	TIMER0_2 function control	0:TIMER0_2 function enable 1:TIMER0_2 function is invalid	R/W
b0	TIMER0_1	TIMER0_1 function control	0:TIMER0_1 function enable 1:TIMER0_1 function is invalid	R/W

5.7.16 Function Clock Control 3(PWC_FCG3)

Reset value: 0xFFFF_FFFF

b31	b30	b29	b28	b27	b26	b25	b24
-	-	-	-	-	-	-	-
b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8
-	-	-	OTS	-	-	-	CMP
b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	ADC2	ADC1

position	Marker	Place Name	Function	Reading and writing
b31	Reserved	-	Read "1", write "1" when writing	R/W
b30	Reserved	-	Read "1", write "1" when writing	R/W
b29	Reserved	-	Read "1", write "1" when writing	R/W
b28	Reserved	-	Read "1", write "1" when writing	R/W
b27	Reserved	-	Read "1", write "1" when writing	R/W
b26	Reserved	-	Read "1", write "1" when writing	R/W
b25	Reserved	-	Read "1", write "1" when writing	R/W
b24	Reserved	-	Read "1", write "1" when writing	R/W
b23	Reserved	-	Read "1", write "1" when writing	R/W
b22	Reserved	-	Read "1", write "1" when writing	R/W
b21	Reserved	-	Read "1", write "1" when writing	R/W
b20	Reserved	-	Read "1", write "1" when writing	R/W
b19	Reserved	-	Read "1", write "1" when writing	R/W
b18	Reserved	-	Read "1", write "1" when writing	R/W
b17	Reserved	-	Read "1", write "1" when writing	R/W
b16	Reserved	-	Read "1", write "1" when writing	R/W
b15	Reserved	-	Read "1", write "1" when writing	R/W
b14	Reserved	-	Read "1", write "1" when writing	R/W
b13	Reserved	-	Read "1", write "1" when writing	R/W
b12	OTS	OTS Function Control	0: Temperature sensor OTS function is valid 1: Temperature sensor OTS function is invalid	R/W
b11	Reserved	-	Read "1", write "1" when writing	R/W
b10	Reserved	-	Read "1", write "1" when writing	R/W
b9	Reserved	-	Read "1", write "1" when writing	R/W
b8	CMP	CMP Function Control	0: Voltage comparator CMP function enable 1: The voltage comparator CMP function is invalid	R/W
b7	Reserved	-	Read "1", write "1" when writing	R/W
b6	Reserved	-	Read "1", write "1" when writing	R/W
b5	Reserved	-	Read "1", write "1" when writing	R/W
b4	Reserved	-	Read "1", write "1" when writing	R/W
b3	Reserved	-	Read "1", write "1" when writing	R/W
b2	Reserved	-	Read "1", write "1" when writing	R/W

b1	ADC2	ADC2 Function Control	0: ADC2 function of analog-to-digital conversion module is enabled 1: Analog-to-digital conversion module ADC2 function is invalid	R/W
b0	ADC1	ADC1 function control	0: ADC1 function of analog-to-digital conversion module is enabled 1: Analog-to-digital conversion module ADC1 function is invalid	R/W

5.7.17 PWC_FCG0 Protection Control (PWC_FCG0PC)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	
b16 FCG0PCWE[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PRT0

position	Marker	Place Name	Function	Reading and writing
b31~b16	FCG0PCWE[15:0]	PWC_FCG0PC write enable	Change the value of PRT0 bit while writing to 0xA5A5	R/W
		Can		
b15-b1	Reserved	-	Read "0", write "0" when writing	R/W
b0	PRT0	Protected position	PWC_FCG0 write enable control bit 0:PWC_FCG0 write invalid 1:PWC_FCG0 write enable	R/W

5.7.18 Function Protection Control Register (PWC_FPRC)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PWC_FPRCWE[7:0]								-	-	-	-	FPRCB3	FPRCB2	FPRCB1	FPRCB0
position	Marker	Place Name	Function	Reading and writing											
b15~b8	PWC_FPRCWE	PWC_FPRC register write enable	Write to 0xA5h while being able to update the PWC_FPRC value, otherwise it is invalid for the lower 8-bit write value. Reads out at 0x00.	R/W											
b7	Reserved	-	Reserved bit, write "0" when writing.	R/W											
b6	Reserved	-	Reserved bit, write "0" when writing.	R/W											
b5	Reserved	-	Reserved bit, write "0" when writing.	R/W											
b4	Reserved	-	Reserved bit, write "0" when writing.	R/W											
b3	FPRCB3	FPRC bit 3	Protection register bit, protection object refer to Table 5-9 O:Write protection 1:Write enable	R/W											
b2	Reserved	-	Reserved bit, write "0" when writing.	R/W											
b1	FPRCB1	FPRC bit 1	Protect register bits, protection objects refer to Table 5-9 O:Write protection 1:Write enable	R/W											

b0	FPRCB0	FPRC bit 0	Protect register bits, protection objects refer to Table 5-9 0:Write protection 1:Write enable	R/W
----	--------	------------	---	-----

5.7.19 STOP mode control register (PWC_STPMCR)

Reset value: 0x4000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
STOP	-	-	-	-	-	-	-	-	-	-	-	-	-	CKSMRC	FLNWT

position	Marker	Place Name	Function	Reading and writing
b15	STOP	STOP mode selection bit	0: STOP mode is invalid 1: STOP mode is valid	R/W
b14	Reserved	-	Read "1", write "1" when writing	R/W
b13-b2	Reserved	-	Read "1", write "0" when writing	R/W
b1	CKSMRC	Clock switch to MRC option	0: Maintain the system clock and dividing frequency before entering STOP mode 1: The system clock is switched to MRC and SCKCFGR registers are initialized when waking up in STOP mode	R/W
b0	FLNWT	FLASH stable wait control	0: Wait for FLASH to stabilize when waking up in STOP mode 1: STOP mode wakes up without waiting for FLASH to stabilize	R/W-

5.7.20 RAM power consumption control register 0(PWC_RAMPC0)

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	RAMPDC8
7	6	5	4	3	2	1	0
RAMPDC7	RAMPDC6	RAMPDC5	RAMPDC4	RAMPDC3	RAMPDC2	RAMPDC1	RAMPDC0

position	Marker	Place Name	Function	Reading and writing
b32-b23	Reserved	-	Read "0", write "0" when writing	R/W
b24	Reserved	-	Read "0", write "0" when writing	R/W
b23	Reserved	-	Read "0", write "0" when writing	R/W
b22	Reserved	-	Read "0", write "0" when writing	R/W
b21	Reserved	-	Read "0", write "0" when writing	R/W
b20	Reserved	-	Read "0", write "0" when writing	R/W
b19	Reserved	-	Read "0", write "0" when writing	R/W
b18	Reserved	-	Read "0", write "0" when writing	R/W
b17	Reserved	-	Read "0", write "0" when writing	R/W
b16	Reserved	-	Read "0", write "0" when writing	R/W
b15-b9	Reserved	-	Read "0", write "0" when writing	R/W
b8	RAMPDC8	RAM power-down control bit 8	0: RAM action for cache 1: RAM power down for cache	R/W
b7	RAMPDC7	RAM power-down control bit 7	0: can with RAM action 1: can with RAM power down	R/W
b6	RAMPDC6	RAM power-down control bit 6	0: sdioc1 with RAM action 1: sdioc1 with RAM power down	R/W
b5	RAMPDC5	RAM power-down control bit 5	0: sdioc0 with RAM action 1: sdioc0 with RAM power down	R/W
b4	RAMPDC4	RAM power-down control bit 4	0: usbfs with RAM action 1: usbfs with RAM power down	R/W
b3	RAMPDC3	RAM power-down control bit 3	0: 0x1FFF_8000~0x1FFF_FFFF Space RAM (SRAMH) Action 1: 0x1FFF_8000~0x1FFF_FFFF Space RAM (SRAMH) Power down	R/W
b2	RAMPDC2	RAM power-down control bit 2	0: 0x2002_0000~0x2002_6FFF Space RAM (SRAM3) Action 1: 0x2002_0000~0x2002_6FFF Space RAM (SRAM3) Power down	R/W

b1	RAMPDC1	RAM power-down control bit 1	0: 0x2001_0000~0x2001_FFFF Space RAM (SRAM2) Action 1: 0x2001_0000~0x2001_FFFF Space RAM (SRAM2) Power down	R/W
----	---------	---------------------------------	--	-----

b0	RAMPDC0	RAM power-down control bit 0	0: 0x2000_0000~0x2000_FFFF Space RAM (SRAM1) Action 1: 0x2000_0000~0x2000_FFFF Space RAM (SRAM1) Power down	R/W
----	---------	------------------------------	--	-----

5.7.21 RAM operation condition register (PWC_RAMOPM)

Reset value: 0x8043

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

position	Marker	Place Name	Function	Reading and writing
b15-b0	PWC_RAMOPM[15:0]	RAM action mode is selected as	The PWC_RAMOPM is set to 0x8043 when the chip is operating in SuperSpeed/High Speed operation mode. When the chip operates in ultra-low speed mode action PWC_RAMOPM set to 0x9062	R/W

5.7.22 XTAL32 controlled by current source (PWC_XTAL32CS)

Reset value: 0x02

b7	b6	b5	b4	b3	b2	b1	b0
CSDIS			-		-	-	-

position	Marker	Place Name	Function	Reading and writing
7	CSDIS	Invalid control of current sources	0: Current source valid 1: Invalid current source CSDIS can be set to reduce power consumption when XTAL32/RTC/WTKM/Ret-SRAM etc. are not needed to be used.	R/W
b6-b2	Reserved	-	Read "0", write "0" when writing	R/W
b1	Reserved	-	Read "0", write "1" when writing	R/W
b0	Reserved	-	Read "0", write "0" when writing	R/W

5.7.23 Wake-up timer control register (PWC_WKTCR)

Reset value: 0x0000

b15	b14-b13	b12	b11	-	b0
WKTCE	WKCKS[1: 0]	WKOVF		WKTMCMP[1: 0]	

position	Marker	Place Name	Function	Reading and writing
b15	WKTCE	WKTME	0: WKTME stop 1: WKTME Counting	R/W
b14-b13	WKCKS[1:0]	WKTME clock selection	00: 64Hz clock 01: XTAL32 10: LRC 11: Reserved	R/W
b12	WKOVF	Timer comparison result flag	0: Counter does not match WKTMCMP value 1: The counter is consistent with the WKTMCMP value	R/W
b11-b0	WKTMCMP[11:0]	WKTME Comparison Bits	WKTME counter comparison values	R/W

5.7.24 PVD control register 0 (PWC_PVDCR0)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	PVD2EN	PVD1EN	-	-	-	-	EXVCCINEN

position	Marker	Place Name	Function	Reading and writing
b7	Reserved	-	Read "0", write "0" when writing	R/W
b6	PVD2EN	Voltage detection 2 allowed	0: Voltage detection 2 circuit is invalid 1: Voltage detection 2 circuit valid	R/W
b5	PVD1EN	Voltage detection 1 allowed	0: Voltage detection 1 circuit is invalid 1: Voltage detection 1 circuit is valid	R/W
b4	Reserved	-	Read "0", write "0" when writing	R/W
b3~b2	Reserved	-	Read "0", write "0" when writing	R/W
b1	Reserved	-	Read "0", write "0" when writing	R/W
b0	EXVCCINEN	External VCC voltage input enable	0: External VCC voltage input is invalid 1: External VCC voltage input is valid	R/W

5.7.25 PVD control register 1 (PWC_PVDCR1)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	PVD2CMPOE	PVD2IRS	PVD2IRE	-	PVD1CMPOE	PVD1IRS	PVD1IRE

position	Marker	Place Name	Function	Reading and writing
b7	Reserved	-	Read "0", write "0" when writing	R/W
b6	PVD2CMPOE	PVD2 comparison result output enable	0: Disable the output of comparator 2 comparison result 1: Allows the comparison result of comparator 2 to be output	R/W
b5	PVD2IRS	PVD2 interrupt reset selection	0: PVD2 interrupt generated when VCC or external input voltage drops past VPVD2 Note: When the PVD1IRS bit is "1" or the PVD2IRS bit is "1", the power-down mode cannot be entered, and to enter the power-down mode, the PVD1IRS bit must be set to "0" and the PVD2IRS bit to "0". PVD1IRS position "0" and PVD2IRS position "0"	R/W
b4	PVD2IRE	PVD2 interrupt reset enable	0: Forbidden 1: Allowed Note: Please write "1" to the PVD2IRE bit when the PVD2EN bit is "1" and the PVD2CMPOE bit is "1".	R/W
b3	Reserved	-	Read "0", write "0" when writing	R/W
b2	PVD1CMPOE	PVD1 comparison result output enable	0: Disable the output of comparator 1 comparison result 1: Allows the output of the comparison result of comparator 1	R/W
b1	PVD1IRS	PVD1 interrupt reset selection	0: VCC generates PVD1 interrupt when falling through VPVD1 1: VCC is falling through VPVD1 to generate PVD1 reset Note 1: When the PVD1IRS bit is "1" or PVD2IRS bit is "1", it cannot enter the power-down mode, to enter the power-down mode, you must set the PVD1IRS position "0" and set the PVD2IRS position "0". " and set PVD2IRS to "0"	R/W
b0	PVD1IRE	PVD1 interrupt reset enable	0: Forbidden 1: Allowed Note: Please write "1" to the PVD1IRE bit when the PVD1EN bit is "1" and the PVD1CMPOE bit is "1".	R/W

5.7.26 PVD Filter Control Register (PWC_PVDFCR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	PVD2NFCKS[1:0]		PVD2NFDIS	-	PVD1NFCKS[1:0]		PVD1NFDIS

position	Marker	Place Name	Function	Reading and writing
b7	Reserved	-	Read "0", write "0" when writing	R/W
b6~b5	PVD2NFCKS	PVD2 digital filtering sampling clock selection	00: 0.25 LRC period 01: 0.5 LRC period 10: LRC's 1-way frequency 11: LRC's 2-way frequency Note: This bit can only be overwritten when the PVD2NFDIS bit is "1".	R/W
b4	PVD2NFDIS	PVD2 digital filter shield	0: Digital filter valid 1: Digital filter is invalid	R/W
b3	Reserved	-	Read "0", write "0" when writing	R/W
b2~b1	PVD1NFCKS	PVD1 digital filtering sampling clock selection	00: 0.25 LRC period 01: 0.5 LRC period 10: LRC's 1-way frequency 11: LRC's 2-way frequency Note: This bit can only be overwritten when the PVD1NFDIS bit is "1".	R/W
b0	PVD1NFDIS	PVD1 digital filter shield	0: Digital filter valid 1: Digital filter is invalid	R/W

5.7.27 PVD Level Control Register (PWC_PVDLCR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-		PVD2LVL[2:0]		-		PVD1LVL[2:0]	

position	Marker	Place Name	Function	Reading and writing
b7	Reserved	-	Read "0", write "0" when writing	R/W
b6~b4	PVD2LVL	PVD2 Threshold Voltage Selection	000: 2.1V 001: 2.3V 010: 2.5V 011: 2.6V 100: 2.7V 101: 2.8V 110: 2.9V 111: 1.1V (only when PWC_PVDCR0_EXVCCINEN=1 (valid when the value is set, otherwise please do not set this value))	R/W
			Note: The thresholds noted above are the thresholds when the chip is working in high speed mode, ultra-low speed mode, and stop mode. Please refer to the electrical characteristics for the thresholds when the chip is working in ultra-high speed mode.	
b3	Reserved	-	Read "0", write "0" when writing	R/W
b2~b0	PVD1LVL	PVD1 Threshold Voltage Selection	000: 2.0V 001: 2.1V 010: 2.3V 011: 2.5V 100: 2.6V 101: 2.7V 110: 2.8V 111: 2.9V	R/W
			Note: The thresholds noted above are the thresholds when the chip is working in high speed mode, ultra-low speed mode, and stop mode. Please refer to the electrical characteristics for the thresholds when the chip is working in ultra-high speed mode.	

5.7.28 PVD Interrupt Control Register (PWC_PVDICR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-		-	PVD2NMIS	-	-	-	PVD1NMIS

position	Marker	Place Name	Function	Reading and writing
b7	Reserved	-	Read '0', write '0**' when writing	R/W
b6~b5	Reserved	-	Read '0', write '0**' when writing	R/W
b4	PVD2NMIS	PVD2 interrupt type selection	0: PVD2 interrupt as non-maskable interrupt 1: PVD2 interrupt as a maskable interrupt	R/W
b3	Reserved	-	Read '0', write '0**' when writing	R/W
b2~b1	Reserved	-	Read '0', write '0**' when writing	R/W
b0	PVD1NMIS	PVD1 interrupt type selection	0: PVD1 interrupt as non-maskable interrupt 1: PVD1 interrupt as a maskable interrupt	R/W

5.7.29 PVD Detection Status Register (PWC_PVDDSR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	-	PVD2DETFLG	PVD2MON	-	-	PVD1DETFLG	PVD1MON
position	Marker	Place Name	Function			Reading and writing	
b7~b6	Reserved	-	Read "0", write "0" when writing			R/W	
b5	PVD2DETFLG	PVD2 detection flag bit	0: PVD2 does not detect VCC passing through VPVD2 1: PVD2 detects VCC passing through VPVD2 Writing 0 to bit 4 after reading out can clear this bit. Note: This flag bit is valid when PVD2EN bit is "1" and PVD2CMPOE bit is "1".			R/W	
b4	PVD2MON	PVD2 monitoring and detection flag bit clear	0: VCC ≤ VPVD2 or external input comparison voltage ≤ PVD2 internal reference voltage 1: When PVD2 is invalid or VCC>VPVD2 or external input comparison voltage>PVD2 internal reference voltage Write 0 to this bit to clear the PVD2DETFLG bit.			R/W	
b3~b2	Reserved	-	Read "0", write "0" when writing			R/W	
b1	PVD1DETFLG	PVD1 detection flag bit	0: PVD1 does not detect VCC passing through VPVD1 1: PVD1 detects that VCC passes through VPVD1 Writing 0 to bit 0 after reading out is able to clear this bit. Note: This flag bit is valid when the PVD1EN bit is "1" and the PVD1CMPOE bit is "1".			R/W	
b0	PVD1MON	PVD1 monitoring and detection flag bit clear	0: VCC ≤ VPVD1 1: When PVD1 is invalid or VCC>VPVD1 write 0 to this bit to clear PVD1DETFLG bit.			R/W	

6 Initialization Configuration (ICG)

6.1 Introduction

After the chip reset is released, the hardware circuit will read FLASH address 0x0000_0400~0x0000_041F and load the data into the initialization configuration register. Addresses 0x0000_0408~0x0000_040F and 0x0000_0418~0x0000_041F are reserved functions, please write all 1 to ensure the normal operation of the chip. The user can modify the **Initial Config** register by programming or erasing sector 0. The register reset value is determined by the FLASH data.

The list of initial configuration register addresses is as follows:

ICG_BASE_ADDR : 0x0000_0400

Table 6-1 List of Registers

Register Name	Symbols	Offset Address	Bit width	Reset value
Initialize configuration register 0	ICG0	0x000	32	Indeterminate
Initialize configuration register 1	ICG1	0x004	32	Indeterminate
Initialize configuration register 2	ICG2	0x008	32	Indeterminate
Initialize configuration register 3	ICG3	0x00C	32	Indeterminate
Initialize configuration register 4	ICG4	0x010	32	Indeterminate
Initialize configuration register 5	ICG5	0x014	32	Indeterminate
Initialize configuration register 6	ICG6	0x018	32	Indeterminate
Initialize configuration register 7	ICG7	0x01C	32	Indeterminate

6.2 Register Description

6.2.1 Initialize configuration register 0 (ICG0)

Reset value: variable

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	WDT SLP OFF	WDTWDPT[3:0]				WDTCKS[3:0]				VWDTPE RI[1 :0]	WDTI TS	WDTA UTS	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	SWDT SLPO FF	SWDTWDPT[3:0]				SWDTCKS[3:0]				SWDTPERI[1:0]	SWDT ITS	SWDTA UTS	

position	Marker	Place Name	Function	Reading and writing
b31~b29	Reserved	-	Reserved space for functions	R
b28	WDTSLPOFF	WDT counts stop in sleep mode	0: WDT does not stop counting in sleep mode 1: WDT in sleep mode count stop	R
b27~b24	WDTWDPT[3:0]	Refresh Allowed Area Count Value Percentage	WDT count value refresh allowed interval 0000: 0%~100% 0001 : 0%~25% 0010: 25%~50% 0011 : 0%~50% 0100: 50%~75% 0101 : 0%~25%,50%~75% 0110: 25%~75% 0111 : 0%~75% 1000: 75%~100% 1001 : 0%~25%,75%~100% 1010 : 25%~50%,75%~100% 1011 : 0%~50%,75%~100% 1100: 50%~100% 1101 : 0%~25%,50%~100% 1110: 25%~100% 1111 : 0%~100%	R
b23~b20	WDTCKS[3:0]	WDT Counting Clock	0010: PCLK3/4 0110: PCLK3/64 0111: PCLK3/128 1000: PCLK3/256 1001: PCLK3/512 1010: PCLK3/1024 1011: PCLK3/2048 1101: PCLK3/8192 Other values: Reserved	R
b19~b18	WDTPERI[1:0]	WDT count overflow cycle	00:256 Cycle 01:4096 Cycle 10: 16384 Cycle	R

			11: 65536 Cycle	
b17	WDTITS	WDT interrupt selection	0: Interrupt request 1: Reset request	R
b16	WDTAUTS	WDT auto-start	0: After reset, WDT starts automatically (hardware start) 1: After reset, WDT stop state	R
b15~b13	Reserved	-	Reserved space for functions	R
b12	SWDTSLOFF	SWDT in Sleep,Stop Count stop in mode	0: SWDT does not stop counting in sleep,stop mode 1: SWDT count stop in sleep,stop mode	R
b11~b8	SWDTWDPT[3:0]	Refresh allowed area count percentage	SWDT count value refresh allowed interval 0000: 0%~100% 0001 : 0%~25% 0010: 25%~50% 0011 : 0%~50% 0100: 50%~75% 0101 : 0%~25%,50%~75% 0110: 25%~75% 0111 : 0%~75% 1000: 75%~100% 1001 : 0%~25%,75%~100% 1010 : 25%~50%,75%~100% 1011 : 0%~50%,75%~100% 1100: 50%~100% 1101 : 0%~25%,50%~100% 1110: 25%~100% 1111 : 0%~100%	R
b7~b4	SWDTCKS[3:0]	SWDT counting clock	0000: SWDTCLK 0100: SWDTCLK/16 0101: SWDTCLK/32 0110: SWDTCLK/64 0111: SWDTCLK/128 1000 : SWDTCLK/256 1011: SWDTCLK/2048 Other values: Reserved	R
b3~b2	SWDTPERI[1:0] 】	SWDT count overflow cycle	00:256 Cycle 01:4096 Cycle 10: 16384 Cycle 11: 65536 Cycle	R
b1	WDTITS	SWDT interrupt selection	0: Interrupt request 1: Reset request	R
b0	WDTAUTS	SWDT auto-start	0: SWDT starts automatically after reset (hardware start) 1: After reset, SWDT stop state	R

6.2.2 Initialize configuration register 1 (ICG1)

Reset value: variable

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
NMI ICG ENA	NFE	NMI ENR	NMI TRG	SMPCLK[1:0] 】	-	-	-	-	-	-	-	BOR DIS	BOR_lev[1: 0]		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	HRC STO P	-	-	-	-	-	-	HRC FRE QSE L	

position	Marker	Place Name	Function	Reading and writing
b31	NMIICGENA	NMI pin ICG setting enable	0: NMI pin ICG setting enable 1: NMI pin ICG setting disabled	R
b30	NFEN	NMI digital filter enable	0: Disable digital filter function 1: Licensed digital filter function	R
b29	NMIENR	NMI pin interrupt selection	0: Disable NMI pin interrupt 1: Permission to interrupt the NMI pin	R
b28	NMITRG	NMI pin edge trigger	0: falling edge 1: Rising edge	R
b27~b26	SMPCLK[1:0]	Filter sampling clock selection	0 0: PCLK3 0 1: PCLK3/8 1 0: PCLK3/32 1 1: PCLK3/64	R
b25~b19	Reserved	-	Reserved space for functions	R
b18	BORDIS	BOR action selection	0: BOR action allowed after reset 1: BOR action is disabled after reset	R
b17~b16	BOR_lev[1:0]	BOR threshold voltage selection	00: 1.9v 01: 2.0v 10: 2.1v 11: 2.3v	R
b15~b9	Reserved	-	Reserved space for functions	R
b8	HRCSTOP	HRC oscillation stop bit	0: HRC oscillation 1: HRC stop	R
b7~b1	Reserved	-	Read "1", write "" when writing	R
b0	HRCFREQSEL	HRC frequency selection	0: 20MHz 1: 16MHz	R

6.2.3 Initialize configuration registers n(ICGn)n=2~3

Reset value: variable

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
ICGn[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ICGn[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b0	ICGn[31:0]	–	Reserved space for functions User setting of all 1s is required to ensure proper chip operation	R

6.2.4 Initialization configuration register 4 (ICG4)

Reset value: variable

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
ICG4 [31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ICG4[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b0	ICG4 [31:0]	Initialization configuration 4	Chip data security protection level 1 enable configuration. If the data is 0xAF180402, the data security protection level 1 is enabled. If the data is other value, the data security protection level 1 is disabled.	R

6.2.5 Initialization configuration register 5 (ICG5)

Reset value: variable

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
ICG5 [31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ICG5 [15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b0	ICG5 [31:0]	Initialization configuration 5	Chip data security protection level 2 enable configuration. If the data is 0xA85173AE, the data security protection level 2 is enabled. When the data is other values, data security protection level 2 is	R

disabled.

6.2.6 Initialize configuration registers n(ICGn)n=6~7

Reset value: variable

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
ICGn[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ICGn[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b0	ICGn[31:0]	–	Reserved space for functions Requires user to set all 1 to ensure proper chip operation	R

7 Embedded FLASH (EFM)

7.1 Introduction

The FLASH interface provides access to FLASH via the FLASH ICODE, DCODE and MCODE buses.

The interface performs programming, sector erase and full erase operations on FLASH; accelerates code execution through instruction prefetch and cache mechanisms.

7.2 Main Features

- Maximum 512KByte FLASH space
- ICODE Bus 16Byte Prefetch
- Shared 64 caches (1KByte) on the ICODE and DCODE buses
- Provides 960Byte One-Time Programming Area (OTP)
- Supports low-power read operations
- Support guide exchange function
- Support data security protection

7.3 Embedded FLASH

FLASH has the following key features:

- The capacity of up to 512 KBytes (32Bytes of which are functionally reserved) is divided into 64 sectors of 8KBytes each.
- The OTP (One Time Program) area is 1020 bytes, divided into 960 bytes of data area and 60 bytes of latch area.
- 128-bit wide data reading.
- The programming unit is 4Bytes and the erase unit is 8KBytes. in the 512KB product, the FLASH address structure is shown in the following figure.

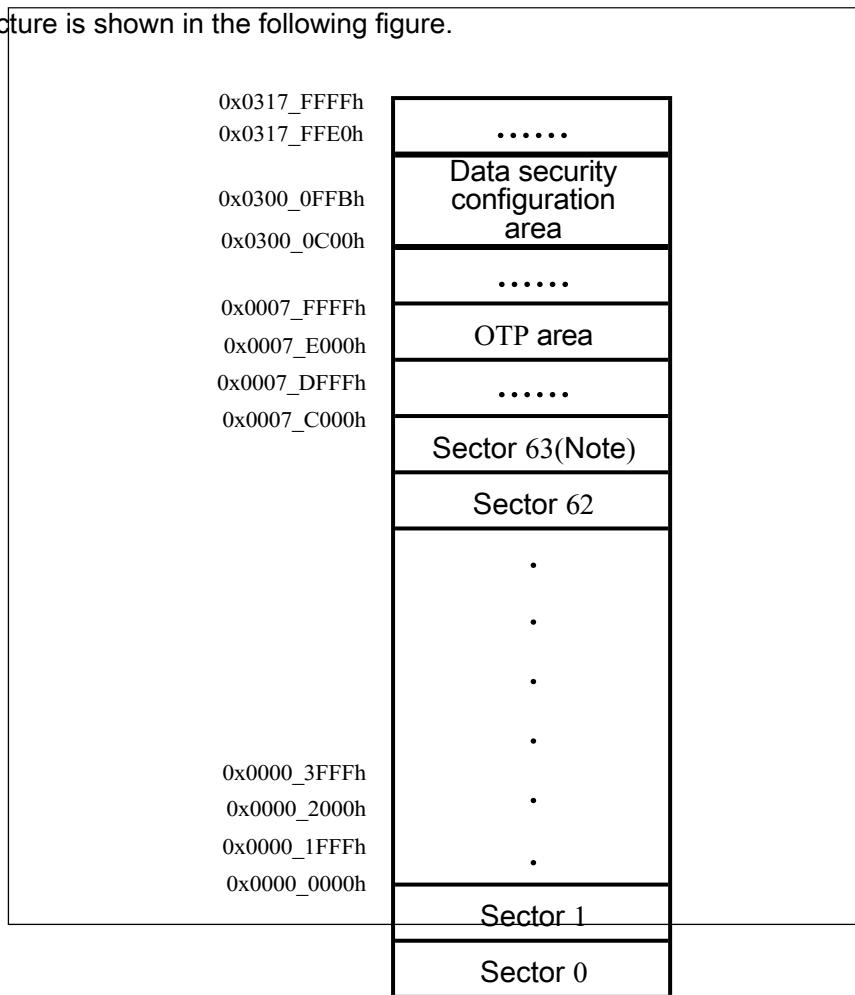


Figure 7-1 FLASH Address Structure (512KB product)

Cautio

n:

- Address 0x0007_FFE0~0x0007_FFFF in sector 63 is the function reserved address; when programming, sector erase and full erase are performed on these 32Bytes

addresses, the FLASH data will not be changed, and when these addresses are read, the read data will be all 1.

In the 256KB product, the FLASH address structure is shown in the figure below.

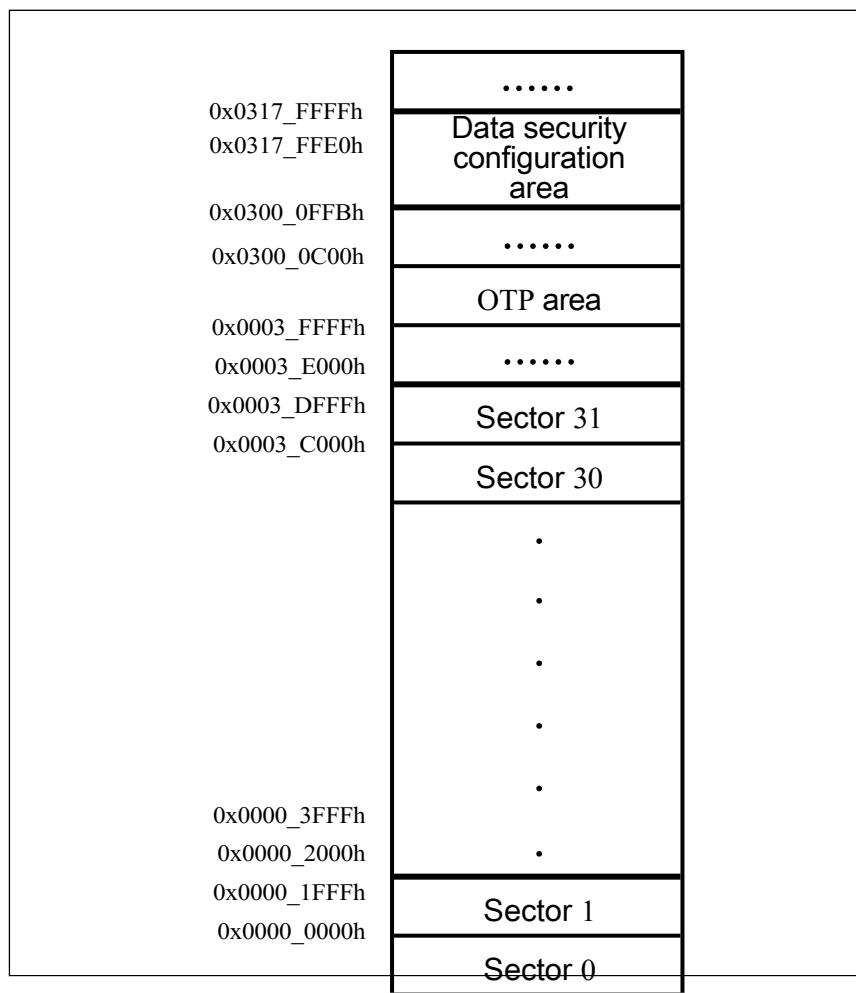


Figure 7-2 FLASH Address Structure (256KB product)

7.4 Read Interface

7.4.1 The relationship between CPU clock and FLASH read time

To read FLASH data correctly, the user needs to set the correct number of wait cycles (**FLWT[3:0]**) in the FLASH Read Mode Register (EFM_FRMC) according to the CPU action frequency.

After system reset, CPU clock source is MRC (8MHz) and FLASH read wait cycle is 0. It is recommended that users modify the CPU main frequency and FLASH read wait cycle bits according to the following steps. Please refer to Table 7-1 for the number of wait cycles.

CPU frequency increase steps:

1. Write the new read wait cycle setting value (**FLWT[3:0]**) to register EFM_FRMC.
2. Read register EFM_FRMC and check if the new wait period is set successfully.
3. The CPU clock frequency is boosted by setting the system clock source switch register CMU_CKSWR(CKSW[2:0]) or the system clock configuration register CMU_SCFGR(HCLKS[2:0]).
4. Read register CMU_CKSWR or CMU_SCFGR to check if the new setting is successful.

CPU frequency reduction steps:

1. The CPU clock frequency is reduced by setting the system clock source switching register CMU_CKSWR(CKSW[2:0]) or the system clock configuration register CMU_SCFGR(HCLKS[2:0]).
2. Read register CMU_CKSWR or CMU_SCFGR to check if the new setting is successful.
3. Write the new read wait cycle setting value (**FLWT[3:0]**) to register EFM_FRMC.
4. Read register EFM_FRMC and check if the new wait period is set successfully.

7.4.2 FLASH Low Power Read

When the CPU clock frequency is below 2MHz, user can set the register bit EFM_FRMC.SLPMD to enter low power read mode to reduce the chip current. After entering the low power read mode, the programming and erase operation of FLASH will be ignored, and this mode erases the FLASH (programming and erase, same below), and the status bit EFM_FSR.COLERR is set.

Enter the ultra-low power read step:

1. Write the new read wait cycle setting value (**FLWT[3:0]**) to register EFM_FRMC according to the recommended value for ultra-low power read mode according to table 7-1.
2. Set the register EFM_FRMC.SLPMD.

Exit ultra-low power read step:

1. Register bit EFM_FRMC.SLPMD is cleared to zero.
2. Write the new read insertion wait period setting (FLWT[3:0]) to register EFM_FRMC according to the recommended value for normal read mode according to table 7-1.

Table 7-1 CPU clock frequency and FLASH read wait cycle comparison table

CPU clock frequency (HCLK)	FRMC register bits FLWT[3:0] set	
	Normal read mode (SLPMD=0)	Ultra low power read mode (SLPMD=1)
168MHz<FHCLK≤200MHz	FLWT[3:0] = 4'b0101 Insert 5 read-wait cycles	Not supported
132MHz<FHCLK≤168MHz	FLWT[3:0]=4'b0100 Insert 4 read-wait cycles	Not supported
99MHz<FHCLK≤132MHz	FLWT[3:0]=4'b0011 Insert 3 read-wait cycles	Not supported
66MHz<FHCLK≤99MHz	FLWT[3:0]=4'b0010 Insert 2 read-wait cycles	Not supported
33MHz<FHCLK≤66MHz	FLWT[3:0]=4'b0001 Insert 1 read-wait cycle	Not supported
2MHz<FHCLK≤33MHz	FLWT[3:0]=4'b0000 No waiting to read	Not supported
1.876MHz<FHCLK≤2MHz	Id.	FLWT[3:0]=4'b1111 Insert 15 read-wait cycles
1.752MHz<FHCLK≤1.876MHz	Id.	FLWT[3:0]=4'b1110 Insert 14 wait read cycles
1.628MHz<FHCLK≤1.752MHz	Id.	FLWT[3:0]=4'b1101 Insert 13 waiting read cycles
1.504MHz<FHCLK≤1.628MHz	Id.	FLWT[3:0]=4'b1100 Insert 12 read-wait cycles
1.38MHz<FHCLK≤1.504MHz	Id.	FLWT[3:0]=4'b1011 Insert 11 read-wait cycles
1.256MHz<FHCLK≤1.38MHz	Id.	FLWT[3:0]=4'b1010 Insert 10 waiting read cycles
1.132MHz<FHCLK≤1.256MHz	Id.	FLWT[3:0]=4'b1001 Insert 9 waiting read cycles
1.008MHz<FHCLK≤1.256MHz	Id.	FLWT[3:0]=4'b1000 Insert 8 waiting read cycles
884KHz<FHCLK≤1.008MHz	Id.	FLWT[3:0]=4'b0111 Insert 7 read-wait cycles
760KHz<FHCLK≤884KHz	Id.	FLWT[3:0]=4'b0110 Insert 6 waiting read cycles
636KHz<FHCLK≤760KHz	Id.	FLWT[3:0]=4'b0101 Insert 5 waiting read cycles
512KHz<FHCLK≤636KHz	Id.	FLWT[3:0]=4'b0100 Insert 4 read-wait cycles

388KHz<FHCLK≤512KHz	Id.	FLWT[3:0]=4'b0011 Insert 3 read-wait cycles
264KHz<FHCLK≤388KHz	Id.	FLWT[3:0]=4'b0010 Insert 2 read-wait cycles
140KHz<FHCLK≤264KHz	Id.	FLWT[3:0]=4'b0001 Insert 1 read-wait cycle
FHCLK≤140KHz	Id.	FLWT[3:0]=4'b0000 No waiting to read

7.5 FLASH Read Acceleration Cache

Each FLASH read operation is a 128-bit read, and the data is sent to the CPU and also stored in the buffer memory. The 128-bit data can be 4 lines of 32-bit instructions or 8 lines of 16-bit instructions, depending on the program burned into the FLASH.

To enable fast reads of FLASH data, the FLASH controller is configured with a read-accelerated cache that optimizes read wait cycles. To take advantage of the processor performance, the gas pedal saves the ICODE,DCODE bus access data to FLASH in the cache registers, thus increasing the program execution speed.

The system provides 1KBytes of space for cache memory, which can effectively reduce the time loss due to instruction hopping. The cache function is enabled through the Cache Enable (CACHE) position 1 in the EFM_FRMC register. Whenever there is an instruction or data miss (i.e., the requested instruction or data does not exist in the currently used instruction line or cache memory), the system copies the newly read data line (128 bits) to the cache memory. If the instruction or data requested by the CPU already exists in the cache, it can be fetched immediately without any delay. When the cache memory is full, the LRU (Least Recently Used) policy is used to determine the data to be replaced in the cache memory.

When CPU reads instruction or data, FLASH address in the buffer, cache hit, read FLASH cycle number will change, please refer to Table 7-2 for details.

Table 7-2 FLASH Actual Read Cycle Count

EFM_FRMC. FLWT[3:0] setting	Cache is not enabled (EFM_FRMC.CACHE=0)		Cache enable (EFM_FRMC.CACHE=1)	
	Buffering, cache hits	Buffering, cache not hitting	Buffering, cache hits	Buffering, cache not hitting
0	1	1	1	1
1	1	2	1	2
2	1	3	1	3
3	1	4	1	5
4	1	5	1	6
N(N>4)	1	N+1	1	N+2

7.6 FLASH Programming and Erase Operations

FLASH supports programming, sector erase, and full erase operations.

The FLASH programming unit is 4Bytes, the last bit of the programming address must be aligned with 4 (the last address is 0x0,0x4,0x8,0xC), repeated programming does not ensure correct programming. flash sector erase unit is 8KBytes, please disable the cache before FLASH erase programming. The following describes the setting procedures for programming and erasing operations.

7.6.1 Single programming without readback mode

The single programming no readback mode is set as follows:

- 1) Unprotect the FLASH registers from write protection. (EFM_FAPRT writes 0x0123 first, then 0x3210)
- 2) Set programming, erase mode permission. (EFM_FWMC.PEMODE=1)
- 3) Set single programming mode. (EFM_FWMC.PEMODE[2:0]=001)
- 4) Writes 32-bit data to the programmed address.
- 5) Wait for FLASH to be in idle state. (EFM_FSR.RDY=1)
- 6) Reading the programmed address value to determine if it is the same as the written value; Consistent means the programming was successful, inconsistent means the FLASH address is corrupted and permanently discarded.
- 7) Clear the end-of-programming flag bit. (EFM_FSR.OPTEND)

7.6.2 Single programming readback mode

The single programmed readback mode means that the programmed address is read and compared with the written data after the programming is finished, and the consistent flag bit EFM_FSR is output - PGMISMTCH.

The single programming readback mode is set as follows:

- 1) Unprotect the FLASH registers from write protection. (EFM_FAPRT writes 0x0123 first, then 0x3210)
- 2) Set programming, erase mode permission. (EFM_FWMC.PEMODE=1)
- 3) Set single programming readback mode. (EFM_FWMC.PEMODE[2:0]=010)
- 4) Writes 32-bit data to the programmed address.
- 5) Wait for FLASH to be in idle state. (EFM_FSR.RDY=1)
- 6) Determines the programming self-read result flag bit. (EFM_FSR.PGMISMTCH) If it is 0, the programming is successful; if it is 1, the FLASH address is corrupted and permanently discarded.

-
- 7) Clear the end-of-programming flag bit. (EFM_FSR.OPTEND)

7.6.3 Continuous programming operation

Continuous programming mode is recommended when programming FLASH addresses continuously.

Continuous programming mode can save more than 50% of time compared to single programming mode. For continuous programming mode, the continuous programming write interval should be less than 16us:

- 1) Unprotect the FLASH registers from write protection. (EFM_FAPRT writes 0x0123 first, then 0x3210)
- 2) Set programming, erase mode permission. (EFM_FWMC.PEMODE=1)
- 3) Set the continuous programming mode. (EFM_FWMC.PEMOD[2:0]=011)
- 4) Transfer steps 8) 9) 10) 11) 12) 13) 14) to a region outside of FLASH for execution.
- 5) Jump to transfer to step 4) destination address.
- 6) Reads the programmed address and determines if it matches the written value.
- 7) Consistent means the programming is successful, inconsistent means the FLASH address is corrupted and permanently discarded.
- 8) Write 32-bit data to the programmed address.
- 9) Wait for the operation end flag bit (EFM_FSR.OPTEND) **to be set**.
- 10) Clear the end-of-operation flag bit until the flag bit EFM_FSR.OPTEND is read as 0.
- 11) Repeat 8), 9), 10) until all data is written.
- 12) Modify the erase mode control register to non-continuous programming mode.
- 13) Wait for FLASH to be in idle state. (EFM_FSR.RDY=1)
- 14) Jump back to the

main program. Caution:

- - If a read operation to FLASH occurs during continuous FLASH programming, an indefinite value will be read. The read operation will cause the read conflict bit EFM_FSR.COLERR to be set and needs to be cleared by setting the EFM_FSCLR register.

7.6.4 Erase operation

EFM provides two types of erase methods, sector erase and full erase. The sector erase and full erase operations are set as follows:

- 1) Unprotect the FLASH registers from write protection. (EFM_FAPRT writes 0x0123 first, then 0x3210)
- 2) Set programming, erase mode permission. (EFM_FWMC.PEMODE=1)
- 3) Set the erase mode. (Sector Erase EFM_FWMC.PEMOD[2:0]=100, full erase EFM_FWMC.PEMOD[2:0]=101)
- 4) Write a 32-bit arbitrary value to any address in the sector to be erased (the address must be aligned with 4).
- 5) Write a 32-bit arbitrary value to any FLASH address (address must be aligned with 4) during full erase.
- 6) Wait for FLASH to be in idle state. (EFM_FSR.RDY=1)
- 7) Clear the end-of-erase flag bit. (EFM_FSR.OPTEND)

7.6.5 Data Security Protection

This product provides 3 levels of protection for FLASH data to prevent untrusted users from reading and tampering with FLASH through the debug interface (JTAG and SWD interfaces) ISP interface (In System Program) and test interface.

Protection level 0: No protection

The debug interface, ISP interface and test interface can access (read and rewrite)

MCU resources, including FLASH data. Protection level 1:

FLASH address 0x0000_0410~0x0000_0413 is programmed to write data 0xAF180402, while address 0x0317_FFE0~0x0317_FFF7 is programmed to write 96-bit password, address 0x0317_FFE0~0x0317_FFEB is programmed to write 96-bit 0, protection level 1 is enabled.

Protection level 1 When enabled

- The debug interface is closed and the ISP interface and test interface cannot access the FLASH data.
- The user program cannot program and erase sector 0, sector 1 and sector 63.
- The data at address 0x0317_FFE0~0x0317_FFF7 cannot be read.

After activating protection level 1, you can revert to protection level 0 by password authentication method and full erase method.

Protection level 2:

FLASH address 0x0000_0414~0x0000_0417 programmed to write data 0xA85173AE, while address 0x0317_FFE0~0x0317_FFEB programmed to write 96 bits 0, protection level 2 enabled. After protection level 2 is enabled

- The debug interface is closed and the ISP interface and test interface cannot access the FLASH data.
- The user program cannot program and erase sector 0, sector 1 and sector 63.
- The data at address 0x0317_FFE0~0x0317_FFF7

cannot be read out. After activating protection level 2, it can be restored to protection level 0 by full erase.

Protection level 1 and protection level 2 can be enabled separately or simultaneously. When enabled at the same time, the protection measures are stacked.

7.6.6 Bus hold function

BUSHLDCTL bit can be set to hold or release the bus during FLASH programming and erase, and the control bit must be set to 0 when the FLASH programming and erase instruction is executed on

FLASH, and can be freely set as needed when the erase instruction is executed in a space other than FLASH (such as high-speed RAM). The control bit can be freely set as needed when the erase instruction is executed in a space other than FLASH (such as high-speed RAM).

When BUSHLDCTL is set to 1 (FLASH programming, bus release state during erase), read and write accesses to FLASH during programming (except continuous programming), before the end of erase (EFM_FSR.RDY=1) will be ignored, flag bit EFM_FSR.COLERR position bit.

7.6.7 FLASH erase, programming window protection

Window protection is provided for FLASH, only in the allowed area FLASH can be sector erased and programmed, otherwise erase error interrupt occurs. When programming and sector erase operation, the hardware circuit will pre-determine whether it is in the allowed area or not, and the full erase mode is not limited by window protection. The start position and end position of the window are set by registers EFM_FPMTEW, EFM_FPMTSW.

The specific protection functions are as follows:

- Register EFM_FPMTEW = Register EFM_FPMTSW The entire FLASH area is erasable and programmable.
- Register EFM_FPMTEW > Register EFM_FPMTSW is erasable and the programming area is in between.
- Register EFM_FPMTEW < Register EFM_FPMTSW The whole FLASH area is not erasable and programmable.

7.6.8 Interruptions

The EFM module has 3 interrupts, namely PE (Program/Erase) error interrupt, read/write conflict interrupt and end of operation interrupt.

When a PE error interrupt is set on FLASH, FLASH cannot be programmed/erased/full erased. The flag bit must be cleared before the FLASH can be programmed/erased/full erased again.

1. PE error interrupt EFM_PEERR:

Positioning:

- The programmed address is not aligned with 4 or the data size is not 32 bits (PGSZERR=1).
- Perform programming of the address inside the FLASH window protection, sector erase operation (PEPRTERR=1).
- When no erase mode is set, a write operation to FLASH is performed (PEWERR=1).
- The programmed address self-read value does not match the write value (PGMISMTCH=1) in single programmed readback mode.

Zeroing:

Register EFM_FSCLR Corresponding flag clear bit is written 1, status bit is cleared to zero.

2. FLASH read-write conflict interrupt EFM_COLERR:

Positioning:

- FLASH read operation occurs in FLASH continuous programming mode.
- FLASH read/write operation occurs during FLASH stop mode.
- FLASH programming, FLASH read/write operation occurs before erase is completed.

- Write operations to FLASH occur in low-power read mode.

Zeroing:

The register EFM_FSCLR corresponds to clear position 1 and the status bit is cleared to zero.

3. End-of-operation interrupt EFM_OPTEND:**Positioning:**

- Programming mode: End of programming for a single address.
- Erase mode: sector erase, end of full erase.

Zeroing:

The register EFM_FSCLR corresponds to clear position 1 and the status bit is cleared to zero.

7.7 One-time programmable bytes

The following table shows the OTP area one-time programmable address composition, divided into 960Bytes of data area and 60Bytes of latch area.

Table 7-3 OTP Address Composition

Block Name	OTP data block address	OTP lock address
Block 0	0x0300_0C00~0x0300_0C3F	0x0300_0FC0~0x0300_0FC3
Block 1	0x0300_0C40~0x0300_0C7F	0x0300_0FC4~0x0300_0FC7
Block 2	0x0300_0C80~0x0300_0CBF	0x0300_0FC8~0x0300_0FCB
Block 3	0x0300_0CC0~0x0300_0cff	0x0300_0FCC~0x0300_0FCF
Block 4	0x0300_0D00~0x0300_0D3F	0x0300_0FD0~0x0300_0FD3
Block 5	0x0300_0D40~0x0300_0D7F	0x0300_0FD4~0x0300_0FD7
Block 6	0x0300_0D80~0x0300_0DBF	0x0300_0FD8~0x0300_0FDB
Block 7	0x0300_0DC0~0x0300_0dff	0x0300_0FDC~0x0300_0FDF
Block 8	0x0300_0E00~0x0300_0E3F	0x0300_0FE0~0x0300_0FE3
Block 9	0x0300_0E40~0x0300_0E7F	0x0300_0FE4~0x0300_0FE7
Block 10	0x0300_0E80~0x0300_0EBF	0x0300_0FE8~0x0300_0FEB
Block 11	0x0300_0EC0~0x0300_0EFF	0x0300_0FEC~0x0300_0FEF
Block 12	0x0300_0F00~0x0300_0F3F	0x0300_0FF0~0x0300_0FF3
Block 13	0x0300_0F40~0x0300_0F7F	0x0300_0FF4~0x0300_0FF7
Block 14	0x0300_0F80~0x0300_0FBF	0x0300_0FF8~0x0300_0FFB

The OTP area is divided into 15 64-byte data blocks, each corresponding to one 4Bytes latch address. The latch address is used to latch the corresponding data block. When the latch address data is all 1s, the corresponding OTP area data block is programmable; when the latch address data is all 0s, the corresponding OTP area data is not programmable. All OTP data blocks and latch

addresses are not erasable.

7.8 Guided exchange

If users want to upgrade the boot program, they need to erase sector 0 (0x0000_0000~0x0000_1FFF), if the erase encounters unpredictable accidents (such as reset, power failure), it may cause the whole chip can not boot normally. For 512KB products, EFM provides a boot sector swap function (256KB products do not have this function). Before erasing sector 0, the new boot program is written to sector 1 (0x0000_2000~0x0000_3FFF), and then the EFM address 0x0007_FFDC is programmed with data 0xFFFF_4321. The CPU starts the new boot program from sector 1 after the MCU is reset, and then erases sector 0 and reprograms the new user program. The user program is reprogrammed.

Refer to Figure 7-3 for the start-up exchange operation.

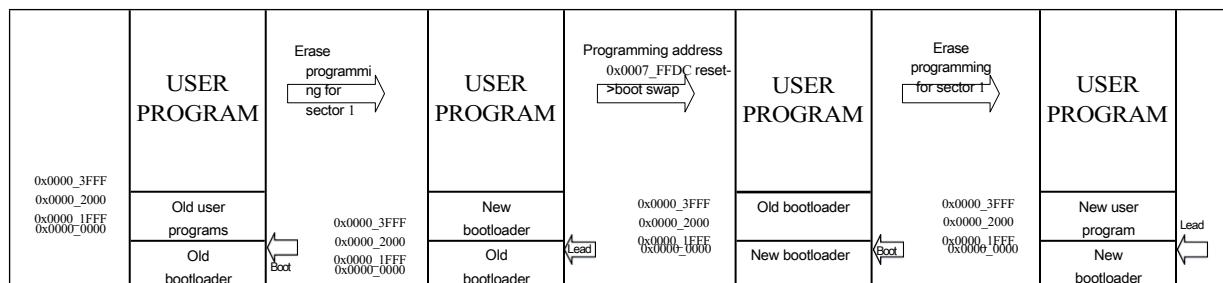


Figure 7-3 Boot Sector Swap Function 1

When the user needs to upgrade the bootloader again, since the address 0x0007_FFDC where the boot sector swap information is stored has already been programmed (the user can determine whether the boot swap function has been used by reading the FLASH address or the EFM_FSWP register), the bootloader needs to be upgraded again by sector erasing sector 63. After the MCU is reset, the CPU starts the new boot program from sector 1, and then erases sector 0 to reprogram the new user program. The operation flow is shown in Figure 7-4.

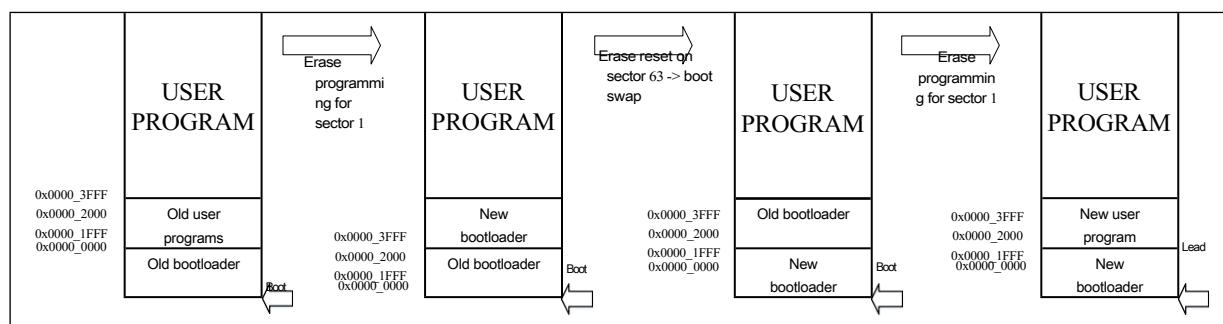


Figure 7-4 Launching the switching function 2

7.9 Register Description

EFM_BASE_ADDR: 0x4001_0400

Table 7-4 Register List

Register Description	Register Name	Offset	Bit width	Reset value
FLASH Access Protection Register	EFM_FAPRT	0x0000	32	0x0000_0000
FLASH Stop Register	EFM_FSTP	0x0004	32	0x0000_0000
FLASH Read Mode Register	EFM_FRMC	0x0008	32	0x0000_0000
FLASH Erase Mode Register	EFM_FWMC	0x000C	32	0x0000_0000
FLASH Status Register	EFM_FSR	0x0010	32	0x0000_0100
FLASH Status Clear Register	EFM_FSCLR	0x0014	32	0x0000_0000
FLASH Interrupt License Register	EFM_FITE	0x0018	32	0x0000_0000
FLASH Boot Swap Status Register	EFM_FSWP	0x001C	32	Indeterminate
FLASH Rewrite Allowed Area Start Address Register	EFM_FPMTSW	0x0020	32	0x0000_0000
FLASH Rewrite Allowed Area End Address Register	EFM_FPMTEW	0x0024	32	0x0000_0000
FLASHuniqueID register	FEM_UQID1	0x0050	32	Indeterminate
FLASHuniqueID register	EFM_UQID2	0x0054	32	Indeterminate
FLASHuniqueID register	EFM_UQID3	0x0058	32	Indeterminate

7.9.1 Access Protection Register (EFM_FAPRT)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FAPRT [15:0]															

position	Marker	Place Name	Function	Reading and writing
b31-b16	Reserved	-	Read 0, write "0" when writing	R
b15-b0	FAPRT [15:0]	EFM register write protection	Access the EFM register protection register. Solution: Write "16-bit data 0x0123" and then write "16-bit data 0x3210" to FAPRT. When any data is written in the unprotected state, the EFM register enters the protected state again. When EFM register access protection is in effect, the value of this register readout is 0x0000_0000. When the EFM register access protection is invalid, this register reads the value 0x0000_0001.	R/W

7.9.2 FLASH Stop Register (EFM_FSTP)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	FSTP

position	Marker	Place Name	Function	Reading and writing
b31-b1	Reserved	-	Read 0, write "0" when writing	R
b0	FSTP	FLASH stop mode control	0: FLASH active status 1 : FLASH is in stop mode When the register bit is set from 1 to 0, please perform FLASH access after confirming that the FSR.RDY bit is 1.	R/W

7.9.3 Read Mode Register (EFM_FRMC)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	CRS T	-	-	-	-	-	-	-	CAC HE
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	LVM	FLWT[3:0]				-	-	-	SLP MD

position	Marker	Place Name	Function	Reading and writing
b31~b25	Reserved	-	Read "0", write "0" when writing	R
b24	CRST	Cache reset bit	0: Cached data is not reset 1: Reset cached data	R/W
b23~b17	Reserved	-	Read "0", write "0" when writing	R
b16	CACHE	Cache license bits	0: Turn off the cache function 1: Cache function enable	R/W
b15~b9	Reserved	-	Read "0", write "0" when writing	R
b8	LVM	Ultra low speed operation mode	0: Turn off ultra-low speed operation mode 1: Open ultra-low speed operation mode	R/W
b7-b4	FLWT[3:0]	FLASH read insert wait cycle	0000b: No insert wait cycle 0001b: Insert 1 wait cycle 0010b: Insert 2 wait cycles 1110b: insert 14 waiting cycles 1111b: insert 15 waiting cycles	R/W
b3~b1	Reserved	-	Read "0", write "0" when writing	R
b0	SLPMD	Ultra low power readout	0: Normal CPU read mode 1: Ultra-low power read mode	R/W

7.9.4 Erase Write Mode Register (EFM_FWMC)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PEM ODE

position	Marker	Place Name	Function	Reading and writing
b31~b9	Reserved	-	Read "/" , write "0" when writing	R
b8	BUSHLDCTL	FLASH erase, bus control during programming	0: The bus is occupied during FLASH programming erase. 1: Bus release during FLASH programming erase.	R/W
b7	Reserved	-	Read "/" , write "0" when writing	R
b6~b4	PEMOD[2:0]	FLASH Erase, Programming Mode	000: Read-only mode 001: Single programming mode 010: Single programming readback mode 011: Continuous programming mode 100: Sector erase mode 101: Full erase mode 110: Read-only mode 111: Read-only mode PEMOD[2:0] can be written only if PEMODE=1.	R/W
b3~b1	Reserved	-	Read "/" , write "0" when writing	R
b0	PEMODE	FLASH Erase, Programming License Mode	0: PEMOD[2:0] is not allowed to be rewritten 1: PEMOD[2:0] rewrite permission	R/W

7.9.5 Status register (EFM_FSR)

Reset value: 0x0000_0100

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

-	-	-	-	-	-	-	RDY	-	-	COL ERR	OPT END	PGM ISM TCH	PGS ZER R	PEP RTE RR	PEW ERR
---	---	---	---	---	---	---	-----	---	---	------------	------------	-------------------	-----------------	------------------	------------

position	Marker	Place Name	Function	Reading and writing
b31~b9	Reserved	-	Read "0", write "0" when writing	R
b8	RDY	FLASH busy/idle status	0 : FLASH busy status 1 : FLASH idle state	R
b7~b6	Reserved	-	Read "0", write "0" when writing	R
b5	COLERR	FLASH read/write access error flag bit	0: FLASH read/write access is normal 1 : FLASH read/write access error so that FLASH read and write access operations in this location bit will be ignored.	R
b4	OPTEND	Operation end flag bit	0: FLASH is not erased or FLASH is being erased 1: End of FLASH erase, set condition: End of program/erase/full erase operation	R
b3	PGMISMTCH	Single programmed readback inconsistent value flag bit	0: Single programming back to read the same value 1: Single programming readback value inconsistent	R
b2	PGSZERR	Programming address and size misalignment flag bits	0: Programmed address and size alignment 1: Programming address and size are not aligned	R
b1	PEPRERR	Program/erase error flag bit for protected address	0: Programming/erasing address is an allowed rewrite area 1: Program/erase action for protection window address	R
b0	PEWERR	Erase mode error flag bit	0: Erase FLASH in erase permission mode 1: Erase FLASH in erase disable mode	R

7.9.6 Status clear register (EFM_FSCLR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	COL ERR CLR	OPT ENDC LR	PG MIS MTC HCL R	PGS ZER RCL R	PEP RTE RRC LR	PEW ERR CLR

position	Marker	Place Name	Function	Reading and writing
b31~b6	Reserved	-	Read "0", write "0" when writing	R
b5	COLERRCLR	Clear the read-write conflict error flag bit	0: Clear action does not occur 1: Clear the read/write conflict error This bit is always 0 when read.	R/W
b4	OPTENDCLR	Clear the end-of-operation flag	0: Clear action does not occur 1: Clear the end-of-operation flag This bit is always 0 when read.	R/W
b3	PGMISMTCHCLR	Clear programming readback inconsistency flag bit	0: Clear action does not occur 1: Clear the programming readback inconsistency flag bit This bit is always 0 when read.	R/W
b2	PGSZERRCLR	Clear programmed address and size misalignment flag bits	0: Clear action does not occur 1: Clear the unaligned address and size error flag bit This bit is always 0 when read.	R/W
b1	PEPRERRCLR	Clear the program/erase error flag bit for protected addresses	0: Clear action does not occur 1: Clear the programming/erase error This bit is always 0 when read.	R/W
b0	PEWERRCLR	Erase mode error flag bit	0: Clear action does not occur 1: Clear mode write error This bit is always 0 when read.	R/W

7.9.7 Interrupt permission register (EFM_FITE)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	COL ERR ITE	OPT END ITE	PEE R R I TE

position	Marker	Place Name	Function	Reading and writing
b31~b3	Reserved	-	Read "0", write "0" when writing	R
b2	COLERRITE	Read-write conflict error interrupt permit	0: Read/write conflict error interrupt is not allowed 1: Read-write conflict error interrupt license	R/W
b1	OPTENDITE	End of operation interrupt permission	0: Operation end interrupt is not allowed 1: End of operation interrupt permission	R/W
b0	PEERRITE	Programming/erasing error interrupt permit	0: Programming/erasing error interrupt not allowed 1: Programming/erasing error interrupt permit	R/W

7.9.8 Bootstrap Switching Status Register (EFM_FSWP)

Reset value: variable

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	FSWP

position	Marker	Place Name	Function	Reading and writing
b31-b1	Reserved	-	Read "0", write "0" when writing	R
b0	FSWP	Sector 0 and sector 1 address swap bits	0: Sector 0 and sector 1 addresses are swapped After reset, the CPU boots from sector 1. 1: Sector 0 and sector 1 addresses are not swapped After reset, CPU starts from sector 0. The initial value of the register is determined by the FLASH address The decision of	R

0x0007_FFDC~0x0007_FFD_F, which
When the address data is 0xFFFF_4321,
the initial value of FSWP register bit FSWP is
0 after reset, otherwise it is 1.

7.9.9 FLASH Window Protection Start Address Register (EFM_FPMTSW)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	FPMTSW [18:16]
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FPMTSW [15:0]															

position	Marker	Place Name	Function	Reading and writing
b31-b19	Reserved	-	Read "0", write "0" when writing	R
b18-b0	FPMTSW [18:0]	Protection window start address	FLASH protection window start address	R/W

7.9.10 FLASH End of Window Protection Address Register (EFM_FPMTEW)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	FPMTEW [18:16]
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FPMTEW [15:0]															

position	Marker	Place Name	Function	Reading and writing
b31-b19	Reserved	-	Read "0", write "0" when writing	R
b18-b0	FPMTEW [18:0]	End of protection window address	End address of FLASH protection window	R/W

7.9.11 UNIQUE ID register (EFM_UQID1)

Reset value: variable

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
UQID1[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
UQID1[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31-b0	UQID1[31:0]	Unique Code	Chip Unique Code	R

7.9.12 UNIQUE ID register (EFM_UQID2)

Reset value: variable

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
UQID2[31:16]															
]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
UQID2[15:0]															
position	Marker	Place Name	Function	Reading and writing											
b31-b0	UQID2[31:0]	Unique Code	Chip Unique Code	R											

7.9.13 UNIQUE ID register (EFM_UQID3)

Reset value: variable

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
UQID3															
[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
UQID3[15:0]															
position	Marker	Place Name	Function	Reading and writing											
b31-b0	UQID3[31:0]	Unique Code	Chip Unique Code	R											

7.10 Cautions

1. When FLASH is erased, reset occurs, erase operation will be forced to stop and FLASH data will not be guaranteed. The user needs to operate again after the reset is lifted and the address is erased.
2. If the erase operation will involve the data in the cache, reset the cache loop after the erase operation (FRMC.CRST0=1).
3. programming, please set the cache to invalid before the erase operation (FRMC.CACHE=0).
4. One-time programmable area programming operation is not controlled by the window protection register (FPMTSW, FPMTEW).
5. In continuous programming mode, FLASH analog circuit will have high voltage status, and the long-term high voltage status will affect FLASH characteristics. Forbid the MCU to enter low-power mode (sleep mode, stop mode, power-down mode).
6. When bus release (FWMC.BUSHLDCTL=1) is set during programming and erasing, set the interrupt vector and interrupt subroutine to RAM if you want to respond to interrupts during programming and erasing.

8 Internal SRAM (SRAM)

8.1 Introduction

This product comes with 4KB power-down mode hold SRAM (Ret_SRAM) and 188KB system SRAM (SRAMH/SRAM1/SRAM2/SRAM3)

SRAM can be accessed by byte, half-word (16-bit), or full-word (32-bit). Read and write operations are performed at CPU speed, and wait cycles can be inserted. The relationship between the wait period setting for read/write access and CPU clock frequency is shown in Table 8-1. The wait period for each SRAM read/write access is set by the SRAM wait control register (SRAM_WTCR).

Table 8-1 Relationship between the wait period setting of SRAM read/write access and CPU clock frequency

Waiting cycle (CPU access cycle)	CPU clock frequency range allowed to access high-speed SRAM (SRAMH)	Access to other SRAMs (SRAM1,2,3,Ret_SRAM) Allowed CPU clock frequency range
0wait(1 CPU cycle access)	0~200MHz	0~100MHz
1wait(2 CPU cycles access)	0~200MHz	0~200MHz
2wait(3 CPU cycles access)	0~200MHz	0~200MHz
3wait(4 CPU cycles access)	0~200MHz	0~200MHz
4wait(5 CPU cycles access)	0~200MHz	0~200MHz
5wait(6 CPU cycles access)	0~200MHz	0~200MHz
6wait(7 CPU cycles access)	0~200MHz	0~200MHz
7wait(8 CPU cycles access)	0~200MHz	0~200MHz

Ret_SRAM provides 4KB of data retention space in power down mode.

SRAM3 has ECC check (Error Checking and Correcting) ECC check is correcting one and checking two codes, that is, it can correct one error and check two errors; SRAMH/SRAM1/SRAM2/Ret_SRAM has Even-parity check, each byte of data has one parity bit. The detailed definition of SRAM is shown in Table 8-2.

Table 8-2 SRAM Space Allocation

Name	Capacity	Address Range	Calibration method
SRAM1	64KB	0x2000_0000~0x2000_FFFF	Even-parity check
SRAM2	64KB	0x2001_0000~0x2001_FFFF	Even-parity check
SRAM3	28KB	0x2002_0000~0x2002_6FFF	ECC check
Ret_SRAM	4KB	0x200F_0000~0x200F_0FFF	Even-parity check
SRAMH	32KB	0x1FFF_8000~0x1FFF_FFFF	Even-parity check

Caution:

- When using SRAM3 as the stack space, the wait time for SRAM3 must be set to **1 wait**, i.e., 2 CPU cycles or more for access.
- Where RAM parity errors are allowed to generate NMI interrupts and resets, the RAM space used must be initialized in words when accessing data, and in words when executing instructions from SRAMH space, the area of RAM space used +3 words must be initialized in words.
- Where RAM ECC checksum errors are allowed to generate NMI interrupts and resets, the RAM space used must be initialized in words when accessing data.
- The border of SRAMH and SRAM1 does not support non-aligned accesses and must be used to avoid **32-bit** accesses to the 0x1FFF_FFFD/0x1FFF_FFFE/0x1FFF_FFFF addresses.
The 0x1FFF_FFFF address initiates a **16bit** access.

8.2 Register Description

Register Name	Start Address	Reset value
SRAM wait control register (SRAM_WTCR)	0x4005_0800	0x0000_0000
SRAM wait protection register (SRAM_WTPR)	0x4005_0804	0x0000_0000
SRAM checksum control register (SRAM_CKCR)	0x4005_0808	0x0000_0000
SRAM checksum protection register (SRAM_CKPR)	0x4005_080C	0x0000_0000
SRAM checksum status register (SRAM_CKSR)	0x4005_0810	0x0000_0000

8.2.1 SRAM Wait Control Register (SRAM_WTCR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Rev	SRAMR_ WWT[2:0]	Rev	SRAMR_ RWT[2:0]	Rev	SRAMH_ WWT[2:0]	Rev	SRAMH_ RWT[2:0]	Rev	SRAMH_ WWT[2:0]	Rev	SRAMH_ RWT[2:0]	Rev	SRAMH_ WWT[2:0]	Rev	SRAMH_ RWT[2:0]
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Rev	SRAM3_ WWT[2:0]	Rev	SRAM3_ RWT[2:0]	Rev	SRAM12_ WWT[2:0]	Rev	SRAM12_ RWT[2:0]	Rev	SRAM12_ WWT[2:0]	Rev	SRAM12_ RWT[2:0]	Rev	SRAM12_ WWT[2:0]	Rev	SRAM12_ RWT[2:0]

position	Marker	Place Name	Function	Reading and writing
b31	Reserved	-	Read "1", write "0" when writing	R/W
b30~b28	SRAMR_ WWT[2:0]	Ret_SRAM Write Period Selection	000b: 1 cycle write 001b: 2-cycle write 010b: 3-cycle write 011b: 4 cycle write 100b: 5 cycles to write 101b: 6 cycle write 110b: 7-cycle write 111b: 8-cycle write	R/W
b27	Reserved	-	Read "1", write "0" when writing	R/W
b26~b24	SRAMR_ RWT[2:0]	Ret_SRAM read cycle selection	000b: 1 cycle read 001b: 2-cycle read 010b: 3-cycle read 011b: 4-cycle reading 100b: 5 cycle read 101b: 6 cycle reading 110b: 7-cycle read 111b: 8-cycle reading	R/W
b23	Reserved	-	Read "1", write "0" when writing	R/W
b22~b20	SRAMH_ WWT[2:0]	SRAMH write cycle selection	000b: 1 cycle write 001b: 2-cycle write 010b: 3-cycle write 011b: 4 cycle write 100b: 5 cycles to write	R/W

			101b: 6 cycle write 110b: 7-cycle write 111b: 8-cycle write	
b19	Reserved	-	Read "1", write "0**" when writing	R/W
b18~b16	SRAMH_ RWT[2:0]	SRAMH read cycle selection	000b: 1 cycle read 001b: 2-cycle read 010b: 3-cycle read 011b: 4-cycle reading 100b: 5 cycle read 101b: 6 cycle reading 110b: 7-cycle read 111b: 8-cycle reading	R/W
b15	Reserved	-	Read "1", write "0**" when writing	R/W
b14~b12	SRAM3_ WWT[2:0]	SRAM3 write cycle selection	000b: 1 cycle write 001b: 2-cycle write 010b: 3-cycle write 011b: 4 cycle write 100b: 5 cycles to write 101b: 6 cycle write 110b: 7-cycle write 111b: 8-cycle write	R/W
b11	Reserved	-	Read "1", write "0**" when writing	R/W
b10~b8	SRAM3_ RWT[2:0]	SRAM3 read cycle selection	000b: 1 cycle read 001b: 2-cycle read 010b: 3-cycle read 011b: 4-cycle reading 100b: 5 cycle read 101b: 6 cycle reading 110b: 7-cycle read 111b: 8-cycle reading	R/W
b7	Reserved	-	Read "1", write "0**" when writing	R/W
b6~b4	SRAM12_ WWT[2:0]	SRAM1 and SRAM2 write weeks Period Selection	000b: 1 cycle write 001b: 2-cycle write 010b: 3-cycle write 011b: 4 cycle write 100b: 5 cycles to write 101b: 6 cycle write 110b: 7-cycle write 111b: 8-cycle write	R/W
b3	Reserved	-	Read "1", write "0**" when writing	R/W
b2~b0	SRAM12_ RWT[2:0]	SRAM1 and SRAM2 read weeks Period Selection	000b: 1 cycle read 001b: 2-cycle read 010b: 3-cycle read 011b: 4-cycle reading 100b: 5 cycle read 101b: 6 cycle reading 110b: 7-cycle read 111b: 8-cycle reading	R/W

8.2.2 SRAM wait protection register (SRAM_WTPR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Rev															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Rev								WTPRKW[6:0]						WTPRC	

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	–	Read "1", write "0" when writing	R/W
b7~b1	WTPRKW[6:0]	Write keycode	When writing to the current register, write "3b" to these bits to enable the current register.	R/W
b0	WTPRC	SRAM wait control register write control	0: SRAM wait control register write disable 1: SRAM wait control register write enable	R/W

WTPRC: Control the SRAMWTCR register write operation. When WTPRC is set to 1, write operations to SRAMWTCR are allowed, if it is set to 0, write operations to SRAMWTCR cannot be performed. When this bit is written, 0x3B must be written to WTPRKW[6:0] at the same time.

8.2.3 SRAM checksum control register (SRAM_CKCR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Rev					ECCMOD [1:0]	Rev					ECC OAD				
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Rev															PYOAD
position	Marker	Place Name					Function					Reading and writing			
b31~b26	Reserved	-					Read "1", write "0" when writing					R/W			
b25~b24	ECC MOD [1:0]	ECC checksum allowable bit of SRAM3					00: ECC check function is disabled 01: If 1 bit is wrong, ECC corrects the error. No 1-bit error flag is generated, no interrupt/reset is generated; if 2-bit error, ECC detects the error and Generate 2-bit error flag, generate interrupt/reset. 10: If 1 bit is wrong, ECC corrects the error. Generate 1-bit error flag, no interrupt/reset; if 2-bit error, ECC detects error Generate 2-bit error flag, generate interrupt/reset. 11: If 1 bit is wrong, ECC corrects the error. Generate 1-bit error flag, generate interrupt/reset; if 2-bit error, ECC detects error Generate 2-bit error flag, generate interrupt/reset.					R/W			
b23~b17	Reserved	-					Read "1", write "0" when writing					R/W			
b16	ECCOAD	ECC check Operation after error					0: Non-maskable interrupt 1: Reset					R/W			
b15~b1	Reserved	-					Read "1", write "0" when writing					R/W			
b0	PYOAD	Parity check Operation after error					0: Non-maskable interrupt 1: Reset					R/W			

Caution:

- Where RAM parity errors are allowed to generate NMI interrupts and resets, the RAM space used must be initialized in words when accessing data, and in words when executing instructions from SRAMH space, the area of RAM space used +3 words must be initialized in words.
- In cases where RAM ECC checksum errors are allowed to generate NMI interrupts and resets, the RAM space used must be initialized in words when accessing data.

8.2.4 SRAM checksum protection register (SRAM_CKPR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Rev															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Rev								CKPRKW[6:0]						CKPRC	

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Read "1", write "0" when writing	R/W
b7~b1	CKPRKW[6:0]	Write keycode	When writing to the current register, write "3b" to these bits to enable the current register.	R/W
b0	CKPRC	SRAM checksum control register write enable	0: SRAM checksum control register write disable 1: SRAM checksum control register write enable	R/W

CKPRC: Control the write of SRAMCKCR register. When CKPRC is set to 1, write operations to SRAMCKCR are allowed, if it is set to 0, write operations to SRAMCKCR cannot be performed. When this bit is written, 0x3B must be written to CKPRKW[6:0] at the same time.

8.2.5 SRAM Calibration Status Register (SRAM_CCSR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Rev															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Rev										SRAMR_ PYERR	SRAMH PYERR	SRAM12 PYERR	SRAM3_ 2ERR	SRAM3_ 1ERR	

position	Marker	Place Name	Function	Reading and writing
b31~b5	Reserved	-	Read '1', write '0' when writing	R/W
b4	SRAMR_PYERR	Ret_SRAM	0: No parity error occurs	R/W
		Parity Error Flag	1: There are parity errors occurring	(Note 1)
b3	SRAMH_PYERR	SRAMH Parity	0: No parity error occurs	R/W
		Checksum error flag	1: There are parity errors occurring	(Note 1)
b2	SRAM12_PYERR	SRAM1 and SRAM2 parity	0: No parity error occurs	R/W
		Checksum error flag	1: There are parity errors occurring	(Note 1)
b1	SRAM3_2ERR	ECC 2-bit error flag occurred in SRAM3	0: No 2-bit ECC errors occur 1: A 2-bit ECC error occurred	R/W (Note 1)
b0	SRAM3_1ERR	ECC 1-bit error flag occurred in SRAM3	0: No 1-bit ECC error occurs 1: A 1-bit ECC error occurred	R/W (Note 1)

Caution:

-- Write 1 Clear 0.

9 General Purpose IO (GPIO)

Some abbreviations used in this chapter:

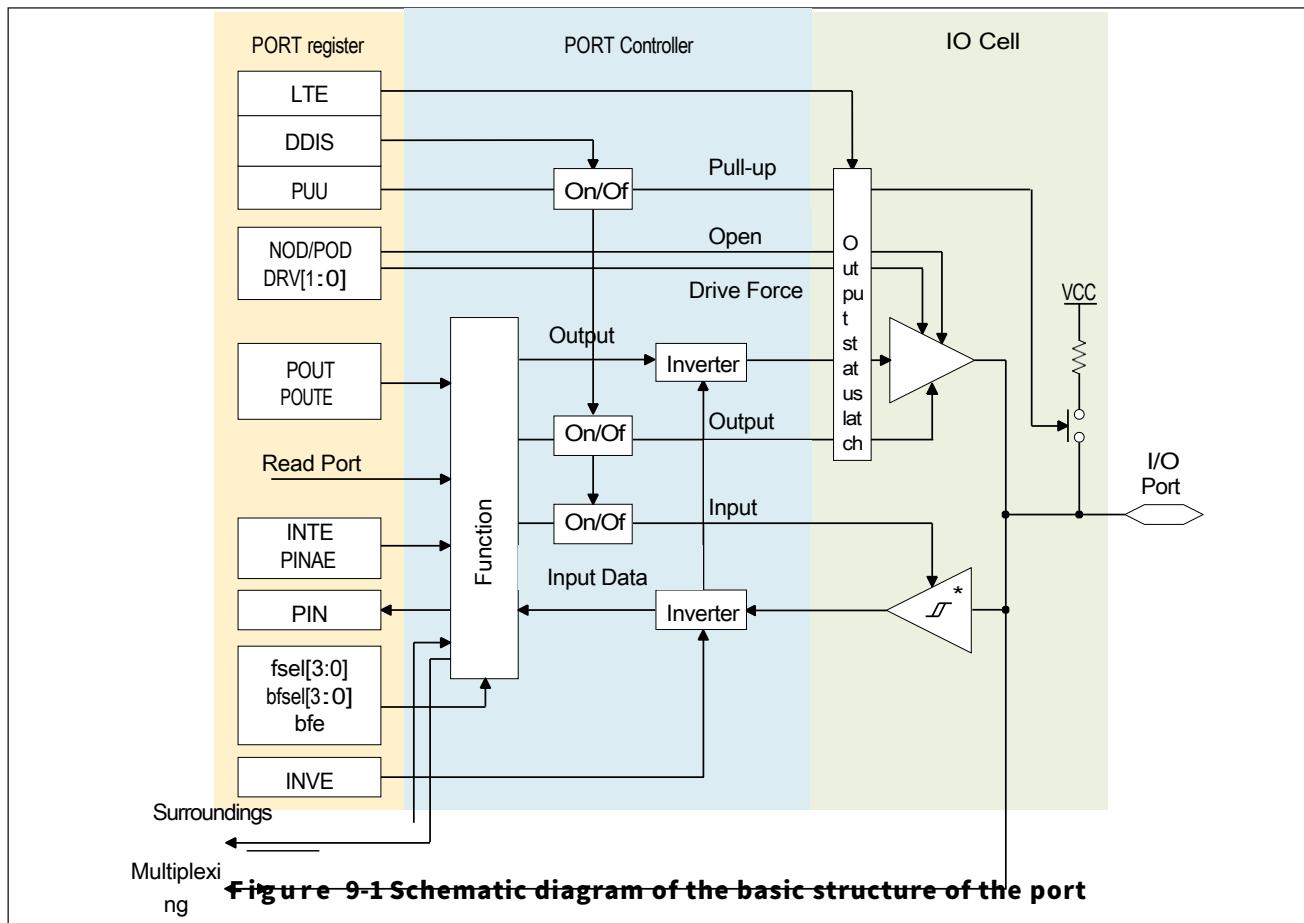
- Px (x=A~E, H) denotes a group of ports, such as PA denotes a group of 16 I/O ports from PA0 to PA15.
- Pxy (x=A~E, H, y=0~15, same as below) indicates a single port, e.g. PB10 port indicates the 10th I/O in PB group.
- GPIO (General Purpose Input **O**utput) General purpose input and output.
- NOD/POD (Nmos/Pmos Open **D**rain) NMOS/PMOS open drain output mode.

9.1 Introduction

Main features:

- 16 pins per group of **ports**, may be less than 16 depending on actual configuration
- Support pull-up
- Support push-pull, open-drain output mode
- Supports high, medium and low drive modes
- Inputs supporting external interrupts
- Support pin peripheral function multiplexing, one I/O pin can have up to 64 selectable multiplexed functions
- Each pin can be programmed independently
- Each pin can be selected to have 2 functions active at the same time (2 output functions active at the same time are not supported)

9.2 Port Function Summary



For detailed GPIO port number, 5V withstand voltage, and drive capability configuration, please refer to the **pin configuration and function** section in the **datasheet**.

9.3 Action Description

9.3.1 General purpose input and output GPIO functions

General purpose input function **GPI**:

Each I/O has a general-purpose input **GPI** function, and the **GPI** function is always active when the digital function disable bit **PCRxy.DDIS** is 0, independent of the **FSEL[5:0]** setting of **PFSRxy** in the function select register. The current port status can be obtained by accessing the Port Input Data Register **PIDRx**. The status of the corresponding single I/O port can also be queried by the PIN bit of the Port Control Register **PCRxy**, and the **PIDRx.PIN[y]** register bit is equivalent to the **PCRxy**.

To reduce power consumption, the I/O's input MOS is turned off when the I/O is not selected for peripheral functions. This mode of operation of the I/O does not allow the input hysteresis feature to function properly because the input MOS switches from the high level of the off state to the high or low (depending on the input value of the I/O) level during each read operation, so the threshold value **VIL** for high to low is normal, but the threshold value **VIH** for low to high does not function properly. It cannot be played. In order for the I/O to function properly with input hysteresis, the MOS needs to be on all the time. This can be achieved by setting register **PINAER.PINAE[x]** to 1, or by setting **PFSRxy** to select a peripheral function (other than GPO).

Due to the delay of I/O input, when the system is running at high speed clock, the input status value may not be read correctly in a single cycle. Set **RDWT[1:0]** to insert a number of wait cycles. Refer to the description of register **PCCR** for details.

General purpose output function **GPO**:

All I/O ports have the general purpose output **GPO** function except for the input-only PB11 port. The **GPO** function can be enabled by setting the port function selection register **PFSRxy.FSEL[5:0]** to 0x0.

When the **GPO** function is active, the output value can be controlled by setting the general output permission register **POERx** to allow or disallow the I/O output, and the general output data register **PODRx** to control the output value. The following three registers can also be used to control the **output value of I/O**: Output Data Clear Register **PORRx**, Output Data Set Register **POSRx**, and Output Data Flip Register **POTRx**. Writing 1 to the corresponding bit in the above registers can make the corresponding I/O output 0, 1, or flip. The I/O output status is not changed when writing 0.

The above registers are all operated together in groups of 16 PORTs. To facilitate control of individual I/Os, the I/O outputs can also be allowed or disabled by setting **PCRxy**. The output value of the I/O can be controlled by setting **PCRxy.POUT**, the **PCRxy_POUT** register

bits are equivalent to PODRx.**POUT[y]**. PCRxy is suitable for controlling a single PORT, and POERx/PODRx is suitable for controlling an entire 16-bit PORT.

After system reset, all ports are initially GP0 (FSEL[5:0]=0x0) and in high resistance state (output disable POUTExy=0) except JTAG multiplex ports PA13, PA14, PA15, PB3, PB4, and sub-oscillator multiplex ports PC14, PC15.

Caution:

-PB11 Port PB11 is multiplexed with MD and is an input-only port with no output function.

9.3.2 Peripheral Features

Up to 64 functions per port can be configured via **FSEL[5:0]** of the function selection register **PFSRxy**. This includes the general-purpose output GPO function corresponding to **FSEL[5:0]=0x0**. Please refer to **the Pin Function Table in the datasheet** for the specific functions configured for each port.

The JTAG/SWD debug function is selected using the register **PSPCR**. **PSPCR.SPFE[z],z=0~4** is 1, the **PFSRxy.FSEL[5:0]** register bits of the corresponding port are invalid, i.e. the SPFE priority is higher than **FSEL**. Initial value of the **PSPCR** register is 0x1F, the JTAG/SWD function is valid. If you want to set these ports to functions other than JTAG/SWD, you need to write 0 to the corresponding **SPFE[z]** bits first.

9.3.3 Dual Peripheral Function

There are some applications where a port needs to be set to two functions at the same time. **FSEL[5:0]**, and then select the second function by setting **PFSRxy.BFE** to 1 and setting the common control register **PCCR**. **BFSEL[3:0]=0x5** and **PFSRxy.BFE=0x1**, then function 2 and function 5 on Pxy will be valid at the same time. It is prohibited to validate 2 output functions on the same port at the same time.

9.3.4 Event Port input and output functions

Event Port1 contains EVNTP100~EVNTP115, **Event Port2** contains EVNTP200~EVNTP215, and so on. trigger other peripheral devices (such as TIMER, ADC, DMA, etc.) to start specific actions. It can also be used as a triggered object to accept events and automatically input or output.

When used as trigger source, set PEVNTRISRm, PEVNTFALRm, PEVNTNFCR to select rising or falling edge detection and digital filtering function, and set function selection register **PFSRxy** to select **EVNTPmn** function. When the selected edge is input from the port, the event **EVENT_PORTm** is generated and output to other peripheral devices to trigger its start action.

When it is the triggered object, set PEVNTTRGSRm to select the trigger event source, and set PEVNTDIRRm to select the output or input function. For the output function, **EVNTPmn** outputs the specified level or flip when the selected event occurs according to the **PEVNTODRm**, **PEVNTORRm**, and **PEVNTOSRm** settings. When the input function is selected, the **EVNTPmn** input status is stored in register **PEVNTIDRx** when the selected event occurs.

To use the **Event Port** function, you need to set the Enable bit of the Autorun System AOS function in the Function Clock Control 0 register (**PWC_FCG0**) to be active first.

Caution:

-Port PB11 is multiplexed with MD and is a dedicated input port, so EVNTP211 has no output function.

9.3.5 External interrupt EIRQ input function

Each I/O port has an external interrupt input function. When the PCRxy.INTE bit is set to 1, this I/O will be allowed as an external interrupt source EIRQy input. More than one I/O can be configured per EIRQy; do not allow multiple I/Os per EIRQy at the same time when in use; the EIRQy input function can be active at the same time as the peripheral functions selected by PFSRxy.FSEL, including GPIO.

In addition, the external non-maskable interrupt **NMI is** multiplexed with the PB11/MD port.

When the I/O port is used as an external interrupt EIRQ, it is necessary to set the filter, interrupt trigger edge, interrupt number, etc. in conjunction with the interrupt controller INTC. Please refer to [Interrupt Controller (INTC) Details].

9.3.6 Simulation function

Some ports have analog input and output functions (including primary and secondary oscillators). When used as analog function, write 1 to register PCRxy.DDIS to disable the digital function of the current port.

9.3.7 Universal control

1. Pull-up/down resistors

Each has an internal pull-up resistor. The PFSR_{xy}.PUU bit can be set to allow this function, and the internal weak 1 state when there is no input to the function is automatically disabled when the I/O port is in the output state.

When the I2Cx_SCL/I2Cx_SDA function is selected for the port, the internal pull-up function is forced to be disabled, ignoring the setting of register PUU.

PA11, PA12 are multiplexed with the USBFS_DM, USBFS_DP pins, have a built-in pull-down resistor of about 400KΩ, and are always active.

2. Drive capacity control

Each I/O port has adjustable high, medium and low drive capability, and the register PFRS_{xy}.DRV[1:0] can be set as required. This function is valid only when the port is in output state.

3. Open-drain output mode

NOD bit to set the I/O port to NMOS open-drain output mode. When NOD is active, the corresponding port can output 0 normally, while the port will be in high resistance state when outputting 1.

When the I2Cx_SCL/I2Cx_SDA function is selected for the I/O port, the open-drain output mode is forced to be active regardless of the register NOD setting.

The above described generic control functions, if not specified, are independent of the function specifically selected for the port, i.e., the FSEL[5:0] setting.

9.4 Register Description

BASE_ADDR : 0x4005_3800

Table 9-1 PORT Register List 1

Register Name	Sym bols	Offset Address	Bit width	Reset value
General purpose input data register	PIDRx	0x00+0x10*n *1	16/32	0xFFFF
General Output Data Register	PODRx	0x04+0x10*n	16/32	0x0000
General Output License Register	POERx	0x06+0x10*n	16/32	0x0000
General Output Reset Register	POSRx	0x08+0x10*n	16/32	0x0000
General purpose output reset register	PORRx	0x0A+0x10*n	16/32	0x0000
General purpose output flip-flop register	POTRx	0x0C+0x10*n	16/32	0x0000
Special control register	PSPCR	0x3F4	16/32	0x001F
Public control register	PCCR	0x3F8	16/32	0x4000
Input control register	PINAER	0x3FA	16/32	0x0000
Write protect register	PWPR	0x3FC	16/32	0x0000
General Control Register	PCRxy	0x400+0x40*n+0x4*y	16/32	0x0X00 *2
Function selection register	PFSRxy	0x402+0x40*n+0x4*y	16/32	0x0000

Note *1: The address calculation formula x=A~E, H corresponds to n=0~4,5

*2: 32K sub oscillator multiplexed port PCRC14, PCRC15 reset value is 0x8100.

BASE_ADDR: 0x4001_0800

Table 9-2 PORT Register List 2

Register Name	Sym bols	Offset Address	Bit width	Reset value
Event Port Direction Selection Register	PEVNTDIRm	0x100+0x1C*(m-1)	32	0x0000_0000
Event Port Input Data Register	PEVNTIDRm	0x104+0x1C*(m-1)	32	0x0000_0000
Event Port Output Data Register	PEVNTODRm	0x108+0x1C*(m-1)	32	0x0000_0000
Event Port Output Data Reset Register	PEVNTORRm	0x10C+0x1C*(m-1)	32	0x0000_0000
Event Port Output Data Reset Register	PEVNTOSRm	0x110+0x1C*(m-1)	32	0x0000_0000
Event Port Rising Edge Input	PEVNTRISRm	0x114+0x1C*(m-1)	32	0x0000_0000

Permission Register				
Event Port Falling Edge Input Permission Register	PEVNTFALRm	0x118+0x1C*(m-1)	32	0x0000_0000
Event Port Input Filter Control Register	PEVNTNFCR	0x170	32	0x0000_0000

Note: m=1~4

9.4.1 General Input Register (PIDRx)

Reset value: 0xFFFF

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PIN[15:0]															

position	Marker	Place Name	Function	Reading and writing
b15~b0	PIN[15:0]	Input Status	0: I/O port input status is low 1: I/O port input status is high	R

This register is a read-only register, and write is invalid. When the digital function is not disabled DDIS=0, the input status of the port can be obtained by reading this register, independent of the PFSRxy.FSEL[5:0] setting value of the function selection register. There is no indeterminate readout value of the corresponding bit of the port. In the digital function disable state DDIS=1 of the port, the corresponding PIN bit readout value is a fixed value 0x1 because the I/O input MOS is in the off state.

9.4.2 General Output Data Register (PODRx)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
POUT[15:0]															

position	Marker	Place Name	Function	Reading and writing
b15~b0	POUT[15:0]	Output Data	0: Output low level 1: Output high level	R/W

When the I/O port is set to GPO function, overwriting this register can change the output status of the corresponding port.

9.4.3 General Output License Register (POERx)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
POUTE[15:0]															

position	Marker	Place Name	Function	Reading and writing
b15~b0	POUTE[15:0]	Output License	0: Output disabled 1: Output licensing	R/W

When the I/O port is set to GPO function and this register is set to 1, the PODRx setting will be output to the corresponding I/O port. When this register is set to 0, the output is off and the port is high resistance.

Do not write 1 if there is no port corresponding bit.

9.4.4 Universal Output Placement Register (POSRx)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
POS[15:0]															

position	Marker	Place Name	Function	Reading and writing
b15~b0	POS[15:0]	High output	0: corresponding to PODRx.POUT no change 1: Corresponding to PODRx.POUT set to 1	R/W

The readout value of this register is always **0x000032bit** access, when POR[y] and POS[y] of the same ~~are~~written 1 at the same time, POR[y] has higher priority, i.e., the corresponding POUT[y] is cleared to zero.

9.4.5 General Output Reset Register (PORRx)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
POR[15:0]															

position	Marker	Place Name	Function	Reading and writing
b15~b0	POR[15:0]	Low output	0: corresponding to PODRx.POUT no change 1: Corresponding to PODRx.POUT clear	R/W

The readout value of this register is always 0x0000.

9.4.6 General-purpose Output Rollover Register (POTRx)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
POT[15:0]															

position	Marker	Place Name	Function	Reading and writing
b15~b0	POT[15:0]	Output Flip	0: corresponding to PODRx.POUT no change 1: corresponding to PODRx.POUT is reversed	R/W

The readout value of this register is always 0x0000.

9.4.7 Special Control Register (PSPCR)

Reset value: 0x001F

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	SPFE[4:0]			

position	Marker	Place Name	Function	Reading and writing
b15~b6	Reserved	–	0 when reading, please write 0 when writing	R
b4	SPFE [4]	Special function selection	0: NJTRST function is invalid 1 : NJTRST function is effective	R/W
b3	SPFE [3]	Special function selection	0: JTDI function is invalid 1: JTDI function is effective	R/W
b2	SPFE [2]	Special function selection	0: JTDO_TRACESWO function is invalid 1: JTDO_TRACESWO function is effective	R/W
b1	SPFE[1]	Special function selection	0: JTMS_SWDIO function is invalid 1: JTMS_SWDIO function is valid	R/W
b0	SPFE[0]	Special function selection	0: JTCK_SWCLK function is invalid 1: JTCK_SWCLK function is valid	R/W

Caution:

-- SPFE[4:0] function select bits have higher priority than PFSRxy.FSEL[5:0] function select bits.

9.4.8 Public Control Register (PCCR)

Reset value: 0x4000

position	Marker	Place Name	Function	Reading and writing
b15-b14	RDWT[1:0]	Read Port Waiting	Set the number of wait cycles to be inserted when reading registers PIDRx, PCRxy	R/W
			Set value	waiting period
			operating frequency	recommended
			00No Wait	~42MHz
			01 (default)	1 cycle
			10	42~84MHz
			11	2 cycles
				84~126MHz
				126~200MHz
b13~b4	Reserved	-	0 when reading, please write 0 when writing	R
b3~b0	BFSEL [3:0]	Sub-function selection	For the functional configuration of each port, please refer to the pinout menu in the datasheet	R/W

9.4.9 Input control register (PINAER)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0											
-	-	-	-	-	-	-	-	-	-	-	-	PINAЕ[5:0]														

position	Marker	Place Name	Function	Reading and writing
b15~b6	Reserved	-	0 when reading, write 0 when writing	R
b5~b0	PINAЕ[5:0]	Input normally open	0: Input MOS normally open is invalid 1: Input MOS is normally open PINAE[0] controls PA0~PA15, PINAE[1] controls PB0~PB15, PINAE[2] controls PC0~PC15, PINAE[3] controls PD0~PD15, PINAE[4] controls PE0~PE15 PINAE [5] controls PH0~PH2	R/W

9.4.10 Write Protect Register (PWPR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
WP[7:0]								-	-	-	-	-	-	-	WE

position	Marker	Place Name	Function	Reading and writing
b15~b8	WP[7:0]	Write protection code	0x00 when read out When b15~b8 write value is 0xA5, b0 value is written to WE when writing value other than 0xA5, WE is automatically cleared to zero	W
b7~b1	Reserved	-	0 when reading, please write 0 when writing	R
b0	WE	Writing permission	0: PSPCR, PCCR, PINAER, PCRxy, PFSRxy registers write disable 1: PSPCR, PCCR, PINAER, PCRxy, PFSRxy register write permission	R/W

9.4.11 General Control Register (PCRxy)

Reset value: b0000_000x_0000_0000 *1

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DDIS	LTE	-	INTE	-	-	INVE	PIN	-	PUU	DRV[1:0]	-	NOD	POUTE	POUT	

position	Marker	Place Name	Function	Reading and writing
b15	DDIS	Digital function prohibition	0: Digital function valid 1: Digital function prohibition	R/W
b14	LTE	Output state latching	0: Output latching is invalid 1: Output latching is valid	R/W
b13	Reserved	-	0 when reading, please write 0 when writing	R
b12	INTE	External Interrupt License	0: External interrupt input disable 1: External interrupt input license	R/W
b11~b10	Reserved	-	0 when reading, please write 0 when writing	R
b9	INVE	Inverse License	0: Input and output data are not inverted 1: Inversion of input and output data	R/W
b8	PIN	Input Status	0: I/O port input status is low 1: I/O port input status is high Same function as PIN[y] in register PIDRx	R
b7	Reserved	-	0 when reading, write 0 when writing	R
b6	PUU	Pull-up permission	0: Internal pullup resistor is invalid 1: Internal pullup resistor is valid	R/W
b5~b4	DRV[1:0]	Drive mode selection	b00: Low drive mode b01: Medium drive mode b1*: High drive mode	R/W
b3	Reserved	-	0 when reading, please write 0 when writing	R
b2	NOD	NMOS Open Leak	0: Normal CMOS output mode 1: NMOS open-drain output	R/W
b1	POUTE	Output License	0: Output disabled 1: Output licensing Same function as POUTE[y] in register POERx	R/W
b0	POUT	Output Data	0: Output low level 1: Output high level Same function as POUT[y] in register PODRx	R/W

When DDIS is set to 1, all digital functions of the corresponding port are forcibly disabled, including general-purpose inputs and outputs, peripheral digital inputs and outputs, pull-up/pull-down functions, and external interrupt input functions. When the port is used as an analog input, set the DDIS bit to 1.

When LTE is set to 1, the current output state of the port is held until LTE is written to 0. This function is mainly used when the port function is switched. In order to avoid the system malfunction caused by unexpected burr of port output during function switching, before function switching, write 1 to LTE to latch the output state of the port, then rewrite the register to select the register to switch the function,

and finally write 0 to LTE to unlatch and update the port state to the new function.

When INVE is set to 1, the input and output data of the port are inverted, including GPIO functions, and other peripheral input and output functions.

*1: The reset value of the following port general control register PCR is not b0000_000x_0000_0000, please note. XTAL32_IN, XTAL32_OUT The reset value of PCRC14,PCRC15 registers of the multiplexed ports PC14, PC15 is 0x8100.

9.4.12 Function Selection Register (PFSRxy)

Reset value: 0x0000 *1

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	BFE	-	-			FSEL [5:0]			

position	Marker	Place Name	Function	Reading and writing
b15~b9	Reserved	-	0 when reading, please write 0 when writing	R
b8	BFE	Sub-functional licensing	Control whether the sub-function selected by PCCR.BFSEL[3:0] is valid 0: Sub-function disabled 1: The sub-function is effective	R/W
b7~b6	Reserved	-	0 when reading, please write 0 when writing	R
b5~b0	FSEL [5:0]	Function Selection	For the functional configuration of each port, please refer to the pinout menu in the datasheet	R/W

Each I/O port can select one of the multiple functions configured on that port via FSEL[5:0]. Referring to the pin menu in the datasheet FSEL[5:0] set to b000000 means Func0 is selected, set to b000001 means Func1 is selected, and so on, set to b001111 means Func15 is selected. where Func0 corresponds to the general purpose output function GPO.

Caution:

- The initial state of PA13, PA14, PA15, PB3, PB4 ports is JTAG/SWD function after reset, when configuring FSEL[5:0] to select function, you need to write 0 to the corresponding bits of register PSPCR to invalidate JTAG/SWD function. The initial state of PC14 and PC15 after reset is digital function disable state, when selecting digital function, you need to write 0 to the DDIS bit of the corresponding register PCRxy first.

9.4.13 Event Port Direction Selection Register (PEVNTDIRRm)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PDIR[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	0 when reading, please write 0 when writing	R
b15~b0	PDIR15:0]	Direction Selection	0: Event Port is the input function 1: Event Port is the output function	R/W

Caution:

-EVNTP211 function without output function (configured on PB11 port)

9.4.14 Event Port Input Data Register (PEVNTIDRm)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PIN[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	0 when reading, write 0 when writing	R
b15~b0	PIN[15:0]	Port Input Status	0: Event Port input state is low when the event is triggered 1: Event Port input status is high when the event is triggered	R

When the direction of Event Port is set to input status, the input status of the corresponding port is saved to this register when the set event is triggered.

9.4.15 Event Port Output Data Register (PEVNTODRm)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
POUT[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	–	0 when reading, write 0 when writing	R
b15~b0	POUT[15:0]	Port Output Value	0: Event Port output low level 1: Event Port output high level	R/W

When the direction of **Event Port** is set to output state, write this register, the initial output value of **Event Port** before the set event triggers. When the selected event triggers, the corresponding bit of PEVNTODRm.POUT is cleared 0, set 1, or flipped according to the PEVNTORRm, PEVNTOSRm set value, and output to EVNTPmn port at the same time.

9.4.16 Event Port Output Data Reset Register (PEVNTORRm)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
POR[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	–	0 when reading, please write 0 when writing	R
b15~b0	POR[15:0]	Output value reset	0: No change in PEVNTODRm.POUT when the event is triggered 1: Event triggered when corresponding to PEVNTODRm.POUT reset	W

When PEVNTORRm.POR and PEVNTm.POS are both set to 1, the event is triggered with the corresponding PEVNTODRm.POUT flipped.

9.4.17 Event Port Output Data Reset Register (PEVNTOSRm)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
POS[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	–	0 when reading, please write 0 when writing	R
b15~b0	POS[15:0]	Output value set	0: No change in PEVNTODRm . POUT when the event is triggered 1: PEVNTODRm . POUT is set when the event is triggered	W

When PEVNTORRm.POR and PEVNTm.POS are both set to 1, the event is triggered with the corresponding PEVNTODRm.POUT flipped.

9.4.18 Event Port rising edge input license register (PEVNTRISRm)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RIS [15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	–	0 when reading, write 0 when writing	R
b15~b0	RIS [15:0]	Up along the detection permit	0: EVNTPmn rising edge event detection is invalid 1: EVNTPmn rising edge event detection is valid PEVNTRISRm . RIS[n] corresponds to EVNTPmn	R/W

The Event Port is used as the event source and outputs an event when the RIS bit is set to 1, corresponding to the rising edge of the input of EVNTPmn, which is used to trigger other peripheral modules. the edge events of EVNTPm0~15 are combined into one event EVENT_PORTm output, where any port detects an edge and outputs the event EVENT_PORTm.

9.4.19 Event Port Falling Edge Input License Register (PEVNTFALRm)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FAL[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	—	0 when reading, write 0 when writing	R
b15~b0	FAL[15:0]	Lowering along the detection permit	0: EVNTPmn falling edge event detection is invalid 1: EVNTPmn falling edge event detection is valid PEVNTRISRm.FAL[n] corresponds to EVNTPmn	R/W

The Event Port is used as the event source, and when the FAL bit is set to 1, it outputs an event corresponding to the falling edge of the input of EVNTP, which is used to trigger other peripheral modules. the edge events of EVNTPm0~15 are combined into one event EVENT_PORTm output, where any port will output the event EVENT_PORTm after detecting an edge.

9.4.20 Event Port Input Filter Control Register (PEVNTNFCR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved				DIVS4[1:0]		NFEN4	Reserved				DIVS3[1:0]		NFEN3		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				DIVS2[1:0]		NFEN2	Reserved				DIVS1[1:0]		NFEN1		

position	Marker	Place Name	Function	Reading and writing
b31~b27	Reserved	–	0 when reading, write 0 when writing	R
b26-b25	DIVS4[1:0]	Digital filtering sampling clock selection	Event Port4 Digital Filter Sample Clock Selection 00: PCLK1 01: PCLK1/8 10: PCLK1/32 11: PCLK1/64	R/W
b24	NFEN4	Digital filtering license	0: Event Port4 digital filtering is invalid 1: Event Port4 digital filtering is valid	R/W
b23~b19	Reserved	–	0 when reading, please write 0 when writing	R
b18-b17	DIVS3[1:0]	Digital filtering sampling clock selection	Event Port3 Digital Filter Sample Clock Selection 00: PCLK1 01: PCLK1/8 10: PCLK1/32 11: PCLK1/64	R/W
b16	NFEN3	Digital filtering license	0: Event Port3 digital filtering is invalid 1: Event Port3 digital filtering is valid	R/W
b15~b11	Reserved	–	0 when reading, write 0 when writing	R
b10-b9	DIVS2[1:0]	Digital filtering sampling sampling clock selection	Event Port2 digital filter sampling clock selection 00: PCLK1 01: PCLK1/8 10: PCLK1/32 11: PCLK1/64	R/W
b8	NFEN2	Digital filtering license	0: Event Port2 digital filtering is invalid 1: Event Port2 digital filtering is valid	R/W
b7~b3	Reserved	–	0 when reading, write 0 when writing	R
b2-b1	DIVS1[1:0]	Digital filtering sampling sampling clock selection	Event Port1 digital filter sampling clock selection 00: PCLK1 01: PCLK1/8 10: PCLK1/32 11: PCLK1/64	R/W
b0	NFEN1	Digital filtering license	0: Event Port1 digital filtering is invalid 1: Event Port1 digital filtering is valid	R/W

filtering is valid

9.4.21 32bit Access

Among the registers mentioned above, except for the Event combined in the following way for 32bit access:

Table 9-3 List of PORT registers for 32bit access

Address	b31 ~	b16 ~	b15 ~	b0
0x4005_3800+0x10*n *1	Reserved		PIDRx	
0x4005_3804+0x10*n	POERx		PODRx	
0x4005_3808+0x10*n	PORRx		POSRx	
0x4005_380C+0x10*n	Reserved		POTRx	
0x4005_3BF4	Reserved		PSPCR	
0x4005_3BF8	PINAER		PCCR	
0x4005_3BFC	Reserved		PWPR	
0x4005_3C00+0x40*n+0x04*y	PFCSRxy		PCRxy	

Note *1: The address calculation formula $x=A\sim E, H$ corresponds to $n=0\sim 4,5$

9.5 Cautions

Please do not set the same function to more than one port.

When using the analog function, please turn off the digital function of the corresponding port (DDIS=1).

Please switch the port function when the output latch is active (LTE=1) to avoid output burrs on the port other than those expected during the switching period.

10 Interrupt Controller (INTC)

10.1 Introduction

The functions of the Interrupt Controller (INTC) are selection of interrupt event request as interrupt input to NVIC to wake up WFI, selection of interrupt event request as event input to wake up WFE, selection of interrupt event request as wake up condition for low power modes (sleep mode and stop mode), interrupt control function for external pins NMI and EIRQ, and interrupt/event selection function for software interrupts.

Main specifications:

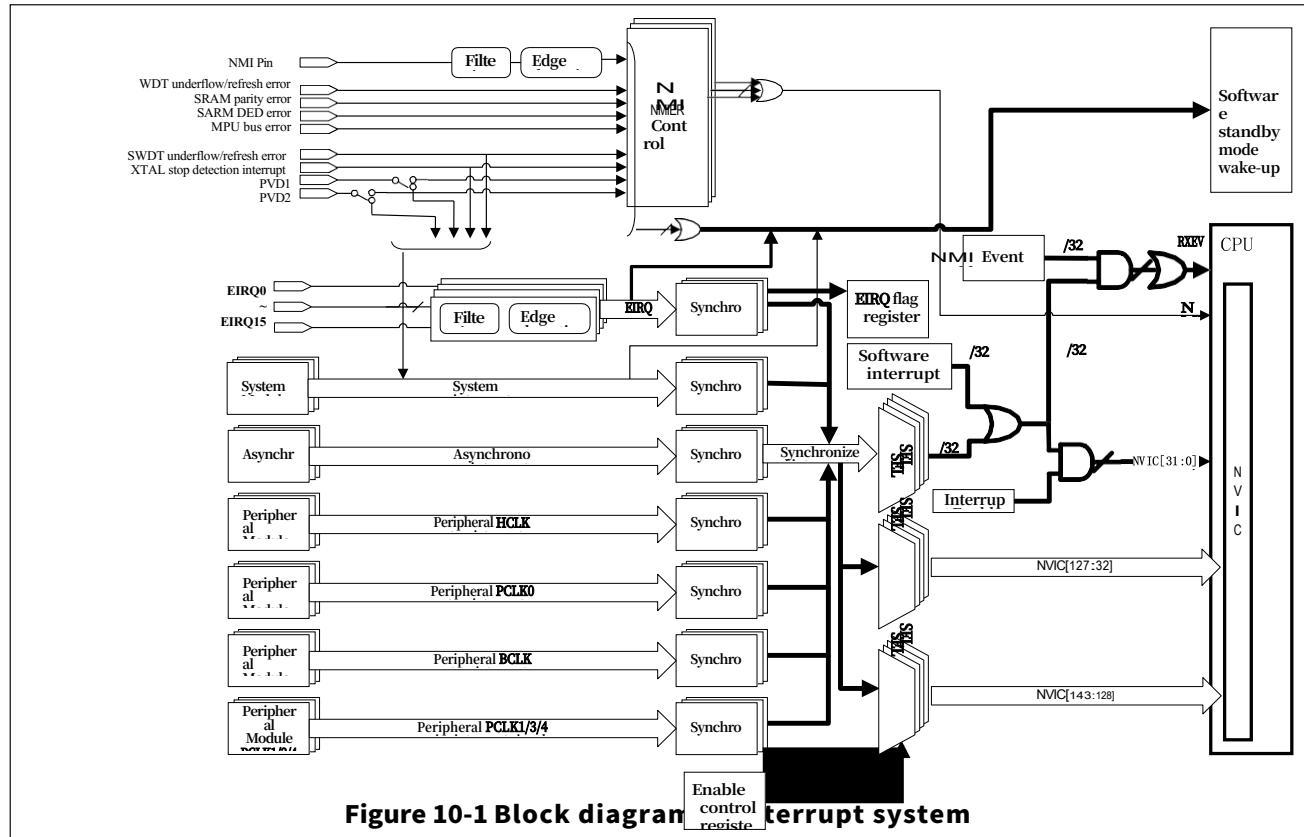
- 1) NVIC interrupt vectors: Refer to 10.3.1 Interrupt Vector Table for the actual number of interrupt vectors used (excluding the 16 interrupt lines of the Cortex™-M4F). Each interrupt vector can be selected according to the interrupt select register for the corresponding peripheral interrupt event request. For more instructions on exceptions and NVIC programming, please refer to Chapter 5: Exceptions and Chapter 8: Nested Vector Interrupt Controllers in the ARM Cortex™-M4F Technical Reference Manual.
- 2) Programmable priority: 16 programmable priority levels (4-bit interrupt priority register is used)
- 3) Non-maskable interrupts: In addition to the **NMI** pin as a non-maskable interrupt source, multiple system interrupt event requests can be independently selected as non-maskable interrupts, and each interrupt event request is equipped with independent enable selection, flag and flag clear registers.
- 4) Equipped with 16 external pin interrupts.
- 5) Configure multiple peripheral interrupt event requests, refer to 10.3.2 Interrupt Event Request Sequence Number for details.
- 6) Equipped with 32 software interrupt event requests.
- 7) The interrupt can wake up the system in sleep mode and stop mode.

Input pins:

Foot er Name	I/O	Des crip tion
NMI	Input	Non-maskable interrupt request pin
EIRQ0	Input	External pin interrupt event request 0
EIRQ1	Input	External pin interrupt event request 1
EIRQ2	Input	External pin interrupt event request 2
EIRQ3	Input	External pin interrupt event request 3
EIRQ4	Input	External pin interrupt event request 4
EIRQ5	Input	External pin interrupt event request 5
EIRQ6	Input	External pin interrupt event request 6
EIRQ7	Input	External pin interrupt event request 7
EIRQ8	Input	External pin interrupt event request 8
EIRQ9	Input	External pin interrupt event request 9
EIRQ10	Input	External pin interrupt event request 10
EIRQ11	Input	External pin interrupt event request 11
EIRQ12	Input	External pin interrupt event request 12
EIRQ13	Input	External pin interrupt event request 13
EIRQ14	Input	External pin interrupt event request 14
EIRQ15	Input	External pin interrupt event request 15

10.2 INTC System Block Diagram

10.2.1 System Block Diagram



10.3 vector table

10.3.1 Interrupt vector table

Table 10-1 Interrupt vector table

Address	vector Serial number	IRQ Serial number	Interrupt source	Description
ARM core interrupt handling vectors				
0x0000_0000	0	–	ARM core	Initial stack pointer
0x0000_0004	1	–	ARM core	Initial Program Counter
0x0000_0008	2	–	ARM core	Non-maskable Interrupt (NMI)
0x0000_000C	3	–	ARM core	Hard Fault
0x0000_0010	4	–	ARM core	MemManage Fault
0x0000_0014	5	–	ARM core	Bus Fault
0x0000_0018	6	–	ARM core	Usage Fault
0x0000_001C	7	–	ARM core	Reserved
0x0000_0020	8	–	ARM core	Reserved
0x0000_0024	9	–	ARM core	Reserved
0x0000_0028	10	–	ARM core	Reserved
0x0000_002C	11	–	ARM core	Supervisor call (SVCall)
0x0000_0030	12	–	ARM core	Debug Monitor
0x0000_0034	13	–	ARM core	Reserved
0x0000_0038	14	–	ARM core	Pendable request for system Service(PendableSrvReq)
0x0000_003C	15	–	ARM core	System tick timer (SysTick)
Non-ARM core interrupt processing vectors				
0x0000_0040	16	0	INT_SEL0	The interrupt event request/software interrupt selected by register INT_SEL0.
0x0000_0044	17	1	INT_SEL1	The interrupt event request/software interrupt selected by register INT_SEL1.
0x0000_0048	18	2	INT_SEL2	The interrupt event request/software interrupt selected by register INT_SEL2.
0x0000_004C	19	3	INT_SEL3	The interrupt event request/software interrupt selected by register INT_SEL3.
0x0000_0050	20	4	INT_SEL4	The interrupt event request/software interrupt selected by register INT_SEL4.
0x0000_0054	21	5	INT_SEL5	The interrupt event request/software interrupt selected by register INT_SEL5.
0x0000_0058	22	6	INT_SEL6	The interrupt event request/software interrupt selected by register INT_SEL6.

0x0000_005C	23	7	INT_SEL7	The interrupt event request/software interrupt selected by register INT_SEL7.
0x0000_0060	24	8	INT_SEL8	Interrupt event request/software interrupt selected by register INT_SEL8.
0x0000_0064	25	9	INT_SEL9	The interrupt event request/software interrupt selected by register INT_SEL9.
0x0000_0068	26	10	INT_SEL10	The interrupt event request/software interrupt selected by register INT_SEL10.
0x0000_006C	27	11	INT_SEL11	The interrupt event request/software interrupt selected by register INT_SEL11.
0x0000_0070	28	12	INT_SEL12	The interrupt event request/software interrupt selected by register INT_SEL12.

Address	vector Serial number	IRQ Serial number	Interrupt source	Description
0x0000_0074	29	13	INT_SEL13	The interrupt event request/software interrupt selected by register INT_SEL13.
0x0000_0078	30	14	INT_SEL14	The interrupt event request/software interrupt selected by register INT_SEL14.
0x0000_007C	31	15	INT_SEL15	The interrupt event request/software interrupt selected by register INT_SEL15.
0x0000_0080	32	16	INT_SEL16	The interrupt event request/software interrupt selected by register INT_SEL16.
0x0000_0084	33	17	INT_SEL17	Interrupt event request/software interrupt selected by register INT_SEL17.
0x0000_0088	34	18	INT_SEL18	The interrupt event request/software interrupt selected by register INT_SEL18.
0x0000_008C	35	19	INT_SEL19	Interrupt event request/software interrupt selected by register INT_SEL19.
0x0000_0090	36	20	INT_SEL20	Interrupt event request/software interrupt selected by register INT_SEL20.
0x0000_0094	37	21	INT_SEL21	The interrupt event request/software interrupt selected by register INT_SEL21.
0x0000_0098	38	22	INT_SEL22	Interrupt event request/software interrupt selected by register INT_SEL22.
0x0000_009C	39	23	INT_SEL23	The interrupt event request/software interrupt selected by register INT_SEL23.
0x0000_00A0	40	24	INT_SEL24	The interrupt event request/software interrupt selected by register INT_SEL24.
0x0000_00A4	41	25	INT_SEL25	The interrupt event request/software interrupt selected by register INT_SEL25.
0x0000_00A8	42	26	INT_SEL26	Interrupt event request/software interrupt selected by register INT_SEL26.
0x0000_00AC	43	27	INT_SEL27	The interrupt event request/software interrupt selected by register INT_SEL27.
0x0000_00B0	44	28	INT_SEL28	Interrupt event request/software interrupt selected by register INT_SEL28.
0x0000_00B4	45	29	INT_SEL29	Interrupt event request/software interrupt selected by register INT_SEL29.
0x0000_00B8	46	30	INT_SEL30	The interrupt event request/software interrupt selected by register INT_SEL30.
0x0000_00BC	47	31	INT_SEL31	The interrupt event request/software interrupt selected by register INT_SEL31.
0x0000_00C0	48	32	INT_SEL32	The interrupt event request selected by register INT_SEL32.
0x0000_00C4	49	33	INT_SEL33	The interrupt event request selected by register INT_SEL33.
0x0000_00C8	50	34	INT_SEL34	The interrupt event request selected by register INT_SEL34.

0x0000_00CC	51	35	INT_SEL35	The interrupt event request selected by register INT_SEL35.
0x0000_00D0	52	36	INT_SEL36	The interrupt event request selected by register INT_SEL36.
0x0000_00D4	53	37	INT_SEL37	The interrupt event request selected by register INT_SEL37.
0x0000_00D8	54	38	INT_SEL38	The interrupt event request selected by register INT_SEL38.
0x0000_00DC	55	39	INT_SEL39	The interrupt event request selected by register INT_SEL39.
0x0000_00E0	56	40	INT_SEL40	The interrupt event request selected by register INT_SEL40.
0x0000_00E4	57	41	INT_SEL41	The interrupt event request selected by register INT_SEL41.
0x0000_00E8	58	42	INT_SEL42	The interrupt event request selected by register INT_SEL42.
0x0000_00EC	59	43	INT_SEL43	The interrupt event request selected by register INT_SEL43.
0x0000_00F0	60	44	INT_SEL44	The interrupt event request selected by register INT_SEL44.
0x0000_00F4	61	45	INT_SEL45	The interrupt event request selected by register INT_SEL45.
0x0000_00F8	62	46	INT_SEL46	The interrupt event request selected by register INT_SEL46.
0x0000_00FC	63	47	INT_SEL47	The interrupt event request selected by register INT_SEL47.
0x0000_0100	64	48	INT_SEL48	The interrupt event request selected by register INT_SEL48.
0x0000_0104	65	49	INT_SEL49	The interrupt event request selected by register INT_SEL49.

Address	vector Serial number	IRQ Serial number	Interrupt source	Des crip tion
0x0000_0108	66	50	INT_SEL50	The interrupt event request selected by register INT_SEL50.
0x0000_010C	67	51	INT_SEL51	The interrupt event request selected by register INT_SEL51.
0x0000_0110	68	52	INT_SEL52	The interrupt event request selected by register INT_SEL52.
0x0000_0114	69	53	INT_SEL53	The interrupt event request selected by register INT_SEL53.
0x0000_0118	70	54	INT_SEL54	The interrupt event request selected by register INT_SEL54.
0x0000_011C	71	55	INT_SEL55	The interrupt event request selected by register INT_SEL55.
0x0000_0120	72	56	INT_SEL56	The interrupt event request selected by register INT_SEL56.
0x0000_0124	73	57	INT_SEL57	The interrupt event request selected by register INT_SEL57.
0x0000_0128	74	58	INT_SEL58	The interrupt event request selected by register INT_SEL58.
0x0000_012C	75	59	INT_SEL59	The interrupt event request selected by register INT_SEL59.
0x0000_0130	76	60	INT_SEL60	The interrupt event request selected by register INT_SEL60.
0x0000_0134	77	61	INT_SEL61	The interrupt event request selected by register INT_SEL61.
0x0000_0138	78	62	INT_SEL62	The interrupt event request selected by register INT_SEL62.
0x0000_013C	79	63	INT_SEL63	The interrupt event request selected by register INT_SEL63.
0x0000_0140	80	64	INT_SEL64	The interrupt event request selected by register INT_SEL64.
0x0000_0144	81	65	INT_SEL65	The interrupt event request selected by register INT_SEL65.
0x0000_0148	82	66	INT_SEL66	The interrupt event request selected by register INT_SEL66.
0x0000_014C	83	67	INT_SEL67	The interrupt event request selected by register INT_SEL67.
0x0000_0150	84	68	INT_SEL68	The interrupt event request selected by register INT_SEL68.
0x0000_0154	85	69	INT_SEL69	The interrupt event request selected by register INT_SEL69.
0x0000_0158	86	70	INT_SEL70	The interrupt event request selected by register INT_SEL70.
0x0000_015C	87	71	INT_SEL71	The interrupt event request selected by register INT_SEL71.
0x0000_0160	88	72	INT_SEL72	The interrupt event request selected by register INT_SEL72.
0x0000_0164	89	73	INT_SEL73	The interrupt event request selected by register INT_SEL73.
0x0000_0168	90	74	INT_SEL74	The interrupt event request selected by register INT_SEL74.
0x0000_016C	91	75	INT_SEL75	The interrupt event request selected by register INT_SEL75.
0x0000_0170	92	76	INT_SEL76	The interrupt event request selected by register INT_SEL76.
0x0000_0174	93	77	INT_SEL77	The interrupt event request selected by register INT_SEL77.
0x0000_0178	94	78	INT_SEL78	The interrupt event request selected by register INT_SEL78.
0x0000_017C	95	79	INT_SEL79	The interrupt event request selected by register INT_SEL79.
0x0000_0180	96	80	INT_SEL80	The interrupt event request selected by register INT_SEL80.
0x0000_0184	97	81	INT_SEL81	The interrupt event request selected by register INT_SEL81.
0x0000_0188	98	82	INT_SEL82	The interrupt event request selected by register INT_SEL82.
0x0000_018C	99	83	INT_SEL83	The interrupt event request selected by register INT_SEL83.
0x0000_0190	100	84	INT_SEL84	The interrupt event request selected by register INT_SEL84.
0x0000_0194	101	85	INT_SEL85	The interrupt event request selected by register INT_SEL85.

0x0000_0198	102	86	INT_SEL86	The interrupt event request selected by register INT_SEL86.
-------------	-----	----	-----------	---

Address	vector Serial number	IRQ Serial number	Interrupt source	Des crip tion
0x0000_019C	103	87	INT_SEL87	The interrupt event request selected by register INT_SEL87.
0x0000_01A0	104	88	INT_SEL88	The interrupt event request selected by register INT_SEL88.
0x0000_01A4	105	89	INT_SEL89	The interrupt event request selected by register INT_SEL89.
0x0000_01A8	106	90	INT_SEL90	The interrupt event request selected by register INT_SEL90.
0x0000_01AC	107	91	INT_SEL91	The interrupt event request selected by register INT_SEL91.
0x0000_01B0	108	92	INT_SEL92	The interrupt event request selected by register INT_SEL92.
0x0000_01B4	109	93	INT_SEL93	The interrupt event request selected by register INT_SEL93.
0x0000_01B8	110	94	INT_SEL94	The interrupt event request selected by register INT_SEL94.
0x0000_01BC	111	95	INT_SEL95	The interrupt event request selected by register INT_SEL95.
0x0000_01C0	112	96	INT_SEL96	The interrupt event request selected by register INT_SEL96.
0x0000_01C4	113	97	INT_SEL97	The interrupt event request selected by register INT_SEL97.
0x0000_01C8	114	98	INT_SEL98	The interrupt event request selected by register INT_SEL98.
0x0000_01CC	115	99	INT_SEL99	The interrupt event request selected by register INT_SEL99.
0x0000_01D0	116	100	INT_SEL100	The interrupt event request selected by register INT_SEL100.
0x0000_01D4	117	101	INT_SEL101	The interrupt event request selected by register INT_SEL101.
0x0000_01D8	118	102	INT_SEL102	The interrupt event request selected by register INT_SEL102.
0x0000_01DC	119	103	INT_SEL103	The interrupt event request selected by register INT_SEL103.
0x0000_01E0	120	104	INT_SEL104	The interrupt event request selected by register INT_SEL104.
0x0000_01E4	121	105	INT_SEL105	The interrupt event request selected by register INT_SEL105.
0x0000_01E8	122	106	INT_SEL106	The interrupt event request selected by register INT_SEL106.
0x0000_01EC	123	107	INT_SEL107	The interrupt event request selected by register INT_SEL107.
0x0000_01F0	124	108	INT_SEL108	The interrupt event request selected by register INT_SEL108.
0x0000_01F4	125	109	INT_SEL109	The interrupt event request selected by register INT_SEL109.
0x0000_01F8	126	110	INT_SEL110	The interrupt event request selected by register INT_SEL110.
0x0000_01FC	127	111	INT_SEL111	The interrupt event request selected by register INT_SEL111.
0x0000_0200	128	112	INT_SEL112	The interrupt event request selected by register INT_SEL112.
0x0000_0204	129	113	INT_SEL113	The interrupt event request selected by register INT_SEL113.
0x0000_0208	130	114	INT_SEL114	The interrupt event request selected by register INT_SEL114.
0x0000_020C	131	115	INT_SEL115	The interrupt event request selected by register INT_SEL115.
0x0000_0210	132	116	INT_SEL116	The interrupt event request selected by register INT_SEL116.
0x0000_0214	133	117	INT_SEL117	The interrupt event request selected by register INT_SEL117.
0x0000_0218	134	118	INT_SEL118	The interrupt event request selected by register INT_SEL118.
0x0000_021C	135	119	INT_SEL119	The interrupt event request selected by register INT_SEL119.
0x0000_0220	136	120	INT_SEL120	The interrupt event request selected by register INT_SEL120.
0x0000_0224	137	121	INT_SEL121	The interrupt event request selected by register INT_SEL121.
0x0000_0228	138	122	INT_SEL122	The interrupt event request selected by register INT_SEL122.

0x0000_022C	139	123	INT_SEL123	The interrupt event request selected by register INT_SEL123.
-------------	-----	-----	------------	--

Address	vector Serial number	IRQ Serial number	Interrupt source	Description
0x0000_0230	140	124	INT_SEL124	The interrupt event request selected by register INT_SEL124.
0x0000_0234	141	125	INT_SEL125	The interrupt event request selected by register INT_SEL125.
0x0000_0238	142	126	INT_SEL126	The interrupt event request selected by register INT_SEL126.
0x0000_023C	143	127	INT_SEL127	The interrupt event request selected by register INT_SEL127.
0x0000_0240	144	128	INT_VSSEL128	Register INT_VSSEL128 shares this vector with the interrupt event request selected by the enable bit.
0x0000_0244	145	129	INT_VSSEL129	Register INT_VSSEL129 shares this vector with interrupt event requests selected by the enable bit.
0x0000_0248	146	130	INT_VSSEL130	This vector is shared by the interrupt event request selected by the enable bit in register INT_VSSEL130.
0x0000_024C	147	131	INT_VSSEL131	Register INT_VSSEL131 shares this vector with the interrupt event request selected by the enable bit.
0x0000_0250	148	132	INT_VSSEL132	Register INT_VSSEL132 shares this vector with the interrupt event request selected by the enable bit.
0x0000_0254	149	133	INT_VSSEL133	Register INT_VSSEL133 shares this vector with the interrupt event request selected by the enable bit.
0x0000_0258	150	134	INT_VSSEL134	Register INT_VSSEL134 shares this vector with the interrupt event request selected by the enable bit.
0x0000_025C	151	135	INT_VSSEL135	Register INT_VSSEL135 shares this vector with the interrupt event request selected by the enable bit.
0x0000_0260	152	136	INT_VSSEL136	Register INT_VSSEL136 shares this vector with interrupt event requests selected by the enable bit.
0x0000_0264	153	137	INT_VSSEL137	Register INT_VSSEL137 shares this vector with interrupt event requests selected by the enable bit.
0x0000_0268	154	138	INT_VSSEL138	Register INT_VSSEL138 shares this vector with the interrupt event request selected by the enable bit.
0x0000_026C	155	139	INT_VSSEL139	Register INT_VSSEL139 shares this vector with interrupt event requests selected by the enable bit.
0x0000_0270	156	140	INT_VSSEL140	Register INT_VSSEL140 shares this vector with the interrupt event request selected by the enable bit.
0x0000_0274	157	141	INT_VSSEL141	Register INT_VSSEL141 shares this vector with the interrupt event request selected by the enable bit.
0x0000_0278	158	142	INT_VSSEL142	Register INT_VSSEL142 shares this vector with the interrupt event request selected by the enable bit.
0x0000_027C	159	143	INT_VSSEL143	Register INT_VSSEL143 shares this vector with the interrupt event request selected by the enable bit.



Caution:

- Refer to the Register Description section for the specific selected interrupt event request number.

10.3.2 Interrupt event request sequence number

Interrupt event requests are generated by the peripheral and are referred to as interrupt sources when the interrupt event request is selected as an input to the NVIC by the interrupt controller; when selected as an event input, it is referred to as an event source. The peripheral interrupt event request can also be used as a condition for MCU low power mode return.

**Table 10-2 Interrupt event
request sequence number
and selection**

Number	Interrupt event request number	Function	Function Name	Whether to select as internal touch interrupt	Can you choose for internal touch Origin	Interrupt selection register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
0	000h	PORT	PORT_EIRQ0	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[0]
1	001h		PORT_EIRQ1	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[1]
2	002h		PORT_EIRQ2	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[2]
3	003h		PORT_EIRQ3	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[3]
4	004h		PORT_EIRQ4	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[4]
5	005h		PORT_EIRQ5	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[5]
6	006h		PORT_EIRQ6	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[6]
7	007h		PORT_EIRQ7	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[7]
8	008h		PORT_EIRQ8	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[8]
9	009h		PORT_EIRQ9	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[9]
10	00Ah		PORT_EIRQ10	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[10]
11	00Bh		PORT_EIRQ11	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[11]
12	00Ch		PORT_EIRQ12	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[12]
13	00Dh		PORT_EIRQ13	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[13]
14	00Eh		PORT_EIRQ14	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[14]
15	00Fh		PORT_EIRQ15	✓	✓	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[15]
16	010h	–	–	–	–	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[16]
17	011h	–	–	–	–	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[17]
18	012h	–	–	–	–	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[18]
19	013h	–	–	–	–	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[19]
20	014h	–	–	–	–	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[20]
21	015h	–	–	–	–	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[21]

22	016h	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[22]
23	017h	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[23]
24	018h	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[24]
25	019h	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[25]
26	01Ah	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[26]
27	01Bh	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[27]
28	01Ch	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[28]
29	01Dh	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[29]
30	01Eh	-	-	-	-	INT_SEL0~31	INT_SEL32~37	int_vsse128[30]

Number	Interrupt event request number	Function	Function Name	Whether to select as internal touch interrupt	Can you choose for internal touch Origin	Interrupt selection register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
31	01Fh	-	-	-	-	INT_SEL0~31	INT_SEL32~37	INT_VSEL128[31]
32	020h	DMA	DMA1_TC0	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSEL129[0]
33	021h		DMA1_TC1	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSEL129[1]
34	022h		DMA1_TC2	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSEL129[2]
35	023h		DMA1_TC3	✓	✓	INT_SEL0~31	INT_SEL38~43	int_vsel129[3]
36	024h		DMA2_TC0	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSEL129[4]
37	025h		DMA2_TC1	✓	✓	INT_SEL0~31	INT_SEL38~43	int_vsel129[5]
38	026h		DMA2_TC2	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSEL129[6]
39	027h		DMA2_TC3	✓	✓	INT_SEL0~31	INT_SEL38~43	int_vsel129[7]
40	028h		DMA1_BT0	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSEL129[8]
41	029h		DMA1_BT1	✓	✓	INT_SEL0~31	INT_SEL38~43	INT_VSEL129[9]
42	02Ah		DMA1_BT2	✓	✓	INT_SEL0~31	INT_SEL38~43	int_vsel129[10]
43	02Bh		DMA1_BT3	✓	✓	INT_SEL0~31	INT_SEL38~43	int_vsel129[11]
44	02Ch		DMA2_BT0	✓	✓	INT_SEL0~31	INT_SEL38~43	int_vsel129[12]
45	02Dh		DMA2_BT1	✓	✓	INT_SEL0~31	INT_SEL38~43	int_vsel129[13]
46	02Eh		DMA2_BT2	✓	✓	INT_SEL0~31	INT_SEL38~43	int_vsel129[14]
47	02Fh		DMA2_BT3	✓	✓	INT_SEL0~31	INT_SEL38~43	int_vsel129[15]
48	030h	DMA	DMA1_ERR	✓	-	INT_SEL0~31	INT_SEL38~43	int_vsel129[16]
49	031h		DMA2_ERR	✓	-	INT_SEL0~31	INT_SEL38~43	int_vsel129[17]
50	032h	EFM	EFM_PEERR	✓	-	INT_SEL0~31	INT_SEL38~43	int_vsel129[18]
51	033h		EFM_COLERR	✓	-	INT_SEL0~31	INT_SEL38~43	int_vsel129[19]
52	034h		EFM_OPTEND	✓	✓	INT_SEL0~31	INT_SEL38~43	int_vsel129[20]
53	035h	USBFS	USBFS_SOF	-	✓	INT_SEL0~31	INT_SEL38~43	int_vsel129[21]
54	036h	QSPI	QSPI_INTR	✓	-	INT_SEL0~31	INT_SEL38~43	int_vsel129[22]
55	037h	DCU	DCU1	✓	✓	INT_SEL0~31	INT_SEL38~43	int_vsel129[23]
56	038h		DCU2	✓	✓	INT_SEL0~31	INT_SEL38~43	int_vsel129[24]
57	039h		DCU3	✓	✓	INT_SEL0~31	INT_SEL38~43	int_vsel129[25]
58	03Ah		DCU4	✓	✓	INT_SEL0~31	INT_SEL38~43	int_vsel129[26]
59	03Bh	-	-	-	-	INT_SEL0~31	INT_SEL38~43	int_vsel129[27]
60	03Ch	-	-	-	-	INT_SEL0~31	INT_SEL38~43	int_vsel129[28]
61	03Dh	-	-	-	-	INT_SEL0~31	INT_SEL38~43	int_vsel129[29]
62	03Eh	-	-	-	-	INT_SEL0~31	INT_SEL38~43	int_vsel129[30]
63	03Fh	-	-	-	-	INT_SEL0~31	INT_SEL38~43	int_vsel129[31]
64	040h	Timer0_1	TMR01_GCMA	✓	✓	INT_SEL0~31	INT_SEL44~49	INT_VSEL130[0]
65	041h		TMR01_GCMB	✓	✓	INT_SEL0~31	INT_SEL44~49	INT_VSEL130[1]
66	042h	Timer0_2	TMR02_GCMA	✓	✓	INT_SEL0~31	INT_SEL44~49	INT_VSEL130[2]
67	043h		TMR02_GCMB	✓	✓	INT_SEL0~31	INT_SEL44~49	INT_VSEL130[3]

68	044h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[4]
69	045h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[5]
70	046h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[6]

Number	Interrupt event request number	Function	Function Name	Whether to select as internal touch interrupt	Can you choose for internal touch Origin	Interrupt selection register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
71	047h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[7]
72	048h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[8]
73	049h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[9]
74	04Ah	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[10]
75	04Bh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[11]
76	04Ch	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[12]
77	04Dh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[13]
78	04Eh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[14]
79	04Fh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[15]
80	050h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[16]
81	051h	RTC	RTC_ALM	✓	✓	INT_SEL0~31	INT_SEL44~49	-
82	052h		RTC_PRD	✓	✓	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[18]
83	053h		-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[19]
84	054h	XTAL32	XTAL32_STOP	✓	-	INT_SEL0~31	INT_SEL44~49	-
85	055h	XTAL	XTAL_STOP	✓	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[21]
86	056h	WKTM	WKTM_PRD	✓	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[22]
87	057h	SWDT	SWDT_REFUDF	✓	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[23]
88	058h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[24]
89	059h	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[25]
90	05Ah	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[26]
91	05Bh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[27]
92	05Ch	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[28]
93	05Dh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[29]
94	05Eh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	int_vsse130[30]
95	05Fh	-	-	-	-	INT_SEL0~31	INT_SEL44~49	INT_VSSEL130[31]
96	060h	Timer6_1	TMR61_GCMA	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[0]
97	061h		TMR61_GCMB	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[1]
98	062h		TMR61_GCMC	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[2]
99	063h		TMR61_GCMD	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[3]
100	064h		TMR61_GCME	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[4]
101	065h		TMR61_GCMF	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[5]
102	066h		TMR61_GOVF	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[6]
103	067h		TMR61_GUDF	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[7]
104	068h		TMR61_GDTE	✓	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[8]
105	069h		-	-	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[9]
106	06Ah		-	-	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[10]
107	06Bh		TMR61_SCMA	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[11]
108	06Ch		TMR61_SCMB	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[12]

109	06Dh	-	-	-	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[13]
110	06Eh	-	-	-	-	INT_SEL0~31	INT_SEL50~55	int_vsse131[14]
111	06Fh	-	-	-	-	INT_SEL0~31	INT_SEL50~55	int_vsse131[15]

Number	Interrupt event request number	Function	Function Name	Whether to select as internal touch Origin	Can you choose for internal touch Origin	Interrupt selection register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
112	070h	Timer6_2	TMR62_GCMA	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[16]
113	071h		TMR62_GCMB	✓	✓	INT_SEL0~31	INT_SEL50~55	int_vsse131[17]
114	072h		TMR62_GCMC	✓	✓	INT_SEL0~31	INT_SEL50~55	int_vsse131[18]
115	073h		TMR62_GCMD	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[19]
116	074h		TMR62_GCME	✓	✓	INT_SEL0~31	INT_SEL50~55	int_vsse131[20]
117	075h		TMR62_GCMF	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[21]
118	076h		TMR62_GOVF	✓	✓	INT_SEL0~31	INT_SEL50~55	int_vsse131[22]
119	077h		TMR62_GUDF	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[23]
120	078h		TMR62_GDTE	✓	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[24]
121	079h		-	-	-	INT_SEL0~31	INT_SEL50~55	int_vsse131[25]
122	07Ah		-	-	-	INT_SEL0~31	INT_SEL50~55	int_vsse131[26]
123	07Bh		TMR62_SCMA	✓	✓	INT_SEL0~31	INT_SEL50~55	int_vsse131[27]
124	07Ch		TMR62_SCMB	✓	✓	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[28]
125	07Dh		-	-	-	INT_SEL0~31	INT_SEL50~55	INT_VSSEL131[29]
126	07Eh		-	-	-	INT_SEL0~31	INT_SEL50~55	int_vsse131[30]
127	07Fh		-	-	-	INT_SEL0~31	INT_SEL50~55	int_vsse131[31]
128	080h	Timer6_3	TMR63_GCMA	✓	✓	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[0]
129	081h		TMR63_GCMB	✓	✓	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[1]
130	082h		TMR63_GCMC	✓	✓	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[2]
131	083h		TMR63_GCMD	✓	✓	INT_SEL0~31	INT_SEL56~61	int_vsse132[3]
132	084h		TMR63_GCME	✓	✓	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[4]
133	085h		TMR63_GCMF	✓	✓	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[5]
134	086h		TMR63_GOVF	✓	✓	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[6]
135	087h		TMR63_GUDF	✓	✓	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[7]
136	088h		TMR63_GDTE	✓	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[8]
137	089h		-	-	-	INT_SEL0~31	INT_SEL56~61	INT_VSSEL132[9]
138	08Ah		-	-	-	INT_SEL0~31	INT_SEL56~61	int_vsse132[10]
139	08Bh		TMR63_SCMA	✓	✓	INT_SEL0~31	INT_SEL56~61	int_vsse132[11]
140	08Ch		TMR63_SCMB	✓	✓	INT_SEL0~31	INT_SEL56~61	int_vsse132[12]
141	08Dh		-	-	-	INT_SEL0~31	INT_SEL56~61	int_vsse132[13]
142	08Eh		-	-	-	INT_SEL0~31	INT_SEL56~61	int_vsse132[14]
143	08Fh		-	-	-	INT_SEL0~31	INT_SEL56~61	int_vsse132[15]
144	090h		-	-	-	INT_SEL0~31	INT_SEL56~61	int_vsse132[16]
145	091h		-	-	-	INT_SEL0~31	INT_SEL56~61	int_vsse132[17]
146	092h		-	-	-	INT_SEL0~31	INT_SEL56~61	int_vsse132[18]
147	093h		-	-	-	INT_SEL0~31	INT_SEL56~61	int_vsse132[19]
148	094h		-	-	-	INT_SEL0~31	INT_SEL56~61	int_vsse132[20]

149	095h	-	-	-	-	INT_SEL0~31	INT_SEL56~61	int_vssel132[21]
150	096h	-	-	-	-	INT_SEL0~31	INT_SEL56~61	int_vssel132[22]
151	097h	-	-	-	-	INT_SEL0~31	INT_SEL56~61	int_vssel132[23]

Number	Interrupt event request number	Function	Function Name	Whether to select as internal touch Origin	Can you choose for internal touch Origin	Interrupt selection register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
152	098h	-	-	-	-	INT_SEL0~31	INT_SEL56~61	int_vssel132[24]
153	099h	-	-	-	-	INT_SEL0~31	INT_SEL56~61	int_vssel132[25]
154	09Ah	-	-	-	-	INT_SEL0~31	INT_SEL56~61	int_vssel132[26]
155	09Bh	-	-	-	-	INT_SEL0~31	INT_SEL56~61	int_vssel132[27]
156	09Ch	-	-	-	-	INT_SEL0~31	INT_SEL56~61	int_vssel132[28]
157	09Dh	-	-	-	-	INT_SEL0~31	INT_SEL56~61	int_vssel132[29]
158	09Eh	-	-	-	-	INT_SEL0~31	INT_SEL56~61	int_vssel132[30]
159	09Fh	-	-	-	-	INT_SEL0~31	INT_SEL56~61	int_vssel132[31]
160	0A0h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[0]
161	0A1h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[1]
162	0A2h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[2]
163	0A3h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[3]
164	0A4h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[4]
165	0A5h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[5]
166	0A6h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[6]
167	0A7h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[7]
168	0A8h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[8]
169	0A9h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[9]
170	0AAh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[10]
171	0Abh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[11]
172	0ACh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[12]
173	0ADh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[13]
174	0AEh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	int_vssel133[14]
175	0AFh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	int_vssel133[15]
176	0B0h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[16]
177	0B1h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	int_vssel133[17]
178	0B2h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	int_vssel133[18]
179	0B3h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[19]
180	0B4h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[20]
181	0B5h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[21]
182	0B6h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[22]
183	0B7h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[23]
184	0B8h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[24]
185	0B9h	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[25]
186	0BAh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[26]
187	0BBh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	int_vssel133[27]
188	0BCh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[28]
189	0BDh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[29]
190	0BEh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	int_vssel133[30]

191	0BFh	-	-	-	-	INT_SEL0~31	INT_SEL62~67	INT_VSSEL133[31]
192	0C0h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[0]
193	0C1h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSSEL134[1]

Number	Interrupt event request number	Function	Function Name	Whether to select as internal touch Origin	Can you choose for internal touch Origin	Interrupt selection register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
194	0C2h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSEL134[2]
195	0C3h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSEL134[3]
196	0C4h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSEL134[4]
197	0C5h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSEL134[5]
198	0C6h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSEL134[6]
199	0C7h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSEL134[7]
200	0C8h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSEL134[8]
201	0C9h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	INT_VSEL134[9]
202	0CAh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[10]
203	0CBh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[11]
204	0CCh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[12]
205	0CDh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[13]
206	0CEh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[14]
207	0CFh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[15]
208	0D0h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[16]
209	0D1h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[17]
210	0D2h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[18]
211	0D3h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[19]
212	0D4h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[20]
213	0D5h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[21]
214	0D6h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[22]
215	0D7h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[23]
216	0D8h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[24]
217	0D9h	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[25]
218	0DAh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[26]
219	0DBh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[27]
220	0DCh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[28]
221	0DDh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[29]
222	0DEh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[30]
223	0DFh	-	-	-	-	INT_SEL0~31	INT_SEL68~73	int_vsel134[31]
224	0E0h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[0]
225	0E1h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[1]
226	0E2h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[2]
227	0E3h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[3]
228	0E4h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[4]
229	0E5h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[5]
230	0E6h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[6]
231	0E7h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[7]
232	0E8h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[8]

233	0E9h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[9]
234	0EAh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[10]
235	0EBh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSSEL135[11]

Number	Interrupt event request number	Function	Function Name	Whether to select as internal touch Origin	Can you choose for internal touch Origin	Interrupt selection register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
236	0ECh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[12]
237	0EDh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[13]
238	0EEh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[14]
239	0EFh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[15]
240	0F0h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[16]
241	0F1h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[17]
242	0F2h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[18]
243	0F3h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[19]
244	0F4h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[20]
245	0F5h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[21]
246	0F6h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[22]
247	0F7h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[23]
248	0F8h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[24]
249	0F9h	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[25]
250	0FAh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[26]
251	0FBh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[27]
252	0FCh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[28]
253	0FDh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[29]
254	0FEh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	int_vsel135[30]
255	0FFh	-	-	-	-	INT_SEL0~31	INT_SEL74~79	INT_VSEL135[31]
256	100h	TimerA_1	TMRA1_OVF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[0]
257	101h		TMRA1_UDF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[1]
258	102h		TMRA1_CMP	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[2]
259	103h	TimerA_2	TMRA2_OVF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[3]
260	104h		TMRA2_UDF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[4]
261	105h		TMRA2_CMP	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[5]
262	106h	TimerA_3	TMRA3_OVF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[6]
263	107h		TMRA3_UDF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[7]
264	108h		TMRA3_CMP	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[8]
265	109h	TimerA_4	TMRA4_OVF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[9]
266	10Ah		TMRA4_UDF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[10]
267	10Bh		TMRA4_CMP	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[11]
268	10Ch	TimerA_5	TMRA5_OVF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[12]
269	10Dh		TMRA5_UDF	✓	✓	INT_SEL0~31	INT_SEL80~85	int_vsel136[13]

270	10Eh		TMRA5_CMP	✓	✓	INT_SEL0~31	INT_SEL80~85	int_vsse136[14]
271	10Fh	-	-	-	-	INT_SEL0~31	INT_SEL80~85	int_vsse136[15]
272	110h	TimerA	TMRA6_OVF	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSSEL136[16]

Number	Interrupt event request number	Function	Function Name	Whether to select as internal touch Origin	Can you choose for internal touch Origin	Interrupt selection register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
273	111h	-6	TMRA6_UDF	✓	✓	INT_SEL0~31	INT_SEL80~85	int_vssel136[17]
274	112h		TMRA6_CMP	✓	✓	INT_SEL0~31	INT_SEL80~85	int_vssel136[18]
275	113h		USBFS_GLB	✓	-	INT_SEL0~31	INT_SEL80~85	int_vssel136[19]
276	114h		-	-	-	INT_SEL0~31	INT_SEL80~85	int_vssel136[20]
277	115h		-	-	-	INT_SEL0~31	INT_SEL80~85	int_vssel136[21]
278	116h		USART 1_REI	✓	✓	INT_SEL0~31	INT_SEL80~85	int_vssel136[22]
279	117h	USART1	USART1_RI	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[23]
280	118h		USART1_TI	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[24]
281	119h		USART 1_TCI	✓	✓	INT_SEL0~31	INT_SEL80~85	int_vssel136[25]
282	11Ah		USART1_RTOI	✓	✓	INT_SEL0~31	INT_SEL80~85	int_vssel136[26]
283	11Bh	USART2	USART 2_REI	✓	✓	INT_SEL0~31	INT_SEL80~85	int_vssel136[27]
284	11Ch		USART2_RI	✓	✓	INT_SEL0~31	INT_SEL80~85	int_vssel136[28]
285	11Dh		USART2_TI	✓	✓	INT_SEL0~31	INT_SEL80~85	INT_VSEL136[29]
286	11Eh		USART 2_TCI	✓	✓	INT_SEL0~31	INT_SEL80~85	int_vssel136[30]
287	11Fh		USART2_RTOI	✓	✓	INT_SEL0~31	INT_SEL80~85	int_vssel136[31]
288	120h	USART3	USART 3_REI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[0]
289	121h		USART3_RI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[1]
290	122h		USART3_TI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[2]
291	123h		USART 3_TCI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[3]
292	124h		USART3_RTOI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[4]
293	125h	USART4	USART 4_REI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[5]
294	126h		USART4_RI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[6]
295	127h		USART4_TI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[7]
296	128h		USART 4_TCI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[8]
297	129h		USART4_RTOI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[9]
298	12Ah		-	-	-	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[10]
299	12Bh	SPI1	SPI1_SPRI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[11]
300	12Ch		SPI1_SPTI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[12]
301	12Dh		SPI1_SPII	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[13]
302	12Eh		SPI1_SPEI	✓	✓	INT_SEL0~31	INT_SEL86~91	int_vssel137[14]
303	12Fh		SPI1_SPTEND	-	✓	INT_SEL0~31	INT_SEL86~91	int_vssel137[15]
304	130h	SPI2	SPI2_SPRI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[16]

305	131h	SPI3	SPI2_SPTI	✓	✓	INT_SEL0~31	INT_SEL86~91	int_vssel137[17]
306	132h		SPI2_SPII	✓	✓	INT_SEL0~31	INT_SEL86~91	int_vssel137[18]
307	133h		SPI2_SPEI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[19]
308	134h		SPI2_SPTEND	–	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[20]
309	135h		SPI3_SPRI	✓	✓	INT_SEL0~31	INT_SEL86~91	int_vssel137[21]
310	136h	SPI3	SPI3_SPTI	✓	✓	INT_SEL0~31	INT_SEL86~91	int_vssel137[22]
311	137h		SPI3_SPII	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[23]
312	138h		SPI3_SPEI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[24]
313	139h		SPI3_SPTEND	–	✓	INT_SEL0~31	INT_SEL86~91	int_vssel137[25]

Number	Interrupt event request number	Function	Function Name	Whether to select as internal touch Origin	Can you choose for internal touch Origin	Interrupt selection register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
314	13Ah	SPI4	SPI4_SPRI	✓	✓	INT_SEL0~31	INT_SEL86~91	int_vssel137[26]
315	13Bh		SPI4_SPTI	✓	✓	INT_SEL0~31	INT_SEL86~91	int_vssel137[27]
316	13Ch		SPI4_SPII	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[28]
317	13Dh		SPI4_SPEI	✓	✓	INT_SEL0~31	INT_SEL86~91	INT_VSEL137[29]
318	13Eh		SPI4_SPTEND	-	✓	INT_SEL0~31	INT_SEL86~91	int_vssel137[30]
319	13Fh	AOS_STRG	AOS_STRG*2	-	✓	INT_SEL0~31	INT_SEL86~91	int_vssel137[31]
320	140h	Timer4_1	TMR41_GCMUH	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[0]
321	141h		TMR41_GCMUL	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[1]
322	142h		TMR41_GCMVH	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[2]
323	143h		TMR41_GCMVL	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[3]
324	144h		TMR41_GCMWH	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[4]
325	145h		TMR41_GCMWL	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[5]
326	146h		TMR41_GOVF	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[6]
327	147h		TMR41_GUDF	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[7]
328	148h		TMR41_RLOU	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[8]
329	149h		TMR41_RLOV	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[9]
330	14Ah		TMR41_RLOW	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[10]
331	14Bh		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[11]
332	14Ch		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[12]
333	14Dh		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[13]
334	14Eh		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[14]
335	14Fh		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[15]
336	150h	Timer4_2	TMR42_GCMUH	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[16]
337	151h		TMR42_GCMUL	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[17]
338	152h		TMR42_GCMVH	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[18]
339	153h		TMR42_GCMVL	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[19]
340	154h		TMR42_GCMWH	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[20]
341	155h		TMR42_GCMWL	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[21]
342	156h		TMR42_GOVF	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[22]
343	157h		TMR42_GUDF	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[23]
344	158h		TMR42_RLOU	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[24]
345	159h		TMR42_RLOV	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[25]
346	15Ah		TMR42_RLOW	✓	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[26]
347	15Bh		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[27]
348	15Ch		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[28]
349	15Dh		-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[29]

350	15Eh	-	-	-	-	INT_SEL0~31	INT_SEL92~97	int_vssel138[30]
351	15Fh	-	-	-	-	INT_SEL0~31	INT_SEL92~97	INT_VSEL138[31]
352	160h	Timer4	TMR43_GCMUH	✓	-	INT_SEL0~31	INT_SEL98~103	INT_VSEL139[0]

Number	Interrupt event request number	Function	Function Name	Whether to select as internal touch Origin	Can you choose for internal touch Origin	Interrupt selection register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
		_3						
353	161h		TMR43_GCMUL	✓	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[1]
354	162h		TMR43_GCMVH	✓	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[2]
355	163h		TMR43_GCMVL	✓	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[3]
356	164h		TMR43_GCMWH	✓	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[4]
357	165h		TMR43_GCMWL	✓	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[5]
358	166h		TMR43_GOVF	✓	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[6]
359	167h		TMR43_GUDF	✓	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[7]
360	168h		TMR43_RLOU	✓	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[8]
361	169h		TMR43_RLOV	✓	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[9]
362	16Ah		TMR43_RLOW	✓	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[10]
363	16Bh		-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[11]
364	16Ch		-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[12]
365	16Dh		-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[13]
366	16Eh		-	-	-	INT_SEL0~31	INT_SEL98~103	int_vssel139[14]
367	16Fh		-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[15]
368	170h	Timer4 _1 EVT	TMR41_SCMUH	-	✓	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[16]
369	171h		TMR41_SCMUL	-	✓	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[17]
370	172h		TMR41_SCMVH	-	✓	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[18]
371	173h		TMR41_SCMVL	-	✓	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[19]
372	174h		TMR41_SCMWH	-	✓	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[20]
373	175h		TMR41_SCMWL	-	✓	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[21]
374	176h	Timer4 _2 EVT	TMR42_SCMUH	-	✓	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[22]
375	177h		TMR42_SCMUL	-	✓	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[23]
376	178h		TMR42_SCMVH	-	✓	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[24]
377	179h		TMR42_SCMVL	-	✓	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[25]
378	17Ah		TMR42_SCMWH	-	✓	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[26]
379	17Bh		TMR42_SCMWL	-	✓	INT_SEL0~31	INT_SEL98~103	int_vssel139[27]
380	17Ch	-	-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[28]
381	17Dh	-	-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[29]
382	17Eh	-	-	-	-	INT_SEL0~31	INT_SEL98~103	int_vssel139[30]
383	17Fh	-	-	-	-	INT_SEL0~31	INT_SEL98~103	INT_VSSEL139[31]
384	180h	Timer4 _3 EVT	TMR43_SCMUH	-	✓	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[0]
385	181h		TMR43_SCMUL	-	✓	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[1]
386	182h		TMR43_SCMVH	-	✓	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[2]
387	183h		TMR43_SCMVL	-	✓	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[3]

388	184h	EMB	TMR43_SCMWH	-	✓	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[4]
389	185h		TMR43_SCMWL	-	✓	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[5]
390	186h		EMB_GR0	✓	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[6]

Number	Interrupt event request number	Function	Function Name	Whether to select as internal touch Origin	Can you choose for internal touch Origin	Interrupt selection register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
391	187h		EMB_GR1	✓	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[7]
392	188h		EMB_GR2	✓	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[8]
393	189h		EMB_GR3	✓	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[9]
394	18Ah	EVENT port	EVENT_PORT1	✓	✓	INT_SEL0~31	INT_SEL104~109	-
395	18Bh		EVENT_PORT2	✓	✓	INT_SEL0~31	INT_SEL104~109	-
396	18Ch		EVENT_PORT3	✓	✓	INT_SEL0~31	INT_SEL104~109	-
397	18Dh		EVENT_PORT4	✓	✓	INT_SEL0~31	INT_SEL104~109	-
398	18Eh	-	-	-	-	INT_SEL0~31	INT_SEL104~109	int_vssel140[14]
399	18Fh	-	-	-	-	INT_SEL0~31	INT_SEL104~109	int_vssel140[15]
400	190h	I2S1	I2S1_TXIRQOUT	✓	✓	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[16]
401	191h		I2S1_RXIRQOUT	✓	✓	INT_SEL0~31	INT_SEL104~109	int_vssel140[17]
402	192h		I2S1_ERRIRQOUT	✓	-	INT_SEL0~31	INT_SEL104~109	int_vssel140[18]
403	193h	I2S2	I2S2_TXIRQOUT	✓	✓	INT_SEL0~31	INT_SEL104~109	int_vssel140[19]
404	194h		I2S2_RXIRQOUT	✓	✓	INT_SEL0~31	INT_SEL104~109	int_vssel140[20]
405	195h		I2S2_ERRIRQOUT	✓	-	INT_SEL0~31	INT_SEL104~109	int_vssel140[21]
406	196h	I2S3	I2S3_TXIRQOUT	✓	✓	INT_SEL0~31	INT_SEL104~109	int_vssel140[22]
407	197h		I2S3_RXIRQOUT	✓	✓	INT_SEL0~31	INT_SEL104~109	int_vssel140[23]
408	198h		I2S3_ERRIRQOUT	✓	-	INT_SEL0~31	INT_SEL104~109	INT_VSSEL140[24]
409	199h	I2S4	I2S4_TXIRQOUT	✓	✓	INT_SEL0~31	INT_SEL104~109	int_vssel140[25]
410	19Ah		I2S4_RXIRQOUT	✓	✓	INT_SEL0~31	INT_SEL104~109	int_vssel140[26]
411	19Bh		I2S4_ERRIRQOUT	✓	-	INT_SEL0~31	INT_SEL104~109	int_vssel140[27]
412	19Ch	-	-	-	-	INT_SEL0~31	INT_SEL104~109	int_vssel140[28]
413	19Dh	-	-	-	-	INT_SEL0~31	INT_SEL104~109	int_vssel140[29]
414	19Eh	-	-	-	-	INT_SEL0~31	INT_SEL104~109	int_vssel140[30]
415	19Fh	-	-	-	-	INT_SEL0~31	INT_SEL104~109	int_vssel140[31]
416	1A0h	ACMP	ACMP1	✓	✓	INT_SEL0~31	INT_SEL110~115	-

417	1A1h		ACMP2	√	√	INT_SEL0~31	INT_SEL110~115	-
418	1A2h		ACMP3	√	√	INT_SEL0~31	INT_SEL110~115	-
419	1A3h	-	-	-	-	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[3]

Number	Interrupt event request number	Function	Function Name	Whether to select as internal touch Origin	Can you choose for internal touch Origin	Interrupt selection register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
420	1A4h	I2C1	I2C1_RXI	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[4]
421	1A5h		I2C1_TXI	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[5]
422	1A6h		I2C1_TEI	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[6]
423	1A7h		I2C1_EE1	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[7]
424	1A8h	I2C2	I2C2_RXI	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[8]
425	1A9h		I2C2_TXI	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[9]
426	1AAh		I2C2_TEI	✓	✓	INT_SEL0~31	INT_SEL110~115	int_vssel141[10]
427	1ABh		I2C2_EE1	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[11]
428	1ACh	I2C3	I2C3_RXI	✓	✓	INT_SEL0~31	INT_SEL110~115	INT_VSSEL141[12]
429	1ADh		I2C3_TXI	✓	✓	INT_SEL0~31	INT_SEL110~115	int_vssel141[13]
430	1AEh		I2C3_TEI	✓	✓	INT_SEL0~31	INT_SEL110~115	int_vssel141[14]
431	1AFh		I2C3_EE1	✓	✓	INT_SEL0~31	INT_SEL110~115	int_vssel141[15]
432	1B0h	USART1	USART1_WUPI	✓	-	INT_SEL0~31	INT_SEL110~115	-
433	1B1h	PVD	PVD_PVD1	✓	✓	INT_SEL0~31	INT_SEL110~115	int_vssel141[17]
434	1B2h		PVD_PVD2	✓	✓	INT_SEL0~31	INT_SEL110~115	int_vssel141[18]
435	1B3h	OTS	OTS	✓	✓	INT_SEL0~31	INT_SEL110~115	-
436	1B4h	FCM	FCMFERRI	✓	-	INT_SEL0~31	INT_SEL110~115	int_vssel141[20]
437	1B5h		FCMMENDI	✓	-	INT_SEL0~31	INT_SEL110~115	int_vssel141[21]
438	1B6h		FCMCOVFI	✓	-	INT_SEL0~31	INT_SEL110~115	int_vssel141[22]
439	1B7h	WDT	WDT_REFUDF	✓	✓	INT_SEL0~31	INT_SEL110~115	int_vssel141[23]
440	1B8h	-	-	-	-	INT_SEL0~31	INT_SEL110~115	int_vssel141[24]
441	1B9h	-	-	-	-	INT_SEL0~31	INT_SEL110~115	int_vssel141[25]
442	1BAh	-	-	-	-	INT_SEL0~31	INT_SEL110~115	int_vssel141[26]
443	1BBh	-	-	-	-	INT_SEL0~31	INT_SEL110~115	int_vssel141[27]
444	1BCh	-	-	-	-	INT_SEL0~31	INT_SEL110~115	int_vssel141[28]
445	1BDh	-	-	-	-	INT_SEL0~31	INT_SEL110~115	int_vssel141[29]
446	1BEh	-	-	-	-	INT_SEL0~31	INT_SEL110~115	int_vssel141[30]
447	1BFh	-	-	-	-	INT_SEL0~31	INT_SEL110~115	int_vssel141[31]
448	1C0h	ADC1	ADC1_EOCA	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[0]
449	1C1h		ADC1_EOCB	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[1]
450	1C2h		ADC1_CHCMP	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[2]
451	1C3h		ADC1_SEQCMP	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[3]
452	1C4h	ADC2	ADC2_EOCA	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[4]
453	1C5h		ADC2_EOCB	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[5]
454	1C6h		ADC2_CHCMP	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[6]
455	1C7h		ADC2_SEQCMP	✓	✓	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[7]
456	1C8h	TRNG	TRNG_END	✓	✓	INT_SEL0~31	INT_SEL116~121	-
457	1C9h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	INT_VSSEL142[9]
458	1CAh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[10]

459	1CBh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[11]
460	1CCh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[12]
461	1CDh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[13]

Number	Interrupt event request number	Function	Function Name	Whether to select as internal touch Origin	Can you choose for internal touch Origin	Interrupt selection register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
462	1CEh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[14]
463	1CFh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[15]
464	1D0h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[16]
465	1D1h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[17]
466	1D2h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[18]
467	1D3h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[19]
468	1D4h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[20]
469	1D5h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[21]
470	1D6h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[22]
471	1D7h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[23]
472	1D8h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[24]
473	1D9h	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[25]
474	1DAh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[26]
475	1DBh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[27]
476	1DCh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[28]
477	1DDh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[29]
478	1DEh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[30]
479	1DFh	-	-	-	-	INT_SEL0~31	INT_SEL116~121	int_vssel142[31]
480	1E0h	SDIOC1	SDIOC1_DMAR	-	✓	INT_SEL0~31	INT_SEL122~127	INT_VSEL143[0]
481	1E1h		SDIOC1_DMAW	-	✓	INT_SEL0~31	INT_SEL122~127	INT_VSEL143[1]
482	1E2h		SDIOC1_SD	✓	-	INT_SEL0~31	INT_SEL122~127	INT_VSEL143[2]
483	1E3h	SDIOC2	SDIOC2_DMAR	-	✓	INT_SEL0~31	INT_SEL122~127	INT_VSEL143[3]
484	1E4h		SDIOC2_DMAW	-	✓	INT_SEL0~31	INT_SEL122~127	INT_VSEL143[4]
485	1E5h		SDIOC2_SD	✓	-	INT_SEL0~31	INT_SEL122~127	INT_VSEL143[5]
486	1E6h	CAN	CAN_INT	✓	-	INT_SEL0~31	INT_SEL122~127	INT_VSEL143[6]
487	1E7h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSEL143[7]
488	1E8h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSEL143[8]
489	1E9h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSEL143[9]
490	1EAh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[10]
491	1EBh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[11]
492	1EcH	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[12]
493	1EDh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[13]
494	1EEh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[14]
495	1EFh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[15]
496	1F0h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	INT_VSEL143[16]
497	1F1h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[17]
498	1F2h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[18]
499	1F3h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[19]
500	1F4h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[20]

501	1F5h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[21]
502	1F6h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[22]
503	1F7h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[23]

Number	Interrupt event request sequence number	Function	Function Name	Whether to select as internal touch Origin	Can you choose for internal touch Origin	Interrupt selection register corresponding to NVIC vector*1		
						NVIC vector 0~31	NVIC vector 32~127	NVIC vector 128~143
504	1F8h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[24]
505	1F9h	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[25]
506	1FAh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[26]
507	1FBh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[27]
508	1FCh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[28]
509	1FDh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[29]
510	1FEh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[30]
511	1FFh	-	-	-	-	INT_SEL0~31	INT_SEL122~127	int_vssel143[31]

*1 : The interrupt event request sequence number selected by the interrupt selection register is invalid if this register or this bit setting is not configured.

*2: AOS_STRG is generated by the SFTG bit of the software set peripheral trigger event register (INTSFTTRG).

10.4 Function Description

10.4.1 Non-maskable interrupts

The non-maskable interrupt sources are as follows:

- **NMI** Pin Interrupt
- Detection of primary oscillator stop interrupt
- WDT underflow/refresh interrupt
- SWDT underflow/refresh interrupt
- Low voltage detection PVD1 interrupt
- Low voltage detection PVD2 interrupt
- SRAM parity error interrupt
- SRAM ECC checksum error interrupt
- MPU bus error interrupt

Non-maskable interrupts have the highest priority. Since non-maskable interrupts can select multiple interrupt event requests, the status of each interrupt event request can be determined by querying the flag register (INT_NMIFR). Please make sure all the flag bits are "**O**" before the **non-maskable interrupt** processing exits. The non-maskable interrupt is disabled by default and can be set by ICG or control register.

When using register settings, follow the following procedure:

1. NFEN bit to disable the digital filter; set the NMITRG bit of the INT_NMICR register to select the NMI trigger edge; set the SMPCLK bit to select the sampling clock of the digital filter; set the NFEN bit to enable the digital filter. set the NFEN bit to enable the digital filter.
2. When using other non-maskable interrupt event requests, please configure the corresponding function.
3. Write "1" to each register bit of INT_NMICFR to clear the INT_NMIFR flag register bit to prevent misoperation.
4. Enable non-maskable interrupt events by setting the INT_NMIENR selection register. Caution:

-Once the corresponding bit of INT_NMIENR is set to "1", it cannot be changed except by RESET. Once the INT_NMIENR corresponding bit is set to "1", it cannot be changed unless reset with RESET. ICG setting is only for external NMI pins, enable ICG setting by configuring ICG register ICG1.NMIIICGENA bit. NMITRG to select the NMI trigger edge; set the ICG1.SMPCLK bit to select the digital filter sample clock; set the ICG1.NFEN bit to enable the digital filter; and set the

ICG1_NMIENR bit to enable the **NMI** pin interrupt. register settings are invalid after ICG setting. see the Initial Configuration (NMIENR bit to enable the NMI pin interrupt.

10.4.2 External pin interrupt event request

If you need to use an external pin interrupt event request, please set up the following flow:

1. Clear the INT_EIRQCRm.EFEN bit ($m=0\sim15$) to disable the digital filter.
2. Set the IRQTRG[1:0] bits of INT_EIRQCRm to select the trigger edge or level; set the SMPCLK[1:0] bits to select the digital filter sampling clock; set the EFEN bits to enable the digital filter.

10.4.3 Interrupt source selection

The interrupt controller uses a total of 144 interrupt vectors and provides three interrupt event request options to meet various interrupt configuration requirements through flexible combinations.

The first way

There are 32 interrupt vectors, and all interrupt events request any 1 as interrupt source, which are selected by the interrupt/event selection registers INT_SEL0~31, and enabled by the INT_IER register, corresponding to the interrupt vectors 0~31 of NVIC;

The second way

A total of 96 interrupt vectors, 32 select 1 as interrupt source, selected by the interrupt selection register INT_SEL32~127, corresponding interrupt vectors are 32~127;

The third way

There are 16 interrupt vectors, 32 peripheral interrupt event requests share one interrupt vector, each peripheral can apply for interrupt, distinguished by the peripheral flag bit, and the interrupt event requests are enabled by the interrupt enable register INT_VSEL128~143, corresponding to the interrupt vector of NVIC 128~143. Select the interrupt event request with peripheral flag bit, refer to 10.3. The interrupt event requests with peripheral flags are selected by referring to the NVIC vectors 128~143 in 10.3.2 Interrupt event request sequence.

Refer to section 10.3.2 Interrupt Event Request Sequence Number for specific interrupt vector assignments.

10.4.4 Software Interruptions

The software interrupt function can be licensed by writing directly to the software interrupt control register INT_SWIER to generate one interrupt event request, which is controlled by the INT_IER interrupt enable bit. A total of 32 software interrupt event requests are configured, corresponding to interrupt vectors 0~31, please refer to section 10.3.1 Interrupt Vector Table for details.

10.4.5 Interrupt/event selection

The interrupts selected by INT_SEL0~31 share the interrupt vector 0~31 of NVIC with software interrupts, and are licensed by INT_IER interrupt enable register; meanwhile, these interrupt event requests can also be selected as event inputs to wake up the kernel (WFE) selected by INT_EVTER event enable register, as shown in the block diagram below.

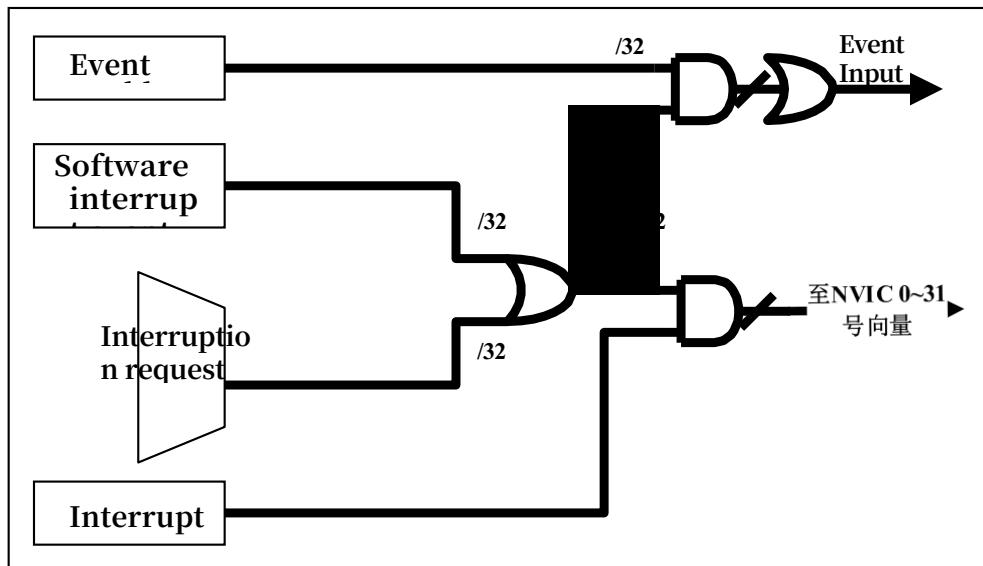


Figure 10-2 Interrupt Event Selection

10.4.6 WFE Wakeup

Event

Management

The MCU is capable of handling events to wake up the core (WFE). The wake-up event can be generated either from the interrupt input of the NVIC or from the event input. The setup flow is as follows.

- Wakes up the WFE from the interrupt input of the NVIC, enables an interrupt in the peripheral's control register, sets the INT_SEL_n and INT_IER registers according to the selected interrupt vector, but not in the NVIC, and sets INT_EVTER to non-enable, and enables the SEVONPEND bit in the SCR of the Cortex™-M4F system control register. When the MCU recovers from WFE, the interrupt flag bits of the corresponding peripheral and the NVIC interrupt flag register need to be cleared.

The WFE stop mode entry and wake-up process is as follows:

- Setting the stop mode register;
- Set the stop mode wake-up register INT_WUPEN;
- Sets the pin EIRQ input and EIRQ control register INT_EIRQCRn;
- Select INT_SEL_n to select the corresponding EIRQ interrupt event request sequence

number;

- 5) INT_IER register for enabling the corresponding interrupt event request, INT_EVTER non-enabling;
- 6) Set the SEVONPEND bit in SCR to "1";
- 7) Do the following to ensure that the system enters stop mode:

SEV()	Set the internal event register
WFE()	Clear the event register

- WFE(). System enters stop mode
- 8) Waiting for the selected interrupt event request to occur, the system will wake up from stop mode but not enter the interrupt handler subroutine.
- Wakes up the WFE from the event input of the NVIC. configure an interrupt event request as an event input, enabled via the event enable register INT_EVTER. When the CPU recovers from the WFE, the interrupt flag bit of the corresponding peripheral needs to be cleared.

The WFE stop mode wake-up process is as follows.

- 1) Setting the stop mode register;
- 2) Set the stop mode wake-up register INT_WUPEN;
- 3) Sets the pin EIRQ input and EIRQ control register INT_EIRQCRn;
- 4) Select INT_SELn to select the corresponding EIRQ interrupt event request sequence number;
- 5) INT_IER register is non-enabled and INT_EVTER enables the corresponding interrupt event request;
- 6) Do the following to ensure that the system enters stop mode:

SEV(). Set the internal event register

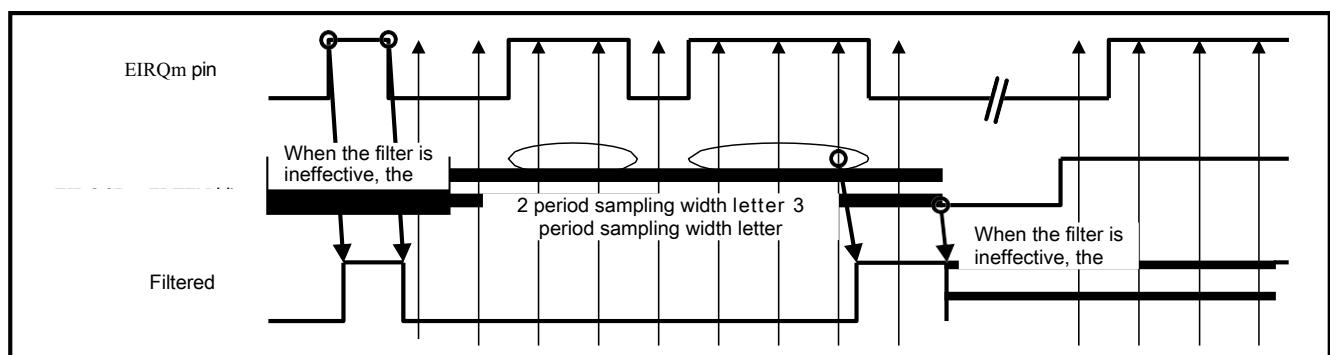
WFE(). Clear the event register

WFE(). System enters stop mode

- 7) Waiting for the selected interrupt event request to occur, the system will wake up from the stop mode but will not enter the interrupt handler subroutine.

10.4.7 Digital Filters

For the **NMI** and **EIRQx** ($x=0\sim15$) pin event inputs, a digital filter can be selected for noise filtering. The sampling clock of the filter is PCLK3, and input signals with less than 3 filter cycles will be filtered out. The specific operating timing diagram is as follows:



**Figure 10-3 Schematic
diagram of digital filter
operation**

Before entering the stop mode, set INT_NMICR.NFEN and INT_EIRQCRm.EFEN to disable the HC32F460_F45x_A460 Series Reference

digital filter. Enable the digital filter again after returning from the stop mode. The setting flow is as follows:

- 1) Set system stop mode register
- 2) Configure stop mode wake-up interrupts
- 3) Stop Digital Filter
- 4) Execute **VVFI** and the system enters stop mode

10.4.8 Low power mode return

10.4.8.1 Hibernation mode return

When selecting an event interrupt source as a sleep mode return condition, the following settings are required:

- Select event as CPU interrupt source
- Enable the control registers in the NVIC
- If you need to use non-maskable interrupt, you need to set INT_NMIENR enable register

10.4.8.2 Stop mode return

Either the non-maskable event interrupt source or the maskable event source selected in the INT_WUPEN register can be selected as the return condition for the stop mode.

The following settings are required to return from the stop mode:

- Select the event interrupt source as the return condition of stop mode A. For non-maskable interrupts, set by INT_NMIENR enable register
NMI pin interrupt, SWDT, PVD1, PVD2 can wake up the stop mode. B. For maskable interrupt, it is set by INT_WUPEN enable register.
- Select event as CPU interrupt source
- Enable the control registers in the NVIC

For unchecked EIRQ pins, they will not be detected because the clock is turned off.

10.4.8.3 Power down mode return

The power-down mode return can be returned by the conditions indicated in the Power Control (PWC) section of Chapter 5, RES# pin reset, power-on reset, and low voltage detect 0 conditions. Upon return the CPU enters reset interrupt processing. Refer to the [Power Control (PWC)] chapter for detailed descriptions.

10.4.8.4 Non-maskable interrupts and **WFI** commands

Before the **VVFI** instruction is executed, please make sure all status bits of INT_NMIFR of the non-maskable interrupt flag register are "0".

10.4.9 Internal trigger events

Peripheral peripherals such as ADC, Timer, DMA, PORT, DCU, etc. can also trigger other modules to start operating by writing the peripheral trigger event register in addition to configuring the module's own registers to start operating. To use the internal trigger event, you need to clear the

PWC_FCG0.AOS bit to enable the peripheral circuit trigger function. Please refer to each module chapter for the detailed setting procedure.

10.5 Register Description

The following table shows the

list of INTC registers. INTC

Register Name	Sym bols	Offset Address	Bit width	Reset value
NMI pin non-maskable interrupt control register	INT_NMICR	0x0000	32	0x0000_0000
Non-maskable interrupt enable register	INT_NMIENR	0x0004	32	0x0000_0000
Non-maskable interrupt flag register	INT_NMIFR	0x0008	32	0x0000_0000
Non-maskable interrupt flag clear register	INT_NMICFR	0x000C	32	0x0000_0000
External pin interrupt control register 0	INT_EIRQCR0	0x0010	32	0x0000_0000
External pin interrupt control register 1	INT_EIRQCR1	0x0014	32	0x0000_0000
External pin interrupt control register 2	INT_EIRQCR2	0x0018	32	0x0000_0000
External pin interrupt control register 3	INT_EIRQCR3	0x001C	32	0x0000_0000
External pin interrupt control register 4	INT_EIRQCR4	0x0020	32	0x0000_0000
External pin interrupt control register 5	INT_EIRQCR5	0x0024	32	0x0000_0000
External pin interrupt control register 6	INT_EIRQCR6	0x0028	32	0x0000_0000
External pin interrupt control register 7	INT_EIRQCR7	0x002C	32	0x0000_0000
External pin interrupt control register 8	INT_EIRQCR8	0x0030	32	0x0000_0000
External pin interrupt control register 9	INT_EIRQCR9	0x0034	32	0x0000_0000
External pin interrupt control register 10	INT_EIRQCR10	0x0038	32	0x0000_0000
External pin interrupt control register 11	INT_EIRQCR11	0x003C	32	0x0000_0000
External pin interrupt control register 12	INT_EIRQCR12	0x0040	32	0x0000_0000
External pin interrupt control register 13	INT_EIRQCR13	0x0044	32	0x0000_0000
External pin interrupt control register 14	INT_EIRQCR14	0x0048	32	0x0000_0000
External pin interrupt control register 15	INT_EIRQCR15	0x004C	32	0x0000_0000
Stop mode wake-up event enable register	INT_WUPEN	0x0050	32	0x0000_0000
External pin interrupt flag register	INT_EIFR	0x0054	32	0x0000_0000
External pin interrupt flag clear register	INT_EICFR	0x0058	32	0x0000_0000
Interrupt/event selection register 0	INT_SEL0	0x005C	32	0x0000_01FF
Interrupt/event selection register 1	INT_SEL1	0x0060	32	0x0000_01FF
Interrupt/event selection register 2	INT_SEL2	0x0064	32	0x0000_01FF
Interrupt/event selection register 3	INT_SEL3	0x0068	32	0x0000_01FF
Interrupt/event selection register 4	INT_SEL4	0x006C	32	0x0000_01FF
Interrupt/event selection register 5	INT_SEL5	0x0070	32	0x0000_01FF
Interrupt/event selection register 6	INT_SEL6	0x0074	32	0x0000_01FF
Interrupt/event selection register 7	INT_SEL7	0x0078	32	0x0000_01FF
...	INT_SEL8	0x007C	32	0x0000_01FF

Register Name	Symbols	Offset Address	Bit width	Reset value
Interrupt/event selection register 10	INT_SEL10	0x0084	32	0x0000_01FF
Interrupt/event selection register 11	INT_SEL11	0x0088	32	0x0000_01FF
Interrupt/event selection register 12	INT_SEL12	0x008C	32	0x0000_01FF
Interrupt/Event Selection Register 13	INT_SEL13	0x0090	32	0x0000_01FF
Interrupt/Event Selection Register 14	INT_SEL14	0x0094	32	0x0000_01FF
Interrupt/event selection register 15	INT_SEL15	0x0098	32	0x0000_01FF
Interrupt/event selection register 16	INT_SEL16	0x009C	32	0x0000_01FF
Interrupt/event selection register 17	INT_SEL17	0x00A0	32	0x0000_01FF
Interrupt/event selection register 18	INT_SEL18	0x00A4	32	0x0000_01FF
Interrupt/Event Selection Register 19	INT_SEL19	0x00A8	32	0x0000_01FF
Interrupt/event selection register 20	INT_SEL20	0x00AC	32	0x0000_01FF
Interrupt/event selection register 21	INT_SEL21	0x00B0	32	0x0000_01FF
Interrupt/event selection register 22	INT_SEL22	0x00B4	32	0x0000_01FF
Interrupt/event selection register 23	INT_SEL23	0x00B8	32	0x0000_01FF
Interrupt/event selection register 24	INT_SEL24	0x00BC	32	0x0000_01FF
Interrupt/event selection register 25	INT_SEL25	0x00C0	32	0x0000_01FF
Interrupt/Event Selection Register 26	INT_SEL26	0x00C4	32	0x0000_01FF
Interrupt/Event Selection Register 27	INT_SEL27	0x00C8	32	0x0000_01FF
Interrupt/Event Selection Register 28	INT_SEL28	0x00CC	32	0x0000_01FF
Interrupt/Event Selection Register 29	INT_SEL29	0x00D0	32	0x0000_01FF
Interrupt/event selection register 30	INT_SEL30	0x00D4	32	0x0000_01FF
Interrupt/event selection register 31	INT_SEL31	0x00D8	32	0x0000_01FF
Interrupt selection register 32	INT_SEL32	0x00DC	32	0x0000_01FF
Interrupt selection register 33	INT_SEL33	0x00E0	32	0x0000_01FF
Interrupt selection register 34	INT_SEL34	0x00E4	32	0x0000_01FF
Interrupt selection register 35	INT_SEL35	0x00E8	32	0x0000_01FF
Interrupt selection register 36	INT_SEL36	0x00EC	32	0x0000_01FF
Interrupt selection register 37	INT_SEL37	0x00F0	32	0x0000_01FF
Interrupt selection register 38	INT_SEL38	0x00F4	32	0x0000_01FF
Interrupt selection register 39	INT_SEL39	0x00F8	32	0x0000_01FF
Interrupt selection register 40	INT_SEL40	0x00FC	32	0x0000_01FF
Interrupt selection register 41	INT_SEL41	0x0100	32	0x0000_01FF
Interrupt selection register 42	INT_SEL42	0x0104	32	0x0000_01FF
Interrupt selection register 43	INT_SEL43	0x0108	32	0x0000_01FF
Interrupt selection register 44	INT_SEL44	0x010C	32	0x0000_01FF
Interrupt selection register 45	INT_SEL45	0x0110	32	0x0000_01FF
Interrupt selection register 46	INT_SEL46	0x0114	32	0x0000_01FF
Interrupt selection register 47	INT_SEL47	0x0118	32	0x0000_01FF

Register Name	Symbols	Offset Address	Bit width	Reset value
Interrupt selection register 48	INT_SEL48	0x011C	32	0x0000_01FF
Interrupt selection register 49	INT_SEL49	0x0120	32	0x0000_01FF
Interrupt selection register 50	INT_SEL50	0x0124	32	0x0000_01FF
Interrupt selection register 51	INT_SEL51	0x0128	32	0x0000_01FF
Interrupt selection register 52	INT_SEL52	0x012C	32	0x0000_01FF
Interrupt selection register 53	INT_SEL53	0x0130	32	0x0000_01FF
Interrupt selection register 54	INT_SEL54	0x0134	32	0x0000_01FF
Interrupt selection register 55	INT_SEL55	0x0138	32	0x0000_01FF
Interrupt selection register 56	INT_SEL56	0x013C	32	0x0000_01FF
Interrupt selection register 57	INT_SEL57	0x0140	32	0x0000_01FF
Interrupt selection register 58	INT_SEL58	0x0144	32	0x0000_01FF
Interrupt selection register 59	INT_SEL59	0x0148	32	0x0000_01FF
Interrupt selection register 60	INT_SEL60	0x014C	32	0x0000_01FF
Interrupt selection register 61	INT_SEL61	0x0150	32	0x0000_01FF
Interrupt selection register 62	INT_SEL62	0x0154	32	0x0000_01FF
Interrupt selection register 63	INT_SEL63	0x0158	32	0x0000_01FF
Interrupt selection register 64	INT_SEL64	0x015C	32	0x0000_01FF
Interrupt selection register 65	INT_SEL65	0x0160	32	0x0000_01FF
Interrupt selection register 66	INT_SEL66	0x0164	32	0x0000_01FF
Interrupt selection register 67	INT_SEL67	0x0168	32	0x0000_01FF
Interrupt selection register 68	INT_SEL68	0x016C	32	0x0000_01FF
Interrupt selection register 69	INT_SEL69	0x0170	32	0x0000_01FF
Interrupt selection register 70	INT_SEL70	0x0174	32	0x0000_01FF
Interrupt selection register 71	INT_SEL71	0x0178	32	0x0000_01FF
Interrupt selection register 72	INT_SEL72	0x017C	32	0x0000_01FF
Interrupt selection register 73	INT_SEL73	0x0180	32	0x0000_01FF
Interrupt selection register 74	INT_SEL74	0x0184	32	0x0000_01FF
Interrupt selection register 75	INT_SEL75	0x0188	32	0x0000_01FF
Interrupt selection register 76	INT_SEL76	0x018C	32	0x0000_01FF
Interrupt selection register 77	INT_SEL77	0x0190	32	0x0000_01FF
Interrupt selection register 78	INT_SEL78	0x0194	32	0x0000_01FF
Interrupt selection register 79	INT_SEL79	0x0198	32	0x0000_01FF
Interrupt selection register 80	INT_SEL80	0x019C	32	0x0000_01FF
Interrupt selection register 81	INT_SEL81	0x01A0	32	0x0000_01FF
Interrupt selection register 82	INT_SEL82	0x01A4	32	0x0000_01FF
Interrupt selection register 83	INT_SEL83	0x01A8	32	0x0000_01FF
Interrupt selection register 84	INT_SEL84	0x01AC	32	0x0000_01FF
Interrupt selection register 85	INT_SEL85	0x01B0	32	0x0000_01FF

Register Name	Sym bols	Offset Address	Bit width	Reset value
Interrupt selection register 86	INT_SEL86	0x01B4	32	0x0000_01FF
Interrupt selection register 87	INT_SEL87	0x01B8	32	0x0000_01FF
Interrupt selection register 88	INT_SEL88	0x01BC	32	0x0000_01FF
Interrupt selection register 89	INT_SEL89	0x01C0	32	0x0000_01FF
Interrupt selection register 90	INT_SEL90	0x01C4	32	0x0000_01FF
Interrupt selection register 91	INT_SEL91	0x01C8	32	0x0000_01FF
Interrupt selection register 92	INT_SEL92	0x01CC	32	0x0000_01FF
Interrupt Select Register 93	INT_SEL93	0x01D0	32	0x0000_01FF
Interrupt selection register 94	INT_SEL94	0x01D4	32	0x0000_01FF
Interrupt selection register 95	INT_SEL95	0x01D8	32	0x0000_01FF
Interrupt selection register 96	INT_SEL96	0x01DC	32	0x0000_01FF
Interrupt selection register 97	INT_SEL97	0x01E0	32	0x0000_01FF
Interrupt selection register 98	INT_SEL98	0x01E4	32	0x0000_01FF
Interrupt selection register 99	INT_SEL99	0x01E8	32	0x0000_01FF
Interrupt selection register 100	INT_SEL100	0x01EC	32	0x0000_01FF
Interrupt selection register 101	INT_SEL101	0x01F0	32	0x0000_01FF
Interrupt selection register 102	INT_SEL102	0x01F4	32	0x0000_01FF
Interrupt selection register 103	INT_SEL103	0x01F8	32	0x0000_01FF
Interrupt selection register 104	INT_SEL104	0x01FC	32	0x0000_01FF
Interrupt selection register 105	INT_SEL105	0x0200	32	0x0000_01FF
Interrupt selection register 106	INT_SEL106	0x0204	32	0x0000_01FF
Interrupt selection register 107	INT_SEL107	0x0208	32	0x0000_01FF
Interrupt selection register 108	INT_SEL108	0x020C	32	0x0000_01FF
Interrupt selection register 109	INT_SEL109	0x0210	32	0x0000_01FF
Interrupt selection register 110	INT_SEL110	0x0214	32	0x0000_01FF
Interrupt selection register 111	INT_SEL111	0x0218	32	0x0000_01FF
Interrupt selection register 112	INT_SEL112	0x021C	32	0x0000_01FF
Interrupt selection register 113	INT_SEL113	0x0220	32	0x0000_01FF
Interrupt selection register 114	INT_SEL114	0x0224	32	0x0000_01FF
Interrupt selection register 115	INT_SEL115	0x0228	32	0x0000_01FF
Interrupt selection register 116	INT_SEL116	0x022C	32	0x0000_01FF
Interrupt selection register 117	INT_SEL117	0x0230	32	0x0000_01FF
Interrupt selection register 118	INT_SEL118	0x0234	32	0x0000_01FF
Interrupt selection register 119	INT_SEL119	0x0238	32	0x0000_01FF
Interrupt selection register 120	INT_SEL120	0x023C	32	0x0000_01FF
Interrupt selection register 121	INT_SEL121	0x0240	32	0x0000_01FF
Interrupt selection register 122	INT_SEL122	0x0244	32	0x0000_01FF
Interrupt selection register 123	INT_SEL123	0x0248	32	0x0000_01FF

Register Name	Sym bols	Offset Address	Bit width	Reset value
Interrupt selection register 124	INT_SEL124	0x024C	32	0x0000_01FF
Interrupt selection register 125	INT_SEL125	0x0250	32	0x0000_01FF
Interrupt selection register 126	INT_SEL126	0x0254	32	0x0000_01FF
Interrupt selection register 127	INT_SEL127	0x0258	32	0x0000_01FF
Vector Shared Interrupt Select Register 128	INT_VSSEL128	0x025C	32	0x0000_0000
Vector Shared Interrupt Select Register 129	INT_VSSEL129	0x0260	32	0x0000_0000
Vector Shared Interrupt Select Register 130	INT_VSSEL130	0x0264	32	0x0000_0000
Vector Shared Interrupt Select Register 131	INT_VSSEL131	0x0268	32	0x0000_0000
Vector Shared Interrupt Select Register 132	INT_VSSEL132	0x026C	32	0x0000_0000
Vector Shared Interrupt Select Register 133	INT_VSSEL133	0x0270	32	0x0000_0000
Vector Shared Interrupt Select Register 134	INT_VSSEL134	0x0274	32	0x0000_0000
Vector Shared Interrupt Select Register 135	INT_VSSEL135	0x0278	32	0x0000_0000
Vector Shared Interrupt Select Register 136	INT_VSSEL136	0x027C	32	0x0000_0000
Vector Shared Interrupt Select Register 137	INT_VSSEL137	0x0280	32	0x0000_0000
Vector Shared Interrupt Select Register 138	INT_VSSEL138	0x0284	32	0x0000_0000
Vector Shared Interrupt Select Register 139	INT_VSSEL139	0x0288	32	0x0000_0000
Vector Shared Interrupt Select Register 140	INT_VSSEL140	0x028C	32	0x0000_0000
Vector shared interrupt selection register 141	INT_VSSEL141	0x0290	32	0x0000_0000
Vector Shared Interrupt Select Register 142	INT_VSSEL142	0x0294	32	0x0000_0000
Vector Shared Interrupt Select Register 143	INT_VSSEL143	0x0298	32	0x0000_0000
Software interrupt event register	INT_SWIER	0x029C	32	0x0000_0000
Event Enable Register	INT_EVTER	0x02A0	32	0x0000_0000
Interrupt enable register	INT_IER	0x02A4	32	0xFFFF_FFFF

10.5.1 NMI pin non-maskable interrupt control register (INT_NMICR)

NMI Interrupt Control Register

(INT_NMICR) reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								NFE N	-	SMPCLK [1. 0]	-	-	-	NMI TRG	

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Read "0", write "0" when writing	R/W
b7	NFEN	NMI digital filter enable	0: Disable digital filter function 1: Licensed digital filter function	R/W
b7~b6	Reserved	-	Read "0", write "0" when writing	R/W
b5~b4	SMPCLK[1:0]	Filter sampling clock selection	0 0: PCLK3 0 1: PCLK3/8 1 0: PCLK3/32 1 1:: PCLK3/64	R/W
b3~b1	Reserved	-	Read "0", write "0" when writing	R/W
b0	NMITRG	Trigger edge selection	0: falling edge 1: Rising edge	R/W

10.5.2 Non-Maskable Interrupt Enable

Register (INT_NMIENR) NMI

Interrupt Enable Register (INT_NMIENR)

Reset Value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	WDT ENR	BUS MEN R	REC CEN R	REP ENR	-	-	XTA LST PEN R	-	PVD 2EN R	PVD 1EN R	SWD TEN R	NMI ENR

position	Marker	Place Name	Function	Reading and writing
b31~b13	Reserved	-	Read "0", write "0" when writing	R/W
b12	Reserved	-	Read "0", write "0" when writing	R/W
b11	WDTENR	WDT underflow/refresh interrupt selection	0: Disable interrupts as non-maskable interrupt sources 1: Select interrupts as non-maskable interrupt sources	R/W
b10	BUSMENR	MPU main bus error interrupt selection	0: Disable interrupts as non-maskable interrupt sources 1: Select interrupts as non-maskable interrupt sources	R/W
b9	RECCENR	SRAM ECC checksum error interrupt selection	0: Disable interrupts as non-maskable interrupt sources 1: Select interrupts as non-maskable interrupt sources	R/W
b8	REPENR	SRAM parity error interrupt selection	0: Disable interrupts as non-maskable interrupt sources 1: Select interrupts as non-maskable interrupt sources	R/W
b7	Reserved	-	Read "0", write "0" when writing	R/W
b6	Reserved	-	Read "0", write "0" when writing	R/W
b5	XATLSTPENR	Detection of primary oscillator stop interrupt selection	0: Disable interrupts as non-maskable interrupt sources 1: Select interrupts as non-maskable interrupt sources	R/W
b4	Reserved	-	Read "0", write "0" when writing	R
b3	PVD2ENR	Low voltage detection PVD2 interrupt selection	0: Disable interrupts as non-maskable interrupt sources 1: Select interrupts as non-maskable interrupt sources	R/W
b2	PVD1ENR	Low voltage detection PVD1 interrupt selection	0: Disable interrupts as non-maskable interrupt sources 1: Select interrupts as non-maskable interrupt sources	R/W

b1	SWDTENR	SWDT underflow/refresh interrupt selection	0: Disable interrupts as non-maskable interrupt sources 1: Select interrupts as non-maskable interrupt sources	R/W
b0	NMIENR	NMI pin interrupt selection	0: Disable interrupts as non-maskable interrupt sources 1: Select interrupts as non-maskable interrupt sources	R/W

10.5.3 Non-Maskable Interrupt Flag Register (INT_NMIFR)

NMI Flag Register

(INT_NMIFR) reset value:

0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	WDT FR	BUS MFR	REC CFR	REP FR	-	-	XTA LST PFR		PVD 2FR	PVD 1FR	SWD TFR	NMI FR

position	Marker	Place Name	Function	Reading and writing
b31~b13	Reserved	-	Read "0", write "0" when writing	R
b12	Reserved	-	Read "0", write "0" when writing	R
b11	WDTFR	WDT underflow/refresh interrupt flag	0: No WDT underflow/refresh request occurred 1: WDT underflow/refresh application occurs	R
b10	BUSMFR	MPU main bus error interrupt flag	0: No MPU master bus error request occurred 1: MPU main bus error request occurred	R
b9	RECCFR	SRAM DED checksum error interrupt flag	0: No SRAM DED checksum error request occurred 1: SRAM DED check error request occurred	R
b8	REPFR	SRAM parity error interrupt flag	0: No SRAM parity error request occurred 1: SRAM parity error request occurred	R
b7	Reserved	-	Read "0", write "0" when writing	R
b6	Reserved	-	Read "0", write "0" when writing	R
b5	XTALSTPFR	Detect the primary oscillator stop interrupt flag	0: No detection of the main oscillator stop application occurred 1: The occurrence of detection of the main oscillator to stop the application	R
b4	Reserved	-	Read "0", write "0" when writing	R
b3	PVD2FR	Low voltage detection PVD2 interrupt flag	0: No low voltage detection PVD2 application occurred 1: Low voltage detection PVD2 application occurs	R
b2	PVD1FR	Low voltage detection PVD1 interrupt flag	0: No low voltage detection PVD1 application occurred 1: Low voltage detection PVD1 application occurs	R
b1	SWDTFR	SWDT underflow/refresh interrupt flag	0: No SWDT underflow/refresh request occurred 1: SWDT underflow/refresh application occurs	R
b0	NMIFR	NMI pin interrupt flag	0: No NMI pin application occurred 1: NMI pin application occurs	R

10.5.4 Non-maskable interrupt flag clear register (INT_NMICFR)

NMI Clear Flag Register

(INT_NMICFR) reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	WDT CFR	BUS MCF R	REC CCF R	REP CFR	-	-	XTA LST PCF R	-	PVD 2CF R	PVD 1CF R	SWD TCF R	NMI CFR

position	Marker	Place Name	Function	Reading and writing
b31~b13	Reserved	-	Read "0", write "0" when writing	R/W [Note 1]
b12	Reserved	-	Read "0", write "0" when writing	R/W [Note 1]
b11	WDTCFR	WDT underflow/refresh interrupt flag clear	0: Invalid 1: Clear the WDT underflow/refresh flag	R/W [Note 1]
b10	BUSMCFR	MPU main bus error interrupt flag cleared	0: Invalid 1: Clear the MPU main bus error flag	R/W [Note 1]
b9	RECCCFR	SRAM DED checksum error interrupt flag cleared	0: Invalid 1: Clear SRAM DED checksum error flag	R/W [Note 1]
b8	REPCFR	SRAM parity error interrupt flag cleared	0: Invalid 1: Clear SRAM parity error flag	R/W [Note 1]
b7	Reserved	-	Read "0", write "0" when writing	R/W [Note 1]
b6	Reserved	-	Read "0", write "0" when writing	R/W [Note 1]
b5	XTALSTPCFR	Detection of primary oscillator stop interrupt flag clear	0: Invalid 1: Clear the detection of the main oscillator stop sign	R/W [Note 1]
b4	Reserved	-	Read "0", write "0" when writing	R
b3	PVD2CFR	Low voltage detection PVD2 interrupt flag cleared	0: Invalid 1: Clear the low voltage detection PVD2 flag	R/W [Note 1]
b2	PVD1CFR	Low voltage detection PVD1 interrupt flag cleared	0: Invalid 1: Clear the low voltage detection PVD1 flag	R/W [Note 1]
b1	SWDTCFR	SWDT underflow/refresh interrupt flag clear	0: Invalid 1: Clear SWDT underflow/refresh flag	R/W [Note 1]
b0	NMICFR	NMI pin interrupt flag clear	0: Invalid 1: Clear NMI pin flags	R/W [Note 1]

[Note 1] Only "1" can be written, and "0" can be read out.

10.5.5 External pin interrupt control register (INT_EIRQCRx) (x=0~15)

EIRQ Control Register (INT_EIRQCRx)

reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	EFE N	-	EISMPCLK [1:0]	-	-	-	EIRQTRG[1 :0]	

position	Marker	Place Name	Function	Reading and writing
b31~b10	Reserved	-	Read "0", write "0" when writing	R/W
b9	Reserved	-	Read "0", write "0" when writing	R/W
b8	Reserved	-	Read "0", write "0" when writing	R/W
b7	EFEN	EIRQ digital filter enable	0: Disable digital filter function 1: Allow digital filter function	R/W
b6	Reserved	-	Read "0", write "0" when writing	R/W
b5~b4	EISMPCLK[1:0]	Filter sampling clock selection 0 1	0 0: PCLK3 0 1: PCLK3/8 1 0: PCLK3/32 1 1:: PCLK3/64	R/W
b3~b2	Reserved	-	Read "0", write "0" when writing	R/W
b1~b0	EIRQTRG[1:0]	Trigger selection 0 0: falling edge 0 1: Rising edge 1 0: Double edge 1 1:: Low level	0 0: falling edge 0 1: Rising edge 1 0: Double edge 1 1:: Low level	R/W

10.5.6 External pin interrupt flag register (INT_EIFR)

EIRQ Flag Register (INT_EIFR)

reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EIF R15	EIF R14	EIF R13	EIF R12	EIF R11	EIF R10	EIF R9	EIF R8	EIF R7	EIF R6	EIF R5	EIF R4	EIF R3	EIF R2	EIF R1	EIF R0

position	Marker	Place Name	Function	Reading and writing
b31~b10	Reserved	-	Read "0", write "0" when writing	R/W
b15~b0	EIFR	EIFR flag bit	0: EIRQ event did not occur, or write EIFCR bit clear bit 1: The selected EIRQ event occurs	R

10.5.7 External Pin Interrupt Flag Clear

Register (EICFR) EIRQ Flag Clear Register

(INT_EICFR) Reset Value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EICFR [15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b10	Reserved	-	Read "0", write "0" when writing	R/W
b15~b0	EICFR	EIFR clear bit	0: Writing "0" is invalid 1: Write "1" to clear INT_EIFR register	R/W

10.5.8 Interrupt Source Select Register

(INT_SEL0~31) Interrupt Source Select Register

(INT_SEL0~31) Reset Value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	INTSEL[8:0]							

position	Marker	Place Name	Function	Reading and writing
b31~b9	Reserved	-	Read "0", write "0" when writing	R/W
b8~b0	INTSEL[8:0]	Interrupt event request selection	9'h000~9'h1FE: 10.3.2 Events corresponding to the interrupt event request sequence number	R/W

10.5.9 Interrupt selection register (INT_SEL32~127)

Interrupt Source Select Register (INT_SEL32~127)

reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	INTSEL[8:0]

position	Marker	Place Name	Function	Reading and writing
b31~b9	Reserved	-	Read "0", write "0" when writing	R/W
b8~b0	INTSEL[8:0]	Interrupt event request selection	Select the events corresponding to the 10.3.2 interrupt event request serial number, respectively, the specific correspondence is as follows: INT_SEL32~INT_SEL37: Select the interrupt event request corresponding to 9'h000~9'h01F, other selections are invalid. INT_SEL38~INT_SEL43: Select the interrupt event request corresponding to 9'h020~9'h03F, other selections are invalid. INT_SEL44~INT_SEL49: Select the interrupt event request corresponding to 9'h040~9'h05F, other selections are invalid. INT_SEL50~INT_SEL55: Select the interrupt event request corresponding to 9'h060~9'h07F, other selections are invalid. INT_SEL56~INT_SEL61: Select the interrupt event request corresponding to 9'h080~9'h09F, other selections are invalid. INT_SEL62~INT_SEL67: Select the interrupt event request corresponding to 9'h0AO~9'h0BF, other selections are invalid. INT_SEL68~INT_SEL73: Select the interrupt event request corresponding to 9'hOC0~9'h0DF, other selections are invalid. INT_SEL74~INT_SEL79: Select the interrupt event request corresponding to 9'h0E0~9'h0FF, other selections are invalid. INT_SEL80~INT_SEL85: Select the interrupt event request corresponding to 9'h100~9'h11F, other selections are invalid. INT_SEL86~INT_SEL91: Select the interrupt event request corresponding to 9'h120~9'h13F, other selections are invalid. INT_SEL92~INT_SEL97: Select the interrupt event request corresponding to 9'h140~9'h15F, other selections are invalid. INT_SEL98~INT_SEL103: Select the interrupt event request corresponding to 9'h160~9'h17F, other selections are invalid. INT_SEL104~INT_SEL109: Select the interrupt event request corresponding to 9'h180~9'h19F, other selections are invalid. INT_SEL110~INT_SEL115: Select the interrupt event request corresponding to 9'h1AO~9'h1BF, other selections are invalid. INT_SEL116~INT_SEL121: Select the interrupt event request corresponding to 9'h1C0~9'h1DF, other selections are invalid. INT_SEL122~INT_SEL127: Select the interrupt event request corresponding to 9'h1E0~9'h1FF, other selections	R/W

The choice is invalid.

10.5.10 Vector shared interrupt selection registers (INT_VSSEL128~143)

Vector Sharing Interrup Source Select Register (INT_VSSEL128~143) reset

value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
VSE															
L31	L30	L29	L28	L27	L26	L25	L24	L23	L22	L21	L20	L19	L18	L17	L16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
VSE															
L15	L14	L13	L12	L11	L10	L9	L8	L7	L6	L5	L4	L3	L2	L1	L0

position	Marker	Place Name	Function	Reading and writing
b31~b0	VSEL [31:0]	Interrupt enable	INT_VSSEL128: Each bit enables the interrupt event request corresponding to 9'h000~9'h01F respectively. INT_VSSEL129: Each bit enables the interrupt event request corresponding to 9'h020~9'h03F respectively. INT_VSSEL130: Each bit enables the interrupt event request corresponding to 9'h040~9'h05F respectively. INT_VSSEL131: Each bit enables the interrupt event request corresponding to 9'h060~9'h07F respectively. INT_VSSEL132: Each bit enables the interrupt event request corresponding to 9'h080~9'h09F respectively. INT_VSSEL133: Each bit enables the interrupt event request corresponding to 9'h0A0~9'h0BF. INT_VSSEL134: Each bit enables the interrupt event request corresponding to 9'h0C0~9'h0DF respectively. INT_VSSEL135: Each bit enables the interrupt event request corresponding to 9'h0E0~9'h0FF respectively. INT_VSSEL136: Each bit enables the interrupt event request corresponding to 9'h100~9'h11F respectively. INT_VSSEL137: Each bit enables the interrupt event request corresponding to 9'h120~9'h13F respectively. INT_VSSEL138: Each bit enables the interrupt event request corresponding to 9'h140~9'h15F respectively. INT_VSSEL139: Each bit enables the interrupt event request corresponding to 9'h160~9'h17F respectively. INT_VSSEL140: Each bit enables the interrupt event request corresponding to 9'h180~9'h19F respectively. INT_VSSEL141: Each bit enables the interrupt event request corresponding to 9'h1A0~9'h1BF respectively. INT_VSSEL142: Each bit enables the interrupt event request corresponding to 9'h1C0~9'h1DF respectively. INT_VSSEL143: Each bit enables the interrupt event request corresponding to 9'h1E0~9'h1FF respectively. Refer to Table 10-2 Interrupt Event Request Sequence and Selection for the interrupt event request number.	R/W

10.5.11 Stop mode wake-up event enable register (INT_WUPEN)

Soft-standby Wake Up Enable Register (INT_WUPEN)

reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	RXW UEN	-	TMR 0WU EN	RTC PRD WUE N	RTC ALM WUE N	WKT MWU EN	CMP IOW UEN	PVD 2WU EN	PVD 1WU EN	SWD TWU EN	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EIRQWUEN [15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~27	Reserved	-	Read "0", write "0" when writing	R/W
b26	Reserved	-	Read "0", write "0" when writing	R/W
b25	RXWUEN	USART1_WUPI Wake-on-Stop Mode Enable	0: Wake up forbidden 1: Wake up permit	R/W
b24	Reserved	-	Read "0", write "0" when writing	R/W
b23	TMR0WUEN	TMR01_GCMA Stop Mode Wake-Up Enable	0: Wake up forbidden 1: Wake up permit	R/W
b22	RTCPRDWUEN	RTC_PRD Stop Mode Wake-Up Enable	0: Wake up forbidden 1: Wake up permit	R/W
b21	RTCALMWUEN	RTC_ALM Stop Mode Wake-Up Enable	0: Wake up forbidden 1: Wake up permit	R/W
b20	WKTMWUEN	WKTM_PRD Cycle Stop Mode Wake- Up Enable	0: Wake up forbidden 1: Wake up permit	R/W
B19	CMPI1WUEN	ACMP1 Stop Mode Wakeup Enable	0: Wake up forbidden 1: Wake up permit	R/W
B18	PVD2WUEN	PVD_PVD2 Stop Mode Wakeup Enable	0: Wake up forbidden 1: Wake up permit	R/W
B17	PVD1WUEN	PVD_PVD1 Stop Mode Wakeup Enable	0: Wake up forbidden 1: Wake up permit	R/W
B16	SWDTWUEN	SWDT_REFUDF Stop Mode Wakeup Enable	0: Wake up forbidden 1: Wake up permit	R/W
B15~b0	EIRQWUEN [15:0]	EIRQ Stop Mode Wake-Up Enable	0: Wake up forbidden 1: Wake up permit	R/W

10.5.12 Software interrupt/event register (INT_SWIER)

Software Interrupt & Event Register

(INT_SWIER) reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
SWI E31	SWI E30	SWI E29	SWI E28	SWI E27	SWI E26	SWI E25	SWI E24	SWI E23	SWI E22	SWI E21	SWI E20	SWI E19	SWI E18	SWI E17	SWI E16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SWI E15	SWI E14	SWI E13	SWI E12	SWI E11	SWI E10	SWI E9	SWI E8	SWI E7	SWI E6	SWI E5	SWI E4	SWI E3	SWI E2	SWI E1	SWI E0

position	Marker	Place Name	Function	Reading and writing
b31~b0	SWIE	Software interrupt/event register bits	0: Invalid 1: Software interruption events occur Note: A software interrupt/event occurs when a "1" is written. Clear after writing "0".	R/W

10.5.13 Event Enable Register

(INT_EVTER) Event Enable Register

(INT_EVTER) Reset Value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EVT E31	EVT E30	EVT E29	EVT E28	EVT E27	EVT E26	EVT E25	EVT E24	EVT E23	EVT E22	EVT E21	EVT E20	EVT E19	EVT E18	EVT E17	EVT E16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EVT E15	EVT E14	EVT E13	EVT E12	EVT E11	EVT E10	EVT E9	EVT E8	EVT E7	EVT E6	EVT E5	EVT E4	EVT E3	EVT E2	EVT E1	EVT E0

position	Marker	Place Name	Function	Reading and writing
b31~b0	EVTE	Event enable register bit	0: Event selection disabled 1: Event selection license	R/W

10.5.14 Interrupt Enable Register

(INT_IER) Interrupt Enable Register

(INT_IER) Reset Value: 0xFFFF_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
IER 31	IER 30	IER 29	IER 28	IER 27	IER 26	IER 25	IER 24	IER 23	IER 22	IER 21	IER 20	IER 19	IER 18	IER 17	IER 16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
IER 15	IER 14	IER 13	IER 12	IER 11	IER 10	IER 9	IER 8	IER 7	IER 6	IER 5	IER 4	IER 3	IER 2	IER 1	IER 0

position	Mark er	Place Name	Function	Reading and writing
b31~b0	IER	Interrupt enable register bit	<p>When disabled, the interrupt event request selected by the selection register INTSEL[8:0] of each interrupt vector will not be accepted by the NVIC.</p> <p>0: Interrupt event request selected by INTSEL[8:0] with software interrupt event request is disabled</p> <p>1: Interrupt event request selected by INTSEL[8:0] with software interrupt event request is licensed</p>	R/W

10.6 Precautions for use

For a description of ARM core interrupts, please refer to the ARM manual ARM

Processor

Cortex®-M4

Technical

Reference Manual (ARM DDI 0439D).

11 Automatic Operating System (AOS)

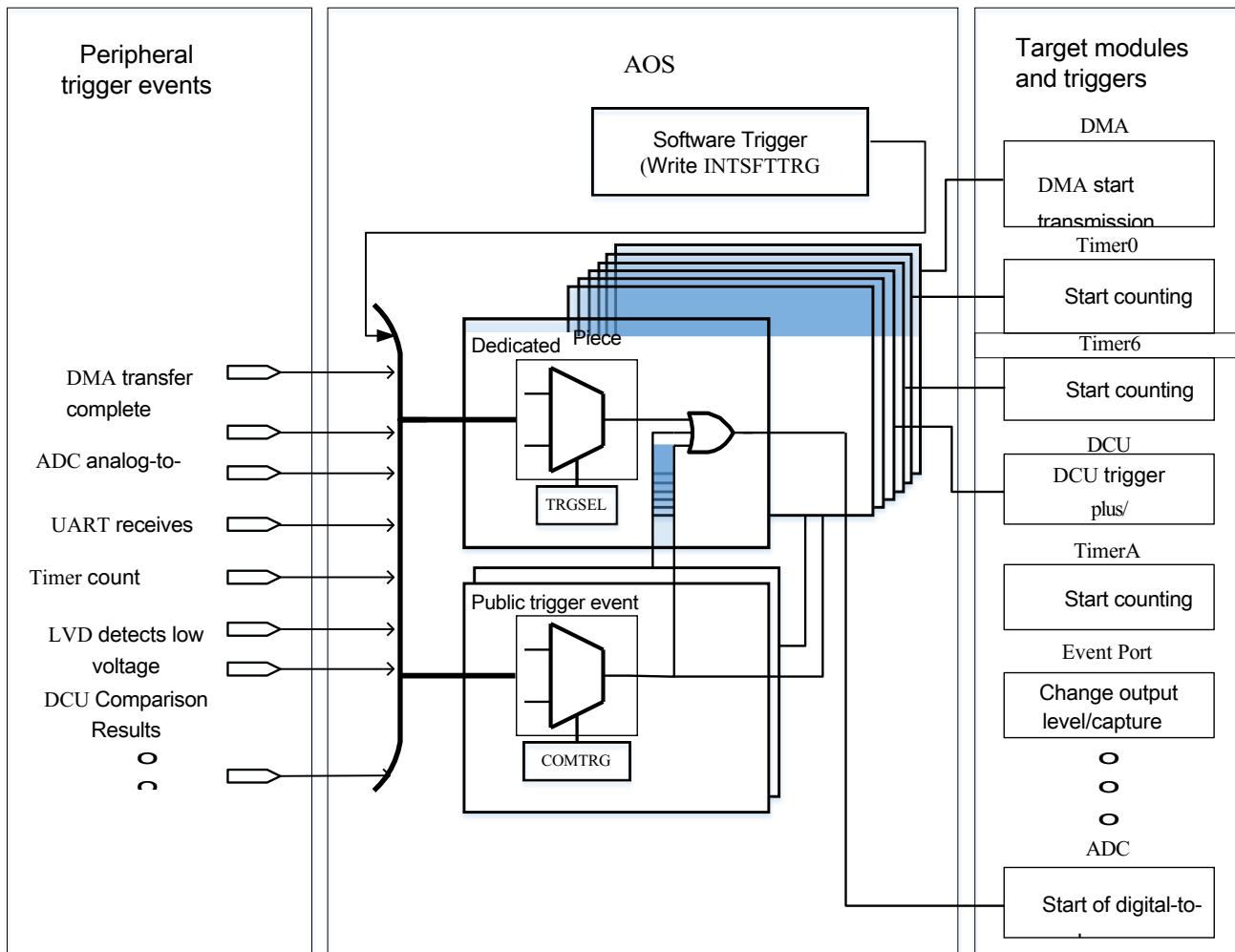
11.1 Introduction

Automatic Operation System (AOS) is used to link peripheral hardware circuits without the help of CPU. Events generated by peripheral circuits are used as AOS **Source**, such as comparison match and timing overflow of timer, period signal of RTC, various states of sending and receiving data of communication module (idle, full of received data, end of sending data, empty of sending data) end of conversion of ADC, etc., to trigger other peripheral circuits to act. The action of the triggered peripheral circuit is called AOS **Target**.

11.1.1 Function Overview

- There are 195 AOS sources, except for special restrictions, each AOS target can select one of them as the trigger source, and two additional trigger sources can be selected through the Common Trigger Source Selection Register 1 and Common Trigger Source Selection Register 2, and the AOS target can be triggered when a trigger event occurs in any of the three trigger sources. All AOS targets share these two common trigger sources.
- It can be triggered by the hardware of the peripheral circuit or by software through writing registers.
- Peripheral circuits capable of acting as AOS targets act as follows:
 - 4 DCU trigger targets for triggering DCU1~DCU4
 - 9 DMA trigger targets for two 4-channel DMA start data transfers and one DMA event to trigger a channel reset
 - 2 advanced control timers (Timer6) trigger targets
 - 1 general-purpose timer 0 (Timer0) trigger target
 - 2 Event **Port** trigger targets, where Event Port Group1 and Event Port Group2 share an AOS target, and Event Port Group3 and Event Port Group4 share an AOS target
 - 2 general-purpose timers A (TimerA) trigger targets
 - 1 temperature sensor (OTS) trigger target
 - 2 groups of 2 AD trigger targets per group for AD1~AD2 sequence triggering

11.1.2 Module schematic



11.2 Function Description

11.2.1 AOS Source Event List

The AOS source event numbers are shown in Table 10-2 in the Interrupt Event Request Number section of the [Interrupt Controller (INTC) chapter. events marked with a check in the "Can be selected as event" column in the table can be used as AOS sources.

11.2.2 AOS Target List

Modules	Action
DCU1	Trigger addition/subtraction operations
DCU2	Trigger addition/subtraction operations
DCU3	Trigger addition/subtraction operations
DCU4	Trigger addition/subtraction operations
DMA1	Channel 0 Start Transmission Channel 1 Start Transmission Channel 2 Start Transmission Channel 3 starts transmission
DMA2	Channel 0 Start Transmission Channel 1 Start Transmission Channel 2 Start Transmission Channel 3 starts transmission
DMA1&2	Event triggered channel reset
Timer6	Start counting
Timer0	Start counting
Event Port	Event Port1&2 Trigger Action Event Port3&4 Trigger Action
TimerA	Start counting/capture
OTS	Start temperature measurement
ADC1	Start analog-to-digital conversion
ADC2	Start analog-to-digital conversion

11.3 Action Description

11.3.1 Dedicated trigger source

The peripheral circuit module with AOS target has a dedicated peripheral trigger source selection register for each AOS target. When the event number corresponding to the AOS source is written in this register, the AOS target selects this AOS source as the trigger source. When the event of the AOS source occurs, this event will be passed to the AOS target through the AOS, and the peripheral circuit as the AOS target will start to act according to its own setting.

11.3.2 Common trigger source

In addition to the dedicated peripheral trigger source selection registers for each AOS target, the AOS is configured with two common trigger source selection registers (**AOS_COMTRG1**,**AOS_COMTRG2**). This is used to implement the function of multiple AOS sources triggering the same AOS target. When using, first enable the common trigger source in the AOS target dedicated peripheral trigger source selection register, and then write the event number corresponding to the AOS source in the common trigger source selection register. When the event of the AOS source occurs, the event will be passed to the AOS target through the public trigger source of the AOS, and the peripheral circuit of the AOS target will start to act according to its own setting. When both the dedicated trigger source and the common trigger source are set, up to three AOS sources can trigger the same AOS target at the same time, and the AOS target will be triggered when any one of the three AOS sources triggers an event.

These two common trigger sources are shared by all AOS targets. Therefore, when other AOS targets do not use the event selected by the public trigger source selection register, it is necessary to set the public trigger source enable position to invalid in its dedicated peripheral trigger source selection register to prevent wrong trigger action.

11.4 Register Description

Register List

Register base address: 0x4001_0800

Abb	Na	Offset
revi	me	Address
atio		
ns		
INTSFTTRG	Peripheral trigger event register	0x00
DCU_TRGSELx(x=1~4)	DCU Trigger Source Selection Register	0x04,0x08,0x0C,0x10
DMA1_TRGSELx(x=0~3)	DMA1 Transmit Start Trigger Source Selection Register	0x14,0x18,0x1C,0x20
DMA2_TRGSELx(x=0~3)	DMA2 Transmit Start Trigger Source Selection Register	0x24,0x28,0x2C,0x30
DMA_TRGSELRC	DMA channel reset trigger source selection register	0x34
TMR6_HTSSRx(x=0~1)	Timer6 hardware trigger event selection register	0x38,0x3C
TMR0_HTSSR	Timer0 Trigger Select Register	0x40
PEVNTTRGSR12	Event Port1,2 Trigger source selection register	0x44
PEVNTTRGSR34	Event Port3,4 Trigger source selection register	0x48
TMRA_HTSSR0	TimerA Internal trigger event selection register 0	0x4C.
TMRA_HTSSR1	TimerA Internal trigger event selection register 1	0x50
OTS_TRG	OTS Trigger Source Selection Register	0x54
ADC1_ITRGSELRx(x=0,1)	A/D1 Start on-chip trigger source selection register	0x58, 0x5C
ADC2_ITRGSELRx(x=0,1)	A/D2 Start on-chip trigger source selection register	0x60, 0x64
AOS_COMTRG1	Common trigger source selection register 1	0x68
AOS_COMTRG2	Common trigger source selection register 2	0x6C

11.4.1 Peripheral Trigger Event

Register (INTSFTTRG) Register Description:

Writing this register will generate a trigger event.

Offset address: 0x00

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															

position	Marker	Place Name	Function	Reading and writing
b31~b1	Reserved	-	Read "0" for read/write "0" for write	R/W
b0	SFTG	Software Trigger	0: No software triggered events are generated 1: Generate a software trigger event The location 1 will generate a peripheral trigger event, software write 0 invalid	W

11.4.2 DCU trigger source selection register (DCU_TRGSELx)(x=1~4)

Register Description: After the DCU selects the hardware trigger start mode, the number of the event to be triggered is written into this register, and when the event of the peripheral circuit corresponding to the number occurs, the DCU will be triggered by the event to start and perform the operation. When DCU_TRGSEL1 writes the event number, DCU1 will be triggered when the numbered event occurs; when DCU_TRGSEL2 writes the event number, DCU2 will be triggered when the numbered event occurs; when DCU_TRGSEL3 writes the event number, DCU3 will be triggered when the numbered event occurs; when DCU_TRGSEL4 writes the event number, DCU3 will be triggered when the numbered event occurs; when DCU_TRGSEL4 writes the event number, DCU4 will be triggered when the numbered event occurs. event occurs, DCU4 will be triggered.

Offset address: 0x04, 0x08, 0x0C,

0x10 Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]		Reserved													
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							

position	Marker	Place Name	Function	Reading and writing

b31	COMEN[1]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG2 to trigger DCU 1: Allow the public trigger event of AOS_COMTRG2 to trigger the DCU	R/W
b30	COMEN[0]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG1 to trigger DCU 1: Allow public trigger event of AOS_COMTRG1 to trigger DCU	R/W
b29~b9	Reserved	-	Read "0" for read/write "0" for write	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected Please refer to the Interrupt Controller (INTC) section for specific numbering.	R/W

11.4.3 DMA1 Transmit Start Trigger Source Select Register (DMA1_TRGSELx) (x=0~3)

Register Description: After DMA1 selects the hardware trigger start mode, the number of the event that will generate the trigger is written into this register, and when the event of the peripheral circuit corresponding to the number occurs, DMA1 will be triggered by the event to start and transfer.

Offset address: 0x14, 0x18, 0x1C,

0x20 Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]		Reserved													
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							

position	Marker	Place Name	Function	Reading and writing
b31	COMEN[1]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG2 to trigger DMA1 transmission 1: Allow public trigger event of AOS_COMTRG2 to trigger DMA1 transfer	R/W
b30	COMEN[0]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG1 to trigger DMA1 transmission 1: Allow public trigger event of AOS_COMTRG1 to trigger DMA1 transfer	R/W
b29~b9	Reserved	-	Read "0" for read/write "0" for write	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Select the event number that initiates the corresponding channel for transmission. Refer to the Interrupt Controller (INTC) chapter for the specific numbering.	R/W

11.4.4 DMA2 transmit start trigger source selection register (DMA2_TRGSELx) (x=0~3)

Register Description: After DMA2 selects the hardware trigger start mode, the number of the event that will generate the trigger is written into this register, and when the event of the peripheral circuit corresponding to the number occurs, DMA2 will be triggered by the event to start and transfer.

Offset address: 0x24, 0x28, 0x2C,,0x30

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]		Reserved													

J															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							

position	Marker	Place Name	Function	Rea ding and writin g
b31	COMEN[1]	Public trigger enable	0: Disable AOS_COMTRG2's public trigger event to trigger DMA2 transfers 1: Allow public trigger event of AOS_COMTRG2 to trigger DMA2 transfer	R/W
b30	COMEN[0]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG1 to trigger DMA2 transmission 1: Allow public trigger event of AOS_COMTRG1 to trigger DMA2 transfer	R/W
b29~b9	Reserved	-	Read "0" for read/write "0" for write	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Select the event number that initiates the corresponding channel for transmission. Refer to the Interrupt Controller (INTC) chapter for the specific numbering.	R/W

11.4.5 DMA channel reset trigger source selection register (DMA_TRGSELRC)

DMA1 and DMA2 share this register.

Offset address: 0x34

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							

position	Marker	Place Name	Function	Reading and writing
b31	COMEN[1]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG2 to trigger the DMA channel reset 1: Allow public trigger event of AOS_COMTRG2 to trigger DMA channel reset	R/W
b30	COMEN[0]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG1 to trigger the DMA channel reset 1: Allow public trigger event of AOS_COMTRG1 to trigger DMA channel reset	R/W
b29~b9	Reserved	-	Read "0" for read/write "0" for write	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Select the event number that triggers the channel for resetting. Refer to the Interrupt Controller (INTC) chapter for the specific numbering. DMA_1, DMA_2 share a common reset trigger source.	R/W

11.4.6 Timer6 hardware trigger event selection register (TMR6_HTSSRx) (x=0~1)

Register Description: After Timer6 selects the hardware trigger start mode, the number of the event to be triggered is written into this register, and Timer6 will be triggered by this event when the event of the peripheral circuit corresponding to the number occurs.

Offset address: 0x38,

0x3C Reset value:

0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							

position	Marker	Place Name	Function	Reading and writing
b31	COMEN[1]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG2 to trigger TMR6 1: Allow the public trigger event of AOS_COMTRG2 to trigger TMR6	R/W
b30	COMEN[0]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG1 to trigger TMR6 1: Allow public trigger event of AOS_COMTRG1 to trigger TMR6	R/W
b29~b9	Reserved	-	Read "0" for read/write "0" for write	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Trigger source number writing Please refer to the Interrupt Controller (INTC) section for specific numbering.	R/W

Caution:

--Trigger Select Register (TMR6_HTSR0~1) The trigger select registers (TMR6_HTSR0~1)

are 2 cell-independent registers, common to all 3 cells of Timer6.

11.4.7 Timer0 Hardware Trigger Event Select Register (TMR0_HTSSR)

Register Description: After Timer0 selects the hardware trigger start mode, the number of the event to be triggered is written into this register, and Timer0 will be triggered by this event when the event of the peripheral circuit corresponding to the number occurs.

Offset address: 0x40

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]															Reserved
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							

position	Marker	Place Name	Function	Reading and writing
b31	COMEN[1]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG2 to trigger TMR0 1: Allow the public trigger event of AOS_COMTRG2 to trigger TMR0	R/W
b30	COMEN[0]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG1 to trigger TMR0 1: Allow public trigger event of AOS_COMTRG1 to trigger TMR0	R/W
b29~b9	Reserved	-	Read "0" for read/write "0" for write	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Trigger source number writing Please refer to the Interrupt Controller (INTC) section for specific numbering.	R/W

Caution:

–Trigger Select Register (TMR0_HTSSR) The Trigger Select Register (TMR0_HTSSR) is a separate register, common to Timer0 of 2 units.

11.4.8 Event Port Trigger Source Select Register (PEVNTRGSR12, PEVNTRGSR34)

PEVNTRGSR12 sets the trigger source for Event Port1 and 2, PEVNTRGSR34 sets the trigger source for Event Port3 and 4.

Offset address: 0x44,

0x48 Reset value:

0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]		Reserved													
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							

position	Marker	Place Name	Function	Reading and writing
b31	COMEN[1]	Public trigger enable	0: Disable public trigger event triggering Event Port of AOS_COMTRG2 1: Allow public trigger events of AOS_COMTRG2 to trigger Event Port	R/W
b30	COMEN[0]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG1 to trigger Event Port 1: Allow public trigger events of AOS_COMTRG1 to trigger Event Port	R/W
b29~b9	Reserved	-	Read "0" for read/write "0" for write	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Set the corresponding event number to trigger the Event Port to output the specified level, or latch the I/O port input state. PEVNTRGSR12 sets the trigger source for Event Port 1 and 2. PEVNTRGSR34 sets the trigger source for Event Port3 and 4. Refer to the Interrupt Controller (INTC) chapter for specific numbering.	R/W

11.4.9 TimerA Internal trigger event selection register 0 (TMRA_HTSSR0)

Register Description: After TimerA selects the hardware trigger start mode, the number of the event to be triggered is written into this register, and TimerA will be triggered by this event when the event of the peripheral circuit corresponding to the number occurs.

Offset address: 0x4C

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]															Reserved
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								CNTTRGSEL[8:0]							

position	Marker	Place Name	Function	Reading and writing
b31	COMEN[1]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG2 from triggering the TMRA counter 1: Allow the common trigger event of AOS_COMTRG2 to trigger the TMRA counter	R/W
b30	COMEN[0]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG1 from triggering the TMRA counter 1: Allow public trigger event of AOS_COMTRG1 to trigger TMRA counter	R/W
b29~b9	Reserved	-	Read "0" for read/write "0" for write	R/W
b8~b0	CNTTRGSEL[8:0]	Counter trigger event trigger source selection	Counter trigger event trigger source number write Please refer to the Interrupt Controller (INTC) section for specific numbering.	R/W

Caution:

- The internal trigger event selection registers (TMRA_HTSSR0~1) are two independent registers, common to the 6-cell TimerA.

11.4.10 TimerA Internal trigger event selection register 1 (TMRA_HTSSR1)

Register Description: After TimerA selects the hardware trigger start mode, the number of the event to be triggered is written into this register, and TimerA will be triggered by this event when the event of the peripheral circuit corresponding to the number occurs.

Offset address: 0x50

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]															Reserved
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								ICPTRGSEL[8:0]							

position	Marker	Place Name	Function	Reading and writing
b31	COMEN[1]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG2 to trigger TMRA capture action 1: Allow public trigger events of AOS_COMTRG2 to trigger TMRA capture actions	R/W
b30	COMEN[0]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG1 to trigger TMRA capture action 1: Allow public trigger event of AOS_COMTRG1 to trigger TMRA capture action	R/W
b29~b9	Reserved	-	Read "0" for read/write "0" for write	R/W
b8~b0	ICPTRGSEL[8:0]	Capture action trigger event trigger source selection	Capture action trigger event trigger source number write Please refer to the Interrupt Controller (INTC) section for specific numbering.	R/W

11.4.11 OTS Trigger source selection register (OTS_TRG)

Register Description: After the OTS selects the hardware trigger start mode, the number of the event that will generate the trigger is written into this register, and the OTS will be triggered by the event when the peripheral circuit event corresponding to the number occurs.

Offset address: 0x54

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]															Reserved
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							

position	Marker	Place Name	Function	Reading and writing

b31	COMEN[1]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG2 to trigger OTS 1: Allow public trigger events of AOS_COMTRG2 to trigger OTS	R/W
b30	COMEN[0]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG1 to trigger OTS 1: Allow public trigger event of AOS_COMTRG1 to trigger OTS	R/W
b29~b9	Reserved	-	Read "0" for read/write "0" for write	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Selects the trigger source number for the hardware trigger start. Please refer to the Interrupt Controller (INTC) section for specific numbering.	R/W

11.4.12 A/D1 conversion start on-chip trigger source selection register

ADC1_ITRGSELRx(x=0,1)

Register Description: After ADC1 selects the hardware trigger start mode, the number of the event to be triggered is written into this register, and ADC1 will be triggered by this event when the event of the peripheral circuit corresponding to the number occurs.

Offset address: 0x58,

0x5C Reset value:

0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							

position	Marker	Place Name	Function	Reading and writing
b31	COMEN[1]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG2 to trigger ADC1 1: Allow the public trigger event of AOS_COMTRG2 to trigger ADC1	R/W
b30	COMEN[0]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG1 to trigger ADC1 1: Allow the public trigger event of AOS_COMTRG1 to trigger ADC1	R/W
b29~b9	Reserved	-	Read "0" for read/write "0" for write	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected Please refer to the Interrupt Controller (INTC) section for specific numbering.	R/W

11.4.13 A/D2 conversion start on-chip trigger source selection register

ADC2_ITRGSELRx(x=0,1)

Register Description: After ADC2 selects the hardware trigger start mode, the number of the event to be triggered is written into this register, and ADC2 will be triggered by the event when the event of the peripheral circuit corresponding to the number occurs.

Offset address: 0x60,

0x64 Reset value:

0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	Reserved														

J															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TRGSEL[8:0]							

position	Marker	Place Name	Function	Reading and writing
b31	COMEN[1]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG2 to trigger ADC2 1: Allow the public trigger event of AOS_COMTRG2 to trigger ADC2	R/W
b30	COMEN[0]	Public trigger enable	0: Disable the public trigger event of AOS_COMTRG1 to trigger ADC2 1: Allow the public trigger event of AOS_COMTRG1 to trigger ADC2	R/W
b29~b9	Reserved	-	Read "0" for read/write "0" for write	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected Please refer to the Interrupt Controller (INTC) section for specific numbering.	R/W

11.4.14 Common trigger source selection register 1 (AOS_COMTRG1)

Register Description: Write the number of the event that will generate the trigger in AOS_COMTRG1. If the COMEN[1] bit of the dedicated trigger source selection register of one or more AOS targets has a value of 1 when the peripheral circuit event corresponding to the number occurs, the peripheral circuit event corresponding to the number will trigger this one or more AOS targets to start.

Offset address: 0x68

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								COMTRG[8:0]							

position	Marker	Place Name	Function	Reading and writing
b31~b9	Reserved	-	Read "0" for read/write "0" for write	R/W
b8~b0	COMTRG[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected Please refer to the Interrupt Controller (INTC) section for specific numbering.	R/W

11.4.15 Common trigger source selection register 2 (AOS_COMTRG2)

Register Description: Write the number of the event that will generate the trigger in AOS_COMTRG2. If the COMEN[0] bit of the dedicated trigger source selection register of one or more AOS targets has a value of 1 when the peripheral circuit event corresponding to the number occurs, the peripheral circuit event corresponding to the number will trigger this one or more AOS targets to start.

Offset address: 0x6C

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								COMTRG[8:0]							

position	Marker	Place Name	Function	Reading and writing
b31~b9	Reserved	-	Read "0" for read/write "0" for write	R/W
b8~b0	COMTRG[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected	R/W

Please refer to the Interrupt Controller (INTC) section for
specific numbering.

12 Keyboard Scan Control Module (KEYSCAN)

12.1 Introduction

This product is equipped with a keyboard control module (KEYSCAN) 1 unit. the KEYSCAN module supports keyboard array (row and column) scanning, the columns are driven by independent scan output KEYOUT_m ($m=0\sim7$), while the row KEYIN_n ($n=0\sim15$) is detected as EIRQ_n ($n=0\sim15$) input. This module implements the key recognition function by the line scan query method.

KEYSCAN main features:

- EIRQ0~EIRQ15 can be independently selected as line inputs for the keyboard array.
- KEYOUT can be selected by register.
- The keypad array is scanned by sequentially outputting low levels at regular intervals.
- The scan time can be set.
- IRQ interrupt detection stops the scan and locates the pressed key based on the SSR.INDEX value and the IRQ interrupt flag (INT_EIFR.EIFR).

12.2 KEYSCAN system block diagram

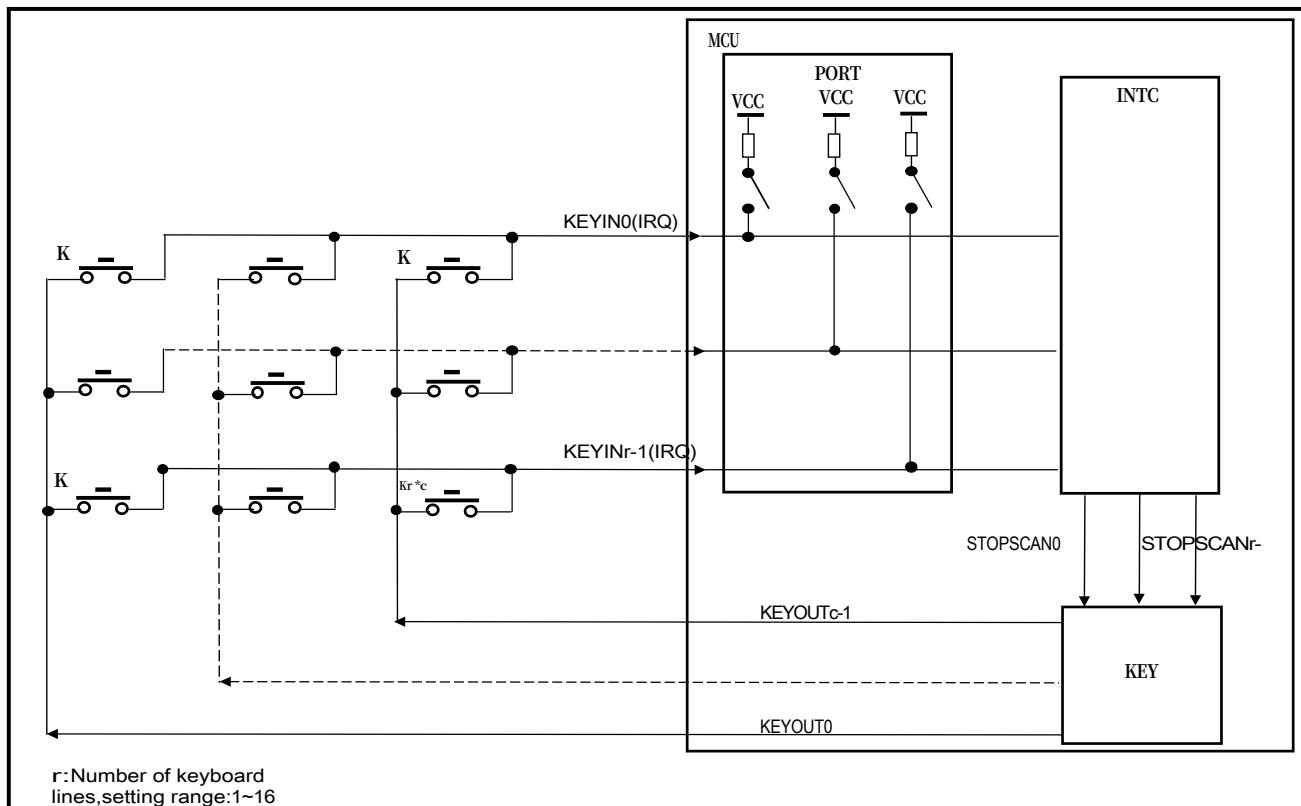


Figure 12-1 KEYSCAN system
block diagram

12.3 Pin

Descript

Table 12-1 KEYS CAN Pin Descriptions

ion

Foot r Name	Dire ctio n	Functio n Descript ion
KEYINn	Input	Keyboard line input signal
KEYOUTm	Output	Keyboard column output signal

n:0~15 m:0~7

12.4 Function Description

This chapter will explain the keyboard scanning function and the key recognition function in detail.

12.4.1 Key recognition function

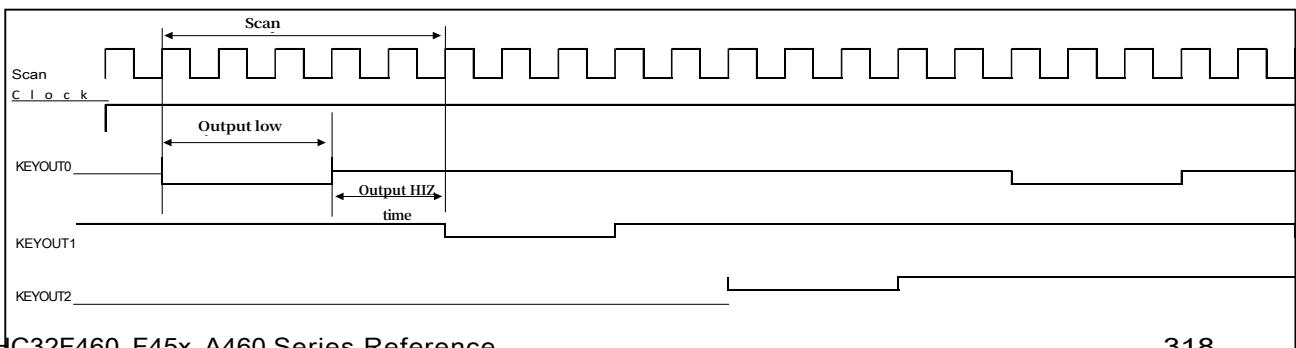
When a key is pressed, the rows and columns of the keyboard are shorted and the rows generate a falling edge, which generates an EIRQ interrupt flag that locates the currently pressed key by comparing the value of the interrupt flag bit (INT_EIFR.EIFR) with SSR_INDEX[2:0].

KEYINSEL[15:0], KEYIN can be independently selected from EIRQ0~EIRQ15, and the KEYOUT pins can be selected by register SCR.KEYOUTSEL[2:0], so that the number of rows of the keyboard can be flexibly selected, and the maximum keyboard array of 16 rows*8 columns can be supported.

12.4.2 Keyboard scanning function

The keyboard scan function is to continuously cycle the low output to the columns of the keyboard array so that when a key is pressed, the corresponding EIRQ interrupt flag is generated.

SEN is set to 1, KEYOUT0 outputs a low level, KEYOUT1~KEYOUTn (n is set by SCR.T_HIZ[2:0]), KEYOUT1 outputs a low level and the rest of KEYOUT pins are HIZ, and so on. When a key is pressed and the EIRQ interrupt flag is generated, the keyboard scanning function stops, and the corresponding interrupt flag is cleared and the scanning is restarted automatically.



**Figure 12-2 Schematic
diagram of keyboard
scanning function**

12.4.3 Precautions for use

This module drives the keyboard column, while the keyboard row detection is implemented by the external EIRQ function of the Interrupt Control Module (INTC). The EIRQ needs to select the falling edge detection and enable the digital filtering function to set the appropriate filtering time.

If you use this function in STOP mode, you need to set the scan-related parameters and select the internal low-speed oscillator LRC or the external low-speed oscillator XTAL32 clock as the scan clock.

If using the chip's internal pull-up resistor, please refer to PORT characteristics to select the appropriate scan time and filter time.

12.5 Register Description

KEYSCAN_BASE_ADDR: 0x4005_0C00

**Table 12-2 List of KEYS defense
registers**

Register Name	Sym bols	Offset Address	Bit width	Reset value
KEYSCAN Scan Control Register	KEYSCAN_SCR	0x00	32	0x0000_0000
KEYSCAN Scan Enable Register	KEYSCAN_SER	0x04	32	0x0000_0000
KEYSCAN Scan Status Register	KEYSCAN_SSR	0x08	32	0x0000_0000

12.5.1 KEYS defense Scan Control Register (KEYSCAN_SCR)

KEYSCAN Scan Control

Register Offset Address: 0x00

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
T_HIZ[2:0]				T_LLEVEL[4:0]			-	-	CKSEL[1:0]	-			KEYOUTSEL[2:0]		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
KEYINSEL[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b29	T-HIZ [2:0]	Output HIZ time	KEYOUT output low level HIZ time (number of scan clocks) Scan period = output low level time + output HIZ time Set value: number of HIZ periods 000b: 4 001b: 8 010b: 16 011b: 32 100b: 64 101b: 256 110b: 512 111b: 1024	R/W
			Note: SCR.T-HIZ[2:0] can only be set valid when SER.SEN=0	
b28~b24	T_LLEVEL[4:0]	Output low time	KEYOUT output low time (number of scan clocks) Scan period = output low time + output HIZ time Output low time = number of scanned clocks to the T_LLEVEL power of 2 T-LLEVEL[4:0] can only be set when SER.SEN=0, and 00000b and 00001b settings are prohibited, the maximum value can be set to 11000b.	R/W
b23~b22	Reserved	-	Read "0", write "0" when writing	R
b21~b20	CKSEL[1:0]	Scan clock source selection bit	Scan clock source selection bit 00b:System clock HCLK 01b: Internal low-speed oscillator LRC 10b:External low-speed oscillator XTAL32 11b:Set Prohibition	R/W
			Note: SCR.CKSEL[1:0] can only be set valid when SER.SEN=0	

b19	Reserved	-	Read "1", write "0" when writing	R
b18~b16	KEYOUTSEL[2:0]	Output Selection	KEYOUT output selection bit set value: Output selection 000b: Prohibit	R/W
			001b: KEYOUT0~KEYOUT1	
			010b: KEYOUT0~KEYOUT2	
			011b: KEYOUT0~KEYOUT3	
			100b: KEYOUT0~KEYOUT4	
			101b: KEYOUT0~KEYOUT5	
			110b: KEYOUT0~KEYOUT6	

111b: KEYOUT0~KEYOUT7

Note: SCR.KEYOUTSEL[2:0] can only be set valid when SER.SEN=0

b15~b0	KEYINSEL[15:0]	Line input selection bit	Line input selection bit, the selected line is used as a line of the keyboard array and is detected as EIRQn (n: 0~15) KEYINSEL[n]=0: KEYINSEL[n] is not used as a line of the keyboard array KEYINSEL[n]=1: KEYINSEL[n] is used as a line of the keyboard array n: range 0~15 Note: SCR.KEYINSEL[15:0] can only be set when SER.SEN=0	R/W
--------	----------------	--------------------------	---	-----

12.5.2 KEYS defense Register (KEYSCAN_SER)

KEYSCAN Scan Enable Register

Offset Address: 0x04

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SEN

position	Marker	Place Name	Function	Reading and writing
b31~b30	Reserved	-	Read "0", write "0" when writing	R
b0	SEN	Scan enable bit	Scan enable bit 0: Scan disable 1: Scan Enable	R/W

12.5.3 KEYS defense Status Register (KEYSCAN_SSR)

KEYSCAN Scan Status Register

Offset Address: 0x08

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	INDEX[2:0]

position	Marker	Place Name	Function	Reading and writing
b31~b3	Reserved	-	Read "0", write "0" when writing	R
b2~b0	INDEX[2:0]	Current operating SCAN pin index bit	The current working SCAN pin index bit 000: The current working SCAN pin is KEYOUT0 001: The current working SCAN pin is KEYOUT1 010: The current working SCAN pin is KEYOUT2 011: The current working SCAN pin is KEYOUT3 100: The current working SCAN pin is KEYOUT4 101: The current working SCAN pin is KEYOUT5 110: The current working SCAN pin is KEYOUT6 111: The current working SCAN pin is KEYOUT7 Note: SSR.INDEX[2:0] bits are read-only registers, and the data read is meaningful only when SER.SEN=1	R

13 Storage Protection Unit (MPU)

13.1 Introduction

The MPU provides protection for memory and can improve system security by blocking unauthorized access. The chip has four built-in MPU units for the host and

Mod ules	Content
ARM MPU	Storage protection unit for CPU 8 regions, see ARM MPU description for details
System DMA_1 MPU: SMPU1	Storage protection unit for system DMA_1 16 regions, 8 regions dedicated to system DMA, 8 regions shared by all DMAs
System DMA_2 MPU: SMPU2	Storage protection unit for system DMA_2 16 regions, 8 regions dedicated to system DMA, 8 regions shared by all DMAs
USBFS-DMA MPU: FMPU	USBFS-DMA's memory protection unit 8 area, shared

The ARM MPU provides CPU access control to the entire 4G address space, which is described in the following section.

SMPU1/SMPU2/FMPU provide read/write access control to all 4G address space for System DMA_1/System DMA_2/USBFS-DMA, respectively. The MPU action can be set to ignore/bus error/non-maskable interrupt/reset when access to the prohibited space occurs.

IPMPU provides access control to the system IP and security-related IPs when in unprivileged mode.

13.2 Function Description

13.2.1 Regional Range Setting

The MPU manages the storage space in terms of permissions on a region-by-region basis. Each area can set the base address and area size independently, and the range can be set from 32Byte to 4GByte, the size must be 2^n Byte ($n=5\sim32$) and the corresponding base address low n bits is 0.

The address space that is not covered by any area is called the background area.

13.2.2 Permission settings

Each area, including the background area, can be set independently for each DMA to allow reads/prohibit reads and allow writes/prohibit writes. If address overlap occurs between different areas, the set disable takes precedence.

13.2.3 MPU Motion Selection

When a prohibited access occurs, the access is ignored (read access read to 0, write access ignored) while the corresponding action can be set, which can be set as follows

- ignore
- Bus error
- Non-maskable interrupts
- Reset

13.2.4 Start MPU

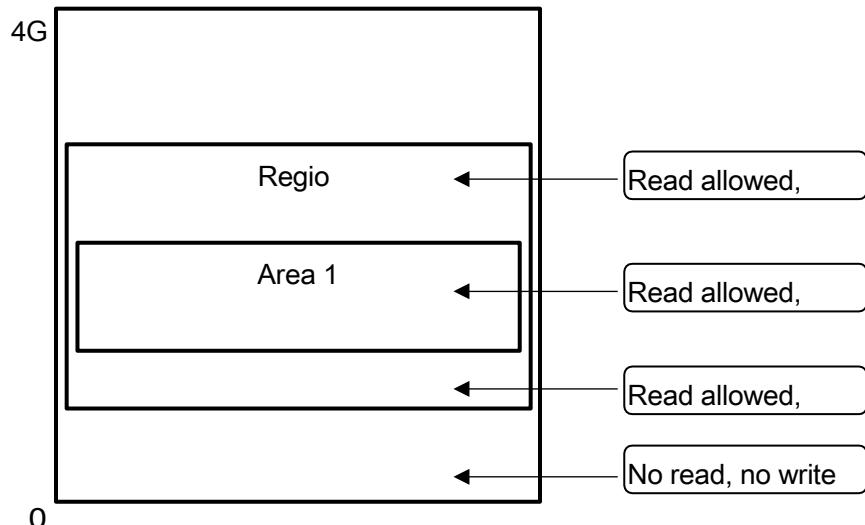
SMPU1/SMPU2/FMPU can be enabled independently.

It is recommended to enable the MPU after setting the area range/permission setting/action selection.

13.3 Application examples

13.3.1 Allow only partial space access

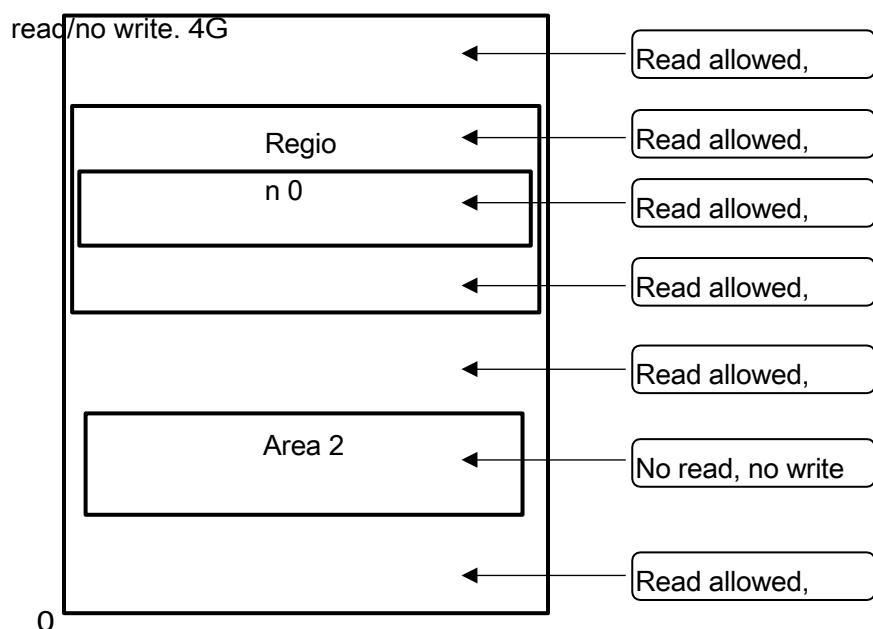
Example: Set the background area permissions to no read/no write, area 0 to allow read/allow write, area 1 to allow read/no write, and area 0 range over area 1.



13.3.2 Block access to only some spaces

Example: Set the background area permissions to allow read/prohibit write, area 0 to allow read/allow write, area 1 to allow read/prohibit

Stop write, area 0 overrides area 1, area 2 is set to no



13.4 Register Description

The registers of this module can only be

set by the CPU. MPU base address:

Offset Address	Register Name	Initial Value	Name	Write protection
+00 ~ +3C	MPU_RGD0~15	0x0000_0000	Area 0~15 Range Description Register	MPUWE
+40 ~ +7C	MPU_RGCR0~15	0x0000_0000	Area 0~15 control registers	MPUWE
+80	MPU_CR	0x0000_0000	MPU control register	MPUWE
+04	MPU_SP	0x0000_0000	MPU Status Register	None

Base address: 0x40054000

Offset Address	Register Name	Initial Value	Name	Write protection
+1C	MPU_IPPR	0x0000_0000	IP Access Protection Register	SYS

13.4.1 Region Range Description Register MPU_RGDr (n=0 to 15)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
MPURGnADDR [31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
MPURGnADDR[15:5]										MPURGnSIZE[4:0]					

position	Marker	Place Name	Function	Reading and writing
b31~b5	MPURGnADDR[31:5]	Regional base address	Set the base address of area n. The number of valid bits depends on the area size, the low (MPURGnSIZE+1) bit is fixed to 0	R/W
b4~b0	MPURGnSIZE[4:0]	Area size	Set the size of area n 00000~00011: Reserved, set forbidden 00100: 32Byte 00101: 64Byte 11110: 2GByte 11111: 4GByte	R/W

13.4.2 Area control register MPU_RGCRn (n=0 to 15)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
S1RGn E	-	-	-	-	-	S1RGn WP	S1RGn RP	S2RGn E	-	-	-	-	-	S2RGn WP	S2RGn RP

position	Marker	Place Name	Function	Reading and writing
b31~b24	reserved	-	Reserved bit, read 0, write 0 when writing	R
b23	FRGnE	FMPU area n	0: Area n of FMPU is invalid enable 1: Area n of FMPU is valid	R/W
b22~b18	reserved	-	Reserved bit, read 0, write 0 when writing	R
b17	FRGnWP	FMPU area n	0: Area n allows USBFS-DMA writes write access 1: Area n disable USBFS-DMA write	R/W
b16	FRGnRP	FMPU area n	0: Area n allows USBFS-DMA reads read access 1: Area n disable USBFS-DMA read	R/W
b15	S1RGnE	SMPU1 area n	0: Area n of SMPU1 is invalid enable 1: Area n of SMPU1 is valid	R/W
b14~b10	reserved	-	Reserved bit, read 0, write 0 when writing	R
b9	S1RGnWP	SMPU1 area n	0: Area n allows system DMA_1 write write access 1: Area n disable system DMA_1 write	R/W
b8	S1RGnRP	SMPU1 area n	0: Area n allows system DMA_1 read read access 1: Area n disable system DMA_1 read	R/W
b7	S2RGnE	SMPU2 area n	0: Area n of SMPU2 is invalid enable 1: Area n of SMPU2 is valid	R/W
b6~b2	reserved	-	Reserved bit, read 0, write 0 when writing	R/W
b1	S2RGnWP	SMPU2 area n	0: Area n allows system DMA_2 writes write access 1: Area n prohibits system DMA_2 writes	R/W
b0	S2RGnRP	SMPU2 area n	0: Area n allows system DMA_2 read read access 1: Area n disable system DMA_2 read	R/W

Caution:

- The control registers b31 to b16 in areas 8 to 15 are reserved bits.

13.4.3 Control register MPU_CR

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	FMPUE	-	-	-	FMPUACT[1:0]	FMPUB	FMPUB	RP
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SMPU 1E	-	-	-	SMPU1ACT[1: 0]	SMPU 1BWP	SMPU 1BRP	SMPU2E	-	-	-	SMPU2ACT[1: 0]	SMPU2 BWP	SMPU2 BRP		

position	Marker	Place Name	Function	Reading and writing
b31~b24	reserved	-	Reserved bit, read 0, write 0 when writing	R
b23	FMPUE	FMPU Enable	0: FMPU invalid 1: FMPU is valid	R/W
b22~b20	reserved	-	Reserved bit, read 0, write 0 when writing	R
b19~b18	FMPUACT[1:0]	FMPU action selection	Set the action when a prohibited access occurs for USBFS-DMA 00: Ignore (read access read to 0, write access ignored) 01: Ignore + bus error 10: ignore + unblockable interruptions 11: Reset	R/W
b17	FMPUBWP	FMPU background write permission settings	0: FMPU background space allows USBFS-DMA writes 1: FMPU background space disable USBFS-DMA write	R/W
b16	FMPUBRP	FMPU background read permission setting	0: FMPU background space allows USBFS-DMA reads 1: FMPU background space disable USBFS-DMA read	R/W
b15	SMPU1E	SMPU1 enable	0: SMPU1 is invalid 1: SMPU1 is valid	R/W
b14~b12	reserved	-	Reserved bit, read 0, write 0 when writing	R
b11~b10	SMPU1ACT[1:0]	SMPU1 action selection	Set the action of the system DMAC_1 when a prohibited access occurs 00: Ignore (read access read to 0, write access ignored) 01: Ignore + bus error 10: ignore + unblockable interruptions 11: Reset	R/W
b9	SMPU1BWP	SMPU1 background write permission setting	0: SMPU1 background space allows system DMAC_1 writes 1: SMPU1 background space prohibit system DMAC_1 write	R/W
b8	SMPU1BRP	SMPU1 background read permission setting	0: SMPU1 background space allows system DMAC_1 read 1: SMPU1 background space prohibit system DMAC_1 read	R/W

b7	SMPU2E	SMPU2 enable	0: SMPU2 is invalid 1: SMPU2 is valid	R/W
b6~b4	reserved	-	Reserved bit, read 0, write 0 when writing	R
b3~b2	SMPUACT[1:0]	SMPU action selection	Set the action when a prohibited access occurs to system DMA_2 00: Ignore (read access read to 0, write access ignored) 01: Ignore + bus error 10: ignore + unblockable interruptions 11: Reset	R/W
b1	SMPU2BWP	SMPU2 background write permission settings	0: SMPU2 background space allows system DMA_2 writes 1: SMPU2 background space prohibits system DMA_2 writes	R/W
b0	SMPU2BRP	SMPU2 background reading rights	0: SMPU2 background space allows system DMA_2 reads	R/W

Limit setting

1: SMPU2 background space disable system DMA_2 read

When multiple zone settings overlap, the priority is: Set Prohibited > Set Allowed.

13.4.4 Status flag register MPU_SR

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	FMPUE AF
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	SMPU1 EAF	-	-	-	-	-	-	-	SMPU2 EAF

position	Marker	Place Name	Function	Reading and writing
b31~b17	reserved	-	Reserved bit, read 0, write 0 when writing	R
b16	FMPUEAF	FMPU error flag	0: USBFS-DMA did not have an error access 1: USBFS-DMA has an error access	R
b15~b9	reserved	-	Reserved bit, read 0, write 0 when writing	R
b8	SMPU1EAF	SMPU1 error flag	0: System DMA_1 is not incorrectly accessed 1: System DMA_1 error access has occurred	R
b7~b1	reserved	-	Reserved bit, read 0, write 0 when writing	R
b0	SMPU2EAF	SMPU2 error flag	0: System DMA_2 is not incorrectly accessed 1: System DMA_2 error access has occurred	R

Write operations to this register are ignored, and to clear the error flag use MPUECLR.

13.4.5 Flag Clear Register MPU_ECLR

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	FMPUE CLR
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	SMPU1 ECLR	-	-	-	-	-	-	-	SMPU2 ECLR

position	Marker	Place Name	Function	Reading and writing
b31~b17	reserved	-	Reserved bit, read 0, write 0 when writing	R
b16	MPUECLR	FMPU error flag	Write 1 to clear FMPUEAF to 0	W
b15~b9	reserved	-	Reserved bit, read 0, write 0 when writing	R
b8	SMPU1ECLR	SMPU1 error flag	Write 1 to clear SMPU1EAF to 0	W
b7~b1	reserved	-	Reserved bit, read 0, write 0 when writing	R
b0	SMPU2ECLR	SMPU2 error flag	Write 1 to clear SMPU2EAF to 0	W

The value of this register readout is fixed to 0x0000_0000.

13.4.6 Write protect register MPU_WP

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
WKEY[15:1]															MPUWE

position	Marker	Place Name	Function	Reading and writing
b31~b16	reserved	-	Reserved bit, read 0, write 0 when writing	R
b15~b1	WKEY[15:1]	Write Code	When writing to MPUWE, you must also write b1001_0110_1010_010 to WKEY, read out as 0	W
b0	MPUWE	MPU register write permission	0: MPU address register/control register is not allowed to be written 1: MPU address registers/control registers are allowed to be written	RW

Writing 0x96A5 to this register will set MPUWE to 1. Writing 0x96A4 will clear MPUWE to 0. Writing other values will not change MPUWE.

13.4.7 IP Access Protection Register MPU_IPPR

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
BUSER RE	-	MSTPW RP	MSTPR DP	SYSCW RP	SYSCR DP	INTCW RP	INTCR DP	SRAMC WRP	SRAMC RDP	DMPUW RP	DMPUR DP	RTCWR P	RTCRD P	BKSRA MWRP	BKSRA MRDP
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SWDTW RP	SWDTR DP	WDTWR P	WDTRD P	-	-	EFMCW RP	EFMCR DP	CRCWR P	CRCRD P	TRNGW RP	TRNGR DP	HASHW RP	HASHR DP	AESWR P	AESRD P

position	Marker	Place Name	Function	Reading and writing
b31	BUSERRE	Bus error allowed	0: Ignore access to protected IP 1: Return bus error when access to the protection IP occurs	RW
b30	-	-	Reserved bit, read 0, write 0 when writing	R
b29	MSTPWRP	MSTP write protection	0: Allow write operations to registers PWC_FCG0/1/2/3, PWC_FCG0PC 1: Disable write operations to registers PWC_FCG0/1/2/3, PWC_FCG0PC	RW
b28	MSTPRDP	MSTP read protection	0: Allow read operation of registers PWC_FCG0/1/2/3, PWC_FCG0PC 1: Disable the read operation of registers PWC_FCG0/1/2/3, PWC_FCG0PC	RW
b27	SYSCWRP	SYSC write protection	0: Allow write operations to RMU/CMU/PWC 1: Disable write operations to RMU/CMU/PWC	RW
b26	SYSCRDP	SYSC read protection	0: Allow read operations to RMU/CMU/PWC 1: Disable the read operation of RMU/CMU/PWC	RW
b25	INTCWRP	INTC write protection	0: Allow write operations to INTC 1: Disable write operations to INTC	RW
b24	INTCRDP	INTC read protection	0: Read operation to INTC is allowed 1: Disable read operations to INTC	RW
b23	SRAMCWRP	SRAMC write protection	0: Allow register write operation to [10. built-in SRAM] 1: Prohibit the register write operation of [10. built-in SRAM]	RW
b22	SRAMCRDP	SRAMC read protection	0: Allow register read operation of [10. built-in SRAM] 1: Disable the register read operation of [10. built-in SRAM]	RW
b21	DMPUWRP	DMPU write protection	0: Allow write operations to SMPU1/SMPU2/FMPU/IPMPU 1: Disable write operations to SMPU1/SMPU2/FMPU/IPMPU	RW
b20	DMPURDP	DMPU read protection	0: Allow read operations on SMPU1/SMPU2/FMPU/IPMPU 1: Disable read operations on SMPU1/SMPU2/FMPU/IPMPU	RW
b19	RTCWRP	RTC write protection	0: Allow write operations to the RTC 1: Disable write operations to the RTC	RW
b18	RTCRDP	RTC read protection	0: Read operation to the RTC is allowed 1: Disable read operation to RTC	RW
b17	BKSRAMWRP	BKSRAM write protection	0: Allow write operation to Ret-SRAM 1: Disable write operations to Ret-SRAM	RW
b16	BKSRAMRDP	BKSRAM read protection	0: Read operation to Ret-SRAM is allowed 1: Disable read operations to Ret-SRAM	RW
b15	SWDTWRP	SWDT write protection	0: Allow write operation to SWDT 1: Disable write operations to SWDT	RW

b14	SWDTRDP	SWDT read protection	0: Read operation to SWDT is allowed 1: Disable the read operation of SWDT	RW
b13	WDTWRP	WDT write protection	0: Allow write operation to WDT	RW

			1: Disable write operations to WDT	
b12	WDTRDP	WDT read protection	0: Read operation to WDT is allowed 1: Disable the read operation of WDT	RW
b11~b10	-	-	Reserved bit, read 0, write 0 when writing	R
b9	EFMCWRP	EFM write protection	0: Allow register write operation to [9 Embedded Flash (EFM) 1: Prohibit register write operations to [9 Embedded Flash (EFM)	RW
b8	EFMCRDP	EFM read protection	0: Allow register read operation for [9 Embedded Flash (EFM) 1: Disable the register read operation of [9 Embedded Flash (EFM)]	RW
b7	CRCWRP	CRC write protection	0: Allow write operations to CRC 1: Disable write operations to CRC	RW
b6	CRCRDP	CRC read protection	0: Read operation to CRC is allowed 1: Disable read operations on CRC	RW
b5	TRNGWRP	TRNG write protection	0: Allow write operations to TRNG 1: Disable write operations to TRNG	RW
b4	TRNGRDP	TRNG read protection	0: Read operation to TRNG is allowed 1: Disable read operations on TRNG	RW
b3	HASHWRP	HASH Write Protection	0: Allow write operations to HASH 1: Prohibit write operations to HASH	RW
b2	HASHRDP	HASH Read Protection	0: Read operation to HASH is allowed 1: Disable read operations on HASH	RW
b1	AESWRP	AES write protection	0: Allow write operations to AES 1: Disable write operations to AES	RW
b0	AESRDP	AES read protection	0: Allow read operations on AES 1: Disable read operations on AES	RW

Privileged mode allows read and write access to the object IP regardless of this register.

14DMA Controller (DMA)

14.1 Introduction

DMA is used to transfer data between memory and peripheral function modules, enabling data exchange between memory, between memory and peripheral function modules, and between peripheral function modules without CPU involvement.

- The DMA bus is independent of the CPU bus and is transmitted according to the AMBA AHB-Lite bus protocol
- 2 DMA control units with 8 independent channels for independent operation of different DMA transfer functions
- The start source for each channel is configured via a separate trigger source selection register
- One block of data is transferred per request
- Data blocks can be as small as 1 data, up to 1024 data
- The width of each data can be configured as **8bit, 16bit or 32bit**
- Up to 65535 transmissions can be configured
- Source and destination addresses can be independently configured as fixed, incremental, decremental, cyclic or jump with specified offsets
- Three types of interrupts can be generated: block transfer completion interrupt, transfer completion interrupt, and transfer error interrupt. Each of these interrupts can be configured to be masked or not. The block transfer completion and transfer completion can be output as events and can be used as trigger sources for other peripheral modules.
- Support chain transfer function, which can transfer multiple data blocks in one request
- Support external events to trigger channel reset
- Can be set to enter module stop state when not in use to reduce power consumption

14.2 Module schematic

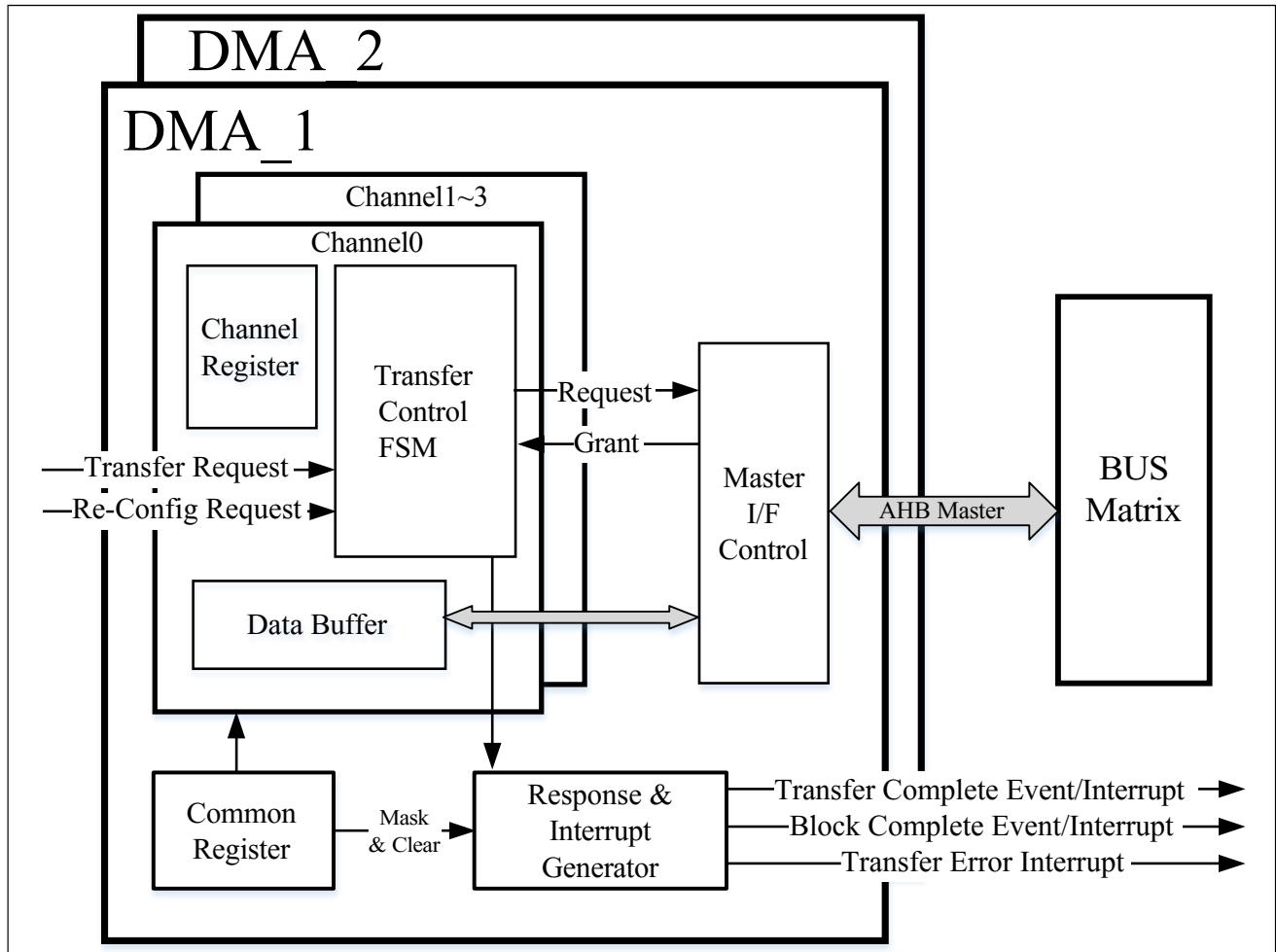


Figure 14-1 DMA structure

diagram

14.3 Function Description

14.3.1 Enabling the DMA controller

When using DMA, you need to write the register to enable the DMA controller first, by writing the DMA enable register DMA_EN.

When DMA is not used, or the chip needs to enter STOP mode, set DMA_EN.EN to 0. Make sure register DMA_CHSTAT.DMAACT is 0 before writing 0 to EN to make sure DMA has completed all transfers.

14.3.2 Channel selection and channel prioritization

Each DMA control unit contains 4 channels, each of which can be configured independently for transfer functions. The order of priority of the 4 channels is: Channel 0 > Channel 1 > Channel 2 > Channel 3.

When a DMA unit has more than one channel requesting a transfer it is executed in priority order. However, channels that are already in transmission are not interrupted, and high-priority channels will not be started until the current channel has completed transmission.

14.3.3 Start DMA

The DMA is initiated by requests generated by the peripheral circuitry. These requests are configured through the trigger source selection register DMA_TRGSEL x ($x=0\sim3$), which allows you to configure the source of the start request for channel x . When a peripheral circuit generates a start request or a software write register generates a start request, and DMA transfer enable is active DMA_EN.EN=1 and the transfer channel is in the licensed state DMA_CHEN.CHEN[x]=1, the channel x transfer is started.

Before use, you need to enable the peripheral circuit trigger function and DMA function in the Function Clock Control 0 register (FCG0).

14.3.4 Data blocks

The amount of data to be transferred per DMA start is represented by a **block**, the size of which is set by the data control register DMA_DTCTL x .BLKSIZE and can be set to a maximum of 1024 data. The data width of each block is determined by DMA_CHxCTL.HIZE.

14.3.5 Transmission address control

The source and destination addresses of the transmission can be set by registers to be fixed, incremental, decremental, reloaded, or discontinuous jumps.

Fixed: The source and destination addresses will be fixed during the transmission.

Incremental and decremental: The source and destination addresses will jump forward or backward according to the value of HSIZE after every 1 data transfer. For example, when HSIZE is **8bit**, the address will be incremented/decremented by 1 each time, **16bit** by 2 each time, and **32bit** by 4 each time.

Reload: After transferring the specified amount of data, the source and destination addresses will be reverted to the initial address setting value. The amount of data to be transferred before address reload, i.e. the size of the repeat area, is set by register DMA_RPT.

Discontinuous address transfer: After transferring the specified amount of data, the source and destination addresses will skip the specified offset. The offset of address hopping and the amount of data to be transferred before hopping, i.e. the size of the discontinuous area, are set by register DMA_SNSEQCTL/DMA_DNSEQCTL. When the conditions of address reload and discontinuous jump are satisfied at the same time, address reload is executed.

14.3.6 Number of transmissions

The total number of data blocks transferred by DMA is set by the CNT bit of the data control register DMA_DTCTLx. The number of transfers can be set up to 65535 times. When the register value is reduced to 0, it means that all data transfer of this channel is completed, and the channel transfer permission bit DMA_CHEN.CHEN[x] is cleared automatically, and the transfer completion interrupt is generated. If the register is set to 0 at the beginning of the transfer, it means infinite transfers, one data block is transferred at each start request, but the channel transfer permit bit is not cleared and the transfer completion interrupt is not generated.

14.3.7 Interrupt and event signal output

The DMA controller can generate the following 3 types of interrupts:

Data block completion interrupt DMA_BTCx: generated after completing a data block transfer.

Transfer completion interrupt DMA_TCx: generated after completing the number of transfers set in register DMA_DTCTLx.CNT.

Transfer error interrupt DMA_ERR: Generates an interrupt when the start request overflows (i.e., the channel triggers another start request while the channel's previous request is still unanswered), or when a bus error occurs during a transfer (e.g., an illegal address or a protected address is accessed), where the bus error immediately terminates the transfer.

The above interrupts can be set as valid or invalid by register DMA_CHxCTL.IE, except for the start request overflow error. In addition, all interrupts are equipped with separate MASK registers to mask the interrupts.

The above DMA_BTCx and DMA_TCx interrupts can also be used as event signal outputs, which can be used as trigger sources for other peripheral circuits.

14.3.8 Chain Transfer

The DMA controller has a chain transfer function. The chain transfer requires the configuration of the following 8 registers with a total of 8 words, called a **descriptor**, which contains the source address, destination address, data control information, address control information, chain pointer, and transmission control information for the chain transfer.

DMA_SARx
DMA_DARx
DMA_DTCTLx
DMA_RPTx
DMA_SNSEQCTLx
DMA_DNSEQCTLx

DMA_LLPx

DMA_CHxCTL

The LLP is called **the Linked-List Pointer**, where the value represents the first address of the next descriptor in memory. When using a chain transfer, first write LL PEN of the channel control register DMA_CHxCTLx to enable the chain transfer and write the information of the first transferred descriptor to the corresponding register. The descriptors of subsequent transfers are then initialized in memory in order. When you need to end the chain transfer, set the LL PEN of DMA_CHxCTLx in the last descriptor of the channel to invalid, and the DMA controller will end the chain transfer after the transfer is completed.

When the last transmission of a descriptor ends, the

If LL PEN=1, LL PRUN=0, the BTC and TC interrupts are generated according to the configuration of the interrupt license, and the channel license CHEN[x] is not automatically cleared to 0. The next descriptor specified by LLP is loaded from memory into the channel configuration register and waits for the next transfer request input to start the first transfer of the new descriptor.

If LL PEN=1 and LL PRUN=1, no BTC and TC interrupts are generated, and channel license CHEN[x] is not automatically cleared to 0. The first transfer of a new descriptor begins directly after the next descriptor specified by LLP is loaded from memory into the channel configuration register.

If LL PEN=0, the chain transfer ends, BTC and TC interrupts are generated according to the configuration of the interrupt license, and the channel license CHEN[x] is automatically cleared to 0.

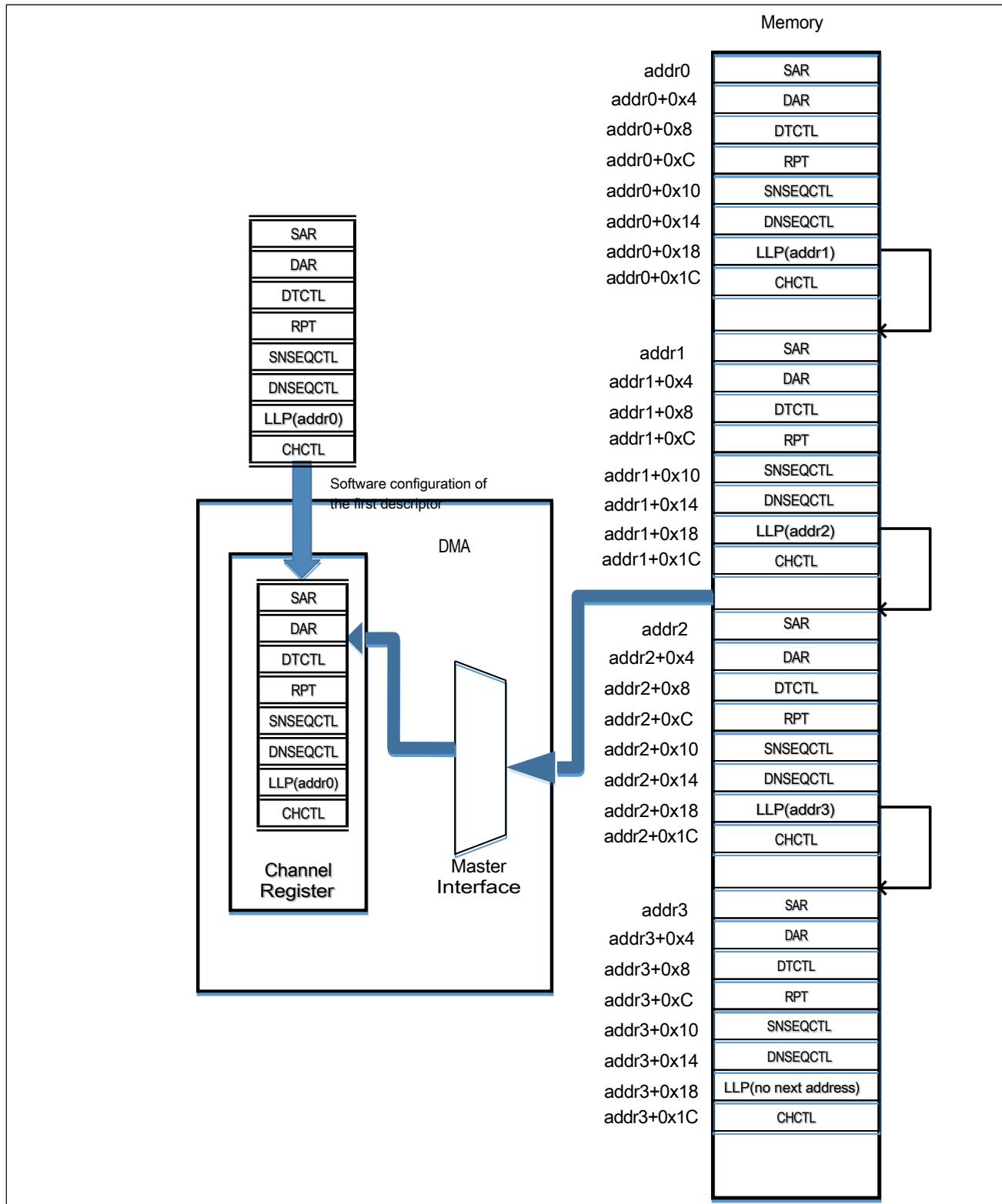


Figure 14-2 Chain transfer diagram

14.3.9 Discontinuous address transmission

The use of discontinuous address transfer enables the source and destination addresses to jump by a certain offset after a certain amount of data has been transferred. The jump direction is forward or backward according to the settings of DMA_CHxCTL_SINC and DMA_CHxCTL_DINC. To use, first set the channel control registers DMA_CHxCTL.SNSEQEN and DMA_CHxCTL.DNSEQEN to 1 as needed to make the discontinuous address transfer valid. Then configure the source and destination discontinuous address transfer ^{Memory} control registers DMA_SNSEQCTLx and DMA_DNSEQCTLx. The transfer process is performed as shown below.

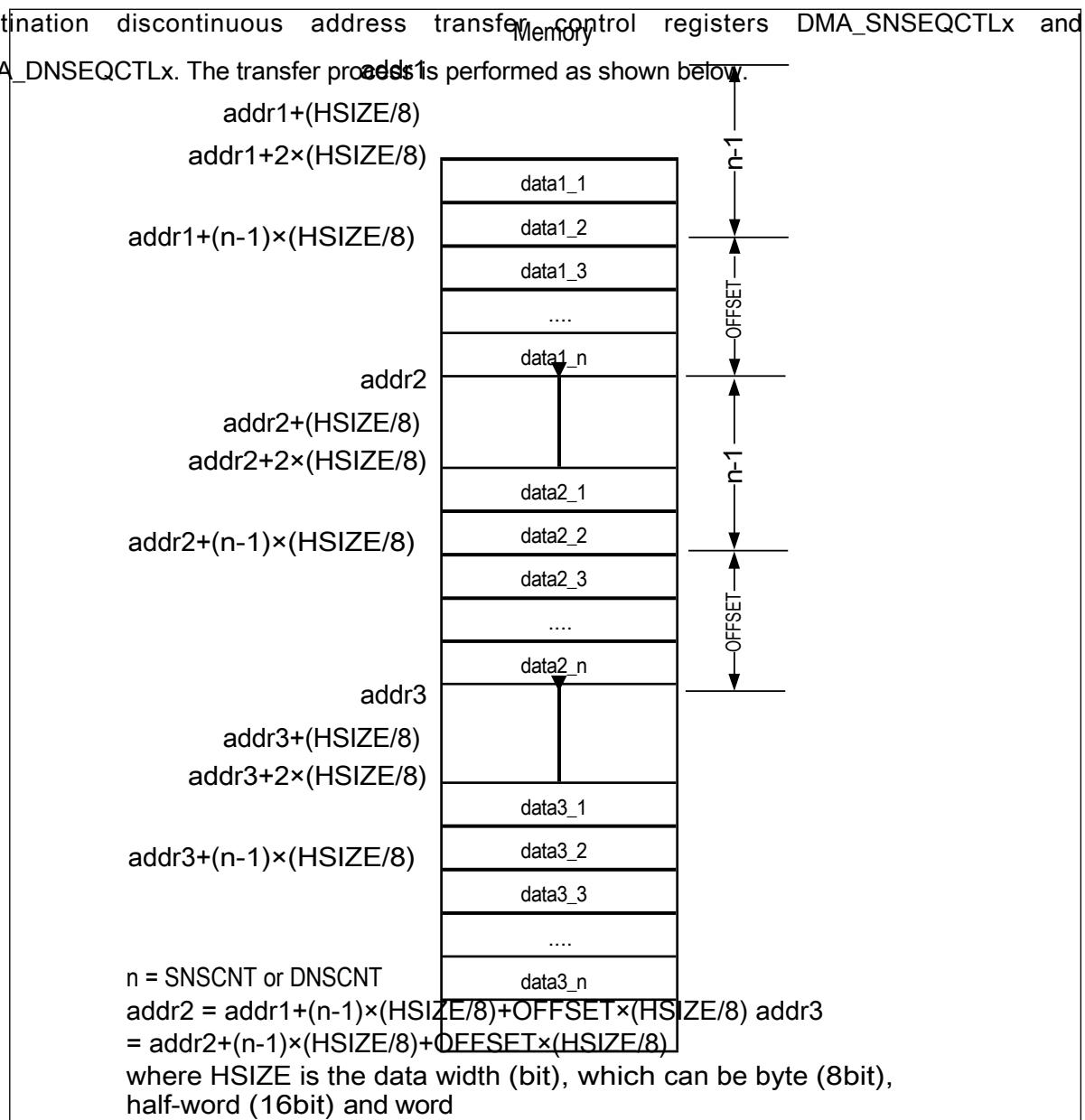


Figure 14-3 Discontinuous Address Transfer Diagram

14.3.10 Channel Reset

The channel reset function refers to the event request from the peripheral circuit to modify the channel internal status register and reconfigure how data is transferred next time. Set register DMA_RCFGCTL_RCFGEN to 1 to allow channel reset. The reset request source is selected via the trigger source selection register DMA_TRGSELRC. When the selected reset request source is input, the channel selected by register DMA_RCFGCTL_RCFGCHS is updated in the specified way. The reset request only updates the internal state and does not initiate the actual data transfer.

The channels can be reset in the following three ways: chain pointer type, discontinuous type, and repetitive type.

When a chain pointer type reset is selected, the descriptor and internal state of the channel are all updated to the new descriptor pointed to by the chain pointer LLP. Subsequent transmission requests are transmitted by the new descriptor.

When discontinuous, heavy duty reset is selected, the internal status of the channel is updated as described in the table below.

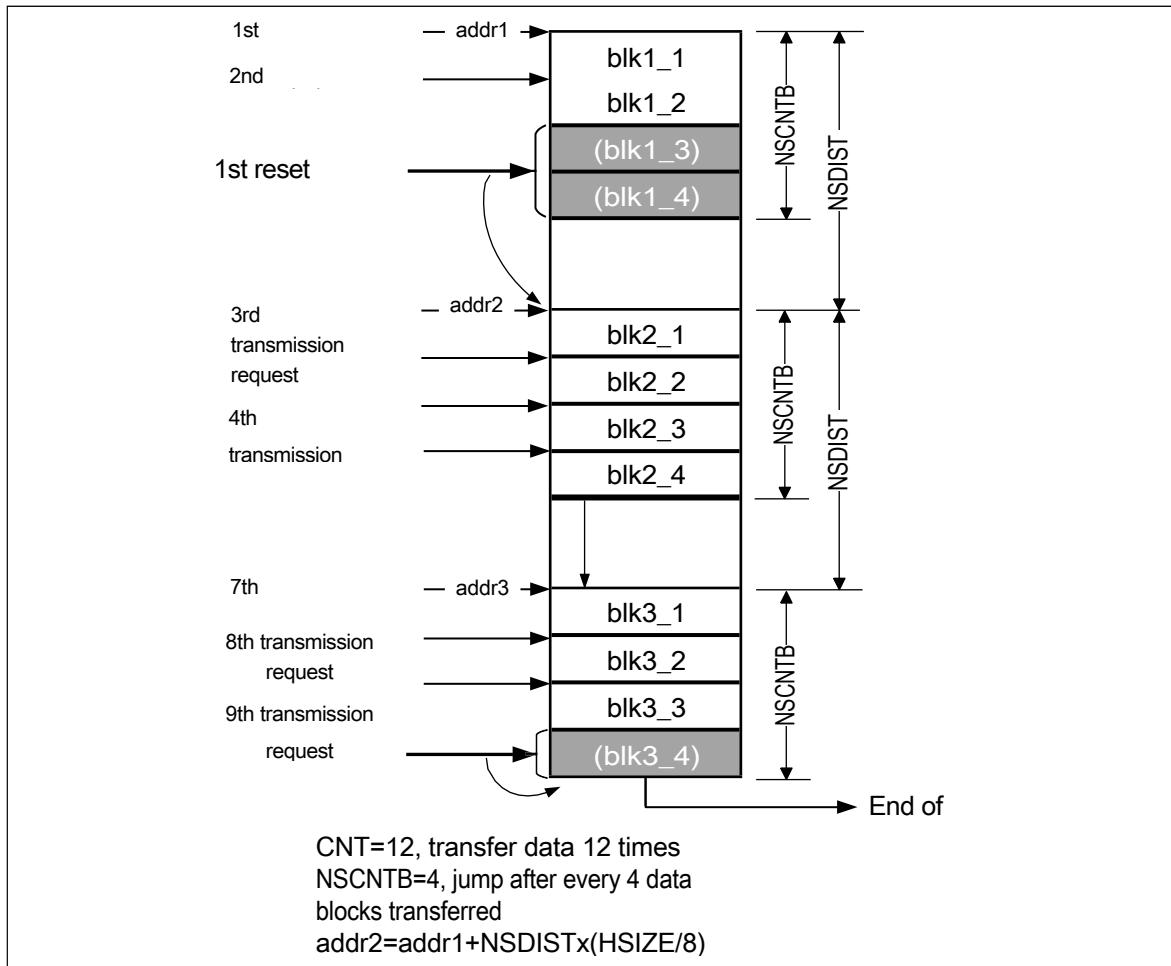
Table 14-1 Channel Reset

Description

Channel internal status	Reset method	
	Discontinuous type	Heavy Duty Type
Remaining transmission counters	Update to the normal state, after the next address discontinuity jump occurs	Update to the normal state, after the next reload occurs
Source/Destination address for next transmission	Update to the first address of the next discontinuous transmission area	Update to the initial setting of register DMA_SARx/DARx

Caution:

- When the reset function is active, the channel uses registers DMA_RPTBx and DMA_SNSEQCTLBx, DMA_DNSEQCTLBx to control the reload and discontinuous hopping of the transfer address. Registers DMA_RPTx and DMA_SNSEQCTLx, DMA_DNSEQCTLx are not valid.



**Figure 14-4 Discontinuous
reset diagram**

In the DMA action shown in Figure 14-4, each transfer request initiates the transfer of one data block. After the 1st reset request occurs, the controller skips the data blocks blk1_3,blk1_4 and the transfer address is updated to the first address of the next discontinuous region, i.e. addr2. After the 2nd reset request occurs, the remaining transfers are updated to 0, i.e. all data transfers are completed and the channel permit bit is automatically cleared to 0, generating a transfer completion interrupt and event.



14.3.11 Transmission early termination

CHENx is valid during the transfer, and the number of transfers set in the data control register DMA_DCTRLx is automatically set to invalid when the non-chain transfer is completed, and the number of transfers set in the last chain transfer is automatically set to invalid when the chain transfer is completed. If the software write DMA_CHEN_.CHENx is 0 during the transfer, the DMA will terminate the transfer after completing the current data read/write.

Caution:

- When the software writes 0 to the CHENx bit to terminate the transfer early, the DMA does not internally save the status of the transfer at the time it was terminated. Without resetting the channel configuration register (descriptor) state, writing CHENx to 1 allows this channel again, and after the transfer request is entered, the DMA will retransmit the terminated data block instead of breaking the transfer.

14.4 Application examples

14.4.1 Memory-to-memory transfer

Target: Transfer 22 data from RAM address 0x2000_0000 to 0x2000_1000 with **32bit** data width.

1. Register Setting

- DMA_EN.EN Write 1 to enable the DMA controller
- Select a channel, e.g. channel 0, and configure the channel registers to achieve:
 - Write DMA_SAR0 Configure source address to SRAM area 0x2000_0000
 - Write DMA_DAR0 Configure source address to SRAM area 0x2000_1000
 - Write DMA_DTCTL0 to configure the data block size to be 4, the number of transfers to be 3, and generate a block transfer completion interrupt after each transfer of 1 data block, and a transfer completion interrupt after 3 transfers are completed
 - Write the DMA_RPT register to configure the source address repeat area size to 6, i.e., reload the initial source address after 6 addresses have been transferred
 - Configure the channel control register DMA_CH0CTL to achieve:
 - * The source and destination address chain transfer is invalid
 - * Source address overloading is valid, and the target address update method is self-incrementing
 - * Data width is word(32bit)
 - * Interrupt enable active
 - Channel enable bit DMA_CHEN.CHEN0 Write 1 to enable channel 0
 - Configure the trigger source controller DMA_TRGSEL0 to select software triggering as the start request for DMA channel 0
- Write peripheral event software trigger register INTSFTTRG_STRG to 1, send the first software start request, DMA start data transfer

2. Transmission process

As the size of the transferred data block is 4, when the software writes INTSFTTRG_STRG to 1 to start the first transfer, when a data block transfer is completed, the transfer count DMA_DTCTL0.CNT is subtracted by 1 and a block transfer completion interrupt is generated, the software can continue to write INTSFTTRG_STRG in the interrupt subroutine to start the second transfer. In the second transfer, since the source address repeat area size is set to 6, the source address will be reloaded to the initial address 0x20000000 after 2 addresses are transferred and continue to transfer the remaining 2 addresses. After the second transfer is completed, the transfer count DMA_DTCTL0.CNT is

subtracted by 1 and a block transfer completion interrupt is generated, and the software can continue to write INTSFTTRG.STRG in the interrupt subroutine to start the third transfer. After the third transfer is completed, the transfer count DMA_DTCTL0.CNT decreases to 0, i.e. the transfer is fully completed, the DMA generates a block transfer completion interrupt and a transfer completion interrupt, and the channel enable bit DMA_CHEN.CHEN0 is automatically cleared.

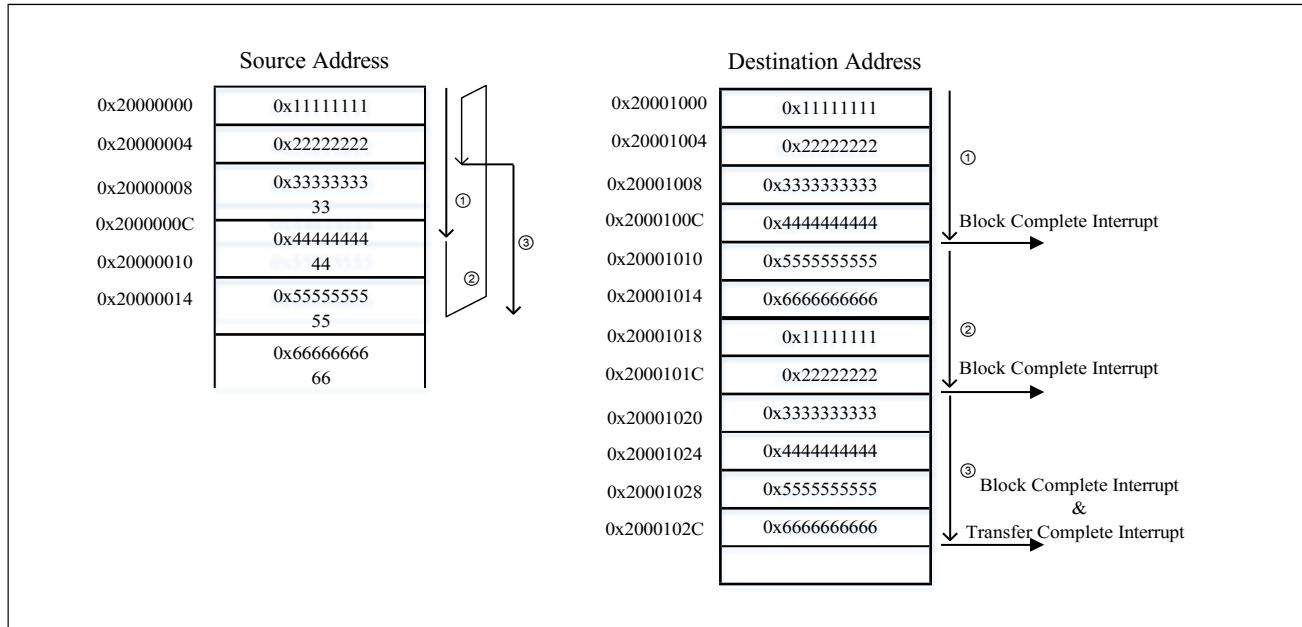


Figure 14-5 Application Example 1: Memory-to-Memory Transfer

14.4.2 Memory to peripheral circuit transfer

Goal: Transfer 10 half-word data from the RAM address 0x2000_0000 to the transmit buffer register of the communication module, which generates a transfer request for each data sent. When the last data is sent, the DMA generates a transfer completion interrupt.

1. Register Setting

- DMA_EN.EN Write 1 to enable the DMA controller
- Configure the DMA_INTMSK register to mask the block transfer completion interrupt and enable the transfer completion interrupt
- Select a channel and configure the channel register, e.g. select channel 0
 - Write DMA_SAR0 Configure source address to SRAM area 0x2000_0000
 - Write DMA_DAR0 Configure the source address as the register address of the peripheral circuit 0x4000_0000
 - Write DMA_DTCTL0 Configure the data block size to 1 and the number of transfers to 10, one transfer per transfer request, 1 data per transfer.
 - Configure the channel control register DMA_CH0CTL to achieve:
 - * The source and destination address chain transfer is invalid
 - * The source address update method is self-incrementing, and the destination address is fixed
 - * Half-word (16bit) data width for source and destination addresses
 - * Interrupt enable active
 - Configure the trigger source controller DMA_TRGSEL0 to select the communication module's transmit register empty event as the start request for DMA channel 0

-
- Channel enable bit DMA_CHEN_CHEN0 Write 1 to enable channel 0

2. Transmission process

After the channel is enabled, the DMA waits for a transfer request from the communication module. When the transfer request is generated, the DMA transfers the data in RAM to the transmit buffer register of the communication module and waits for the second transfer request from the communication module. When all 10 data transfers are completed, the DMA generates

a transfer completion interrupt and the channel enable bit DMA_CHEN.CHEN0 will be automatically cleared to zero. DMA

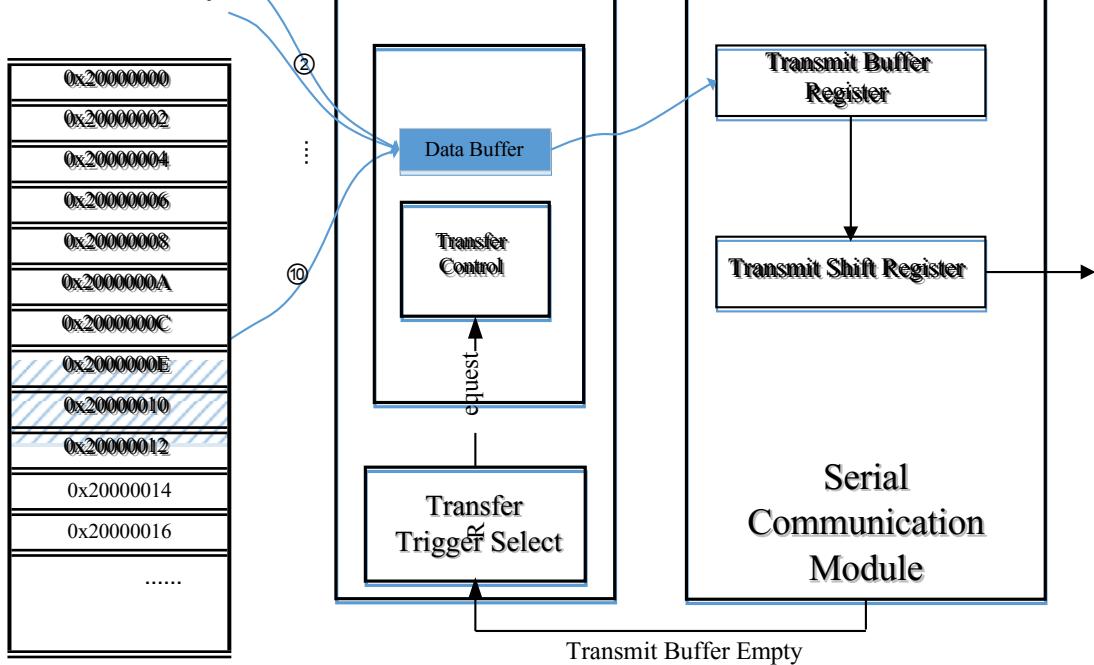


Figure 14-6 Application Example 2: Memory to Peripheral Circuit Transfer

14.4.3 Memory-to-memory chain transfer

1. Register Setting

- DMA_EN.EN Write 1 to enable the DMA controller
- Select a channel and configure the channel register, e.g. select channel 0 & configure the descriptor for the first transmission (**descriptor0**)
 - Write DMA_SAR0 Configure source address to SRAM area 0x2000_0000
 - Write DMA_DAR0 Configure source address to SRAM area 0x2000_1000
 - Write DMA_DTCTL0 Configure the data block size to 10
 - Write the address of the second descriptor (**descriptor1**) in the chain pointer register DMA_LLPO 0x20002000
 - Configure the channel control register DMA_CH0CTL to configure the transfer parameters of the first data block to achieve:
 - * Chain transfer is effective
 - * Chain transfer mode is to start the next transfer directly
 - * The source and destination addresses are updated in a self-incrementing manner
 - * Data width in words (**32bit**)
 - * Interrupt enable is invalid
- Configure the descriptor for the second transfer (**descriptor1**) in the 0x2000_2000 address of the RAM space, including
 - 0x2000_2000 writes 32-bit data 0x2000_0100, which is the source address of the second transfer
 - 0x2000_2004 writes 32-bit data 0x2000_1100, which is the destination address for the second transfer
 - The size of the configured data block in 0x2000_2008 is 20
 - 0x2000_2018 writes the 32-bit data 0x2000_2100, which is the address of **the descriptor2 of the third transfer**
 - The control data for the second transmission is written in 0x2000_021C, which implements:
 - * Chain transfer is effective
 - * Chain transfer mode is to start the next transfer directly
 - * The source and destination addresses are updated in a self-incrementing manner
 - * Data width is half word (**16bit**)
 - * Interrupt enable is invalid
- Configure the descriptor for the third transfer (**descriptor2**) in the 0x2000_2100 address of the RAM space, including
 - 0x2000_2100 writes 32-bit data 0x2000_0200, which is the source address of the third transmission

- Write 32-bit data 0x2000_1200 in 0x2000_2104, which is the destination address for the third transfer
- 0x2000_2108 Configure the size of the data block to 40
- 0x2000_2118 writes the 32-bit data 0x0, which means that this transmission is the last transmission of the chain
- Control data implementation for writing the third transmission in 0x2000_211C:
 - * Chain transfer invalid

- * The source and destination addresses are updated in a self-incrementing manner
- * Data width in bytes (**8bit**)
- * Interrupt enable active
- Channel enable bit DMA_CHEN.CHENO Write 1 to enable channel 0
- Configure the transfer start trigger source selection register DMA_TRGSEL0 to select software trigger as the start request for DMA channel 0
- Write software trigger register INTSFTTRG STRG to 1 to send a start request and the DMA starts transferring data

2. Transmission process

The software starts the DMA to start the transfer. After the first transfer is completed, the DMA reads the **descriptor** for the second transfer (**descriptor1**) **into** the channel register since the chain transfer mode is set to start the next transfer directly and the interrupt is disabled. The second transfer is started directly according to the parameters configured in the descriptor. After the second transfer is completed, the descriptor for the third transfer (**descriptor2**) **is read into the channel** register. The third transmission is started according to the parameters configured in the descriptor. After the third transfer is completed, the DMA generates a transfer completion interrupt and clears the channel enable bit DMA_CHEN.CHENO since the interrupt enable is active, according to the configuration information that this is the last in the chain of transfers.

14.5 Register Description

DMA_1 BASE_ADDR:0x4005_3000

DMA_2 BASE_ADDR:0x4005_3400

Register Name	Sym bols	Offset Address	Bit width	Reset value
DMA enable register	DMA_EN	0x00	32	0x0000_0000
Interrupt status register 0	DMA_INTSTAT0	0x04	32	0x0000_0000
Interrupt status register 1	DMA_INTSTAT1	0x08	32	0x0000_0000
Interrupt Mask Register 0	DMA_INTMASK0	0x0C	32	0x0000_0000
Interrupt Mask Register 1	DMA_INTMASK1	0x10	32	0x0000_0000
Interrupt reset register 0	DMA_INTCLR0	0x14	32	0x0000_0000
Interrupt reset register 1	DMA_INTCLR1	0x18	32	0x0000_0000
Channel enable register	DMA_CHEN	0x1C	32	0x0000_0000
Transmission request status register	DMA_REQSTAT	0x20	32	0x0000_0000
Channel monitoring register in transmission	DMA_CHSTAT	0x24	32	0x0000_0000
Channel Reset Control Register	DMA_RCFGCTL	0x2c	32	0x0000_0000
Transfer source address register	DMA_SARx *1	0x40+0x40*x	32	0x0000_0000
Transfer destination address register	DMA_DARx	0x44+0x40*x	32	0x0000_0000
Data Control Register	DMA_DTCTLx	0x48+0x40*x	32	0x0000_0001
Repeat Area Size Register	DMA_RPTx	0x4C+0x40*x	32	0x0000_0000
Repeat Area Size Register B	DMA_RPTBx			
Source device discontinuous address transfer control register	DMA_SNSEQCTRLx	0x50+0x40*x	32	0x0000_0000
Source device discontinuous address transfer control register B	DMA_SNSEQCTRLBx			
Destination device discontinuous address transfer control register	DMA_DNSEQCTRLx	0x54+0x40*x	32	0x0000_0000
Destination device discontinuous address transfer control register B	DMA_DNSEQCTBx			
Chain Pointer Register	DMA_LLPx	0x58+0x40*x	32	0x0000_0000
Channel Control Register	DMA_CHxCTL	0x5C+0x40*x	32	0x0000_1000
Transfer source address monitoring register	DMA_MONSARx	0x60+0x40*x	32	0x0000_0000

14.5.1 DMA enable register (DMA_EN)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EN

position	Marker	Place Name	Function	Reading and writing
b31-b1	Reserved	-	Read "0", write "0" when writing	R/W
b0	EN	DMA enable bit	0: DMA invalid 1: DMA enable	R/W

14.5.2 Interrupt status register 0(DMA_INTSTAT0)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	REQERR[3:0]

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	TRNERR[3:0]

position	Marker	Place Name	Function	Reading and writing
b31-b20	Reserved	-	Read "0", write "0" when writing	R/W
b19-b16	REQERR[3:0]	Transmit request overflow error	0: No transmission request overflow error occurred on this channel 1: The channel has a transmission request overflow error, that is, the last request is still in the waiting state when the transmission request comes again	R
b15-b4	Reserved	-	Read "0", write "0" when writing	R/W
b3-b0	TRNERR[3:0]	Transmission error interrupt bit	0: No transmission error occurred on this channel 1: A transmission error has occurred on this channel	R

14.5.3 Interrupt status register 0(DMA_INTSTAT1)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	BTC[3:0]	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	TC[3:0]	
position	Marker	Place Name	Function	Reading and writing											
b31-b20	Reserved	-	Read "0", write "0" when writing	R/W											
b19-b16	BTC[3:0]	Block transfer completion interrupt bit	This interrupt occurs after the completion of a block of data transfer 0: No block transfer interrupt occurs for this channel 1: Block transmission interruption occurred on this channel	R											
b15-b4	Reserved	-	Read "0", write "0" when writing	R/W											
b3-b0	TC[3:0]	Transfer completion interrupt bit	This interrupt occurs after completing the number of transfers set in the transfer count register DMA_CNTx 0: No transfer completion interrupt occurs for this channel 1: Transmission completion interruption occurred on this channel	R											

14.5.4 Interrupt Mask Register (DMA_INTMASK0)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	MSKREQERR[3:0]	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	MSKTRNERR[3:0]	
position	Marker	Place Name	Function	Reading and writing											
b31-b20	Reserved	-	Read "0", write "0" when writing	R/W											
b19-b16	MSKREQERR[3:0]	Transfer request overflow interrupt mask	0: Do not mask transmission request overflow 1: Masking transmission request overflow interrupts	R/W											
b15-b4	Reserved	-	Read "0", write "0" when writing	R/W											
b3-b0	MSKTRNERR[3:0]	Transmission error interrupt mask	0: Do not mask transmission error interrupts 1: Masking transmission error interrupt	R/W											

14.5.5 Interrupt Mask Register (DMA_INTMASK1)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MSKBTC[3:0]
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MSKTC[3:0]
<hr/>															
position	Marker	Place Name	Function	Reading and writing											
b31-b20	Reserved	-	Read "0", write "0" when writing	R/W											
b19-b16	MSKBTC[3:0]	Block transfer completion interrupt mask	0: Do not mask block transfer completion interrupt 1: Mask block transfer completion interrupt	R/W											
b15-b4	Reserved	-	Read "0", write "0" when writing	R/W											
b3-b0	MSKTC[3:0]	Transmission completion interrupt mask	0: Do not mask transmission completion interrupt 1: Shield transmission completion interrupt	R/W											

14.5.6 Interrupt reset register (DMA_INTCLR0)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CLRREQERR[3:0]
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CLRTRNERR[3:0]
<hr/>															
position	Marker	Place Name	Function	Reading and writing											
b31-b20	Reserved	-	Read "0", write "0" when writing	R/W											
b19-b16	CLRREQERR[3:0]	Transmission request overflow error interrupt reset	Write 0 has no effect, write 1 resets the transmission request overflow error interrupt status bit read out always to 0	W											
b15-b4	Reserved	-	Read "0", write "0" when writing	R/W											
b3-b0	CLRTRNERR[3:0]	Transmission error interrupt reset	Write 0 has no effect, write 1 resets the transmission error interrupt status bit read out to 0 forever	W											

14.5.7 Interrupt reset register (DMA_INTCLR1)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CLRBTC[3:0]

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CLRTC[3:0]

position	Marker	Place Name	Function	Reading and writing
b31-b20	Reserved	-	Read "0", write "0" when writing	R/W
b19-b16	CLRBTC[3:0]	Block transfer completion interrupt reset	Write 0 has no effect, write 1 reset block transfer complete interrupt status bit read out always 0	W
b15-b4	Reserved	-	Read "0", write "0" when writing	R/W
b3-b0	CLRTC[3:0]	Transmission completion interrupt reset	Write 0 has no effect, write 1 reset transmission completion interrupt status bit read out is always 0	W

14.5.8 Channel enable register (DMA_CHEN)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CHEN[3:0]

position	Marker	Place Name	Function	Reading and writing
b31-b4	Reserved	-	Read "0", write "0" when writing	R/W
b3-b0	CHEN[3:0]	Channel enable bit	Each bit corresponds to one channel. The enable bit is held at 1 during transmission and will be cleared automatically when the number of transfers set in the Transfer Count Register DMA_DTCTLx.CNT is completed. If DMA_DTCTLx.CNT is set to 0, it will not be cleared automatically after the transfer is completed, i.e., there will be an unlimited number of transfers. 0: The channel is invalid 1: The channel is valid	R/W

14.5.9 Channel reset control register (DMA_RCFGCTL)

Reset value: 0x0000_0000

b3 1	b3 0	b2 9	b2 8	b27	b26	b25	b24	b23	b22	b2 1	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	CNTMD[1:0] 1	DARMD[1:0]	SARMD[1:0]			

b1 5	b1 4	b1 3	b1 2	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	RCFGCHS[3:0]				-	-	-	-	-	-	RCFG LLP	RCFG EN

position	Marker	Place Name	Function	Reading and writing
b31-b22	Reserved	-	Read '0', write '0' when writing	R/W
b21-b20	CNTMD[1:0]	Reset method of remaining transmission counters	00: Stay the same 01: By source address method 10, 11: By destination address method When the source address method is selected and the source address is selected as discontinuous reset, the remaining transmission count counter is updated to the state after the number of transmissions specified by DMA_SNSEQCTLBx.SNSCNTB; when the source address is selected as repetitive, the remaining transmission count counter is updated to the state after the number of transmissions specified by DMA_RPTBx.SRPTB. When Hold is selected for the source address, the remaining transmission count counter also remains unchanged. When the target address method is selected, it is similar to the source address method.	R/W
b19-b18	DARMD[1:0]	Destination address reset method	00: Stay the same 01: Discontinuous reset The destination address for the next transmission is updated to addr_base + (DNSDIST x HSIZE(bit)/8) where: addr_base indicates the first address of the current discontiguous transmission area 10, 11: Repetitive reset The destination address of the next transmission is updated to the initial setting value of the DMA_DARx register. Note: It is only allowed to set DARMD[1:0] to 01 when the target address of this channel is not continuously transferred valid (DMA_CHxCTL.DNSEQEN=1). Only allowed in the state of this channel target address reload valid (DMA_CHxCTL.DRPTEN=1) Set DARMD[1] to 1.	R/W

b17-b16	SARMD[1:0]	Source address reset method	00: Stay the same 01: Discontinuous reset The source address of the next transmission is updated to addr_base + (SNSDIST x HSIZE(bit)/8) where: addr_base indicates the first address of the current discontinuous transmission area 10, 11: Repetitive reset The source address of the next transmission is updated to the initial setting value of the DMA_SARx register. Note: Only discontinuous transmission of source address in this channel is allowed to be valid (DMA_CHxCTL.SNSEQEN=1) state to set SARMD[1:0] to 01. It is only allowed to set SARMD[1] to 1 in the state of this channel source address overload valid (DMA_CHxCTL.SRPTEN=1).	R/W
b15-b12	Reserved	-	Read '1', write '0' when writing	R/W
b11-b8	RCFGCHS[3:0]	Reset channel selection	0x0: Channel 0	R/W

0x1: Channel 1
 0x2: Channel 2
 0x3: Channel 3
 Other: Reserved, set forbidden

b7-b2	Reserved	-	Read '0', write '0' when writing	R/W
b1	RCFGLLP	Chain pointer type channel reset	0: Chain pointer type reset is invalid 1: Chain pointer type reset is effective Note: When RCFGPLL is set to 1, the channel will reload the new descriptor in memory, so this send All bits 16-bit21 of the memory are invalid.	R/W
b0	RCFGEN	Channel Reset License	0: event-triggered channel configuration register forced update is disabled 1: Allow events to trigger a forced update of the channel configuration register	R/W

Caution:

- Please set this register when DMA_EN.EN is 0. This register must be set before resetting the first transmission of the channel.

14.5.10 Transfer request status register (DMA_REQSTAT)

Reset value: 0x0000_0000

b 31	b 30	b29	b28	b 2	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RCFGREQ
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CHREQ[3:0]

position	Marker	Place Name	Function	Reading and writing
b31-b17	Reserved	-	Read "0", write "0" when writing	R/W
b16	RCFGREQ	Channel reset request flag	When the external reset request input is followed by a 1, the channel reset is initiated, or when channel reset is disabled. 0: No channel reset request 1: There is a channel reset request	R
b15-b4	Reserved	-	Read "0", write "0" when writing	R/W
b3~b0	CHREQ[3:0]	Channel transmission request flag bit	Each bit corresponds to one channel. When the external transmission request is entered after corresponding position 1, the This bit is cleared to 0 when a transmission is initiated for this channel, or when a transmission error occurs, or when the transmission transmission permission bit (DMAEN or CHEN[x]) is written to 0. When this bit is in state 1 and the channel transmission request is entered again, a transmission request overflow error occurs and the second request is ignored. 0: No transmission request for this channel 1: The channel has a transmission request	R

14.5.11 Channel status observation register (DMA_CHSTAT)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-							CHACT[3:0]	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	RCFGACT	DMAACT
<hr/>															
position	Marker	Place Name	Function	Reading and writing											
b31-b20	Reserved	-	Read "0", write "0" when writing	R/W											
b19-b16	CHACT[3:0]	Channel monitoring bit in transmission action	Each bit corresponds to one channel. 0: The channel is idle 1: The channel is in action	R											
b15-b2	Reserved	-	Read "0", write "0" when writing	R/W											
b1	RCFGACT	Monitor bit in DMA channel reset action	0: DMA is not in channel reset action 1: DMA is in channel reset action	R											
b0	DMAACT	Monitoring bit in DMA action	0: DMA is not in transfer action 1: DMA is in transfer action	R											

14.5.12 Transfer source address register (DMA_SARx) (x=0~3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16												
SAR [31:16]																											
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0												
SAR [15:0]																											
<hr/>																											
position	Marker	Place Name	Function	Reading and writing																							
b31-b0	SAR [31:0]	Transmission source address	Set the transmission source address	R/W																							
<hr/>																											
Note:																											
- When the transferred data width is 16bit, i.e. DMA_CHxCTL.HSIZE=01, SAR[0] is invalid. When the transmitted data width is 32bit, i.e. DMA_CHxCTL.HSIZE=1x, SAR[1:0] is not effect.																											

14.5.13 Transfer destination address register (DMA_DARx) (x=0~3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DAR[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DAR[15:0]															
position	Marker	Place Name				Function				Reading and writing					
b31-b0	DAR[31:0]	Transfer destination address				Set the transmission destination				R/W					
						address Note:									
						- When the transfer data width is 16bit, i.e. DMA_CHxCTL.HSIZE=01, DAR[0] is invalid. When the transfer data width is 32bit, i.e.									
						DMA_CHxCTL.HSIZE=1x, DAR[1:0] is not Effect.									

14.5.14 Data control register (DMA_DTCTLx) (x=0~3)

Reset value: 0x0000_0001

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CNT [15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	BLKSIZE[9:0]									
position	Marker	Place Name				Function				Reading and writing					
b31-b16	CNT [15:0]	Number of transmissions				The total number of transfers, each request to initiate the transfer of one block of data, the transfer count counter decreases by 1 when completed, and the transfer completion interrupt occurs when it decreases to 0. If set to 0, it means unlimited transfers, one data block is requested to be transferred each time it is started, and the transfer count counter remains unchanged at completion, and no transfer completion interrupt is generated.				R/W					
b15-b10	Reserved	-				Read "1", write "0" when writing				R/W					
b9-b0	BLKSIZE[9:0]	Size of data blocks				Set the size of the data block, up to 1024 data can be configured. The width of each data is determined by the HSIZE bit of the DMA_CHxCTL register. The register value is set to 1 to transfer 1 data at a time, and set to 0 to transfer 1024 data at a time.				R/W					

14.5.15 Repeat Region Size Register (DMA_RPTx) (x=0~3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	DRPT[9:0]

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SRPT [9:0].

position	Marker	Place Name	Function	Reading and writing
b31-b26	Reserved	-	Read "", write "" when writing	R/W
b25-b16	DRPT[9:0]	Destination address repeat area size	Set the target address repeat area size The target device reloads the target address after every DRPT data transfer to the initial setting of the DMA_DARx register.	R/W
			The register is set to 10 for address reload after every 10 data transfers, and to 0 for address reload after every 1024 data transfers. The width of each data is determined by the HSIZE bit of the DMA_CHxCTL register determines.	
b15-b10	Reserved	-	Read "", write "" when writing	R/W
b9-b0	SRPT [9:0].	Source address repeat area size	Set source address repeat area size The source device reloads the source address after every SRPT data transfer to the initial setting of the DMA_SARx register. A register setting of 10 reloads the address after every 10 data transfers, and a setting of 0 reloads the address after every 1024 data transfers. The width of each data is determined by the HSIZE bit of the DMA_CHxCTL register.	R/W

This register configures the size of the duplicate area for the source and destination addresses. Using duplicate address transfer requires configuring the **SRPTEN/DRPREN** bits of the DMA_CHxCTL register to be valid and configuring the SINC/DINC bits of the DMA_CHxCTL register so that the address update method is self-incrementing or self-decrementing; if it is fixed, the address reload function is disabled.

DMA_RPTx, DMA_RPTBx These two registers share the same address and are both used to define the repeat area size. Which one to use depends on whether or not reset is enabled for that channel. DMA_RPTx is used when reset is not enabled and DMA_PRTBx is used when reset is enabled.

14.5.16 Repeat area size register B(DMA_RPTBx) (x=0~3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	DRPT[9:0]
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SRPT[9:0]
<hr/>															
position	Marker	Place Name	Function	Reading and writing											
b31-b26	Reserved	-	Read "0", write "0" when writing	R/W											
b25-b16	DRPTB[9:0]	Destination address repeat area size	Set the target address repeat area size The target device reloads the target address to the initial set value of the DMA_DARx register after every DRPTB data block transfer. The data block size is determined by the difference between DMA_DTCTLx.BLKSIZE and DMA_CHxCTL.HSIZE is determined.	R/W											
b15-b10	Reserved	-	Read "0", write "0" when writing	R/W											
b9-b0	SRPTB[9:0]	Source address duplication area size	Set source address repeat area size The source device reloads the source address to the initial DMA_SARx register setting after every SRPTB data block transfer. The data block size is determined by DMA_DTCTLx.BLKSIZE and DMA_CHxCTL.HSIZE.	R/W											

This register configures the size of the duplicate area for the source and destination addresses. Using duplicate address transfer requires configuring the **SRPTEN/DRPREN** bits of the DMA_CHxCTL register to be valid and configuring the SINC/DINC bits of the DMA_CHxCTL register so that the address update method is self-incrementing or self-decrementing; if it is fixed, the duplicate address transfer function is disabled.

DMA_RPTx, DMA_RPTBx These two registers share the same address and are both used to define the repeat area size. Which one to use depends on whether or not reset is enabled for that channel. DMA_RPTx is used when reset is not enabled and DMA_PRTBx is used when reset is enabled.

14.5.17 Source device discontinuous address transfer control register (DMA_SNSEQCTLx) (x=0~3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16				
SNSCNT [11:0]												SOFFSET [19:16]							
SOFFSET[15: 0]																			
<hr/>																			
position	Marker	Place Name	Function	Reading and writing															
b31-b20	SNSCNT [11:0]	The amount of data jumped by the source address	Set the size of the amount of data to be transferred before the source address jump. The source device jumps the source address at the offset specified by SOFFSET after every SNSCNT data transfer. Set the register to 10 to jump the address after every 10 data transfers, and to 0 to jump the address after every 4096 data transfers.	R/W															
b19-b0	SOFFSET[19:0]	Address offset of the source address jump	Sets the offset of the source address jump when discontinuous addresses are transmitted. The offset is relative to the current transfer address, which is the last transfer address before the jump. The direction of the jump is jumped forward or backward depending on the value of the channel control register DMA_CHxCTL. Refer to Figure 14-3. When DMA_CHxCTL.SINC is set to Address Fixed, discontinuous address transfers are invalid. The jump address will be calculated based on the number of bits set by the width of the data (DMA_CHxCTL.HSIZE) and the value of SOFFSET. Address offset=SOFFSET×(HSIZE(bit)/8) For example, when SOFFSET is set to 10 and HSIZE is a word (32bit), the address offset is $10 \times 4 = 40$, if HSIZE is a half-word (16bit), the offset is $10 \times 2 = 20$, and if HSIZE is a byte (8bit), the offset is $10 \times 1 = 10$. Source address of next transmission = source address of current transmission ± address offset	R/W															

Using source device discontinuous transfers requires configuring the SNSEQEN bit of the DMA_CHxCTL register to be valid and configuring the SINC bit of the DMA_CHxCTL register so that the address update method is self-increasing or self-decreasing.

DMA_SNSEQCTLx, DMA_SNSEQCTLBx These two registers share the same address and are both used to define discontinuous transfers. Which one to use depends on whether or not reset is enabled for that channel. DMA_SNSEQCTLx is used when reset is not enabled, and HC32F460_F45x_A460 Series Reference

DMA_SNSEQCTLBx is used when reset is enabled.

14.5.18 Source device discontinuous address transfer control register

B(DMA_SNSEQCTLBx) (x=0~3)

Reset value: 0x0000_0000

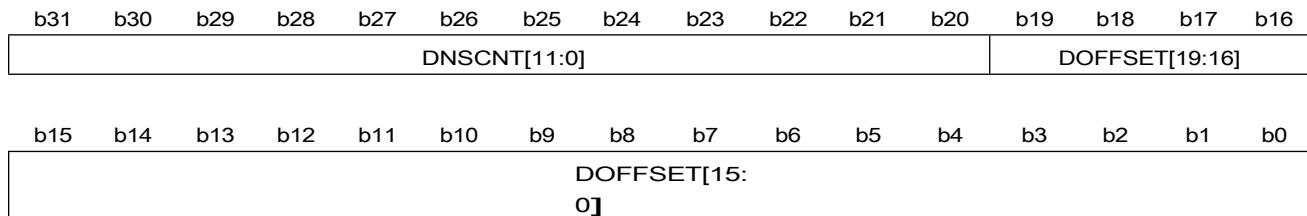
b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
SNSCNTB[11: 0]												SNSDIST [19:16]			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SNSDIST[15: 0]															
position	Marker	Place Name	Function	Reading and writing											
b31-b20	SNSCNTB[11:0]	The amount of data jumped by the source address	Set the size of the amount of data to be transferred before the source address jump. The source device hops the source address at the SNSDIST specified address spacing after each SNSCNTB data block transfer. The block size is determined by DMA_DTCTLx.BLKSIZE and DMA_CHxCTL.HSIZE.	R/W											
b19-b0	SNSDIST [19:0]	Source discontinuity area address spacing	When discontinuous addresses are transmitted, set the spacing between the two discontinuous areas of the source device. The direction of the jump is forward or backward depending on the value of the channel control register DMA_CHxCTL.SINC. Refer to Figure 14-4. When DMA_CHxCTL.SINC is set to address fixed, discontinuous address transfer is invalid. The address spacing will be calculated based on the number of bits set by the width of the data (DMA_CHxCTL.HSIZE) and the value of SNSDIST. Address spacing=SNSDIST×(HSIZE(bit)/8) For example, when SNSDIST is set to 10 and HSIZE is word (32bit), the address spacing is 10×4=40, if HSIZE is half-word (16bit), the spacing is 10×2=20, and if HSIZE is byte (8bit), the spacing is 10×1=10. Source address of next transmission = first address of current source discontinuity area ± address spacing	R/W											

Using source device discontinuous transfers requires configuring the SNSEQEN bit of the DMA_CHxCTL register to be valid and configuring the SINC bit of the DMA_CHxCTL register so that the address update method is self-increasing or self-decreasing.

DMA_SNSEQCTLx, DMA_SNSEQCTLBx These two registers share the same address and are both used to define discontinuous transfers. Which one to use depends on whether or not reset is enabled for that channel. DMA_SNSEQCTLx is used when reset is not enabled, and DMA_SNSEQCTLBx is used when reset is enabled.

14.5.19 Destination device discontinuous address transfer control register (DMA_DNSEQCTLx) (x=0~3)

Reset value: 0x0000_0000



position	Marker	Place Name	Function	Reading and writing
b31-b20	DNSCNT[11:0]	Amount of data for destination address jumping	Set the size of the amount of data to be transferred before the destination address is jumped. The target device jumps the target address at the offset specified by DOFFSET after every DNSCNT data transmission. The register is set to 10 for address jump after every 10 data transfers, and to 0 for address jump after every 4096 data transfers.	R/W
b19-b0	DSOFFSET[19:0]	Address offset for target address jump	Sets the offset of the destination address jump when discontinuous addresses are transmitted. The offset is relative to the current transfer address, which is the last transfer address before the jump. The direction of the jump is jumped forward or backward according to the value of the channel control register DMA_CHxCTL.DINC. Refer to Figure 14-3. When DMA_CHxCTL.DINC is set to address fixed, discontinuous address transfers will not be valid. The jump address will be calculated based on the number of bits set by the width of the data (DMA_CHxCTL.HSIZE) and the value of DOFFSET. Address offset=DOFFSET×(HSIZE(bit)/8) For example, when DOFFSET is set to 10 and HSIZE is a word (32bit), the address offset is $10 \times 4 = 40$, if HSIZE is a half-word (16bit), the offset is $10 \times 2 = 20$, and if HSIZE is a byte (8bit), the offset is $10 \times 1 = 10$. Destination address of next transmission = Destination address of current transmission ± address offset	R/W

Discontinuous transfers using the target device require configuring the DNSEQEN bit of the DMA_CHxCTL register to be valid and configuring the DINC bit of the DMA_CHxCTL register so that the address update method is self-increasing or self-decreasing.

DMA_DNSEQCTLx, DMA_DNSEQCTLBx These two registers share the same address and are both used to define discontinuous transfers. Which one to use depends on whether or not reset is enabled for that channel. DMA_DNSEQCTLx is used when reset is not enabled, and DMA_DNSEQCTLBx is used when reset is enabled.

14.5.20 Destination device discontinuous address transfer control register

B(DMA_DNSEQCTLBx) (x=0~3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DNSCNTB[11: 0]												DNSDIST [19:16]			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DNSDIST[15: 0]															
position	Marker	Place Name	Function	Reading and writing											
b31-b20	DNSCNTB[11:0]	Amount of data for destination address jumping	Set the size of the amount of data to be transferred before the destination address is jumped. The target device hops at the DNSDIST specified address spacing after each DNSCNTB data block transfer. The data block size is determined by DMA_DTCTLx.BLKSIZE and DMA_CHxCTL.HSIZE.	R/W											
b19-b0	DNSDIST [19:0]	Target discontinuous area address spacing	When discontinuous addresses are transmitted, set the spacing between two discontinuous areas of the target device. The direction of the jump is forward or backward depending on the value of the channel control register DMA_CHxCTL.DINC. Refer to Figure 14-4. When DMA_CHxCTL.DINC is set to address fixed, discontinuous address transfer will be invalid. The address spacing will be calculated based on the number of bits set by the width of the data (DMA_CHxCTL.HSIZE) and the value of DNSDIST. Address spacing=DNSDIST×(HSIZE(bit)/8) For example, when DNSDIST is set to 10 and HSIZE is a word (32bit), the address spacing is 10×4=40, if HSIZE is a half-word (16bit), the spacing is 10×2=20, and if HSIZE is a byte (8bit), the spacing is 10×1=10. Destination address of next transmission = first address of current target discontinuity area ± address spacing	R/W											

Discontinuous transfers using the target device require configuring the DNSEQEN bit of the DMA_CHxCTL register to be valid and configuring the DINC bit of the DMA_CHxCTL register so that the address update method is self-increasing or self-decreasing.

DMA_DNSEQCTLx, DMA_DNSEQCTLBx These two registers share the same address and are both used to define discontinuous transfers. Which one to use depends on whether or not reset is enabled for that channel. DMA_DNSEQCTLx is used when reset is not enabled, and DMA_DNSEQCTLBx is used when reset is enabled.

14.5.21 Chain pointer register (DMA_LLPx) (x=0~3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LLP [31:16]															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LLP[15:2]															

position	Marker	Place Name	Function	Reading and writing
b31-b2	LLP[31:2]	Chain Pointer	When the chain transfer is valid, set the address where the descriptor of the next transfer is located, the address is word-aligned, i.e., LLP[1:0] is fixed to 0	R/W
b1-b0	Reserved	-	Read "0", write "0" when writing	R/W

14.5.22 Channel control register (DMA_CHxCTL) (x=0~3)

Reset value: 0x0000_1000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	IE	LLP RUN	LLP EN	HSIZE[1:0] 1	DNSE QEN	SNSE QEN	DRP TEN	SRP TEN	DINC[1:0]	SINC[1:0]			

position	Marker	Place Name	Function	Reading and writing
b31-b13	Reserved	-	Read "0", write "0" when writing	R/W
b12	IE	Interrupt enable bit	Configure whether the channel generates interrupts. 0: The channel does not generate interrupts 1: The channel generates an interrupt	R/W
b11	LLPRUN	Chain transfer mode selection	When the chain transfer is valid, set whether to start the transfer corresponding to the new descriptor immediately after loading the new descriptor pointed by the chain pointer when the current transfer is completed 0: Do not transfer immediately, wait for the next transfer request to be generated and start the transfer 1: Transfer starts immediately after the new descriptor is loaded	R/W
b10	LLPEN	Chained transmission enable	0: Chain transmission invalid 1: Chain transmission is effective	R/W
b9-b8	HSIZE[1:0]	Width of transmitted data	00: 8bit 01: 16bit 10, 11: 32bit	R/W
b7	DNSEQEN	Destination address discontinuous transmission enable	0: Discontinuous address transmission is not allowed 1: Discontinuous address transmission is allowed	R/W
b6	SNSEQEN	Source address discontinuous transmission enable	0: Discontinuous address transmission is not allowed 1: Discontinuous address transmission is allowed	R/W
b5	DRPTEN	Target repeat transmission function enable bit	Set whether to allow the target address to reload the initial value 0: no reload 1: Heavy load	R/W
b4	SRPTEN	Source repeat function enable bit	Set whether to allow the source address to reload the initial value 0: No reload 1: Heavy load	R/W

b3-b2	DINC[1:0]	Update method of destination address	00: Fixed 01: Incremental 10, 11: decreasing	R/W
b1-b0	SINC[1:0]	Update method of source address	00: Fixed 01: Incremental 10, 11: decreasing	R/W

14.5.23 Channel Monitor Register(DMA_MONSARx, DMA_MONDARx, DMA_MONDTCTLx, DMA_MONRPTx, DMA_MONSSEQCTLx, DMA_MONDNSEQCTLx) (x=0~3)

These monitoring registers correspond to the corresponding channel configuration registers with the same register bit configuration, but all are read-only registers.

The channel configuration register remains unchanged before and after the DMA transfer, while the channel monitoring register is updated after each request corresponding to a transfer completed by the DMA, i.e., after each data block transfer is completed. The updates are as follows and in the following manner:

- DMA_MONSARx.SAR[31:0], DMA_MONDARx.DAR[31:0]: updated to the address of the next transmission by the fixed/increasing/decreasing/reloading/discontinuous hopping method set by the channel configuration register.
- DMA_MONDTCTLx.CNT[15:0]: subtract 1, or keep it as 0 if it is already 0.
- DMA_MONRPTx.SRPT[9:0], DRPT[9:0]: When the channel reset is invalid, subtract the block size, minus when it reaches 0 or reload the DMA_RPTx setting when the value is smaller than the block size. When reset is valid, subtract 1, and reload the DMA_RPTBx setting when it reaches 0.
- DMA_MONSSEQCTLx.SNSCNT[11:0],
DMA_MONDNSEQCTLx.DNSCNT[11:0]:
When the channel reset is invalid, subtract the block size, and when it reaches 0 or when the value is less than the block size value, the original DMA_SNSEQCTLx/DMA_DNSEQCTLx setting is set. When reset is valid, subtract 1 and reload the DMA_SNSEQCTLBx/DMA_DNSEQCTLBx setting when the value is reduced to 0.

Monitor register bits other than the above remain the same as the configuration register.

14.6 Precautions for use

- 1 The DMA registers only support 32bit read/write, **8/16bit** read/write operations are not valid.
- 2 Write channel configuration registers are invalid during DMA transfer. Channel configuration registers include: DMA_SARx, DMA_DARx, **DMA_DTCTLx**, DMA_RPTx, DMA_RPTBx, DMA_SNSEQCTLx, DMA_SNSEQCTLBx, DMA_DNSEQCTLx, DMA_DNSEQCTLBx, DMA_LLPx, DMA_CHxCTL (x=0~3). Please make sure to write the above registers when DMA is idle, or read them out after writing them to confirm if they are written successfully.

15 Voltage Comparator (CMP)

15.1 Introduction

The voltage comparator (CMP) is a peripheral module that compares two analog voltages (INP and INM) and outputs the comparison result. The positive and negative voltages of each comparison channel

The voltages have 4 input sources, each of which can be selected for a single comparison, or multiple positive voltages can be selected for a scan comparison with the same negative voltage.

The comparison results can be read from registers or output to external pins, and interrupts and events can be generated.

This series is also equipped with two **8-bit** digital-to-analog converters (hereafter referred to as 8bitDACs) whose analog outputs can be used as negative voltage input sources for CMP.

Table 15-1 CMP Detailed Specifications

Proj ects	Spe cific atio n
Compare channels	3 comparison channels: CMP1~3
Positive voltage input source	12 input sources: 10 external analog inputs, 2 internal PGA outputs
Negative side voltage input source	7 input sources: 4 external analog inputs, 1 internal Vref, 2 internal D/A outputs
Compare Results	Read from register or output to external pin VCOUT, with selectable polarity
Working mode	Single comparison mode and scan comparison mode
Digital filtering	7 sampling frequencies selectable, can be turned off when filtering is not needed
Interrupts and events	Valid edges of comparison results can be detected, and interrupts and peripheral trigger events can be generated
Low power control	Module stop function reduces system power consumption

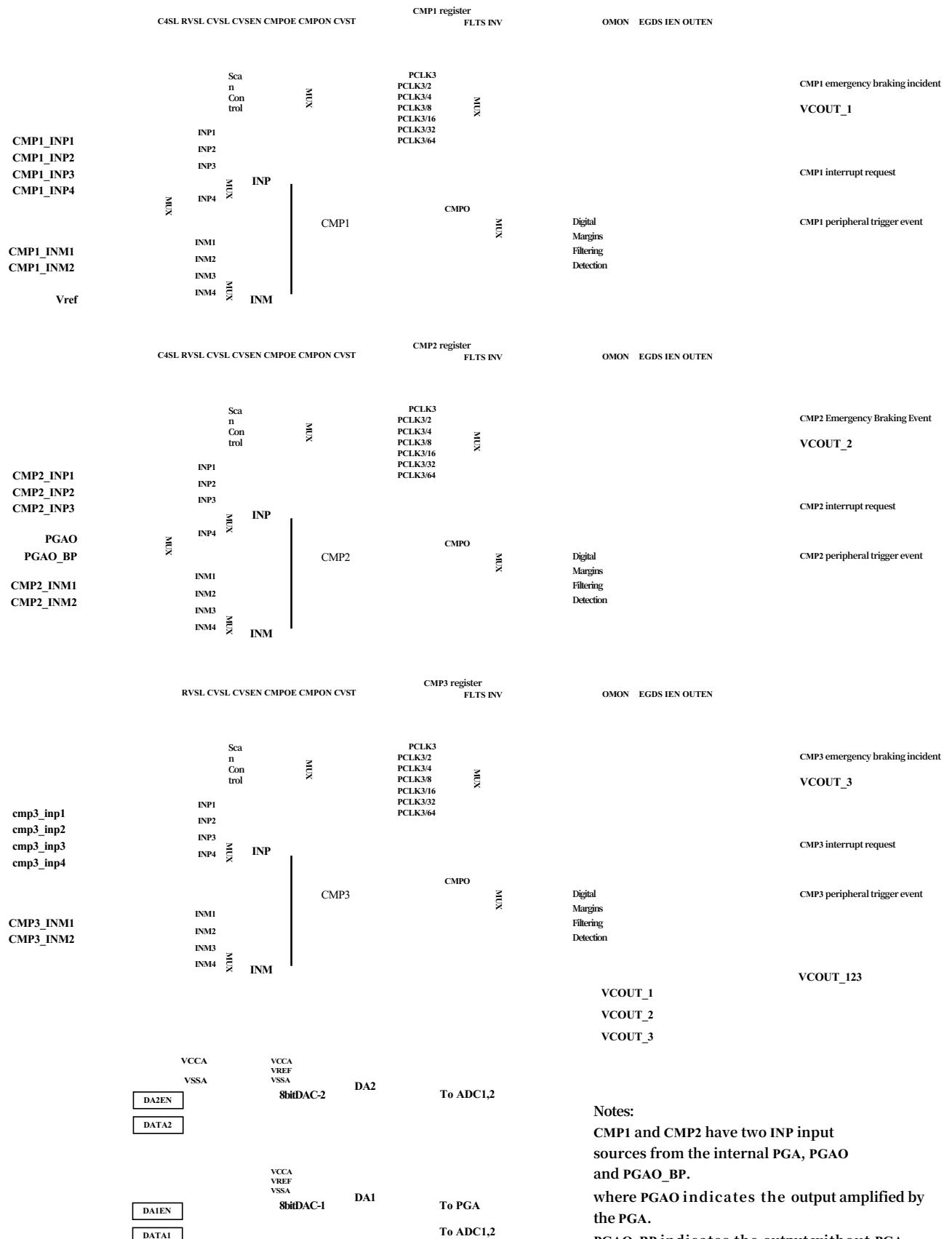
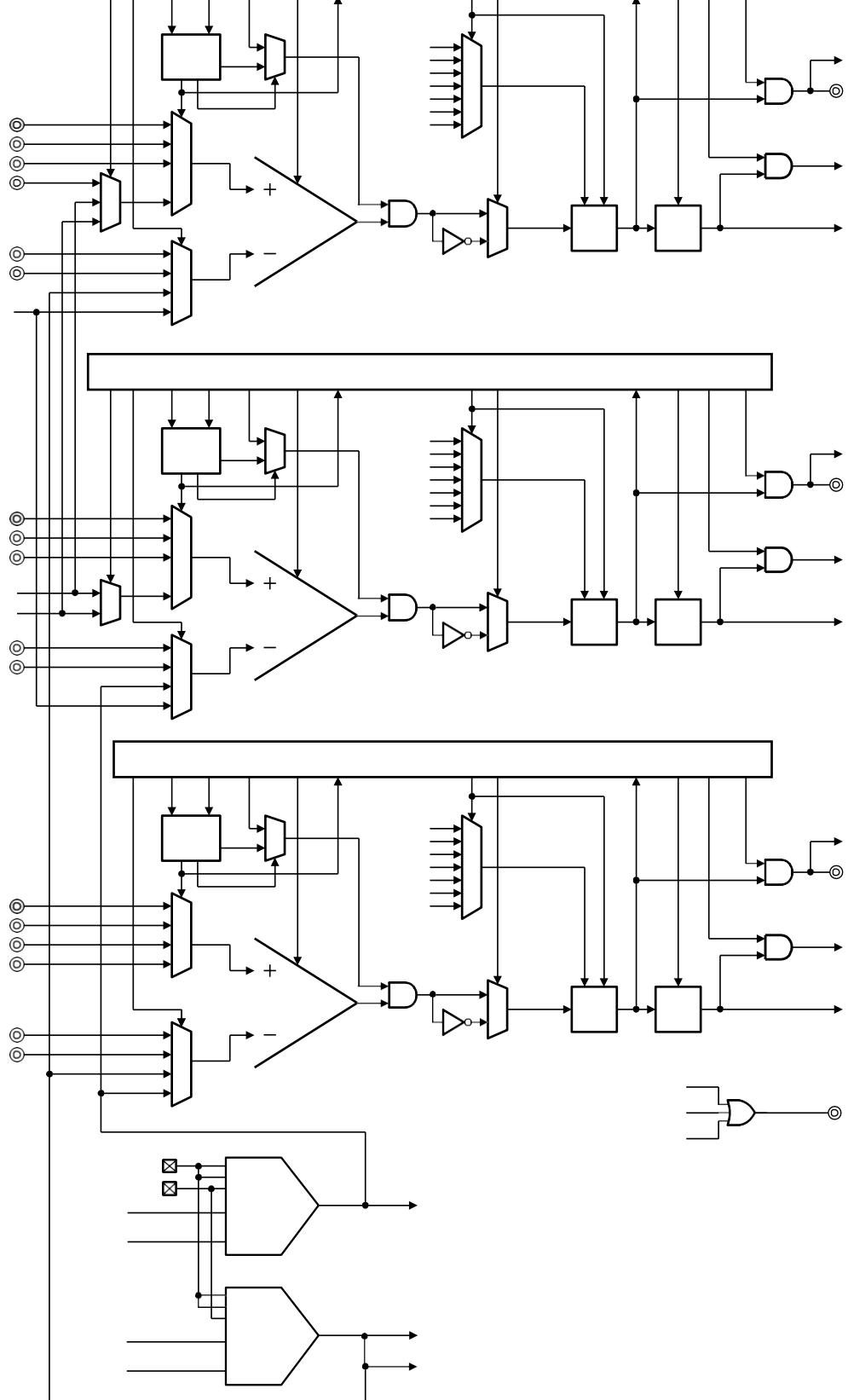


Figure 15-1 CMP, 8bitDAC function connection diagram



15.2 Function Description

15.2.1 Voltage Comparison

When the CMP is operating in single comparison mode, the comparison of the two voltages is achieved by selecting any of the comparison channels and choosing the positive INP and negative INM for them. Figure 15-2 shows a schematic of the CMP operation. If the positive output is set, the CMP outputs high when INP is higher than INM and low when INP is lower than INM. Conversely, if the negative polarity output is set, the CMP outputs low when INP is above INM and high when INP is below INM. Finally, the comparison result can be read from the respective comparison result monitoring register CMPMON.OMON bit of each comparison channel or output to the external pin VCOUT.

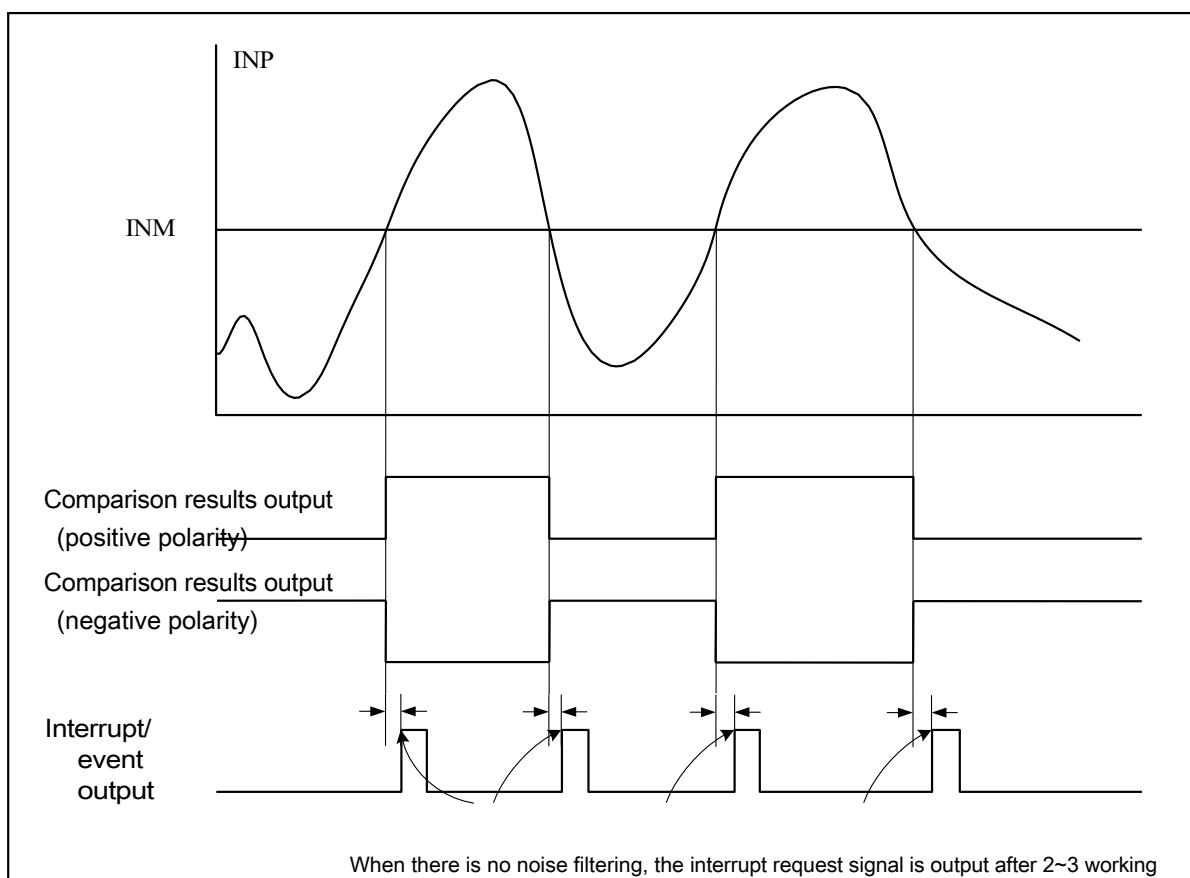


Figure 15-2 CMP working diagram

15.2.2 Digital filtering

Each comparison channel has a digital filtering circuit to filter the comparison result for noise. The filter circuit compares the comparator output CMPO after three samples, and the sampled result will be output as the comparison result if the three samples are the same, and the comparison result will be output if the three samples are not the same. The sampling frequency can be selected in 7 steps, please refer to the function description of register **CMP_CTRL.FLTSL[2:0]** for details. The filtering function can also be turned off, when the comparison result output is exactly the same as the comparator output CMPO.

15.2.3 Interrupts and events

Interrupt requests and events triggered by other peripherals can be generated when the comparison result output changes, and the edges of the generated interrupts and events can be selected as rising, falling or double edges of the comparison result output. The interrupt request of comparison channel 1 (CMP1) can also wake up/stop the low power mode, but the interrupt edge must be selected as rising edge and the digital filtering must be turned off.

Please refer to the following procedure to set up the comparison interrupt and event for the first time.

- 1) Set CMP_VLTSEL and select INP and INM input sources.
- 2) Set CMP_CTRL.INV to select comparator output polarity.
- 3) Set CMP_CTRL.EDGSL[1:0] to select the interrupt generation edge.
- 4) Set CMP_CTRL.CMPON to start the comparator and wait for 300ns stabilization time.
- 5) Set CMP_CTRL.CMPOE to allow comparator output.
- 6) If interrupts are used, set CMP_CTRL.IEN to allow interrupts.

If using events, select the trigger event of the started peripheral as the CMP compare event.

15.2.4 Scan comparison mode

CVSL[3:0] bits to select two or more INPs for comparison, the CMP enters the scan comparison mode, and then sets the CMP_CTRL.CVSEN bits to "1" to start the scanning action. The INM input remains unchanged while the INP input is automatically switched according to the set scan period. If a valid edge selected by the CMP_CTRL.EDGSL[1:0] bits occurs during the scan when the comparison result is monitored, the CVSEN bits in register CMP_CTRL.CVSEN are automatically cleared and the scan is paused. At this point, the register CMP_OUTMON.CVST[3:0] bits can be read to determine the current INP input. When the register CMP_CTRL.CVSEN bit is written to "1" again, the scanning action continues.

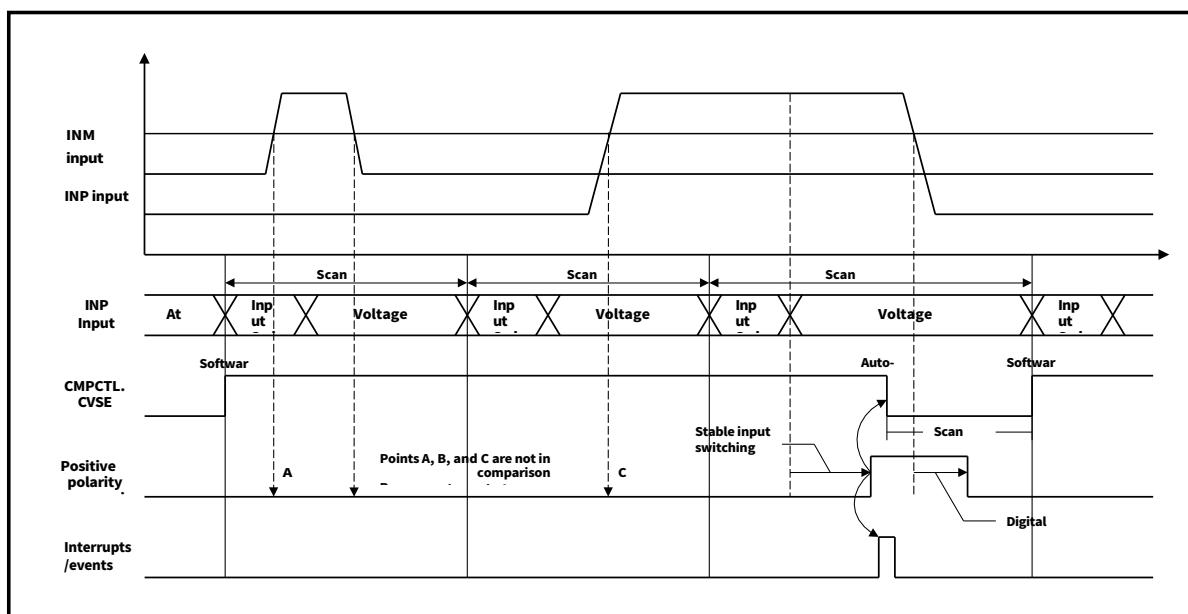


Figure 15-3 Scan mode action diagram

The diagram above shows the action of scanning two INP inputs. During the first two scan cycles, the comparison result output remains unchanged as neither INP input 1 nor INP input 2 changes within their respective comparison time windows. When scan cycle 3 is entered, INP input 1 is higher than INM causing the comparator to invert and the output is digitally filtered and passed to the comparison result output. At this point, the scanning action is automatically stopped and interrupts and events are generated. The CMP then remains in the INP input 1 to INM comparison state until the software restarts the scan mode.

Please refer to the following procedure to set up using the scan comparison mode.

- 1) Set CMP_VLTSEL[3:0] to select INP and INM input sources (INP selects two or more)
- 2) Set CMP_CVSSTB to set the voltage switching stabilization time.
- 3) Set CMP_CVSPRD to set the voltage sweep period.
- 4) Set CMP_CTRL.INV to select comparator output polarity.

-
- 5) Set **CMP_CTRL.FLTS[2:0]** to select the digital filtering sampling frequency.
 - 6) Set **CMP_CTRL.EDGSL[1:0]** to select the interrupt generation edge.
 - 7) Set **CMP_CTRL.IEN** to allow interrupts.
 - 8) Set **CMP_CTRL.CMPON** to start the comparator and wait for 300ns stabilization time.

9) Set CMP_CTRL.CMPOE to allow comparator output.

10) Set CMP_CTRL.CVSEN to start scanning.

The voltage switching stabilization time and the voltage sweep period correspond to the PCLK3 cycle number. To ensure correct operation, make sure that the setting value meets the condition of " $\text{Sweep period} > \text{Switching stabilization time} + \text{Filter sampling period} \times 4 + 5$ ".

15.2.5 8bit-DAC setting

The 2 8bit-DACs provide the CMP with two internal voltages DA1 and DA2. The output voltages during operation are

$$\text{Output voltage} = \text{VCCA} \times \text{conversion}$$

data/255 DA1, DA2 conversion data is set by CMP_DADR1, CMP_DADR2 respectively.

Please refer to the following procedure for setting when using:

- 1) Set CMP_DADR1, CMP_DADR2 to set the conversion data.
- 2) Set CMP_DACR to start DA1 and DA2.
- 3) Wait for the DA conversion stabilization time.
- 4) Sets the comparator.

15.3 Cautions

15.3.1 Module stop function

The CMP has a module stop function, which allows the digital part of the module to be turned off by setting the module stop register. the CMP is initially stopped, and the CMP register can only be accessed when the module is set to work. Please refer to the Low Power chapter for detailed description.

15.3.2 Action when the module is stopped

When the CMP enters the stop state in the operating state, the analog part of the CMP continues to operate and the power consumption is equal to the operating state. To reduce the power consumption, set CMP_CTRL.CMPON to "0" to stop the operation of the analog part of CMP.

Similarly, when the 8bit-DAC enters the stop state in the operating state, the analog part of the DA continues to operate and the power consumption is equal to the operating state. To reduce the power consumption, set the DA1EN and DA2EN bits of DACR to "0" to stop the operation of the analog part of the 8bit-DAC.

15.3.3 Stopping the action in low-power mode

When HC32F46xx enters Stop Low Power mode in CMP operation state, CMP will continue to HC32F460_F45x_A460 Series Reference

operate and the power consumption is equal to the level before entering Stop Low Power mode. If you need to further reduce the power consumption in Stop Low Power mode, please clear CMP_CTRL.CMPON to "0" **before entering Stop Low Power mode** to stop CMP operation.

Similarly, when the HC32F46xx enters Stop Low Power mode during 8bit-DAC conversion, the D/A output will be held and the power consumption will be equal to the level before entering Stop Low Power mode. To further reduce the power consumption in Stop Low Power mode, set the DA1EN and DA2EN bits of DACR to "0" before entering Stop Low Power mode to stop the 8bit-DAC operation.

15.3.4 Action in power-down low-power mode

When HC32F46xx enters the power-down low-power mode, both CMP and 8bit-DAC will stop working. the CMP comparison result output is Low and 8bit-DAC output becomes high resistance state.

15.4 Register Description

CMP1 Base Address:

0x4004_A000 CMP2 Base

Address: 0x4004_A010 CMP3

Base Address: 0x4004_A020

Table 15-2 List of CMP registers

Register Name	Sym bols	Offset Address	Bit width	Reset value
CMP control register	CMP_CTRL	0x000	16	0x0000
CMP voltage selection register	CMP_VLTSEL	0x002	16	0x0000
CMP result monitoring register	CMP_OUTMON	0x004	16	0x0000
CMP stability time register	CMP_CVSSTB	0x006	16	0x0005
CMP scan period register	CMP_CVSPRD	0x008	16	0x000F
CMP with 8bit DAC data register 1	CMP_DADR1	0x100	16	0x0000
CMP with 8bit DAC data register 2	CMP_DADR2	0x102	16	0x0000
8bit DAC control register for CMP	CMP_DACR	0x108	16	0x0000
CMP internal reference voltage AD conversion register	CMP_RVADC	0x10C	16	0x0000

Caution:

-CMP_DADR1,CMP_DADR2,CMP_DACR are common registers and only exist in CMP1.

15.4.1 CMP control register (CMP_CTRL)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CMPON	CMPOE	INV	OUTEN	-	-	-	CVSEN	IEN	EDGSL[1:0]	-	-	FLTSL[2:0]			

position	Marker	Place Name	Function	Reading and writing
b15	CMPON	Comparator work permit	0: Comparator stop (comparator output Low) (reset value) 1: The comparator works with the following notes: -The comparator is set to "1". After setting the comparator operating permit CMPON to "1", it is necessary to wait for about 300ns of stable operation before subsequent operations can be performed. Work.	R/W
b14	CMPOE	Comparator output license	0: Comparator output disable (comparator output Low) (reset value) 1: Comparator output license	R/W
b13	INV	Comparator output polarity selection	0: Comparator positive polarity output (reset value) 1: Negative polarity output of comparator (comparator output is inverted) Caution: -Please rewrite INV when the comparator output is disabled (i.e., CMPOE bit is "0"). Changing the INV bit may cause an interrupt or peripheral trigger event, so please set this register in the interrupt disabled or peripheral trigger function disabled state. After the register is set, please clear the corresponding interrupt flag.	R/W
b12	OUTEN	Compare results output license	0: Disable comparison result output (comparison result output Low) (reset value) 1: Allow comparison result output	R/W
b11~b9	Reserved	-	Read "0", write "0" when writing	R
b8	CVSEN	Comparative Voltage Scan License	0: Comparison voltage scan stopped (reset value) 1: Comparative voltage scan starts with the following notes: - When a valid edge of the comparison result is detected, CVSEN will automatically clear the zero, write "1" again if you want to continue scanning.	R/W
b7	IEN	Compare interruption licenses	0: Disable comparison interrupt (reset value) 1: Allow comparison interruptions	R/W
b6~b5	EDGSL	Comparison result effective edge	0: edge of the comparison result is not detected (reset value)	R/W

[1:0]		Select	0 1: Rising edge of the detection comparison result 1 0: Detect the falling edge of the comparison result 1 1: Detect the rising and falling edges of the comparison result	
b4~b3	Reserved	-	Read "0", write "0" when writing	R
b2~b0	FLTSL	Filter sampling selection	0 0 0: No noise filtering is used (reset value) [2:0] 0 0 1: Sampling with PCLK3 0 1 0: 2-division sampling using PCLK3 0 1 1: 4-part sampling using PCLK3 1 0 0: 8-division sampling using PCLK3 1 0 1: 16-division sampling using PCLK3 1 1 0: 32-division sampling using PCLK3 1 1 1: 64-component sampling using PCLK3	R/W
Caution:				

-
- Please rewrite **FLTSL[2:0]** when the comparator output is disabled (i.e., CMPOE bit is "0"). **fltsl[2:0] is set** from When "000b" is switched to other values, use the filtered output after 4 samples as an interrupt request or peripheral trigger event.
 - Changing **FLTSL[2:0]** may cause an interrupt or peripheral trigger event, so please set this register in the interrupt disable or peripheral trigger function disabled state. After the register is set, please clear the corresponding interrupt flag.
-

15.4.2 CMP Voltage Select Register (CMP_VLTSEL)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	C4SL[2:0]			CVSL[3:0]			-	-	-	-	-		RVSL[3:0]		

position	Marker	Place Name	Function	Reading and writing
b15	Reserved	-	Read "0", write "0" when writing	R
b14~b12	C4SL[2:0]	INP4 input selection	0 0 0: No input (reset value) 0 0 1: Select internal PGAO output (valid only for CMP1 and CMP2) 0 1 0: Select internal PGAO_BP output (valid only for CMP1 and CMP2) 1 0 0: Select external CMP1_INP4 input (valid only for CMP1) Caution: 1. The setting is not valid for CMP3. 2. Setting other values is prohibited. 3. When rewriting C4SL[2:0], please refer to the following steps: (1) CMP_CTRL.CMPOE bit is cleared to "0". (2) C4SL[2:0] bits are written to the new value (note that only 1 bit is ** 1 **) (3) Wait for the input to stabilize. (4) CMP_CTRL.CMPOE position "1". (5) Clear the corresponding interrupt flag bit.	R/W

b11~b8	CVSL[3:0]	INP input selection	0 0 0 0: No input (reset value) 0 0 0 1: Select INP1 0 0 1 0: Select INP2 0 0 1 1: Select INP1, INP2 (scan mode) 0 1 0 0: Select INP3 0 1 0 1: Select INP1, INP3 (scan mode) 0 1 1 0: Select INP2, INP3 (scan mode) 0 1 1 1: Select INP1, INP2, INP3 (scan mode) 1 0 0 0: Select INP4 1 0 0 1: Select INP1, INP4 (scan mode) 1 0 1 0: Select INP2, INP4 (scan mode) 1 0 1 1: Select INP1, INP2, INP4 (scan mode) 1 1 0 0: Select INP3, INP4 (scan mode) 1 1 0 1: Select INP1, INP3, INP4 (scan mode) 1 1 1 0: Select INP2, INP3, INP4 (scan mode) 1 1 1 1: Select INP1, INP 2, INP 3, INP 4 (scan mode)	R/W
		Caution:		

1. See "Figure 15" for specific input sources.
 2. Please set C4SL[2:0] to select input source when INP4 is selected.
 3. When rewriting CVSL[3:0], please refer to the following steps:
 (1) CMP_CTRL.CMPOE bit is cleared to "0".
 (2) The CVSL[3:0] bits are written to the new value.
 (3) Wait for the input to stabilize.
 (4) CMP_CTRL.CMPOE position "1".
 (5) Clear the corresponding interrupt flag bit.

b7~b4	Reserved	-	Read "0", write "0" when writing	R
b3~b0	RVSL[3:0]	INM input selection	0 0 0 0: No input (reset value) 0 0 0 1: Select INM1 0 0 1 0: Select INM2 0 1 0 0: Select INM3 1 0 0 0: Select INM4 Caution: 1. See "Figure 14" for specific input sources. 2. Setting other values is prohibited. 3. When changing RVSL[3:0], please refer to the following steps: (1) CMP_CTRL.CMPOE bit is cleared to "0". (2) The RVSL[3:0] bits are written to the new value (note that only 1 bit is "1") (3) Wait for the input to stabilize. (4) CMP_CTRL.CMPOE position "1". (5) Clear the corresponding interrupt flag bit.	R/W

15.4.3 CMP result monitoring register (CMP_OUTMON)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-		CVST[3:0]		-	-	-	-	-	-	-	-	OMON

position	Marker	Place Name	Function	Reading and writing
b15~b12	Reserved	-	Read "0", write "0" when writing	R
b11~b8	CVST[3:0]	INP input status bit	0 0 0 0: No input for current INP 0 0 0 1: Current INP input selected is INP1 0 0 1 0: The current INP input is selected as INP2 0 1 0 0: The current INP input selected is INP3 1 0 0 0: The current INP input selected is INP4 Note: 1. This register is read only, write all "0" when writing. 2. The specific input sources are shown in Figure 15-1.	R
b7~b1	Reserved	-	Read "0", write "0" when writing	R
b0	OMON	Comparison result monitoring bit	0: The comparison result is output as "0" (reset value) 1: The result of the comparison is output as "1" Note that 1. This register is read only, write all "0" when writing. 2. When setting comparator operation with CMP_CTRL.FLTS[2:0]=000b (no noise filtering circuit is not used), determine the comparison status by reading the OMON bit status the same way 2 times.	R

15.4.4 CMP Stabilization Time Register (CMP_CVSSTB)

Reset value: 0x0005

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	STB[3:0]

position	Marker	Place Name	Function	Reading and writing
b15~b4	Reserved	-	Read "0", write "0" when writing	R
b3~b0	STB[3:0]	Input switching stabilization time	<p>Set the output stabilization time of CMP when switching INP input in scan compare mode. The reset value is 0x5, and the setting value can be any value between 0x0 ~ 0xF.</p> <p>CMP output stabilization time = PCLK3 cycle × STB setting value</p> <p>For example, if PCLK3 is 40MHz and STB is set to 0x5, then the scan period = 25(nS) × 0x5 = 125(nS)</p> <p>Note that</p> <ol style="list-style-type: none"> 1. Please refer to "Comparator Characteristics - Input Channel Switching Stability" in the data sheet Electrical Characteristics. 2. To rewrite the STB, stop scanning and switch to single comparison mode first. 	R/W

15.4.5 CMP Compare Voltage Sweep Period Register (CMP_CVSPRD)

Reset value: 0x000F

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PRD[7:0]

position	Marker	Place Name	Function	Reading and writing
b15~b8	Reserved	-	Read **0" for read, write "0" for write	R
b7~b0	PRD[7:0]	Comparison voltage scan period	<p>Set the time interval for switching INP input in scan compare mode, i.e. the scan period. The reset value is 0x0F, and the setting value can be any value between 0x0F ~ 0xFF.</p> <p>Scan period = PCLK3 period × PRD setting value</p> <p>For example, if PCLK3 is 40MHz and PRD is set to 0x50, then the scan period = 25(nS) × 0x50 = 2(uS)</p> <p>Caution:</p> <ol style="list-style-type: none"> 1. To ensure correct operation, please check when setting PRD Scan period > Switching stabilization time + Filter sampling period × 4 + 5 2. To rewrite the PRD, stop scanning and switch to single comparison mode first. 	R/W

15.4.6 CMP with 8bit-DAC control register (CMP_DACR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	DA2 EN	DA1 EN

position	Marker	Place Name	Function	Reading and writing
b15~b2	Reserved	-	Please read "0" when reading, and write "0" when writing.	R
b1	DA2EN	DA2 start bit	0: DA2 stop (reset value) 1: DA2 work	R/W
b0	DA1EN	DA1 start bit	0: DA1 stop (reset value) 1: DA1 work	R/W

15.4.7 CMP with 8bit-DAC data register (CMP_DADR1,CMP_DADR2)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-							DATA[7:0]

position	Marker	Place Name	Function	Reading and writing
b15~b8	Reserved	-	Please read "0" when reading, and write "0" when writing.	R
b7~b0	DATA[7:0]	DA conversion data	Any value between 8'h00 (reset value) and 8'hFF	R/W

15.4.8 CMP internal reference voltage AD conversion register (CMP_RVADC)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
							WPRT[7:0]	-	-	-	VREF SW	-	-	DA2 SW	DA1 SW

position	Marker	Place Name	Function	Reading and writing
b15~b8	WPRT[7:0]	Write protect bit	8'h55 : VREFSW, DA2SW, DA1SW bits write valid others : VREFSW, DA2SW, DA1SW bit write invalid Note: WPRT will revert to "8'h00" when writing any value other than "8'h55" .	R/W
b7~b5	Reserved	-	Please read "0" when reading, and write "0" when writing.	R
b4	VREFSW	Internal Vref AD changeover switch	0: Internal Vref AD conversion path disconnected 1: Internal Vref AD conversion path connection Note: VREFSW, DA2SW, DA1SW can only have one bit as "1".	R/W
b3~b2	Reserved	-	Please read "0" when reading, and write "0" when writing.	R
b1	DA2SW	DA2 AD transfer switch	0: DA2 AD conversion path disconnected 1: DA2 AD conversion path connection Note: VREFSW, DA2SW, DA1SW can only have one bit as "1".	R/W
b0	DA1SW	DA1 AD transfer switch	0: DA1 AD conversion path disconnected 1: DA1 AD conversion path connection Note: VREFSW, DA2SW, DA1SW can only have one bit as "1".	R/W

16 Analog to Digital Conversion Module (ADC)

16.1 Introduction

The ADC is a successive approximation analog-to-digital converter with up to 12-bit resolution and supports up to 17 analog input channels to convert analog signals from external pins as well as from inside the chip. These analog input channels can be arbitrarily combined into a sequence, and a sequence can be converted in a single scan, or in a continuous scan. The ADC module also features an analog watchdog function that monitors the conversion results of any given channel to see if the user-set threshold value is exceeded.

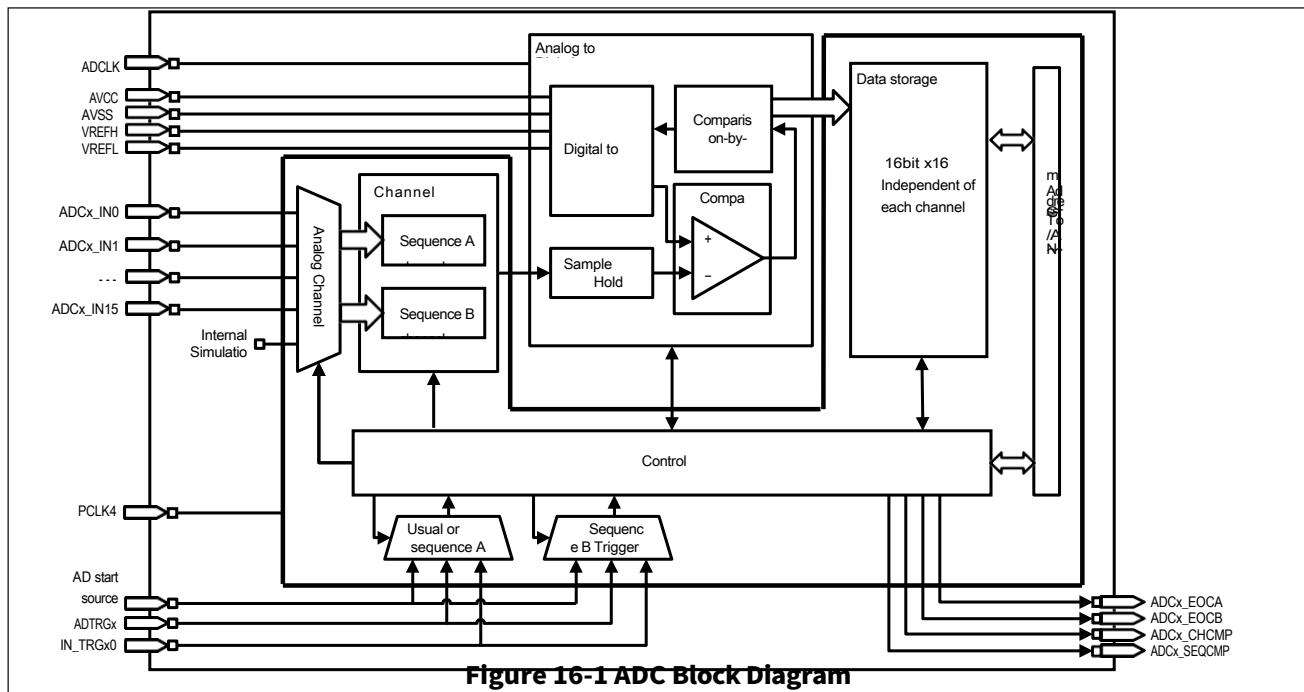
ADC Key Features:

- High Performance
 - Configurable for 12-, 10-, and 8-bit resolution
 - The frequency ratio between the digital interface clock PCLK4 and the A/D converter clock ADCLK can be selected:
 $PCLK4:ADCLK=1:1, 2:1, 4:1, 8:1, 1:2, 1:4$ ADCLK can be selected as a PLL clock asynchronous to the system clock HCLK, where $PCLK4:ADCLK=1:1$
 - Sampling rate: 2.5MSPS ($PCLK4=ADCLK=60MHz$, 12-bit, sampling 11 cycles)
 - Independent programming of sampling time for each channel
 - Independent data register for each channel
 - Data register configurable data alignment
 - Continuous multiple conversion averaging function
 - Analog watchdog to monitor conversion results
 - The ADC module can be set to stop when not in use
- Analog input channels
 - Up to 16 external analog input channels
 - 1 internal reference voltage / **8bit** DAC output detection channel
- Conversion start conditions
 - Software Settings Conversion Start
 - External events trigger the start of the conversion
 - External pins trigger the start of conversion
- Conversion Mode
 - 2 scan sequences A and B, single or multiple channels can be specified at will
 - Sequence A Single Scan
 - Sequence A Continuous Scan

-
- Double sequence scanning, sequence A, B independent selection of trigger source, sequence B priority than A

- Synchronous mode (for devices with two or three ADCs)
- Interrupt and event signal output
 - Sequence A End of Scan Interrupt and Event ADC_EOCA
 - Sequence B End of Scan Interrupt and Event ADC_EOCB
 - Analog watchdog channel comparison interrupt and event ADC_CHCMP, sequence comparison interrupt and event ADC_SEQCMP
 - Each of the 4 event outputs above can initiate DMA

16.2 ADC System Block Diagram



The chip is equipped with 2 ADC module units, each with a different configuration, as shown in the following table:

Table 16-1 Specifications for each ADC unit

Proj ects	Unit 1 (ADC1)		Module 2 (ADC2)
Power supply	AVCC		
	AVSS/VREFL		
Base voltage	VREFH *1		
Analog Channel *2	CH0	ADC1_IN0	ADC12_IN4
	CH1	ADC1_IN1	ADC12_IN5
	CH2	ADC1_IN2	ADC12_IN6
	CH3	ADC1_IN3	ADC12_IN7
	CH4	ADC12_IN4	ADC12_IN8
	CH5	ADC12_IN5	ADC12_IN9
	CH6	ADC12_IN6	ADC12_IN10
	CH7	ADC12_IN7	ADC12_IN11
	CH8	ADC12_IN8	Internal analog channel (reference voltage/8bitDAC output)
	CH9	ADC12_IN9	–
	CH10	ADC12_IN10	–
	CH11	ADC12_IN11	–
	CH12	ADC1_IN12	–
	CH13	ADC1_IN13	–
	CH14	ADC1_IN14	–
	CH15	ADC1_IN15	–
	CH16	Internal analog channel (reference voltage/8bitDAC output)	–
PGA	ADC1_IN0~3, ADC12_IN4~7. Any 1 channel of 8bitDAC_1 output		–
Hardware trigger source	External Pins	ADTRG1	ADTRG2
	Inside and outside the film set	IN_TRG10	IN_TRG20
		IN_TRG11	IN_TRG21

Caution:

- VREFH is available in LQFP100 package, not available in other packages, use AVCC instead of VREFH.
- ADC analog channels CH0~CH15 and the actual input ADCx_INy can be set register free mapping, this table shows the default mapping relationship after reset.

16.3 Function Description

16.3.1 ADC Clock

The ADC module requires the use of two clocks: the analog circuit clock ADCLK, and the digital interface clock PCLK4.

Both clocks come from the divider in the clock controller. ADCLK is equivalent to PCLK2 and is synchronized with PCLK4, which has a frequency ratio of 1:1, 2:1, 4:1, 8:1, 1:2, and 1:4.

ADCLK can be selected as a PLL clock source asynchronous to the system clock HCLK, where PCLK4 is the same as ADCLK and is synchronous to the same frequency.

16.3.2 Channel selection

The ADC module has multiple channels that can be configured to convert in two sequences: Sequence A and Sequence B. Sequences A and B have separate channel selection registers ADC_CHSELRA and ADC_CHSELRB. Sequence A and B have independent channel selection registers ADC_CHSELRA, ADC_CHSELRB. each register represents one channel, e.g. bit0 bit written 1 means convert CH0 write 0 means not convert CH0. two sequences can select any one or more channels independently for conversion. For example, if ADC_CHSELRA is set to 0x0055 and ADCHSELRB is set to 0x0002, the four channels CH0, CH2, CH4 and CH6 will be converted in sequence when the trigger condition of sequence A occurs. When the trigger condition of sequence B occurs, one channel CH1 will be converted.

For the internal analog channels, there are 3 selectable analog quantities: 8bitDAC_1 output, 8bitDAC_2 output, internal reference voltage (approx).

(1.1V) As shown in the figure below. When selecting the internal analog channel as the ADC conversion object, you need to set the internal channel selection register first, and then set the ADC's channel selection register ADC_CHSELRA/ADC_CHSELRB to select the internal channel before starting the ADC conversion.

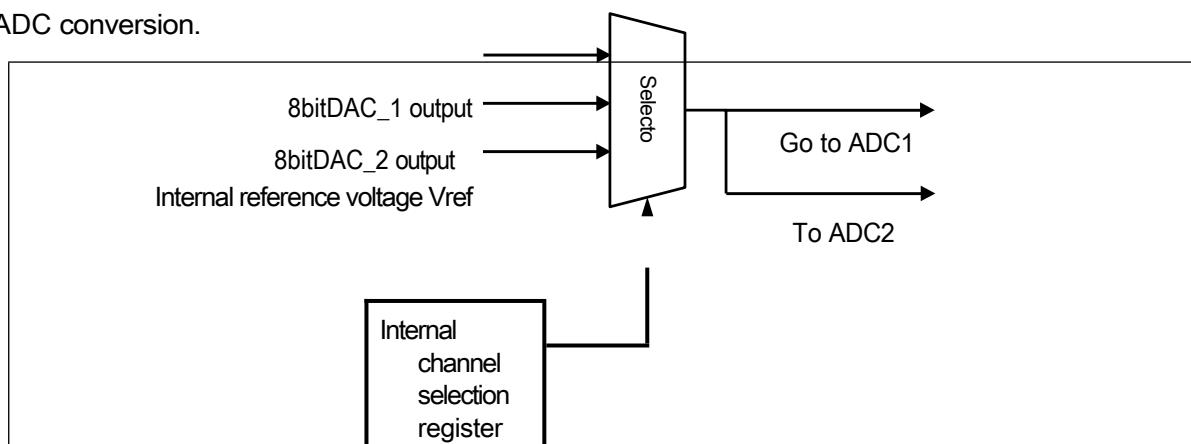


Figure 16-2 Internal Analog Channel Selection

For details on setting the internal channel selection register, please refer to the description of register

CMP_RVADC in the chapter [Voltage Comparator (CMP)] and the description of register PWC_PWCMR in the chapter [Power Control (PWC)].

Table 16-2 Register setting method when converting internal channels

Conversion Goals	ADC_CHSELR	CMP_RVADC	CMP_DADR	CMP_DACR	PWC_PWCMR
8bitDAC_1 -> ADC1	ADC1_CHSELR A1 or ADC1_CHSELRB 1 bit0 write 1, select CH16	Write 0x0001 *1	Write CMP_DADR1 means Constant 8bit DAC_1 output voltage value	bit0 write 1, allow 8bitDAC_1 output	-
8bitDAC_2 -> ADC1				bit1 write 1, allow 8bitDAC_2 output	-
Internal reference voltage vref -> ADC1			Write 0x0010 *1	-	Write 0x80 *2
8bitDAC_1 -> ADC2	ADC2_CHSELR A0 or ADC2_CHSELRB 0 bit8 write 1, select CH8	Write 0x0001 *1	Write CMP_DADR1 means Constant 8bit DAC_1 output voltage value	bit0 write 1, allow 8bitDAC_1 output	-
8bitDAC_2 -> ADC2				bit1 write 1, allow 8bitDAC_2 output	-
Internal Reference Voltage vref -> ADC2			Write 0x0010 *1	-	Write 0x80 *2

Caution:

- CMP_RVADC needs to write 0x5500 first and then write the target setting value.
- The PWC_PWCMR register needs to be turned on first to protect the bit PWC_FPRC.FPRCB1.

Note: Do not select the same channel in sequence A and B. For channels that do not exist, do not set the corresponding registers and leave them as they are after reset.

CH0 indicates channel 0, and the correspondence to the actual analog input channel ADCx_INy can be freely set via register ADC_CHMUXR. For example, for ADC1, CH00MUX set to 0x0 means CH0 is mapped to ADC1_IN0, and set to 0xf means CH0 is mapped to ADC1_IN15. For ADC2, CH00MUX set to 0x0 means CH0 is mapped to ADC12_IN4, and set to 0xF means invalid mapping. Similarly, CH1~CH15 can be mapped by the corresponding ADC_CHMUXR register.

16.3.3 Trigger source selection

The trigger source is selected independently for Sequence A, Sequence B. The selectable trigger sources include external port ADTRGx, internal events IN_TRGx0, IN_TRGx1. Where port ADTRGx falling edge input is valid. IN_TRGx0,IN_TRGx1 are set by register ADC_ITRGSEL0,1, which can select the chip internal rich event source. In addition, write register ADC_STR generates a software trigger signal, which can only be used when the ADC is in standby and only for sequence A.

16.3.4 Sequence A Single Scan Mode

A/D control register ADC_CR0.MS[1:0] is set to 00b to select sequence A single scan mode.

In this mode, when the start condition of sequence A selected by register ADC_TRGSR occurs, or ADC_STR.START is triggered by writing 1, ADC starts to sample and convert all channels selected in sequence A channel selection register ADC_CHSELRA, and the conversion result is stored in the corresponding data register ADC_DR. ADC_STR.START is kept as 1 during the conversion process, and is automatically cleared to 0 when all channels are converted, and the ADC enters the conversion standby state, waiting for the next trigger condition.

If ADC_ICR.EOCAIEN is 1, the end of sequence A conversion flag bit ADC_ISR.EOCAF is set to 1, and the end of sequence A conversion event ADC_EOCA is generated, which can be used to start DMA.

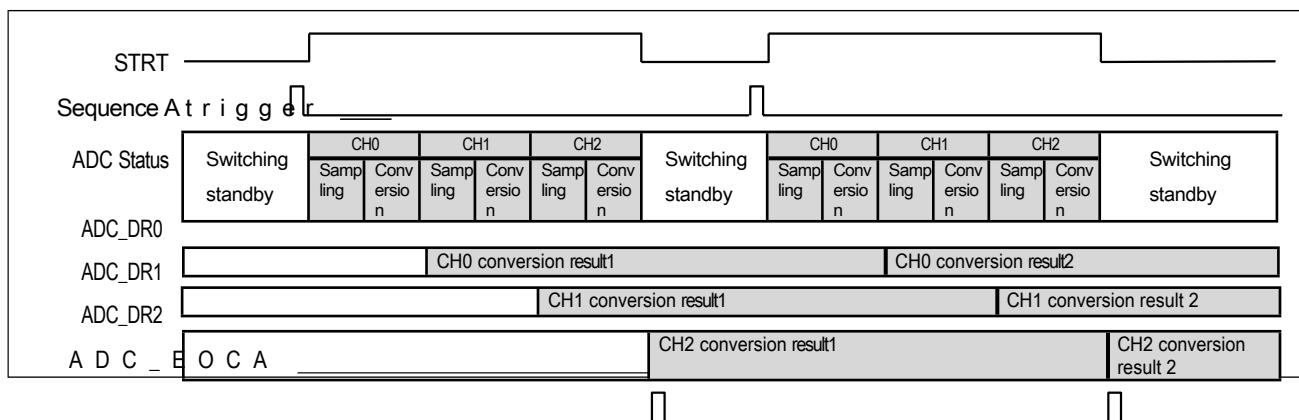


Figure 16-3 Sequence A Single Scan Mode

Sequence A Software flow for single scan mode:

1. Verify that ADC_STR.START is 0 and the ADC is in conversion standby.
2. A/D control register ADC_CR0.MS[1:0] is set to 00b to select sequence A single scan mode.
3. Set the sequence A channel selection register ADC_CHSELRA.
4. Set the sample time register ADC_SSTR.
5. ADC_STR.START Write 1 software to trigger sequence A, or set register ADC_TRGSR to select sequence A trigger condition.
6. Query the end-of-conversion flag bit EOCAF for sequence A.
7. Read each channel data register ADC_DR.
8. Write 0 to clear the EOCAF flag bit in preparation for the next conversion.

The CPU query method in steps 6 to 8 above can also be replaced with an interrupt method, using the ADC_EOCA interrupt to process the conversion data. Or use ADC_EOCA event to start DMA read data.

16.3.5 Sequence A Continuous Scan Mode

A/D control register ADCR0.MS[1:0] is set to 01b to select sequence A continuous scan mode.

Sequence A continuous scan mode is similar to Sequence A single scan mode, except that instead of going into conversion standby after the conversion of the used channel, the continuous mode restarts conversion of Sequence A. The STRT bit is also not automatically cleared to 0.

When continuous scanning needs to be stopped, write 0 to the STRT bit and read STRT to confirm 0 to determine that the ADC is in conversion standby.

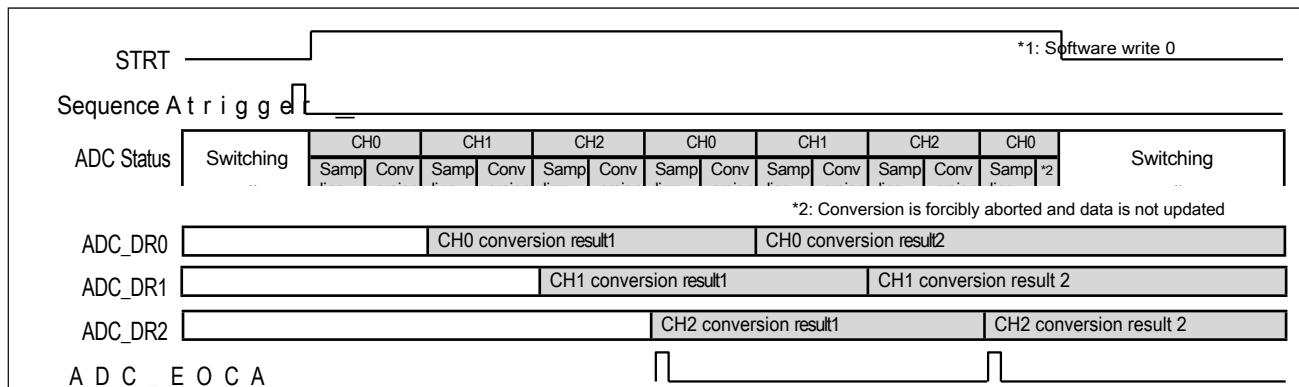


Figure 16-4 Continuous Scan

Software flow for Sequence A continuous scan mode:

1. Verify that ADC_STR.STRT is 0 and the ADC is in conversion standby.
2. A/D control register ADC_CR0.MS[1:0] is set to 01b to select sequence A continuous scan mode.
3. Set the sequence A channel selection register ADC_CHSELRA.
4. Set the sample time register ADC_SSTR.
5. ADC_STR.STRT Write 1 software to trigger sequence A, or set register ADC_TRGSR to select sequence A trigger condition.
6. Query the end-of-conversion flag bit EOCAF for sequence A.
7. Read each channel data register ADC_DR.
8. Write 0 to clear the EOCAF flag bit in preparation for the next conversion.
9. When no further conversion is required, write 0 to the STRT bit and read STRT to confirm 0 to determine that the ADC is in conversion standby. The query method in steps 6 to 8 above can also be replaced with an interrupt method, using the ADC_EOCA interrupt to process the conversion data. Or use ADC_EOCA event to start DMA to read data.

Caution:

- Due to the continuous conversion, the interval between each scan is relatively short, especially when only 1 channel is selected for conversion. It is recommended to use ADC_EOCA event to start DMA reading data to avoid data loss

due to untimely processing in query mode.

16.3.6 Dual sequence scanning mode

A/D control register ADC_CR0.MS[1:0] is set to 10b or 11b to select the dual sequence scan mode, i.e. both sequence A and sequence B can be started by the respective selected trigger conditions.

When MS[1:0]=10b, sequences A and B are equivalent to two independent single-scan sequences. ms[1:0]=11b Sequence A is in continuous scan mode and B is in single-scan mode.

TRGSELA[2:0] to select the trigger source and ADC_CHSELRA to select the channel to be converted. TRGSELB[2:0] selects the trigger source and ADC_CHSELRB selects the channel to be converted.

EOCAF is set to 1 when all channels of sequence A are converted, and the end-of-conversion event ADC_EOCA is generated, if ADC_ISCR. EOCBF is set to 1 when all channels of Sequence B are converted, and the Sequence B end-of-conversion event ADC_EOCB is generated, and if ADC_ISCR.

In dual sequence scan mode, when sequence A competes with sequence B, sequence B will be given priority, i.e. sequence B will have higher priority than sequence A. See the following table for details.

Table 16-3 Various competitions for sequences A and B

A/D conversion	Trigger signal generation	Processing	
		ADC_CR1.RSCHSEL=0	ADC_CR1.RSCHSEL=1
Sequence A conversion process	Sequence A trigger	Invalid trigger signal	
	Sequence B trigger	1) The conversion of sequence A is interrupted and the conversion of sequence B is started 2) After the conversion of sequence B is completed, sequence A is converted from The interrupted channel starts to continue the conversion	1) The conversion of sequence A is interrupted and the conversion of sequence B is started 2) After the conversion of sequence B is completed, sequence A is reconverted from the first channel
Sequence B conversion process	Sequence A trigger	After the conversion of all channels of sequence B is completed, start the conversion of sequence A	
	Sequence B trigger	Invalid trigger signal	
AD idle, sequence A, B triggered at the same time		Sequence B is started first, and after all channels are converted, sequence A conversion is started	

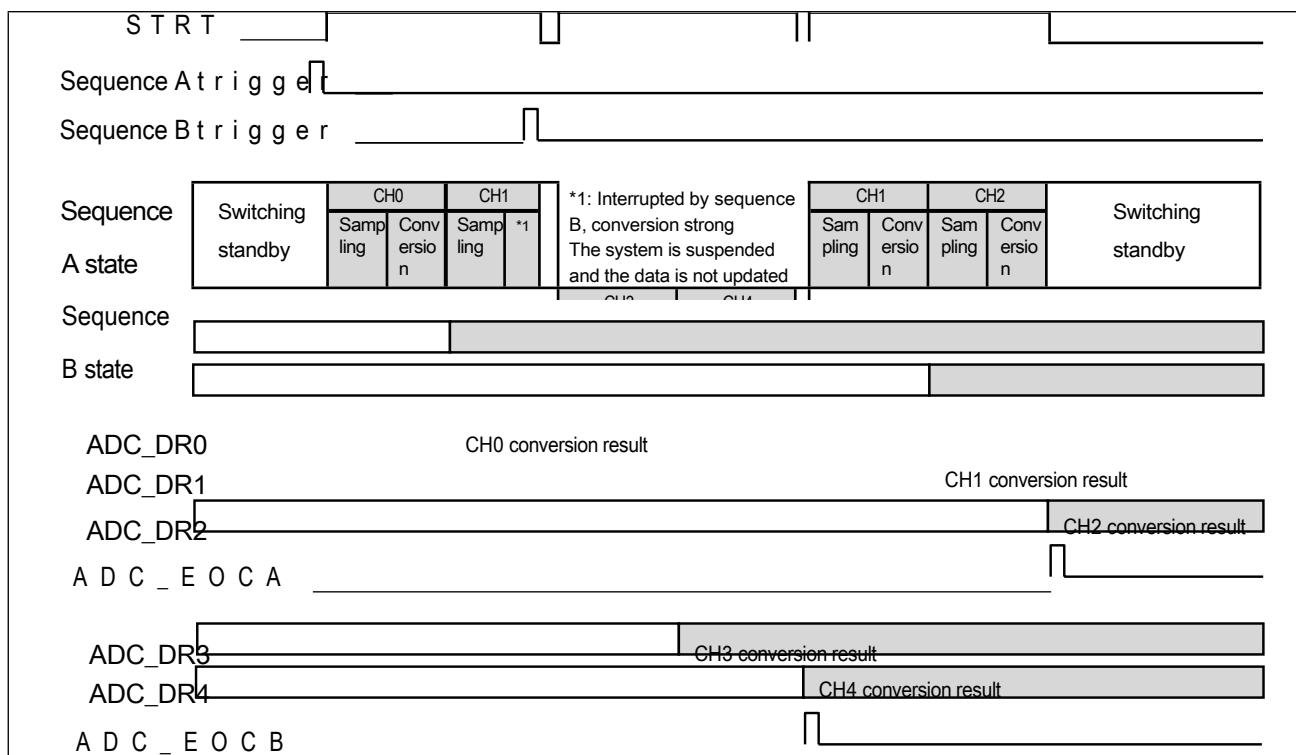


Figure 16-5 Dual Sequence Scan Mode (Sequence A restarted from interrupted channel)

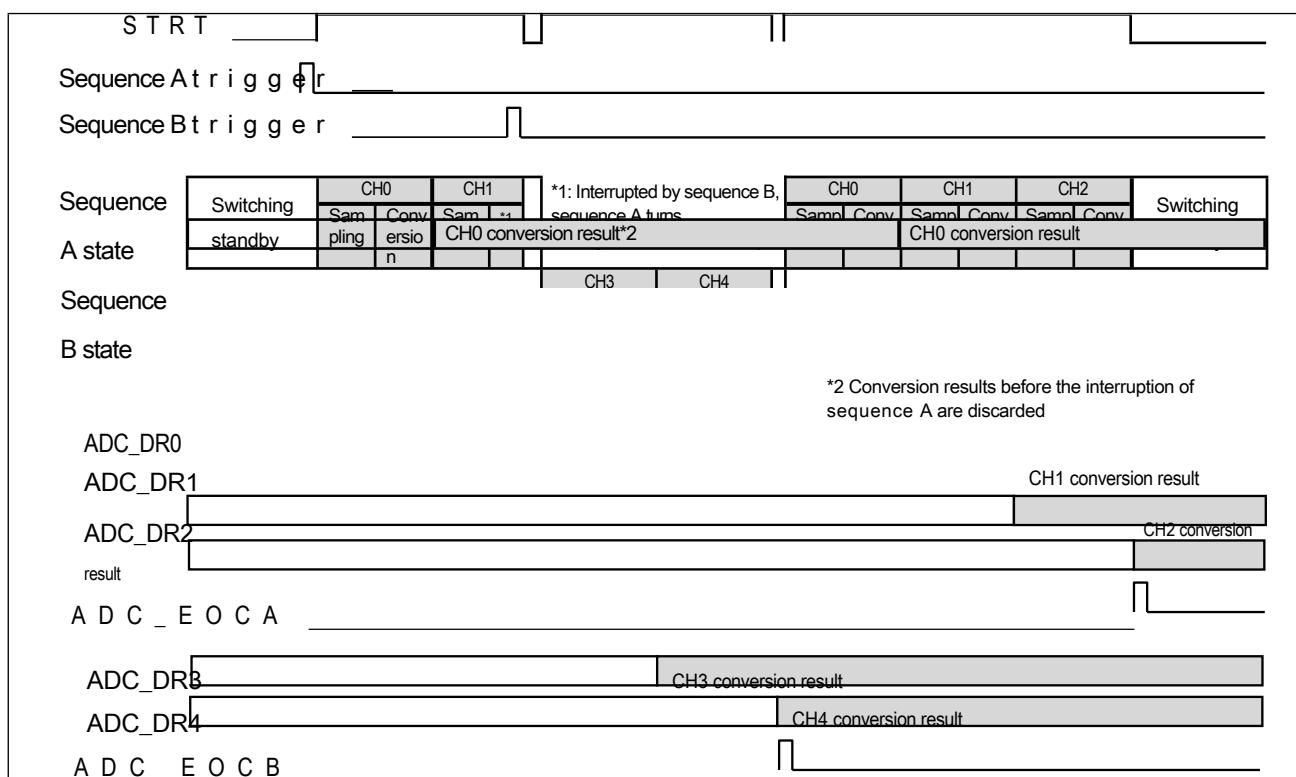


Figure 16-6 Dual Sequence Scan Mode (Sequence A restarted from the first channel)

Software process for dual sequence scanning mode:

1. Verify that ADC_STR.STRT is 0 and the ADC is in conversion standby.
2. A/D control register ADC_CR0.MS[1:0] is set to 10b or 11b to select the dual sequence scan mode.

-
3. Set register ADC_CR1.RSCHSEL to select the start method after sequence A is interrupted.
 4. Set the sequence A channel selection register ADC_CHSELRA.
 5. Set the sequence B channel select register ADC_CHSELRB.
 6. Set the sample time register ADC_SSTR.

7. Set register ADC_TRGSR to select sequence A and B trigger conditions.
8. Interrupt by querying EOCAF, EOCBF, or ADC_EOCA, ADC_EOCB, or start a DMA to process the conversion data after the sequence A or B conversion.

Caution:

- Do not select the same channel in Sequence A and B. Do not select the same trigger source for sequences A and B.

16.3.7 Analog watchdog function

The analog watchdog function is to compare the conversion results at the end of the A/D conversion of a channel, as shown in the figure below, generating a channel comparison interrupt and event ADC_CHCMP if the conversion result is within the protected area. and generating a sequence comparison interrupt and event ADC_SEQCMP at the end of the entire sequence scan, based on the results of each channel comparison. The comparison can be done for any single or multiple channels. Multiple channel comparisons are valid, and a sequence comparison interrupt and event is generated if any of the channels agree.

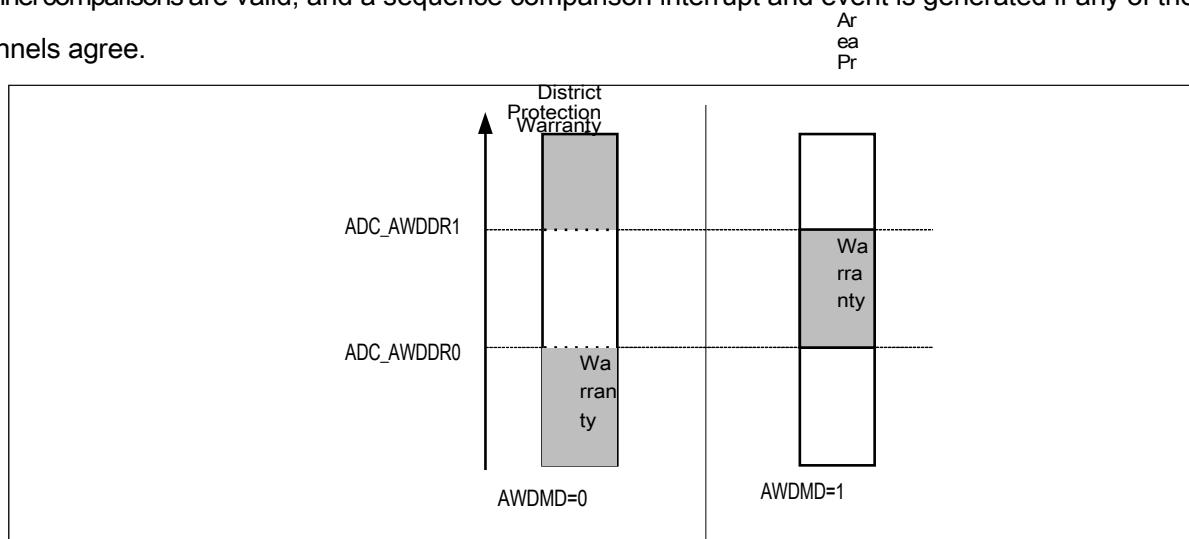


Figure 16-7 Simulated watchdog protection area (comparison condition)

Software flow using the analog watchdog function:

1. Set threshold registers AD_AWDDR0, ADC_AWDDR1
2. Set the compare channel register ADC_AWDCHSR to select any single or multiple channels to be compared
3. Set ADC_AWDCR_AWDMD to select the comparison condition
4. Set ADC_AWDCR.AWDSS[1:0] to select sequence comparison interrupt and event output, and set ADC_AWDCR.AWDIEN interrupt permit bits.
5. Set ADC_AWDCR_AWDEN to allow analog watchdog function
6. According to the previous section, set the scan mode and start AD for conversion.
7. In the ADC_CHCMP/ADC_SEQCMP interrupt or after the A/D conversion, the comparison status register ADC_AWDSR is queried and the comparison result is

processed accordingly.

16.3.8 Sampling time and conversion time of analog inputs

In single scan mode, the A/D conversion can be set by software, internally triggered IN_TRGx0,1 and externally triggered ADTRGx start mode. The ADC module starts sampling and converting the analog channels only after the scan conversion delay time t_D . After all conversions are completed, the ADC module enters the standby state after the end-of-conversion delay time t_E , and a scan is finally completed. The continuous scan mode is similar to the single scan except that there is no t_D time for the second and subsequent starts of the sequence.

The conversion time for a single channel $t_{CONV} = t_{SPL} + t_{CMP}$. Where t_{SPL} indicates the sampling time of the analog input, the number of sampling periods can be adjusted according to the input impedance setting register ADC_SSTRx. t_{CMP} indicates the comparison time one by one, 13 ADCLKs for 12-bit precision, 11 ADCLKs for 10-bit precision, and 9 ADCLKs for 8-bit precision.

The conversion time for one scan $t_{SCAN} = t_D + \sum t_{CONV} + t_E$. Where $\sum t_{CONV}$ represents the sum of the conversion times for all scan channels, and since the sampling time t_{SPL} can be set independently, the conversion time t_{CONV} can be different for each channel.

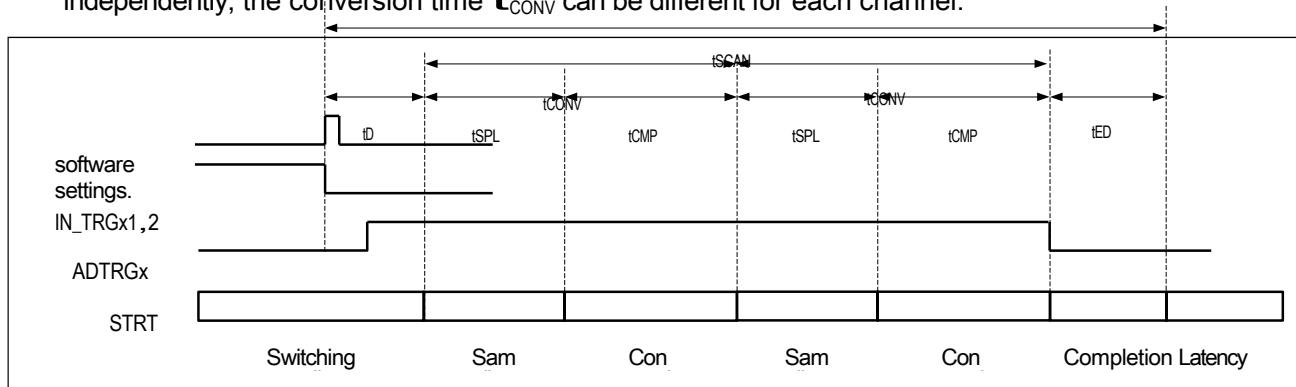


Figure 16-8 A/D Conversion Time

Table 16-4 AD Conversion Time

Marker	Description	Conditions			
		Synchronized peripheral triggering	Asynchronous peripheral trigger *Note	External pin triggering	Software Trigger
t_D	Scan start	ADC idle, start conversion	1 PCLK4 + 4 ADCLK	3 PCLK4 + 4 ADCLK + 1 PCLK4_SYNC	3 PCLK4 + 4 ADCLK
	processing time	Sequence A conversion in being Interrupt to	2 PCLK4 + 6 ADCLK	4 PCLK4 + 6 ADCLK + 1 PCLK4_SYNC	4 PCLK4 + 6 ADCLK

		initiate sequence B conversion						
t _{CONV}	t _{SPL}	Sampling time		ADSSTRx.SST[7:0] × ADCLK				
		t _{CMP}	one	12-bit resolution				
			after	10-bit resolution				
			anoth er Con versi on time	8-bit resolution				
t _{ED}		Scan completion processing time		1 PCLK4 + 3 ADCLK				
t _{TD}		Minimum continuous trigger time interval		$\Sigma t_{CONV} + 2 \text{ PCLK4} + 5 \text{ ADCLK}$				

Caution:

-Asynchronous peripheral trigger refers to the case when the ADC module selects the PLL clock action asynchronous with the system clock. PCLK4_SYNC means the original synchronous clock of ADC module, PCLK4 and ADCLK are the same, both are asynchronous PLL clocks.

16.3.9 A/D data register auto-clear function

When ADC_CR0.CLREN is "1", the A/D conversion data register ADC_DR will be cleared to "0x0000" automatically after it is read by CPU or DMA.

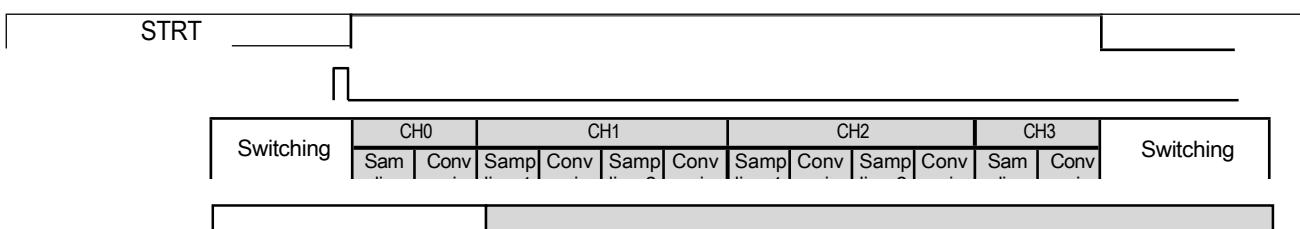
Use this function to detect if the data register ADC_DR is updated. Examples are given below.

- When ADC_CR0.CLREN is "0" and the auto clear function is disabled, the A/D conversion result (0x0222) is not updated to the data register ADC_DR for some reason, and the ADC_DR register continues to hold the previous conversion value (0x0111). The non-updated (0x0111) will be read in the A/D conversion completion interrupt processing. In order to check whether the A/D conversion value is updated or not, it is necessary to store the previous conversion value into RAM additionally and determine it by comparing the conversion result.
- If ADC_CR0.CLREN is "1", the ADC_DR register will be automatically cleared to "0x0000" after the previous conversion result (0x0111) is read by CPU or DMA if the auto clear function is allowed. After the A/D conversion, if the conversion result is not correctly transferred to ADC_DR register, ADC_DR register will keep "0x0000", then, if "0x0000" is read in the interrupt processing, it will be easy to judge the ADC_DR register. It will be easy to determine whether the A/D conversion data is stored correctly.

16.3.10 Average calculation function for conversion data

The A/D conversion averaging function is a function that performs 2, 4, 8, 16, 32, 64, 128, or 256 consecutive conversions of the same channel, averages the conversion results, and saves them in the data register. The averaging function removes certain noise components to make the conversion results more accurate.

Register ADC_CR0.AVCNT[2:0] sets the number of consecutive conversions, and register ADC_AVCHSEL selects any one or more channels to be averaged.



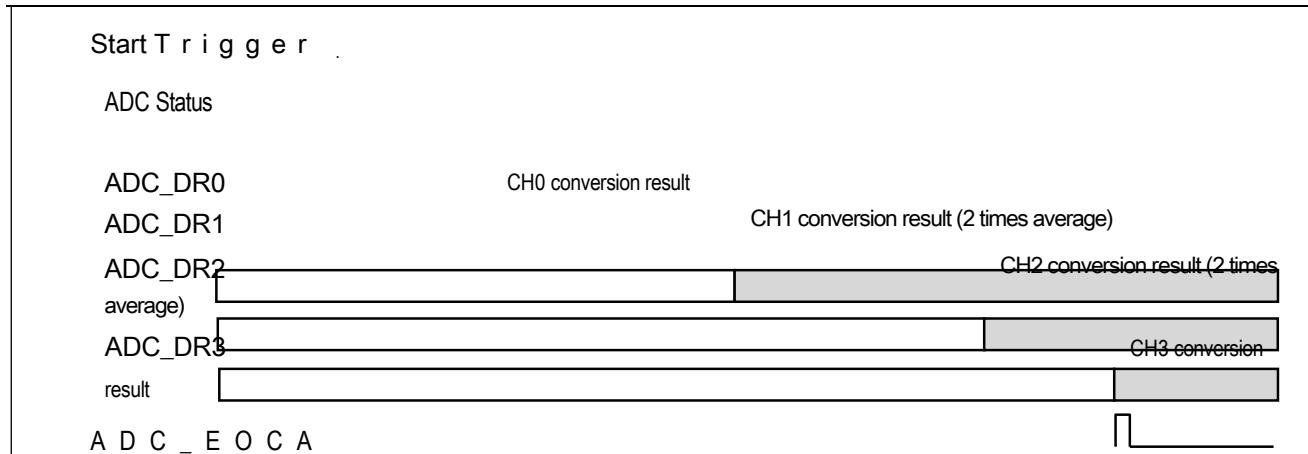


Figure 16-9 Switching action when the average function is active

In Figure 16-9, sequence A single scan mode is selected to convert the 4 channels CH0~3, where CH1 and 2 are set to 2 times averaging mode. During the scanning process, CH1 and 2 will perform two consecutive conversions and save the averaged results to the data registers ADC_DR1 and 2 of the corresponding channels.

16.3.11 Programmable Gain Amplifier PGA

搭载可编程增益放大器 PGA 时，可以设置寄存器 ADC_PGAINSR 选择 PGA 的输入源，设置寄存器 ADC_PGACR，使 PGA 电路有效，设置寄存器 ADC_PGAGSR 选择增益倍数，增益范围 x2~x32 可选择。此时，模拟输入先经过 PGA 电路进行放大，然后再输入到 ADC 模块进行转换。

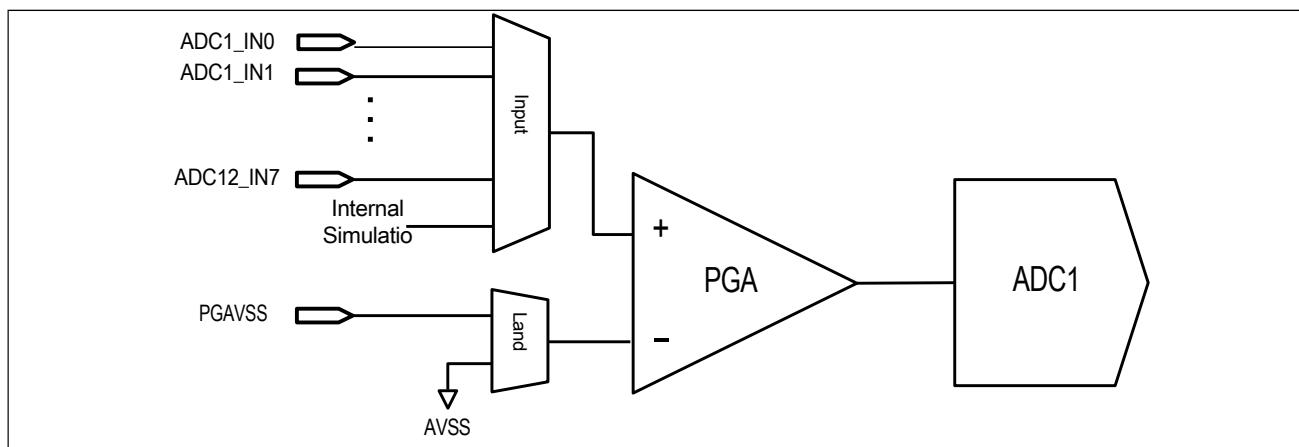


Figure 16-10 PGA working diagram

For example, if you need to convert ADC1_IN2 after amplification, set the channel selection register ADC_CHSELRA0 to 0x4 and select CH2 (ADC1_IN2), set the PGA input source register ADC_PGAINSR to 0x4 and select ADC1_IN2, set the PGA validity and amplification and then start the conversion. Similarly, if you need to amplify the internal analog channel, set ADC_CHSELRA1 to 0x1, select CH16, set ADC_PGAINSR to 0x100, and then start the conversion.

Caution:

- When the internal analog channel is selected for the PGA input source, the analog is fixed to 8bitDAC_1 output, independent of the CMP_RVADC setting.
- Only ADC1 supports PGA.

16.3.12 Multi ADC co-working mode

On chips with two or three ADC modules, ADC co-working mode can be used.

In ADC co-working mode, the conversion of ADC2 and ADC3 is synchronized by the trigger signal of ADC1. That is, the sequence A trigger source select register ADC_TRGSR.TRGSEL[2:0] setting of ADC2 and ADC3 is invalid. All ADC modules are triggered by the trigger source selected by the Sequence A Trigger Source Select Register of ADC1. Writing 1 to ADC_STR.START register in

this mode will not start the conversion, i.e. software start is invalid.

When using the co-working mode, disable the sequence B action to avoid disrupting the synchronization.

ADC1 and ADC2 can be set to work together, or ADC1, ADC2, and ADC3 can work together.

Depending on the product specifications, ADC3 may not be available.

The ADC can be configured to work in the following four co-operative modes:

- Single parallel trigger mode
- Single delayed trigger mode
- Cyclic parallel trigger mode
- Or cyclic delayed trigger mode

Single parallel trigger mode

The sequence A trigger condition of ADC1 triggers all ADC modules in co-working mode at the same time, and only once.

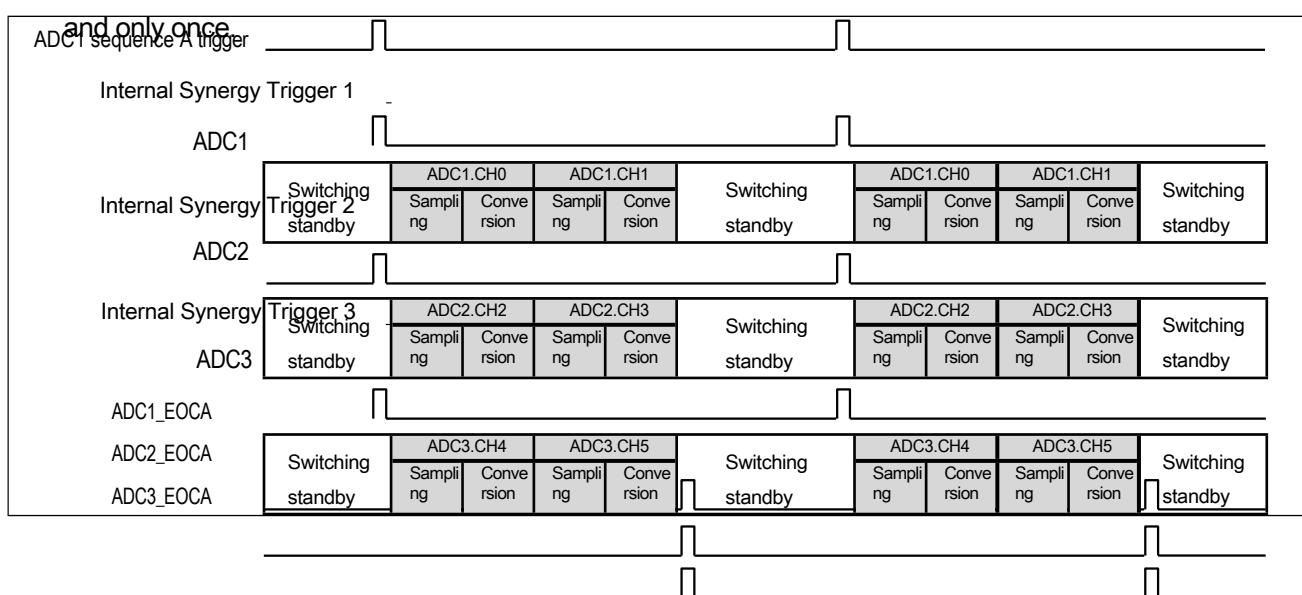


Figure 16-11 Single Parallel Trigger Mode (Triple ADC)

Cautio

n:

- Prohibits multiple ADCs from converting the same analog input at the same time, and only one ADC module can be sampled by an analog channel at the same time, same below.

The software setup process for this mode is as follows:

1. Co-working permission register ADC_SYNCCR.SYNCEN Write 0 to confirm that co-working is invalid.
2. Setting up the ADC1 module
 - a) Verify that ADC1_STR STRT is 0 and ADC1 is in conversion standby.
 - b) Set control register ADC1_CR0.MS[1:0] to 00b: Sequence A single scan mode, or 01b: Sequence A continuous scan mode
 - c) Set Sequence A Channel Select Register ADC1_CHSELRA

- d) Set the sample time register ADC1_SSTR
 - e) Set the sequence A trigger source selection register ADC1_TRGSR
3. Setting up the ADC2 module

- Verify that ADC2_STR.SRT is 0 and ADC1 is in conversion standby.
- Set control register ADC2_CR0.MS[1:0], channel selection register ADC2_CHSELRA, and channel sampling time register ADC2_SSTR.

Caution:

- To ensure that ADC2 and ADC1 work synchronously, the above registers are set to the same values as ADC1 registers as much as possible. The specific channels do not need to be the same, as long as the number of channels and the sampling time of the corresponding channels are the same.

4. Setting up the ADC3 module (when three ADCs are working together)

- Verify that ADC3_STR.SRT is 0 and ADC2 is in conversion standby.
- Set control register ADC3_CR0.MS[1:0], channel selection register ADC3_CHSELRA, and channel sampling time register ADC3_SSTR.

Caution:

- Same as ADC2, in order to ensure the synchronous operation of ADC3 and ADC1, the above registers should be set to the same value as ADC1 as much as possible.

5. Set the co-mode control register ADC_SYNCCR.SYNCMD[2:0] write 010b: ADC1, ADC2 two ADCs work together. Or write 011b: ADC1, ADC2, ADC3 three ADCs work together.

6. Synergy permission register ADC_SYNCCR.SYNCEN Write 1, synergy is valid.

7. Wait for ADC1 sequence A trigger source input and process the results after ADC1, ADC2, ADC3 have completed conversion.

Single delayed trigger mode

The sequence A trigger condition of ADC1 triggers ADC1, then after a set delay it triggers ADC2 to start the conversion, then after a set delay it triggers ADC3 to start the conversion, only once for each

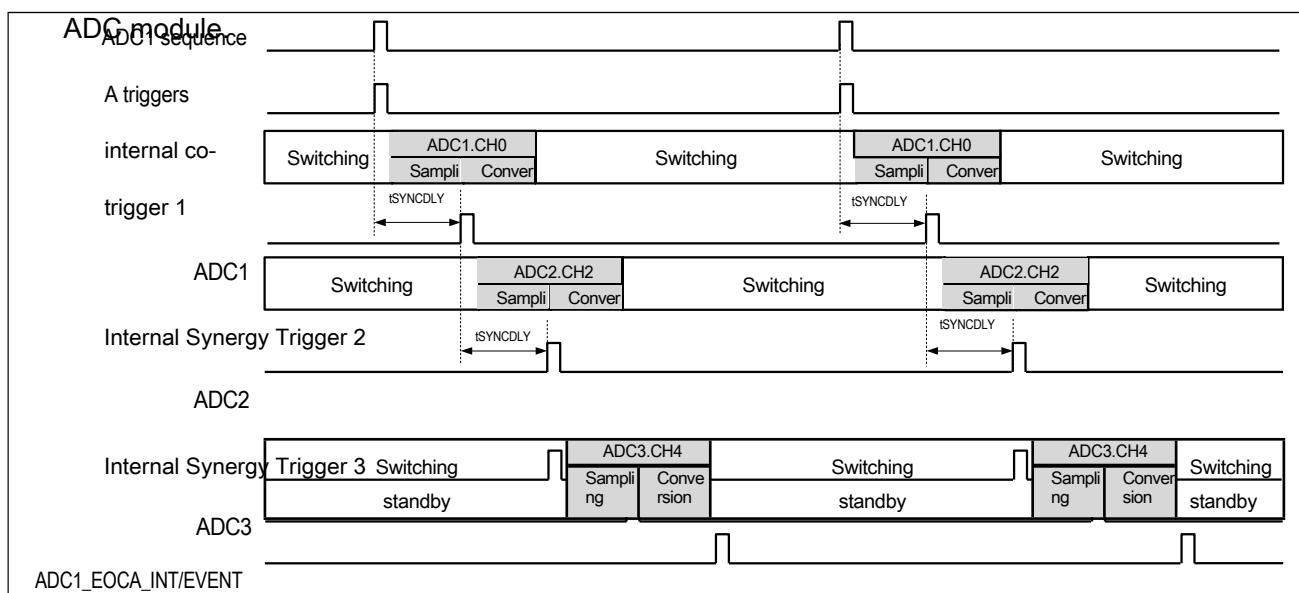


Figure 16-12 Single Delayed Trigger Mode (Triple ADC)

Caution:

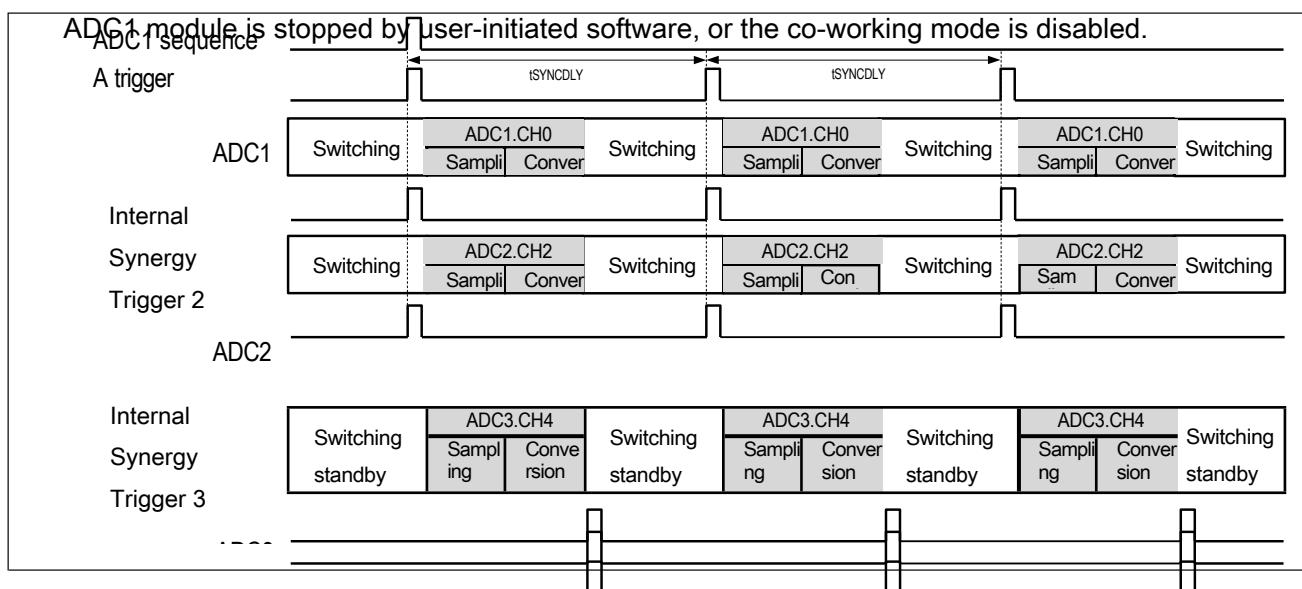
- After the first input of ADC1 sequence A trigger and before the ADC3 co-trigger occurs, the re-entry of ADC1 sequence A trigger will be ignored.
- If each ADC unit converts the same analog channel, the sampling times need to be staggered, i.e., the delay time $t_{SYNCDLY}$ and the channel sampling time t_{SPL} need to meet: $t_{SYNCDLY} > t_{SPL}$.

The software setup process for this mode is as follows:

1. Co-working permission register ADC_SYNCCR.SYNCEN Write 0 to confirm that co-working is invalid.
2. Setting up ADC1, ADC2, ADC3 modules (reference single parallel mode)
3. Set the Synergy Mode Control Register ADC_SYNCCR.SYNCDLY[7:0] to set the start-up delay of both ADCs.
4. Set the co-mode control register ADC_SYNCCR.SYNCMD[2:0] write 000b: ADC1, ADC2 two ADCs work together. Or write 001b: ADC1, ADC2, ADC3 three ADCs work together.
5. Synergy permission register ADC_SYNCCR.SYNCEN Write 1, synergy is valid.
6. Wait for ADC1 sequence A trigger source input and process the results after ADC1, ADC2, ADC3 have completed conversion.

Cyclic parallel trigger mode

The sequence A trigger condition of ADC1 triggers all ADC modules in co-working mode at the same time, and then triggers all ADC modules again at the same time after each specified delay. Until the



n:

Caution

Fig u**re 16-13 Cyclic Parallel Trigger Mode (Triple ADC)**

- The delay time $t_{SYNCDLY}$ and the time of one scan transition t_{SCAN} must meet: $t_{SYNCDLY} > t_{SCAN}$. The software setting process for this mode is as follows:
 1. Co-working permission register ADC_SYNCCR.SYNCEN Write 0 to confirm that co-working is invalid.
 2. Set the ADC1, ADC2, ADC3 modules to reference single parallel mode. ADC_CR0_MS[1:0] is set to 00b:

Sequence A Single Scan Mode

3. Set the Synergy Mode Control Register ADC_SYNCCR.SYNCDLY[7:0] to set the delay of each parallel trigger.
4. Set the co-mode control register ADC_SYNCCR.SYNCMD[2:0] write 110b: ADC1, ADC2 two ADCs work together. Or write 111b: ADC1, ADC2, ADC3 three ADCs work together.
5. Synergy permission register ADC_SYNCCR.SYNCEN Write 1, synergy is valid.
6. Wait for ADC1 sequence A trigger source input and process the results after ADC1, ADC2, ADC3 have completed conversion.

Cyclic delayed trigger mode

After ADC1 is triggered by the sequence A trigger condition, ADC2, ADC3, ADC1, ADC2... are triggered in a continuous loop after each set delay until the ADC1 module is stopped by user-initiated software or the co-working mode is disabled. until the ADC1 module is stopped by user-initiated software, or the co-working mode is disabled.

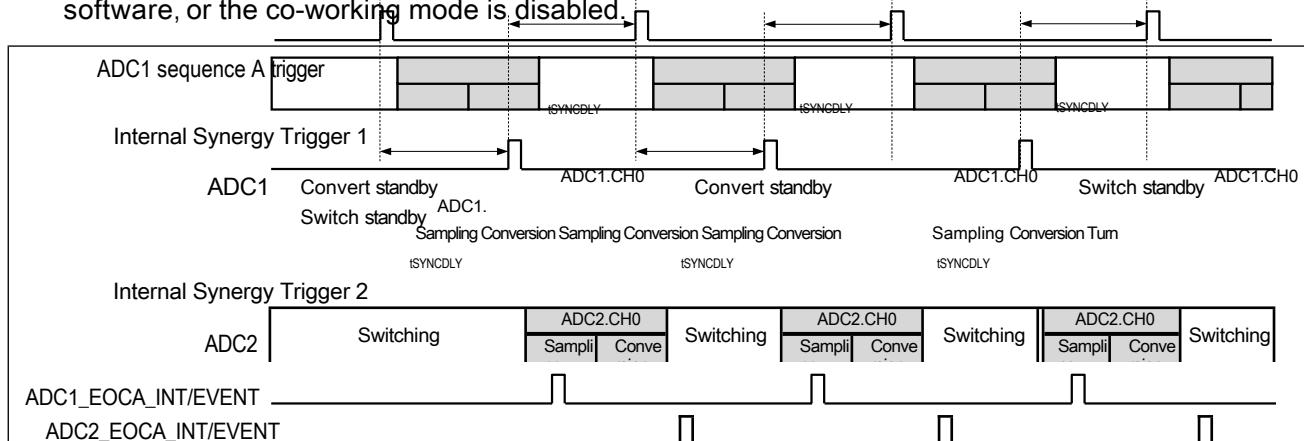
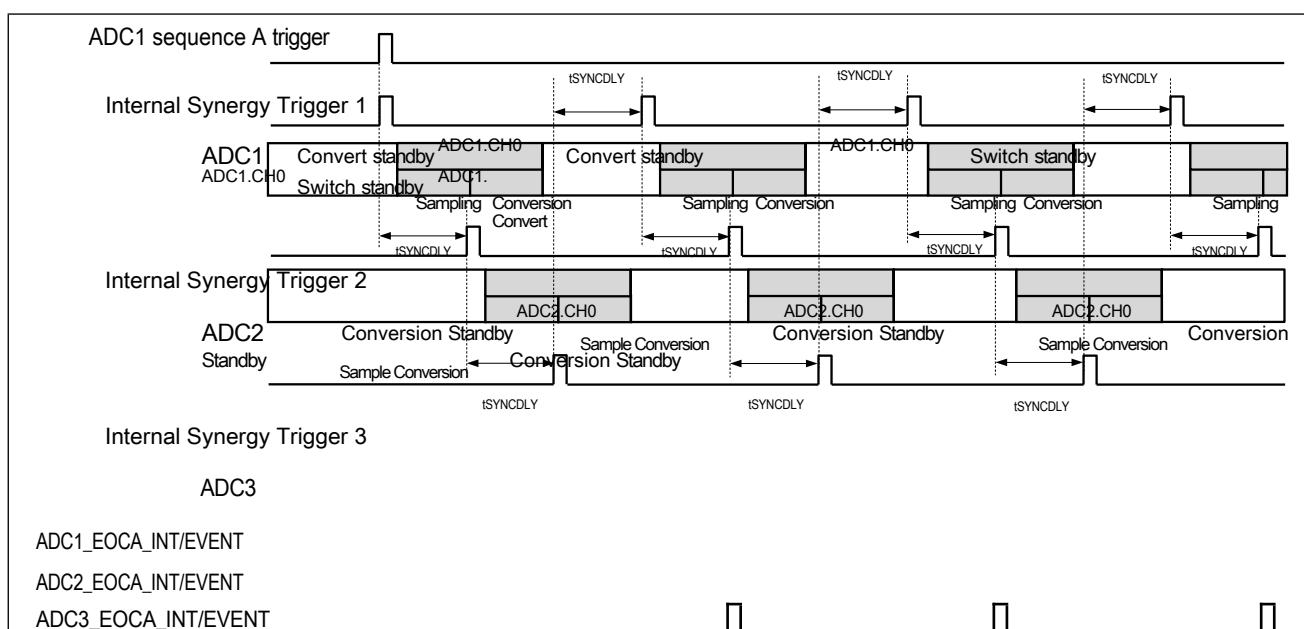


Figure 16-14 Cyclic delay trigger mode (two ADCs)



**Figure 16-15 Cyclic Delay Trigger Mode (Triple ADC)**

Cautio

n:

-When two ADCs work together, the delay time t and the time of one scan conversion t need to be satisfied: $t > t /2$.

For two ADCs working together, the delay time $t_{SYNCDLY}$ and the time of one scan conversion t_{SCAN} should be satisfied $t_{SYNCDLY} > t_{SCAN} /2$. For three ADCs working together: $t_{SYNCDLY} > t_{SCAN} /3$. Also, if ADC1, ADC2 and ADC3 are converting the same analog channel, the sampling time should be staggered, i.e. $t_{SYNCDLY} > t_{SPL}$.

The software setup process for this mode is as follows:

1. Co-working permission register ADC_SYNCCR.SYNCEN Write 0 to confirm that co-working is invalid.
2. Set up ADC1, ADC2, ADC3 modules with reference to cyclic parallel trigger mode.
3. Set the Synergy Mode Control Register ADC_SYNCCR.SYNCDLY[7:0] to set the delay of each trigger.
4. Set the co-mode control register ADC_SYNCCR.SYNCMD[2:0] write 100b: ADC1, ADC2 two ADCs work together. Or write 101b: ADC1, ADC2, ADC3 three ADCs work together.
5. Synergy permission register ADC_SYNCCR.SYNCEN Write 1, synergy is valid.
6. Wait for ADC1 sequence A trigger source input and process the results after ADC1, ADC2, ADC3 have completed conversion.

16.3.13 Interrupt and event signal output

The ADC module can generate the following four event outputs, each of which outputs an interrupt request at the same time if the corresponding interrupt permit register is set to valid.

1. Sequence A End of scan ADC_EOCA, corresponding to interrupt permit register ADC_ICR.EOCAIEN
2. Sequence B End of scan ADC_EOCB, corresponding to interrupt permit register ADC_ICR.EOCBIEN
3. Channel comparison ADC_CHCMP, corresponding to interrupt license register ADC_AWDCR.AWDIEN
4. Sequence comparison ADC_SEQCMP, corresponding to interrupt license register ADC_AWDCR.AWDIEN

The above four event outputs enable other on-chip peripheral modules, including the initiation of DMA transfers. The DMA transfer allows continuous reading of A/D conversion results without software intervention and is implemented entirely in hardware, reducing the load on the CPU. see the DMA

description chapter for DMA settings. The event signal output is independent of the control of the interrupt enable bit and its output whenever the condition occurs.

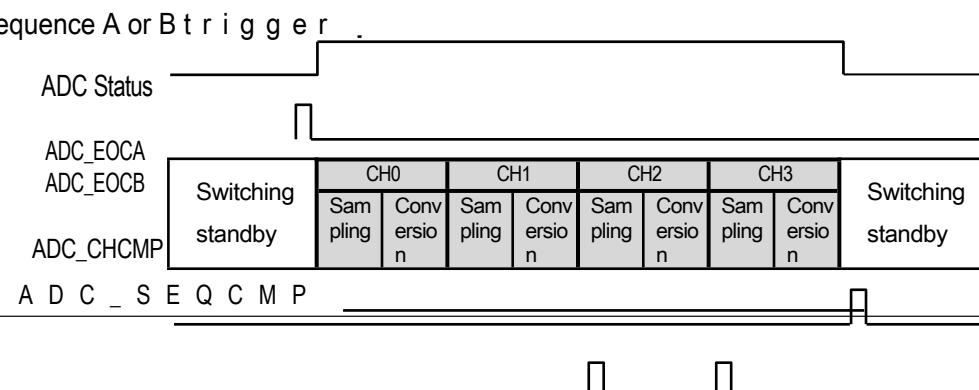


Figure 16-16 ADC Interrupt and Event Output Timing

In the above figure, the comparison conditions are met for the CH1, CH2 watchdog. The channel compare event occurs at the end of each channel transition, and the sequence compare event occurs at the end of the sequence scan, one cycle after the sequence scan event ADC_EOCA/B.

16.4 Register Description

Register List

Unit 1 BASE_ADDR: 0x4004_0000

Unit 2 BASE_ADDR: 0x4004_0400

Table 16-5 ADC Register List 1/2

Register Name	Sym bols	Offset Address	Bit width	Reset value
A/D start register	ADC_STR	0x00	8	0x00
A/D control register 0	ADC_CR0	0x02	16	0x0000
A/D control register 1	ADC_CR1	0x04	16	0x0000
A/D conversion start trigger register	ADC_TRGSR	0xA	16	0x0000
A/D channel selection register A0	ADC_CHSELRA0	0xC	16	0x0000
A/D channel selection register A1	ADC_CHSELRA1	0xE	16	0x0000
A/D channel selection register B0	ADC_CHSELRB0	0x10	16	0x0000
A/D channel selection register B1	ADC_CHSELRB1	0x12	16	0x0000
A/D average channel selection register 0	ADC_AVCHSELR0	0x14	16	0x0000
A/D average channel selection register 1	ADC_AVCHSELR1	0x16	16	0x0000
A/D Sample Period Register	ADC_SSTRx	0x20+x	8	0xB
	ADC_SSTRL	0x30	8	0xB
A/D channel mapping control register 0	CHMUXR0	0x38	16	0x3210
A/D channel mapping control register 1	CHMUXR1	0x3A	16	0x7654
A/D channel mapping control register 2	CHMUXR2	0x3C	16	0xBA98
A/D channel mapping control register 3	CHMUXR3	0x3E	16	0xFEDC
A/D interrupt status register	ADC_ISR	0x46	8	0x00
A/D interrupt license register	ADC_ICR	0x47	8	0x03
A/D co-mode control register	ADC_SYNCCR	0x4C	16	0x0C00
A/D data register	ADC_DRx	0x50+2*x	16	0x0000
Analog watchdog control register	ADC_AWDCR	0xA0	16	0x0000
Analog Watchdog Threshold Register	ADC_AWDDR0	0xA4	16	0x0000
	ADC_AWDDR1	0xA6	16	0x0000
Analog Watchdog Compare Channel Select Register	ADC_AWDCHSR0	0xAC	16	0x0000
Analog watchdog compare channel selection register 1	ADC_AWDCHSR1	0xAE	16	0x0000
Analog watchdog status register	ADC_AWDSR0	0xB0	16	0x0000
Analog watchdog status register 1	ADC_AWDSR1	0xB2	16	0x0000
A/D programmable gain amplifier control register	ADC_PGACR	0xC0	16	0x0000
A/D programmable gain multiplier register	ADC_PGAGSR	0xC2	16	0x0000
A/D programmable gain amplifier input selection	ADC_PGAISNR0	0xCC	16	0x0000

register

Register Name	Sym bols	Offset Address	Bit width	Reset value
A/D programmable gain amplifier input selection register 1	ADC_PGAINS1	0xCE	16	0x0000

16.4.1 A/D start register ADC_STR

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	STRT

position	Marker	Place Name	Function	R/W
b7-b1	-	-	0 on read, 0 on write	R/W
b0	STRT	AD conversion starts	0: Stop conversion 1: Start conversion set "1" condition: (1) Software Setup (2) The selected trigger condition occurs (3) Clear "0" condition in A/D conversion: (1) Software Clear "0" (2) Automatic clearing of "0" at the end of the conversion Note: <ul style="list-style-type: none"> - Writing 1 when STRT is 0 (ADC is idle) generates a software trigger to start sequence A - Write 1 is invalid when STRT is 1 (in ADC action). - Writing 0 when STRT is 1 means force stop AD conversion. If ADC_TRGSR is set to a value other than 0x0 and you do not want the ADC to start again, first set ADC_TRGSR to 0, then write 0 to STRT. - Write 0 is invalid when STRT is 0. 	R/W

16.4.2 A/D control register 0 ADC_CR0

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
-	-	-	-	-	AVCNT[2:0]		
b7	b6	b5	b4	b3	b2	b1	b0
DFMT	CLREN	ACCSEL[1:0]		-	-	MS[1:0]	

position	Marker	Place Name	Function	R/W
b15-	-	-	0 on read, 0 on write	R/W
b11				
b10-b8	AVCNT[2:0]	Number of times to choose	0 0 0: 2 consecutive conversions average 0 0 1: Average of 4 consecutive conversions 0 1 0: 8 consecutive conversions average 0 1 1: Average of 16 consecutive conversions 1 0 0: 32 consecutive conversions average 1 0 1: Average of 64 consecutive conversions 1 1 0: 128 consecutive conversions average 1 1 1: 256 consecutive conversions averaged	R/W
b7	DFMT	Data Format	0: Conversion data right-aligned 1: Convert data to left-aligned	R/W
b6	CLREN	Automatic clearing of data registers	0: Automatically clear the ban 1: Automatic clearance of permits Note: After CLREN bit is set, the register ADC_DRx will be cleared automatically after read by CPU, DMA, etc. The auto-clear function is mainly used to detect whether the data register is updated or not.	R/W
b5-b4	ACCSEL[1:0]	Resolution Selection	0 0: 12-bit resolution 0 1: 10-bit resolution 1 0: 8-bit resolution 1 1: Set the ban	R/W
b3-b2	-	-	0 on read, 0 on write	R/W
b1-b0	MS[1:0]	Mode Selection	0 0: Sequence A single scan mode, sequence B invalid 0 1: Sequence A continuous scan mode, sequence B invalid 1 0: Serial A single scan mode, Serial B single scan mode 1 1: Sequence A continuous scan mode, sequence B single scan mode	R/W

Caution:

-STRT. Please set this register when ADC_STR.STRT is "0".

16.4.3 A/D control register 1 ADC_CR1

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
-	-	-	-	-	-	-	-
b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	RSCHSEL	-	-

position	Marker	Place Name	Function	R/W
b15-b3	-	-	0 on read, 0 on write	R/W
b2	RSCHSEL	Sequence A restart channel selection	0: After being interrupted by sequence B, sequence A restarts to continue scanning from the interrupted channel 1: After being interrupted by sequence B, sequence A restarts with a new scan from the first channel	R/W
b1-b0	-	-	0 on read, 0 on write	R/W

Caution:

-STRT. Please set this register when ADC_STR.STRT is "0".

16.4.4 A/D conversion start trigger register ADC_TRGSR

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
TRGENB	-	-	-	-	TRGSELB [2]	TRGSELB [1]	TRGSELB[0]
b7	b6	b5	b4	b3	b2	b1	b0
TRGENA	-	-	-	-	TRGSEL A [2]	TRGSEL A [1]	TRGSEL A[0]

position	Marker	Place Name	Function	R/W
b15	TRGENB	Sequence B trigger enable	0: Sequence B on-chip or external pin trigger disable 1: Sequence B on-chip or external pin triggering permission Note: Selecting external pin trigger is valid. If ADTRGx changes from "High" to "Low" and a falling edge is detected, the scan transition starts, please keep "Low" for 1.5*PCLK4 cycles. or more.	R/W
b14-b11	-	-	0 on read, 0 on write	R/W
b10-b8	TRGSELB[2:0]	Sequence B trigger condition selection	In the valid mode of sequence B (ADC_CR0.MS[1]=1), as the trigger condition of sequence B 000b: ADTRGx 001b: IN_TRGx0 010b: IN_TRGx1 011b: IN_TRGx0 + IN_TRGx1 Other: not selected x=1, 2 Note: Valid only in Sequence B valid mode. Other mode settings are not valid. The interval between the two triggers must be greater than or equal to the scan period tSCAN, if less than then the trigger is not	R/W
			Effect.	
b7	TRGENA	Sequence A trigger enable	0: Serial A on-chip or external pin trigger disable 1: Sequence A on-chip or external pin trigger permit Note: Selecting the external pin trigger is valid. If ADTRGx changes from "High" to "Low", the check If the falling edge is measured, the scan transition starts, please keep "Low" for more than 1.5*PCLK4 cycles.	R/W
b6-b3	-	-	0 on read, 0 on write	R/W
b2-b0	TRGSEL A[2:0]	Sequence A trigger condition selection	The trigger condition for sequence A. 000b: ADTRGx 001b: IN_TRGx0 010b: IN_TRGx1 011b: IN_TRGx0 + IN_TRGx1 Other: not selected n=1, 2 Caution: ADC_STR.START write 1 software trigger, ignore TRGENA, or TRGSEL A[2:0] setting, and start A/D conversion directly. The interval between two triggers must be greater than or equal to the scan	R/W

period tSCAN, if less than then the trigger is invalid.

Caution:

-STRT. STRT is set when ADC_STR.STRT is "0".

16.4.5 A/D channel selection register A ADC_CHSELRA0

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
CHSEL A [15:8]							
b7	b6	b5	b4	b3	b2	b1	b0
CHSEL A[7:0]							

position	Marker	Place Name	Function	R/W
b15-b0	CHSEL A [15:0]	Conversion channel selection	The channel selection of sequence A, each bit represents a channel, and CHSEL A[x] represents channel CHx, which can be selected in any combination. 0: No corresponding channel selected 1: Select the corresponding channel Note: Please do not select the same channel in Sequence A and Sequence B. The corresponding bit of the non-existent channel is reserved bit, which is 0 when reading out and 0 when writing in.	R/W

Caution:

-STRT. Please set this register when ADC_STR.STRT is "0".

16.4.6 A/D channel selection register A 1 ADC_CHSELRA1

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
CHSEL A [31:24]							
b7	b6	b5	b4	b3	b2	b1	b0
CHSEL A [23:16]							

position	Marker	Place Name	Function	R/W
b15-b0	CHSEL A [31:16]	Conversion channel selection	The channel selection of sequence A, each bit represents a channel, and CHSEL A[x] represents channel CHx, which can be selected in any combination. 0: No corresponding channel selected 1: Select the corresponding channel Note: Please do not select the same channel in Sequence A and Sequence B. The corresponding bit of the non-existent channel is reserved bit, which is 0 when reading out and 0 when writing in.	R/W

Caution:

-STRT. Please set this register when ADC_STR.STRT is "0".

16.4.7 A/D Channel Select Register B ADC_CHSELRB0

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
CHSELB [15:8]							
b7	b6	b5	b4	b3	b2	b1	b0
CHSELB[7:0]							

position	Marker	Place Name	Function	R/W
b15-b0	CHSELB[15:0]	Conversion channel selection	Channel selection for sequence B. Each bit represents a channel, and CHSELB[x] represents channel CHx, which can be selected in any combination. Only valid in dual sequence scan mode. 0: No corresponding channel selected 1: Select the corresponding channel Note: Please do not select the same channel in Sequence A and Sequence B. The corresponding bit of the non-existent channel is reserved bit, which is 0 when reading out and 0 when writing in.	R/W

Caution:

-STRT. Please set this register when ADC_STR.STRT is "0".

16.4.8 A/D Channel Select Register B1 ADC_CHSELRB1

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
CHSELB [31:24]							
b7	b6	b5	b4	b3	b2	b1	b0
CHSELB [23:16]							

position	Marker	Place Name	Function	R/W
b15-b0	CHSELB [31:16]	Conversion channel selection	Channel selection for sequence B. Each bit represents a channel, and CHSELB[x] represents channel CHx, which can be selected in any combination. Only valid in dual sequence scan mode. 0: No corresponding channel selected 1: Select the corresponding channel Note: Please do not select the same channel in Sequence A and Sequence B. The corresponding bit of the non-existent channel is reserved bit, which is 0 when reading out and 0 when writing in.	R/W

Caution:

-STRT. Please set this register when ADC_STR.STRT is "0".

16.4.9 A/D Average Channel Select Register ADC_AVCHSELR0

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
AVCHSEL[15:8]							
b7	b6	b5	b4	b3	b2	b1	b0
AVCHSEL[7:0]							

position	Marker	Place Name	Function	R/W
b15-b0	AVCHSEL[15: 0]	Average channel selection	<p>Each bit represents a channel, AVCHSEL[x] represents channel CHx, any combination can be selected. 0: No corresponding channel is selected</p> <p>1: Select the corresponding channel</p> <p>The corresponding bit of the non-existent channel is the reserved bit, which is 0 when reading out and 0 when writing in.</p> <p>Note: When AVCHSEL is selected at the same time as the corresponding channel of ADC_CHSELRA or ADC_CHSELB, the channel will perform the set number of A/D conversions continuously during scanning, and the conversion result will be averaged and updated into the data register. If the corresponding channel AVCHSEL is not set, the channel will perform a normal conversion.</p>	R/W

Caution:

-STRT. Please set this register when ADC_STR.STRT is "0".

16.4.10 A/D Average Channel Select Register 1 ADC_AVCHSELR1

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
AVCHSEL[31:24]							
b7	b6	b5	b4	b3	b2	b1	b0
AVCHSEL[23:16]							

position	Marker	Place Name	Function	R/W
b15-b0	ADAVSEL [31:16]	Average channel selection	<p>Each bit represents a channel, AVCHSEL[x] represents channel CHx, any combination can be selected. 0: No corresponding channel is selected</p> <p>1: Select the corresponding channel</p> <p>The corresponding bit of the non-existent channel is the reserved bit, which is 0 when reading out and 0 when writing in.</p> <p>Note: When AVCHSEL is selected at the same time as the corresponding channel of ADC_CHSELRA or ADC_CHSELB, the channel will perform the set number of A/D conversions continuously during scanning, and the conversion result will be averaged and updated into the data register. If the corresponding channel AVCHSEL is not set, the channel will perform a normal conversion.</p>	R/W

Caution:

-STRT. Please set this register when ADC_STRT is "0".

16.4.11 A/D Sample Status Register ADC_SSTR

Reset value: 0x0B

b7	b6	b5	b4	b3	b2	b1	b0
SST[7:0]							

position	Marker	Place Name	Function	R/W
b7-b0	SST[7:0]	Number of sampling cycles	The number of sampling cycles can be set from 5 to 255 cycles. Channels CH0~15 are set by ADC_SSTRx, x=0~15, and other channels are set by ADC_SSTRL. Note: When the ADCLK frequency is 50MHz, one sample period is 20ns, and the initial conversion state has 11 sample periods. When the external input impedance RAIN is too large sampling time is not enough or ADCLK frequency is low, you can set the register to adjust the sampling time. The sampling time should not be less than 5 cycles. $SST \geq (RAIN + RADC) * CADC * In^{(2N+2)} * fADC + 1$ Where: RAIN denotes external input impedance (Ω), RADC denotes internal sampling switching resistance (Ω), CADC denotes internal sampling and hold capacitance (F), N denotes AD resolution (12/10/8) and fADC denotes ADCLK frequency (Hz). Specific Refer to the instructions related to electrical characteristics.	R/W

Caution:

-STRT. Please set this register when ADC_STR.STRT is "0".

16.4.12 A/D Channel Mapping Control Register ADC_CHMUXR

ADC_CHMUXR0 Reset value: 0x3210

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CH03MUX[3:0]				CH02MUX[3:0]				CH01MUX[3:0]				CH00MUX[3:0]			

ADC_CHMUXR1 Reset value: 0x7654

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CH07MUX[3:0]				CH06MUX[3:0]				CH05MUX[3:0]				CH04MUX[3:0]			

ADC_CHMUXR2 reset value: 0xBA98

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CH11MUX[3:0]				CH10MUX[3:0]				CH09MUX[3:0]				CH08MUX[3:0]			

ADC_CHMUXR3 Reset value: 0xFEDC

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CH15MUX[3:0]				CH14MUX[3:0]				CH13MUX[3:0]				CH12MUX[3:0]			

position	Marker	Place Name	Function	R/W
n	ChxMUX[3:0]	Channel x mapping selection x=0~15	The corresponding bits of non-existent channels are read as 0 and written as 0 to the different ADC cells Chx are mapped as follows:	R/W
		Set value	ADC1 Mapped Objects	ADC2 Mapped Objects
		0x0	ADC1_IN0	ADC12_IN4
		0x1	ADC1_IN1	ADC12_IN5
		0x2	ADC1_IN2	ADC12_IN6
		0x3	ADC1_IN3	ADC12_IN7
		0x4	ADC12_IN4	ADC12_IN8
		0x5	ADC12_IN5	ADC12_IN9
		0x6	ADC12_IN6	ADC12_IN10
		0x7	ADC12_IN7	ADC12_IN11
		0x8	ADC12_IN8	Internal analog channels
		0x9	ADC12_IN9	-
		0xa	ADC12_IN10	-
		0xb	ADC12_IN11	-
		0xc	ADC1_IN12	-
		0xd	ADC1_IN13	-
		0xe	ADC1_IN14	-
		0xf	ADC1_IN15	-

Note: Do not set to a non-existent analog input.

Cautio

n:

-STRT. Please set this register when ADC_STRT STRT is "0".

16.4.13 A/D Interrupt Status Register ADC_ISR

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	EOCBF	EOCAF

position	Marker	Place Name	Function	R/W
b7-b2	-	-	0 on read, 0 on write	R/W
b1	EOCBF	Sequence B conversion completion flag	Set 1 after all channels selected in sequence B have been scanned When the register is set, please read "1" and write "0" when you need to clear the register.	R/W
b0	EOCAF	Sequence A conversion completion flag	Set 1 after all channels selected in sequence A have been scanned When the register is set, please read "1" and write "0" when you need to clear the register.	R/W

16.4.14 A/D Interrupt Permit Register ADC_ICR

Reset value: 0x03

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	EOCBIEN	EOCAIEN

position	Marker	Place Name	Function	R/W
b7-b2	-	-	0 on read, 0 on write	R/W
b1	EOCBIEN	Sequence B conversion completion interrupt enable	0: Serial B conversion completion interrupt disable 1: Sequence B conversion completion interrupt permit	R/W
b0	EOCAIEN	Sequence A conversion completion interrupt enable	0: Serial A conversion completion interrupt disable 1: Sequence A conversion completion interrupt permit	R/W

16.4.15 A/D Cooperative Mode Control Register ADC_SYNCCR

Reset value: 0x0C00

b15	b14	b13	b12	b11	b10	b9	
b8 SYNCDELAY[7:0]							
b7	b6	b5	b4	b3	b2	b1	b0
-	SYNCMD[2]	SYNCMD[1]	SYNCMD[0]	-	-	-	SYNCEN

position	Marker	Place Name	Function	R/W	
b15-b8	SYNCDELAY[7:0]	Synchronization delay time	The start-up delay time $t_{SYNCDLY}$ for both ADCs when in delayed trigger mode. n delay time 0x1 means $t_{SYNCDLY} = 1 \times PCLK4$, 0xff means $t_{SYNCDLY} = 255 \times PCLK4$ Note: Set this register when SYNCEN is "0". Please do not write 0x00. According to the sampling time and conversion time of each ADC, set a reasonable delay time to avoid the increase of error caused by multiple ADCs in the sampling state at the same time, and avoid the trigger occurring again before the end of ADC conversion, resulting in synchronization failure. The recommended settings are as follows: Single delayed trigger mode: $t_{SYNCDLY} > t_{SPL}$ Two ADC cyclic delay trigger mode: $t_{SYNCDLY} > t_{SPL}$, and $t_{SYNCDLY} > t_{SCAN}/2$ Three ADC cyclic delay trigger mode: $t_{SYNCDLY} > t_{SPL}$, and $t_{SYNCDLY} > t_{SCAN}/3$ Single parallel trigger mode: This register setting is invalid. Cyclic parallel trigger mode: $t_{SYNCDLY} > t_{SCAN}$		R/W
b7	-	-	0 on read, 0 on write	R/W	
b6-b4	SYNCMD[2:0]	Sync mode selection	SYNCMD[2] 0: Single trigger 1: Cyclic trigger SYNCMD[1] 0: Delayed trigger mode 1: Parallel trigger mode mode SYNCMD[0] 0: ADC1 and ADC2 work synchronously, ADC3 works independently 1: ADC1, ADC2 and ADC3 work synchronously, if there is no ADC3, this bit is forbidden to set to 1. Note: Set this register when SYNCEN is "0". When using single trigger, set the ADC to be synchronized to "0". Set to Sequence A single scan, or Sequence A continuous scan mode. When using cyclic trigger mode, set the ADC to Sequence A single scan mode.	R/W	
b3-b1	-	-	0 on read, 0 on write	R/W	
b0	SYNCEN	Synchronous mode license	0: Synchronization mode is invalid 1: Synchronous mode effective Note: The Sync mode only supports sequence A. Before SYNCEN write 1, please turn off sequence B of several ADCs involved in synchronization (ADC_CR0.MS[1]=0) and select the same number of channels for sequence A and set the same channel sampling time ADC_SSTRx. to avoid inconsistent scan time t_{SCAN} for each ADC, which causes subsequent synchronization failure.	R/W	

When the software writes 0 to ADC1_STR.STRT to force the conversion to stop,
SYNCEN is automatically cleared to 0.

Caution:

- – This register is only carried in ADC1, there is no such register in ADC2.

16.4.16 A/D Data Register ADC_DR

ADC_DRx (ADC1 x=0~16, ADC2 x=0~8)

Channel x Data Register

The ADC_DR register is a read-only register used to store the A/D conversion data for each channel. The reset value is 0x0000. Depending on the data alignment and conversion resolution, the conversion result data storage method is different.

Data right alignment - 12-bit resolution

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0			
0	0	0	0	AD[11:0]														

Data right alignment - 10 bit resolution

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0					
0	0	0	0	0	0	AD[9:0]														

Data right-aligned - 8-bit resolution

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0							
0	0	0	0	0	0	0	0	AD[7:0]														

Data left alignment - 12-bit resolution

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0			
AD[11:0]															0	0	0	0

Data left alignment - 10-bit resolution

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0			
AD[9:0]															0	0	0	0

Data left-aligned - 8-bit resolution

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0			
AD[7:0]															0	0	0	0

16.4.17 Analog Watchdog Control Register ADC_AWDCR

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
-	-	-	-	-	-	-	AWDIEN
b7	b6	b5	b4	b3	b2	b1	b0
AWDSS [1]	AWDSS[0]	-	AWDMD	-	-	-	AWDEN

position	Marker	Place Name	Function	R/W
b15-b9	-	-	0 on read, 0 on write	R/W
b8	AWDIEN	Watchdog interrupt enable	0: Disable ADC_CHCMP, ADC_SEQCMP interrupts 1: Allow ADC_CHCMP, ADC_SEQCMP interrupts Note: This register does not affect ADC_CHCMP, ADC_SEQCMP event output	R/W
b7-b6 0】	AWDSS[1: 0】	Watchdog Sequence Selection	00: ADC_SEQCMP interrupt/event is output when both sequence A and sequence B scanning is completed 01: ADC_SEQCMP interrupt/event is output when the scan of sequence A is completed, sequence B is not output 10: ADC_SEQCMP interrupt/event is output when the scan of sequence B is completed, sequence A is not output 11: Same as 00 Notes: The channel watchdog interrupt/event ADC_CHCMP is not controlled by this register and is output normally at the end of each channel conversion based on the comparison result. The setting of 1 in the ADC_AWDSR each channel comparison status register is not controlled by this register.	R/W
b5	-	-	0 on read, 0 on write	R/W
b4	AWDMD	Watchdog comparison mode	0: The comparison condition is satisfied when AWDDR0>conversion result, or conversion result>AWDDR1 1: When AWDDR0≤conversion result≤AWDDR1, the comparison condition is satisfied When the comparison condition is satisfied, the ADC_CHCMP event is output, and if the interrupt allows AWDIEN=1 then the interrupt is output at the same time. When the sequence scan is completed, if one or more channels in this sequence meet the comparison condition and AWDSS[1:0] allows this sequence, the ADC_SEQCMP event is output, and if the interrupt allows AWDIEN=1 then the interrupt is output at the same time.	R/W
b3-b1	-	-	0 on read, 0 on write	R/W
b0	AWDEN	Watchdog comparison function enabled	0: Watchdog comparison function is invalid 1: Watchdog comparison function is effective	R/W

16.4.18 Analog Watchdog Threshold Register ADC_AWDDR0, ADC_AWDDR1

Reset value: ADC_AWDDR0=0x0000, ADC_AWDDR1=0xFFFF

b15	b14	b13	b12	b11	b10	b9	b8
AWDDR[15:8]							
b7	b6	b5	b4	b3	b2	b1	b0
AWDDR[7:0]							

position	Marker	Place Name	Function	R/W
b15-b0	ADDR[15:0]	Compare Data	Compare Data	R/W

AWDDR0 sets the low threshold and AWDDR1 sets the high threshold. The data can also be written to registers in the A/D conversion to achieve dynamic threshold comparison function.

AWDDR0 and AWDDR1 have different resolutions (12-bit, 10-bit or 8-bit) depending on the alignment (right- or left-aligned data)

- Data right-aligned-12-bit resolution Low 12-bit [11:0] available
- Data right-aligned-10-bit resolution Lower 10 bits [9:0] available
- Data right-aligned - 8-bit resolution Lower 8 bits [7:0] available
- Data left-justified-12-bit resolution High 12-bit [15:4] available
- Data left-justified-10-bit resolution High 10-bit [15:6] available
- Data left alignment – 8-bit resolution High 8-bit [15:8] available When the multiple averaging function is valid, only the final average is compared.

16.4.19 Analog Watchdog Compare Channel Select Register ADC_AWDCHSR0

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
AWDCH [15:8]							
b7	b6	b5	b4	b3	b2	b1	b0
AWDCH[7:0]							

position	Marker	Place Name	Function	R/W
b15-b0	AWDCH[15:0]	Watchdog comparison function	0: The watchdog comparison function for this channel is not selected 1: The channel watchdog comparison function selection Each bit corresponds to one channel. The corresponding bit for a non-existent channel is the reserved bit, which is 0 for reads and 0 for writes. Note: Only if the corresponding channel is selected in the sequence A or B scan, i.e., the channel selection register ADC_CHSELRA or The corresponding bit of ADC_CHSELRB is valid when it is "1".	R/W

Caution:

-STRT. Please set this register when ADC_STR.STRT is "0" and ADC_AWDCR.AWDEN is "0".

16.4.20 Analog Watchdog Compare Channel Select Register 1 ADC_AWDCHSR1

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
AWDCH [31:24]							
b7	b6	b5	b4	b3	b2	b1	b0
AWDCH [23:16]							

position	Marker	Place Name	Function	R/W
b15-b0	AWDCH [31:16]	Watchdog comparison function	0: The watchdog comparison function for this channel is not selected 1: The channel watchdog comparison function selection Each bit corresponds to one channel. The corresponding bit for a non-existent channel is the reserved bit, which is 0 for reads and 0 for writes. Note: Only if the corresponding channel is selected in the sequence A or B scan, i.e., the channel selection register ADC_CHSELRA or The corresponding bit of ADC_CHSELRB is valid when it is "1".	R/W

Caution:

-STRT. Please set this register when ADC_STR.STRT is "0" and ADC_AWDCR.AWDEN is "0".

16.4.21 Analog Watchdog Status Register ADC_AWDSR0

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
AWDF [15:8]							
b7	b6	b5	b4	b3	b2	b1	b0
AWDF[7:0]							

position	Marker	Place Name	Function	R/W
b15-b0	AWDF [15:0]	Watchdog Compare Status Bits	0: The comparison condition is not valid 1: The comparison condition holds note that Each bit corresponds to one channel. The corresponding bit for a non-existent channel is the reserved bit, which is 0 for reads and 0 for writes. When the register is set, please read "1" and write "0" when you need to clear the register.	R/W

16.4.22 Analog Watchdog Status Register 1 ADC_AWDSR1

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
AWDF [31:24]							
b7	b6	b5	b4	b3	b2	b1	b0
AWDF [23:16]							

position	Marker	Place Name	Function	R/W
b15-b0	AWDF [31:16]	Watchdog Compare Status Bits	0: The comparison condition is not valid 1: The comparison condition holds note that Each bit corresponds to one channel. The corresponding bit for a non-existent channel is the reserved bit, which is 0 for reads and 0 for writes. When the register is set, please read "1" and write "0" when you need to clear the register.	R/W

16.4.23 A/D Programmable Gain Amplifier Control Register ADC_PGACR

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8			
-	-	-	-	-	-	-	-			
b7	b6	b5	b4	b3	b2	b1	b0			
-	-	-	-	PGACTL[3:0]						

position	Marker	Place Name	Function	R/W
b15-b12	-	-	0 on read, 0 on write	R/W
b3~b0	PGACTL[3:0]	Amplifier control	0000: Amplifier invalid 1110: The amplifier is valid, and the signal is amplified by the value set in ADC_PGAGSR.GAIN[3:0] Note: Setting values other than the above is prohibited.	R/W

16.4.24 A/D Programmable Gain Multiplier Register ADC_PGAGSR

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8			
-	-	-	-	-	-	-	-			
b7	b6	b5	b4	b3	b2	b1	b0			
-	-	-	-	GAIN[3:0]						

position	Marker	Place Name	Function	R/W
b15-b4	-	-	0 on read, 0 on write	R/W
b3-b0	GAIN[3:0]	Amplifier gain setting	0 0 0 0: x 2.000 0 0 0 1: x 2.133 0 0 1 0: x 2.286 0 0 1 1: x 2.667 0 1 0 0: x 2.909 0 1 0 1: x 3.2 0 1 1 0: x 3.556 0 1 1 1: x 4.000 1 0 0 0 0: x 4.571 1 0 0 1: x 5.333 1 0 1 0: x 6.4 1 0 1 1: x 8 1 1 0 0: x 10.667 1 1 0 1: x 16 1 1 1 1 0: x 32	R/W

Note: Other values are forbidden to be set.

16.4.25 A/D Programmable Gain Amplifier Input Select Register ADC_PGAINSR0

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
-	-	-	-	-	-	-	PGAINSEL [8]
b7	b6	b5	b4	b3	b2	b1	b0
PGAINSEL [7:0]							

position	Marker	Place Name	Function	R/W
b15-b9	-	-	0 on read, 0 on write	R/W
b8~b0	PGAINSEL[8:0]	Amplifier	0x000: Input not selected	R/W
]	analog input selection	0x001: ADC1_IN0 0x002: ADC1_IN1 0x004: ADC1_IN2 0x008: ADC1_IN3 0x010: ADC12_IN4 0x020: ADC12_IN5 0x040: ADC12_IN6 0x080: ADC12_IN7 0x100: Internal analog channel (8bitDAC_1 output) Other: Forbidden to set	

16.4.26 A/D Programmable Gain Amplifier Input Select Register 1

ADC_PGAINSR1

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
-	-	-	-	-	-	-	-
b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	PGAVSSEN

position	Marker	Place Name	Function	R/W
b15-b1	-	-	0 on read, 0 on write	R/W
b0	PGAVSSEN	Amplifier ground removal control selection	0: Use external port PGAVSS as PGA negative phase input 1: Use the internal analog ground AVSS as the PGA negative phase input	R/W



16.5 Precautions for use

16.5.1 Precautions when reading data registers

A/D Data Register ADC_DR Please access in half-word units. Please do not access the data registers in byte units.

16.5.2 Notes on Scan Completion Interrupt Handling

When two consecutive scan conversions are performed for the same channel, the second conversion data will overwrite the first conversion data if the CPU does not read the first conversion data in time between the first conversion completion interrupt processing and the second conversion completion interrupt processing.

16.5.3 Notes on Module Stop and Low Power Setting

By setting register PWC_FCG3, you can set the ADC module to stop and reduce power consumption, and the initial state of ADC is stop. When you need the A/D module to work, please set the corresponding bit of PWC_FCG3 register to cancel the stop and wait for 1us before starting the A/D conversion.

Before setting the module stop, make sure that the A/D is in conversion stop, i.e.

ADC_STR. Before setting the system to STOP mode, set the ADC to module stop mode.

For details, please refer to the section on low power consumption instructions.

16.5.4 Pin setting for A/D conversion analog channel input

When the chip pin is set to A/D analog channel input, please disable the digital function of the corresponding pin first (PCRxy.DDIS). Refer to the GPIO chapter.

16.5.5 Noise Control

To prevent abnormal voltages such as surges from damaging the analog input pins, it is recommended to use the protection circuitry shown **in the Electrical Characteristics** section of the **datasheet**.

17 Temperature Sensor (OTS)

17.1 Introduction

The On-chip Temperature Sensor (hereafter referred to as OTS) can acquire the temperature inside the chip to support the reliable operation of the system. It can be turned off when not in use by the module stop function to reduce system power consumption.

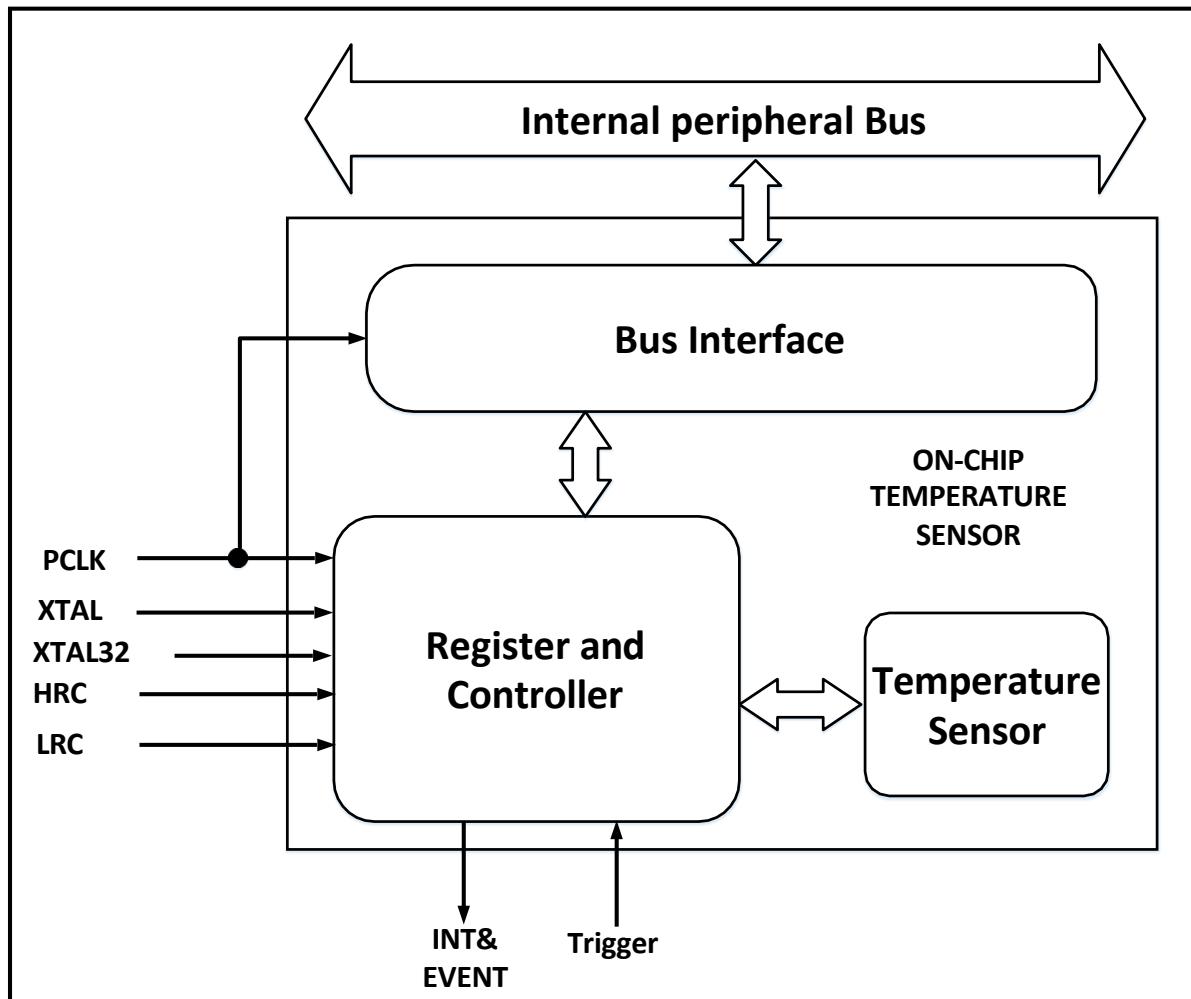


Figure 17-1 OTS Function Block Diagram

17.2 Instructions for use

Before using the OTS to obtain the internal temperature of the chip, turn off the module stop function and start the internal low-speed clock LRC. Also, select to start the internal high-speed clock HRC, the external high-speed clock XTAL and the external low-speed clock XTAL32 depending on the usage.

OTSST is set to "1", and the OTSST bit will be cleared automatically when the temperature measurement is completed. Therefore, after confirming that OTSST is "0", read the temperature parameters in registers OTS_DR1,2 and OTS_ECR, and use the following formula to find the temperature value.

$$T = K \times (1/D1 - 1/D2) \times E_{hrc} + M$$

[Parameter

Description

] T :

Temperature

(°C)

K: temperature slope (determined by calibration experiment)
D1:

temperature parameter 1 (read from register OTS_DR1) D2: temperature

parameter 2 (read from register OTS_DR2)

E_{hrc} : HRC frequency error compensation amount
(read from register OTS_ECR) M: Temperature offset amount (determined by calibration experiment)

[Calibration experiment].

Calibration experiments were performed at

two determined temperatures to

calculate K and M. $K = (T2 - T1) / (A2 - A1)$

$M = T1 - K \times A1 = T2 - K \times A2$

T1: Experimental temperature 1

T2: Experimental temperature 2

$A1(1/D1_{T1} - 1/D2_{T1}) \times E_{hrcT1}$

D1_{T1}, D1 E_{T1}, hrc_{T1} are D1, D2, E_{hrc} measured at temperature T1 respectively;

$A2(1/D1_{T2} - 1/D2_{T2}) \times E_{hrcT2}$

D1_{T2}, D1 E_{T2}, hrc_{T2} are measured at temperature T2 for D1, D2, E_{hrc}

respectively .

Register OTS_CTL.OTSCK is used to select the temperature measurement clock. When selecting the HRC action, the frequency error may affect the accuracy of the final calculated temperature. To eliminate this error, start XTAL32 before the temperature measurement and use Ehrc when calculating the temperature.

TSSTP is used to select whether to turn off the analog temperature sensor after the temperature measurement is completed; the initial value of TSSTP is "**0**", which means the analog temperature sensor will be turned on after a temperature measurement is completed, so that the stabilization time of the analog temperature sensor from off to on will be automatically skipped in the next temperature measurement. To turn off the analog temperature sensor after each temperature measurement, set TSSTP to "**1**".

The temperature measurement can be started by other peripheral events, so please set the trigger target of the trigger source to OTS. When using the temperature measurement completion interrupt, please set the register OTS_CTL.OTSIE to "1".

17.3 Register Description

Base address: 0x4004_A400

Table 17-1 List of OTS registers

Register Name	Symbols	Offset Address	Bit width	Reset value
OTS control register	OTS_CTL	0x00	16	0x0000
OTS data register 1	OTS_DR1	0x02	16	0x0000
OTS data register 2	OTS_DR2	0x04	16	0x0000
OTS error compensation register	OTS_ECR	0x06	16	0x0000

17.3.1 OTS control register (OTS_CTL)

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	TSS TP	OTS IE	OTS CK	OTS ST

Reset value: 0x0000

position	Marker	Place Name	Function	Reading and writing
b15~b4	Reserved	-	Read 0 when reading, please write 0 when writing	R
b3	TSSTP	Turn off the analog temperature sensor	Select whether to automatically turn off the analog temperature sensor after the temperature measurement 0: Do not turn off the analog temperature sensor 1: Turn off the analog temperature sensor	R/W
b2	OTSIE	Interrupt enable bit	0: Temperature measurement end interrupt request is disabled 1: Allow temperature measurement end interrupt request	R/W
b1	OTSCK	Clock Select Bit	0: Select external high-speed clock (XTAL) action 1: Select internal high-speed clock (HRC) action	R/W
b0	OTSST	Temperature measurement start position	0: Stop measuring temperature 1: Start temperature measurement Set "1" condition: (1) Software set "1" (2) Hardware trigger to set "1" to clear "0" conditions: (1) Software Clear "0" (2) Automatic clearing of "0" at the end of temperature	R/W

measurement

17.3.2 OTS Data Register 1 (OTS_DR1)

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TSDC [15:0]															

Reset value: 0x0000

position	Marker	Place Name	Function	Reading and writing
b15~b0	TSDC [15:0]	Temperature data1	Temperature data D1 Automatic update after temperature measurement is completed. Please make sure that OTS_CTL.OTSST is "0" before reading it.	R

17.3.3 OTS Data Register 2 (OTS_DR2)

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TSDC [15:0]															

Reset value: 0x0000

position	Marker	Place Name	Function	Reading and writing
b15~b0	TSDC [15:0]	Temperature data2	Temperature data D2 Automatic update after temperature measurement is completed. Please make sure that OTS_CTL.OTSST is "0" before reading it.	R

17.3.4 OTS Error Compensation Register (OTS_ECR)

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TSEC [15:0]															

Reset value: 0x0000

position	Marker	Place Name	Function	Reading and writing
b15~b0	TSEC [15:0]	Error factor	Error factor Ehrc Automatic update after temperature measurement is completed. Please make sure that OTS_CTL.OTSST is "0" before reading it.	R

18 Advanced Control Timer (Timer6)

18.1 Introduction

Advanced Control Timer 6 (Timer6) is a 16-bit count width high performance timer that can be used to count different forms of clock waveforms and output for external use. The timer supports both triangle waveform and sawtooth waveform modes and can generate various PWM waveforms; software-synchronous counting and hardware-synchronous counting between units; cache function for each reference value register; 2-phase quadrature encoding and 3-phase quadrature encoding; and EMB control. Timer6 with 3 units is included in this series.

18.2 Basic Block Diagram

The basic functions and features of Timer6 are shown in Table 18-1.

Table 18-1 Basic Functions and Features of Timer6

Waveform mode	Sawtooth wave (incremental and decremental counting) triangular wave (incremental and decremental counting)
Basic Functions	- Capture input
	- Software Synchronization
	- Hardware Synchronization
	- Cache function
	- Pulse Width Measurement
	- Periodic measurements
	- Orthogonal coding count
	- Universal PWM output
Interrupt output	- EMB Control
	Counting comparison match interrupts
	Counting cycle matching interrupts
Event Output	Dead time error interrupt
	Counting comparison match events
	Counting cycle matching events

The basic block diagram of Timer6 is shown in Figure 18-1. The "<t>" shown in the block diagram indicates the cell number, i.e., "<t>" is 1~3, and the references to "<t>" later in this chapter refer to the cell number. later in this chapter, all references to "<t>" refer to the unit number, and will not be repeated.

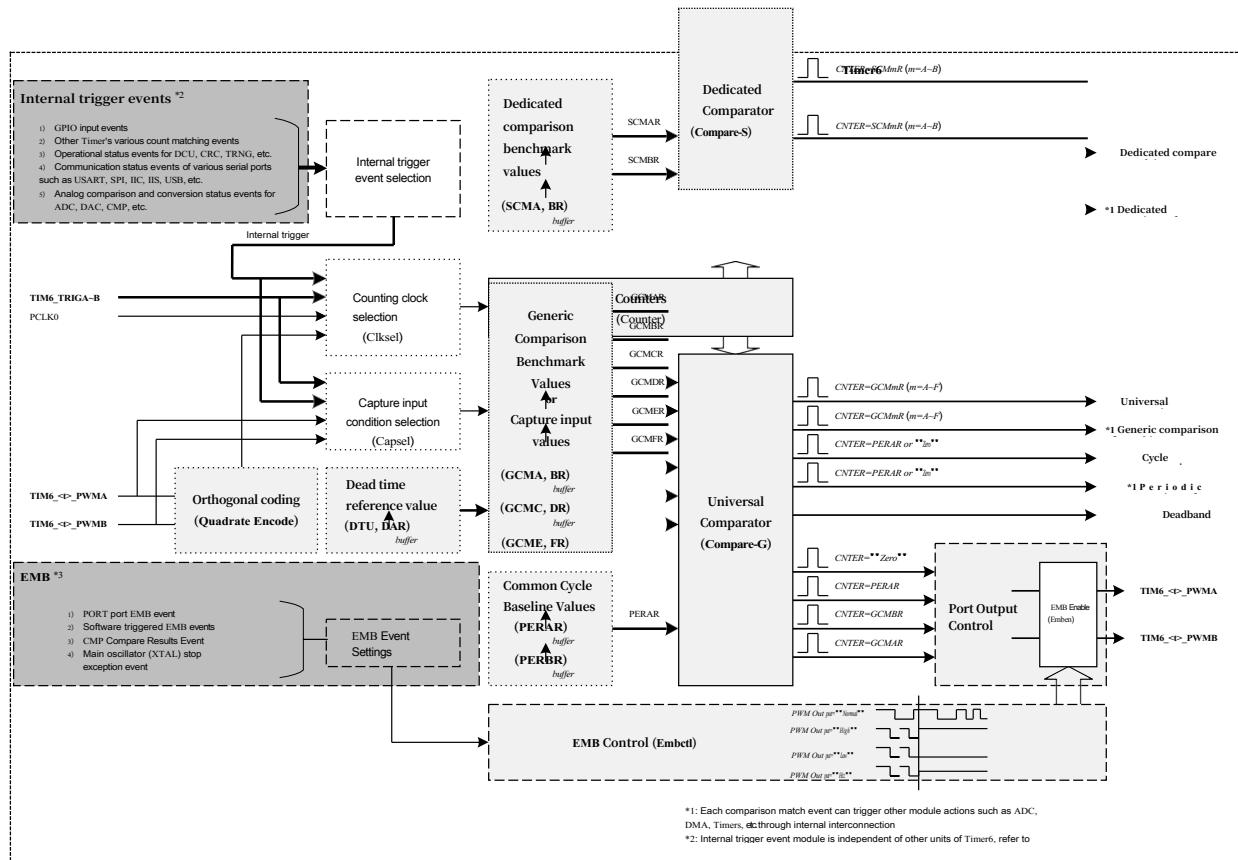


Figure 18-1 Timer6 Basic Block Diagram

Table 18-2 shows the list of input and output ports for Timer6.

Table 18-2 Timer6 Port List

Port Name	Direction	Function
TIM6_<t>_PWMA	in or out	1) Quadrature coded count clock input port or capture input port or compare output port
TIM6_<t>_PWMB		2) Hardware start, stop, and clear condition input ports
TIM6_TRIGA	in	1) Hardware count clock input port or capture input port
TIM6_TRIGB		2) Hardware start, stop, and clear condition input ports

18.3 Function Description

18.3.1 Basic movements

18.3.1.1 Waveform mode

Timer6 has 2 basic counting waveform modes, sawtooth waveform mode and triangle waveform mode. The triangle waveform mode is further divided into triangle waveform A mode and triangle waveform B mode due to different internal counting actions. The basic waveforms of sawtooth and triangle waveforms are shown in Figure 18-2 and Figure 18-3.

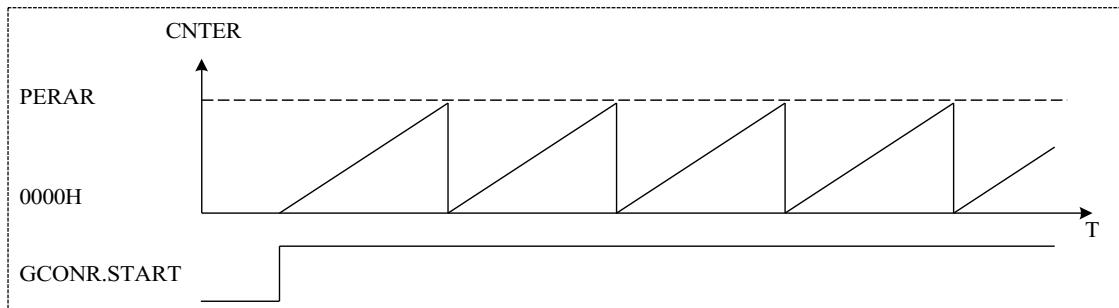


Figure 18-2 Sawtooth waveform (recursive counting)

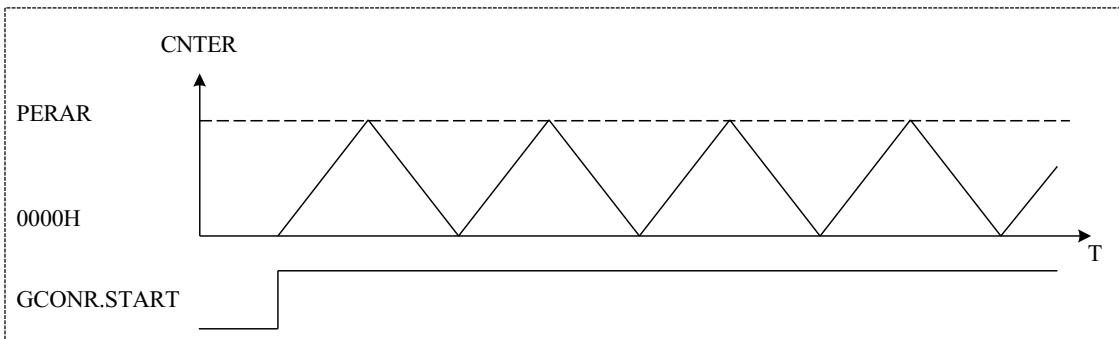


Figure 18-3 Triangular waveform

18.3.1.2 Compare Outputs

Timer6 of each cell has 2 comparison output ports (TIM6_<t>_PWMA, TIM6_<t>_PWMB) which can output the specified level when the count value matches the comparison reference value. The GCMAR and GCMBR registers correspond to the count comparison reference values of TIM6_<t>_PWMA and TIM6_<t>_PWMB, respectively. The TIM6_<t>_PWMA port outputs the specified level when the timer count value is equal to GCMAR, and the TIM6_<t>_PWMB port outputs the specified level when the timer count value is equal to GCMBR.

The count start level, count stop level, count compare match level and count period match level of TIM6_<t>_PWMA and TIM6_<t>_PWMB ports can be set by PCONR.STACA and PCONR.**STPCA** of the Port Control Register (PCONR). STPCA,
PCONR, PCONR.STASTPSA, PCONR. PCONR,
PCONR.CMPCA[1:0], PCONR. , PCONR,

PCONR.PERCA[1:0] bits are set. Figure 18-4 shows the action example of comparison output.

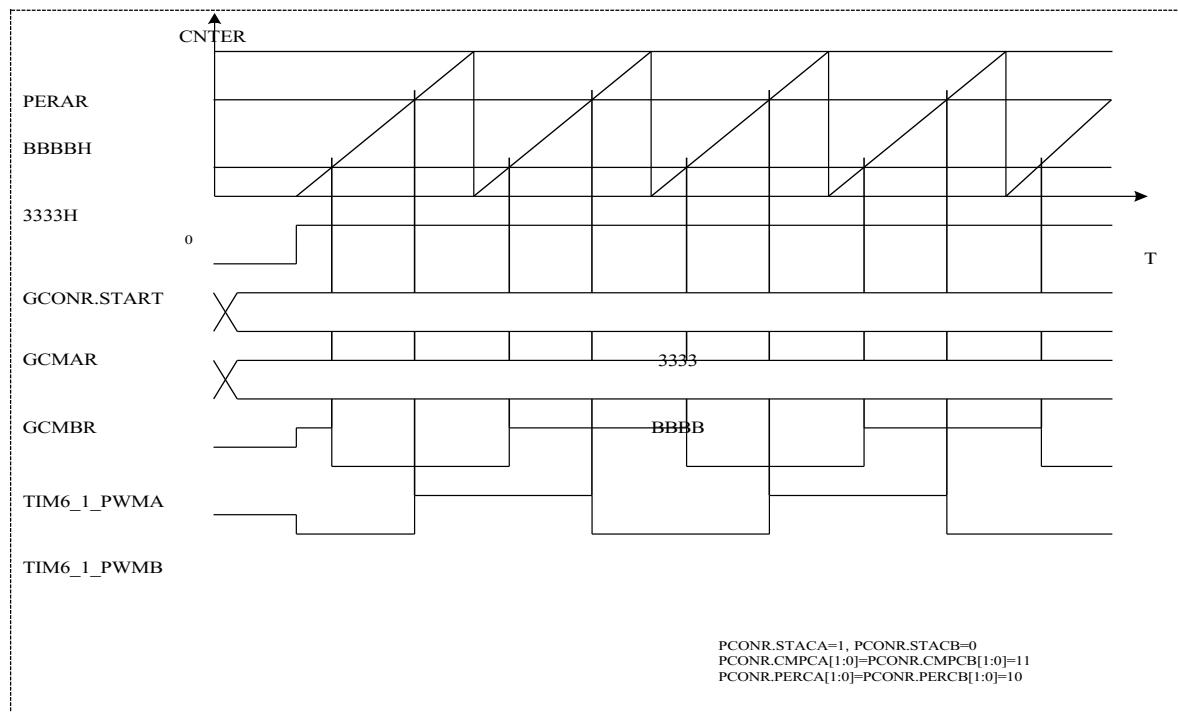


Figure 18-4 Compare Output Actions

18.3.1.3 Capture input

Each unit has a capture input function with 2 sets of capture input registers (GCMAR, GCMBR) for saving the captured count value. CAPMDA and PCONR.CAPMDB bits of the Port Control Register (PCONR) are set to 1 to make the capture input function valid. When the corresponding capture input condition is set and the condition is valid, the current count value is saved in the corresponding capture registers (GCMAR, GCMBR).

The conditions for each set of capture inputs of each unit can be internal trigger event inputs, TIM6_TRIGA or TIM6_TRIGB port inputs, TIM6_<t>_PWMA or TIM6_<t>_PWMB port inputs, etc. The specific selection of the conditions can be done through the hardware capture event selection registers (HCPAR, HCPBR) to be set. Figure 18-5 shows an example of an action to capture an input.

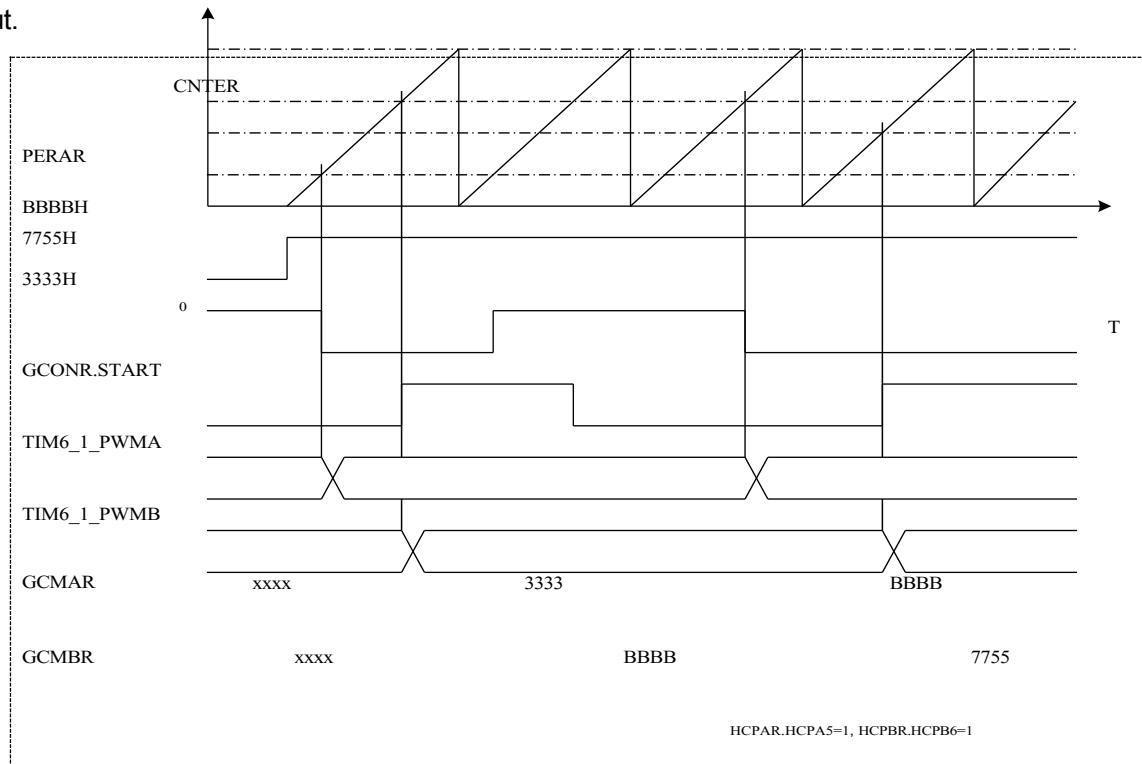


Figure 18-5 Capture Input Action

18.3.2 Clock source selection

Timer6's count clock can have the following options:

- a) PCLK0 and 2, 4, 8, 16, 64, 256, 1024 divisions of PCLK0 (GCONR.CKDIV[2:0] setting)
- b) Internal trigger event trigger input (HCUPR[17:16] or HCDOR[17:16] setting)
- c) Port input for TIM6_TRIGA-B (HCUPR[11:8] or HCDOR[11:8] setting)
- d) Quadrature coded inputs (HCUPR[7:0] or HCDOR[7:0]) for TIM6_<t>_PWMA and TIM6_<t>_PWMB
(Setting)

The counting clock source selection a is the software counting mode, and the counting clock source selection b, c, and d is the hardware counting mode. As you can see from the above description, the b, c, and d clocks are independent of each other and can be set to be valid or invalid respectively, and the a clock is automatically invalidated when the b, c, and d clocks are selected.

18.3.3 Counting direction

Timer6's timer count direction can be changed by software. The method of changing the count direction is slightly different for different waveform modes.

18.3.3.1 Sawtooth wave counting direction

In sawtooth wave mode, the counting direction can be set while the timer is counting or while it is stopped.

DIR=0 (decreasing counting)
In decreasing counting, the timer will change to decreasing counting mode after counting to upper overflow; when in decreasing counting, set GCONR.DIR=1 (increasing counting)
The timer will change to increasing counting mode after counting to lower overflow.

The GCONR.DIR bit is set when the count is stopped, and the setting of GCONR.DIR is reflected in the count after the count starts until the overflow or underflow.

18.3.3.2 Triangular wave counting direction

In triangle wave mode, the set counting direction is invalid, and the counting direction will be changed automatically when the count reaches the counting peak or counting valley.

18.3.4 Digital filtering

The TIM6_<t>_PWMA, TIM6_<t>_PWMB, and TIM6_TRIGA~B port inputs of Timer6 have a digital filtering function. The filtering function of the corresponding port can be enabled by setting the relevant enable bit in the Filter Control Register (FCONR). The filtering reference clock when filtering is active can also be set by the filtering control register (FCONR).

After the filtered sampling reference clock samples 3 consistent levels on the port, the level is transmitted to the module internally as a valid level; a level less than

The 3 consistent levels are filtered out as external interference and are not transmitted to the internal module. An example of this action is shown in Figure 18-6.

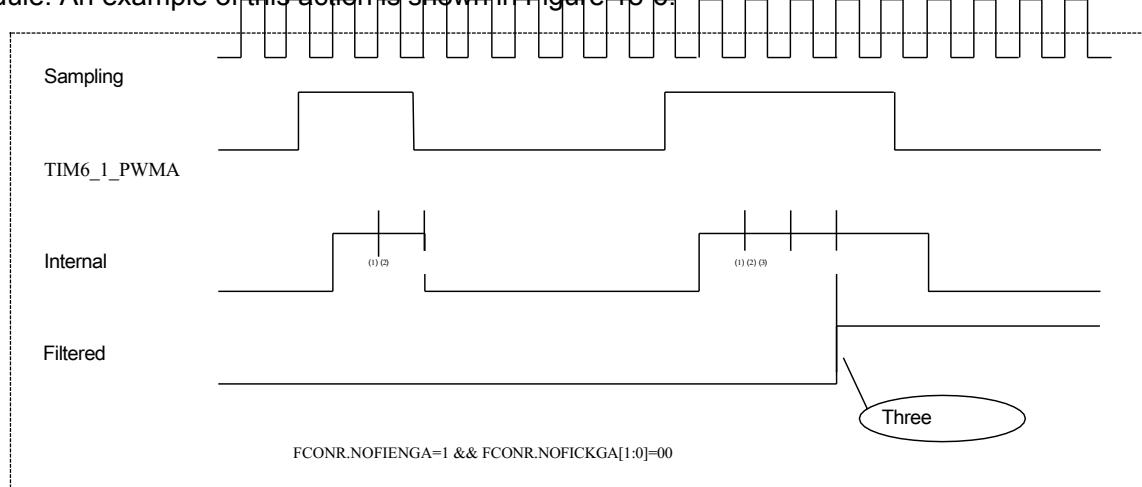


Figure 18-6 Capture the filtering function of the input port

The TIM6_TRIGA~B port is a set of ports shared between units, the digital filter function of this set of ports is set by the FCONR of unit 1, the FCONR of other units has no effect on the digital filter function setting of this set of ports. When any unit uses the digital filter function, it is necessary to clear the TIMER6_1 bit in the function controller (PWC_FCG2) to zero.

18.3.5 Software Synchronization

18.3.5.1 Software Synchronization Launch

Each unit can be synchronized with the target unit by setting the relevant bit in the Software Synchronized Start Control Register (SSTAR).

18.3.5.2 Software synchronization stop

Each unit can be synchronized to stop the target unit by setting the relevant bit in the Software Synchronized Stop Control Register (SSTPR).

18.3.5.3 Software synchronization zeroing

Each unit can achieve synchronous clearing of the target unit by setting the relevant bit in the software synchronous clearing control register (SCLRR).

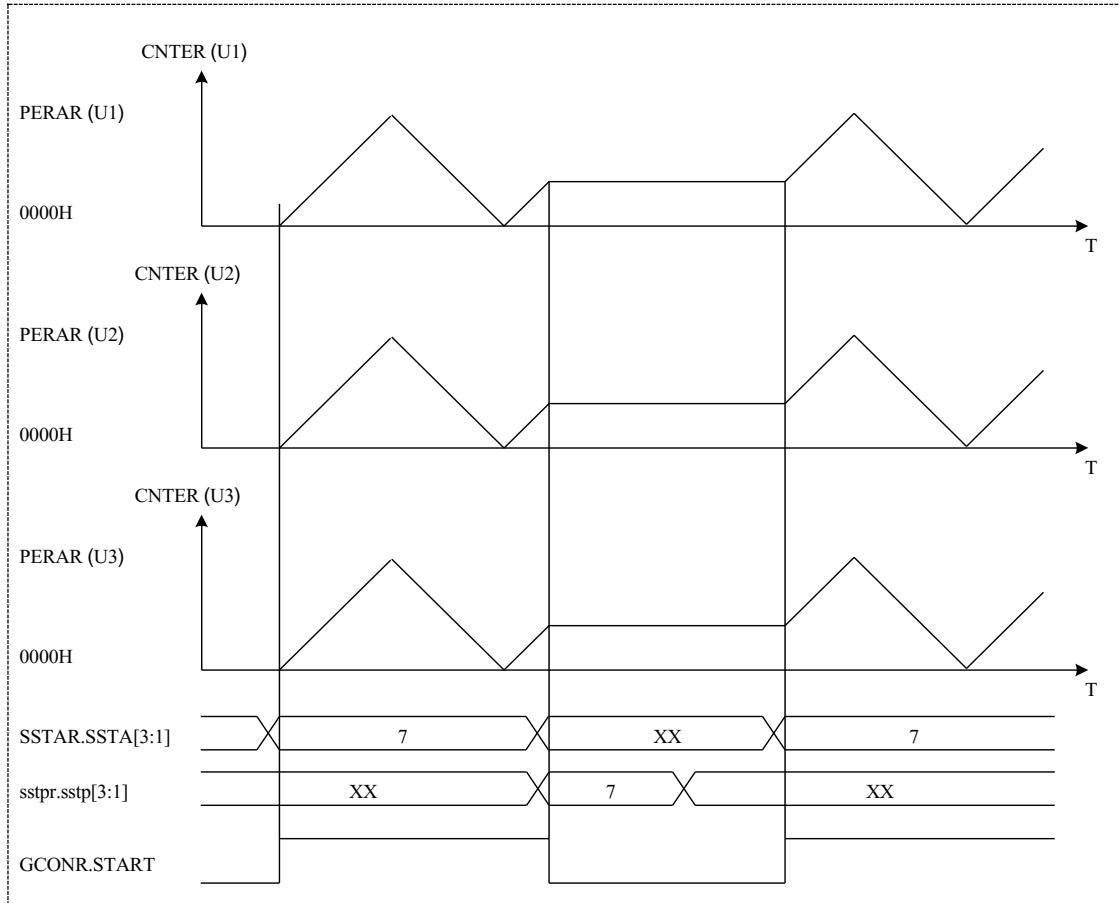


Figure 18-7 Software synchronization action

SSTAR.SSTA1=SSTAR.SSTA2=SSTAR.SSTA3=1 as shown in Figure 18-7, to realize cell 1~3
The software is launched simultaneously.

The Software Synchronization Action Related Registers (SSTAR, SSTPR, and SCLRR) are a set of registers that are independent of the unit and shared among the units. The bits in this set of registers are only valid when writing a 1 and invalid when writing a 0. When the SSTAR register is read, the timer status of each unit is read, and when the SSTPR or SCLRR is read, a 0 is read.

18.3.6 Hardware Synchronization

In addition to 2 general-purpose input ports (TIM6_<t>_PWMA, TIM6_<t>_PWMB), each unit also has 2 external general-purpose input ports (TIM6_TRIGA, TIM6_TRIGB) and 2 internal trigger event input conditions, which can realize hardware synchronization between units.

The event source of the internal hardware trigger event can be selected by the corresponding number setting in the hardware trigger event selection registers (HTSSR0~1) please refer to the Interrupt Controller (INTC) chapter for the specific event correspondence. When using the internal hardware trigger function, you need to enable the peripheral circuit trigger function in the function clock control register 0 (PWC_FCG0) to position 1 first.

18.3.6.1 Hardware Synchronized Start

Each unit can choose to start the timer in hardware mode. Selecting the unit with the same hardware start condition can realize synchronous start when the start condition is valid. The specific hardware start condition is determined by the setting of the hardware start event selection register (HSTAR).

18.3.6.2 Hardware synchronization stop

Each unit can choose to stop the timer by hardware, and the unit with the same hardware stop condition can achieve synchronous stop when the stop condition is valid. The specific hardware stop condition is determined by the setting of the hardware stop event selection register (HSTPR).

18.3.6.3 Hardware Synchronous Zeroing

Each unit can choose to clear the timer in hardware mode, and the unit with the same hardware clear condition can achieve synchronous clearing when the clear condition is valid. The specific hardware clearing condition is determined by the setting of the hardware clearing event selection register (HCLRR).

18.3.6.4 Hardware synchronized capture input

Each unit can choose to implement the capture input function in hardware, and the unit with the same capture input function condition can synchronize the capture input when the capture input function condition is valid. The specific hardware capture input function condition is determined by the setting of the hardware capture event selection registers (HCPAR, HCPBR).

18.3.6.5 Hardware synchronized counting

Each unit can choose to count with hardware input as CLOCK, and units with the same hardware count condition can achieve synchronous counting when the hardware count clock is valid. The specific hardware count condition is determined by the setting of the hardware decrement event select register (HCUPR) and the hardware decrement event select register (HCDOR).

Figure 18-8 shows an example of the hardware synchronization action for units 1~3.

When the hardware synchronous counting function is selected, only the external input clock source is selected, which does not affect the start, stop, and clear action of the timer. The timer start, stop, and zero clear still need to be set separately.

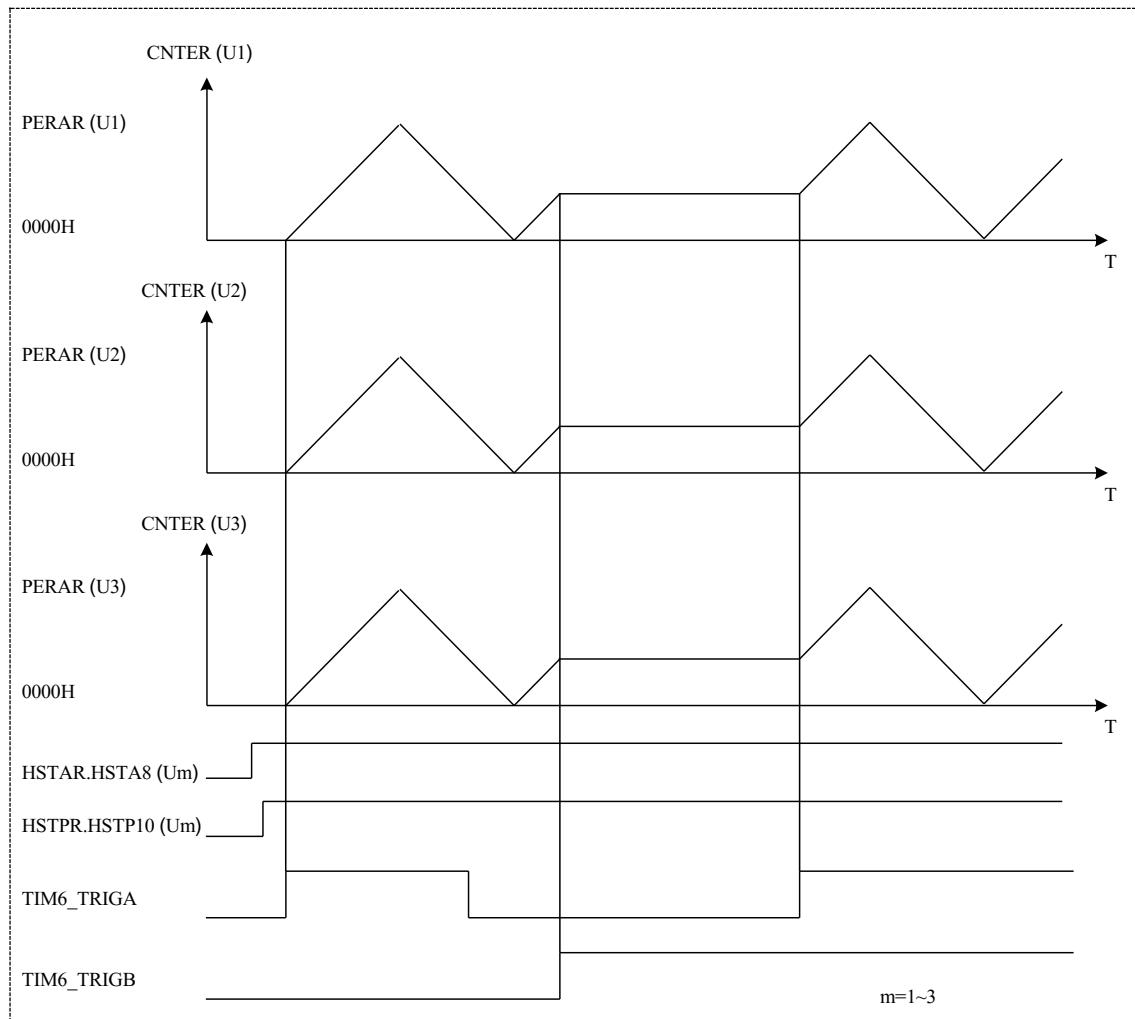


Figure 18-8 Hardware synchronization action

18.3.7 Pulse Width Measurement

When using the hardware trigger-related functions of the TIM6_<t>_TRIGA~B port (refer to the Hardware Synchronization chapter), each unit can implement 2 independent pulse width measurement functions.

For example, by setting the hardware start condition of the counter to the rising edge of TIM6_<t>_TRIGA and the hardware clear condition, the stop condition and the capture input condition of the GCMAR register to the falling edge of TIM6_<t>_TRIGA, a continuous pulse width measurement can be achieved.

18.3.8 Periodic measurements

When using the hardware trigger-related functions of the TIM6_<t>_TRIGA~B port (refer to the Hardware Synchronization chapter), each unit can implement 2 independent cycle measurement functions.

For example, by setting the hardware start condition of the counter, the hardware clear condition, and the capture input condition of the GCMBR register to the rising edge of TIM6_<t>_TRIGB, a continuous cycle measurement is possible.

18.3.9 Cache function

Timer6 has a cache function for the count cycle value, general comparison reference value, special comparison reference value, and dead time setting value, which enables the cycle change, duty cycle change, and dead time change during the count. The count cycle value, general comparison reference value, and special comparison reference value have single cache and double cache functions, and the dead time setting value has a single cache function.

18.3.9.1 Single cache action

Single cache action means that by setting the cache control register (BCONR) and the deadband control register (DCONR), the following events are selected to occur at the cache transfer time point:

- a) The value of the General Periodic Reference Value Buffer Register (PERBR) is automatically transferred to the General Periodic Reference Value Register (PERAR)
- b) The value of the General Comparison Reference Value Buffer Register (GCMCRGCMDR) is automatically transferred to the General Comparison Reference Value Register (GCMAR, GCMDR)
- c) The value of the General Comparison Base Value Register (GCMARGCMBR) is automatically transferred to the General Comparison Base Value Buffer Register (GCMCR, GCMDR) (when capturing input)
- d) The values of the dedicated comparison reference cache registers (SCMCR, SCMDR) are automatically transferred to the dedicated comparison reference registers (SCMAR, SCMBR)

-
- e) The value of the dead time reference cache register (DTUBRD, DTDBR) is automatically transferred to the dead time reference register (DTUAR, DTDAR)

As shown in Figure 18-9, it is the timing diagram of the single cache mode of the General Comparison Reference Value Register when Unit 1 compares the output action. As can be seen from the figure, changing the value of the General Comparison Reference Value Register (GCMAR) during counting adjusts the output duty cycle, and changing the value of the General Periodic Reference Value Register (PERAR) adjusts the output period.

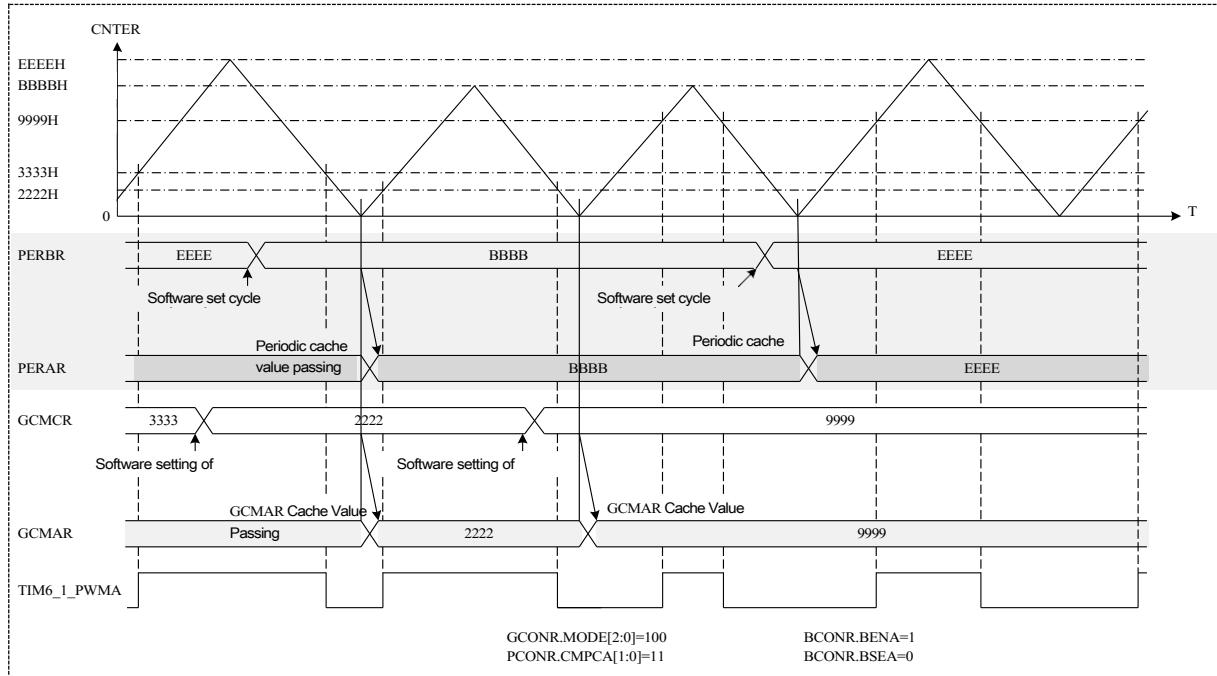


Figure 18-9 Single cache method to compare output timings

18.3.9.2 Dual cache action

A double cache action is the selection of the following events to occur at the cache transfer time point by setting the cache control register (BCONR)

- The value of the general period reference cache register (PERBR) is automatically transferred to the general period reference register (PERAR), and the value of the general period reference double cache register (PERCR) is automatically transferred to the general period reference cache register (PERBR)
- The values of the General Comparison Base Value Cache Registers (GCMCR, GCMDR) and the values of the General Comparison Base Value Dual Cache Registers (GCMER, GCMFR) are automatically transferred to the General Comparison Base Value Cache Registers (GCMCR, GCMDR) (when comparing outputs)
- The value of the general comparison reference cache register (GCMCR, GCMDR) is automatically transferred to the general comparison reference double cache register (GCMER, GCMFR) and the value in the General Comparison Base Value Register (GCMAR, GCMBR) are automatically transferred to the General Comparison Base Value Cache Register

(GCMCR, GCMDR) (when capturing input)

- d) The values of the dedicated comparison reference cache registers (SCMCR\$CMDR),~~the~~ the reference registers (SCMAR, SCMBR), and the values of the dedicated comparison reference dual cache registers (SCMER, SCMFR) are automatically transferred to the dedicated comparison reference cache registers (SCMCR, SCMDR)

Figure 18-10 shows the timing diagram for the dual cache method when internal trigger event 0 triggers the capture input.

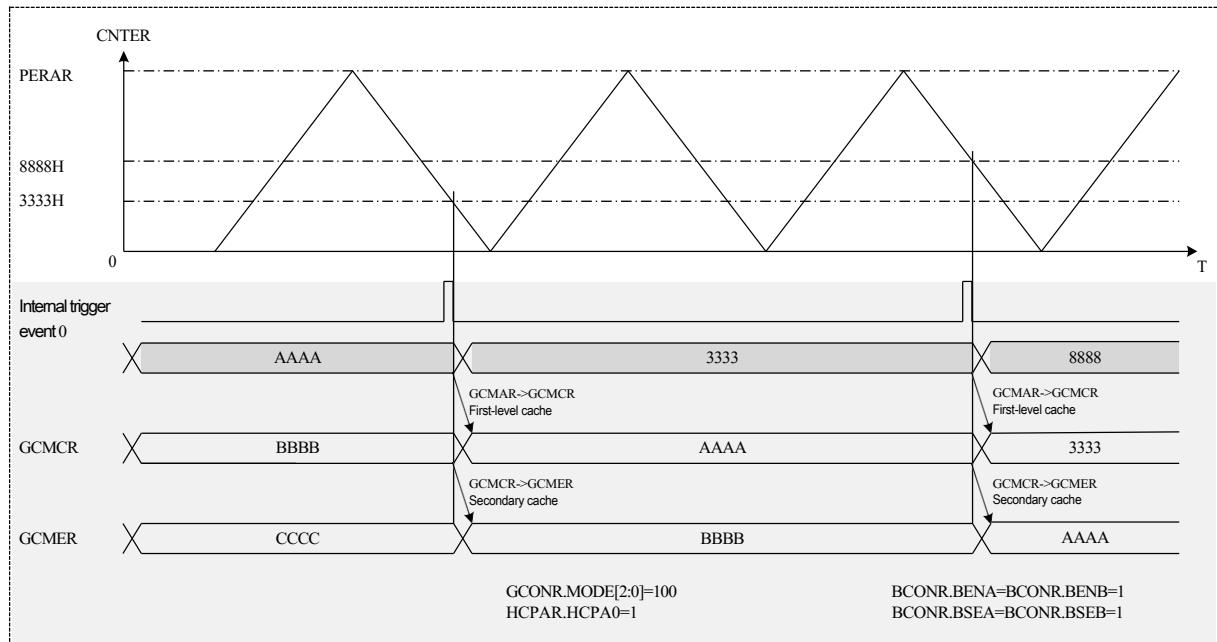


Figure 18-10 Dual Cache Approach to Capture Input Timing

18.3.9.3 Cache transfer time point

Generic cycle reference value cache transfer time point

The cycle reference value can be selected as single cache function or double cache function (BCONR.BSEP) The cache transfer time point is the incremental count up overflow point or decremental count down overflow point for sawtooth waves and the count valley point for delta waves.

Generic comparison of reference value cache transfer time points

BCONR.BENA=1 or BCONR.BNEB=1 to make the cache action effective when the ramp mode is set. The cache action can be selected as single cache function or double cache function. The cache transfer occurs at the upper or lower overflow point, as shown in Figure 18-11.

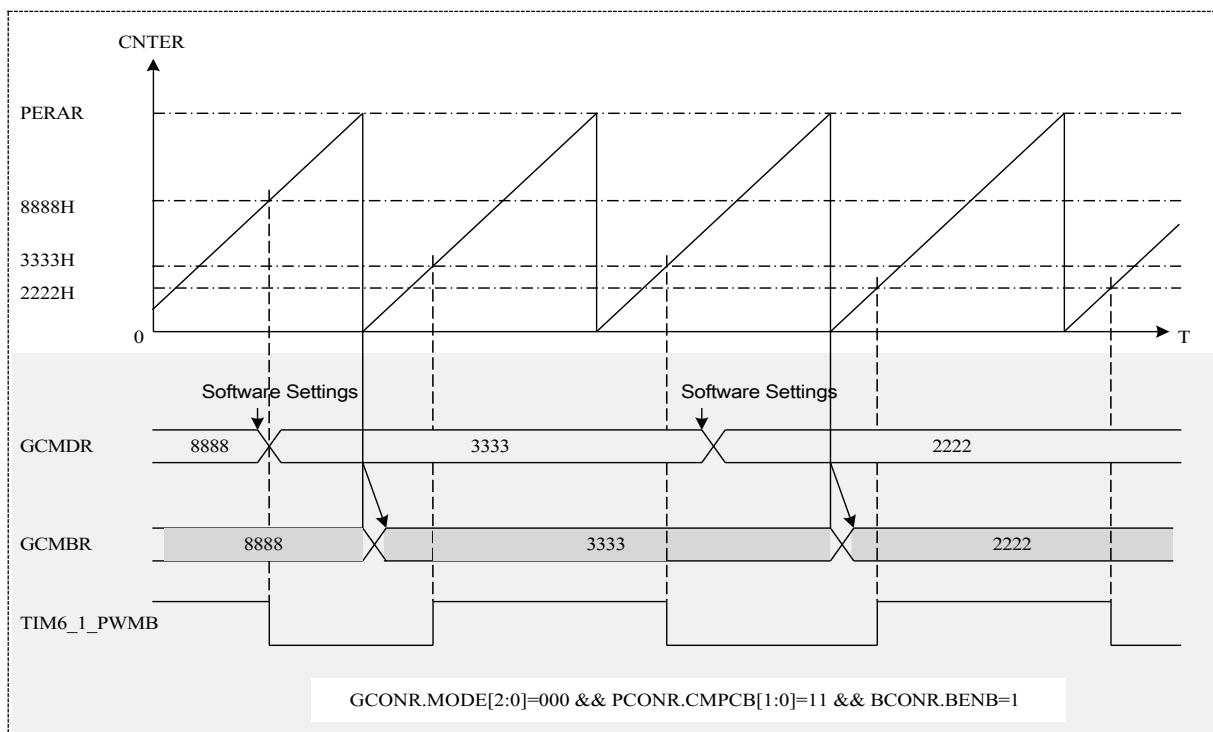


Figure 18-11 Counting cache action during sawtooth wave mode

BCONR.BENA=1 or BCONR.BNEB=1 when triangle wave A mode is set, the cache action is valid. The cache action can be selected as single cache function or double cache function. The cache transmission occurs at the count valley as shown in Figure 18-12.

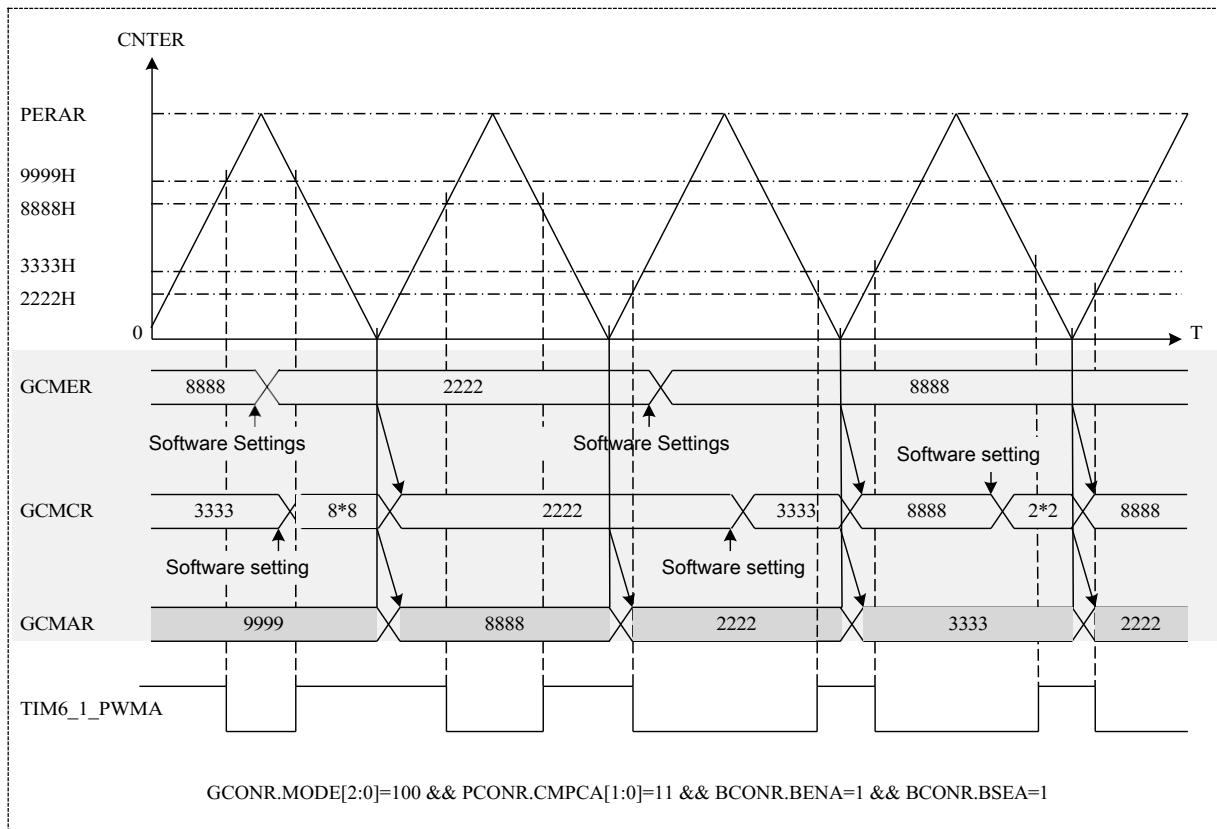


Figure 18-12 Counting cache action in triangular wave A mode

BCONR.BENA=1 or BCONR.BNEB=1 when triangle wave B mode is set, the cache action is valid. The cache action can be selected as single cache function or double cache function. The cache transmission occurs at the count valley or count peak, as shown in Figure 18-13.

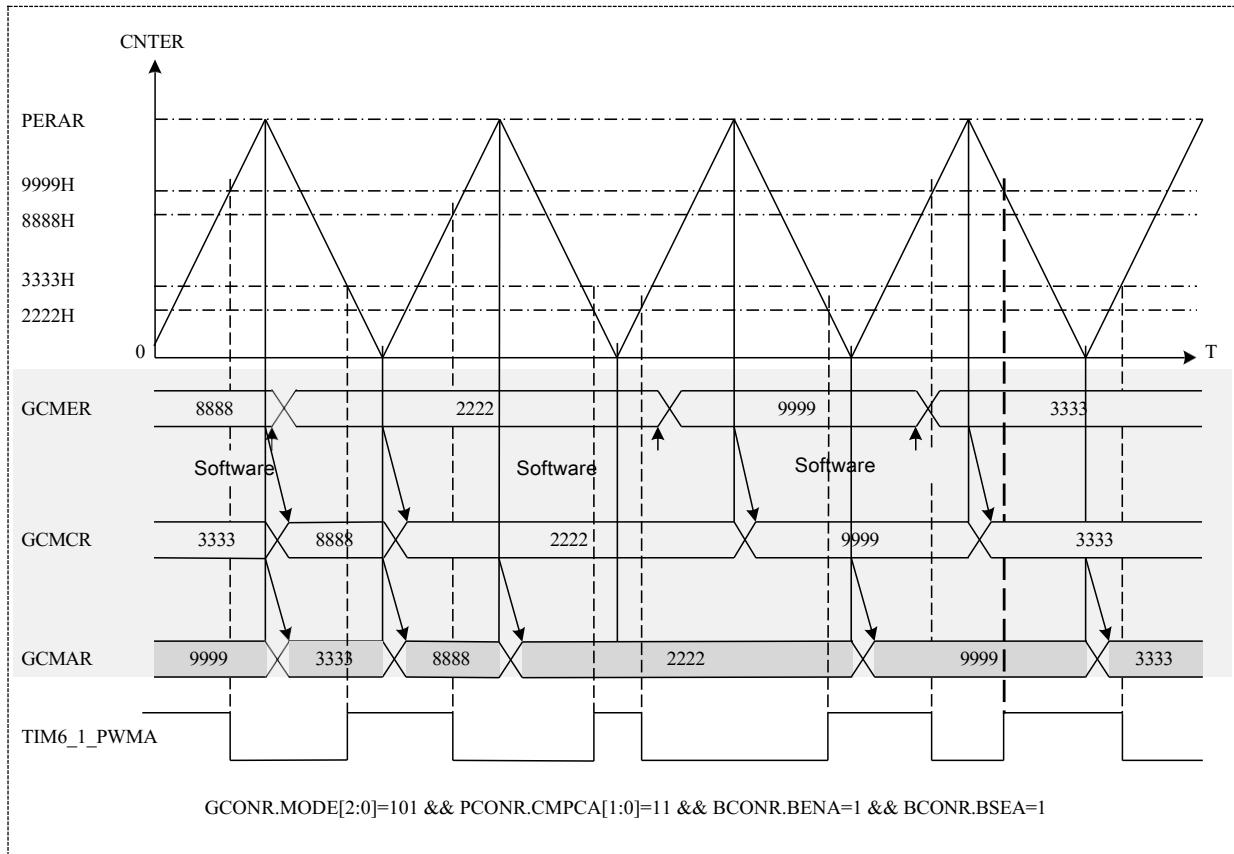


Figure 18-13 Counting cache action in triangular wave B mode

Single cache transfer or double cache transfer is determined by BCONR.BENA, BCONR.BENB, BCONR.BSEA, BCONR.BSEB.

Capture input value cache transfer time point

The capture input action can be selected as a single cache function or a dual cache function (BCONR.BSEA or BCONR.BSEB) The cache transfer time point is when the input action is captured.

Dedicated comparison reference value cache transfer time point

The dedicated comparison reference value can be selected for single cache function or dual cache function (BCONR.BESPA or BCONR.BESPB) The cache transfer time point is set by BCONR.BTRSPA and BCONR.BTRSPB of the cache control register BCONR.

Dead time reference value cache transfer time point

The dead time reference value has a single cache function. The cache transmission time point is the incremental count up overflow point or decremental count down overflow point for sawtooth waves and the count valley point for delta waves.

Cache transfer during clear action

If a clearing action is generated during the compare output action in the sawtooth counting mode or hardware counting mode, a cache transfer occurs in the general-purpose cycle reference value, general-purpose comparison reference value, dedicated comparison reference value, and dead time reference value registers according to the corresponding cache action setting status (single cache, double cache, etc.).

18.3.10 Universal PWM Output

18.3.10.1 Independent PWM output

The 2 ports of each unit, TIM6_<t>_PWMA and TIM6_<t>_PWMB, can output PWM waves independently.

As shown in Figure 18-14, the TIM6_<t>_PWMA port outputs a PWM waveform.

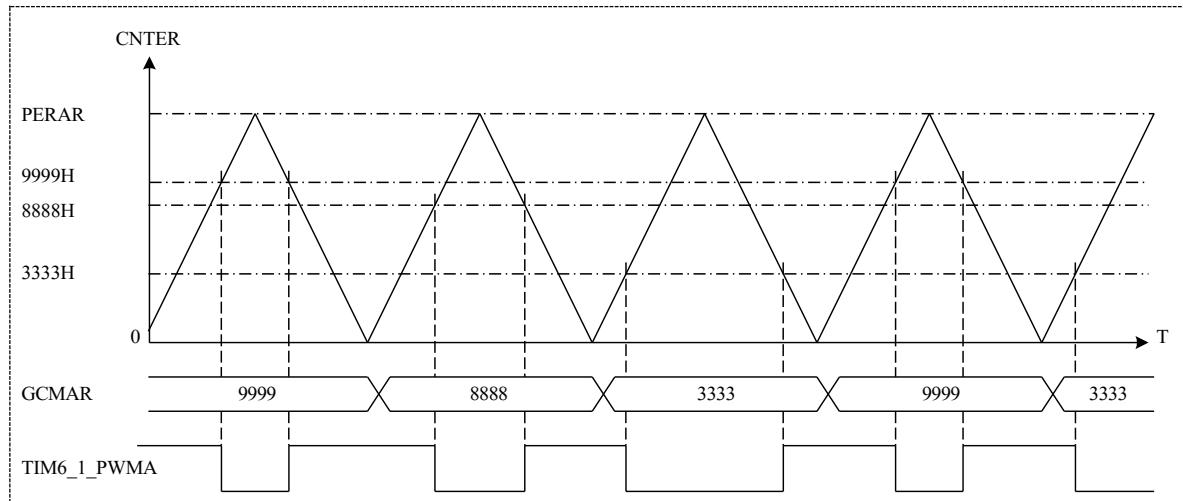


Figure 18-14 TIM6_<t>_PWMA Output PWM Wave

18.3.10.2 Complementary PWM Outputs

The TIM6_<t>_PWMA port and the TIM6_<t>_PWMB port can be combined to output complementary PWM waveforms in different modes.

Software Setting GCMBR Complementary PWM Output

The software setting GCMBR complementary PWM output means that the General Comparison Reference Value Register (GCMBR) used for the waveform output of TIM6_<t>_PWMB port is written directly by the CPU, etc. in the sawtooth and delta waveform modes, and is not directly related to the value of GCMAR.

Figure 18-15 shows an example of software setting the output of the GCMBR complementary PWM waveform.

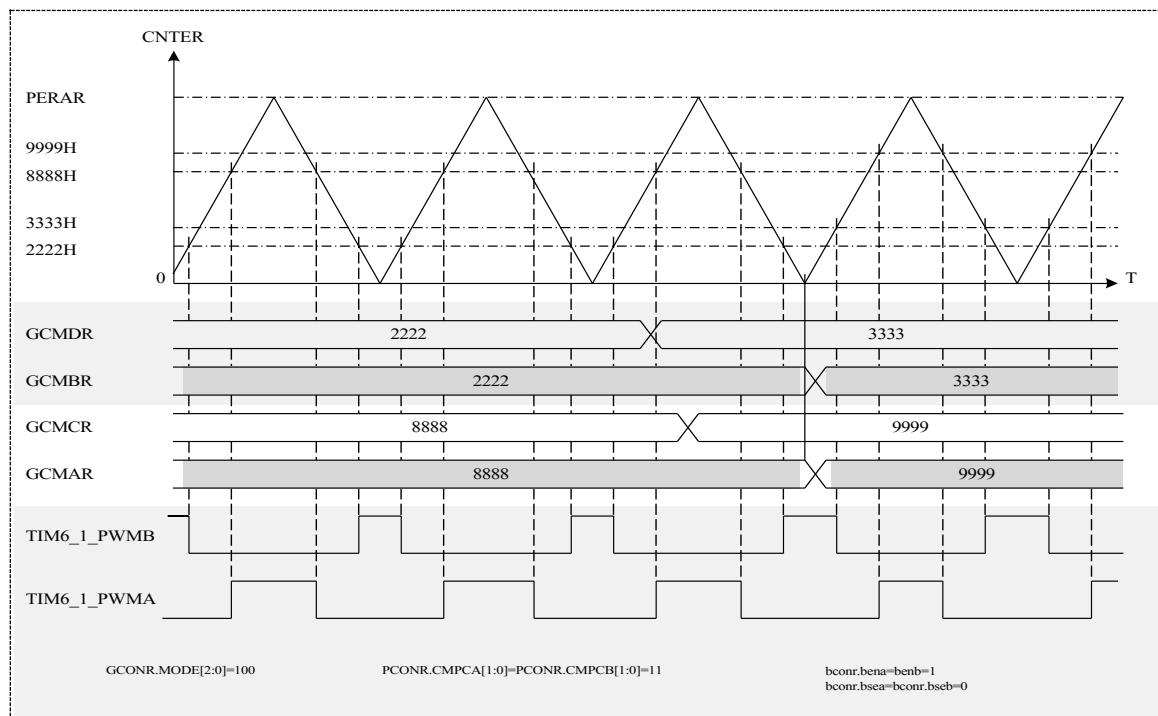


Figure 18-15 Software Setting GCMBR Complementary PWM Wave Output in Triangle A Mode

Hardware Setting GCMBR Complementary PWM Output

The hardware setting GCMBR complementary PWM output means that the value of the General Comparison Base Register (GCMBR) used for the TIM6_<1>_PWMB port waveform output in delta mode is determined by the General Comparison Base Register (GCMAR) and the Dead Time Base Register (DTU<D>AR) is determined by the value of the operation.

The dead time setting also has a cache function. When the cache function is active (DCONR.DTBENU/DTBEND=1), the value of DTUBR is transmitted to DTUAR and the value of DTDBR is transmitted to DTDAR at the cache transmission time point (count valley at delta wave)

Figure 18-16 shows the example of hardware setting GCMBR complementary PWM wave output.

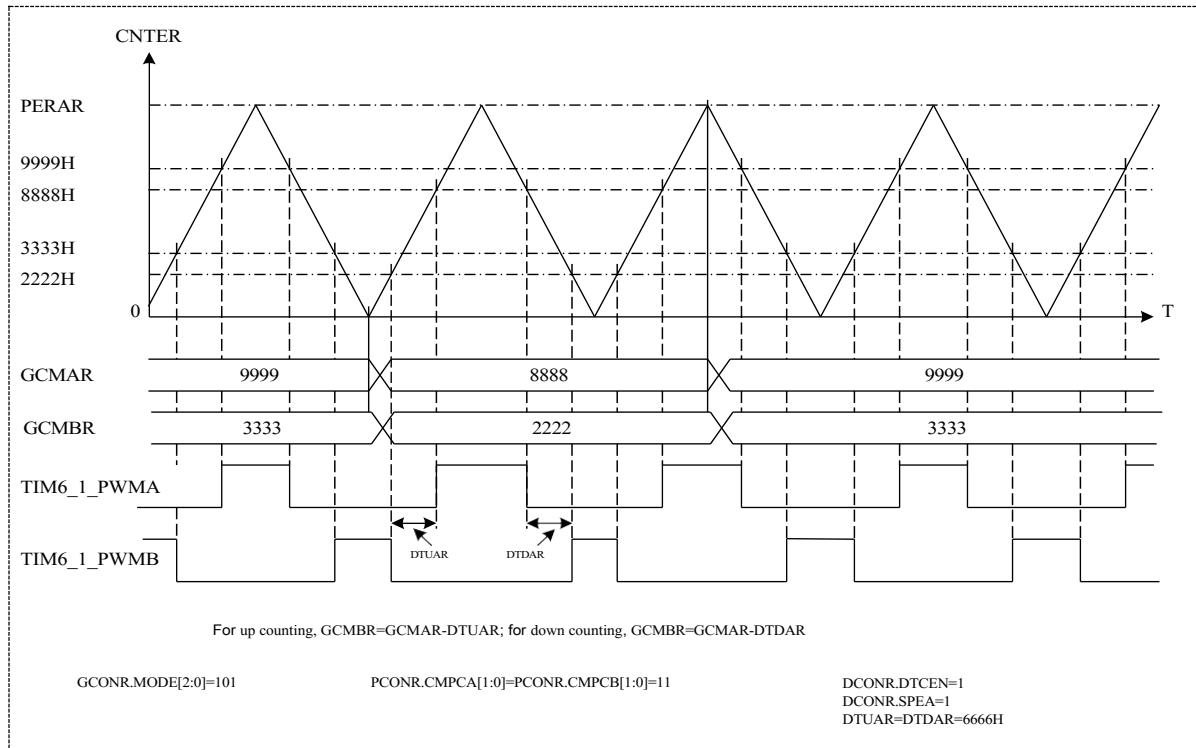


Figure 18-16 Hardware Setting GCMBR Complementary PWM Wave Output in Triangle B Mode (Symmetric Deadband)

18.3.10.3 Multi-phase PWM output

The TIM6_<t>_PWMA and TIM6_<t>_PWMB ports of each unit can output two independent PWM waves or a set of complementary PWM waves, which can be combined among multiple units and combined with software and hardware synchronization actions to achieve multi-phase PWM output. In Figure 18-17, Unit 1, Unit 2, and Unit 3 combine to output a 6-phase PWM waveform; in Figure 18-18, Unit 1, Unit 2, and Unit 3 combine to output a 3-phase complementary PWM waveform.

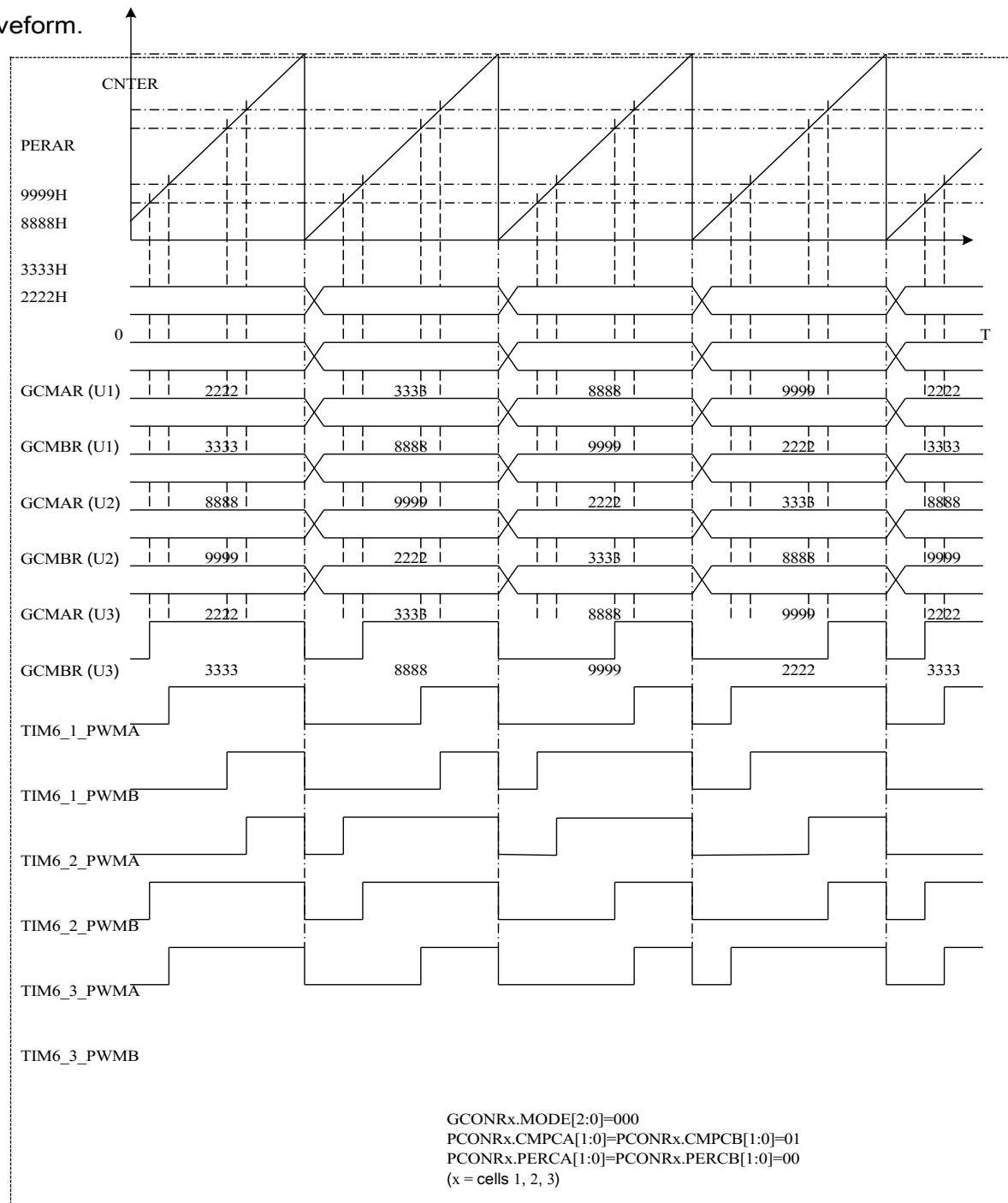
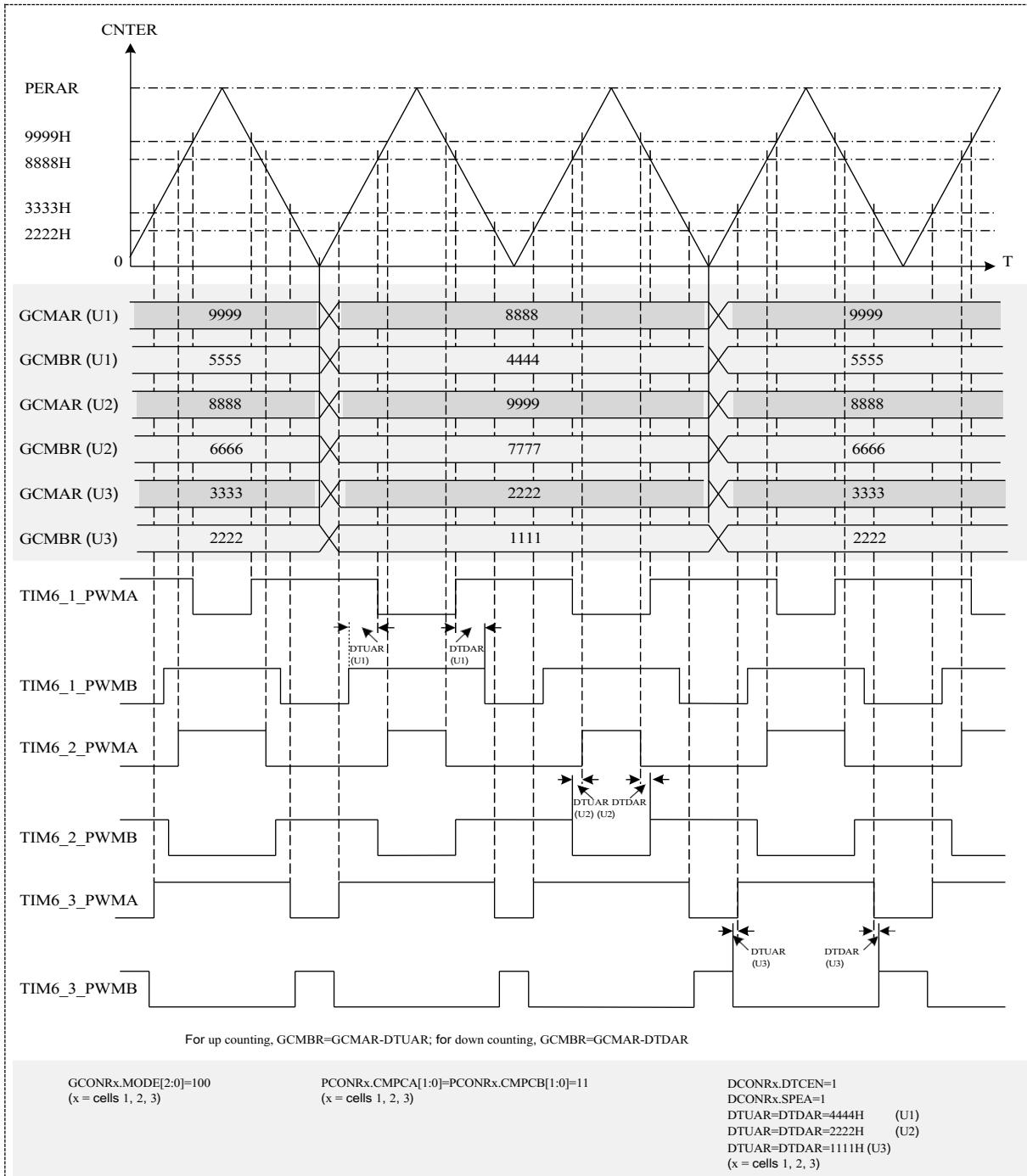


Figure 18-17 6-Phase PWM Wave



**Figure 18-18 Three-phase complementary PWM wave output
with dead time in triangular A mode**

18.3.11 Orthogonal coding count

Consider the TIM6_<t>_PWMA input as AIN input, the TIM6_<t>_PWMB input as BIN input, and the **TIM6_TRIGA-**

Any one of the inputs in B is regarded as ZIN input, and Timer6 can realize the quadrature encoding count of three inputs.

The AIN and BIN of one unit alone can realize the position counting mode; the AIN, BIN and ZIN of two units combined can realize the revolution counting mode, one unit is used for position counting and one unit is used for revolution counting.

In the revolution counting mode, unit 1 and 2 are combined, with unit 1 as the position counting unit and unit 2 as the revolution counting unit to achieve position counting and revolution counting, respectively. Unit 3 is not used in the revolution mode.

The count conditions for AIN and BIN are set by setting the Hardware Incremental Event Select Register (HCUPR) and the Hardware Decremental Event Select Register

(The input action of ZIN is realized by setting the hardware clear event selection register (HCLRR) of the position counter unit to clear the position timer of the position counter unit, and by setting the hardware recurrence event selection register (HCUPR) of the revolution counter unit to count the revolution timer of the revolution counter unit. The input action of ZIN is realized by setting the hardware clear event selection register (HCLRR) of the position unit to clear the position timer of the position counter unit, and by setting the hardware increment event selection register (HCUPR) of the rotation counter unit to count the rotation timer.

18.3.11.1 Position counting mode

The orthogonal encoding position counting mode means that the basic counting function, phase difference counting function and direction counting function are implemented according to the input of AIN and BIN.

Basic Count

The basic counting action is based on the input clock of the AIN or BIN port, as shown in Figure 18-19 below.

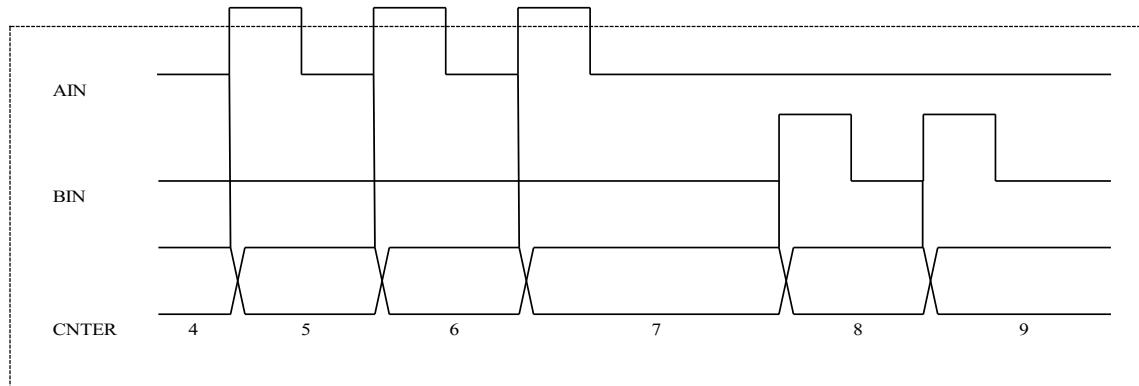
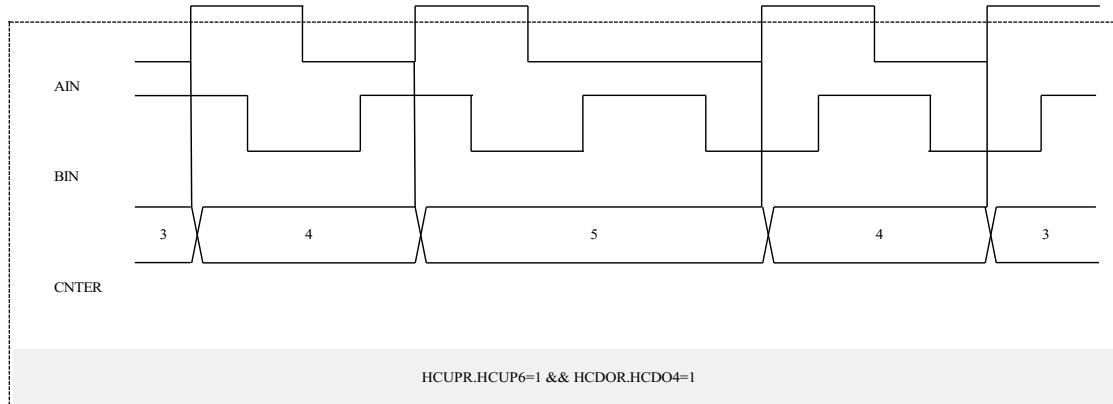


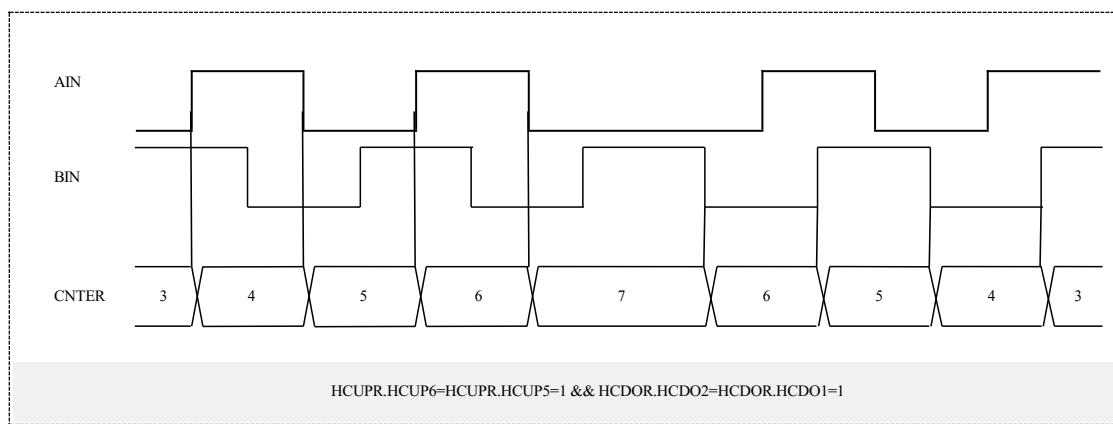
Figure 18-19 Position Mode - Basic Counting

Phase difference counting

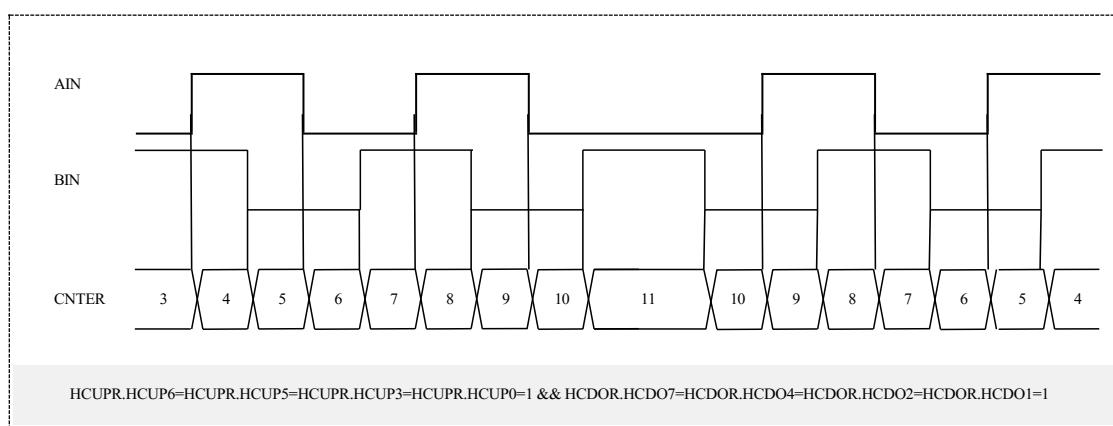
Phase difference counting refers to counting based on the phase relationship between AIN and BIN. Depending on the setting, 1X counting, 2X counting, 4X counting, etc. can be realized as shown in Figure 18-20 to Figure 18-22 below.



**Figure 18-20 Position Counting Mode - Phase Difference Counting
(1X Count)**



**Figure 18-21 Position Counting Mode - Phase Difference Counting
(2x Counting)**



**Figure 18-22 Position Counting Mode - Phase Difference Counting
(4x Counting)**

Direction Counting

Direction counting means setting the input state of AIN as direction control and using the input of BIN as clock counting, as shown in Figure 18-23 below.

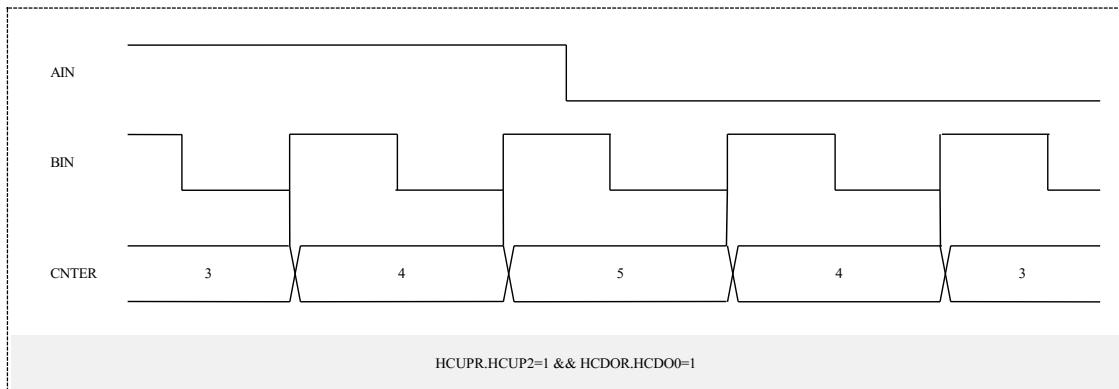


Figure 18-23 Position Count Mode - Direction Count

18.3.11.2 Rotation counting mode

The orthogonal coding revolution counting mode is to add ZIN input events to the AIN and BIN counting to judge the revolution number, etc. The Z-phase counting function, the position overflow counting function and the mixed counting function can be realized according to the counting method of the rotation timer in the rotation counting mode.

Z-phase count

Z-phase counting is a counting action in which the revolution counter unit counts according to the input of ZIN and the position counter unit is cleared to zero at the same time. As shown in Figure 18-24 below.

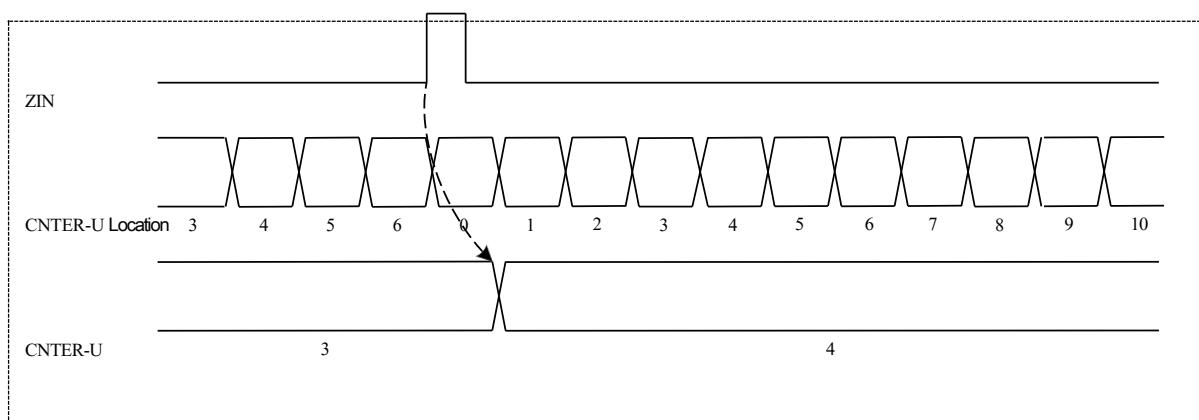


Figure 18-24 Rotation counting mode - Z-phase counting

Position Overflow Count

Position overflow counting means that an overflow event is generated when the count of the position counter unit overflows or underflows, which triggers the timer of the revolution counter unit to perform a count (the input of ZIN does not perform the count action of the revolution counter unit and the zero action of the position counter unit during this counting method)

The overflow event of the position counting unit is selected through the internal trigger event interface to realize the counting of the revolution counting unit, which can realize the position overflow counting. The hardware incremental (decremental) event selection register (HCUPR or HCDOR) of the rotation counter unit selects bit 1 of Bit16~Bit7 for the incremental (decremental) event, and sets the event number in the corresponding event trigger selection register (HTSSR0~1) to the overflow or underflow event of the position counter unit. Refer to the Interrupt Controller (INTC) chapter for the specific event number. As shown in Figure 18-25 below.

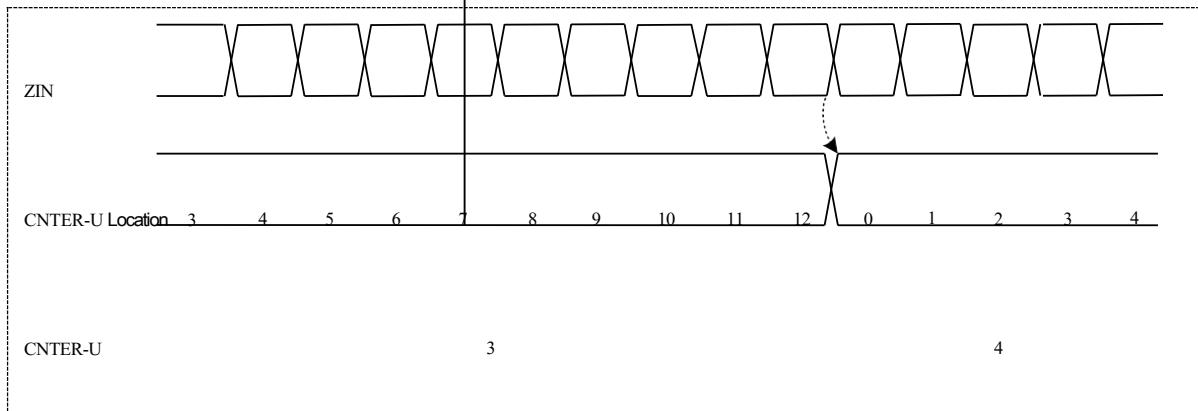


Figure 18-25 Rotation Counting Mode - Position Overflow Count

Mixed

countin

g

Mixed counting is a counting action that combines the above two counting methods, Z-phase counting and position overflow counting, and its implementation is also a combination of the above two counting methods. As shown in Figure 18-26 below.

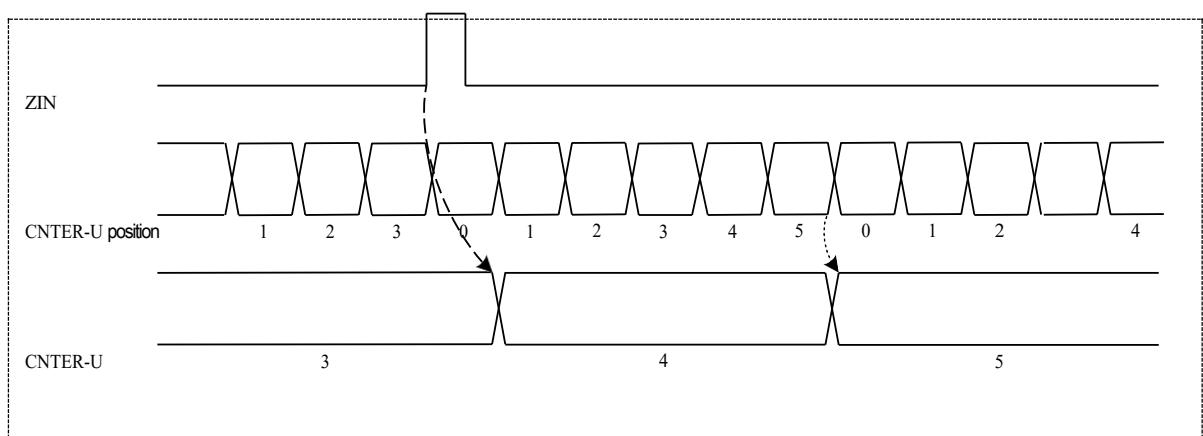


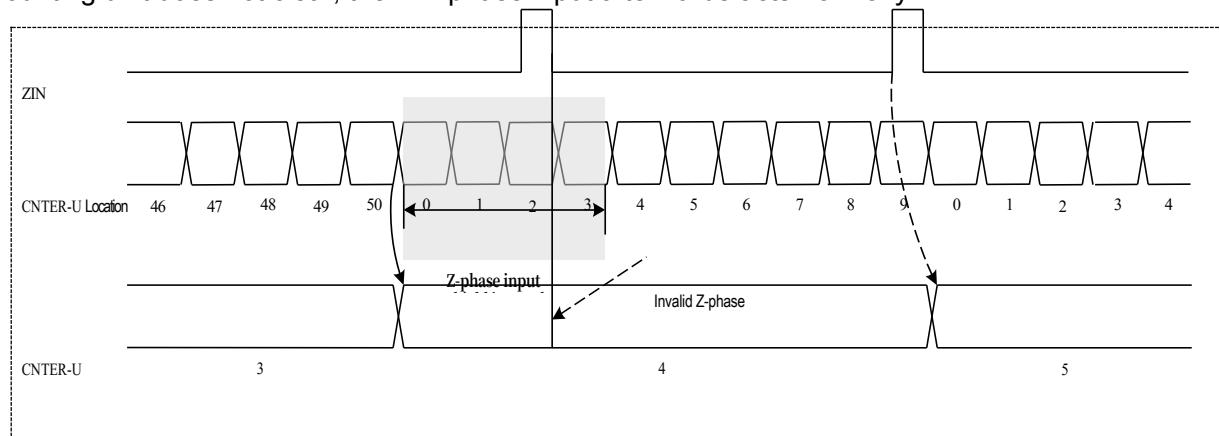
Figure 18-26 Rotation counting mode - mixed counting

18.3.11.3 Z-phase action shield

In the Z-phase counting function or mixed counting function of the revolution counting mode, you can set the valid input of ZIN to be masked from counting of the revolution counting unit and clearing of the position counting unit for several cycles after the upper or lower overflow point of the position timer (set by GCONR.ZMSKVAL[0 : 1]). When using this function, only Unit 1 and 2 can be combined, with Unit 1 as the position counter unit and Unit 2 as the revolution counter unit.

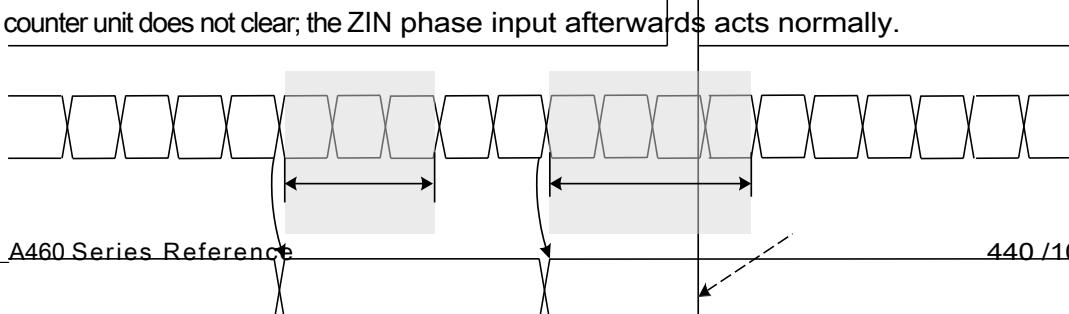
ZMSKPOS of the general control register (GCONR) of the position counter unit is 1, the Z-phase mask function of the position counter unit is enabled, and the cycle number of Z-phase mask is set by GCONR.ZMSKVAL; when GCONR.ZMSKREV of the general control register (GCONR) of the revolution counter unit is 1, the Z-phase mask function of the revolution counter unit is enabled. ZMSKREV of the general control register (GCONR) is 1.

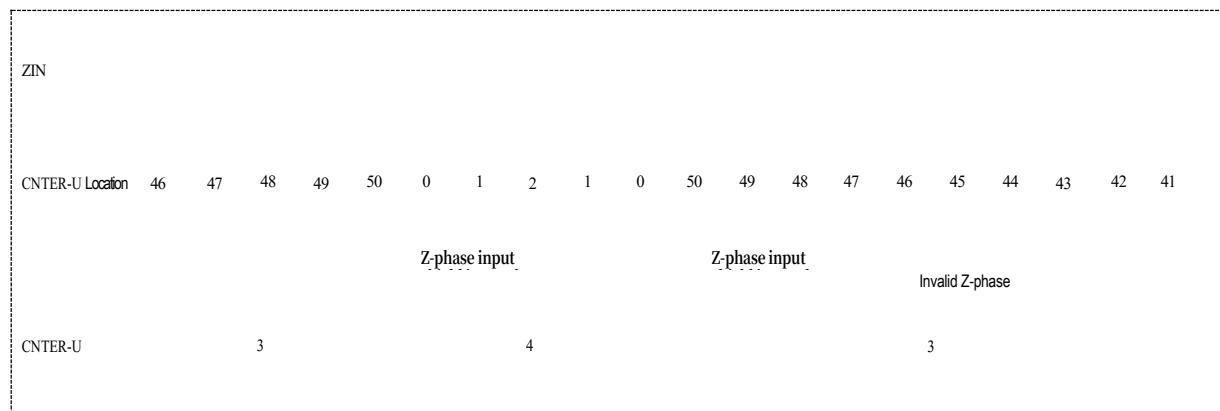
Figure 18-27 shows that in the mixed counting of the revolution counting mode, when there is a ZIN phase input within 4 counting cycles after the position counting unit counts overflow, the action of the ZIN phase input is invalid, i.e., the revolution counting unit does not count and the position counting unit does not clear; the ZIN phase input afterwards acts normally.



**Figure 18-27 Rotation counting mode - Hybrid
counting Z phase shield action example 1**

Figure 18-28 shows that in the mixed counting of the revolution counting mode, the counting direction changes at the 3rd cycle after the position counting unit counts overflow, and then the set 4 cycles of masking becomes invalid (the actual ZIN phase masking function is maintained for 3 cycles). Downward counting starts. After the count overflow of the position counter unit, the ZIN phase masking function is turned back on and becomes invalid after 4 cycles. During the ZIN phase blocking period, the ZIN phase input function is invalid, i.e., the revolution counter unit does not count and the position counter unit does not clear; the ZIN phase input afterwards acts normally.





**Figure 18-28 Rotation counting mode - Hybrid
counting Z phase shield action example 2**

18.3.12 Cycle interval response

Timer6 has two dedicated comparison reference registers (SCMAR, SCMBR) which can output dedicated comparison match interrupt A signal and dedicated comparison match interrupt B signal respectively to INTC to generate corresponding interrupts when counting comparison matches; at the same time, it can output dedicated comparison match event A signal and dedicated comparison match event B signal respectively, which can be used to associate with other modules for action, mostly used to start ADC, etc.

This interrupt and event request signal can generate a valid request signal after every few cycles, i.e., to achieve a cycle interval response. This function is enabled by setting the VPERR.PCNTE[1:0] bits and VPERR.SPPerIA/B bits of the Valid Period Register (VPERRPCNTS[2:0] bits to specify how many cycles the request signal is valid, and no valid request signal will be output in other cycles even if the count value is equal to the value of the dedicated comparison reference value register SCMAR or SCMBR.

If you stop and restart the timer while using the period interval response function, configure VPERR.PCNTE[1:0]=00 before stopping the timer, otherwise there may be a deviation in the moment when the period interval valid request signal is first generated after restarting.

When this function is valid, the period match interrupt and period match event in each waveform mode are also output only in the valid period of the dedicated compare match interrupt and event output (the period with STFLR.VPERNUM=0 in the figure below). Figure 18-29 shows an example of the action of the period interval valid request signal.

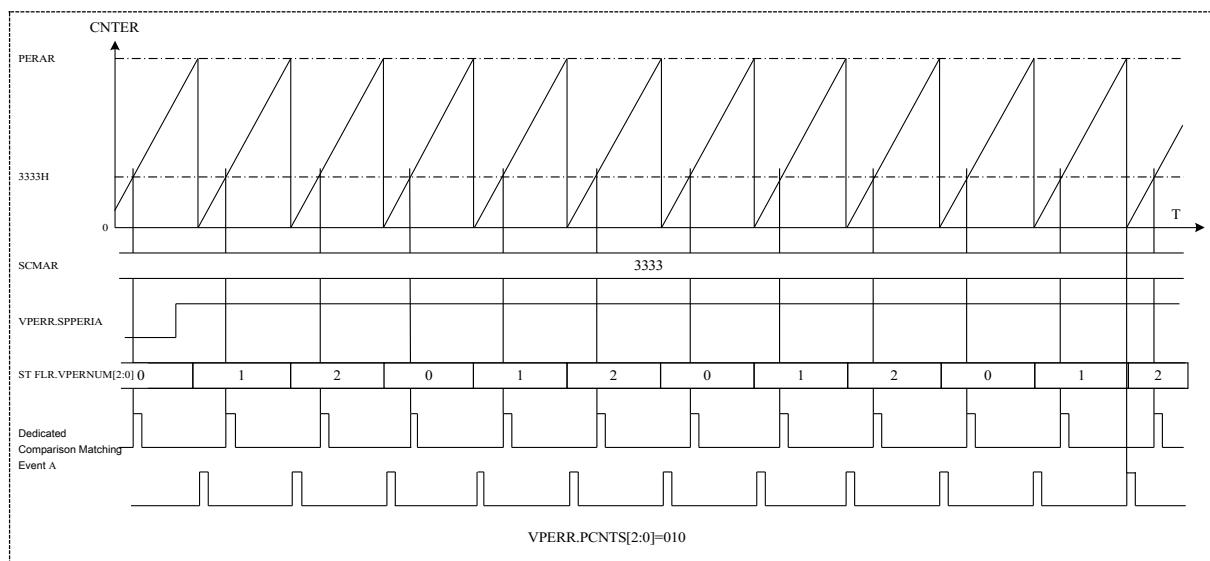


Figure 18-29 Cycle interval valid request signal action

18.3.13 EMB

Control

Timer6 provides protection control of the port output state, fixing the port state to a pre-defined safe state in the event of an exception. All units have a common port output control interface, which connects to a group of EMB events output by the EMB module. Also the abnormal condition events selected on the interface can be set from the EMB side (see EMB section) enabling control of the generic PWM outputs when an abnormal condition is monitored on these interfaces.

The port's output state can change to a predefined state if an EMB event from the EMB is monitored during normal output. The general-purpose PWM output port can change its state to output high resistance, output low, or output

High level (determined by the setting of PCONR.EMBVALA, PCONR.EMBVALB) For example, if PCONR.EMBVALA=01 set

timed, the output on the TIM6_<t>_PWMA port changes to a high resistance state if an EMB event is generated during the normal output of the TIM6_<t>_PWMA port.

After the abnormal event selected by the EMB module disappears and the EMB module resets the corresponding event status bit, Timer6 will automatically release the protection state at the next immediately adjacent cycle point (count trough of triangle wave, upflow or downflow of sawtooth wave) and turn into normal PWM output, thus realizing the Cycle By Cycle control of the PWM port.

18.3.14 Typical application examples

The following describes the basic settings of Timer6-related registers in several typical application cases for users' reference.

18.3.14.1 Basic counting and interrupt action

- a) Set the common cycle reference value (PERAR)
- b) Set the required comparison reference values, including the general comparison reference values (GCMAR~GCMFR) and the special comparison reference values (SCMAR~SCMBR), etc.
- c) Set the required interrupt enable bits, including count up interrupt (ICONR.INTENOVF) count down interrupt (ICONR.INTENUDF) count matching interrupt (ICONR.INTENA~F, ICONR.INTENSAU, ICONR.INTENSAD, ICONR.INTENSBU, ICONR.INTENSBD), etc.
- d) Set internal count clock division (GCONR.CKDIV[2:0])
- e) Set waveform mode (GCONR.MODE[2:0])
- f) Set the counting direction (required only when GCONR.MODE[2:0]=000 in sawtooth mode)
- g) Start counter (GCONR.START=1)

18.3.14.2 Compare output and interrupt action

- a) Set the common cycle reference value (PERAR)
- b) Set the comparison reference value for each channel, including the general comparison reference value A (GCMAR) general comparison reference value B (GCMBR)
- c) Set the required interrupt enable bits, including count up interrupt (ICONR.INTENOVF) count down interrupt (ICONR.INTENUDF) count matching interrupt (ICONR.INTENA~B), etc.
- d) Set the port output state for each channel at different count states (refer to bit7~bit1 and bit23~bit17 of PCONR for related controls)

- e) Set internal count clock division (GCONR.CKDIV[2:0])
- f) Set the waveform mode (GCONR.MODE[2:0])
- g) Set the counting direction (required only when GCONR.MODE[2:0]=000 in sawtooth mode)
- h) Set the comparison output mode for each channel (PCONR.CAPMDA=0, PCONR.CAPMDB=0)
- i) Set each channel output enable (PCONR.OUTENA=1, PCONR.OUTENB=1)
- j) Start counter (GCONR.START=1)

18.3.14.3 Capture input and interrupt actions

- a) Set the common cycle reference value (PERAR)
- b) Set the required interrupt enable bits, including count up interrupt (ICONR.INTENOVF) count down interrupt (ICONR.INTENUDF) capture input interrupt (ICONR.INTENA~B), etc.
- c) Set the capture input external conditions for each channel (refer to all valid control bits of HCPAR or HCPBR. The valid control bits are independent of each other, so that multiple ones can be selected at the same time as the capture input condition for a channel)
- d) Set internal count clock division (GCONR.CKDIV[2:0])
- e) Set the waveform mode (GCONR.MODE[2:0])
- f) Set the counting direction (required only when GCONR.MODE[2:0]=000 in sawtooth mode)
- g) Set capture input mode (PCONR.CAPMDA=1, PCONR.CAPMDB=1)
- h) Start counter (GCONR.START=1)
- i) Wait for a capture input condition to be generated, read the capture input value of the corresponding channel (GCMAR or GCMBR) or wait for the corresponding interrupt to be generated

18.3.14.4 Cache transfer action (cycle reference value)

- a) Set the required generic cycle reference values (PERAR, PERBR, PERCR)
- b) Set single and double cache transfer method (BCONR.BSEP)
- c) Set internal count clock division (GCONR.CKDIV[2:0])
- d) Set the waveform mode (GCONR.MODE[2:0])(different cache transfer time points for different waveform modes)
- e) Set the counting direction (required only when GCONR.MODE[2:0]=000 in sawtooth mode)
- f) Set cache function active (BCONR.BENP=1)
- g) Start counter (GCONR.START=1)
- h) Wait for the corresponding cache transfer time point for the cache action to occur (PERBR->PERAR (when BCONR.BSEP=0)PERCR->PERBR->PERAR (when BCONR.BSEP=1))

18.3.14.5 Cache transfer action (generic comparison of reference values)

- a) Set the required General Comparison Basis values (GCMAR, GCMCR, GCMER, GCMBR, GCMDR, GCMFR)
- b) Set single and double cache transfer mode for each channel (BCONR.BSEA, BCONR.BSEB)
- c) Set internal count clock division (GCONR.CKDIV[2:0])
- d) Set the waveform mode (GCONR.MODE[2:0])(different cache transfer time points for

different waveform modes)

- e) Set the counting direction (required only when GCONR.MODE[2:0]=000 in sawtooth mode)
- f) Set each channel cache function valid (BCONR.BENA=1, BCONR.BENB=1)
- g) Start counter (GCONR.START=1)
- h) Wait for the corresponding cache transfer time point set by each channel to generate a cache action (GCMCR->GCMAR)
(when BCONR.BSEA=0), GCMER->GCMCR->GCMAR (when BCONR.BSEA=1), and

~~SCMAR~~ when BCONR.BSEB=0 ~~SCMCR~~ when BCONR_BSEB=1)
 (hour)

18.3.14.6 Cache transfer action (dedicated comparison of reference values)

- a) Set the desired dedicated comparison reference values (SCMAR, SCMCR, SCMER, SCMBR, SCMDR, SCMFR)
- b) Set single and double cache transfer mode for each channel (BCONR.BESPA, BCONR.BESPB)
- c) Set the cache transfer time point for each channel (BCONR.BTRSPA[1:0], BCONR.BTRSPB[1:0])
- d) Set internal count clock division (GCONR.CKDIV[2:0])
- e) Set the waveform mode (GCONR.MODE[2:0])
- f) Set the counting direction (required only when GCONR.MODE[2:0]=000 in sawtooth mode)
- g) Set each channel cache function valid (BCONR.BENSPA=1, BCONR.BENSPB=1)
- h) Start counter (GCONR.START=1)
- i) Wait for the corresponding cache transfer time point set by each channel to generate a cache action (SCMCR->SCMAR
 (when BCONR.BESPA=0) SCMER->SCMCR->SCMAR (when BCONR.BESPA=1),
 SCMDR->SCMBR (BCONR.BESPB=0 when), SCMFR->SCMDR
 SCMFR->SCMDR->SCMBR
 (When BCONR.BESPB=1))

18.3.14.7 Cache transfer action (deadband reference value)

- a) Set the desired dead time reference value (DTUAR, DTUBR, DTDAR, DTDBR)
- b) Set internal count clock division (GCONR.CKDIV[2:0])
- c) Set the waveform mode (GCONR.MODE[2:0])
- d) Set the counting direction (required only when GCONR.MODE[2:0]=000 in sawtooth mode)
- e) Set cache function valid (DCONR.DTBENU=1, DCONR.DTBEND=1)
- f) Set hardware deadband function active (DCONR.DTCEN=1)
- g) Start counter (GCONR.START=1)
- h) Wait for the corresponding cache transfer time point and cache action occurs (DTUBR->DTUAR, DTDBR->DTDAR)

18.3.14.8 Synchronized start-up action (software method)

- a) Refer to steps a~f in [Basic Counting and Interrupt Action] section to set each unit to be started synchronously.

-
- b) Synchronous start counter (set the corresponding bit of SSTAR register to 1, each cell corresponds to one register bit)

18.3.14.9 Synchronous start-up action (hardware method)

- a) Set the common cycle reference value (PERAR)
- b) Set the required comparison reference values, including the general comparison reference values (GCMAR~GCMFR) and the special comparison reference values (SCMAR~SCMBR), etc.
- c) Set the required interrupt enable bits, including count up interrupt (ICONR_INTENOVF) count down interrupt

- (ICONR.INTENUDF) count matching interrupt (ICONR.INTENA~F, ICONR.INTENSAU, ICONR._INTENSAD, ICONR._INTENSBU, ICONR._INTENSD), etc.
- d) Set hardware start condition (selected by HSTAR.HSTAx, x=0~1, 8~11)
 - e) Set hardware boot enable (HSTAR._STARTS=1)
 - f) Repeat steps a to e above for each unit that needs to be started synchronously (the settings in step d should be the same for each unit that needs to be started synchronously)
 - g) Wait for the set trigger event to be generated to confirm the synchronized start of each unit's counter

18.3.14.10 Orthogonal coding counting action (2 phases)

- a) Set the common cycle reference value (PERAR)
- b) Set the required comparison reference values, including the general comparison reference values (GCMAR~GCMFR) and the special comparison reference values (SCMAR~SCMBR), etc.
- c) Set the required interrupt enable bits, including count up interrupt (ICONR._INTENOVF) count down interrupt (ICONR._INTENUDF) count matching interrupt (ICONR.INTENA~F, ICONR.INTENSAU, ICONR._INTENSAD, ICONR._INTENSBU, ICONR._INTENSD), etc.
- d) Set the desired hardware up count condition (selected via HCUPR.HCUPx, x=0~7)
- e) Set the desired hardware down count condition (selected via HCDOR.HCDOx, x=0~7)
- f) Start counter (GCONR._START=1)
- g) Wait for the set orthogonal code count event to be generated to confirm that the counter is counting properly

18.3.14.11 Orthogonal coding counting action (3 phases)

- a) Refer to steps a~e in the section [Quadrature code counting action (2 phases)] to set the position counting unit.
- b) Set the hardware clear condition for the position counter unit (selected via HCLRR.HCLR x , $x=8\sim11$)
- c) Set position counter unit hardware clear enable (HCLRR.CLEARS=1)
- d) Sets the common period reference value (PERAR) for the revolution counter unit
- e) Set the comparison reference value of the revolution counter unit, including the general comparison reference value (GCMAR~GCMFR) and the special comparison reference value (SCMAR~SCMBR), etc.
- f) Set the interrupt enable bits required for the metric count unit, including the count up overflow interrupt (ICONR._INTENOVF) count down overflow interrupt (ICONR._INTENUDF) count match interrupt (ICONR.INTENA~F, ICONR.INTENSAU, ICONR._INTENSAD, ICONR._INTENSBU, ICONR._INTENSD)

ICONR.INTENSB), etc.

- g) Set the hardware up count condition 1 (ZIN phase input) for revolution counter unit (selected by HCUPR.HCUPx, x=8~11, the set event here should be the same as the event set by the position counter unit in step b)
- h) Set hardware up-count condition 2 of the revolution counter unit (overflow event input of the position counter unit)(internal hardware trigger event 0 selected via HCUPR.HCUP16)
- i) Sets the hardware down count condition for the revolution counter unit (downflow event input for the position counter unit) (via

HCDOR.HCDO17 Select internal hardware trigger event 1)

- j) Set the trigger source number in HTSSR0 to the count overflow event of the position counter unit (refer to the INTC chapter for this overflow event number)
- k) Set the trigger source number in HTSSR1 to the count overflow event of the position counter unit (refer to the INTC chapter for this overflow event number)
- l) Start the revolution counter unit counter (GCONR.START=1)
- m) Start position counter unit counter (GCONR.START=1)
- n) Wait for the set AIN, BIN, and ZIN phase count events to be generated to confirm that the counter is counting properly

18.3.14.12 Single PWM output

- a) Refer to the setting of steps **a to j** in the **【Compare Outputs and Interrupt Action】** section (the output states of the 2 PWM channels TIM6_<t>_PWMA and TIM6_<t>_PWMB inside each unit can be set independently, creating 2 uncorrelated single PWM outputs)

18.3.14.13 Complementary PWM output (software deadband)

- a) Set the common cycle reference value (PERAR)
- b) Setting the General Comparison Base A (GCMAR) General Comparison Base B (GCMBR)
- c) Set the required interrupt enable bits, including count up interrupt (ICONR.INTENOVF) count down interrupt (ICONR.INTENUDF) count matching interrupt (ICONR.INTENA~B), etc.
- d) Set the port output state at different count states (refer to the PCONR bit7~bit1, bit23~bit16 related control, combined with the GCMAR and GCMBR settings, you need to ensure that the 2 PWM outputs form a complementary deadband)
- e) Set internal count clock division (GCONR.CKDIV[2:0])
- f) Set the waveform mode to triangle mode (GCONR.MODE=100 or 101)
- g) Set comparison output mode (PCONR.CAPMDA=0, PCONR.CAPMDB=0)
- h) Set output enable (PCONR.OUTENA=1, PCONR.OUTENB=1)
- i) Start counter (GCONR.START=1)

18.3.14.14 Complementary PWM output (hardware deadband)

- a) Set the common cycle reference value (PERAR)
- b) Sets the general comparison reference value A (GCMAR) dead time reference value (DTUAR, DTDAR)
- c) Set the required interrupt enable bits, including count up interrupt (ICONR.INTENOVF) count down interrupt (ICONR.INTENUDF) count-match interrupt (ICONR.INTENA~B) dead-zone

error interrupt

(ICONR.INTENDTE), etc.

- d) Set the port output state at different count states (refer to the PCONR bit7~bit1, bit23~bit16 related control, combined with the GCMAR, DTUAR and DTDAR setting values, it is necessary to ensure that the 2 PWM outputs form a complementary deadband)

- e) Set internal count clock division (GCONR.CKDIV[2:0])
- f) Set the waveform mode to triangle mode (GCONR.MODE[2:0] = 100 or 101)
- g) Set the comparison output mode for each channel (PCONR.CAPMDA=0, PCONR.CAPMDB=0)
- h) Set each channel output enable (PCONR.OUTENA=1, PCONR.OUTENB=1)
- i) Set hardware deadband function active (DCONR.DTCEN=1)
- j) Start counter (GCONR.START=1)

18.3.14.15 EMB Monitoring and Interrupt Action

- a) Refer to steps a~h in the [Complementary PWM Output (Software Deadband)] section or steps a~i in the [Complementary PWM Output (Hardware Deadband)] section to set the complementary PWM output action
- b) Set the state of the PWM port (PCONR.EMBVALA, PCONR.EMBVAB) when an EMB event occurs (select the appropriate protection state depending on the system application)
- c) Set the relevant registers of the EMB module (including EMB interrupt permit register (EMB_INTEN0) EMB control register 0 (EMB_CTL0), etc.)
- d) Start counter (GCONR.START=1) the EMB module monitors the system status in real time

18.3.15 Function Summary Table

A summary of the main functions of Timer6 in sawtooth mode and triangle wave A and B modes is shown in the table below.

Table 18-3 Comparison of functions in different modes

PWM output function			Sawtoo th Wave	Triangular wave		Corresponding register setting (X = A, B)	Re mar ks
				Triang ular wave A	Triang ular wave B		
Inde pend ent PWM Output	Port Status Control	Port output setting at startup	Support	Support	Support	PCONR.STACX PCONR.STASTPSX	
		Port output setting when stopped	Support	Support	Support	PCONR.STOPCX PCONR.STASTPSX	
		Port output setting when comparing matches	Support	Support	Support	PCONR.CMPCX	
		Port output setting during cycle matching	Support	Support	Support	PCONR.PERCX	
Cache Trans fer	Cycl e Bas e Valu e	Sing le Ca che	Support	Support	Support	Control bit: BCONR. Baseline values: PERAR, PERBR	
		Dual Ca che	Support	Support	Support	Control bit: BCONR.BSEP Base value: PERAR, PERBR, PERCR	
	Com pare ben chm ark valu	Sing le Ca che	Support	Support	Support	Control bit: BCONR. Baseline values: GCMAR, GCMCR GCMBR, GCMDR	Different cache transfer points for triangular wave A mode and triangular
		Dual Ca che	Support	Support	Support	Control bit: BCONR.BSEX Base value: GCMAR, GCMCR, GCMER GCMBR, GCMDR, GCMFR	

		es						wave B mode
		Emergency braking			Support	Support	Support	PCONR.EMBVALX
Complementary PWM Output	Port Status Control	Port output setting at startup			Support	Support	Support	PCONR.STACX PCONR.STASTPSX
		Port input when stopped		Support	Support	Support	PCONR.STACX PCONR.STASTPSX	
		Out Settings		Support	Support	Support	PCONR.CMPCX	
		Compare Matching Time Ends		Support	Support	Support		
		Port output setting		Support	Support	Support	PCONR.PERCX	
	Cache Transmission	Periodicity	Single	Support	Support	Support	Control bit: BCONR. Baseline values: PERAR, PERBR	
		Baseline	slow Deposit					

PWM output function			Sawtooth Wave	Triangular wave		Corresponding register setting (X = A, B)	Remarks
				Triangular wave A	Triangular wave B		
Comparison between channels and values	Compare between channels and values	Value	Dual Cache	Support	Support	Support	Control bit: BCONR.BSEP Base value: PERAR, PERBR, PERC
		Compare between channels and values	Single Cache	Support	Support	Support	Control bit: BCONR. Base value: GCMAR, GCMCR GCMBR, GCMDR
		Compare between channels and values	Dual Cache	Support	Support	Support	Control bit: BCONR.BSEX Base value: GCMAR, GCMCR, GCMER GCMBR, GCMDR, GCMFR
	Deadband reference value	Deadband reference value	Single Cache	Support	Support	Support	Control bit: DCONR.DTBENU DCONR.DTBEND Baseline values: DTUAR, DTDAR DTUBR, DTDBR
		Deadband-free PWM output		Support	Support	Support	GCMAR=GCMBR
	PWM output with deadband	Software Mode	Support	Support	Support	GCMAR ≠ GCMBR	
		Hardware method	Not supported	Support	Support	Control bit: DCONR.DTCEN Base value: GCMAR, DTUAR, DTDAR	
	Emergency braking		Support	Support	Support	PCONR.EMBVALX	

18.4 Interrupts and event descriptions

18.4.1 Interrupt output

Timer6 contains 6 general purpose count compare match interrupts (including 2 capture input interrupts) 2 dedicated count compare match interrupts, 2 count cycle match interrupts, and 1 dead time error interrupt.

18.4.1.1 Counting comparison match interrupts

There are 6 General Comparison Reference Registers (GCMAR-GCMFR), which can be compared with the count value to generate a comparison match. CMAF~STFLR.CMFF bits in the Status Flag Register (STFLR) are set to 1. If the corresponding bits in INTENA~INTENF of the Interrupt Control Register (ICONR) are set to 1 to enable the interrupt, the corresponding interrupt request is set to 1.

(TMR6_U<t>_GCMA~F) will also be triggered.

The capture input action occurs when the capture input valid condition selected by the hardware capture event selection registers (HCPAR/HCPBR) is generated. If the INTENA or INTENB bit of the interrupt control register (ICONR) is set to 1 to enable the interrupt at this time, the corresponding interrupt request (TMR6_U<t>_GCMA~B) is triggered.

The two dedicated comparison reference registers (SCMAR-SCMBR) can also be used to generate a comparison match with the count value comparison respectively. If the corresponding bit in INTENSAU<D> or INTENSBU<D> of the Interrupt Control Register (ICONR) is set to 1 to enable the interrupt, the corresponding interrupt request (TMR6_U<t>_SCMA~B) will also be triggered.

18.4.1.2 Counting cycle matching interrupts

The STFLR.OVFF or STFLR_UDFF bit of the Status Flag Register (STFLR) is set to 1 when the Ramp count reaches the upper overflow point, the Ramp decrement count reaches the lower overflow point, the Triangle count reaches the valley point, or the Triangle count reaches the peak point. INTENUDF bit of the interrupt control register (ICONR) is set to enable the interrupt, the count cycle matching interrupt (TMR6_U<t>_GOVF and TMR6_U<t>_GUDF) can be triggered at the corresponding time point.

18.4.1.3 Dead time error interrupt

If the value of the Dead Time Base Register (DTU<D>AR) is loaded into the General Comparison Base Register (GCMBR), a dead time error is generated if the cycle limit is exceeded, and the STFLR_DTEF bit in the Status Flag Register (STFLR) is set to 1. If the INTENDTE bit in the Interrupt Control Register (ICONR) is set to enable the interrupt, a dead time error interrupt is triggered at that time.

bit of the interrupt control register (ICONR) to enable the interrupt, the dead time error interrupt will be triggered at that time.

(TMR6_U<t>_GDTE)

18.4.2 Event Output

During the clock counting process, if a period matching event (upflow and downflow points of sawtooth wave, counting peak or valley points of triangle wave) a general counting comparison matching event, or a special counting comparison matching event is generated, the corresponding event output signal will be generated to select to trigger another module.

The following figure shows the action examples of the general-purpose compare match interrupt A-F & event A-F, dedicated compare match interrupt A-B & event A-B, and cycle match interrupt & event for Unit 1.

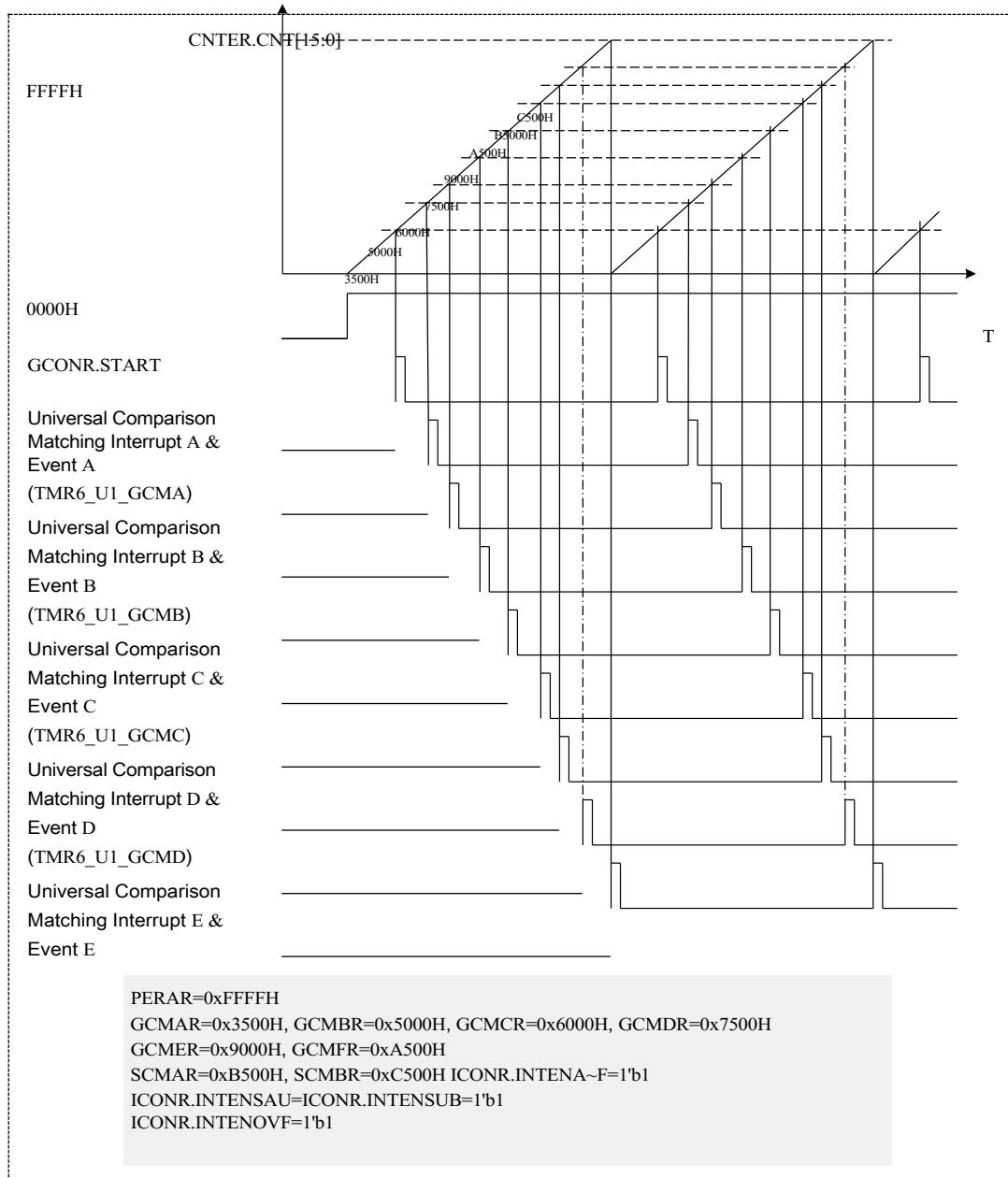


Figure 18-30 Example of interrupt & event output during sawtooth wave mode

18.5 Register Description

Table 18-4 shows the list of registers for the Timer6 module.

BASE ADDR_ 0x4001_8000 (U1) 0x4001_8400 (U2) 0x4001_8800 (U3)

Table 18-4 Register List

Register Name	Sym bols	Offset	Bit width	Reset value
General purpose count value register	TMR6_CNTER	0x0000	32	0x0000_0000
General Cycle Reference Value Register A	TMR6_PERAR	0x0004	32	0x0000_FFFF
General cycle reference value register B	TMR6_PERBR	0x0008	32	0x0000_FFFF
General cycle reference value register C	TMR6_PERCR	0x000C	32	0x0000_FFFF
General comparison reference value register A	TMR6_GCMAR	0x0010	32	0x0000_FFFF
General comparison reference value register B	TMR6_GCMBR	0x0014	32	0x0000_FFFF
General comparison reference value register C	TMR6_GCMCR	0x0018	32	0x0000_FFFF
General comparison reference value register D	TMR6_GCMDR	0x001C	32	0x0000_FFFF
General comparison reference value register E	TMR6_GCMER	0x0020	32	0x0000_FFFF
General comparison reference value register F	TMR6_GCMFR	0x0024	32	0x0000_FFFF
Dedicated comparison reference value register A	TMR6_SCMAR	0x0028	32	0x0000_FFFF
Dedicated comparison reference value register B	TMR6_SCMBR	0x002C	32	0x0000_FFFF
Dedicated comparison reference value register C	TMR6_SCMCR	0x0030	32	0x0000_FFFF
Dedicated comparison reference value register D	TMR6_SCMDR	0x0034	32	0x0000_FFFF
Dedicated comparison reference value register E	TMR6_SCMER	0x0038	32	0x0000_FFFF
Dedicated comparison reference value register F	TMR6_SCMFR	0x003C	32	0x0000_FFFF
Dead time reference value register UA	TMR6_DTUAR	0x0040	32	0x0000_FFFF
Dead time reference value register DA	TMR6_DTDAR	0x0044	32	0x0000_FFFF
Dead time reference value register UB	TMR6_DTUBR	0x0048	32	0x0000_FFFF

Dead time reference value register DB	TMR6_DTDBR	0x004C	32	0x0000_FFFF
General Control Register	TMR6_GCONR	0x0050	32	0x0000_0100
Interrupt control register	TMR6_ICONR	0x0054	32	0x0000_0000
Port Control Register	TMR6_PCONR	0x0058	32	0x0000_0000
Cache Control Register	TMR6_BCONR	0x005C	32	0x0000_0000
Deadband control register	TMR6_DCONR	0x0060	32	0x0000_0000
Filter control register	TMR6_FCONR	0x0068	32	0x0000_0000
Effective period register	TMR6_VPERR	0x006C	32	0x0000_0000
Status Flag Register	TMR6_STFLR	0x0070	32	0x8000_0000
Hardware boot event selection register	TMR6_HSTAR	0x0074	32	0x0000_0000
Hardware stop event selection register	TMR6_HSTPR	0x0078	32	0x0000_0000
Hardware Clear Event Select Register	TMR6_HCLRR	0x007C	32	0x0000_0000
Hardware capture event selection register A	TMR6_HCPAR	0x0080	32	0x0000_0000

Register Name	Sym bols	Offset	Bit width	Reset value
Hardware capture event selection register B	TMR6_HCPBR	0x0084	32	0x0000_0000
Hardware recursive event selection register	TMR6_HCUPR	0x0088	32	0x0000_0000
Hardware decrement event selection register	TMR6_HCDOR	0x008C	32	0x0000_0000
Software synchronization start control register	TMR6_SSTAR	(0x4001_83F4)	32	0x0000_0000
Software synchronization stop control register	TMR6_SSTPR	(0x4001_83F8)	32	0x0000_0000
Software synchronous zero control register	TMR6_SCLRR	(0x4001_83FC)	32	0x0000_0000

Caution:

-The software synchronization registers (TMR6_SSTAR, TMR6_SSTPR, TMR6_SCLRR) are 3 unit-independent registers, common to all 3 units of Timer6.

18.5.1 General purpose count value register (TMR6_CNTER)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CNT [15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Reads out as "0"	R
b15~b0	CNT [15:0]	Counting value	Current timer count value	R/W

18.5.2 General Periodic Reference Value Register (TMR6_PERAR-PERCR)

Reset value: 0x0000_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PERA-C [15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Reads out as "0"	R
b15~b0	PERA-C [15:0]	Counting cycle value	Set the count cycle value and corresponding cache value for each round of counting	R/W

18.5.3 General Comparison Reference Value Register (TMR6_GCMAR-GCMFR)

Reset value: 0x0000_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
GCMA-F[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Reads out as "0"	R
b15~b0	GCMA-F[15:0]	Counting comparison benchmark values	Comparison reference value setting, matching signal valid when equal to the count value	R/W

18.5.4 Dedicated comparison reference value register (TMR6_SCMAR-SCMFR)

Reset value: 0x0000_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SCMA-F[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Reads out as "1**"	R
b15~b0	SCMA-F[15:0]	Dedicated comparison benchmark values	Set comparison reference value and cache value	R/W

18.5.5 Dead Time Reference Register (TMR6_DTU<D>AR)

Reset value: 0x0000_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
dtua-b[15:0]/dtda-b[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Reads out as "1**"	R
b15~b0	DTU/DA-B[15:0]	Dead time value	Dead time setting value and cache value	R/W

18.5.6 General Control Register (TMR6_GCONR)

Reset value: 0x0000_0100

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	ZMSK VAL[1:0]	ZMSK POS	ZMSK REV	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	DIR	-	CKDIV[2:0]		MODE[2:0]		START		

position	Marker	Place Name	Function	Reading and writing
b31~b20	Reserved	-	Read "0", write "0" when writing	R/W
b19~b18	ZMSKVAL[1:0]	Number of Z-phase input selection	Count cycle value of orthogonal phase input masked 00: Z-shield cycles 01: Z-phase input is masked for 4 count cycles after position count overflow or underflow 10: Z-phase input is masked for 8 count cycles after position count overflow or underflow 11: Z-phase input within 16 count cycles after position count overflow or underflow is screened shield	R/W
b17	ZMSKPOS	Z-phase input position timer selection	0: The unit acts as a position timer during the Z-phase input, and the position timer clear function operates normally during the shield cycle. 1: This unit acts as a position timer at the Z-phase input, and the position is fixed during the shield cycle. Timer clear function is blocked	R/W
b16	ZMSKREV	Z-phase input rotation timer selection	0: The unit acts as a revolution timer during the Z-phase input, and the revolution timer counting function operates normally during the shield cycle. 1: This unit acts as a rotation timer at the Z-phase input, and the rotation is fixed during the shield cycle. Timer counting function is blocked	R/W
b15~b9	Reserved	-	Read "0", write "0" when writing	R/W
b8	DIR	Counting direction	0: decreasing count 1: Incremental counting	R/W
b7	Reserved	-	Read "0", write "0" when writing	R/W
b6~b4	CKDIV[2:0]	Counting clock selection	000: PCLK0 001: PCLK0/2 010 : PCLK0/4 011: PCLK0/8 100: PCLK0/16 101: PCLK0/64 110: PCLK0/256 111: PCLK0/1024	R/W
b3~b1	MODE[2:0]	Counting Mode	000: Sawtooth wave mode 100: Delta wave A mode 101: Triangle wave B mode please do	R/W

not set other values

b0	START	Timer start	0: Timer off 1: Timer start Note: This bit will automatically change to 0 when a software stop condition or hardware stop condition is in effect	R/W
----	-------	-------------	--	-----

18.5.7 Interrupt Control Register (TMR6_ICONR)

Reset value: 0x0000_0000

b3 1	b3 0	b2 9	b2 8	b2 7	b2 6	b2 5	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b1 5	b1 4	b1 3	b1 2	b1 1	b1 0	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	INTE N DTE	INTE N UDF	INTE N OVF	INTE N F	INTE N E	INTE N D	INTE N C	INTE N B	INTE N A

position	Marker	Place Name	Function	Reading and writing
b31~b20	Reserved	-	Read "0", write "0" when writing	R/W
b19	INTENSBD	Dedicated Count Down Interrupt Enable B	0: The interrupt is invalid when the SCMBR register and the count value are equal during down counting 1: This interrupt is enabled when the SCMBR register and the count value are equal during the count down period Note: This bit is also enabled when output as a dedicated compare match event	R/W
b18	INTENSBU	Dedicated count-up interrupt enable B	0: The interrupt is invalid when the SCMBR register and the count value are equal during up counting 1: This interrupt is enabled when the SCMBR register and the count value are equal during the up count Note: This bit is also enabled when output as a dedicated compare match event	R/W
b17	INTENSAD	Dedicated Count Down Interrupt Enable A	0: The interrupt is invalid when the SCMAR register and the count value are equal during down counting 1: This interrupt is enabled when the SCMAR register and the count value are equal during the count down period Note: This bit is also enabled when output as a dedicated compare match event	R/W
b16	INTENSAU	Dedicated count-up interrupt enable A	0: The interrupt is invalid when the SCMAR register and the count value are equal during up counting 1: This interrupt is enabled when the SCMAR register and the count value are equal during up-counting Note: This bit is also enabled when output as a dedicated compare match event	R/W
b15~b9	Reserved	-	Read "0", write "0" when writing	R/W
b8	INTENDTE	Dead time error interrupt enable	0: The interrupt is invalid when the dead time is wrong 1: This interrupt is enabled when the dead time is wrong	R/W
b7	INTENUDF	Underflow interrupt enable	0: The interrupt is invalid when the underflow occurs during sawtooth wave or when the count reaches the valley point during delta wave 1: The interrupt is enabled when the underflow occurs during sawtooth wave or when the count reaches the valley point during delta wave	R/W
b6	INTENOVF	Overflow interrupt enable	0: The interrupt is invalid when the overflow occurs during sawtooth wave or when the count reaches the peak point during delta wave 1: The interrupt is enabled when the overflow occurs during	R/W

			sawtooth wave or when the count reaches the peak point during delta wave	
b5	INTENF	Count Match Interrupt Enable F	0: If the GCMFR register is equal to the count value, the interrupt is invalid 1: The interrupt is enabled when the GCMFR register is equal to the count value	R/W
b4	INTENE	Count Match Interrupt Enable E	0: If the GCMER register is equal to the count value, the interrupt is invalid 1: The interrupt is enabled when the GCMER register is equal to the count value	R/W
b3	INTEND	Count Match Interrupt Enable D	0: If the GCMDR register is equal to the count value, the interrupt is invalid 1: The interrupt is enabled when the GCMDR register is equal to the count value	R/W
b2	INTENC	Count Match Interrupt Enable C	0: If the GCMCR register is equal to the count value, the interrupt is invalid 1: When the GCMCR register is equal to the count value, the interrupt is enabled	R/W
b1	INTENB	Count Match Interrupt Enable B	0: The interrupt is invalid when the GCMBR register is equal to the count value, or when a capture input event occurs 1: When the GCMBR register is equal to the count value, or when a capture input event occurs, the Interrupt Enable	R/W

b0	INTENA	Count Match Interrupt Enable A	0: The interrupt is invalid when the GCMAR register is equal to the count value, or when a capture input event occurs 1: When the GCMAR register is equal to the count value, or when a capture input event occurs, the Interrupt Enable	R/W
----	--------	-----------------------------------	---	-----

18.5.8 Port Control Register (TMR6_PCONR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	EMBVAL B[1:0]	-	-	OUT ENB	PER CB[1:0]	CMP CB[1:0]	STASTP SB	STP CB	STA CB	CAP MDB			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	EMBVAL A[1:0]	-	-	OUT ENA	PER CA[1:0]	CMP CA[1:0]	STASTP SA	STP CA	STA CA	CAP MDA			

position	Marker	Place Name	Function	Reading and writing
b31~b29	Reserved	-	Read "0", write "0" when writing	R/W
b28~b27	EMBVALB[1:0]	EMB Status Control B 1	00: TIM6_<gt>_PWMB port outputs normally when EMB condition is established 01: TIM6_<gt>_PWMB port outputs high resistance state when EMB condition is established 10: TIM6_<gt>_PWMB port outputs low when EMB condition is established 11: TIM6_<gt>_PWMB port outputs high when EMB condition is established	R/W
b26~b25	Reserved	-	Read "0", write "0" when writing	R/W
b24	OUTENB	Output enable B	0: Invalid TIM6_<gt>_PWMB port output for Timer6 function 1: TIM6_<gt>_PWMB port output is valid for Timer6 function	R/W
b23~b22	PERCB[1:0]	Port status setting when cycle value matches B	00: TIM6_<gt>_PWMB port output is set low when the timer count value is equal to the cycle value 01: TIM6_<gt>_PWMB port output is set high when the timer count value is equal to the cycle value 10: When the timer count value is equal to the cycle value, the TIM6_<gt>_PWMB port output remains in the previous state 11: TIM6_<gt>_PWMB port output when the timer count value is equal to the cycle value Set to invert level	R/W
b21~b20	CMPCB[1:0]	Port status setting B when comparison values match	00: When the timer count value is equal to GCMBR, the TIM6_<gt>_PWMB port output is set low 01: When the timer count value is equal to GCMBR, the TIM6_<gt>_PWMB port output is set high 10: When the timer count value is equal to GCMBR, the TIM6_<gt>_PWMB port output remains in the previous state 11: When the timer count value is equal to GCMBR, the TIM6_<gt>_PWMB port loses Out set to invert level	R/W
b19	STASTPSB	Count Start Stop Port Status Selection B	0: When counting starts or stops, TIM6_<gt>_PWMB port output is determined by STACB, STPCB 1: TIM6_<gt>_PWMB port output keeps previous state when counting starts or stops Note: Count start here means initial count start or stop and start again; count stop is Refers to stop at the beginning or stop after the count starts	R/W

b18	STPCB	Counting stop port status setting B	0: TIM6_<t>_PWMB port output is set to low when counting is stopped 1: TIM6_<t>_PWMB port output is set to high when counting is stopped	R/W
b17	STACB	Counting start port status setting B	0: TIM6_<t>_PWMB port output is set to low when counting starts 1: TIM6_<t>_PWMB port output is set to high when counting starts	R/W
b16	CAPMDB	Function mode selection B	0: Comparison output function 1: Capture input function	R/W
b15~b13	Reserved	-	Read "0", write "0" when writing	R/W
b12~b11	EMBVALA[1:0]	EMB Status Control A	00: TIM6_<t>_PWMA port outputs normally when EMB condition is established 01: TIM6_<t>_PWMA port outputs high resistance state when EMB condition is established 10: TIM6_<t>_PWMA port outputs low when EMB condition is established 11: TIM6_<t>_PWMA port outputs high when EMB condition is established	R/W
b10~b9	Reserved	-	Read "0", write "0" when writing	R/W
b8	OUTENA	Output enable A	0: Invalid TIM6_<t>_PWMA port output for Timer6 function 1: TIM6_<t>_PWMA port output is valid for Timer6 function	R/W
b7~b6	PERCA[1:0]	Port status setting when cycle value matches A	00: TIM6_<t>_PWMA port output is set low when the timer count value is equal to the cycle value 01: TIM6_<t>_PWMA port output is set high when the timer count value is equal to the cycle value 10: When the timer count value is equal to the cycle value, the TIM6_<t>_PWMA port output remains in the previous state 11: TIM6_<t>_PWMA port output when the timer count value is equal to the cycle value Set to invert level	R/W
b5~b4	CMPCA[1:0]	Port status setting A when the comparison value matches	00: When the timer count value is equal to GCMAR, the TIM6_<t>_PWMA port output is set low 01: TIM6_<t>_PWMA port output is set high when the timer count value is equal to GCMAR 10: When the timer count value is equal to GCMAR, the TIM6_<t>_PWMA port output remains in the previous state 11: When the timer count value is equal to GCMAR, the TIM6_<t>_PWMA port loses Out set to invert level	R/W
b3	STASTPSA	Count Start Stop Port Status Selection A	0: When counting starts or stops, TIM6_<t>_PWMA port output is determined by STACA, STPCA 1: TIM6_<t>_PWMA port output keeps previous state when counting starts or stops Note: Count start here means initial count start or stop and start again; count stop is Refers to stop at the beginning or stop after the count starts	R/W
b2	STPCA	Counting stop port status Set A	0: TIM6_<t>_PWMA port output is set to low when counting is stopped 1: TIM6_<t>_PWMA port output is set to high when counting is stopped	R/W
b1	STACA	Counting start port status	0: TIM6_<t>_PWMA port output is set to low when counting starts 1: TIM6_<t>_PWMA port output is set to high when counting starts	R/W

Set A

b0	CAPMDA	Function mode selection A	0: Comparison output function 1: Capture input function	R/W
----	--------	------------------------------	--	-----

18.5.9 Cache Control Register (TMR6_BCONR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	BTRSPB[1:0]	-	-	BSE SPB	BEN SPB	-	-	BTRSPA[1:0]	-	-	BSE SPA	BEN SPA		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	BSE P	BEN P	-	-	-	-	BSE B	BEN B	BSE A	BEN A

position	Marker	Place Name	Function	Reading and writing
b31~b30	Reserved	-	Read '0', write "0" when writing	R/W
b29~28	BTRSPB[1:0]	Dedicated comparison reference value cache transfer time point B	Sawtooth wave when 00: not transmitted Beyond 00: When transmitting triangular waves during upflow or downflow 00: No transmission 01: Cache transfer when counting to the peak 10: Cache transfer when the count reaches the valley point 11: Cache transfer when counting to peak or trough	R/W
b27-b26	Reserved	-	Read '0', write "0" when writing	R/W
b25	BSESPB	Dedicated comparison reference value cache transfer option B	0: Single cache transfer (SCMDR->SCMBR) 1: Dual cache transfer (SCMFR->SCMDR->SCMBR)	R/W
b24	BENSPB	Dedicated comparison baseline value cache transfer B	0: Cache transfer invalid 1: Cache transfer enable	R/W
b23~b22	Reserved	-	Read '0', write "0" when writing	R/W
b21~b20	BTRSPA[1:0]	Dedicated comparison reference value cache transfer time point A	Sawtooth wave when 00: not transmitted Beyond 00: When transmitting triangular waves during upflow or downflow 00: No transmission 01: Cache transfer when counting to the peak 10: Cache transfer when the count reaches the valley point 11: Cache transfer when counting to peak or trough	R/W
b19~b18	Reserved	-	Read '0', write "0" when writing	R/W
b17	BSESPA	Dedicated comparison	0: Single cache transfer (SCMCR->SCMAR) 1: Dual cache transfer (SCMER->SCMCR->SCMAR)	R/W

		reference value cache transfer option A		
b16	BENSPA	Dedicated comparison reference value cache transfer A	0: Cache transfer invalid 1: Cache transfer enable	R/W
b15~b10	Reserved	-	Read "0", write "0" when writing	R/W
b9	BSEP	Periodic value cache transfer selection	0: Single cache transfer (PERBR->PERAR) 1: Dual cache transfer (PERCR->PERBR->PERAR) Note: The transmission time point is not related to the counting mode, but only the upper overflow point of the sawtooth wave, the lower overflow point or the The trough of a triangular wave	R/W
b8	BENP	Periodic value cache transfer	0: Cache transfer invalid 1: Cache transfer enable	R/W
b7~b4	Reserved	-	Read "0", write "0" when writing	R/W

b3	BSEB	Universal	When comparing output functions:	R/W
		Comparison	0: Single cache transfer (GCMDR->GCMBR)	
		Value Cache	1: Dual cache transfer (GCMFR->GCMDR->GCMBR)	
		Transfer Option	When capturing the input function:	
		B	0: Single cache transfer (GCMBR->GCMDR) 1: Dual cache transfer (GCMBR->GCMDR->GCMFR)	
b2	BENB	Universal	0: Cache transfer invalid	R/W
		Comparison	1: Cache transfer enable	
		Value Cache		
		Transfer B		
b1	BSEA	Universal	When comparing output functions:	R/W
		Comparison	0: Single cache transfer (GCMCR->GCMAR)	
		Value Cache	1: Dual cache transfer (GCMER->GCMCR->GCMAR)	
		Transfer Option	When capturing the input function:	
		A	0: Single cache transfer (GCMAR->GCMCR) 1: Dual cache transfer (GCMAR->GCMCR->GCMER)	
b0	BENA	Universal	0: Cache transfer invalid	R/W
		Comparison	1: Cache transfer enable	
		Value Cache		
		Transfer A		

18.5.10 Deadband Control Register (TMR6_DCONR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved							SEPA	-	-	DTB END	DTB ENU	-	-	-	DTC EN
<hr/>															
position	Marker	Place Name	Function	Reading and writing											
b31~b9	Reserved	-	Read "0", write "0" when writing	R/W											
b8	SEPA	Separation settings	0: DTUAR and DTDAR are set separately 1: The value of DTDAR and the value of DTUAR are automatically equal	R/W											
b7~b6	Reserved	-	Read "0", write "0" when writing	R/W											
b5	DTBEND	Dead time value cache transfer D	0: Cache transfer invalid 1: Cache transfer enable (DTDBR->DTDAR)	R/W											
b4	DTBENU	Dead time value cache transfer U	0: Cache transfer invalid 1: Cache transfer enable (DTUBR->DTUAR)	R/W											
b3~b1	Reserved	-	Read "0", write "0" when writing	R/W											
b0	DTCEN	Dead zone function	0: Dead zone function is invalid 1: Dead zone function is effective	R/W											

18.5.11 Filter Control Register (TMR6_FCONR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved								NOFI CKTB[1:0]	NOFI ENTB	-	NOFI CKTA [1:0]	NOFI ENTA			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								NOFI CKGB[1:0]	NOFI ENGB	-	NOFI CKGA[1:0]	NOFI ENGA			

position	Marker	Place Name	Function	Reading and writing
b31~b23	Reserved	-	Read "0", write "0" when writing	R/W
b22~b21	NOFICKTB[1:0]	Filter sampling reference clock selection TB	00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64	R/W
b20	NOFIENTB	Capture input port filtering TB	0: TIM6_TRIGB port input filtering function is invalid 1: TIM6_TRIGB port input filtering function is enabled	R/W
b19	Reserved	-	Read "0", write "0" when writing	R/W
b18~b17	NOFICKTA[1:0]	Filter sampling reference clock selection TA	00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64	R/W
b16	NOFIENTA	Capture input port filter TA	0: TIM6_TRIGA port input filtering function is invalid 1: TIM6_TRIGA port input filtering function is enabled	R/W
b15~b7	Reserved	-	Read "0", write "0" when writing	R/W
b6~b5	NOFICKGB[1:0]	Filter sampling reference clock selection GB	00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64	R/W
b4	NOFIENGB	Capture input port filtering GB	0: The unit TIM6_<t>_PWMB input port filtering function is invalid 1: The unit TIM6_<t>_PWMB input port filtering function is enabled	R/W
b3	Reserved	-	Read "0", write "0" when writing	R/W
b2~b1	NOFICKGA[1:0]	Filter sampling reference clock selection GA	00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64	R/W
b0	NOFIENGA	Capture input port filtering GA	0: This unit TIM6_<t>_PWMA input port filtering function is invalid 1: The unit TIM6_<t>_PWMA input port filtering function is enabled	R/W

18.5.12 Valid Period Register (TMR6_VPERR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
Reserved										PCNTS [2:0]	PCNTE[1:0]					
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
Reserved					SP PERIB	SP PERIA	Reserved									

position	Marker	Place Name	Function	Reading and writing
b31~b21	Reserved	-	Read 0, write "0" when writing	R/W
b20~b18	PCNTS [2:0]	Effective period selection	000: Valid period selection function is invalid 001: Valid every 1 cycle 010: Valid every 2 cycles 011: Valid every 3 cycles 100: Valid every 4 cycles 101: Valid every 5 cycles 110: Valid every 6 cycles 111: Valid every 7 cycles	R/W
b17~b16	PCNTE[1:0]	Effective cycle counting condition selection	00: Valid period selection function is invalid 01: Sawtooth wave counting upper and lower overflow points or triangle wave valley as counting conditions 10: Sawtooth wave counting upper and lower overflow points or triangular wave crest as counting conditions 11: Sawtooth wave counting upper and lower overflow points or triangular wave valley and wave peak as counting conditions	R/W
b15~b10	Reserved	-	Read 0, write "0" when writing	R/W
b9	SPPERIB	Dedicated signal effective period selection B	0: Valid period selection function is invalid 1: Valid period selection function is enabled	R/W
b8	SPPERIA	Dedicated signal effective period selection A	0: Valid period selection function is invalid 1: Valid period selection function is enabled	R/W
b7~b0	Reserved	-	Read 0, write "0" when writing	R/W

18.5.13 Status Flag Register (TMR6_STFLR)

Reset value: 0x8000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DIRF	Reserved								VPERNUM[2:0]		Reserved				
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	CMS BDF	CMS BUF	CMS ADF	CMS AUF	DTE F	UDF F	OVF F	CMF F	CME F	CMD F	CMC F	CMB F	CMA F
<hr/>															
position	Marker	Place Name			Function								Reading and writing		
b31	DIRF	Counting direction			0: decreasing count 1: Incremental counting								R		
b30~b24	Reserved	-			Read "0", write "0" when writing								R		
b23~b21	VPERNUM[2:0]	Number of cycles			The number of cycles after counting when the effective cycle selection function is enabled								R		
b20~b13	Reserved	-			Read "0", write "0" when writing								R		
b12	CMSBDF	Count down dedicated comparison base value match B			0: When counting down, the value of SCMBR register is not equal to the count value 1: When counting down, the value of SCMBR register is equal to the count value								R/W		
b11	CMSBUF	Upward Counting Dedicated Comparison Base Value Match B			0: When counting up, the value of SCMBR register is not equal to the count value 1: When counting up, the value of SCMBR register is equal to the count value								R/W		
b10	CMSADF	Downward Counting Dedicated Comparison Base Value Match A			0: When counting down, the value of SCMAR register is not equal to the count value 1: When counting down, the value of SCMAR register is equal to the count value								R/W		
b09	CMSAUF	Upward Counting Dedicated Comparison Base Value Match A			0: When counting up, the value of SCMAR register is not equal to the count value 1: When counting up, the value of SCMAR register is equal to the count value								R/W		
b8	DTEF	Dead time error			0: No dead time error occurred 1: Dead time error occurred								R		
b7	UDFF	Underflow matching			0: No sawtooth wave underflow or delta wave count to valley point 1: Sawtooth wave underflow or delta wave count to the valley occurs								R/W		
b6	OVFF	Overflow matching			0: No sawtooth wave overflow or delta wave count to peak point 1: Occurrence of sawtooth wave overflow or triangular wave counting to the peak point								R/W		
b5	CMFF	Counting Match F			0: The value of GCMFR register is not equal to the count value 1: The value of GCMFR register is equal to the count value								R/W		
b4	CMEF	Counting Match E			0: The value of GCMER register is not equal to the count value 1: The value of GCMER register is equal to the count value								R/W		
b3	CMDF	Counting Match D			0: The value of GCMDR register is not equal to the count value 1: The value of GCMDR register is equal to the count value								R/W		

b2	CMCF	Counting Match C	0: The value of GCMCR register is not equal to the count value 1: The value of GCMCR register is equal to the count value	R/W
b1	CMBF	Counting Match B	0: The value of the GCMBR register is not equal to the count value and no TIM6_<1>_PWMB capture completion action has occurred 1: The value of GCMBR register is equal to the count value, or occurs TIM6_<1>_PWMB Capture Completion Action	R/W
b0	CMAF	Counting Match A	0: The value of the GCMAR register is not equal to the count value and no TIM6_<1>_PWMA capture completion action has occurred 1: The value of GCMAR register is equal to the count value, or the occurrence of TIM6_<1>_PWMA Capture Completion Action	R/W

18.5.14 Hardware Boot Event Select Register (TMR6_HSTAR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
STA	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HSTA 11	HSTA 10	HSTA 9	HSTA 8	HSTA 7	HSTA 6	HSTA 5	HSTA 4	-	-	HSTA 1	HSTA 0

position	Marker	Place Name	Function	Reading and writing
b31	STARTS	Hardware Boot Enable	0: Hardware boot is invalid 1: Hardware boot is valid Note: The SSTAR setting is not valid when the hardware boot is valid	R/W
b30~b12	Reserved	-	Read "1", write "0" when writing	R/W
b11	HSTA11	Hardware start-up conditions11	Condition: TIM6_TRIGB port up-sampled to falling edge 0: Hardware start invalid when condition matched 1: Hardware start is valid when the conditions are matched	R/W
b10	HSTA10	Hardware startup conditions 10	Condition: TIM6_TRIGB port is sampled to rising edge 0: Hardware start is invalid when condition is matched 1: Hardware start is valid when the conditions are matched	R/W
b9	HSTA9	Hardware start-up conditions9	Condition: TIM6_TRIGA port up-sampled to falling edge 0: Hardware start invalid when condition matched 1: Hardware start is valid when the conditions are matched	R/W
b8	HSTA8	Hardware startup conditions8	Condition: TIM6_TRIGA port sampled to rising edge 0: Hardware start invalid when condition matched 1: Hardware start is valid when the conditions are matched	R/W
b7	HSTA7	Hardware startup conditions7	Condition: TIM6_PWMB port up-sampled to falling edge 0: Hardware start invalid when condition matched 1: Hardware start is valid when the conditions are matched	R/W
b6	HSTA6	Hardware start-up conditions6	Condition: TIM6_PWMB port sampled to rising edge 0: Hardware start invalid when condition matched 1: Hardware start is valid when the conditions are matched	R/W
b5	HSTA5	Hardware start-up conditions5	Condition: TIM6_PWMA port up-sampled to falling edge 0: Hardware start invalid when condition matched	R/W

1: Hardware start is valid when the conditions are matched

b4	HSTA4	Hardware start-up condition 4	Condition: TIM6_<>_PWMA port up-sampled to rising edge 0: Hardware start invalid when condition matched 1: Hardware start is valid when the conditions are matched	R/W
b3~b2	Reserved	-	Read "0", write "0" when writing	R/W
b1	HSTA1	Hardware start-up conditions1	Condition: Internal hardware trigger event 1 is valid 0: Hardware start is invalid when condition is matched 1: Hardware start is valid when the conditions are matched	R/W
b0	HSTA0	Hardware start condition 0	Condition: Internal hardware trigger event 0 is valid 0: Hardware start is invalid when condition is matched 1: Hardware start is valid when the conditions are matched	R/W

18.5.15 Hardware Stop Event Select Register (TMR6_HSTPR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
STOPS	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HSTP 11	HSTP 10	HSTP 9	HSTP 8	HSTP 7	HSTP 6	HSTP 5	HSTP 4	-	-	HSTP 1	HSTP 0

position	Marker	Place Name	Function	Reading and writing
b31	STOPS	Hardware Stop Enable	0: Hardware stop invalid 1: Hardware stop valid Note: The setting of SSTPR is invalid when hardware stop is valid	R/W
b30~b12	Reserved	-	Read "0", write "0" when writing	R/W
b11	HSTP11	Hardware stop condition 11	Condition: TIM6_TRIGB port up-sampled to falling edge 0: Hardware stop invalid when condition matches 1: Hardware stop valid when conditions match	R/W
b10	HSTP10	Hardware stop condition 10	Condition: TIM6_TRIGB port sampled to rising edge 0: Hardware stop invalid when condition matched 1: Hardware stop valid when conditions match	R/W
b9	HSTP9	Hardware stop condition 9	Condition: TIM6_TRIGA port up-sampled to falling edge 0: Hardware stop invalid when condition matches 1: Hardware stop valid when conditions match	R/W
b8	HSTP8	Hardware stop condition 8	Condition: TIM6_TRIGA port sampled to rising edge 0: Hardware stop invalid when condition matched 1: Hardware stop valid when conditions match	R/W
b7	HSTP7	Hardware stop condition 7	Condition: TIM6_PWMB port up-sampled to falling edge 0: Hardware stop invalid when condition matched 1: Hardware stop valid when conditions match	R/W
b6	HSTP6	Hardware stop condition 6	Condition: TIM6_PWMB port sampled to rising edge 0: Hardware stop invalid when condition matched 1: Hardware stop valid when conditions match	R/W
b5	HSTP5	Hardware stop condition 5	Condition: TIM6_PWMA port up-sampled to falling edge 0: Hardware stop invalid when condition matched 1: Hardware stop valid when conditions match	R/W
b4	HSTP4	Hardware stop condition 4	Condition: TIM6_PWMA port up-sampled to rising edge 0: Hardware stop invalid when condition is matched 1: Hardware stop valid when conditions match	R/W
b3~b2	Reserved	-	Read "0", write "0" when writing	R/W

b1	HSTP1	Hardware stop condition 1	Condition: Internal hardware trigger event 1 is valid 0: Hardware stop is invalid when condition is matched 1: Hardware stop valid when conditions match	R/W
b0	HSTP0	Hardware stop condition 0	Condition: internal hardware trigger event 0 is valid 0: hardware stop is invalid when condition is matched 1: Hardware stop valid when conditions match	R/W

18.5.16 Hardware Clear Event Select Register (TMR6_HCLRR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CLEARs	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HCLE 11	HCLE 10	HCLE 9	HCLE 8	HCLE 7	HCLE 6	HCLE 5	HCLE 4	-	-	HCLE 1	HCLE 0

position	Marker	Place Name	Function	Reading and writing
b31	CLEARs	Hardware Clear Enable	0: Hardware clearing is invalid 1: Hardware clearing is valid Note: The SCLRR setting is invalid when hardware clear is valid	R/W
b30~b12	Reserved	-	Read "0", write "0" when writing	R/W
b11	HCLE11	Hardware clearing condition 11	Condition: TIM6_TRIGB port up-sampled to falling edge 0: hardware clear invalid when condition matches 1: Hardware clearing is valid when the conditions are matched	R/W
b10	HCLE10	Hardware clearing condition 10	Condition: TIM6_TRIGB port sampled to rising edge 0: hardware clear invalid when condition matched 1: Hardware clearing is valid when the conditions are matched	R/W
b9	HCLE9	Hardware clearing condition 9	Condition: TIM6_TRIGA port up-sampled to falling edge 0: hardware clear invalid when condition matches 1: Hardware clearing is valid when the conditions are matched	R/W
b8	HCLE8	Hardware clearing condition 8	Condition: TIM6_TRIGA port sampled to rising edge 0: hardware clear invalid when condition matches 1: Hardware clearing is valid when the conditions are matched	R/W
b7	HCLE7	Hardware clearing condition 7	Condition: TIM6_PWMB port up-sampled to falling edge 0: Hardware clear invalid when condition matches 1: Hardware clearing is valid when the conditions are matched	R/W
b6	HCLE6	Hardware clearing condition 6	Condition: TIM6_PWMB port sampled to rising edge 0: hardware clear invalid when condition matched 1: Hardware clearing is valid when the conditions are matched	R/W
b5	HCLE5	Hardware clearing condition 5	Condition: TIM6_PWMA port up-sampled to falling edge 0: Hardware clear invalid when condition matches 1: Hardware clearing is valid when the conditions are matched	R/W

matched

b4	HCLE4	Hardware clearing condition 4	Condition: TIM6 _{<=>} PWMA port up-sampled to rising edge 0: Hardware clear invalid when condition matched 1: Hardware clearing is valid when the conditions are matched	R/W
b3~b2	Reserved	-	Read "0", write "0" when writing	R/W
b1	HCLE1	Hardware clearing condition 1	Condition: Internal hardware trigger event 1 is valid 0: Hardware clear is invalid when condition is matched 1: Hardware clearing is valid when the conditions are matched	R/W
b0	HCLE0	Hardware clear condition 0	Condition: internal hardware trigger event 0 is valid 0: hardware clear is invalid when condition is matched 1: Hardware clearing is valid when the conditions are matched	R/W

18.5.17 Hardware Capture Event Selection Register (TMR6_HCPAR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HCP A11	HCP A10	HCP A9	HCP A8	HCP A7	HCP A6	HCP A5	HCP A4	-	-	HCP A1	HCP A0

position	Marker	Place Name	Function	Reading and writing
b31~b12	Reserved	-	Read "1", write "0" when writing	R/W
b11	HCPA11	Hardware capture A condition 11	Condition: Tim6_TRIGB port sampled to falling edge 0: Hardware capture A invalid when condition matches 1: Hardware capture A is valid when the conditions match	R/W
b10	HCPA10	Hardware capture A condition 10	Condition: TIM6_TRIGB port sampled to rising edge 0: Hardware capture A invalid when condition matches 1: Hardware capture A is valid when the conditions match	R/W
b9	HCPA9	Hardware capture A condition 9	Condition: TIM6_TRIGA port sampled to falling edge 0: Hardware capture A invalid when condition matches 1: Hardware capture A is valid when the conditions match	R/W
b8	HCPA8	Hardware capture A condition 8	Condition: TIM6_TRIGA port sampled to rising edge 0: Hardware capture A invalid when condition matches 1: Hardware capture A is valid when the conditions match	R/W
b7	HCPA7	Hardware capture A condition 7	Condition: TIM6_PWMB port sampled to falling edge 0: Hardware capture A invalid when condition matches 1: Hardware capture A is valid when the conditions match	R/W
b6	HCPA6	Hardware capture A condition 6	Condition: TIM6_PWMB port sampled to rising edge 0: Hardware capture A invalid when condition matches 1: Hardware capture A is valid when the conditions match	R/W
b5	HCPA5	Hardware capture A condition 5	Condition: TIM6_PWMA port up-sampled to falling edge 0: Hardware capture A invalid when condition matches 1: Hardware capture A is valid when the conditions match	R/W

b4	HCPA4	Hardware capture A condition 4	Condition: TIM6_<>_PWMA port up-sampled to rising edge 0: Hardware capture A invalid when condition matches 1: Hardware capture A is valid when the conditions match	R/W
b3~b2	Reserved	-	Read "0", write "0" when writing	R/W
b1	HCPA1	Hardware capture A condition 1	Condition: Internal hardware trigger event 1 is valid 0: Hardware capture A is invalid when condition is matched 1: Hardware capture A is valid when the conditions match	R/W
b0	HCPA0	Hardware capture A condition 0	Condition: internal hardware trigger event 0 is valid 0: hardware capture A is invalid when condition matches 1: Hardware capture A is valid when the conditions match	R/W

18.5.18 Hardware Capture Event Select Register (TMR6_HCPBR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HCP B11	HCP B10	HCP B9	HCP B8	HCP B7	HCP B6	HCP B5	HCP B4	-	-	HCP B1	HCP B0

position	Marker	Place Name	Function	Reading and writing
b31~b12	Reserved	-	Read "0", write "0" when writing	R/W
b11	HCPB11	Hardware capture B condition 11	Condition: TIM6_TRIGB port sampled to falling edge 0: Hardware capture B invalid when condition matches 1: Hardware capture B is valid when the conditions match	R/W
b10	HCPB10	Hardware capture B condition 10	Condition: TIM6_TRIGB port sampled to rising edge 0: Hardware capture B invalid when condition matches 1: Hardware capture B is valid when the conditions match	R/W
b9	HCPB9	Hardware capture B condition 9	Condition: TIM6_TRIGA port sampled to falling edge 0: Hardware capture B invalid when condition matches 1: Hardware capture B is valid when the conditions match	R/W
b8	HCPB8	Hardware capture B condition 8	Condition: TIM6_TRIGA port sampled to rising edge 0: Hardware capture B invalid when condition matches 1: Hardware capture B is valid when the conditions match	R/W
b7	HCPB7	Hardware capture B condition 7	Condition: TIM6_ \triangleleft _PWMB port sampled to falling edge 0: Hardware capture B invalid when condition matches 1: Hardware capture B is valid when the conditions match	R/W
b6	HCPB6	Hardware capture B condition 6	Condition: TIM6_ \triangleleft _PWMB port sampled to rising edge 0: Hardware capture B invalid when condition matches 1: Hardware capture B is valid when the conditions match	R/W
b5	HCPB5	Hardware capture B condition 5	Condition: TIM6_ \triangleleft _PWMA port up-sampled to falling edge 0: Hardware capture B invalid when condition matches 1: Hardware capture B is valid when the conditions match	R/W

b4	HCPB4	Hardware capture B condition 4	Condition: TIM6_<=>PWMA port up-sampled to rising edge 0: Hardware capture B invalid when condition matches 1: Hardware capture B is valid when the conditions match	R/W
b3~b2	Reserved	-	Read "0", write "0" when writing	R/W
b1	HCPB1	Hardware capture B condition 1	Condition: Internal hardware trigger event 1 is valid 0: Hardware capture B is invalid when condition is matched 1: Hardware capture B is valid when the conditions match	R/W
b0	HCPB0	Hardware capture B condition 0	Condition: internal hardware trigger event 0 is valid 0: hardware capture B is invalid when condition matches 1: Hardware capture B is valid when the conditions match	R/W

18.5.19 Hardware recursive event selection register (TMR6_HCUPR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved														HC UP17	HC UP16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HC UP11	HC UP10	HC UP9	HC UP8	HC UP7	HC UP6	HC UP5	HC UP4	HC UP3	HC UP2	HC UP1	HC UP0

position	Marker	Place Name	Function	Reading and writing
b31~b18	Reserved	-	Read "0", write "0" when writing	R/W
b17	HCUP17	Hardware incremental condition 17	Condition: internal hardware trigger event 1 is valid 0: hardware recursion is invalid when condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W
b16	HCUP16	Hardware incremental condition 16	Condition: internal hardware trigger event 0 is valid 0: hardware recursion is invalid when condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W
b15~b12	Reserved	-	Read "0", write "0" when writing	R/W
b11	HCUP11	Hardware incremental condition 11	Condition: TIM6_TRIGB port up-sampled to falling edge 0: Hardware recursion invalid when condition matches 1: Hardware recursion is valid when the conditions are matched	R/W
b10	HCUP10	Hardware incremental condition 10	Condition: TIM6_TRIGB port up-sampled to rising edge 0: Hardware recursion invalid when condition matches 1: Hardware recursion is valid when the conditions are matched	R/W
b9	HCUP9	Hardware incremental condition 9	Condition: TIM6_TRIGA port up-sampled to falling edge 0: Hardware recursion invalid when condition matches 1: Hardware recursion is valid when the conditions are matched	R/W
b8	HCUP8	Hardware incremental condition 8	Condition: TIM6_TRIGA port up-sampled to rising edge 0: Hardware recursion invalid when condition matches 1: Hardware recursion is valid when the conditions are matched	R/W
b7	HCUP7	Hardware incremental condition 7	Condition: TIM6_<t>_PWMB port is high when TIM6_<t>_PWMA port is upsampled to the falling edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions	R/W

are matched

b6	HCUP6	Hardware incremental condition 6	Condition: TIM6_<>>_PWMB port is high when TIM6_<>>_PWMA port is upsampled to the rising edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W
b5	HCUP5	Hardware incremental condition 5	Condition: TIM6_<>>_PWMB port is low when TIM6_<>>_PWMA port is upsampled to the falling edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W
b4	HCUP4	Hardware incremental condition 4	Condition: TIM6_<>>_PWMB port is low when TIM6_<>>_PWMA port is upsampled to a rising edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W

b3	HCUP3	Hardware incremental condition 3	Condition: TIM6_<t>_PWMA port is high when TIM6_<t>_PWMB port is upsampled to the falling edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W
b2	HCUP2	Hardware incremental condition 2	Condition: TIM6_<t>_PWMA port is high when TIM6_<t>_PWMB port is upsampled to the rising edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W
b1	HCUP1	Hardware incremental condition 1	Condition: TIM6_<t>_PWMA port is low when TIM6_<t>_PWMB port is upsampled to the falling edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W
b0	HCUP0	Hardware recurrence condition 0	Condition: TIM6_<t>_PWMA port is low when TIM6_<t>_PWMB port is upsampled to a rising edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W

18.5.20 Hardware decrement event selection register (TMR6_HCDOR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved														HC DO17	HC DO16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
				HC DO11	HC DO10	HC DO9	HC DO8	HC DO7	HC DO6	HC DO5	HC DO4	HC DO3	HC DO2	HC DO1	HC DO0

position	Marker	Place Name	Function	Reading and writing
b31~b18	Reserved	-	Read "0", write "0" when writing	R/W
b17	HCDO17	Hardware decreasing condition 17	Condition: internal hardware trigger event 1 is valid 0: hardware decrement is invalid when condition is matched 1: Hardware decrement is valid when the conditions are matched	R/W
b16	HCDO16	Hardware decreasing condition 16	Condition: internal hardware trigger event 0 is valid 0: hardware decrement is invalid when condition is matched 1: Hardware decrement is valid when the conditions are matched	R/W
b15~b12	Reserved	-	Read "0", write "0" when writing	R/W
b11	HCDO11	Hardware decreasing condition 11	Condition: TIM6_TRIGB port up-sampled to falling edge 0: Hardware decrement invalid when condition matches 1: Hardware decrement is valid when the conditions are matched	R/W
b10	HCDO10	Hardware decreasing condition 10	Condition: TIM6_TRIGB port up-sampled to rising edge 0: Hardware decrement invalid when condition matches 1: Hardware decrement is valid when the conditions are matched	R/W
b9	HCDO9	Hardware decreasing condition 9	Condition: TIM6_TRIGA port up-sampled to falling edge 0: Hardware decrement invalid when condition matches 1: Hardware decrement is valid when the conditions are matched	R/W
b8	HCDO8	Hardware decreasing condition 8	Condition: TIM6_TRIGA port up-sampled to rising edge 0: Hardware decrement invalid when condition matches 1: Hardware decrement is valid when the conditions are matched	R/W
b7	HCDO7	Hardware decreasing condition 7	Condition: TIM6_<t>_PWMB port is high when TIM6_<t>_PWMA port is upsampled to the falling edge 0: hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions	R/W

are matched

b6	HCDO6	Hardware decreasing condition 6	Condition: TIM6_<t>_PWMB port is high when TIM6_<t>_PWMA port is upsampled to the rising edge 0: Hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched	R/W
b5	HCDO5	Hardware decreasing condition 5	Condition: TIM6_<t>_PWMB port is low when TIM6_<t>_PWMA port is upsampled to the falling edge 0: hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched	R/W
b4	HCDO4	Hardware decreasing condition 4	Condition: TIM6_<t>_PWMB port is low when TIM6_<t>_PWMA port is upsampled to the rising edge 0: hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched	R/W

b3	HCDO3	Hardware decreasing condition 3	Condition: TIM6_<T>_PWMA port is high when TIM6_<T>_PWMB port is upsampled to the falling edge 0: hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched	R/W
b2	HCDO2	Hardware decreasing condition 2	Condition: TIM6_<T>_PWMA port is high when TIM6_<T>_PWMB port is upsampled to the rising edge 0: hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched	R/W
b1	HCDO1	Hardware decreasing condition 1	Condition: When TIM6_<T>_PWMA port is low, TIM6_<T>_PWMB port is upsampled to the falling edge 0: Hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched	R/W
b0	HCDO0	Hardware decrement condition 0	Condition: TIM6_<T>_PWMA port is low when TIM6_<T>_PWMB port is upsampled to the rising edge 0: hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched	R/W

18.5.21 Software Synchronous Start Control Register (TMR6_SSTAR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved												SSTA3	SSTA2	SSTA1	

position	Marker	Place Name	Function	Reading and writing
b31~b3	Reserved	-	Read "1", write "0" when writing	R/W
b2	SSTA3	Module 3 software startup	0: Software startup is invalid 1: Software startup enable	R/W
b1	SSTA2	Module 2 software startup	0: Software startup is invalid 1: Software startup enable	R/W
b0	SSTA1	Module 1 software startup	0: Software startup is invalid 1: Software startup enable	R/W

18.5.22 Software Synchronization Stop Control Register (TMR6_SSTPR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved												SSTP3	SSTP2	SSTP1	

position	Marker	Place Name	Function	Reading and writing
b31~b3	Reserved	-	Read "1", write "0" when writing	R/W
b2	SSTP3	Unit 3 software stop	0: Software stop invalid 1: Software stop enable	R/W
b1	SSTP2	Unit 2 software stop	0: Software stop invalid 1: Software stop enable	R/W
b0	SSTP1	Unit 1 software stop	0: Software stop invalid 1: Software stop enable	R/W

18.5.23 Software Synchronous Clear Control Register (TMR6_SCLRR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															

position	Marker	Place Name	Function	Reading and writing
b31~b3	Reserved	-	Read "1", write "0" when writing	R/W
b2	SCLE3	Unit 3 software zeroing	0: Software clearing is invalid 1: Software zero enable	R/W
b1	SCLE2	Unit 2 software zeroing	0: Software clearing is invalid 1: Software zero enable	R/W
b0	SCLE1	Unit 1 software zeroing	0: Software clearing is invalid 1: Software zero enable	R/W

19 General Control Timer (Timer4)

19.1 Introduction

The Universal Control Timer 4 (Timer4) is a timer module for three-phase motor control, providing a variety of three-phase motor control solutions for different applications. The timer supports both triangle waveform and sawtooth waveform modes, generates various PWM waveforms, supports cache function, and supports EMB control. This series is equipped with 3 units of Timer4.

19.2 Basic Block Diagram

The basic functions and features of Timer4 are shown in Table 19-1.

Table 19-1 Basic Functions and Features of Timer4

Waveform mode	Sawtooth wave, triangle wave
Basic Functions	• Adding and subtracting counting direction
	• Cache function
	• Universal PWM output
	• Dedicated event output
	• EMB Control
Interrupt Type	Counting comparison match interrupts
	Counting cycle matching interrupts
	Reload count matching interrupts

The basic architecture of the general-purpose control timer Timer4 is depicted in Figure 19-1. The "<t>" shown in the block diagram indicates the cell number, i.e., "<t>" is 1~3. The references to "<t>'" later in this chapter refer to the cell number, and will not be repeated. are mentioned later in this chapter and will not be repeated.

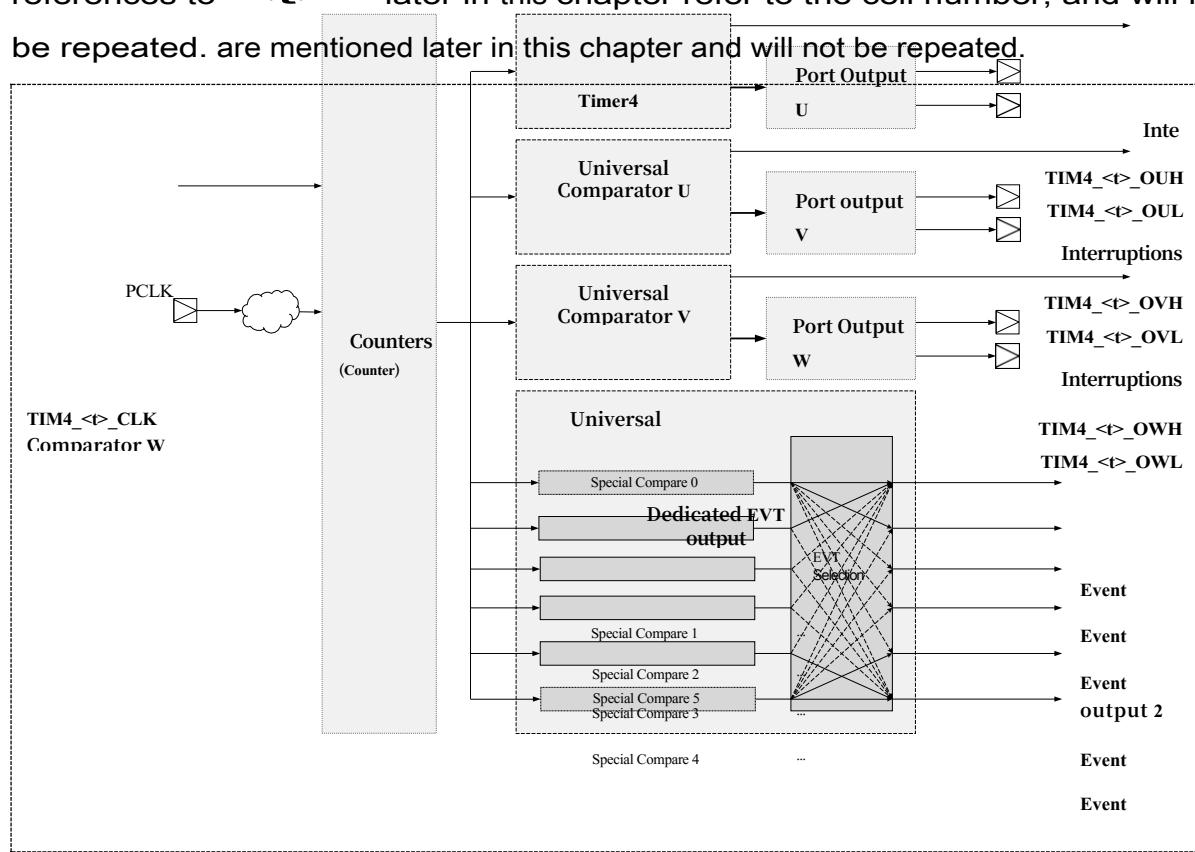


Figure 19-1 Timer4 basic block diagram

Table 19-2 shows the list of input and output ports of Timer4.

Table 19-2 Timer4 Port List

Port Name	Direction	Function
TIM4_<t>_CLK	in	Counting clock input port
TIM4_<t>_OUH	out	
TIM4_<t>_OUL		
TIM4_<t>_OVH		
TIM4_<t>_OVL		PWM output port
TIM4_<t>_OWH		
TIM4_<t>_OWL		

19.3 Function Description

19.3.1 Basic movements

19.3.1.1 Waveform mode

Timer4 has 2 basic counting waveform modes, sawtooth mode and triangle waveform mode. MODE can be set according to CCSR.MODE to realize the two waveform modes respectively. Figure 19-2 and Figure 19-3 below show the waveform diagrams of sawtooth waveform and triangle waveform, respectively.

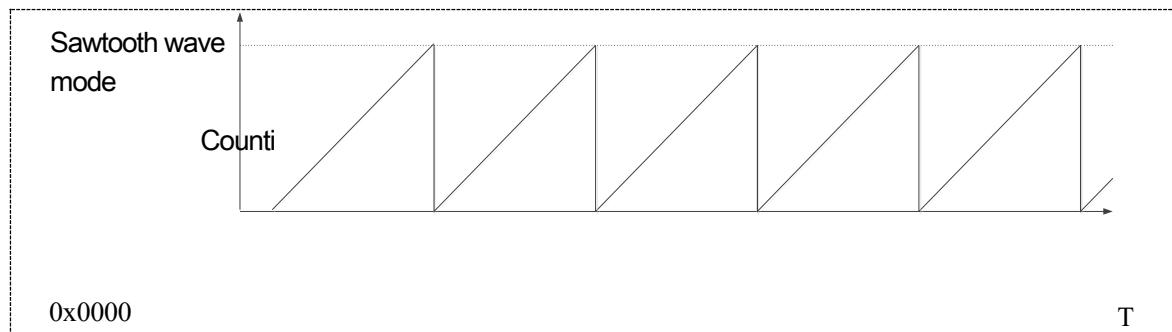


Figure 19-2 Timer4 sawtooth waveform

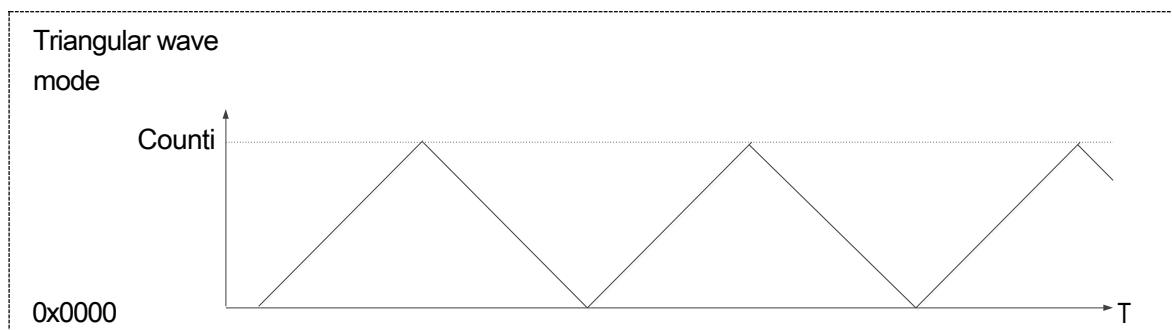


Figure 19-3 Timer4 Triangle Waveform

19.3.1.2 Countin

g action

- 1) The operation and control flow of the sawtooth wave counting is shown in Figure 19-4.

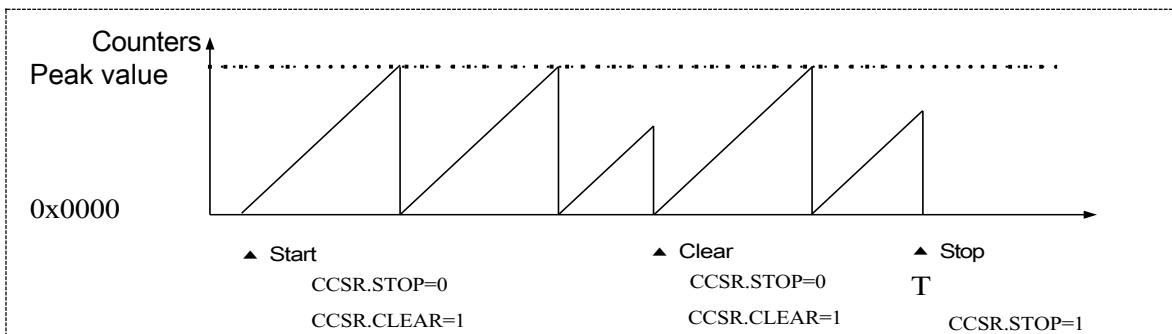


Figure 19-4 Timer4 Ramp Pattern Counting Action

- a) Set mode CCSR.MODE=0.
 - b) Set the count peak CPSR register.
 - c) STOP=0 and CCSR.CLEAR=1, the counter count value (CNTR)s initialized to 0x0000 and the counting operation is started. The counter value is incremented from 0x0000, and when the peak value (CPSR) is reached, **the count** value returns to 0x0000, and the operation is repeated.
 - d) Counting period = (CPSR+1) × counting clock period
 - e) During counting, writing CCSR.STOP=0 and CCSR.CLEAR=1 initializes the count value to 0x0000 and continues the counting operation.
 - f) During counting, writing CCSR.STOP=1 and CCSR.CLEAR=1 initializes the count value to 0x0000 and stops the counting operation.
- 2) The triangular wave counting operation and control flow is shown in Figure 19-5.

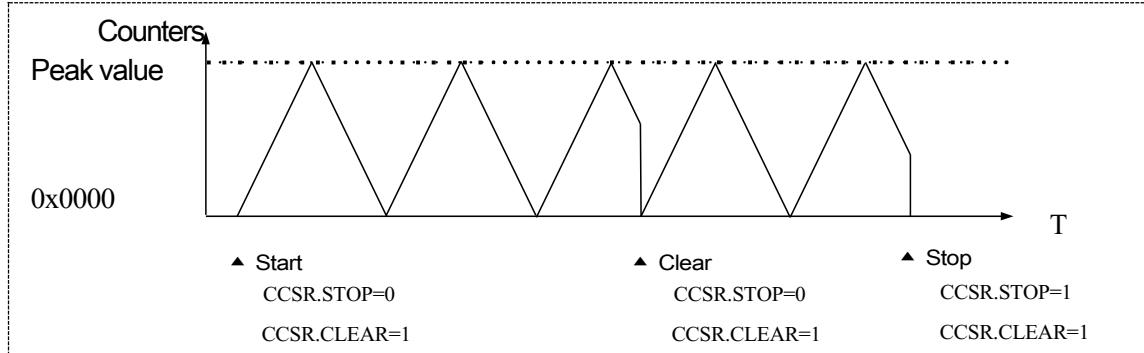


Figure 19-5 Timer4 Triangle Wave Mode Counting Action

- a) Set mode CCSR.MODE=1.
- b) Set the count peak CPSR register.
- c) STOP=0 and CCSR.CLEAR=1, the counter count value (CNTR)s initialized to 0x0000 and the counting operation is started. The counter value is counted incrementally from 0x0000 until the peak count is reached; when the peak count (CPSR) is reached, the counter is counted decrementally until the count value returns to 0x0000; after that, the count is counted incrementally again and the operation is repeated in sequence.
- d) Counting period = (CPSR) × 2 × counting clock period
- e) During counting, writing CCSR.STOP=0 and CCSR.CLEAR=1 initializes the count value to 0x0000 and resumes the incremental counting operation, and then repeats the above operation.
- f) STOP=1 and CCSR.CLEAR=1 to initialize the count value to 0x0000 and stop the count operation.

19.3.1.3 Compare Outputs

- 1) Figure 19-6 below shows an example of the waveform output of the comparison output module in sawtooth counting mode.

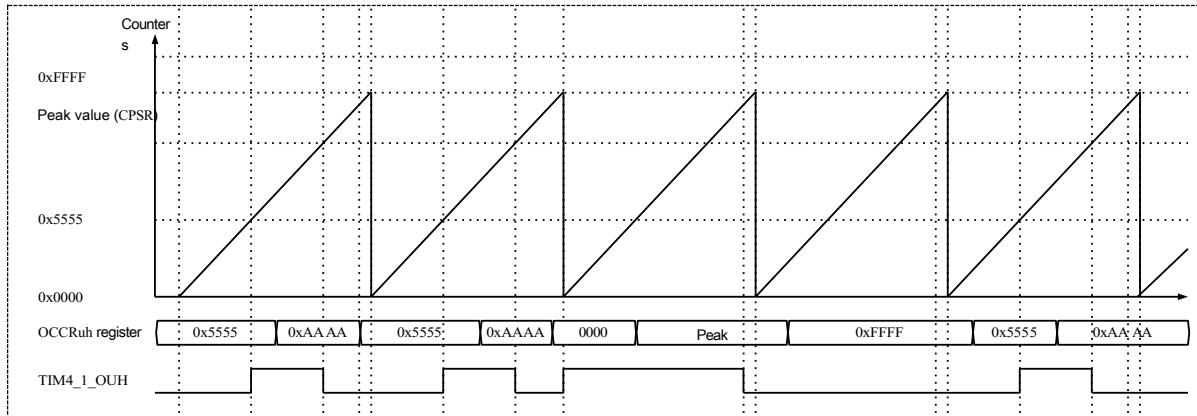


Figure 19-6 Example of Sawtooth Mode Waveform Output

- 2) Figure 19-7 below shows an example of the waveform output of the comparison output module in delta counting mode.

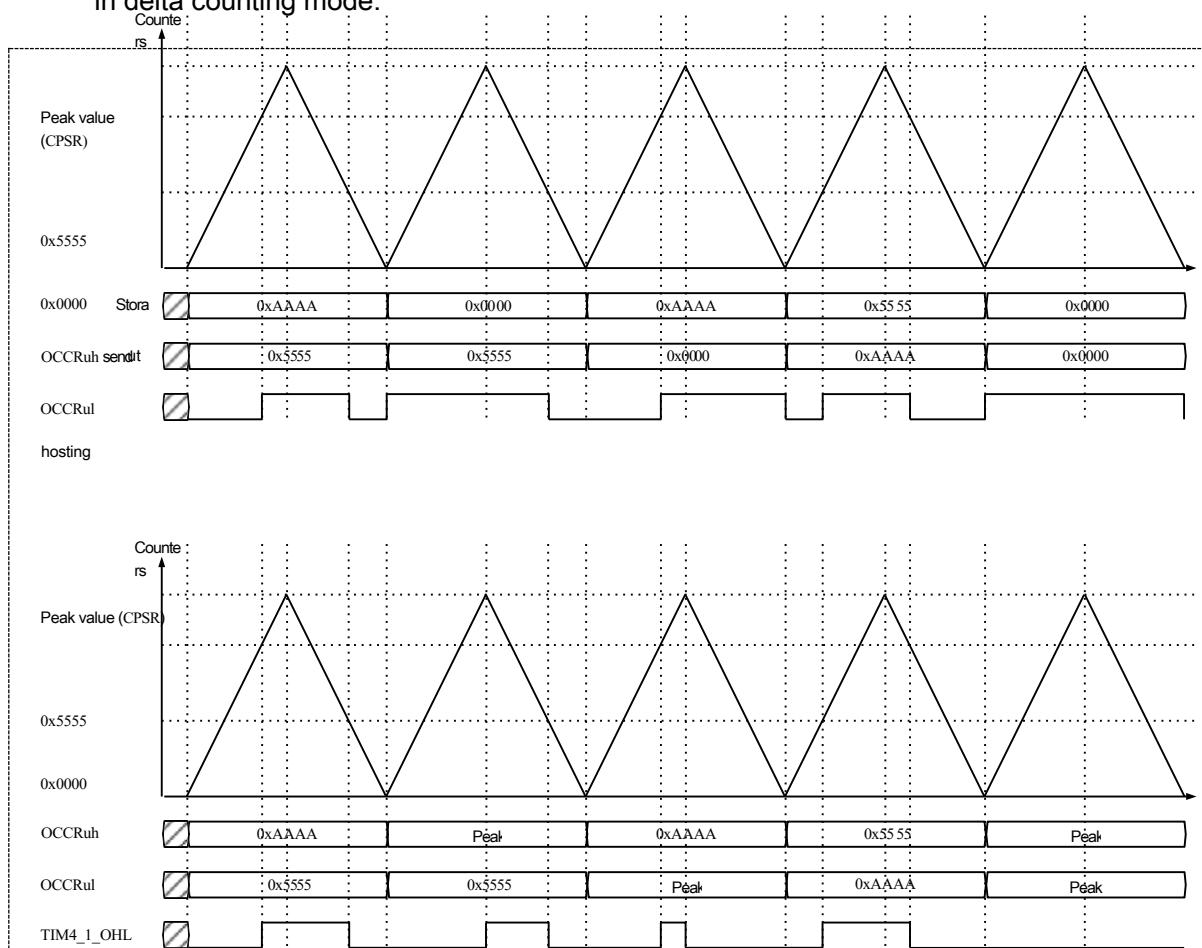


Figure 19-7 Example of waveform output in triangle mode

19.3.2 Cache function

Timer4's Cycle Reference Register (CPSR) General Comparative Reference Register (OCCR) General Purpose Mode Control Register (OCMR) Special Purpose Comparative Reference Register (SCCR), and Special Purpose Mode Control Register (SCMR) all have cache functions.

19.3.2.1 Periodic reference register cache function

The CPSR has a cache function register, and the written count peak data is first stowed in the buffer register. The data is transferred from the buffer register to the CPSR register under the following conditions

- When the buffer function is disabled (CCSR.BUFEN=0) the write data is immediately transferred from the buffer register to the CPSR register.
- When the buffer function is enabled (CCSR.BUFEN=1) the data is transferred from the buffer register to the CPSR register when the counter is stopped (CCSR.STOP=1) or when the counter count value is "0x0000".

Caution:

- - When data is read from the CPSR, the value read is not the value of the CPSR buffer register, but the value of the CPSR register. When the buffer function is enabled, the value read before the transfer is complete is not the most recently written value, but the last written CPSR value.

Figure 19-8 shows the operation of modifying the peak count CPSR when the sawtooth mode is in place and the buffering function is disabled.

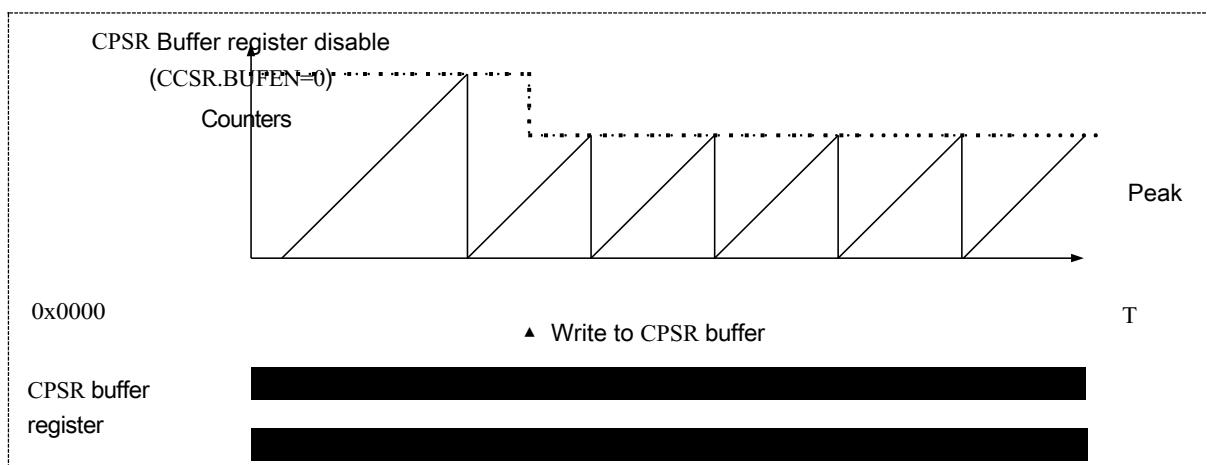


Figure 19-8 Modifying the Ramp Count Period when the Cache is Invalid

Caution:

n:

- When the buffering function is disabled, the write data is immediately transferred from the buffer register to the CPSR counter, and the count period is changed immediately after the write operation is completed. In this case, if the written value is smaller than the current

counter value, the counter will continue to count incrementally until it reaches "0xFFFF", so special attention should be paid to this condition.

Figure 19-9 shows the operation of modifying the peak count CPSR when the sawtooth mode is enabled and the buffer function is enabled.

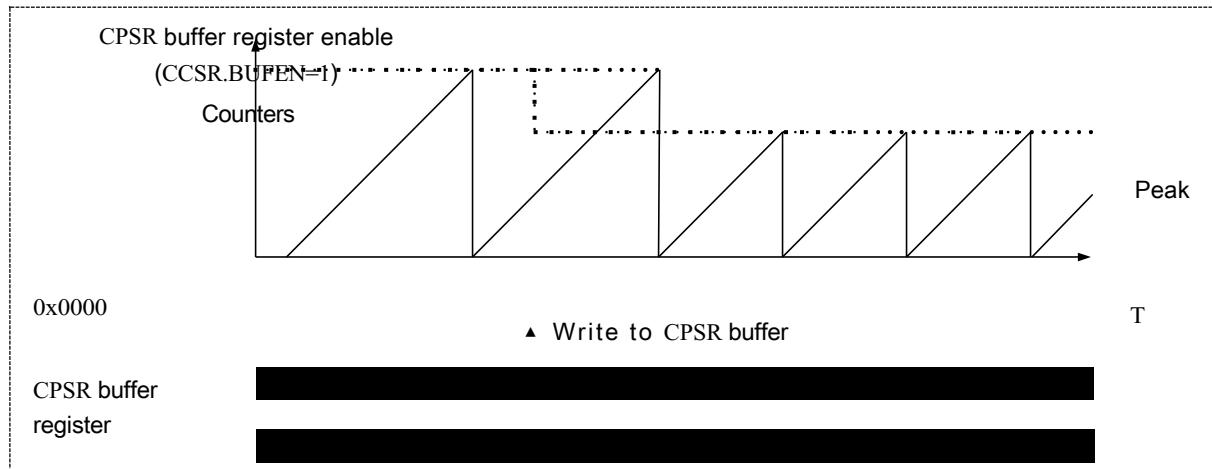


Figure 19-9 Modifying the Ramp Count Period when Cache is Enabled

As shown in the figure, when the buffer function is enabled, the written data is transferred from the buffer register to the CPSR register when the counter is stopped or when the counter count value is "0x0000".

Figure 19-10 shows the operation of modifying the peak count CPSR when the delta wave mode is enabled and the buffer function is enabled.

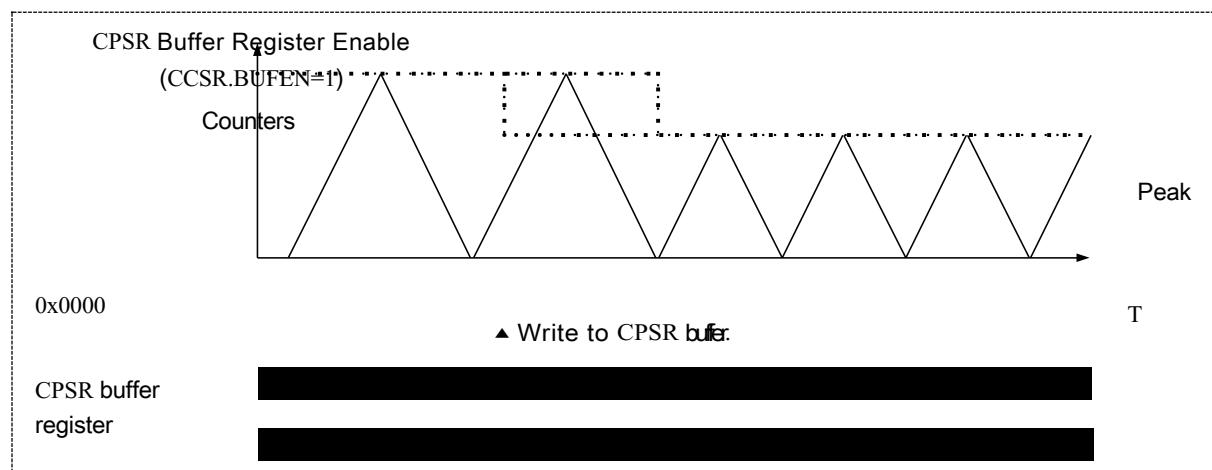


Figure 19-10 Modifying the Triangle Count Period when Cache Enabled

As shown in the figure, when the buffer function is enabled in the delta wave counting mode, the written data is transferred from the buffer register to the CPSR register when the counter is stopped or when the next count of the counter is "0x0000". The counter cycle change starts at the next counter cycle after the write operation is completed.

19.3.2.2 General comparison register cache function

The general purpose comparison reference register (OCCR) and general purpose mode control register

(OCMR) both have a buffer register function that, when enabled, transfers the load to the OCCR and OCMR registers at the specified transfer moment. the OCCR buffer function can be used to change the comparison value synchronously during counting, and the OCMR buffer function can be used to change the internal PWM output synchronously during counting at the count overflow point (sawtooth wave) the The OCCR buffer function can be used to change the internal PWM output synchronously during the count overflow point (sawtooth wave), count valley point or peak point (triangle wave).

a) When the link transfer of the output comparison and counter period interval response functions is disabled, the buffered value is loaded into the register at the set count state. The loading situation at this time is independent of the counter cycle interval counter.

Figure 19-11 shows the waveform (x=L or H) when the general-purpose output comparison OCCR buffer function is enabled, the counter is loaded at zero value (OCER.CxBUFEN=01), and the counter cycle interval response link is disabled (OCER.LMCx=0).

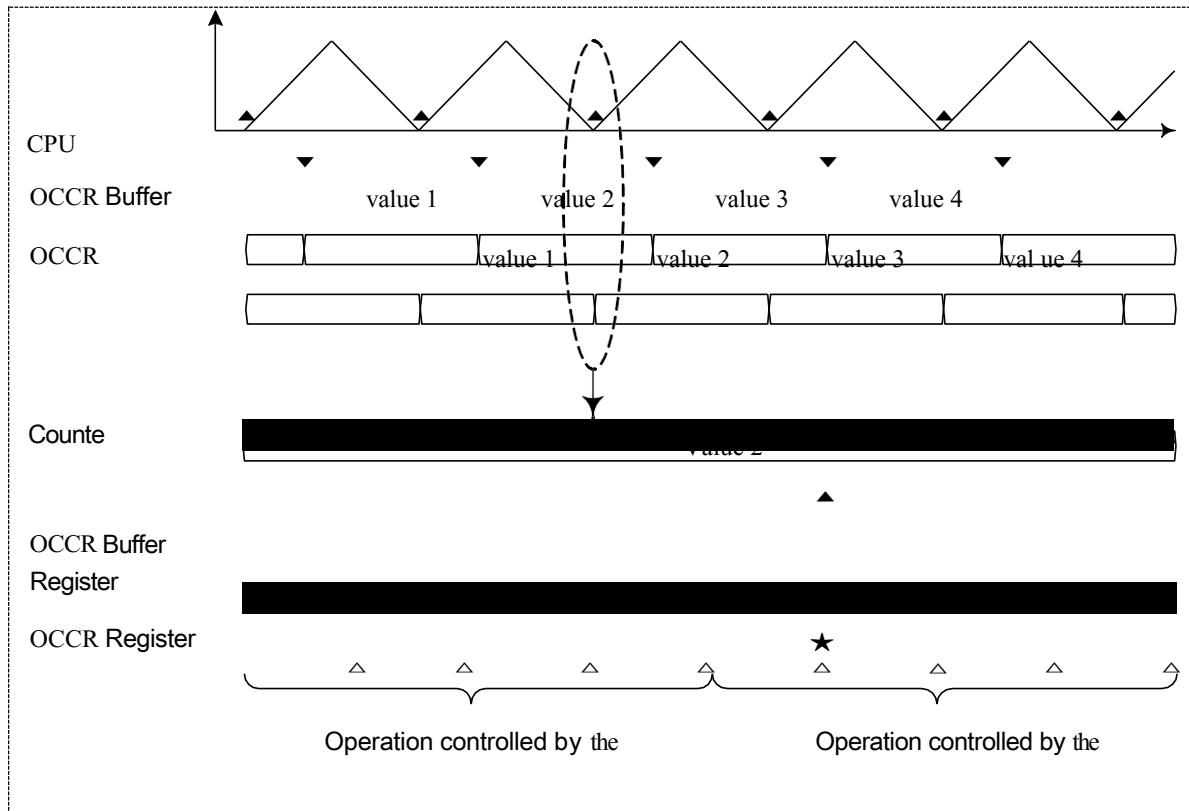


Figure 19-11 OCCR Buffered Data Transfer (Periodic Interval Response When Link is Disabled)

The top half of the figure is a global schematic and the bottom half is a zoomed-in view during the transfer operation.

The counter is in delta wave counting mode, and the underflow interrupt is generated at the flag ▲ (count valley) moment. At the moment of ▼, the CPU rewrites the OCCR register and the written data is stored in the OCCR buffer register. Afterwards, when the counter count value is 0x0000, the data loading operation is performed and the data is transferred from the buffer register to the OCCR register, while the interrupt flag IRQZF is generated.

At moment ▲, the output compares the event of matching the set OCCR register value with the count value, performs a change to the PWM output and sets the OCSR.OCFx bit (x=L or H). After moment ★ (count value = 0x0000), the port output performs the operation based on the new OCCR data. Before moment ★ the operation is performed based on the original OCCR data.

The figure illustrates the transfer operation of the OCCR buffer register at the count valley. Similar to the cache transfer operation of the OCMR buffer register; similarly, the transfer operation at the count peak is similar. The new data takes effect immediately after the transfer moment (the new write data controls the PWM output and the interrupt flag is set)

- b) When the counter period interval response link is enabled, the buffer value is in the set count state and the period interval counter count value is 0 to perform the buffer register transfer operation.**

Figure 19-12 shows the waveform (x=L or H) when the general-purpose output comparison OCCR buffer function is enabled, the counter is loaded at zero value (OCER.CxBUFEN=01), the counter cycle interval response link is enabled (OCER.LMCx=1).

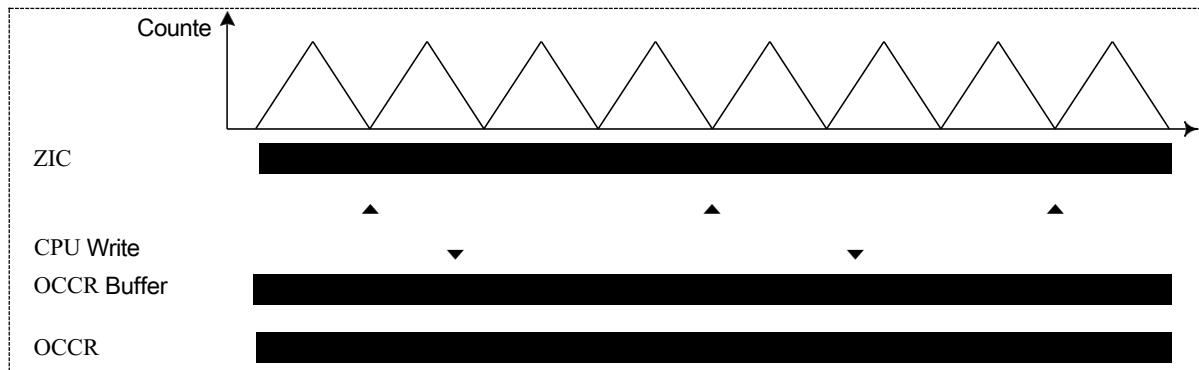


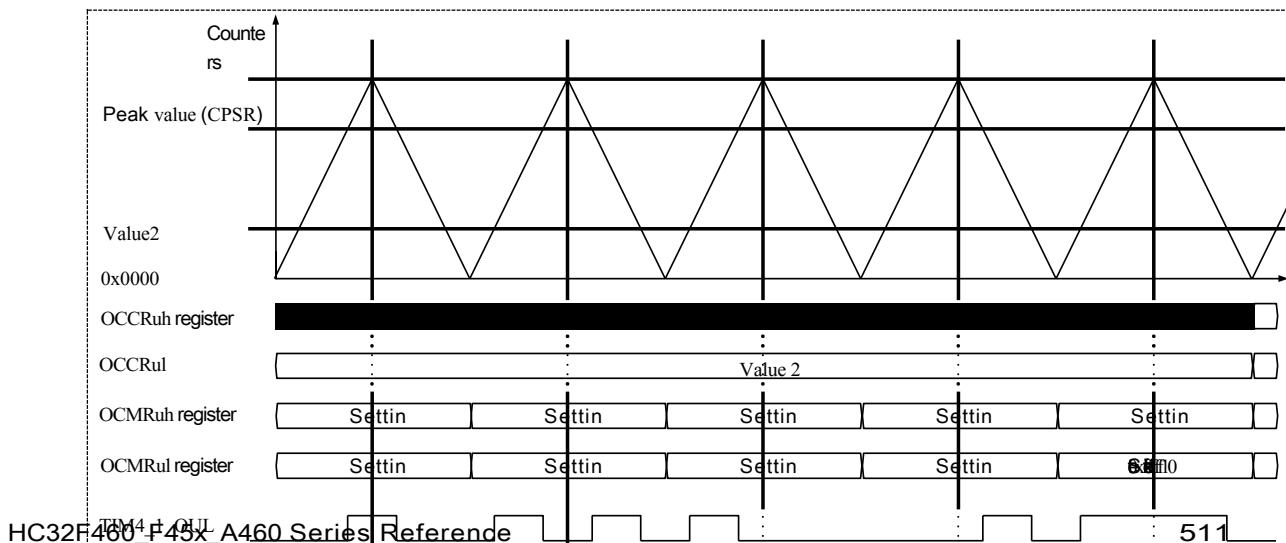
Figure 19-12 OCCR Buffered Data Transfer (Cycle Interval Response Link Enable)

The counter is in delta counting mode, and the period interval counter (CVPR.ZIC) counts decreasingly from 2 to 0. The underflow interrupt is at flag ▲.

(Count valley point) moment generated. At the ▼ moment, the CPU rewrites the OCCR register and the written data is stored in the OCCR buffer register. Later, when the counter count value is 0x0000 and the cycle interval counter (CVPR.ZIC) is 0, the data load operation is performed and the data is transferred from the buffer register to the OCCR register, and the interrupt flag IRQZF is generated.

The figure illustrates the transfer operation of the OCCR buffer register at the count valley. Similar to the cache transfer operation of the OCMR buffer register; similarly, the transfer operation at the count peak is similar. The new data takes effect immediately after the transfer moment (the new write data controls the PWM output and the interrupt flag is set)

When using the Channel Link operation mode, enabling both the OCCRh and OCCRu buffers can generate different PWM output waveforms. Figure 19-13 illustrates the case where the OCMR register value is changed to generate different output waveforms in TIM4_<t>_OUL mode while the output comparison registers OCCRh and **OCCRu** remain unchanged.



**Figure 19-13 Output Compare Buffer Data Transfer (OCMR
Buffer Enable)**

19.3.2.3 Dedicated comparison register cache function

The dedicated comparison reference register (SCCR) and the dedicated mode control register (SCMR) both have buffer function registers. When the buffer function is enabled, the CPU writes to the SCCR and SCMR buffer registers and loads the values into the SCCR and SCMR registers at the set counter state.

- a) When the counter cycle interval response link transfer is disabled, the buffered transfer operation is only related to the counter state and is not affected by the counter cycle interval counter.**

Figure 19-14 shows the enable register buffer function, which disables the counter cycle interval response link transfer (SCSR.LMC=0) and the transfer to the SCCR and SCMR registers when the counter is zero (SCSR.BUFEN=01).

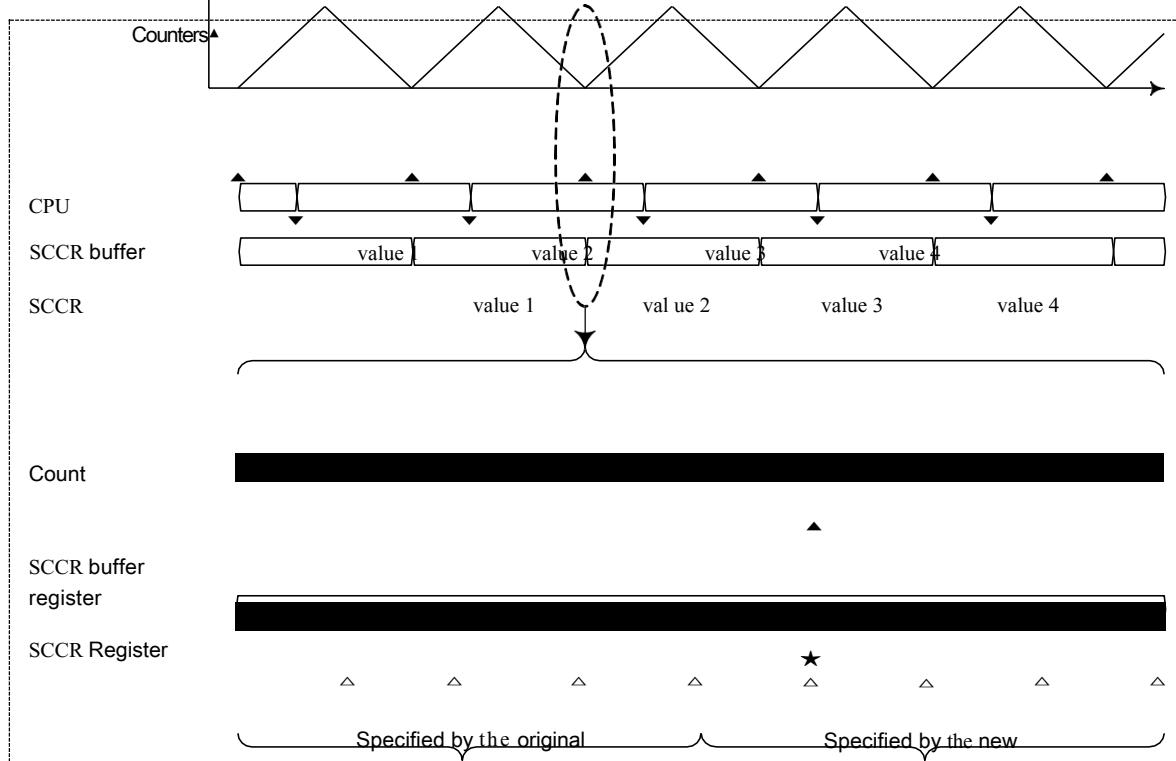


Figure 19-14 SCCR Buffered Transfer Operation (Periodic interval response when link transfer is disabled)

The top half of the figure shows a global schematic and the bottom half a partial zoomed-in view of the buffer register transfer operation.

The counter is in delta wave counting mode, and the underflow interrupt is generated at the flag ▲ (count valley) moment. At the moment of ▼, the CPU rewrites the SCCR register and the written data is stored in the SCCR buffer register. Afterwards, when the counter count value is 0x0000, the data loading operation is performed and the data is transferred from the buffer register to the SCCR register, while the interrupt flag IRQZF is generated.

At moment △, a comparison operation is performed with the count value based on the set SCCR

register value. After moment ★ (count value = 0x0000), the dedicated event output signal performs the operation based on the new SCCR data. Before the moment ★, the operation is executed according to the SCCR data.

The figure illustrates the transfer operation of the SCCR buffer register at the count valley, which is similar to the cache transfer operation of the SCMR buffer register; similarly, the transfer operation at the count peak is similar. The new data takes effect immediately after the transfer moment (the new write data sets the dedicated event output signal and the interrupt flag)



b) When the counter interval response link is enabled, the buffer value is in the set count state and the interval counter count value is 0, the buffer register transfer operation is performed.

Figure 19-15 shows the diagram when the SCCR buffer function is enabled, the counter is loaded at zero value (SCSR.BUFEN=01) and the counter cycle is spaced in response to link transfer enable (SCSR.LMC=1).

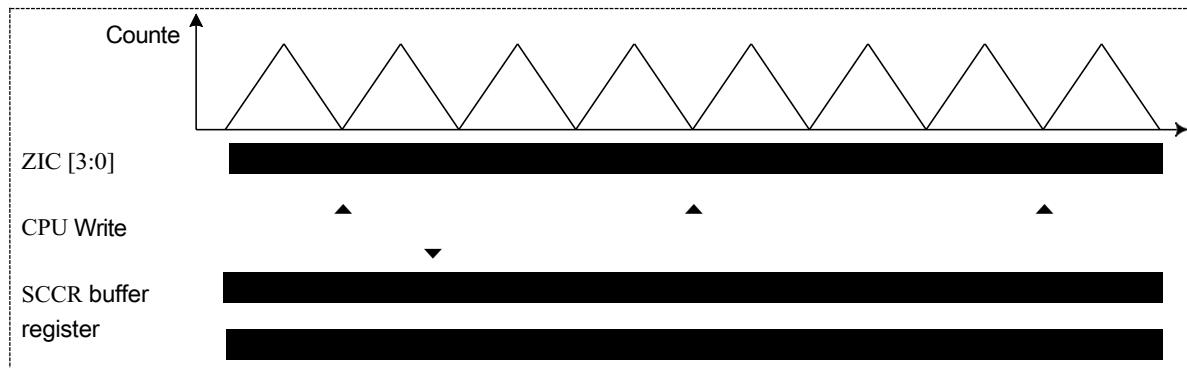


Figure 19-15 SCCR Buffered Transfer Operation (When Link Transfer Enable is Responded to at Periodic Intervals)

The counter is in delta counting mode, and the period interval counter (CVPR.ZIC) counts decreasingly from 2 to 0. The underflow interrupt is at flag ▲.

(Count valley point) moment generated. At the ▼ moment, the CPU rewrites the SCCR register and the written data is stored in the SCCR buffer register. Later, when the counter count value is 0x0000 and the cycle interval counter (CVPR.ZIC) is 0, the data load operation is performed and the data is transferred from the buffer register to the SCCR register, and the interrupt flag IRQZF is generated.

19.3.3 Universal PWM Output

19.3.3.1 Independent PWM output

In the pass-through mode (POCR.PWMMD=00), the reference values of OCCRxh and OCCRxI and the port output states of OCMRxh and OC MRxI (x=u, v, w) are set separately to achieve different PWM outputs. In this case, the PWM outputs of each port are controlled independently. Figure 19-16 and Figure 19-17 show examples of independent PWM outputs under sawtooth and delta waves of Unit 1, respectively.

Caution:

- The direct pass-through mode means that the internal output signals (in_opxh, in_opxl) resulting from the comparative matching of the values of the universal comparison reference registers (OCCRxh, OCCRxI) are output directly to the corresponding ports (TIM4_<t>_OXH, TIM4_<t>_OXL) (X=U, V, W, x=u, v, w)

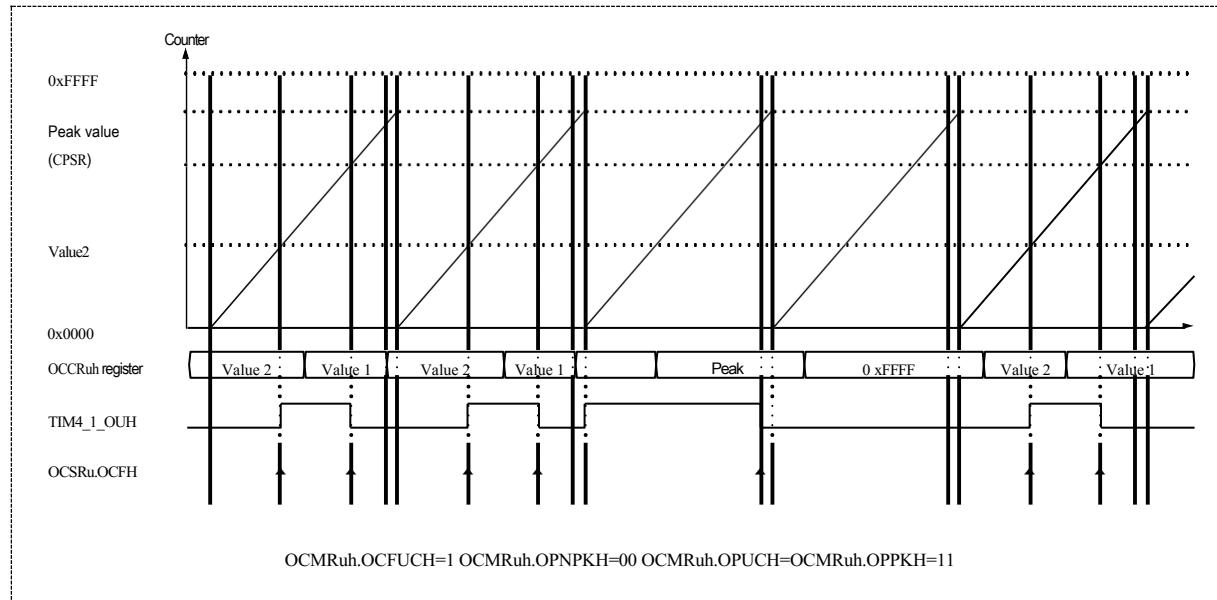


Figure 19-16 Example of Sawtooth Independent PWM Output

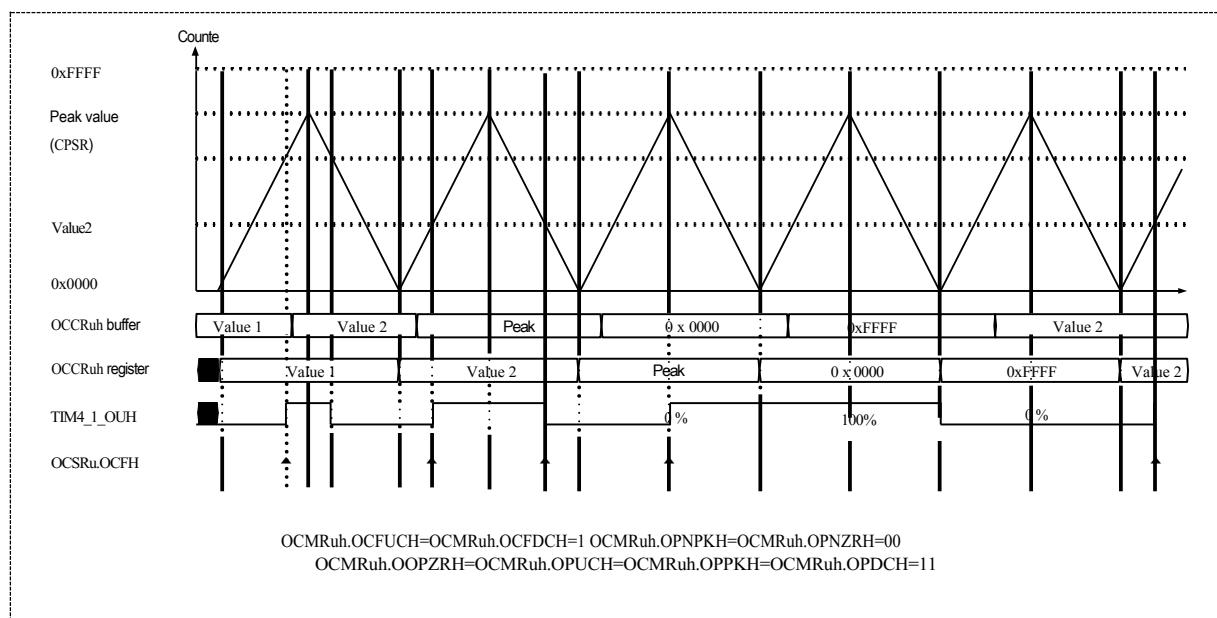


Figure 19-17 Example of triangular wave independent PWM output

19.3.3.2 Extended PWM Output

In pass-through mode (POCR.PWMMD=00), the port output state of TIM4_<t>_OXL can also be determined by an extension bit (bit32~16) in the OCMRxI register, which is set in relation to the OCCRxh reference value, thus enabling extended PWM outputs on the TIM4_<t>_OXL port ($x=u,v,w$, $X=u,v,w$). Figure 19-18 shows the PWM outputs of the TIM4_<t>_OEH and TIM4_<t>_OUL ports in this mode.

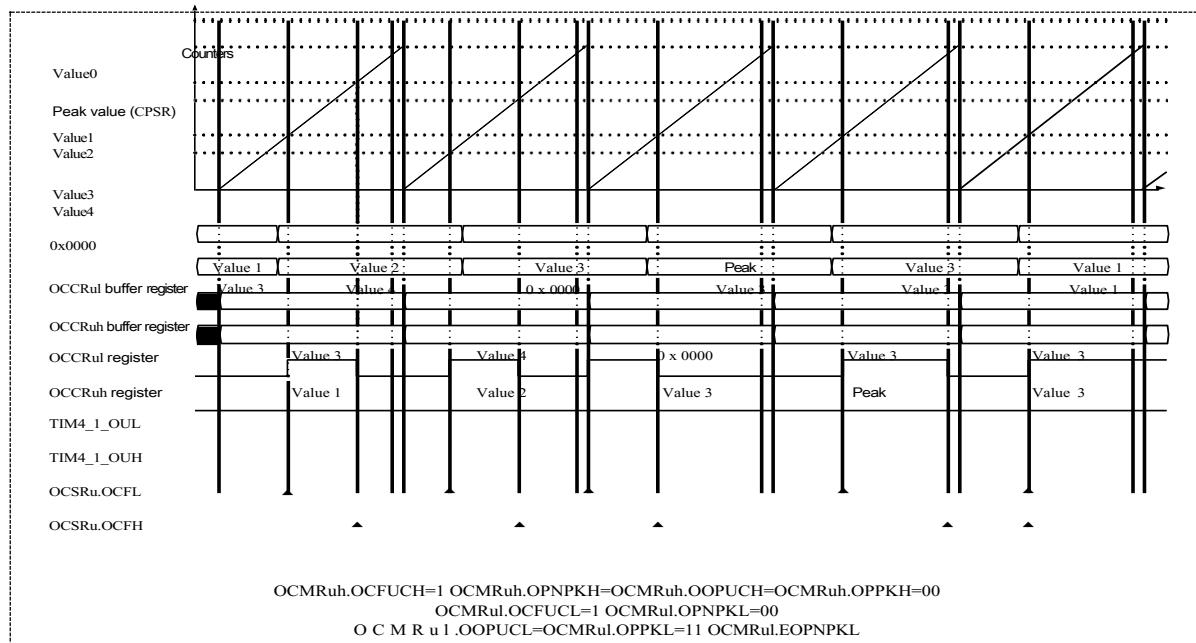


Figure 19-18 Sawtooth Wave Extended PWM Output

Cautio

n:

- In the independent PWM output mode, the port status of TIM4_<t>_OXL is determined by bit15~bit4 of the OCMRxI register. In independent PWM output mode, the port status of TIM4_<t>_OXL is determined by bit15~bit4 of OCMRxI register, which is only related to OCCRxI base value ($X=u, V, W$, $x=u, v, w$)

19.3.3.3 Complementary PWM Outputs

Software settings for complementary PWM output

In straight-through mode (POCR.PWMMD=00), the OCCRxh and OCCRxl ($x=u, v, w$) reference values are set directly to achieve a pair of complementary PWM waveform outputs to the ports, and 3 groups of ports can be set in the same way to achieve 3 complementary PWM outputs. This is shown in Figure 19-19.

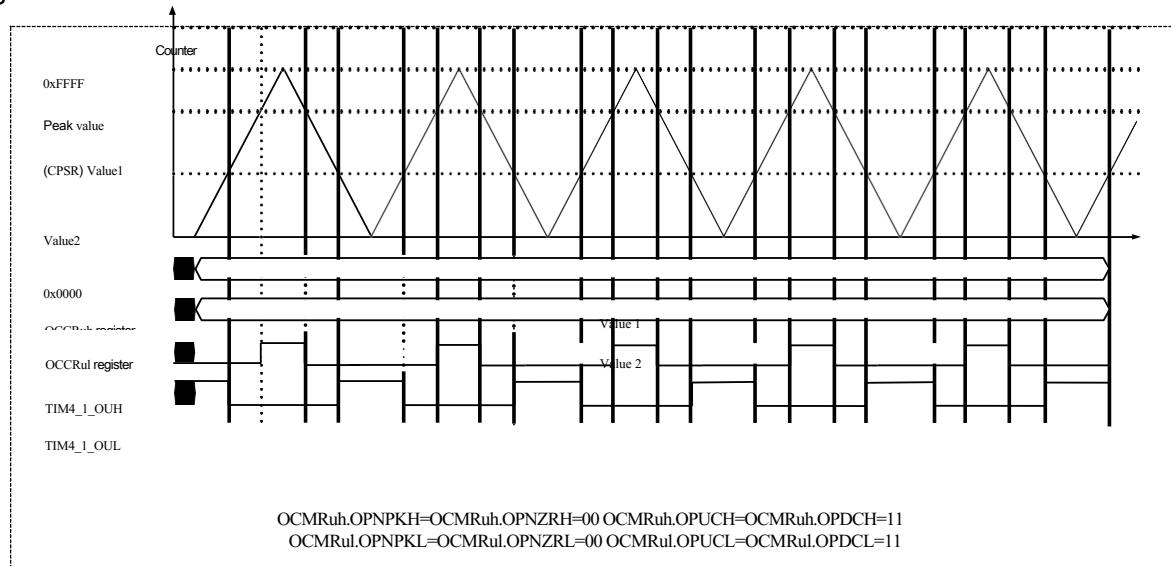


Figure 19-19 Software Implementation of Complementary PWM Outputs

Hardware settings for complementary PWM outputs

In the dead timer mode (POCR.PWMMD=01), the internal output signal (`in_opxl`) generated by the comparative matching of the value of the general comparison reference register (OCCRxl) and the set value of the PWM dead control register (PDAR/PDBR) is offset by timing to achieve complementary PWM outputs in hardware.

In this mode, the `TIM4_<t>_OXH` port output has the same polarity as **in_opxl**, and the `TIM4_<t>_OXL` port output has the opposite polarity as **in_opxl** ($X=U, V, W, x=u, v, w$)

Figure 19-20 shows an example of the complementary PWM output in dead timer mode.

If **in_opxl** rising edge is detected, the `TIM4_<t>_OXL` output goes low, the dead counter loads the PDBRx register setting and starts decreasing counting, when the count value becomes 0x0000, the counter stops and makes the `TIM4_<t>_OXH` output go high; if **in_opxl** falling edge is detected, the `TIM4_<t>_OXH` output goes low, the dead counter loads the PDARx register setting and starts decreasing counting, when the count value becomes 0x0000, the counter stops and makes the `TIM4_<t>_OXH` output go high.

TIM4_<t>_OXH goes low, the deadband counter loads the set value of the PDARx register and starts the decrement count, when the count value becomes 0x0000, the counter stops and makes TIM4_<t>_OXL go high (X=U, V, W, x=u, v, w)

By setting the PWM dead time control registers PDAR and PDBR, the dead time of output rise and fall variation can be set accordingly.

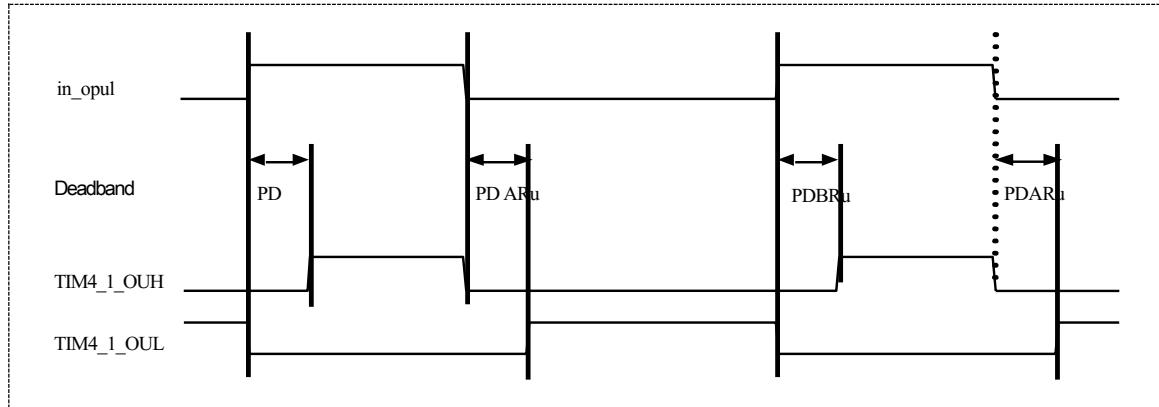


Figure 19-20 Complementary PWM Output in Dead Timer Mode

When the high pulse width of `in_opxl` is less than the dead time set by `PDBR`, only the `TIM4_<t>_OXL` output goes low. The condition for the `TIM4_<t>_OXL` output level to go high is when the falling edge of `in_opxl` is followed by the dead time set in the `PDAR` register. In this case, the `TIM4_<t>_OXH` output will remain low.

When the low pulse width of `in_opxl` is less than the dead time set by `PDAR`, only the `TIM4_<t>_OXH` output goes low. The condition for the `TIM4_<t>_OXH` output level to go high is when the rising edge of `in_opxl` is followed by the dead time set in the `PDBR` register. In this case, the `TIM4_<t>_OXL` output will remain low ($x=u, v, w$, $x=u, v, w$) As shown in Figure 19-21.

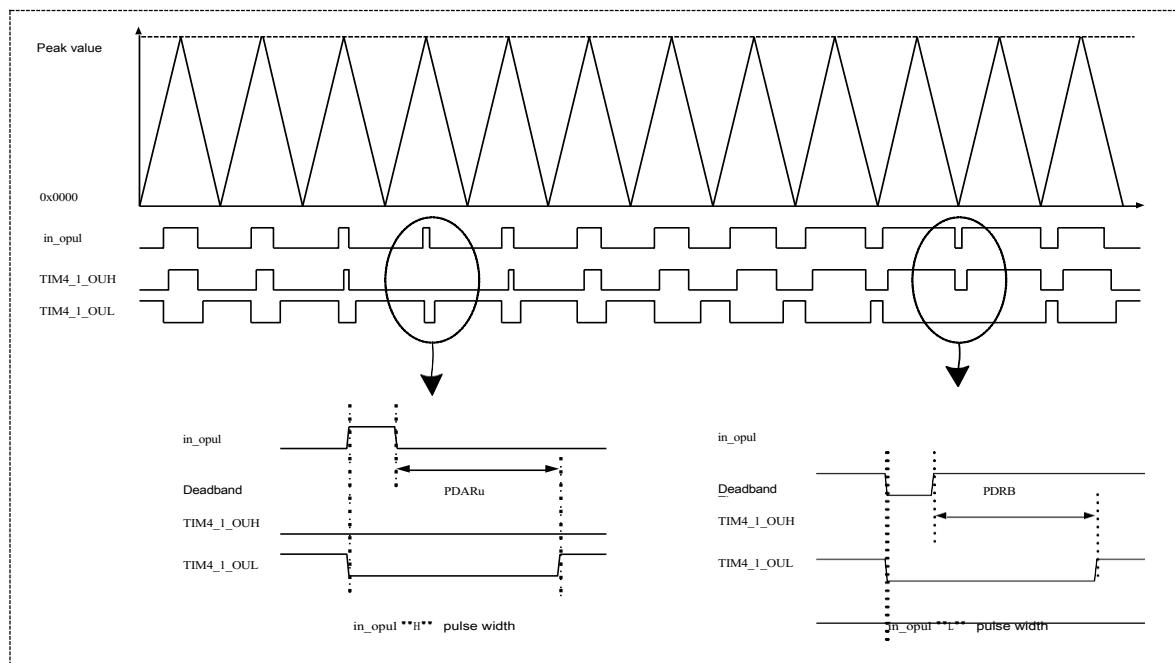


Figure 19-21 Waveform output in dead timer mode when pulse width is abnormal

In addition to the above hardware implementation of the deadband output mode, it is possible to monitor the pulse width of the internal comparison match signal (`in_opxl`) to achieve filtering control of the `in_opxl` signal. This `in_opxl` deadband output implementation with pulse width filtering is called deadband counter filtering mode (POCR.PWMMD=10) $x=u$,

v, w)

In the deadband counter filter mode, the filter width is determined by the setting value of the PWM filter control register (PFSRn). When the pulse width of in_opxl is greater than the time set by register PFSRn, the filter counter delays the **in_opxl** signal for the time set by PFSR and then outputs it after the complementary PWM output ($x=u, v, w$) described by the dead timer mode.

If the rising edge of the signal in_opxl is detected, the filter counter loads the value of the PFSR register and starts to measure the high level width of **in_opxl**, when the high level pulse width of in_opxl is greater than the time set in the PFSR register, after the time set in the PFSR, the TIM4_<t>_OXL output goes low and the dead counter When the count value becomes 0x0000, the counter stops and makes TIM4_<t>_OXH output high; if the falling edge of the signal **in_opxl is detected**, the filter counter loads the PFSR register value and starts to measure the low level width of **in_opxl**, when the low level pulse width of **in_opxl is larger than that of the** PFSR register, the filter counter starts to count the low level pulse width of in_opxl. When the low-level pulse width of in_opxl is greater than the time set by the PFSR register, after the time set by the PFSR, the TIM4_<t>_OXH output goes low, the deadband counter loads the value set by the PDAR register and starts decreasing counting, when the count value becomes 0x0000, the counter stops and makes the TIM4_<t>_OXL output go high. when the count value becomes 0x0000. The outputs TIM4_<t>_OXH and TIM4_<t>_OXL will remain unchanged ($X=U, V, W, x=u, v, w$) when the level pulse width of **in_opxl is less than the time set by the PFSR register.**

Figure 19-22 shows an example of the complementary PWM output in dead counter filter mode.

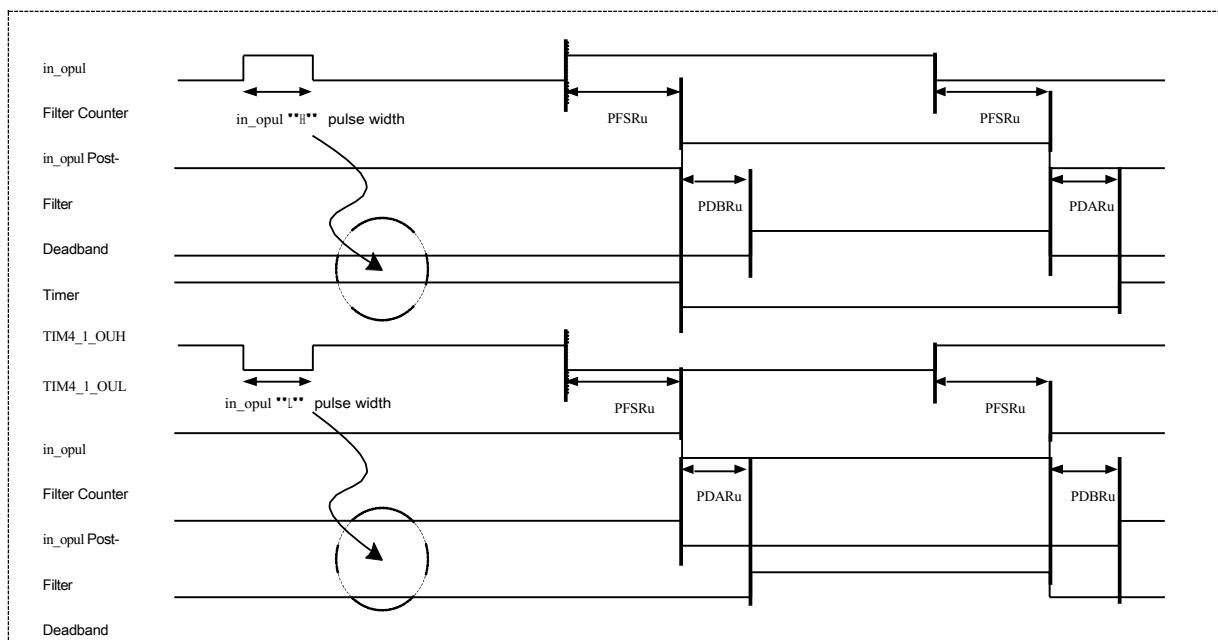


Figure 19-22 Complementary PWM Output in Dead Timer Filter Mode

19.3.4 Cycle interval response

The underflow interrupt mask counter is used to mask the number of times the underflow flag bit (CCSR.IRQZF) has been set. Underflow interrupt mask counter

(CVPR.ZIC[3:0]) operates as a decrement counter, loading the value set by CVPR.ZIM[3:0] at the beginning, and when CVPR.ZIC[3:0] = "0", the underflow flag bit (CCSR.IRQZF) is set to "1".

The overflow interrupt mask counter is used to mask the number of times the overflow flag bit (CCSR.IRQPF) has been set. Overflow interrupt mask counter

(CVPR.PIC[3:0]) operates as a decrement counter, loading the value set by CVPR.PIM[3:0] at the beginning, and when CVPR.PIC[3:0] = "0", the overflow flag bit (CCSR.IRQPF) is set to "1".

Figure 19-23 below shows the timing diagrams of IRQZF and IRQPF when the cycle interval response is set.

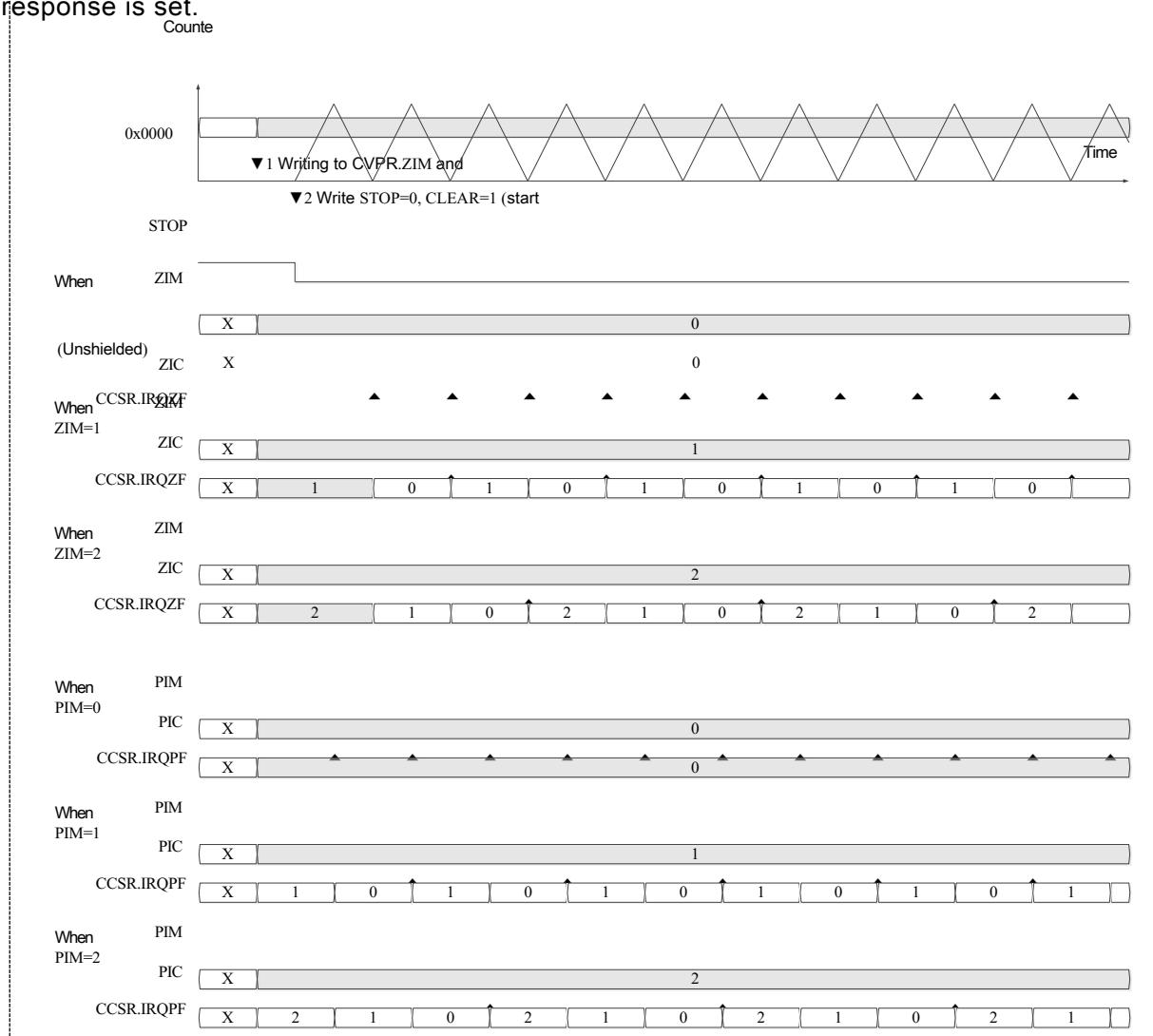


Figure 19-23 Cycle interval response time sequence diagram

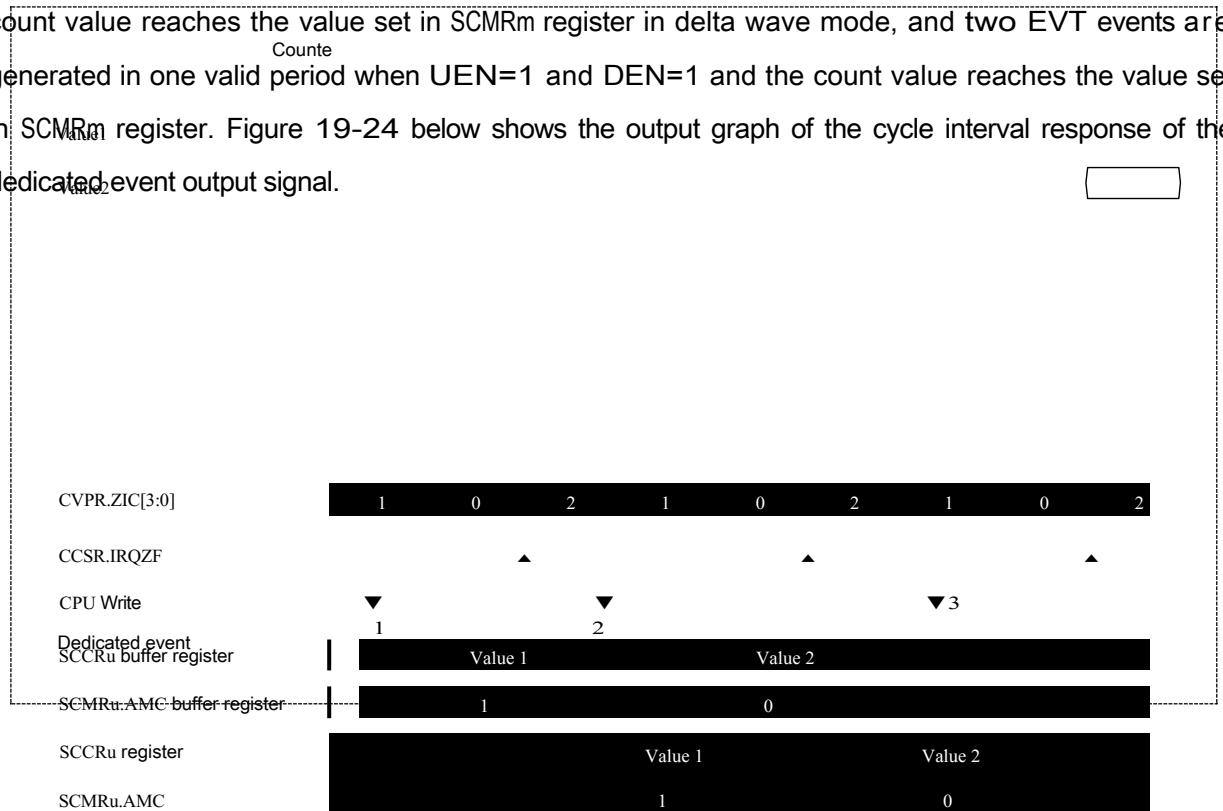
- ▼1 When the counter is stopped, the CVPR.ZM and CVPR.PM initialization values are written and the initial values are immediately reflected in the internal counters (CVPR.ZIC, CVPR.PIC).

▼2 Initialize and start the counter (STOP=0 and CLEAR=1), the counter will count incrementally from zero after bus reset or from software initialization CLEAR=1, the CCSR.IRQZF flag will not be set immediately at this moment, then whenever the count value of the interrupt mask counter is 0x0000 and the count value of the counter is 0x0000 and CPSR, the flag IRQZF or CCSR.IRQPF is set.

Caution:

- ZIM and CVPR.PIM are written while the counter is running, the set values will not immediately react to the interrupt mask counters (CVPR.ZIC and CVPR.PIC). If a soft reset (CLEAR=1) is written, the written CVPR.ZIM and CVPR.PIM values will be immediately loaded as the initial value of the interrupt mask counter.

The comparison match event (dedicated event output) of the dedicated comparison reference register (SCCRm) also has a cycle interval response function. When the dedicated control status register SCSR is set to the comparison mode output (EVTMS=0), the EVT event start enable is at the up overflow point (ZEN) at the up count (UEN), at the down overflow point (DEN), and at the down count (DEN). When the cycle interval function is not used, the dedicated event output signal is set when the corresponding count match event occurs when the enable is turned on. Among them, the up-count matching event and down-count matching event do not support the period interval function. When the period interval function is used, one EVT event is generated in one valid period when UEN=1 or DEN=1 and the count value reaches the value set in SCMRm register in delta wave mode, and two EVT events are generated in one valid period when UEN=1 and DEN=1 and the count value reaches the value set in SCMRm register. Figure 19-24 below shows the output graph of the cycle interval response of the dedicated event output signal.



**Figure 19-24 Dedicated Event Output Signal Cycle Interval Response
Output**

The counter is in delta wave counting mode and the underflow interrupt mask counter (CVPR.ZIC) counts decreasingly from 2-0. The underflow interrupt is counted at moment

- ▲ Generate.

At moment ▼1, value 1 is written to the SCCR_U buffer register, and MZCE=1, MPCE=0, and AMC=0001 are written to the SCMR_U buffer register. Turn on the up-count EVT enable (UEN=1) in the SCSR register, **and the** period interval response function link enable (LMC=1) and set the underflow cache (BUFEN=01) and the buffer register SCCR_U and SCMR_U transfer operation is executed at the ▲ moment. Because the count value = SCCR_U=Value 1 and MZCE=1 and AMC=1 at the up count, **the** dedicated event output signal is set at moment △1.

At moment ▼2, **value** 2 is written to the SCCR_U buffer register. at the same time, MZCE=1, MPCE=0, AMC=0000 are written to the SCMR_U buffer register. After that, the buffer register SCCR_U and SCMR_U transfer operation is executed at the ▲ moment. Because MZCE=1 and AMC=0 and count value=SCCR_U=Value 2 at up count, **the** dedicated event output signal is set at moment △2.

19.3.5 EMB Control

Each Timer4 unit has an Output Invalid Event interface that connects to EMB events output by the EMB module. The abnormal condition events selected on this interface can be set from the EMB (see the EMB section)

If an abnormal EMB event is detected from the EMB during the normal output of the three PWM ports in each unit, the output state of the port can be changed to a predefined state. This preset port state can be Output High Resistance, Output Low, Output High, Maintain Normal Output, and Maintain Previous State. (set by ECER.EMBVAL and ECSR.HOLD)

For example, if an EMB event is generated during the normal output of the PWM port of Timer4 when ECER.EMBVAL=01 is set, the output on the PWM port becomes a high resistance state.

19.4 Interrupts and event descriptions

19.4.1 Counting comparison match interrupts

There are 6 general-purpose comparison reference registers (OCCR_m), each of which can be compared with a count value to generate a valid comparison match signal. If OCSR_n.OCIEH and OCSR_n.OCIEL are set to enable interrupt, the corresponding interrupt request (TMR4_U<t>_GCM_m, m=U, V, W; n=H, L) will be triggered. H, L will also be triggered.

19.4.2 Counting cycle matching interrupts

The CCSR.IRQPF or CCSR.IRQZF bit of the Control Status Register (CCSR) is set to 1 when the sawtooth wave progressive count reaches the upper overflow point, the triangle wave count reaches the valley point, or the triangle wave count reaches the peak point, and the CCSR.IRQPEN or CCSR.IRQZEN bit is set at this time.

If the interrupt is enabled, the count cycle matching interrupt (TMR4_U<t>_GOVF and TMR4_U<t>_GUDF) can be

triggered at the corresponding time point.

19.4.3 Reload count match interrupt

When the reload function is active, TMR4_PFSRn is used as the cycle count value of the reload timer, and it is reloaded from register TMR4_PFSRn as the initial counter value to perform decrement operation when the count is enabled, and after completing one cycle, a reload count interrupt request is generated, and the RCSR_RTIFU, RCSR_RTIFV, and RCSR_RTIFW bits in the reload control status register (RCSR) are set to 1 respectively, RTIFV, and RCSR_RTIFW bits in the reload control status register (RCSR) will be set to 1 respectively.

RTIDU, RCSR.RTIDV and RCSR.RTIDW interrupt masks are invalid, the corresponding reload count match interrupt request (TMR4_U<t>_RLOm, m=U, V, W) will also be triggered.

19.4.4 Dedicated comparison match events

Timer4's six dedicated comparison reference registers (SCCRm) correspond to six dedicated event output signals that can be used to selectively trigger other modules, such as starting an ADC.

During clock counting, if a count comparison match event (TMR4_U<t>_SCMmn|m=U, V, W; n=H, L) occurs for the dedicated comparison reference value (SCCRm), a corresponding valid request signal is generated, which can be configured to any event EVT output signal (set by the SCSR.EVTOS bit) is used to trigger other modules.

The output of this event request signal can be selected in either compare start mode or delayed start mode. In compare start mode (SCSR.EVTMS=0) the dedicated event output signal is output directly after generating a count comparison match event of SCCR; in delayed start mode (SCSR.EVTMS=1) event output signal is output after generating a count comparison match event of OCCRxh or OCCRxl (selected by SCSR.EVTDS bit; x=u, v, w), and after the reference time set by SCCR. After the cycle time set by SCCR, the dedicated event output signal is output. Figure 19-25 below shows an example of the requested output of the dedicated event output signal in delayed start mode.

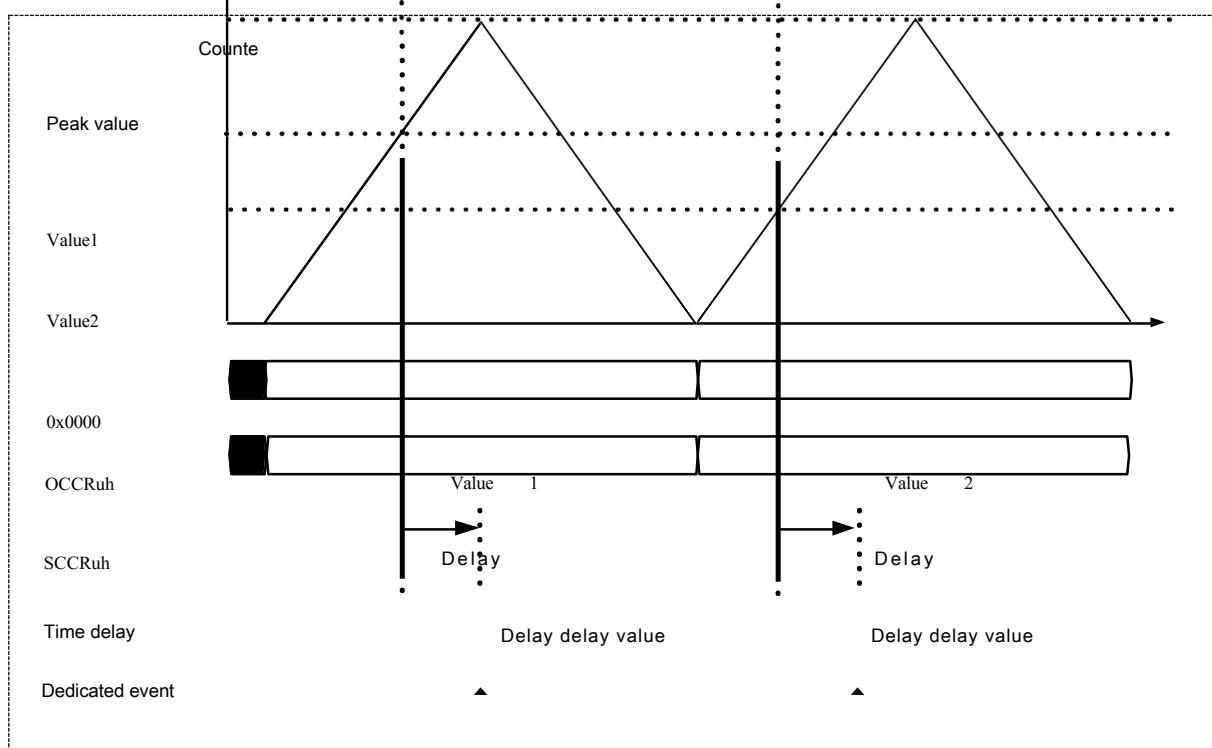


Figure 19-25 Output Timing of Dedicated Event Output Signal in Delayed Start Mode

Cautio
n:

- **During** a delayed count run, if another OCCR match event occurs with the counter, the delay counter reloads the count value and re-decrements the count. Therefore, if the OCCR match event interval is less than the set delay time SCCR, the request signal for the dedicated event output may never be generated.

19.5 Register Description

Table 19-3 shows the register list of the Timer4 module.

BASE ADDR: 0x4001_7000 (U1) 0x4002_4800 (U2) 0x4002_4C00 (U3)

Table 19-3 Register List

Register Name	Sym bols	Offset	Bit width	Reset value
Count value register	TMR4_CNTR	0x0046	16	0x0000
Periodic Reference Register	TMR4_CPSR	0x0042	16	0xFFFF
Control Status Register	TMR4_CCSR	0x0048	16	0x0040
Effective period register	TMR4_CVPR	0x004A	16	0x0000
Universal Comparison Reference Register UH	TMR4_OCCRuh	0x0002	16	0x0000
Universal Comparison Reference Register UL	TMR4_OCCRul	0x0006	16	0x0000
General comparison reference register VH	TMR4_OCCRvh	0x000A	16	0x0000
General comparison reference register VL	TMR4_OCCRvl	0x000E	16	0x0000
Universal Comparison Reference Register WH	TMR4_OCCRwh	0x0012	16	0x0000
General Comparison Reference Register WL	TMR4_OCCRwl	0x0016	16	0x0000
General Control Status Register U	TMR4_OCSRu	0x0018	16	0xFF00
General Control Status Register V	TMR4_OCSRv	0x001C	16	0xFF00
General Control Status Register W	TMR4_OCSRw	0x0020	16	0xFF00
General Extension Control Register U	TMR4_OCERu	0x001A	16	0x0000
General Extension Control Register V	TMR4_OCERv	0x001E	16	0x0000
General Extension Control Register W	TMR4_OCERw	0x0022	16	0x0000
General-purpose mode control register UH	TMR4_OCMRuh	0x0024	16	0x0000
General-purpose mode control register UL	TMR4_OCMRul	0x0028	32	0x0000_0000
General mode control register VH	TMR4_OCMRvh	0x002C	16	0x0000
General-purpose mode control register VL	TMR4_OCMRvl	0x0030	32	0x0000_0000

General mode control register WH	TMR4_OCMRwh	0x0034	16	0x0000
General mode control register WL	TMR4_OCMRwl	0x0038	32	0x0000_0000
Dedicated comparison reference register UH	TMR4_SCCRuh	0x00B2	16	0x0000
Dedicated comparison reference register UL	TMR4_SCCRul	0x00B6	16	0x0000
Dedicated comparison reference register VH	TMR4_SCCRvh	0x00BA	16	0x0000
Dedicated comparison reference register VL	TMR4_SCCRvl	0x00BE	16	0x0000
Dedicated comparison reference register WH	TMR4_SCCRwh	0x00C2	16	0x0000
Dedicated comparison reference register WL	TMR4_SCCRwl	0x00C6	16	0x0000
Dedicated control status register UH	TMR4_SCSRuh	0x00C8	16	0x0000
Dedicated control status register UL	TMR4_SCSRul	0x00CC	16	0x0000
Dedicated control status register VH	TMR4_SCSRvh	0x00D0	16	0x0000
Dedicated control status register VL	TMR4_SCSRvl	0x00D4	16	0x0000

Dedicated control status register WH	TMR4_SCSRwh	0x00D8	16	0x0000
Dedicated control status register WL	TMR4_SCSRwl	0x00DC	16	0x0000
Dedicated mode control register UH	TMR4_SCMRuh	0x00CA	16	0xFF00
Dedicated mode control register UL	TMR4_SCMRul	0x00CE	16	0xFF00
Dedicated mode control register VH	TMR4_SCMRvh	0x00D2	16	0xFF00
Dedicated mode control register VL	TMR4_SCMRvl	0x00D6	16	0xFF00
Dedicated mode control register WH	TMR4_SCMRwh	0x00DA	16	0xFF00
Dedicated mode control register WL	TMR4_SCMRwl	0x00DE	16	0xFF00
PWM basic control register U	TMR4_POCRu	0x0098	16	0xFF00
PWM basic control register V	TMR4_POCRv	0x009C	16	0xFF00
PWM basic control register W	TMR4_POCRw	0x00A0	16	0xFF00
PWM filtering control register U	TMR4_PFSRu	0x0082	16	0x0000
PWM filtering control register V	TMR4_PFSRv	0x008A	16	0x0000
PWM filtering control register W	TMR4_PFSRw	0x0092	16	0x0000
PWM Deadband Control Register AU	TMR4_PDARu	0x0084	16	0x0000
PWM Deadband Control Register BU	TMR4_PDBRu	0x0086	16	0x0000
PWM Deadband Control Register AV	TMR4_PDARv	0x008C	16	0x0000
PWM Deadband Control Register BV	TMR4_PDBRv	0x008E	16	0x0000
PWM Deadband Control Register AW	TMR4_PDARw	0x0094	16	0x0000
PWM Deadband Control Register BW	TMR4_PDBRw	0x0096	16	0x0000
Overload control status register	TMR4_RCSR	0x00A4	16	0x0000
EMB Control Status Register	TMR4_ECSR	0x00F0	16	0x0000
EMB Extended Control Register	TMR4_ECER	U1: (0x4005_5408) U2: (0x4005_540C) U3: (0x4005_5410)	32	0x0000_0000

Caution:

- In the following detailed description of the registers, m=uH, uL, vH, vL, wH, wL, n=u, v, w.
 m refers to the registers corresponding to the output control of ports
 TIM4_<t>_OUH, TIM4_<t>_OUL, TIM4_<t>_OVH, TIM4_<t>_WL, TIM4_<t>_OWH,
 TIM4_<t>_OWL, etc. when the function is implemented; n refers to the registers
 corresponding to the output control of ports TIM4_<t>_OUx, TIM4_<t>_OWL, etc.
 when the function is implemented. OVL, TIM4_<t>_OWH, TIM4_<t>_OWL output

control, etc.; the registers referred to by n correspond to TIM4_<t>_OUx, TIM4_<t>_OVx, TIM4_<t>_OWx output control, etc. at the time of function implementation, respectively. where x=H or L and the specific control of H or L has a corresponding symmetric bit in these registers.

19.5.1 Count value register (TMR4_CNTR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CNTR[15:0]															
position	Marker	Place Name		Function										Reading and writing	
b15~b0	CNTR[15:0]	Current value of the counter		This bit indicates the current counter count value while counting is in progress by writing a value to this register when counting is stopped Note: While counting, the value cannot be written to this register										R/W	

19.5.2 Cycle Reference Register (TMR4_CPSR)

Reset value: 0xFFFF

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CPSR [15:0]															
position	Marker	Place Name		Function										Reading and writing	
b15~b0	CPSR [15:0]	Common Cycle Baseline Values		Counting cycle value of the counter Note: When reading data from this address area, the value is not read from the buffer register, but from the CPSR register										R/W	

19.5.3 Control Status Register (TMR4_CCSR)

Reset value: 0x0040

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ECK EN	IRQ ZF	IRQ ZEN	-	-	-	IRQ PF	IRQ PEN	BUF EN	STOP	MODE	CLEAR				CKDIV[3:0]

position	Marker	Place Name	Function	Reading and writing
b15	ECKEN	Clock source selection	0: Internal PCLK1 clock 1: External TIM4_<t>_CLK port input clock Note 1: This bit is set when the counter is stopped Note 2: When using the external TIM4_<t>_CLK port input clock, after writing STOP="1", the first edge of the external input clock is considered invalid and the counting action Starting from the second edge, both the rising and falling edges are valid edges.	R/W
b14	IRQZF	Underflow status	0: No count underflow 1: Counting underflow occurs Note 1: When using the period interval response function, the setting condition of this bit is set by the period interval counter set by CVPR Note 2: When the counter is reset by the bus or CLEAR="1" is written, the IRQZF bit will Will not be set	R/W
b13	IRQZEN	Underflow interrupt enable	0: Disable IRQZF generation of interrupts to the CPU 1: Allow IRQZF to generate interrupts to the CPU	R/W
b12~b10	Reserved	-	Read "0", write "0" when writing	R/W
b9	IRQPF	Overflow status	0: No count overflow occurred 1: Count overflow occurs Note 1: When using the period interval response function, the setting condition of this bit is set by the period interval counter set by CVPR. Note 2: When the counter is reset by the bus or CLEAR="1" is written, the IRQZF bit will Will not be set	R/W
b8	IRQOPEN	Overflow interrupt enable	0: Disable IRQPF generated interrupts to the CPU 1: Allow IRQPF to generate interrupts to the CPU	R/W
b7	BUFEN	Cache Enable	0: Disable CPSR cache function 1: Enable CPSR cache function	R/W
b6	STOP	Counter Enable	0: Counter start 1: Counter stop	R/W
b5	MODE	Waveform mode	0: Sawtooth wave mode (upward counting only supported) 1: Triangular wave mode	R/W
b4	CLEAR	Counter zeroing	0: No operation 1: Counter zeroing Note: This bit is always 0 when read out	R/W

b3~b0	CKDIV	Counting clock division	This bit indicates the count clock division of the basic counter 0000: the count clock is PCLK1 0001: Counting clock is PCLK1/2 0010: Counting clock is PCLK1/4 0011: Counting clock is PCLK1/8 0100: Counting clock is PCLK1/16 0101: Counting clock is PCLK1/32	R/W
-------	-------	-------------------------	---	-----

0110: Counting clock is PCLK1/64

0111: Counting clock is PCLK1/128

1000: Counting clock is PCLK1/256

1001: Counting clock is PCLK1/512

1010: Counting clock is

PCLK1/1024 Please do not set

other values

Note: The count clock source is the external TIM4_<t>_CLK port

input clock, and the divider setting is invalid

19.5.4 Valid Period Register (TMR4_CVPR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PIC[3:0]				ZIC [3:0]				PIM[3:0]				ZIM[3:0]			
position	Marker		Place Name				Function				Reading and writing				
b15~b12	PIC[3:0]		Overflow interrupt mask status				Current number of overflow interrupts to be blocked Note 1: If the PIM is rewritten when the counter is stopped, the PIC is immediately updated to the new PIM value Note 2: If the PIM is rewritten before the overflow occurs after the count starts, the PIC is updated to the new PIM value at the time of the overflow, and the overflow interrupt is still triggered this time. The PIC decrements for each overflow thereafter Note 3: If the PIM is rewritten in the overflow interrupt, the PIC is updated at the next overflow to the new PIM value and still triggers the next overflow interrupt. Thereafter, each overflow PIC decrements				R				
b11~b8	ZIC [3:0]		Underflow interrupt mask state				Current number of underflow interrupts to be blocked Note 1: If the ZIM is rewritten when the counter is stopped, the ZIC is immediately updated to the new ZIM value Note 2: If the ZIM is rewritten before the overflow occurs after the count starts, the ZIC is updated to the new ZIM value at the time of the overflow, and the overflow interrupt is still triggered. Thereafter, ZIC decreases with each overflow. Note 3: If the ZIM is overwritten in the overflow interrupt, the PIC is updated to the new ZIM value at the next overflow and the next overflow interrupt is still triggered. Thereafter, each overflow ZIC passes Less				R				
b7~b4	PIM[3:0]		Overflow interrupt mask setting				Set the number of overflow interrupts to be masked				R/W				
b3~b0	ZIM[3:0]		Underflow interrupt mask setting				Set the number of underflow interrupts to be masked				R/W				

19.5.5 General comparison reference register (TMR4_OCCRm)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
OCCR[15:0]															
position	Marker		Place Name				Function				Reading and writing				
b15~b0	OCCR[15:0]		Generic Comparison Benchmark Values				Generic Comparison Benchmark Values				R/W				

19.5.6 General Control Status Register (TMR4_OCSRn)

Reset value: 0xFF00

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0			
Reserved								OCFL	OCFH	OCIEL	OCIEH	OCPL	OCPH	OCEL	OCEH			
<hr/>																		
position	Marker	Place Name								Function								Reading and writing
b15~b8	Reserved	-								Read "1", write "1" when writing								R/W
b7	OCFL	Counting Match L								0: Counter count value is not equal to the OCCR_xI setting 1: The counter count value is equal to the OCCR_xI setting (x = u, v, w) Note: This bit must be valid when OCEL=1								R/W
b6	OCFH	Counting Match H								0: Counter count value is not equal to the OCCRxh setting 1: The counter count value is equal to the OCCRxh setting (x = u, v, w) Note: This bit must be valid when OCEH=1								R/W
b5	OCIEL	Count Match L interrupt enable								0: When OCFL is set, no interrupt occurs 1: Interrupt occurs when OCFL is set								R/W
b4	OCIEH	Count Match H interrupt enable								0: When OCFH is set, no interrupt occurs 1: Interrupt occurs when OCFH is set								R/W
b3	OCPL	Port status when OCEL=0								0: When OCEL=0, writing "0" to this bit will output low on TIM4_<t>_OxL 1: When OCEL=0, write "1" to this bit, then output high on TIM4_<t>_OxL (m = U, V, W) Note: When OCEL=1, the write operation is invalid								R/W
b2	OCPH	Port status when OCEH=0								0: When OCEH=0, writing "0" to this bit will output low on TIM4_<t>_OxH 1: When OCEH=0, write "1" to this bit, then TIM4_<t>_OxH will output high level. (m = U, V, W) Note: When OCEH=1, the write operation is invalid								R/W
b1	OCEL	Port output enable L								0: TIM4_<t>_OxL output is invalid, port status is determined by OCPL 1: TIM4_<t>_OxL output enable, port state is determined by OCMR_yI setting and OCFL state (x = U, V, W, y = u, v, w)								R/W
b0	OCEH	Port Output Enable H								0: TIM4_<t>_OxH output is invalid, port status is determined by OCPH 1: TIM4_<t>_OxH output enable, port state is determined by OCMR_yH setting and OCFH state (x = U, V, W, y = u, v, w)								R/W

19.5.7 General Extension Control Register (TMR4_OCERn)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	MCEC	MCEC	LMM	LMM	LMC	LMC	MLBUF	MHBUF	CLBUF	CHBUF				

position	Marker	Place Name	Function	Reading and writing
b15~b14	Reserved	-	Read "0", write "0" when writing	R/W
b13	MCECL	Upper overflow point match enable L	0: When the count value reaches the upper overflow point (CNTR=CPSR) and OCCRxi > CPSR , the comparison match L is prohibited 1: When the count value reaches the upper overflow point (CNTR=CPSR) and OCCRxi > CPSR , it is considered that a comparison match L occurs (x = u, v, w)	R/W
b12	MCECH	Upper overflow point matching enable H	0: When the count value reaches the upper overflow point (CNTR=CPSR) and OCCRxi > CPSR , the comparison match H is prohibited 1: When the count value reaches the upper overflow point (CNTR=CPSR) and OCCRxi > CPSR , it is considered that a comparison match H occurs (x = u, v, w)	R/W
b11	LMML	Cycle interval response function link L	0: The period interval response function link is invalid, and the cache transfer of OCMRxi is determined by the MLBUFEN setting 1: The period interval response function link is valid, and the cache transfer of OCMRxi must meet CVPR.PIC[3:0] = 0000 (when counting overflow) or CVPR.ZIC[3:0]=0000 (when counting down) (x = u, v, w)	R/W
b10	LMMH	Cycle interval response function link H	0: period interval response function link is invalid, cache transfer of OCMRxh is determined by MHBUFEN[1:0] setting 1: The period interval response function link is valid, and the cache transfer of OCMRxh must meet CVPR.PIC[3:0]=0000 (at count up overflow) or CVPR.ZIC[3:0]=0000 (at count down overflow) in addition to the MHBUFEN[1:0] setting (x = u, v, w)	R/W
b9	LMCL	Cycle interval response function link L	0: The period interval response function link is invalid, and the cache transfer of OCCRxi is determined by the CLBUFEN[1:0] setting 1: The period interval response function link is valid, and the cache transfer of OCCRxi must meet CVPR.PIC[3:0]=0000 (when counting overflow) or CVPR.ZIC[3:0]=0000 (when counting underflow) in addition to the CLBUFEN[1:0] setting (x = u, v, w)	R/W

b8	LMCH	Cycle interval response function link H	0: The period interval response function link is invalid, and the cache transfer of OCCRxh is determined by the CHBUFEN[1:0] setting 1: The period interval response function link is valid, and the cache transfer of OCCRxh must meet CVPR.PIC[3:0]=0000 (when counting up overflow) or CVPR.ZIC[3:0]=0000 (when counting down overflow) in addition to the CHBUFEN[1:0] setting (x = u, v, w)	R/W
----	------	---	---	-----

b7~b6	MLBUFEN[1:0]	OCMRxI cache transfer	00: The value of the OCMRxI cache register is written directly to OCMRxI 01: The value of OCMRxI cache register is written to OCMRxI when the count overflows 10: The value of the OCMRxI cache register is written to OCMRxI when the count overflows 11: The value of the OCMRxI cache register is written to OCMRxI when the count overflows or overflows (x = u, v, w)	R/W
b5~b4	MHBUFEN[1:0]	OCMRxh Cache Forwarding	00: The value of the OCMRxh cache register is written directly to OCMRxh 01: The value of OCMRxh cache register is written to OCMRxh when the count overflows 10: The value of OCMRxh cache register is written to OCMRxh when the count overflows 11: The value of the OCMRxh cache register is written to OCMRxh when the count overflows or overflows (x = u, v, w)	R/W
b3~b2	CLBUFEN[1:0]	OCCRxI cache transfer	00: The value of the OCCRxI cache register is written directly to OCCRxI 01: The value of OCCRxI cache register is written to OCCRxI when the count overflows 10: The value of OCCRxI cache register is written to OCCRxI when the count overflows 11: The value of OCCRxI cache register is written to OCCRxI when the count overflows or overflows (x = u, v, w)	R/W
b1~b0	CHBUFEN[1:0]	OCCRxh cache transfer	00: The value of the OCCRxh cache register is written directly to OCCRxh 01: The value of OCCRxh cache register is written to OCCRxh when the count overflows 10: The value of OCCRxh cache register is written to OCCRxh when the count overflows 11: The value of the OCCRxh cache register is written to OCCRxh when the count overflows or overflows (x = u, v, w)	R/W

19.5.8 General Purpose Mode Control Register (TMR4_OCMRm)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
OPN ZRH[1:0]	OPN PKH[1:0]	OP ZRH[1:0]	OP UCH[1:0]	OP PKH[1:0]	OP DCH[1:0]	OP ZRH	OCF UCH	OCF PKH	OCF DCH						

Note: This register bit description is used when OCMRuh, OCMRvh, OCMRwh

position	Marker	Place Name	Function	Reading and writing
b15~b14	OPNZRH[1:0]	Lower overflow point port status H	Condition: Count overflow & OCCRxh count mismatch (x=u, v, w) 00: TIM4_<t>_OyH port bits remain unchanged when condition is met 01: The TIM4_<t>_OyH port bit outputs high when the condition is met 10: When the condition is met, the TIM4_<t>_OyH port bit is output low 11: When the condition is met, the TIM4_<t>_OyH port bit output is reversed (y = U, V, W)	R/W
b13~b12	OPNPKH[1:0]	Upper overflow point port status H	Condition: Count overflow & OCCRxh count mismatch (x=u, v, w) 00: TIM4_<t>_OyH port bits remain unchanged when condition is met 01: The TIM4_<t>_OyH port bit outputs high when the condition is met 10: When the condition is met, the TIM4_<t>_OyH port bit is output low 11: When the condition is met, the TIM4_<t>_OyH port bit output is reversed (y = U, V, W)	R/W
b11~b10	OPZRH[1:0]	Lower overflow point port status H	Condition: Count overflow & OCCRxh count match (x=u, v, w) 00: TIM4_<t>_OyH port bits remain unchanged when condition is met 01: The TIM4_<t>_OyH port bit outputs high when the condition is met 10: When the condition is met, the TIM4_<t>_OyH port bit is output low 11: When the condition is met, the TIM4_<t>_OyH port bit output is reversed (y = U, V, W)	R/W
b9~b8	OPUCH[1:0]	Upward counting port status H	Condition: Counter counts up & OCCRxh counts match (x=u, v, w) 00: TIM4_<t>_OyH port bits remain unchanged when condition is met 01: The TIM4_<t>_OyH port bit outputs high when the condition is met 10: The TIM4_<t>_OyH port bit outputs low when the condition is met 11: When the condition is met, the TIM4_<t>_OyH port bit output is reversed (y = U, V, W)	R/W

b7~b6	OPPKH[1:0]	Upper overflow point port status H	Condition: Count overflow & OCCRxh count match (x=u, v, w) 00: TIM4_<t>_OyH port bits remain unchanged when condition is met 01: The TIM4_<t>_OyH port bit outputs high when the condition is met 10: When the condition is met, the TIM4_<t>_OyH port bit is output low 11: When the condition is met, the TIM4_<t>_OyH port bit output is reversed (y = U, V, W)	R/W
b5~b4	OPDCH[1:0]	Count down port status H	Condition: Counter counts down & OCCRxh counts match (x=u, v, w) 00: TIM4_<t>_OyH port bits remain unchanged when condition is met 01: The TIM4_<t>_OyH port bit outputs high when the condition is met 10: When the condition is met, the TIM4_<t>_OyH port bit is output low 11: When the condition is met, the TIM4_<t>_OyH port bit output is reversed (y = U, V, W)	R/W
b3	OCFZRH	Lower overflow point OCFH state H	Conditions: Count overflow & OCCRxh count match (x=u, v, w)	R/W

				0: The OCSR.OCFH bit remains unchanged when the condition is met 1: When the condition is met, the OCSR.OCFH position bit	
b2	OCFUCH	Upward counting OCFH status H	Condition: Counter counts up & OCCRxh counts match (x=u, v, w) 0: OCSR.OCFH bits remain unchanged when condition is met 1: When the condition is met, the OCSR.OCFH position bit	R/W	
b1	OCFPKH	Upper overflow point OCFH state H	Condition: Count overflow & OCCRxh count match (x=u, v, w) 0: OCSR.OCFH bits remain unchanged when condition is met 1: When the condition is met, the OCSR.OCFH position bit	R/W	
b0	OCFDCH	Count down OCFH status H	Condition: Counter counts down & OCCRxh counts match (x=u, v, w) 0: OCSR.OCFH bits remain unchanged when condition is met 1: When the condition is met, the OCSR.OCFH position bit	R/W	

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EOPN ZRL[1:0]		EOPN PKL[1:0]		EOP ZRL[1:0]		EOP UCL[1:0]		EOP PKL[1:0]		EOP DCL[1:0]		EOPN UCL[1:0]		EOPN DCL[1:0]	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
OPN ZRL[1:0]	OPN PKL[1:0]	OP ZRL[1:0]	OP UCL[1:0]	OP PKL[1:0]	OP DCL[1:0]	OPC ZRL	OPC UCL	OPC PKL	OPC DCL						

Note: This register bit description is used when **OCMRuI**, **OCMRvI**, **OCMRwI**

position	Marker	Place Name	Function	Reading and writing
b31~b30	EOPNZRL[1:0]	Expansion under overflow point port state L	Conditions: Count overflow & OCCRxi count mismatch & OCCRxh count match (x = u, v, w) 00: TIM4_<t>_OyL port bits remain unchanged when conditions are met 01: The TIM4_<t>_OyL port bit outputs high when the condition is met 10: The TIM4_<t>_OyL port bit outputs low when the condition is met 11: When the condition is met, the TIM4_<t>_OyL port bit output is reversed (y = U, V, W)	R/W
b29~b28	EOPNPKL[1:0]	Expansion on the overflow point port state L	Conditions: Count overflow & OCCRxi count mismatch & OCCRxh count match (x = u, v, w) 00: TIM4_<t>_OyL port bits remain unchanged when conditions are met 01: The TIM4_<t>_OyL port bit outputs high when the condition is met 10: The TIM4_<t>_OyL port bit outputs low when the condition is met 11: When the condition is met, the TIM4_<t>_OyL port bit output	R/W

is reversed
 $(y = U, V, W)$

b27~b26	EOPZRL[1:0]	Expansion under the overflow point port state L	Conditions: Count Overflow & OCCRxi Count Match & OCCRxh Count Match $(x = u, v, w)$ 00: TIM4_<t>_OyL port bits remain unchanged when conditions are met 01: The TIM4_<t>_OyL port bit outputs high when the condition is met 10: The TIM4_<t>_OyL port bit outputs low when the condition is met 11: When the condition is met, the TIM4_<t>_OyL port bit output is reversed $(y = U, V, W)$	R/W
b25~b24	EOPUCL[1:0]	Expansion up count port status L	Conditions: Counter up count & OCCRxh count match & OCCRxh count match ($x=u, v, w$) 00: TIM4_<t>_OyL port bit remains unchanged when the condition is met 01: The TIM4_<t>_OyL port bit outputs high when the condition is met 10: The TIM4_<t>_OyL port bit outputs low when the condition is met 11: When the condition is met, the TIM4_<t>_OyL port bit output is reversed $(y = U, V, W)$	R/W
b23~b22	EOPPKL[1:0]	Expansion on the overflow point port state L	Conditions: Count overflow & OCCRxi count match & OCCRxh count match $(x = u, v, w)$ 00: TIM4_<t>_OyL port bits remain unchanged when conditions are met 01: The TIM4_<t>_OyL port bit outputs high when the condition is met 10: The TIM4_<t>_OyL port bit outputs low when the condition is met 11: When the condition is met, the TIM4_<t>_OyL port bit output is reversed $(y = U, V, W)$	R/W
b21~b20	EOPDCL[1:0]	Expansion down counting port status L	Conditions: Counter down count & OCCRxh count match & OCCRxh count match ($x=u, v, w$) 00: TIM4_<t>_OyL port bit remains unchanged when the condition is met 01: The TIM4_<t>_OyL port bit outputs high when the condition is met 10: The TIM4_<t>_OyL port bit outputs low when the condition is met 11: When the condition is met, the TIM4_<t>_OyL port bit output is reversed $(y = U, V, W)$	R/W

b19~b18	EOPNUCL[1:0]	Expansion up count port status L	Conditions: Counter count up & OCCRxI count mismatch & OCCRxh count match ($x=u, v, w$) 00: TIM4_<t>_OyL port bits remain unchanged when the condition is met 01: The TIM4_<t>_OyL port bit outputs high when the condition is met 10: The TIM4_<t>_OyL port bit outputs low when the condition is met 11: When the condition is met, the TIM4_<t>_OyL port bit output is reversed ($y = U, V, W$)	R/W
b17~b16	EOPNDCL[1:0]	Expansion down counting port status L	Conditions: Counter count down & OCCRxI count mismatch & OCCRxh count match ($x=u, v, w$) 00: TIM4_<t>_OyL port bit remains unchanged when the condition is met 01: The TIM4_<t>_OyL port bit outputs high when the condition is met 10: The TIM4_<t>_OyL port bit outputs low when the condition is met 11: When the condition is met, the TIM4_<t>_OyL port bit output is reversed ($y = U, V, W$)	R/W
b15~b14	OPNZRL[1:0]	Lower overflow point port status L	Conditions: Count overflow & OCCRxI count mismatch & OCCRxh count mismatch ($x = u, v, w$) 00: TIM4_<t>_OyL port bits remain unchanged when conditions are met 01: The TIM4_<t>_OyL port bit outputs high when the condition is met 10: The TIM4_<t>_OyL port bit outputs low when the condition is met 11: When the condition is met, the TIM4_<t>_OyL port bit output is reversed ($y = U, V, W$)	R/W
b13~b12	OPNPKL[1:0]	Upper overflow point port status L	Conditions: Count overflow & OCCRxI count mismatch & OCCRxh count mismatch ($x = u, v, w$) 00: TIM4_<t>_OyL port bits remain unchanged when conditions are met 01: The TIM4_<t>_OyL port bit outputs high when the condition is met 10: The TIM4_<t>_OyL port bit outputs low when the condition is met 11: When the condition is met, the TIM4_<t>_OyL port bit output is reversed ($y = U, V, W$)	R/W
b11~b10	OPZRL[1:0]	Lower overflow point port status L	Conditions: Count overflow & OCCRxI count match & OCCRxh count mismatch ($x = u, v, w$) 00: TIM4_<t>_OyL port bits remain unchanged when conditions are met 01: The TIM4_<t>_OyL port bit outputs high when the condition is met 10: The TIM4_<t>_OyL port bit outputs low when the condition is met	R/W

11: When the condition is met, the TIM4_<t>_OyL port bit output is reversed
 (y = U, V, W)

b9~b8	OPUCL[1:0]	Upward counting port status L	Conditions: Counter count up & OCCRxI count match & OCCRxh count mismatch (x=u, v, w) 00: TIM4_<t>_OyL port bits remain unchanged when the condition is met 01: The TIM4_<t>_OyL port bit outputs high when the condition is met 10: The TIM4_<t>_OyL port bit outputs low when the condition is met 11: When the condition is met, the TIM4_<t>_OyL port bit output is reversed (y = U, V, W)	R/W
b7~b6	OPPKL[1:0]	Upper overflow point port status L	Conditions: Count overflow & OCCRxI count match & OCCRxh count mismatch (x = u, v, w) 00: TIM4_<t>_OyL port bits remain unchanged when conditions are met 01: The TIM4_<t>_OyL port bit outputs high when the condition is met 10: The TIM4_<t>_OyL port bit outputs low when the condition is met 11: When the condition is met, the TIM4_<t>_OyL port bit output is reversed (y = U, V, W)	R/W
b5~b4	OPDCL[1:0]	Count down port status L	Conditions: Counter count down & OCCRxI count match & OCCRxh count mismatch (x=u, v, w) 00: TIM4_<t>_OyL port bit remains unchanged when the condition is met 01: The TIM4_<t>_OyL port bit outputs high when the condition is met 10: The TIM4_<t>_OyL port bit outputs low when the condition is met 11: When the condition is met, the TIM4_<t>_OyL port bit output is reversed (y = U, V, W)	R/W
b3	OCFZRL	Lower overflow point OCFL state L	Condition: Count overflow & OCCRxI count match (x=u, v, w) 0: OCSR.OCFL bits remain unchanged when condition is met 1: When the condition is met, the OCSR.OCFL position bit	R/W
b2	OCFUCL	Upward counting OCFL status L	Condition: Counter counts up & OCCRxI counts match (x=u, v, w) 0: OCSR.OCFL bits remain unchanged when condition is met 1: When the condition is met, the OCSR.OCFL position bit	R/W
b1	OCFPKL	Upper overflow point OCFL state L	Condition: Count overflow & OCCRxI count match (x=u, v, w) 0: OCSR.OCFL bits remain unchanged when condition is met 1: When the condition is met, the OCSR.OCFL position bit	R/W

b0	OCFDCL	Count down OCFL status L	Condition: Counter counts down & OCCRxI counts match (x=u, v, w) 0: OCSR.OCFL bits remain unchanged when condition is met 1: When the condition is met, the OCSR.OCFL position bit	R/W
----	--------	-----------------------------	--	-----

Caution:

- When reading data from this address area, the value of the OCMR register is not read instead of the value of the buffer register.
- TIM4_<t>_OyL can be determined by the count value of **OCCRxI with the counter** (standalone operation mode) or by the count value of OCCRxh with the counter and the count value of **OCCRxI** with the counter (linked operation mode). By writing the same 12-bit value to **bits[31:20]** and **bits[15:4]** of register **OCMRxI** and writing '0000' to **OCCRxI[19:16]**, the output of TIM4_<t>_OyL will not be affected by OCCRxh, but only by **OCCRxI only**. This mode is called

The independent operation mode channel yH is determined by OCCRxh and channel yL is configured by **OCCRxi**. If the

The independent mode of operation for the above condition is the linked mode of operation
-----channel yL output is subject to both OCCRxh

and **OCCRxi**

Impact.

($x = u, v, w, y = U, V, W$)

19.5.9 Dedicated comparison reference register (TMR4_SCCRm)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SCCR[15:0]															
position	Marker	Place Name		Function										Reading and writing	
b15~b0	SCCR[15:0]	Dedicated comparison benchmark values		Dedicated comparison reference value (comparison start mode) or delayed reference value (delayed start mode) Note: When data is read from this address area, it is not read from the buffer register value, but rather the value of the SCCR register										R/W	

19.5.10 Dedicated Control Status Register (TMR4_SCSRm)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ZEN	UEN	PEN	DEN	-	-	EVT DS	EVT MS	-	-	LMC	EVTOS [2:0]	BUFEN[1:0]			
position	Marker	Place Name	Function											Reading and writing	
b15	ZEN	Lower overflow point	0: No EVT operation when count overflow											R/W	
		EVT enable	1: When counting down the overflow: EVT start output when EVTMS=0&SCCR compare match & SCMR set match EVT delay mode when EVTMS=1&OCCR compare match & SCMR set match												
			Start												
b14	UEN	Count Up EVT Enable	0: No EVT operation when counting up											R/W	
			1: When counting upward: EVT start output when EVTMS=0&SCCR compare match & SCMR set match EVT delay mode when EVTMS=1&OCCR compare match & SCMR set match												
			Start												
b13	PEN	Upper overflow point	0: No EVT operation when counting overflow											R/W	
		EVT enable	1: When counting overflow: EVT start output when EVTMS=0&SCCR compare match & SCMR set match EVT delay mode when EVTMS=1&OCCR compare match & SCMR set match												
			Start												
b12	DEN	Count down EVT enable	0: No EVT operation when counting down											R/W	
			1: When counting down: EVT start output when EVTMS=0&SCCR compare match & SCMR set match EVT delay mode when EVTMS=1&OCCR compare match & SCMR set match												
			Start												
b11~b10	Reserved	-	Read "0", write "0" when writing											R/W	
b9	EVTDS	EVT time delay object selection	0: In delayed start mode, OCCRxh is used as a delayed comparison match 1: In delayed start mode, OCCRxI is used as a delayed comparison match (x = u, v, w)											R/W	
			Note: This bit is not valid when EVTMS=0												
b8	EVTMS	EVT mode selection	0: Comparison start mode (triggered by the comparison result of CNTR and SCCR) 1: Delayed start mode (comparison of matching events triggered by SCCR delay)											R/W	
b7~b6	Reserved	-	Read "0", write "0" when writing											R/W	
b5	LMC	Cycle interval response function link	0: Cycle interval response function link is invalid, SCCR cache transmission is determined by BUFEN setting 1: Cycle interval response function link is valid, the cache transmission of SCCR must also meet CVPR.PIC[3:0]=0000 (when counting overflow) on top of the BUFEN setting or CVPR.ZIC[3:0]=0000 (when the count overflows)											R/W	

b4~b2	EVTOS [2:0]	EVT output selection	000: EVT output of Special Event 0 is valid 001: EVT output of Special Evnet 1 is valid 010: EVT output of Special Evnet 2 is valid 011: EVT output of Special Evnet 3 is valid 100: EVT output of Special Evnet 4 is valid 101: Special Evnet 5's EVT output is valid	R/W
-------	-------------	----------------------	--	-----

Please do not set other values				
b1~b0	BUFEN[1:0]	SCCR & SCMR cache transfer	00: The values of SCCR and SCMR cache registers are written directly to SCCR and SCMR 01: The values of SCCR and SCMR cache registers are written to SCCR and SCMR when the count overflows 10: The values of SCCR and SCMR cache registers are written to SCCR and SCMR when the count overflows 11: SCCR, SCMR cache register values are written when the count overflows or overflows SCCR, SCMR	R/W

19.5.11 Dedicated Mode Control Register (TMR4_SCMRm)

Reset value: 0xFF00

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								MPCE	MZCE	-	-	AMC [3:0]			

position	Marker	Place Name	Function	Reading and writing
b15~b8	Reserved	-	Read "1", write "1" when writing	R/W
b7	MPCE	Cycle interval response enable	0: Prohibit AMC and CVPR.PIC comparison 1: Enabling AMC and CVPR.PIC comparison	R/W
b6	MZCE	Cycle interval response enable	0: Prohibit AMC comparison with CVPR.ZIC 1: Enabling AMC and CVPR.ZIC comparison	R/W
b5~b4	Reserved	-	Read "0", write "0" when writing	R/W
b3~b0	AMC [3:0]	Dedicated event output cycle interval value	This bit sets the cycle interval value for the dedicated event output function, which is valid when AMC and CVPR.	R/W

Caution:

- When reading data from this address area, the value of the SCMR register is not read instead of the value of the buffer register.

19.5.12 PWM Basic Control Register (TMR4_POCRn)

Reset value: 0xFF00

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0							
Reserved								LVLS[1:0]		PWMMD[1:0]		-		DIVCK[2:0]								
position	Marker		Place Name				Function								Reading and writing							
b15~b8	Reserved		-				Read "1", write "1" when writing								R/W							
b7~b6	LVLS[1:0]		PWM output polarity control				00: Both TIM4_<t>_OxH and TIM4_<t>_OxL outputs are not inverted 01: Both TIM4_<t>_OxH and TIM4_<t>_OxL outputs are inverted 10: The output of TIM4_<t>_OxH is inverted, the output of TIM4_<t>_OxL is not inverted 11: The output of TIM4_<t>_OxH is not inverted and the output of TIM4_<t>_OxL Reversal								R/W							
b5~b4	PWMMD[1:0]		PWM output mode				00: Straight-through mode 01: Dead time timer mode 10: Dead-time timer filter mode 11: Set prohibition								R/W							
b3	Reserved		-				Read "0", write "0" when writing								R/W							
b2~b0	DIVCK[2:0]		Counting clock division				This bit indicates the count clock division of the filter counter and the dead counter 000: the count clock is PCLK1 001: Counting clock is PCLK1/2 010: Counting clock is PCLK1/4 011: Counting clock is PCLK1/8 100: Counting clock is PCLK1/16 101: Counting clock is PCLK1/32 110: Counting clock is PCLK1/64 111: Counting clock is PCLK1/128								R/W							

19.5.13 PWM Filter Control Register (TMR4_PFSRn)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PFSR[15:0]															
position	Marker	Place Name		Function										Reading and writing	
b15~b0	PFSR[15:0]	Initial value of filtering		Initial value of filter count Note: When the PWM waveform output mode is not selected as the deadband timer filter mode, the 16-bit filter counter is used as the 16-bit reload counter, when the 16-bit filter counter can periodically generate interrupt output, this function is not related to the PWM waveform generator function.										R/W	

19.5.14 PWM Deadband Control Register (TMR4_PDARn)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PDA/BR [15:0]															
position	Marker	Place Name		Function										Reading and writing	
b15~b0	PDA/BR [15:0]	Deadband initial value		Initial value of deadband count										R/W	

19.5.15 Reload Control Status Register (TMR4_RCSR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RTS W	RTE W	RTIC W	RTIF W	RTS V	RTE V	RTIC V	RTIF V	RTS U	RTE U	RTIC U	RTIF U	-	RTID W	RTID V	RTID U
<hr/>															
position	Marker	Place Name	Function												Reading and writing
b15	RTSW	Heavy load counter stop W	0: No operation 1: Stop reloading counter W and clear RTIFW Note: This bit is always 0 when read out												R/W
b14	RTEW	Heavy-duty counter start W	0: Heavy load counter W stops 1: Heavy duty counter W start												R/W
b13	RTICW	Clear Count Match Status W	0: No operation 1: Clear the RTIFW flag bit Note: This bit is always 0 when read out												R/W
b12	RTIFW	Counting Matching Status W	0: The reload counter count value does not match the PFSRw 1: The reload counter count value is matched with the PFSRw												R
b11	RTSV	Heavy load counter stop V	0: No operation 1: Stop reload counter V and clear RTIFV Note: This bit is always 0 when read out												R/W
b10	RTEV	Heavy load counter start V	0: Heavy load counter V stops 1: Heavy load counter V start												R/W
b9	RTICV	Clear Count Match Status V	0: No operation 1: Clear the RTIFV flag bit Note: This bit is always 0 when read out												R/W
b8	RTIFV	Counting match state V	0: The reload counter count value does not match the PFSRv occurrence 1: The reload counter count value is matched with the PFSRv occurrence comparison												R
b7	RTSU	Heavy load counter stop U	0: No operation 1: Stop reloading counter U and clear RTIFU Note: This bit is always 0 when read out												R/W
b6	RTEU	Heavy-duty counter start U	0: reload counter U stop 1: Heavy-duty counter U start												R/W
b5	RTICU	Clear Count Match Status U	0: No operation 1: Clear the RTIFU flag bit Note: This bit is always 0 when read out												R/W
b4	RTIFU	Counting Matching Status U	0: The reload counter count value does not match the PFSRu occurrence 1: The value of the reload counter counts match the PFSRu												R

b3	Reserved	-	Read '0', write "0" when writing	R/W
b2	RTIDW	Overload interrupt mask W	0: When reload function is valid, reload interrupt W output is valid 1: When reload function is valid, reload interrupt W output is invalid	R/W
b1	RTIDV	Heavy-duty interrupt mask V	0: When the reload function is valid, the reload interrupt V output is valid 1: When reload function is valid, reload interrupt V output is invalid	R/W
b0	RTIDU	Overload Interrupt Mask U	0: When reload function is valid, reload interrupt U output is valid 1: When reload function is valid, reload interrupt U output is invalid	R/W

19.5.16 EMB Control Status Register (TMR4_ECSR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
Reserved								HOLD	Reserved							
position	Marker		Place Name		Function								Reading and writing			
b15~b8	Reserved		-		Read '0', write "0" when writing								R/W			
b7	HOLD		PWM Hold		0: PWM normal output when EMB input event is detected 1: When an EMB input event is detected, the current PWM output state is maintained and will not change again								R/W			
b6~b0	Reserved		-		Read '0', write "0" when writing								R/W			

19.5.17 EMB Extended Control Register (TMR4_ECER)

Reset value: 0x0000

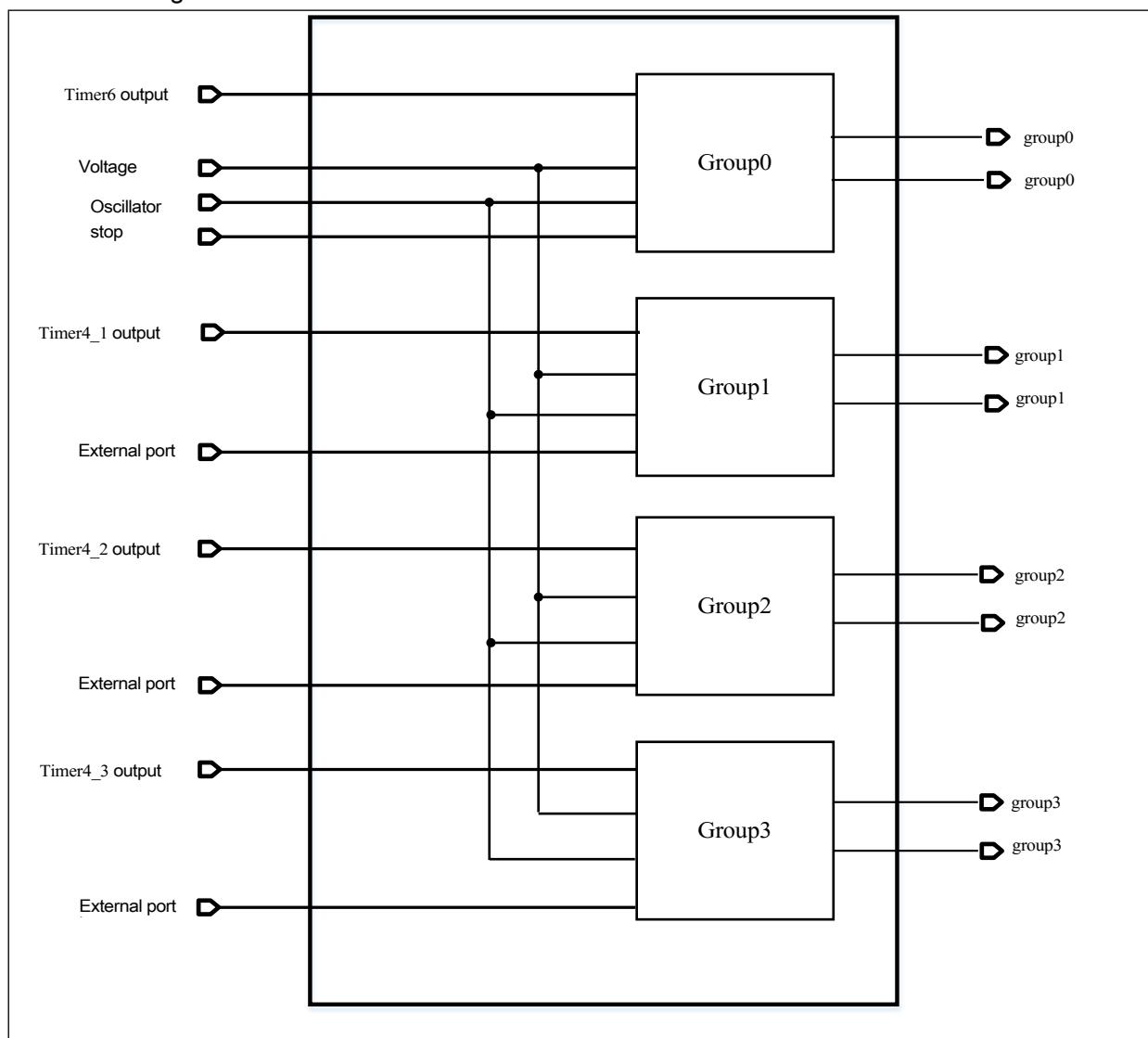
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								EMBVAL[1:0]							
position	Marker		Place Name		Function								Reading and writing		
b15~b2	Reserved		-		Read '0', write "0" when writing								R/W		
b1~b0	EMBVAL[1:0]		EMB Status Control		00: When an EMB event occurs, the PWM port state is determined by the ECSR.HOLD bit setting 01: PWM port output becomes Hiz when EMB event occurs 10: PWM port output is fixed low when an EMB event occurs 11: PWM port output is fixed high when EMB event occurs								R/W		

20 Emergency Brake Module (EMB)

20.1 Introduction

The emergency brake module is a functional module that notifies the timer when certain conditions are met so that the timer stops outputting PWM signals to the external motor, and the following events are used to generate the notification:

- External port input level change
- PWM output port level occurs in phase (same high or same low)
- Voltage comparator comparison results
- External oscillator stops oscillating
- Write register software control



20.2 Function Description

20.2.1 Overview

EMB is used to output a notification signal to the timer module (Timer4, Timer6) with PWM function when certain conditions are met to notify the timer module to turn off the current PWM output or correspond to 3 units of Timer4.

20.2.2 Stop PWM signal output when external port input level changes

Each cluster of the EMB is assigned one port to implement stopping the PWM signal output when the external port input level changes. The port assignments are shown in the table below.

Function Name	Function	Corresponding timer module
EMB_IN1	group 0 port input control signal	Timer6
EMB_IN2	group 1 port input control signal	Timer4_1
EMB_IN3	group 2 port input control signal	Timer4_2
EMB_IN4	group 3 port input control signal	Timer4_3

EMB can send a notification signal when the port level is high (INVSEL=0) or when the port level is low (INVSEL=1) according to the setting of control register EMB_CTLx (x=0~3). Also, each port is equipped with a digital filtering function, and the filtering strength can be set as required. When a qualified level change occurs on the port, the EMB generates a notification signal to Timer6 and Timer4, which can set the output port to high, low or high resistance state according to the register settings. The EMB is also capable of generating interrupt requests. When the port level becomes invalid, the user can clear the notification signal by writing the EMB status reset register (EMB_STATCLR_x) (x=0~3), which will enable Timer6 and Timer4 to resume outputting PWM waves.

20.2.3 Stop PWM signal output when the PWM output port level is in phase (same high or same low)

The EMB can monitor the complementary PWM output signals of Timer6 and Timer4, and generate a notification signal to Timer6 and Timer4 when the output signals are equal high or equal low. The EMB can also generate interrupt requests. Group0 can be used to monitor the complementary PWM output signals of Timer6, and groups1-3 can be used to monitor the complementary PWM output signals of Timer4_1/ **Timer4_2/** Timer4_3. When the same high or same low condition of the port is released, i.e. the PWMSF bit of the EMB Status Register (**EMB_STATx**) ($x=0\sim 3$) is reset, the user can clear the notification signal by writing the EMB Status Reset Register (**EMB_STATCLRx**) ($x=0\sim 3$) to restore the output PWM waveform of Timer6 and Timer4.

Port Name	Function	Corresponding groups	EMB_CTLx control bits	EMB_PWMLVx control bit
TIM6Am($m=1\sim 3$)	Timer6's complementary PWM output signal	group0	PWMSEN[2:0]	PWMLV[2:0]
TIM6Bm($m=1\sim 3$)				
TIM4OUH_1	Complementary PWM output signal of Timer4_1	group1	PWMSEN [2]	PWMLV [2]
TIM4OUL_1			PWMSEN[1]	PWMLV[1]
TIM4OVH_1			PWMSEN[0]	PWMLV[0]
TIM4OVL_1		group2	PWMSEN [2]	PWMLV [2]
TIM4OWH_1			PWMSEN[1]	PWMLV[1]
TIM4OWL_1			PWMSEN[0]	PWMLV[0]
TIM4OUH_2	Complementary PWM output signal of Timer4_2	group2	PWMSEN [2]	PWMLV [2]
TIM4OUL_2			PWMSEN[1]	PWMLV[1]
TIM4OVH_2			PWMSEN[0]	PWMLV[0]
TIM4OVL_2		group3	PWMSEN [2]	PWMLV [2]
TIM4OWH_2			PWMSEN[1]	PWMLV[1]
TIM4OWL_2			PWMSEN[0]	PWMLV[0]
TIM4OUH_3	Complementary PWM output signal of Timer4_3	group3	PWMSEN [2]	PWMLV [2]
TIM4OUL_3			PWMSEN[1]	PWMLV[1]
TIM4OVH_3			PWMSEN[0]	PWMLV[0]
TIM4OVL_3			PWMSEN [2]	PWMLV [2]
TIM4OWH_3			PWMSEN[1]	PWMLV[1]
TIM4OWL_3			PWMSEN[0]	PWMLV[0]

20.2.4 Stop PWM signal output according to the result of voltage comparator comparison

Each group of the EMB can send a notification signal to Timer6 and Timer4 based on the comparison result of the voltage comparator, see the Voltage Comparator chapter for the voltage comparator output result setting. The EMB generates a notification signal to Timer6 and Timer4 when the voltage comparator comparison result flag is in position and Timer6 and Timer4 can set the output port to high, low or high resistance depending on the register settings. The EMB can also generate interrupt requests. When the voltage comparator flag is reset, the user can clear the notification signal by writing the EMB status reset register (EMB_STATCLR x) ($x=0\sim3$), which will enable Timer6 and Timer4 to resume outputting PWM waves.

20.2.5 Stop PWM signal output when external oscillator stops oscillating

Each group of the EMB can send a notification signal to Timer6 and Timer4 when the external oscillator stops oscillating, see the Voltage Comparator section for the external oscillator stop setting. When the external oscillator stop flag is activated, the EMB generates a notification signal to Timer6 and Timer4. Timer6 and Timer4 can set the output port to high, low, or high resistance, depending on the register settings, after receiving the notification. The EMB is also capable of generating interrupt requests. When the external oscillator stop oscillation flag is reset, the user can clear the notification signal by writing the EMB status reset register (EMB_STATCLR x) ($x=0\sim3$), so that Timer6 and Timer4 can resume outputting PWM waves.

20.2.6 Write register software control PWM signal output

The software output enable control register (EMB_SOEx) ($x=0\sim3$) of EMB can allow users to send notification signals to Timer6 and Timer4 by software direct reset and reset, and software control of PWM output will not generate interrupt requests.

20.3 Register Description

Register List

Name	English Abbreviation	Description	Offset Address
EMB Output Control Register	EMB_CTL	Controls the conditions for each output enable	0x0
EMB feedback level selection register	EMB_PWMLV	Selects the valid level of the PWM feedback signal	0x4
EMB Software Output Enable Control Register	EMB_SOE	Software generates output ban events	0x8
EMB Status Register	EMB_STAT	Indicates the state of the output enable	0xC
EMB status reset register	EMB_STATCLR	Clear output enable state	0x10
EMB Interrupt License Register	EMB_INTEN	Control interrupt enable	0x14

20.3.1 EMB control register 0 (EMB_CTL0)

This register is a single write register, i.e. it can be written only once after reset.

Address: 0x4001_7C00

Reset value:

0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16			
INV SEL	NFE N	NFSEL[1: 0]		Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0			
Reserved								PWM SEN2	PWM SEN1	PWM SEN0	OSCS TPEN	Res -	CMP EN3	CMP EN2	CMP EN1	PORT INEN		

Bit tag	bit name	function	read and write	
b31	INVSEL	Port input active level selection	0: Active high 1: Active low	R/W
b30	NFEN	Port input digital filter enable	0: Filter is not valid 1: Filter is valid	R/W
b29~b28	Digital Filter Clock Selection		00: Use Bus Clock Filter 01: 8-division filtering using the bus clock 10: 32-division filtering using the bus clock 11: 128-division filtering using the bus clock	R/W
NFSEL[1:0]				
b27~b9	Reserved	reads 0 when	reading, please write 0	when writing
R				
b8	PWMSEN2	TIM6A/B_3 short-circuit output control enable	0: Invalid output control in case of short circuit	R/W
b7	PWMSEN1	TIM6A/B_2 short-circuit output control enable	1: Output control is valid in case of short circuit	R/W
b6	PWMSEN0	TIM6A/B_1 short-circuit output control enable	0: Invalid output control in case of short circuit 1: Output control is valid in case of short circuit	R/W
			0: Invalid output control in case of short circuit 1: Output control is valid in case of short circuit	
b5	OSCSTPEN when the oscillator stops oscillating	Oscillator stop output control enable	0: Output control is invalid 1: Output control is valid when the oscillator stops oscillating	R/W
b4	Reserved	reads 0 when	reading, please write 0	when writing
R				
b3	CMPEN3	CMP3 Voltage comparator comparison result control enable	b1	CMPEN1
b2	CMPEN2	CMP2 Voltage comparator comparison result control enable		CMP2 Voltage comparator comparison result output control is invalid 1: Voltage

comparator comparison result output control is valid		R/W R/W
0: The voltage comparator comparison result output control is invalid		
1: Voltage comparator comparison result output control is valid		
0: The voltage comparator comparison result output control is invalid	R/W	
1: Voltage comparator comparison result output control is valid		
b0	PORTINEN	port
		enable
		input
		control
		0: port input control disabled
		1: Port input control is valid

20.3.2 EMB control registers 1~3 (EMB_CTL1~3)

This register is a single write register, i.e. it can be written only once after reset.

Address: 0x4001_7C20, 0x4001_7C40, 0x4001_7C60

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
INVSEL	NFEN	NFSEL[1:0]		Reserved											
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								PWM SEN2	PWM SEN1	PWM SEN0	OSCST PEN	Res. EN3	CMP EN2	CMP EN1	PORT I NEN

position	Marker	Place Name	Function	Reading and writing
b31	INVSEL	Port input active level selection	0: High level active 1: Active low	R/W
b30	NFEN	Port input digital filter enable	0: Filter is invalid 1: The filter is effective	R/W
b29~b28	NFSEL[1:0]	Digital filter clock selection	00: Using bus clock filtering 01: 8-division filtering using the bus clock 10: 32-division filtering using the bus clock 11: 128-division filtering using the bus clock	R/W
b27~b9		Reserved	Read 0 when reading, please write 0 when writing	R
b8	PWMSEN2	U-phase short-circuit output control enable for TIM4_x (x=1~3)	0: Invalid output control in case of short circuit 1: Output control is valid in case of short circuit	R/W
b7	PWMSEN1	V-phase short-circuit output control enable for TIM4_x (x=1~3)	0: Invalid output control in case of short circuit 1: Output control is valid in case of short circuit	R/W
b6	PWMSEN0	W-phase short-circuit output control enable for TIM4_x (x=1~3)	0: Invalid output control in case of short circuit 1: Output control is valid in case of short circuit	R/W
b5	OSCSTPEN	Oscillator stop output control enable	0: Output control is invalid when the oscillator stops oscillating 1: Output control is valid when the oscillator stops oscillating	R/W
b4	Reserved	Read 0 when reading, please write 0 when writing		R
b3	CMPEN3	CMP3 Voltage comparator comparison result control enable	0: The voltage comparator comparison result output control is invalid 1: Voltage comparator comparison result output control is valid	R/W
b2	CMPEN2	CMP2 Voltage Comparator Comparison Result Control Enable	0: The voltage comparator comparison result output control is invalid 1: Voltage comparator comparison result output control is valid	R/W

b1	CMPEN1	CMP1 Voltage comparator comparison result control enable	0: The voltage comparator comparison result output control is invalid 1: Voltage comparator comparison result output control is valid	R/W
b0	PORTINEN	Port Input Control Enable	0: Port input control is invalid 1: Port input control is valid	R/W

20.3.3 EMB Feedback Level Select Register 0 (EMB_PWMLV0)

This register is a single write register, i.e. it can only be written once after reset

Address: 0x4001_7C04

Reset value:

0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
PWMLV2	PWMLV1	PWMLV0													

position	Marker	Place Name	Function	Reading and writing
b31~3	Reserved		Read 0 when reading, please write 0 when writing	R
b2	PWMLV2	TIM6A/B_3 output valid level selection	0: Low level is active level 1: High level is active level	R/W
b1	PWMLV1	TIM6A/B_2 output valid level selection	0: Low level is active level 1: High level is active level	R/W
b0	PWMLV0	TIM6A/B_1 output valid level selection	0: Low level is active level 1: High level is active level	R/W

20.3.4 EMB feedback level selection registers 1~3 (EMB_PWMLV1~3)

This register is a single write register, i.e. it can only be written once after reset

Addresses: 0x4001_7C24, 0x4001_7C44, 0x4001_7C64

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
PWML V2	PWML V1	PWML V0													

position	Marker	Place Name	Function	Reading and writing
b31~3	Reserved		Read 0 when reading, please write 0 when writing	R
b2	PWMLV2	Valid level selection for the U-phase output of TIM4_x (x=1~3)	0: Low level is active level 1: High level is active level	R/W
b1	PWMLV1	V-phase output valid level selection for TIM4_x (x=1~3)	0: Low level is active level 1: High level is active level	R/W
b0	PWMLV0	W-phase output valid level selection for TIM4_x (x=1~3)	0: Low level is active level 1: High level is active level	R/W

20.3.5 EMB software output enable control register (EMB_SOEx) (x=0~3)

Addresses: 0x4001_7C08, 0x4001_7C28, 0x4001_7C48, 0x4001_7C68

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															

position	Marker	Place Name	Function	Reading and writing
b31~1	Reserved		Read 0 when reading, please write 0 when writing	R
b0	SOE	Software controlled output	0: PWM normal output 1: PWM stop output	R/W

20.3.6 EMB Status Register (EMB_STATx) (x=0~3)

Addresses: 0x4001_7C0C, 0x4001_7C2C, 0x4001_7C4C, 0x4001_7C6C

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
PWM ST	PORT INST	OSF F	CMP F	PWM SF	POR TIN F										

position	Marker	Place Name	Function	Reading and writing
b31~6	Reserved		Read 0 when reading, please write 0 when writing	R
b5	PWMST	PWM output status	0: No PWM output in-phase occurs 1: Generate PWM output in phase	R
b4	PORTINST	Port Input Control Status	0: Port input control is inactive 1: Port input control is in active state	R
b3	OSF	Oscillator stop oscillation stop PWM output status	0: No PWM output is currently stopped because the oscillator has stopped oscillating 1: Current PWM output is stopped because the oscillator stops oscillating	R
b2	CMPF	State of voltage comparator stop PWM output	0: No PWM output is currently stopped due to voltage comparator comparison result 1: Current PWM output stopped due to voltage comparator comparison result	R
b1	PWMSF	PWM output in-phase stop Status of PWM output	0: No PWM output is currently stopped due to the PWM output feedback being in phase 1: The current PWM output is stopped due to the same phase of PWM output feedback	R
b0	PORTINF	Port input controls the state of the stop PWM output	0: No PWM output is currently stopped due to port input control 1: Current PWM output is stopped due to port input control	R

20.3.7 EMB Status Reset Register (EMB_STATCLR x) ($x=0\sim3$)

Addresses: 0x4001_7C10, 0x4001_7C30, 0x4001_7C50, 0x4001_7C70

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															

position	Marker	Place Name	Function	Reading and writing
b31~4	Reserved		Read 0 when reading, please write 0 when writing	R
		Reset	0: No effect 1: When CMU.MOSCSTP=0, clear the EMB_STAT.OSF bit and resume the PWM output that has been stopped due to the oscillator stopping oscillation	W
b3	OSFCLR	EMB_STAT.OSF		
b2	CMPFCLR	EMB_STAT.CMPF	0: No effect 1: When CMPMONn. OMON=0, clear the EMB_STAT. CMPF bit and resume the PWM output that was stopped due to the result of the voltage comparator comparison	W
b1	PWMSFCLR	EMB_STAT.PWMSF	0: No effect 1: When EMB_STAT.PWMST=0, clear the EMB_STAT. PWMSF bit and restore the Reset PWM output stopped due to PWM output feedback in phase	W
b0	PORTINFCLR	EMB_STAT.PORTINF	0: No effect 1: Clear EMB_STAT when EMB_STAT.PORTINST=0. PORTINF bit and resumes PWM output that was stopped due to port input control	W

20.3.8 EMB interrupt permission register (EMB_INTENx)(x=0~3)

Addresses: 0x4001_7C14, 0x4001_7C34, 0x4001_7C54, 0x4001_7C74

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
OSIN TEN	CMPI NTEN	PWMI NTEN	PORTI NTEN												

position	Marker	Place Name	Function	Reading and writing
b31~4	Reserved		Read 0 when reading, please write 0 when writing	R
b3	OSINTEN	Oscillator stop oscillation stop PWM interrupt enable	0: No interrupt is generated when the oscillator stops oscillating and stops PWM 1: Oscillator stops oscillating to stop PWM when generating interrupts	R/W
b2	CMPINTEN	Interrupt enable for voltage comparator stop PWM	0: No interrupt is generated when the voltage comparator comparison result stops PWM 1: Interrupt is generated when the voltage comparator comparison result stops PWM	R/W
b1	PWMINTEN	PWM output in-phase stop Interrupt enable for PWM	0: PWM output is in-phase to stop PWM without generating interrupt 1: PWM output in-phase stop PWM when generating interrupts	R/W
b0	PORTINTEN	Port input controls interrupt enable for stopping PWM	0: No interrupt is generated when the port input control stops PWM 1: Port input control generates an interrupt when stopping PWM	R/W

21 General-purpose timer (TimerA)

21.1 Introduction

TimerA is a timer with 16-bit count width and 8 PWM outputs. The timer supports two waveform modes, triangle waveform and sawtooth waveform, and can generate various PWM waveforms; it supports synchronous counter start; it supports cache function for the comparison reference register; it supports cascading between units to achieve 32-bit counting; it supports 2-phase quadrature coding counting and 3-phase quadrature coding counting. This product series is equipped with Load 6 units of TimerA for a maximum of 48 PWM outputs.

21.2 Basic Block Diagram

The basic functions and features of TimerA are shown in Table 21-1.

Table 21-1 Basic Functions and Features of TimerA

Waveform mode	Sawtooth wave, triangle wave
Basic Functions	— Additive, decremental counting direction
	— Synchronous start counter
	— Base value caching function
	-32-bit cascade counting
	— Orthogonal counts
	— 8-channel PWM output
	— Compare the output of matching events
Interrupt Type	— Compare Match Breaks
	— Periodic Matching Interruptions

The basic block diagram of TimerA is shown in Figure 21-1. In the diagram, "<t>" is the cell number, i.e., "<t>" is 1~6. When "<t>*" is mentioned later in this chapter, it refers to unit number, and will not repeat.

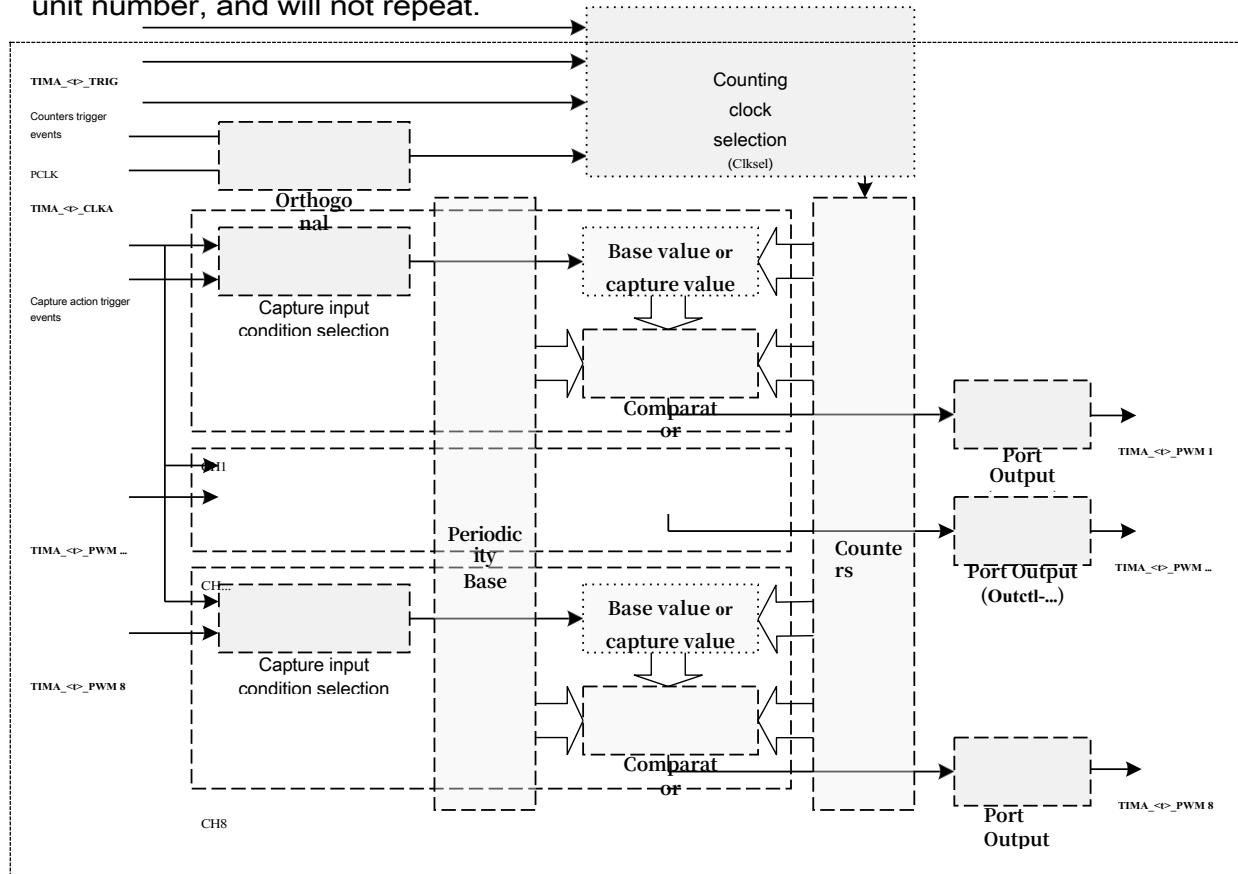


Figure 21-1 TimerA Basic Block Diagram

Table 21-2 shows the list of input and output ports of TimerA.

Table 21-2 TimerA Port List

Port Name	Direction	Function
TIMA_<t>.PWMM _m	in or out	Capture input event port or PWM output port ($m=1\sim 8$)
TIMA_<t>.CLKA	in	Orthogonal coded count event input port
TIMA_<t>.CLKB		
TIMA_<t>.TRIG	in	Hardware triggered start, stop, and clear event input port

21.3 Function Description

21.3.1 Basic movements

21.3.1.1 Waveform mode

TimerA has 2 basic counting waveform modes, sawtooth waveform mode and triangle waveform mode. The basic waveforms of the two waveform modes are shown in Figure 21-2 and Figure 21-3.

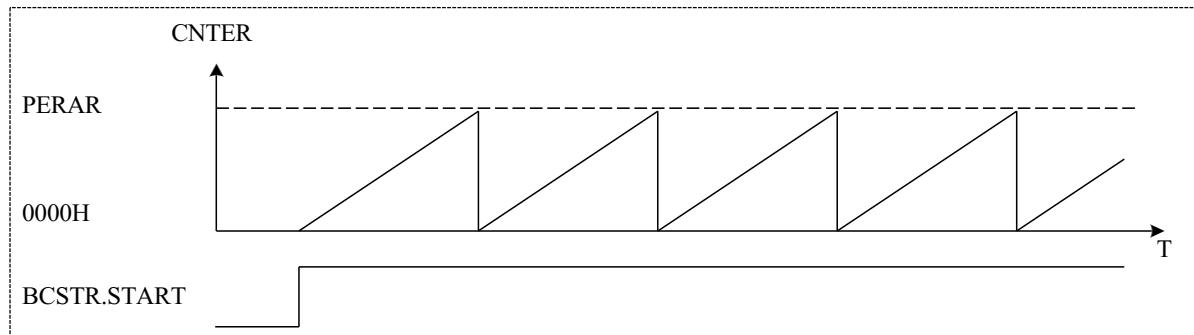


Figure 21-2 Sawtooth waveform (recursive counting)

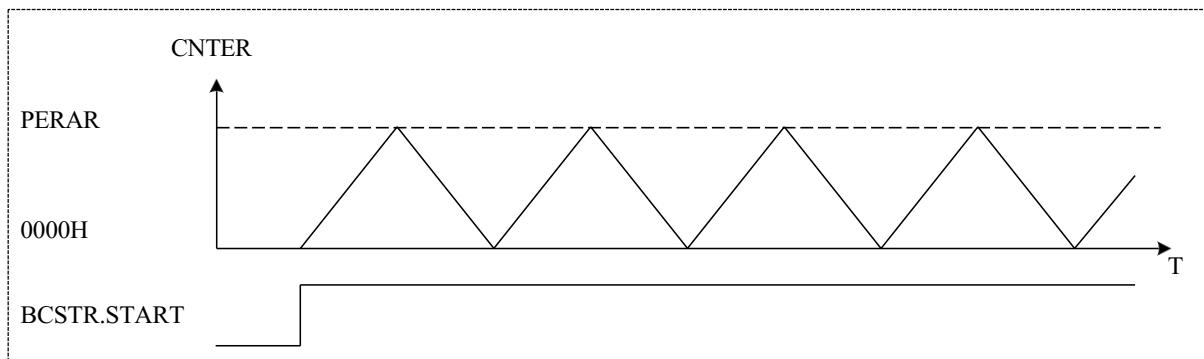


Figure 21-3 Triangular waveform

21.3.1.2 Compare Outputs

Each TimerA unit contains an internal 8-channel comparison output (TIMA_<t>_PWMn) that outputs the specified level when the count value matches the comparison reference value. the TMRA_CMPARn register corresponds to the count comparison reference value of the TIMA_<t>_PWMn output port. When the timer count value and TMRA_CMPARn are equal, the TIMA_<t>_PWMn port outputs the specified level

(n=1~8)

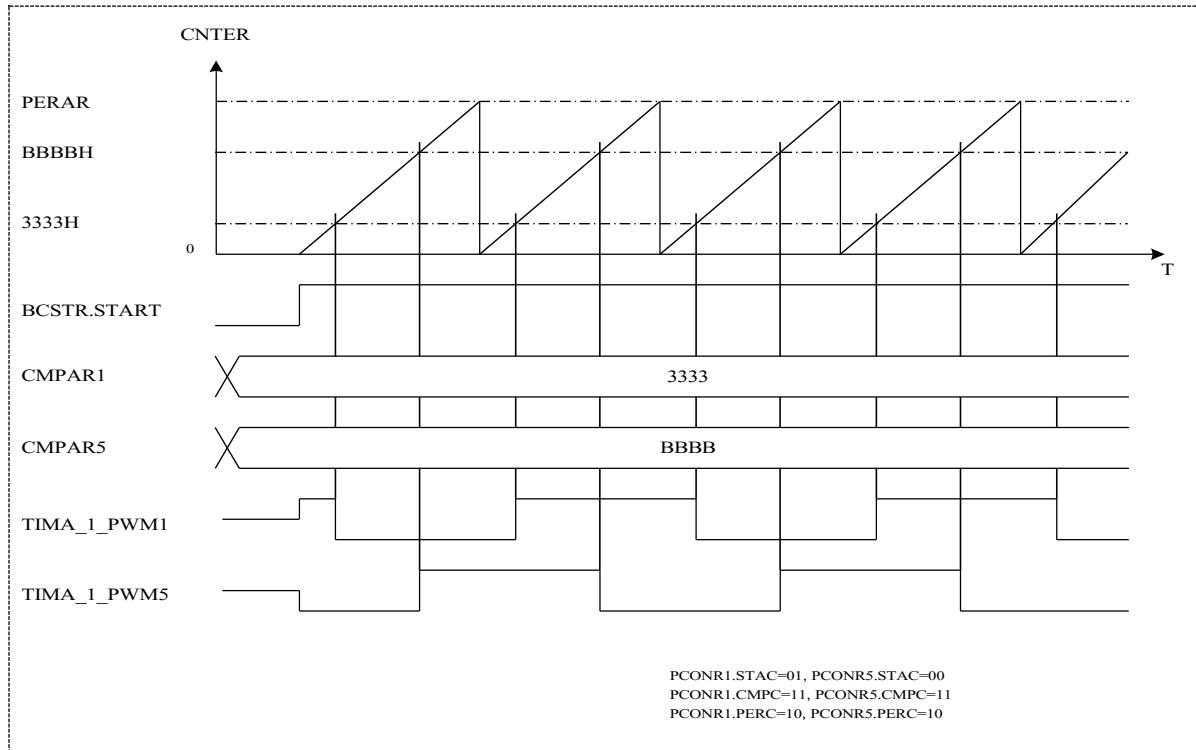


Figure 21-4 Compare Output Actions

The TIMA_<t>_PWMn port's level at count start, level at count stop, level at count compare match, and level at count cycle match can be controlled by setting the STAC, STPC, CMPC, PERC, and FORC bits in the port control register (PCONRn) (n=1 to 8) Figure 21-4 shows an example of the comparison output action of Unit 1.

21.3.1.3 Capture input

Each PWM output channel of each TimerA unit has a capture input function to save the captured count value. CAPMD bit in the Capture Control Register (CCONRn) is set to 1, and the capture input function becomes active. When the corresponding capture input condition is selected and the condition is valid, the current count value is saved in the corresponding register (CMPARn) (n=1~8)

The capture input conditions can be selected from internal capture action trigger events (selected via HTSSR1 register), TIMA_<t>_PWMn port input, etc. The specific condition selection can be set via the HICP bit of the Capture Control Register (CCONRn)(n=1~8)Figure 21-5

Example of a capture input action for cell 2.

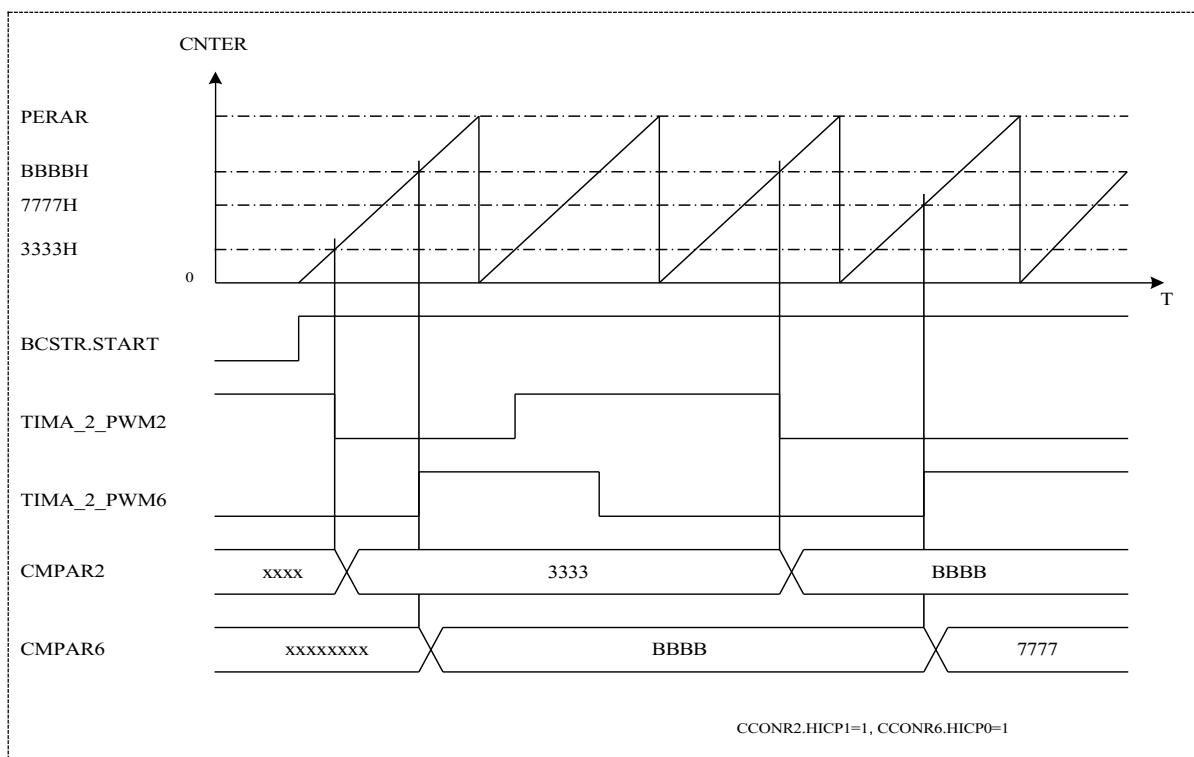


Figure 21-5 Capture Input Action

21.3.2 Clock source selection

TimerA's count clock can have the following options:

- a) 1248,1632,64,128,256,512,1024 divisions of PCLK1 (BCSTR.CKDIV[3:0])
(Setting)
- b) TIMA_<T>_TRIG Port event input (HCUPR[9:8] or HCDOR[9:8] setting)
- c) Internal counter trigger event input (HCUPR[10] or HCDOR[10] setting)
- d) Count overflow or count underflow event input for symmetric units (HCUPR[12:11] or HCDOR[12:11] setting)
- e) End port quadrature code input for TIMA_<T>_CLKA, TIMA_<T>_CLKB (HCUPR[7:0] or HCDOR[7:0] setting)

Counting clock source selection a is for software counting mode, and counting clock source selections b, c, d, and e are for hardware counting mode. The counting clock selection d is mostly used for the three-phase quadrature coded counting in the metric counting mode (see **【Position Overflow Counting】** and **【Mixed Counting】** sections) and can also be used for cascade counting. As can be seen from the above description, the b,c,d and e clocks are independent of each other and can be set valid or invalid respectively, and the a clock is automatically invalid when the b, c, d and e clocks are selected.

21.3.3 Synchronized start

TimerA of the 6 units equipped with this product can be synchronized with software start-up or hardware start-up. Unit 2 to Unit 6 can be selected to synchronize with Unit 1. When the BCSTR.SYNST bit in Unit 2 to Unit 6 is set to 1, the synchronous start function of the corresponding Unit and Unit 1 is available. START bit of Unit 1 is set to 1, the counter of the synchronized unit (Unit 2~Unit 6) starts software synchronous counting; if any bit of HCONR.HSTA1~0 of Unit 1 is set to 1 in hardware, and the corresponding hardware event of Unit 1 occurs, the counter of the synchronized unit (Unit 2~Unit 6) starts hardware synchronous counting. When the hardware synchronous counting start function is selected, the corresponding bits of HCONR.HSTA1~0 of the synchronized unit (Unit 2~Unit 6) must also be set to valid.

Figure 21-6 shows an example of software synchronization start when BCSTR.SYNST=1 is set for Unit 2, Unit 4, and Unit 5.

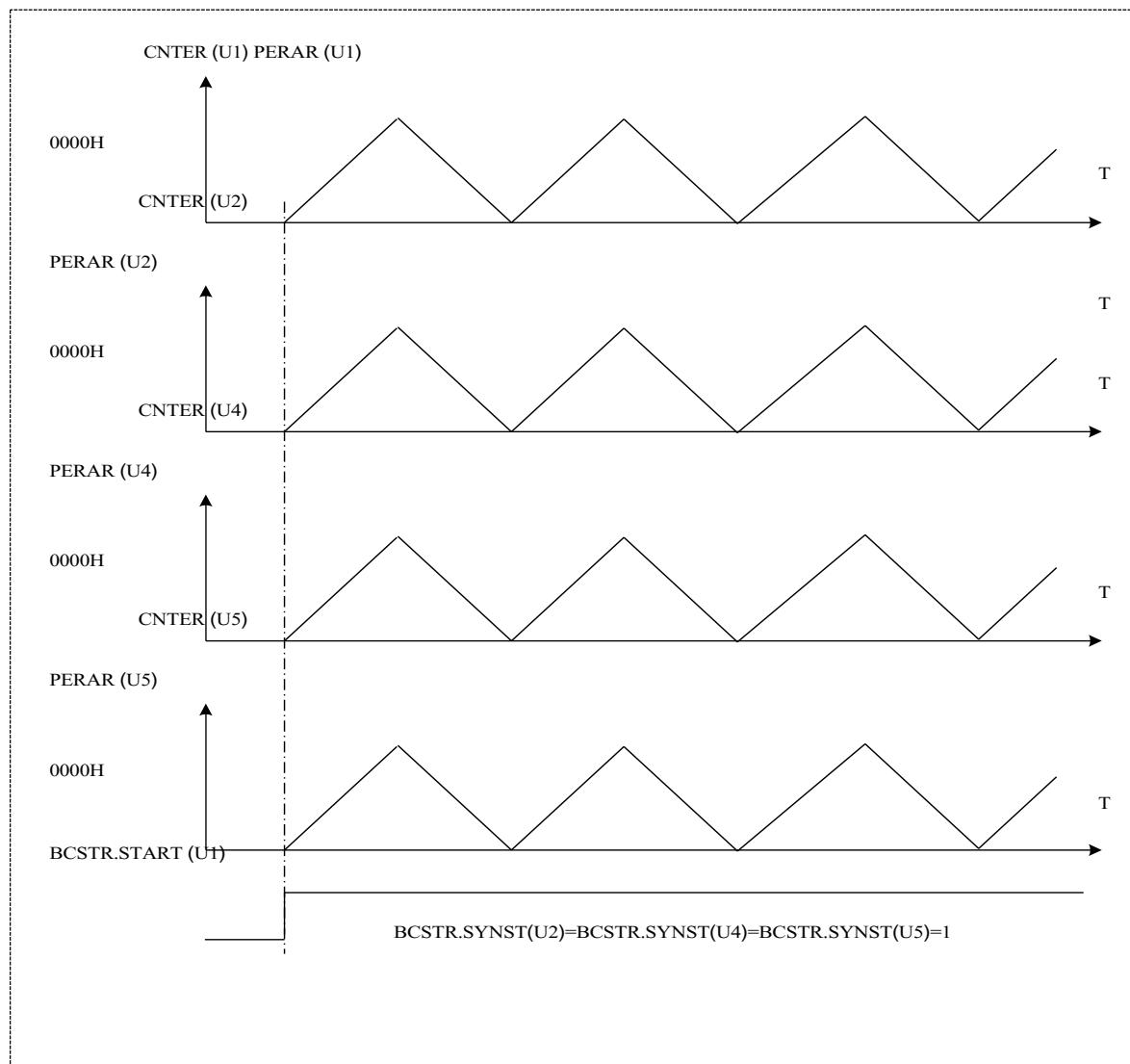


Figure 21-6 Software synchronization action

21.3.4 Digital filtering

The TIMA_<t>_CLKA, TIMA_<t>_CLKB, TIMA_<t>_TRIG, and TIMA_<t>_PWMn (at capture input function) port inputs of each unit have digital filtering functions. The enable of the filter function and the selection of the filter clock for each port can be achieved by setting the corresponding bits of the filter control register (FCONR) and the capture control register (CCONRn) (n=1~8).

After the filtered sampling reference clock samples 3 consistent levels on the port, the level is transmitted to the module internally as a valid level; a level less than

Three consistent levels are filtered out as external interference and are not transmitted internally to the module. Figure 21-7 shows an example of the CLKA port filtering action for Unit 1.

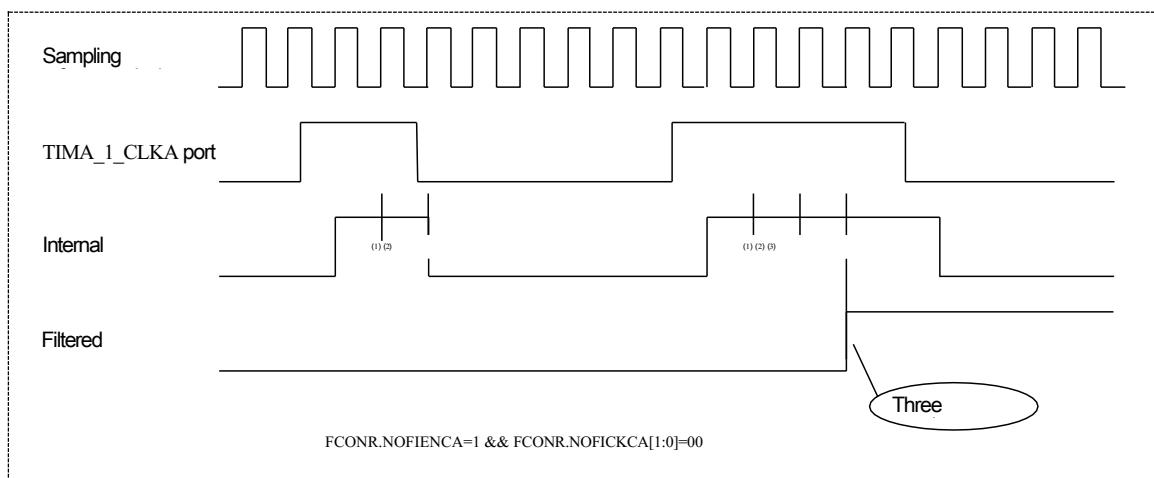


Figure 21-7 Filtering Function of Clock Input Port

21.3.5 Cache function

TimerA has a total of 8 comparison reference registers (CMPARn) that can be paired to implement the cache function ($n=1$ to 8). That is, CMPAR2 is used as the cache reference value for CMPAR1, CMPAR4 is used as the cache reference value for CMPAR3, CMPAR6 is used as the cache reference value for CMPAR5, and CMPAR8 is used as the cache reference value for CMPAR7. The cache control register (BCONRm) implements control of each of the four cache functions ($m=1\sim 4$)

The cache function becomes active when the BEN bit of the cache control register (BCONRm) is set ($m=1\sim 4$). A cache transfer occurs when the counter counts to a specific time point (CMPAR8/6/4/2->CMPAR7/5/3/1). This "**specific point**" in time can be in the following cases:

- a) Count to the upper overflow point (when BCSTR.DIR=1) or the lower overflow point (when BCSTR.DIR=0) in hardware count mode
- b) In ramp count mode (BCSTR.MODE=0), the counter counts to the upper overflow point (when BCSTR.DIR=1) or the lower overflow point (when BCSTR.DIR=0)
- c) Triangular wave counting mode (BCSTR.MODE=1) when counting to the peak point (BCSTR.
DIR=1&& BCONRn.BSE0=1)($n=1\sim 4$)
- d) Triangular wave counting mode (BCSTR.MODE=1) when counting to the valley point (BCSTR.
DIR=0&& BCONRn.BSE1=1)($n=1\sim 4$)
- e) Zeroing action occurs in hardware counting mode or sawtooth counting mode. Figure 21-8 below shows the cache transfer diagram for Unit 2 in sawtooth mode.

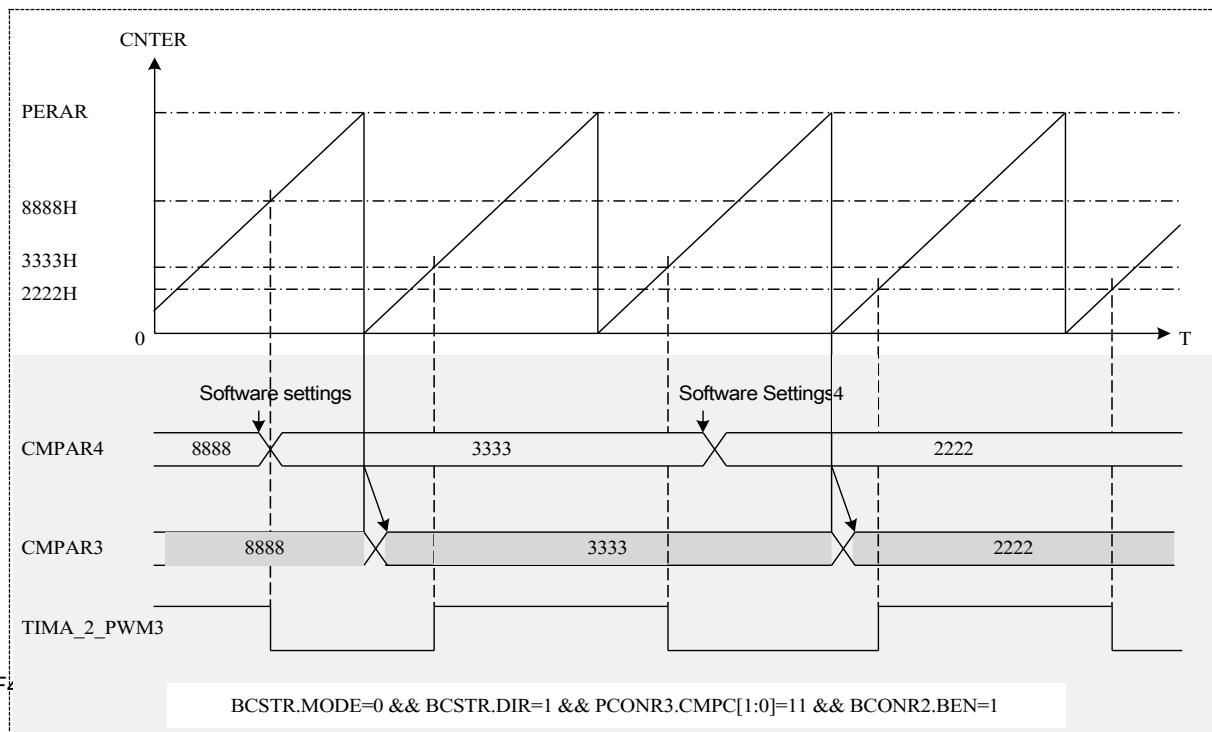


Figure 21-8 Cache action in sawtooth wave mode

21.3.6 Cascade Counting

In the Counting Clock Source Selection section, when the clock source is selected as d), the counting clock of this unit is selected as the counting overflow (counting up or counting down) event of the symmetrical unit, when the two units are cascaded and a 32-bit counter can be implemented. In cascaded counting, the CNTER of the symmetric unit is the low 16-bit counter and the CNTER of this unit is the high 16-bit counter.

For example, in delta wave up counting mode (BCSTR.MODE=0, BCSTR.DIR=1), set the counting clock of unit 1 to PCLK1 and set the counting clock source of unit 2 to the counting overflow event of unit 1 (TMRA_HCUPR.HCUP11=1 of unit 2) and start the counting of unit 1 and 2 (start unit 2 first, then start unit 1). The cascade count is achieved. The CNTER of cell 1 is a low 16-bit counter, and the CNTER of cell 2 is a high 16-bit counter. Figure 21-9 shows the diagram of the cascade counting of cells 1 and 2.

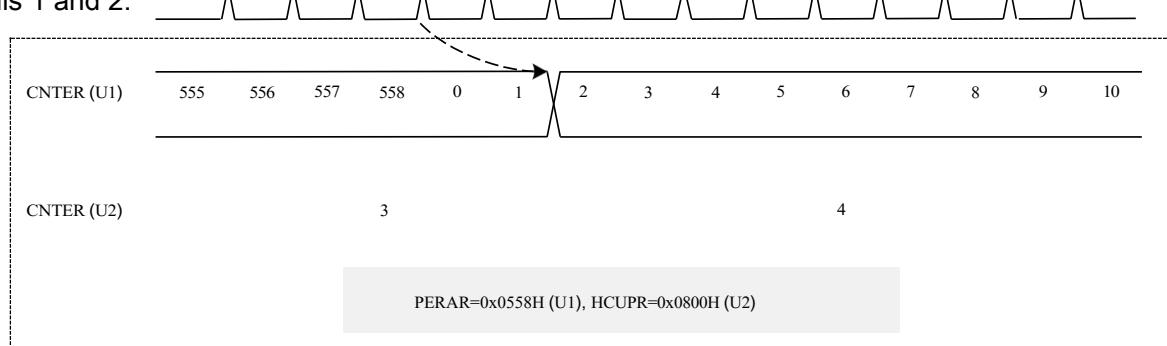


Figure 21-9 32-bit Cascade Counting Action

21.3.7 PWM Output

21.3.7.1 Universal PWM Output

Each of the 8 internal outputs of TimerA, TIMA_<t>_PWMn, can be implemented with different output waveforms ($n=1\sim 8$) via the relevant control bits in the Port Control Register (PCONRn).

Figure 21-10 shows the PWM output waveforms of channels 1, 2, 6 and 7 in the triangle mode of Unit 1.

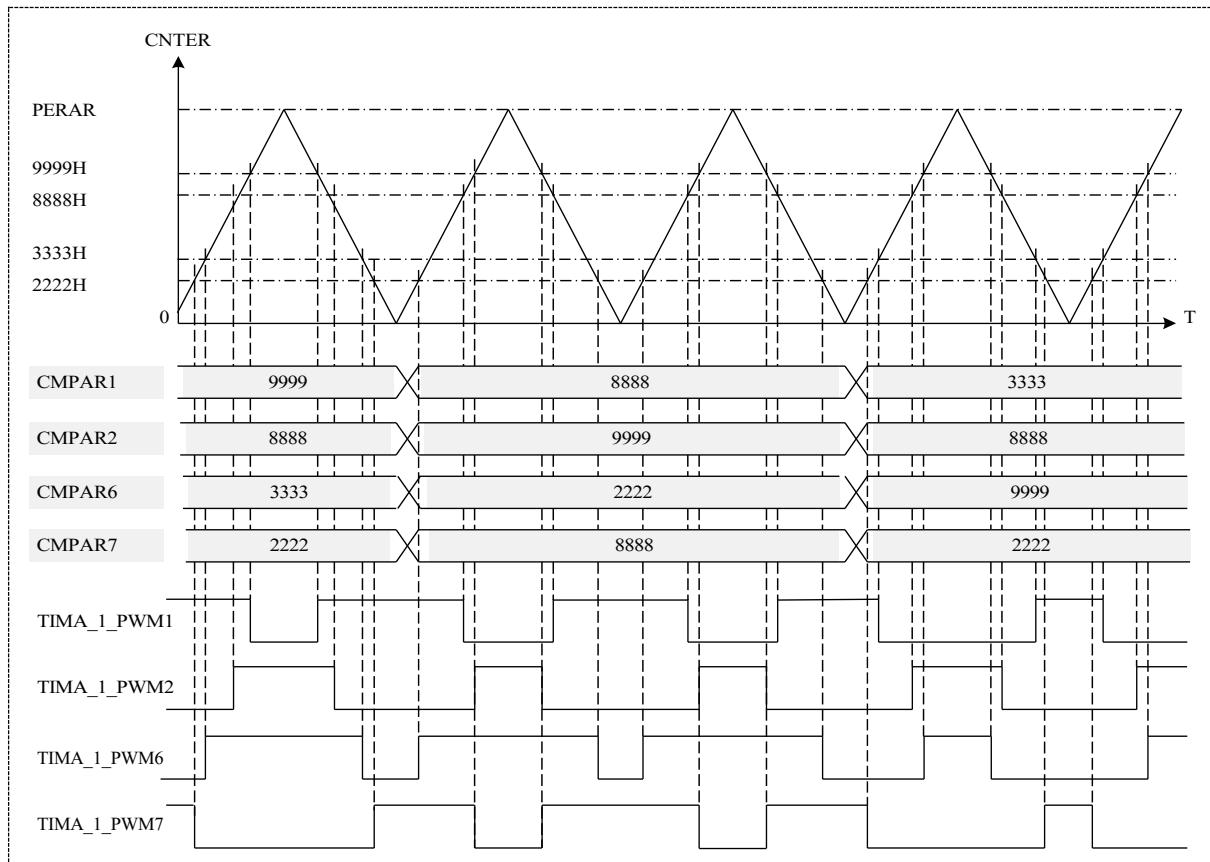


Figure 21-10 General PWM Output Example

21.3.7.2 Single pulse PWM output

In sawtooth counting mode, the PWM output that generates a single pulse in one cycle can be achieved by setting the flip when the comparison reference value is matched (PCONR_CMPC=11) and the flip when the period reference value is matched (PCONR.PERC=11)

Figure 21-12 shows an example of a single pulse PWM output waveform in Unit 1 triangle mode.

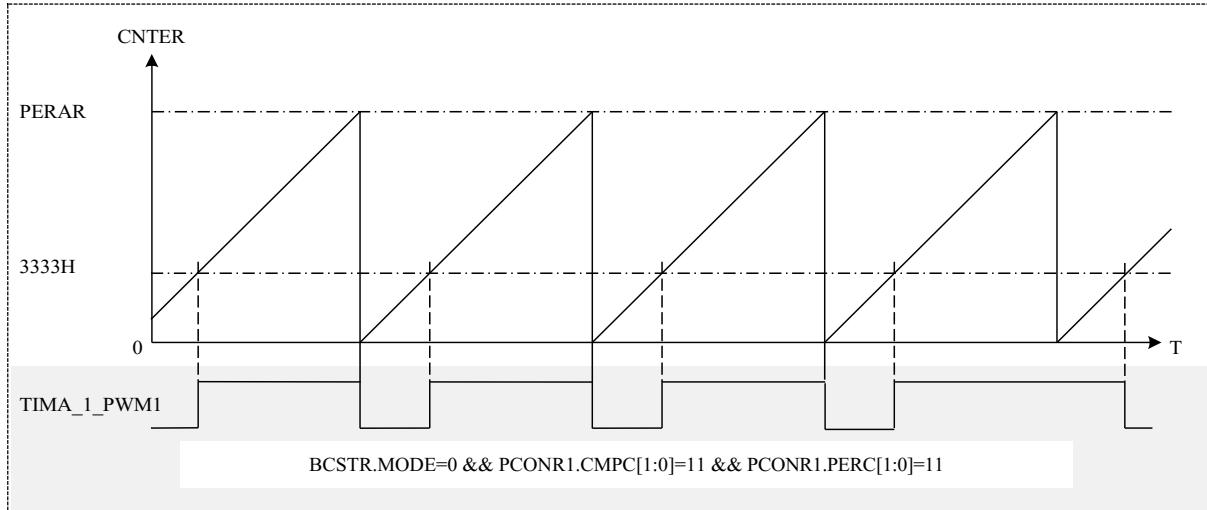


Figure 21-11 Single Pulse PWM Output

21.3.8 Orthogonal coding count

By treating the TIMA_<t>_CLKA input as an AIN input, the TIMA_<t>_CLKB input as a BIN input, and the TIMA_<t>_TRIG input as a ZIN input, TimerA can achieve quadrature coding of the three inputs.

The AIN and BIN of each unit alone can realize the position counting mode; the combination of AIN, BIN and ZIN of two units can realize the revolution counting mode, in which the unit for position counting is called position counting unit and the unit for revolution counting is called revolution counting unit. In the metric counting mode, every two units are combined with each other (unit 1, 2 combination; unit 3, 4 combination; unit 5, 6 combination), **and the position counting unit and the metric counting unit can be specified arbitrarily within the combination.**

The count conditions of AIN and BIN are enabled by setting the orthogonal relationship of TIMA_<t>_CLKA and TIMA_<t>_CLKB in the Hardware Incremental Event Selection Register (HCUPR) and Hardware Decremental Event Selection Register (HCDOR); the input action of ZIN is enabled by setting the hardware trigger event of the position counter unit. The input action of ZIN is realized by setting the zero enable bit of the zero select register (HCONR) of the position counting unit to clear the position timer, and by setting the hardware recursive event select register

(HCUPR) of the rotation unit to count the rotation timer.

21.3.8.1 Position counting mode

The orthogonal encoding position counting mode means that the basic counting function, phase difference counting function and direction counting function are implemented according to the input of AIN and BIN.

Basic Count

The basic counting action is based on the input clock of the AIN or BIN port, as shown in Figure 21-11 below.

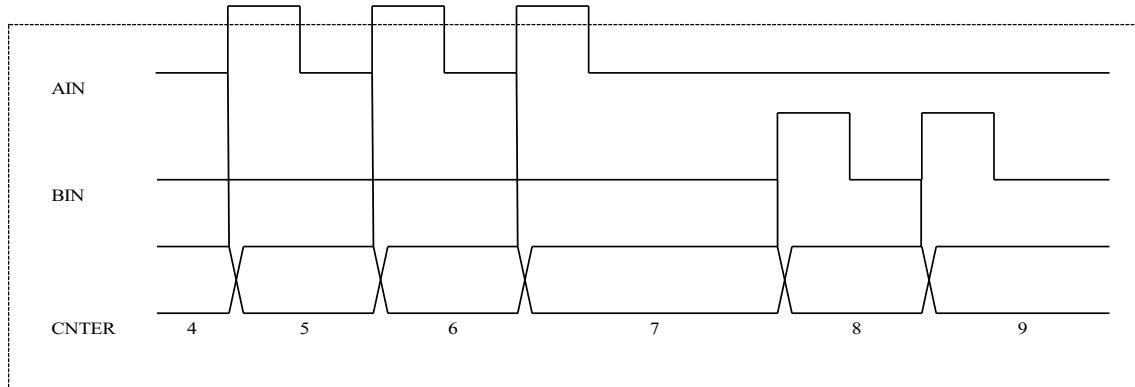


Figure 21-11 Position Mode - Basic Counting

Phase difference counting

Phase difference counting refers to counting based on the phase relationship between AIN and BIN. Depending on the setting, 1X counting, 2X counting, 4X counting, etc. can be realized as shown in Figure 21-12 to Figure 21-14 below.

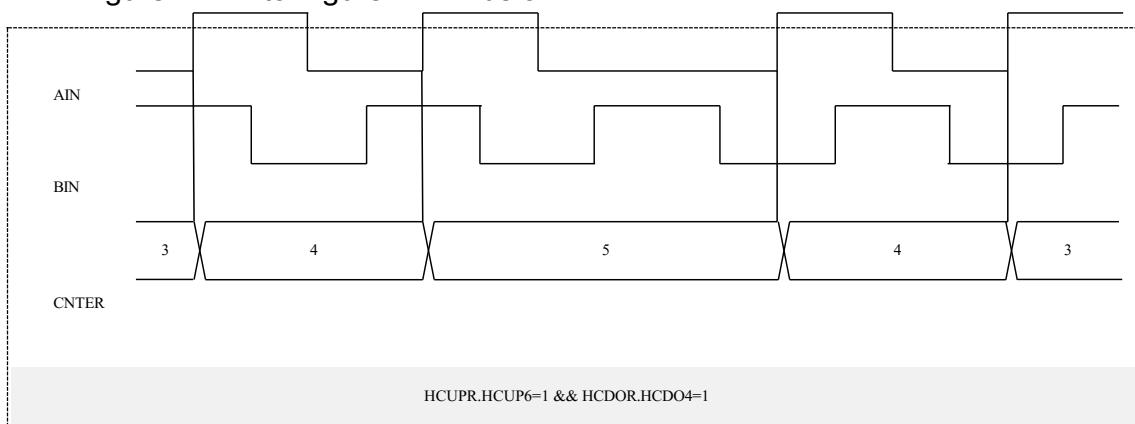


Figure 21-12 Position Counting Mode - Phase Difference Counting (1X Count)

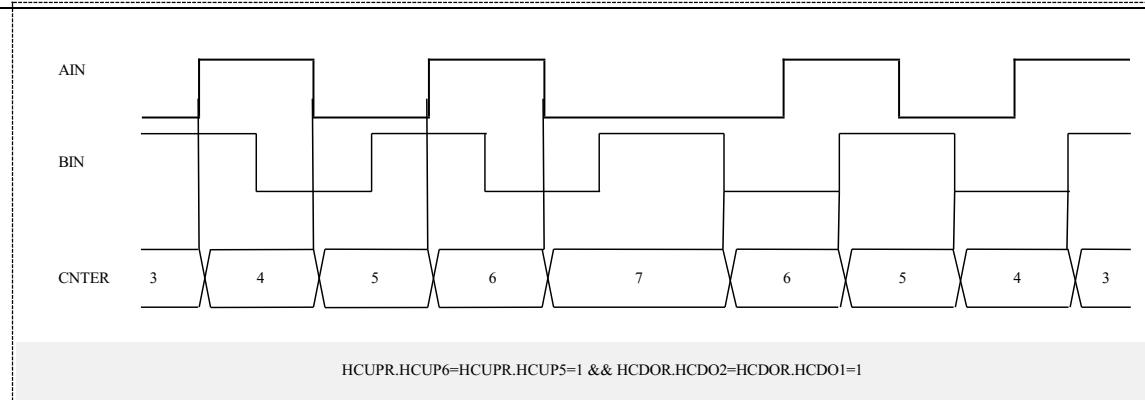


Figure 21-13 Position Counting Mode - Phase Difference Counting (2x Counting)

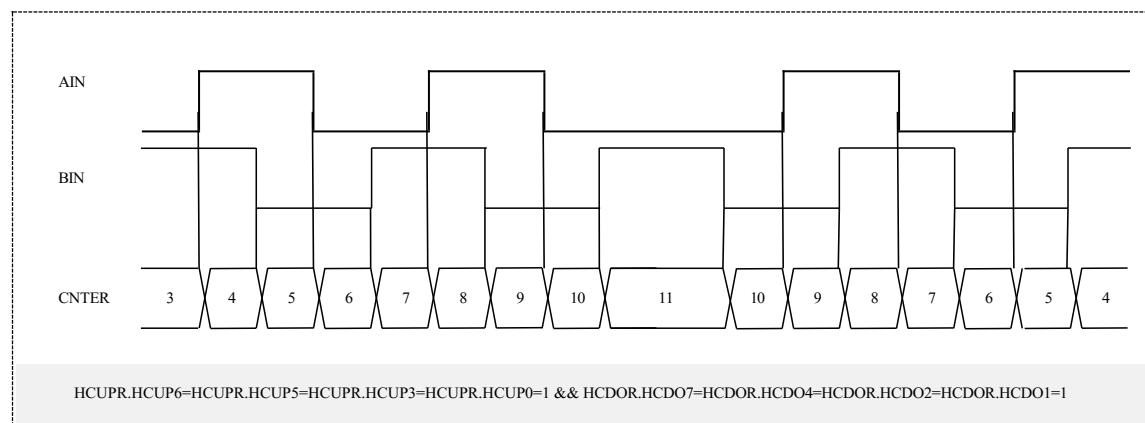


Figure 21-14 Position Counting Mode - Phase Difference Counting (4x Counting)

Direction Counting

Direction counting means setting the input state of AIN as direction control and the input of BIN as clock counting, as shown in Figure 21-15 below.

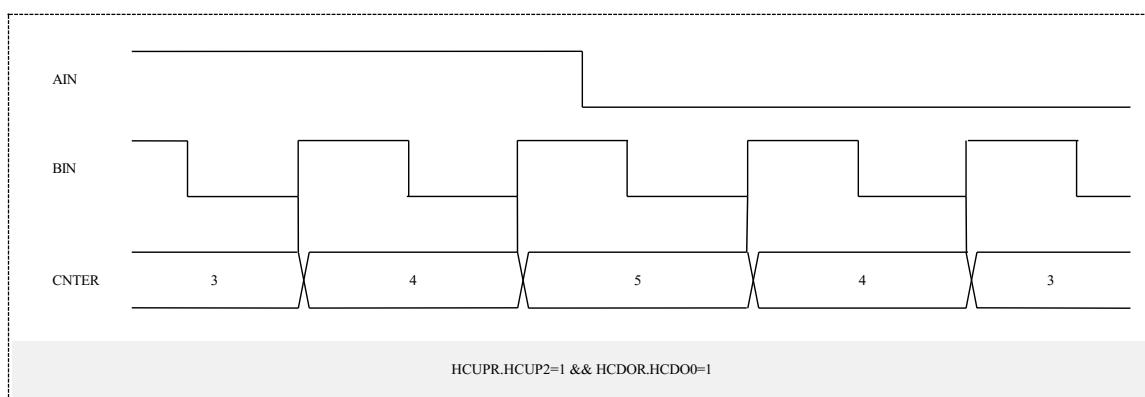


Figure 21-15 Position Count Mode - Direction Count

21.3.8.2 Rotation counting mode

The orthogonal coding revolution counting mode is to add ZIN input events to the AIN and BIN counting to judge the revolution number, etc. The Z-phase counting function, the position overflow counting function and the mixed counting function can be realized according to the counting method of the rotation timer in the rotation counting mode.

Z-phase count

Z-phase counting is a counting action in which the revolution counter unit counts according to the input of ZIN and the position counter unit is cleared to zero at the same time. As shown below.

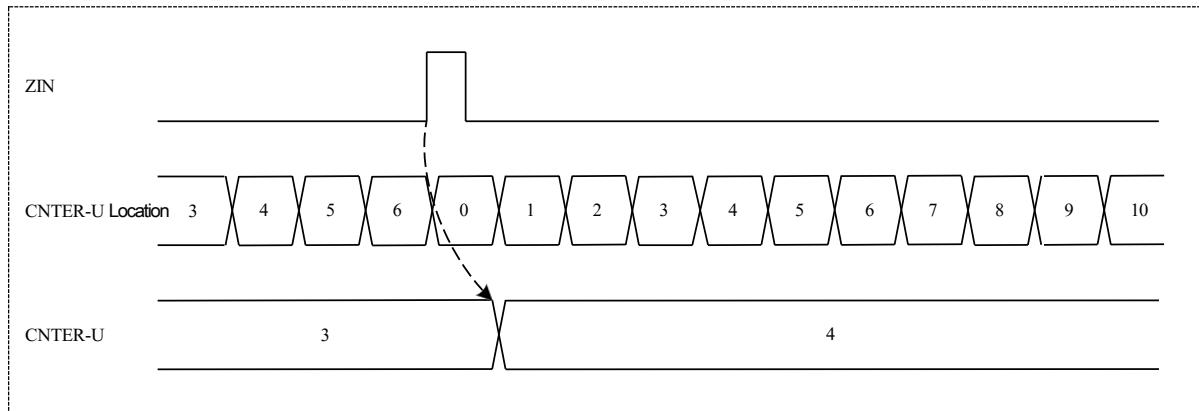


Figure 21-16 Rotation counting mode - Z-phase counting

Position Overflow Count

Position overflow counting means that an overflow event is generated when the count of the position counter unit overflows or underflows, which triggers the timer of the revolution counter unit to perform a count (the input of ZIN does not perform the count action of the revolution counter unit and the zero action of the position counter unit during this counting method)

The hardware incremental (decremental) event selection register (HCUPR or HCDOR) bits 12~11 of the rotary counter unit enable the incremental (decremental) event, and the overflow event of the position counter unit can trigger the rotary counter unit to achieve a count. As shown in Figure 21-17 below.

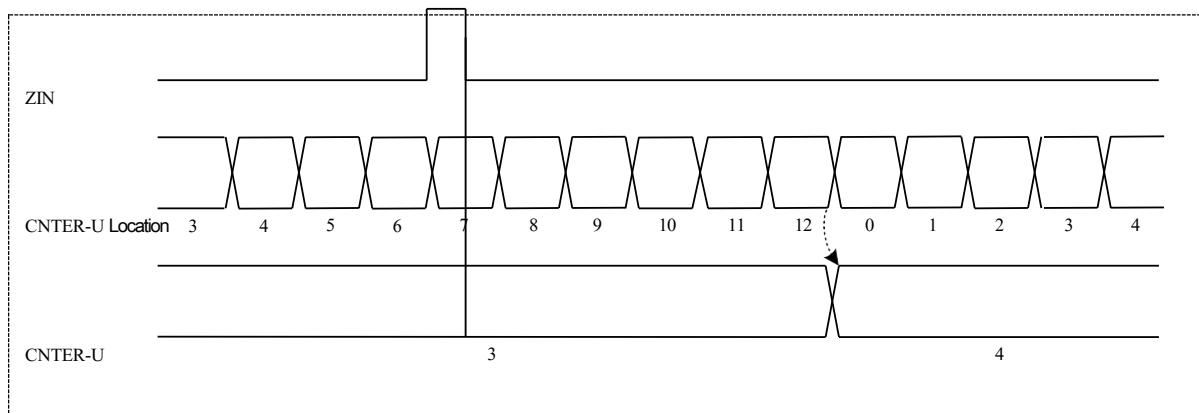


Figure 21-17 Rotation Counting Mode - Position Overflow Count

Mixed counting

Mixed counting is a counting action that combines the above two counting methods, Z-phase counting and position overflow counting, and its implementation is also a combination of the above two counting methods. As shown in Figure 21-18 below.

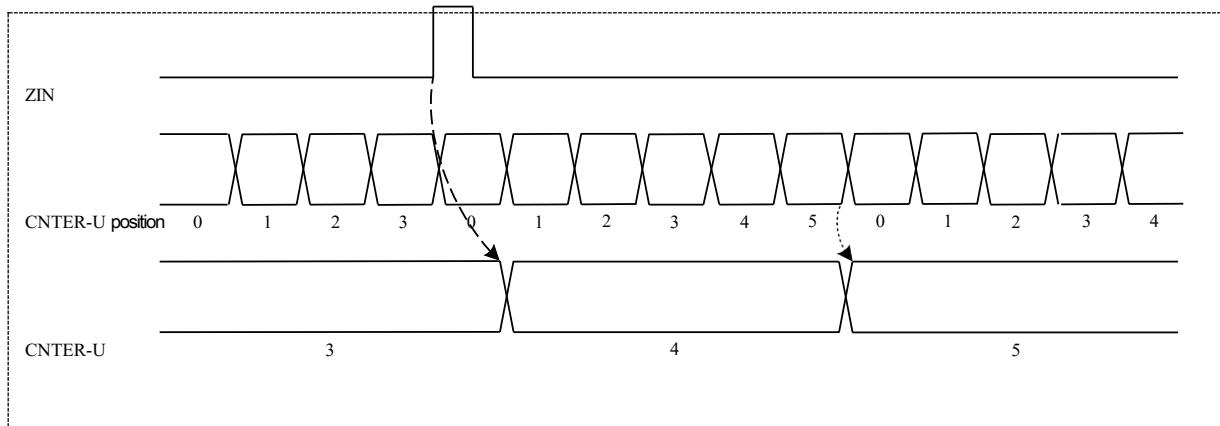


Figure 21-18 Rotation Counting Mode - Mixed Counting

21.4 Interrupts and event descriptions

TimerA contains 3 interrupt outputs and 3 event outputs, 1 compare match interrupt and event, and 2 cycle match interrupts and events.

21.4.1 Compare match interrupts and events

When a comparison match occurs between the comparison reference value register (CMPARn) and the count value comparison, the corresponding bit (STFLR.CMPFn) in the status flag register (STFLR) will be set to 1. At this time, if the corresponding bit (ICONR.ITEFn) in the interrupt control register (ICONR) is set to 1, the compare match interrupt request (TMRA_U<t>_CMP) will be triggered; if the corresponding bit (ECONR.ETENn) of the event control register (ECONR) is set to 1, the corresponding event request (TMRA_U<t>_CMP) will be triggered (n=1~8)

The capture input action occurs when the capture input valid condition selected by the capture control register (CCONRn) is generated. In this case, if the corresponding bit (ICONR.ITEFn) of the interrupt control register (ICONR) is set to 1, the corresponding interrupt request (TMRA_U<t>_CMP) is triggered; if the corresponding bit (ECONR.ETENn) of the event control register (ECONR) is set to 1, the corresponding event request

(TMRA_U<t>_CMP) will be triggered (n=1~8)

The compare match interrupt and compare match event for the 8 reference values inside each unit are not independent outputs, the compare match interrupt is aggregated into an interrupt output to the interrupt module via "or logic" (see INTC chapter). Compare match event is aggregated into an event output via "or logic". The compare match interrupt is aggregated into an event output via "or logic" to the interrupt module (see INTC chapter) and the compare match event is aggregated into an event output via "or logic" to select another module to trigger.

21.4.2 Cycle Matching Breaks and Events

The OVFF or UDFF bit of the Control Status Register (BCSTR) is set to 1 when the ramp mode counts up to the upper overflow point, the ramp mode counts down to the lower overflow point, and the triangle mode counts up to the valley or peak point. If the BCSTR.ITENOVF or BCSTR.ITENUDF bit is set to 1 to enable the interrupt, the cycle matching interrupt (TMRA_U<t>_OVF and TMRA_U<t>_UDF) can be triggered at the corresponding cycle point and output to the interrupt module (INTC). The cycle matching event (TMRA_U<t>_OVF and TMRA_U<t>_UDF) is triggered at the corresponding count cycle point and is used to select a trigger for another module.

21.5 Register Description

Table 21-3 shows the register list of the TimerA

module. BASE ADDR -

0x4001_5000 (U1) 0x4001_5400 (U2) 0x4001_5800 (U3)

0x4001_5C00 (U4) 0x4001_6000 (U5) 0x4001_6400 (U6)

Table 21-3 Register List

Register Name	Sym bols	Offset	Bit width	Reset value
General purpose count value register	TMRA_CNTER	0x0000	16	0x0000
Cycle reference value register	TMRA_PERAR	0x0004	16	0xFFFF
Compare reference value register 1	TMRA_CMPAR1	0x0040	16	0xFFFF
Compare reference value register 2	TMRA_CMPAR2	0x0044	16	0xFFFF
Compare reference value register 3	TMRA_CMPAR3	0x0048	16	0xFFFF
Compare reference value register 4	TMRA_CMPAR4	0x004C	16	0xFFFF
Compare reference value register 5	TMRA_CMPAR5	0x0050	16	0xFFFF
Compare reference value register 6	TMRA_CMPAR6	0x0054	16	0xFFFF
Compare reference value register 7	TMRA_CMPAR7	0x0058	16	0xFFFF
Compare reference value register 8	TMRA_CMPAR8	0x005C	16	0xFFFF
Control Status Register	TMRA_BCSTR	0x0080	16	0x0002
Interrupt control register	TMRA_ICONR	0x0090	16	0x0000
Event Control Register	TMRA_ECONR	0x0094	16	0x0000
Filter control register	TMRA_FCONR	0x0098	16	0x0000
Status Flag Register	TMRA_STFLR	0x009C	16	0x0000
Cache control register 1	TMRA_BCONR1	0x00C0	16	0x0000
Cache control register 2	TMRA_BCONR2	0x00C8	16	0x0000
Cache control register 3	TMRA_BCONR3	0x00D0	16	0x0000
Cache control register 4	TMRA_BCONR4	0x00D8	16	0x0000
Capture control register 1	TMRA_CCONR1	0x0100	16	0x0000
Capture control register 2	TMRA_CCONR2	0x0104	16	0x0000
Capture control register 3	TMRA_CCONR3	0x0108	16	0x0000
Capture control register 4	TMRA_CCONR4	0x010C	16	0x0000
Capture control register 5	TMRA_CCONR5	0x0110	16	0x0000
Capture control register 6	TMRA_CCONR6	0x0114	16	0x0000
Capture control register 7	TMRA_CCONR7	0x0118	16	0x0000
Capture control register 8	TMRA_CCONR8	0x011C	16	0x0000
Port Control Register 1	TMRA_PCONR1	0x0140	16	0x0000
Port Control Register 2	TMRA_PCONR2	0x0144	16	0x0000

Register Name	Sym bols	Offset	Bit width	Reset value
Port Control Register 3	TMRA_PCONR3	0x0148	16	0x0000
Port Control Register 4	TMRA_PCONR4	0x014C	16	0x0000
Port Control Register 5	TMRA_PCONR5	0x0150	16	0x0000
Port Control Register 6	TMRA_PCONR6	0x0154	16	0x0000
Port Control Register 7	TMRA_PCONR7	0x0158	16	0x0000
Port Control Register 8	TMRA_PCONR8	0x015C	16	0x0000
Hardware trigger event selection register	TMRA_HCONR	0x0084	16	0x0000
Hardware recursive event selection register	TMRA_HCUPR	0x0088	16	0x0000
Hardware decrement event selection register	TMRA_HCDOR	0x008C	16	0x0000

21.5.1 General purpose count value register (TMRA_CNTER)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CNT [15:0]															
position	Marker	Place Name	Function				Reading and writing								
b15~b0	CNT [15:0]	Counting value	Current timer count value				R/W								

21.5.2 Periodic Reference Value Register (TMRA_PERAR)

Reset value: 0xFFFF

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PER[15:0]															
position	Marker	Place Name	Function				Reading and writing								
b15~b0	PER[15:0]	Counting cycle value	Set the count period value for each round of counting				R/W								

21.5.3 Compare reference value registers (TMRA_CMPAR1~8)

Reset value: 0xFFFF

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CMP[15:0]															
position	Marker	Place Name	Function				Reading and writing								
b15~b0	CMP[15:0]	Counting comparison benchmark values	Set comparison base value				R/W								

21.5.4 Control Status Register (TMRA_BCSTR)

Reset value: 0x0002

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
UDF F	OVF F	ITEN UDF	ITEN OVF	-	-	-	-		CKDIV[3:0]		SYN ST	MODE	DIR	START	

position	Marker	Place Name	Function	Reading and writing
b15	UDFF	Underflow sign	0: When counting down, no count overflow occurs 1: Count down overflow occurs when counting down	R/W
b14	OVFF	Overflow sign	0: No count overflow when counting up 1: Count overflow occurs when counting up	R/W
b13	ITENUDF	Underflow interrupt enable	0: Count underflow interrupt is not enabled 1: Count underflow interrupt enable	R/W
b12	ITENOVF	Overflow interrupt enable	0: Count overflow interrupt is not enabled 1: Count overflow interrupt enable	R/W
b11~b8	Reserved	-	Read "0", write "0" when writing	R/W
b7~b4	CKDIV[3:0]	Counting clock selection	0000: PCLK 0001: PCLK/2 0010: PCLK/4 0011: PCLK/8 0100: PCLK/16 0101: PCLK/32 0110: PCLK/64 0111: PCLK/128 1000: PCLK/256 1001: PCLK/512 1010: PCLK/1024	R/W
b3	SYNST	Synchronous start enable	0: Synchronous start function with unit 1 is invalid 1: Synchronous start function with unit 1 is effective Note: The setting of this bit of unit 1 is invalid and read out as "0"	R/W
b2	MODE	Counting Mode	0: Sawtooth wave mode 1: Triangular wave mode	R/W
b1	DIR	Counting direction	0: Counter counts down 1: Counter counts up	R/W
b0	START	Timer start	0: Timer off 1: Timer start Note 1: This bit will automatically change to 0 when the hardware stop condition is valid Note 2: When the synchronous start function of cells 2~6 is valid, this bit of the corresponding cell in cell 1 It will also be set after the software starts	R/W

21.5.5 Interrupt Control Register (TMRA_ICONR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								IT EN8	IT EN7	IT EN6	IT EN5	IT EN4	IT EN3	IT EN2	IT EN1

position	Marker	Place Name	Function	Reading and writing
b15~b8	Reserved	-	Read "0", write "0" when writing	R/W
b7	ITEN8	Count Match Interrupt Enable 8	0: The interrupt is invalid when the CMPAR8 register is equal to the count value, or when a capture input event occurs 1: When the CMPAR8 register is equal to the count value, or when a capture input event occurs This interrupt enables	R/W
b6	ITEN7	Counting match interrupt enable 7	0: The interrupt is invalid when the CMPAR7 register is equal to the count value, or when a capture input event occurs 1: When the CMPAR7 register is equal to the count value, or when a capture input event occurs This interrupt enables	R/W
b5	ITEN6	Counting match interrupt enable 6	0: The interrupt is invalid when the CMPAR6 register is equal to the count value, or when a capture input event occurs 1: When the CMPAR6 register is equal to the count value, or when a capture input event occurs This interrupt enables	R/W
b4	ITEN5	Counting match interrupt enable 5	0: The interrupt is invalid when the CMPAR5 register is equal to the count value, or when a capture input event occurs 1: When the CMPAR5 register is equal to the count value, or when a capture input event occurs This interrupt enables	R/W
b3	ITEN4	Counting match interrupt enable 4	0: The interrupt is invalid when the CMPAR4 register is equal to the count value, or when a capture input event occurs 1: When the CMPAR4 register is equal to the count value, or when a capture input event occurs This interrupt enables	R/W
b2	ITEN3	Counting match interrupt enable 3	0: The interrupt is invalid when the CMPAR3 register is equal to the count value, or when a capture input event occurs 1: When the CMPAR3 register is equal to the count value, or when a capture input event occurs This interrupt enables	R/W
b1	ITEN2	Counting match interrupt enable 2	0: The interrupt is invalid when the CMPAR2 register is equal to the count value, or when a capture input event occurs 1: When the CMPAR2 register is equal to the count value, or when a capture input event occurs This interrupt enables	R/W
b0	ITEN1	Counting match interrupt enable 1	0: The interrupt is invalid when the CMPAR1 register is equal to the count value, or when a capture input event occurs 1: When the CMPAR1 register is equal to the count value, or when a capture input event occurs This interrupt enables	R/W

21.5.6 Event Control Register (TMRA_ECONR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								ET EN8	ET EN7	ET EN6	ET EN5	ET EN4	ET EN3	ET EN2	ET EN1
position	Marker	Place Name	Function								Reading and writing				
b15~b8	Reserved	—	Read "0", write "0" when writing								R/W				
b7	ETEN8	Counting match event enable 8	0: The event output is invalid when the CMPAR8 register is equal to the count value, or when a capture input event occurs 1: When the CMPAR8 register is equal to the count value, or when a capture input event occurs This event output enables								R/W				
b6	ETEN7	Counting match event enable 7	0: The event output is invalid when the CMPAR7 register is equal to the count value, or when a capture input event occurs 1: When the CMPAR7 register is equal to the count value, or when a capture input event occurs This event output enables								R/W				
b5	ETEN6	Counting match event enable 6	0: The event output is invalid when the CMPAR6 register is equal to the count value, or when a capture input event occurs 1: When the CMPAR6 register is equal to the count value, or when a capture input event occurs This event output enables								R/W				
b4	ETEN5	Counting match event enable 5	0: The event output is invalid when the CMPAR5 register is equal to the count value, or when a capture input event occurs 1: When the CMPAR5 register is equal to the count value, or when a capture input event occurs This event output enables								R/W				
b3	ETEN4	Counting match event enable 4	0: The event output is invalid when the CMPAR4 register is equal to the count value, or when a capture input event occurs 1: When the CMPAR4 register is equal to the count value, or when a capture input event occurs This event output enables								R/W				
b2	ETEN3	Counting match event enable 3	0: The event output is invalid when the CMPAR3 register is equal to the count value, or when a capture input event occurs 1: When the CMPAR3 register is equal to the count value, or when a capture input event occurs This event output enables								R/W				
b1	ETEN2	Counting match event enable 2	0: The event output is invalid when the CMPAR2 register is equal to the count value, or when a capture input event occurs 1: When the CMPAR2 register is equal to the count value, or when a capture input event occurs This event output enables								R/W				
b0	ETEN1	Counting match event enable 1	0: When the CMPAR1 register is equal to the count value, or when a capture input event occurs, the event output is invalid 1: When the CMPAR1 register is equal to the count value, or when a capture input event occurs This event output enables								R/W				

21.5.7 Filter Control Register (TMRA_FCONR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	NOFI CKCB[1:0]	NOFI ENCB	-	NOFI CKCA[1:0]	NOFI ENCA	-	-	-	-	-	-	NOFI CKTG [1:0]	NOFI ENTG		

position	Marker	Place Name	Function	Reading and writing
b15	Reserved	-	Read "0", write "0" when writing	R/W
b14~b13	NOFICKCB[1:0]	Filter sampling reference clock selection CB	00: PCLK 01: PCLK/4 10: PCLK/16 11: PCLK/64	R/W
b12	NOFIENCB	Capture input port filtering CB	0: TIMA_<t>_CLKB port input filtering function is invalid 1: TIMA_<t>_CLKB port input filtering function is enabled	R/W
b11	Reserved	-	Read "0", write "0" when writing	R/W
b10~b9	NOFICKCA[1:0]	Filter sampling reference clock selection CA	00: PCLK 01: PCLK/4 10: PCLK/16 11: PCLK/64	R/W
b8	NOFIENCA	Capture input port filtering CA	0: TIMA_<t>_CLKA port input filtering function is invalid 1: TIMA_<t>_CLKA port input filtering function is enabled	R/W
b7~b3	Reserved	-	Read "0", write "0" when writing	R/W
b2~b1	NOFICKTG[1:0]	Filter sampling reference clock selection TG	00: PCLK 01: PCLK/4 10: PCLK/16 11: PCLK/64	R/W
b0	NOFIENTG	Capture input port filtering TG	0: TIMA_<t>_TRIG input port filtering function is invalid 1: TIMA_<t>_TRIG input port filtering function is enabled	R/W

21.5.8 Status Flag Register (TMRA_STFLR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								CMP F8	CMP F7	CMP F6	CMP F5	CMP F4	CMP F3	CMP F2	CMP F1

position	Marker	Place Name	Function	Reading and writing
b15-b8	Reserved	-	Read "0", write "0" when writing	R
b7	CMPF8	Counting match flag 8	0: The value of CMPAR8 register is not equal to the count value, and no TIMA_<gt>_PWM8 capture completion action has occurred 1: The value of CMPAR8 register is equal to the count value, or TIMA_<gt>_PWM8 capture occurs Complete the action	R/W
b6	CMPF7	Counting match flag 7	0: The value of CMPAR7 register is not equal to the count value, and the TIMA_<gt>_PWM7 capture completion action has not occurred 1: The value of CMPAR7 register is equal to the count value, or TIMA_<gt>_PWM7 capture occurs Complete the action	R/W
b5	CMPF6	Counting match flag 6	0: The value of CMPAR6 register is not equal to the count value, and no TIMA_<gt>_PWM6 capture completion action has occurred 1: The value of CMPAR6 register is equal to the count value, or TIMA_<gt>_PWM6 capture occurs Complete the action	R/W
b4	CMPF5	Counting match flag 5	0: The value of CMPAR5 register is not equal to the count value, and no TIMA_<gt>_PWM5 capture completion action has occurred 1: The value of CMPAR5 register is equal to the count value, or TIMA_<gt>_PWM5 capture occurs Complete the action	R/W
b3	CMPF4	Counting match flag 4	0: The value of CMPAR4 register is not equal to the count value, and no TIMA_<gt>_PWM4 capture completion action has occurred 1: The value of CMPAR4 register is equal to the count value, or TIMA_<gt>_PWM4 capture occurs Complete the action	R/W
b2	CMPF3	Counting match flag 3	0: The value of CMPAR3 register is not equal to the count value, and the TIMA_<gt>_PWM3 capture completion action has not occurred 1: The value of CMPAR3 register is equal to the count value, or TIMA_<gt>_PWM3 capture occurs Complete the action	R/W
b1	CMPF2	Counting match flag 2	0: The value of CMPAR2 register is not equal to the count value, and no TIMA_<gt>_PWM2 capture completion action has occurred 1: The value of CMPAR2 register is equal to the count value, or TIMA_<gt>_PWM2 capture occurs Complete the action	R/W
b0	CMPF1	Counting match flag 1	0: The value of CMPAR1 register is not equal to the count value, and the TIMA_<gt>_PWM1 capture completion action has not occurred 1: The value of CMPAR1 register is equal to the count value, or TIMA_<gt>_PWM1 capture occurs Complete the action	R/W

21.5.9 Cache control registers (TMRA_BCONR1~4)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
Reserved														BSE1	BSE0	BEN
position	Marker	Place Name				Function				Reading and writing						
b15~b3	Reserved	-				Read "0", write "0" when writing				R/W						
b2	BSE1	Triangular wave cache transmission option 1	-	0: The cache value is not transmitted when the delta wave counting mode counts to the valley point	1: The cache value is transmitted when the delta wave counting mode counts to the valley point, i.e: CMMARm -> CMPARn (m=2, 4, 6, 8, n=1, 3, 5, 7)					R/W						
b1	BSE0	Triangle wave cache transmission option 0	-	0: The cache value is not transmitted when the triangular wave counting mode counts to the peak point	1: The cache value is transmitted when the triangular wave counting mode counts to the peak point, i.e: CMMARm -> CMPARn (m=2, 4, 6, 8, n=1, 3, 5, 7)					R/W						
b0	BEN	Cache Enable	-	0: CMPARn base value caching function is invalid	1: The caching function of CMPARn benchmark value is effective (n=1, 3, 5, 7)					R/W						

21.5.10 Capture control registers (TMRA_CCONR1~8)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	NOFI CKCP[1:0]	NOFI ENCP	-	-	HICP 4	HICP 3	-	HICP 2	HICP 1	HICP 0	-	-	-	CAP MD	

position	Marker	Place Name	Function	Reading and writing
b15	Reserved	-	Read "0", write "0" when writing	R/W
b14~b13	NOFICKCP[1:0]	Filter sampling reference clock selection CP	00: PCLK 01: PCLK/4 10: PCLK/16 11: PCLK/64	R/W
b12	NOFIENCP	Capture input port filtering CP	0: TIMA_<t>_PWMin port input filtering function is invalid 1: TIMA_<t>_PWMin port input filtering function is enabled (n=1~8)	R/W
b11~b10	Reserved	-	Read "0", write "0" when writing	R/W
b9	HICP4	Capture input condition enable 4	0: When the TIMA_<t>_TRIG port input is sampled to the falling edge, no capture input action occurs on channel m+1 1: When the TIMA_<t>_TRIG port input is sampled to the falling edge, channel m+1 generates a capture input action Note: This bit is valid only for the CCONR3 register. That is, after this bit is valid and the corresponding event occurs If CCONR4.CAPMD=1, the current counter value is captured and saved to CMPAR4, and STFLR.CMPF4 is set.	R/W
b8	HICP3	Capture input condition enable 3	0: When the TIMA_<t>_TRIG port input is sampled to the rising edge, no capture input action occurs on channel m+1 1: When the TIMA_<t>_TRIG port input is sampled to the rising edge, channel m+1 generates a capture input action Note: This bit is valid only for the CCONR3 register. That is, after this bit is valid and the corresponding event occurs If CCONR4.CAPMD=1, the current counter value is captured and saved to CMPAR4, and STFLR.CMPF4 is set.	R/W
b7	Reserved	-	Read "0", write "0" when writing	R/W
b6	HICP2	Capture input condition enable 2	0: When the event specified in the TMRA_HTSSR1 register occurs, no capture input action occurs 1: When the event specified in the TMRA_HTSSR1 register occurs, a capture input action is generated	R/W
b5	HICP1	Capture input condition enable 1	0: When the TIMA_<t>_PWMin port input is sampled to the falling edge, no capture input action occurs 1: Capture input action is generated when the TIMA_<t>_PWMin port input is sampled to the falling edge (n=1~8)	R/W
b4	HICP0	Capture input condition enable 0	0: When the TIMA_<t>_PWMin port input is sampled to the rising edge, no capture input action occurs 1: Capture input action is generated when the TIMA_<t>_PWMin port input is sampled to the rising edge (n=1~8)	R/W

b3~b1	Reserved	-	Read "0", write "0" when writing	R/W
b0	CAPMD	Function mode selection	0: Comparison output function 1: Capture input function	R/W

21.5.11 Port control registers (TMRA_PCONR1~8)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	OUTEN	-	-	FORC[1:0]	PERC[1:0]	CMPC[1:0]	STPC[1:0]	STAC[1:0]					

position	Marker	Place Name	Function	Reading and writing
b15~b13	Reserved	-	Read "0", write "0" when writing	R/W
b12	OUTEN	Output Enable	0: Invalid TIMA_<t>_PWMn port output for PWM output function 1: TIMA_<t>_PWMn port output valid for PWM output function (n=1~8)	R/W
b11~b10	Reserved	-	Read "0", write "0" when writing	R/W
b9~b8	FORC[1:0]	Forced port status setting	0x: set invalid fixed 10: Next cycle starts, TIMA_<t>_PWMn port output set to low 11: Next cycle starts, TIMA_<t>_PWMn port output set to high Note 1: Lower cycle means hardware counting mode or sawtooth wave counting to the upper or lower overflow point, triangle wave counting to the valley point Note 2: This register bit can be used to achieve 0% or 100% PWM output duty cycle control	R/W
b7~b6	PERC[1:0]	Port status setting when cycle value matches	00: When the count value is equal to PERAR, the TIMA_<t>_PWMn port output is set low 01: When the count value is equal to PERAR, the TIMA_<t>_PWMn port output is set high 10: The TIMA_<t>_PWMn port output remains in the previous state when the count value is equal to PERAR 11: When the count value is equal to PERAR, the TIMA_<t>_PWMn port output is set to the inverted level (n=1~8)	R/W
b5~b4	CMPC[1:0]	Port status setting when comparison values match	00: When the count value is equal to CMPARn, the TIMA_<t>_PWMn port output is set low 01: When the count value is equal to CMPARn, the TIMA_<t>_PWMn port output is set high 10: When the count value is equal to CMPARn, the TIMA_<t>_PWMn port output remains in the previous state 11: When the count value is equal to CMPARn, the TIMA_<t>_PWMn port output is set to the inverted level (n=1~8)	R/W
b3~b2	STPC[1:0]	Port status setting when counting is stopped	00: TIMA_<t>_PWMn port output is set low when counting is stopped 01: TIMA_<t>_PWMn port output is set to high when counting is stopped 10: When counting stops, the TIMA_<t>_PWMn port output remains in the previous state 11: When counting stops, the TIMA_<t>_PWMn port output remains in the previous state (n=1~8)	R/W

b1~b0	STAC[1:0]	Port status setting at the start of counting	00: The TIMA_<t>_PWMn port output is set low when counting starts 01: TIMA_<t>_PWMn port output is set to high when counting starts 10: The TIMA_<t>_PWMn port output remains in the previous state when counting starts 11: The TIMA_<t>_PWMn port output remains in the previous state when counting starts (n=1~8)	R/W
Note: This bit is set only in the case of no crossover (BCSTR.CKDIV=4'h0)				

Valid, other crossover frequency please set to 2'b10 or 2'b11

21.5.12 Hardware Trigger Event Selection Register (TMRA_HCONR)

Reset value: 0x0000

b15	b14	b13	b12	b1	b10	b9	b8	b	b6	b5	b4	b	b2	b1	b0
HCL E	HCL E	HCL E	HCL E	-	HCL E	HCL E	HCL E	-	HST P	HST P	HST P	-	HST A	HST A	HST A

position	Marker	Place Name	Function	Reading and writing
b15	HCLE6	Hardware clearing condition 6	Condition: TIMA_<P>_PWM3 port input sampled to falling edge 0: Hardware clear invalid when condition matched 1: Hardware clearing is valid when the conditions are matched	R/W
b14	HCLE5	Hardware clearing condition 5	Condition: TIMA_<P>_PWM3 port input sampled to rising edge 0: hardware clear invalid when condition matched 1: Hardware clearing is valid when the conditions are matched	R/W
b13	HCLE4	Hardware clearing condition 4	Condition: When this unit is unit m, the TRIG port input of unit n is sampled to the falling edge (n=2, 4, 6 when m=1, 3, 5; n=1, 3, 5 when m=2, 4, 6) 0: Hardware clearing is invalid when the condition is matched 1: Hardware clearing is valid when the conditions are matched	R/W
b12	HCLE3	Hardware clearing condition 3	Conditions: When this unit is unit m, the TRIG port input of unit n is sampled to the rising edge (n=2, 4, 6 when m=1, 3, 5; n=1, 3, 5 when m=2, 4, 6) 0: Hardware clearing is invalid when the condition is matched 1: Hardware clearing is valid when the conditions are matched	R/W
b11	Reserved	-	Read "1", write "0" when writing	R/W
b10	HCLE2	Hardware clearing condition 2	Condition: The event specified in the TMRA_HTSSR0 register occurs 0: The hardware clear is invalid when the condition is matched 1: Hardware clearing is valid when the conditions are matched	R/W
b9	HCLE1	Hardware clearing condition 1	Condition: TIMA_TRIG port input sampled to falling edge 0: hardware clear invalid when condition matched 1: Hardware clearing is valid when the conditions are matched	R/W
b8	HCLE0	Hardware clear condition 0	Condition: TIMA_TRIG port input sampled to rising edge 0: hardware clear invalid when condition is matched 1: Hardware clearing is valid when the conditions are matched	R/W
b7	Reserved	-	Read "1", write "0" when writing	R/W
b6	HSTP2	Hardware stop condition 2	Condition: The event specified in the TMRA_HTSSR0 register occurs 0: Hardware stop is invalid when the condition is matched 1: Hardware stop valid when conditions match	R/W
b5	HSTP1	Hardware stop condition 1	Condition: TIMA_<P>_TRIG port input sampled to falling edge 0: hardware stop invalid when condition is matched 1: Hardware stop valid when conditions match	R/W

b4	HSTP0	Hardware stop condition 0	Condition: TIMA_<t>_TRIG port input sampled to rising edge 0: Hardware stop invalid when condition is matched 1: Hardware stop valid when conditions match	R/W
b3	Reserved	-	Read "1", write "0" when writing	R/W
b2	HSTA2	Hardware start-up condition 2	Condition: The event specified in the TMRA_HTSSR0 register occurs 0: Hardware boot is invalid when the condition is matched	R/W

1: Hardware start is valid when the conditions are matched			
b1	HSTA1	Hardware start-up conditions1	Condition: 1) This unit TIMA_<>>_TRIG port input is sampled to the falling edge (synchronous start function is disabled) 2) TIMA_1_TRIG port input sampled to falling edge (synchronous start function active) 0: Hardware boot is invalid when the condition is matched 1: Hardware start is valid when the conditions are matched Note: Condition 2) Only cells 2~6 can be selected, cell 1 is not valid
b0	HSTA0	Hardware start condition 0	Condition: 1) This unit TIMA_<>>_TRIG port input is sampled to the rising edge (synchronous start function is disabled) 2) TIMA_1_TRIG port input sampled to rising edge (synchronous start function active) 0: Hardware boot is invalid when the condition is matched 1: Hardware start is valid when the conditions are matched Note: Condition 2) Only cells 2~6 can be selected, cell 1 is not valid

21.5.13 Hardware recursive event selection register (TMRA_HCUPR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	HC UP12	HC UP11	HC UP10	HC UP9	HC UP8	HC UP7	HC UP6	HC UP5	HC UP4	HC UP3	HC UP2	HC UP1	HC UP0

position	Marker	Place Name	Function	Reading and writing
b15~b13	Reserved	-	Read '0', write "0" when writing	R/W
b12	HCUP12	Hardware incremental condition 12	Conditions: this unit is unit m when the count overflow occurs in unit n (when m = 1, 3, 5, n = 2, 4, 6; when m = 2, 4, 6, n = 1, 3, 5) 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W
b11	HCUP11	Hardware incremental condition 11	Conditions: this unit is unit m when the count overflow occurs in unit n (when m = 1, 3, 5, n = 2, 4, 6; when m = 2, 4, 6, n = 1, 3, 5) 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W
b10	HCUP10	Hardware incremental condition 10	Condition: The event specified in the TMRA_HTSSR0 register occurs 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W
b9	HCUP9	Hardware incremental condition 9	Condition: TIMA_<t>_TRIG port up-sampled to falling edge 0: Hardware recursion invalid when condition matches 1: Hardware recursion is valid when the conditions are matched	R/W
b8	HCUP8	Hardware incremental condition 8	Condition: TIMA_<t>_TRIG port up-sampled to rising edge 0: Hardware recursion invalid when condition matches 1: Hardware recursion is valid when the conditions are matched	R/W
b7	HCUP7	Hardware incremental condition 7	Condition: TIMA_<t>_CLKB port is high when TIMA_<t>_CLKA port is upsampled to falling edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W
b6	HCUP6	Hardware incremental condition 6	Condition: TIMA_<t>_CLKB port is high when TIMA_<t>_CLKA port is sampled to rising edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W
b5	HCUP5	Hardware incremental condition 5	Condition: TIMA_<t>_CLKB port is low when TIMA_<t>_CLKA port is upsampled to falling edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W
b4	HCUP4	Hardware incremental condition 4	Condition: TIMA_<t>_CLKB port is low when TIMA_<t>_CLKA port is sampled to rising edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W
b3	HCUP3	Hardware incremental condition 3	Condition: TIMA_<t>_CLKA port is high when TIMA_<t>_CLKB port is upsampled to falling edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched	R/W

b2	HCUP2	Hardware incremental port on condition 2	Condition: When TIMA_<t>_CLKA port is high, the TIMA_<t>_CLKB R/W
----	-------	---	---

Sampling to rising edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched			
b1	HCUP1	Hardware incremental condition 1	Condition: TIMA_<t>_CLKA port is low when TIMA_<t>_CLKB port is upsampled to falling edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched
b0	HCUP0	Hardware recurrence condition 0	Condition: TIMA_<t>_CLKA port is low when TIMA_<t>_CLKB port is upsampled to a rising edge 0: Hardware recursion is invalid when the condition is matched 1: Hardware recursion is valid when the conditions are matched

21.5.14 Hardware decrement event selection register (TMRA_HCDOR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	HC DO12	HC DO11	HC DO10	HC DO9	HC DO8	HC DO7	HC DO6	HC DO5	HC DO4	HC DO3	HC DO2	HC DO1	HC DO0
position	Marker	Place Name	Function												Reading and writing
b15~b13	Reserved	-	Read "0", write "0" when writing												R/W
b12	HCDO12	Hardware decreasing condition 12	Conditions: this unit is unit m when the count overflow occurs in unit n (when m = 1, 3, 5, n = 2, 4, 6; when m = 2, 4, 6, n = 1, 3, 5) 0: Hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched												R/W
b11	HCDO11	Hardware decreasing condition 11	Conditions: this unit is unit m when the count overflow occurs in unit n (when m = 1, 3, 5, n = 2, 4, 6; when m = 2, 4, 6, n = 1, 3, 5) 0: Hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched												R/W
b10	HCDO10	Hardware decreasing condition 10	Condition: The event specified in the TMRA_HTSSR0 register occurs 0: Hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched												R/W
b9	HCDO9	Hardware decreasing condition 9	Condition: TIMA_<t>_TRIG port up-sampled to falling edge 0: hardware decrement invalid when condition matches 1: Hardware decrement is valid when the conditions are matched												R/W
b8	HCDO8	Hardware decreasing condition 8	Condition: TIMA_<t>_TRIG port up-sampled to rising edge 0: hardware decrement invalid when condition matched 1: Hardware decrement is valid when the conditions are matched												R/W
b7	HCDO7	Hardware decreasing condition 7	Condition: TIMA_<t>_CLKB port is high when TIMA_<t>_CLKA port is upsampled to falling edge 0: Hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched												R/W
b6	HCDO6	Hardware decreasing condition 6	Condition: TIMA_<t>_CLKB port is high when TIMA_<t>_CLKA port is sampled to rising edge 0: Hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched												R/W
b5	HCDO5	Hardware decreasing condition 5	Condition: TIMA_<t>_CLKB port is low when TIMA_<t>_CLKA port is upsampled to falling edge 0: Hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched												R/W
b4	HCDO4	Hardware decreasing condition 4	Condition: TIMA_<t>_CLKB port is low when TIMA_<t>_CLKA port is sampled to rising edge 0: Hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched												R/W

b3	HCDO3	Hardware decreasing condition 3	Condition: TIMA_<t>_CLKA port is high when TIMA_<t>_CLKB port is upsampled to falling edge 0: Hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched	R/W
b2	HCDO2	Hardware decreasing condition 2	Condition: TIMA_<t>_CLKA port is high when TIMA_<t>_CLKB port is upsampled to rising edge	R/W

			0: Hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched	
b1	HCDO1	Hardware decreasing condition 1	Condition: TIMA_<t>_CLKA port is low when TIMA_<t>_CLKB port is upsampled to falling edge 0: Hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched	R/W
b0	HCDO0	Hardware decrement condition 0	Condition: TIMA_<t>_CLKA port is low when TIMA_<t>_CLKB port is upsampled to a rising edge 0: Hardware decrement is invalid when the condition is matched 1: Hardware decrement is valid when the conditions are matched	R/W

22 General-purpose timer (Timer0)

22.1 Introduction

The general-purpose Timer0 is a basic timer that can be used for both synchronous counting and asynchronous counting. This timer contains 2 channels (CH-A and CH-B) and can generate a compare match event during counting. This event can trigger an interrupt or be used as an event output to control other modules, etc. Timer0 with 2 units is equipped in this series.

22.2 Basic Block Diagram

The basic block diagram of Timer0 is shown in Figure 22-1.

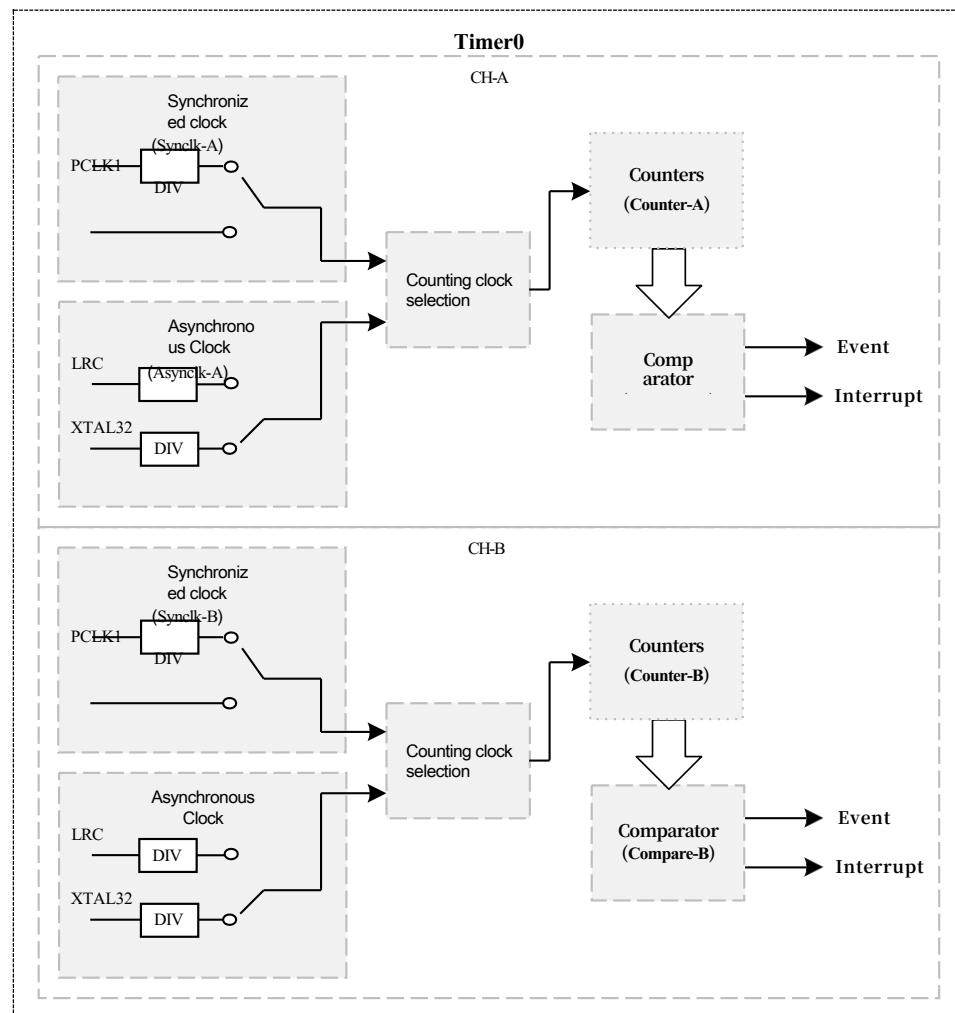


Figure 22-1 Timer0 Basic Block Diagram



22.3 Function Description

22.3.1 Clock source selection

Timer0 can be counted either synchronously or asynchronously.

The synchronous counting method means that the timer's counting clock and the bus access clock (register read/write operation clock) have a synchronous timing relationship; the asynchronous counting method means that the timer's counting clock and the bus access clock (register read/write operation clock) have a non-synchronous timing relationship. When the register read operation is performed in the asynchronous counting method, the state of the timer, etc. may be changing and unpredictable state is read out. Therefore, in the asynchronous counting method, the register read operation must be implemented in the count stop state.

22.3.1.1 Synchronous counting clock source

In the synchronous counting mode (BCONR.SYNSA=0), the clock source can be selected as follows (BCONR.SYNCLKA set)

- a) PCLK1 and 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 divisions of PCLK1 as synchronous count clock (BCONR.SYNCLKA=0 & BCONR.CKDIVA [3:0] set)
- b) Internal hardware triggered event input as synchronous count clock (BCONR.SYNCLKA=1)

22.3.1.2 Asynchronous counting clock source

In the asynchronous counting mode (BCONR.SYNSA=1), the clock source can be selected as follows (BCONR.ASYNCLK A setting)

- a) LRC clock source input and its 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 divisions as asynchronous count clock (BCONR.ASYNCLK A=0 & BCONR.CKDIVA [3:0] set)
- b) XTAL32 clock source input and its 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 divisions as asynchronous count clock (BCONR.ASYNCLK A=1 & BCONR.CKDIVA [3:0] set)

22.3.2 Basic counting action

Each channel of Timer0 can set a reference count value and generate a count comparison match event when the count value and the reference value are equal. This is shown in Figure 22-2.

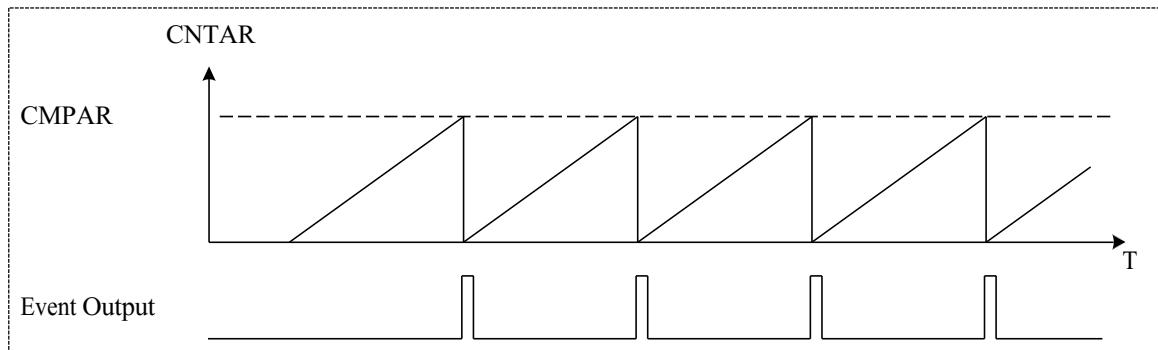


Figure 22-2 Timer0 Counting Timing Diagram

22.3.3 Hardware triggered actions

triggered

actions

The 2 channels of Timer0 have a common internal hardware trigger source, which can be used to control the timer state (count, start, stop, clear) and capture input actions, etc. through the relevant settings in the Basic Control Register (BCONR).

The source selection of this hardware trigger source is achieved by entering the corresponding number into the Hardware Trigger Select Register (HTSSR). Please refer to the Interrupt Controller (INTC) chapter for the specific event correspondence. To use the internal hardware trigger function, the peripheral circuit trigger function bit (AOS) of the Function Clock Control 0 register (PWC_FCG0) needs to be enabled first.

22.4 Interrupts and event descriptions

22.4.1 Interrupt output

A Timer0 contains 2 interrupts, a count compare match interrupt or an input capture interrupt for channel A and channel B, respectively.

The base value registers (CMPAR, CMPBR), of which there are two, can be compared with the count value registers (CNTAR, CNTBR) to generate a valid signal for comparison matching. If the BCONR.INTENA bit of the basic control register (BCONR) is set to enable the interrupt, the corresponding interrupt request (TMR0_Um_GCMn, m=1, 2; n=A, B) will be triggered. = A, B) will also be triggered.

When the internal hardware trigger input is used as the capture input condition, the corresponding capture input action can be generated. In this case, if the basic control register is set (BCONR) of the BCONR.INTENA bit enables the interrupt, the corresponding interrupt request (TMR0_Um_GCMn, m=1, 2; n=A, B) is triggered.

Caution:

- - The compare match interrupt (TMR0_U1_GCMA) for channel A of unit 1 is only available in asynchronous counting mode (BCONR.SYNSA=1).

22.4.2 Event Output

A Timer0 contains 2 event outputs, a count comparison match event for channel A and channel B, or a capture input event.

When a count comparison match or capture input action occurs during counting, the corresponding event request (TMR0_Um_GCMn, m=1, 2; n=A, B) output signal is generated respectively, which can be used to selectively trigger other modules.

22.5 Register Description

Table 22-1 shows the register list of the Timer0 module.

BASE ADDR: 0x4002_4000 (U1) 0x4002_4400 (U2)

Table 22-1 Register List

Register Name	Sym bols	Offset	Bit width	Reset value
Count value register	TMR0_CNTAR	0x0000	32	0x0000_0000
Count value register	TMR0_CNTBR	0x0004	32	0x0000_0000
Base value register	TMR0_CMPAR	0x0008	32	0x0000_FFFF
Base value register	TMR0_CMPBR	0x000C	32	0x0000_FFFF
Basic control register	TMR0_BCONR	0x0010	32	0x0000_0000
Status Flag Register	TMR0_STFLR	0x0014	32	0x0000_0000

22.5.1 Count value register (TMR0_CNTAR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CNTA< B>[15:0]															
position	Marker	Place Name	Function	Reading and writing											
b31~b16	Reserved	-	Reads out as "0"	R											
b15~b0	CNTA[15:0]	Counting value	Current timer count value	R/W											

22.5.2 Base value register (TMR0_CMPAR)

Reset value: 0x0000_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
(PA[15:0]															
position	Marker	Place Name	Function	Reading and writing											
b31~b16	Reserved	-	Reads out as "0"	R											
b15~b0	CMPA[15:0]	Base value	Set the count reference value and generate Compare Match event	R/W											

22.5.3 Basic Control Register (TMR0_BCONR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
HICP B	HCLE B	HSTP B	HSTA B	-	ASYN CLKB	SYN CLKB	SYN SB		CKDIV B[3:0]		-	INT ENB	CAP MDB	CST B	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
HICP A	HCLE A	HSTP A	HSTA A	-	ASYN CLKA	SYN CLKA	SYN SA		CKDIV A[3:0]		-	INT ENA	CAP MDA	CST A	

position	Marker	Place Name	Function	Reading and writing
b31	HICPB	Hardware Trigger Input Capture B	Condition: internal hardware trigger event is valid 0: capture input is invalid when condition is matched 1: Capture input is valid when the conditions match	R/W
b30	HCLEB	Hardware Trigger Clear B	Condition: Internal hardware trigger event is valid 0: Timer clear is invalid when condition is matched 1: Timer clearing is valid when the condition is matched	R/W
b29	HSTPB	Hardware Trigger Stop B	Condition: Internal hardware trigger event is valid 0: Timer stop is invalid when condition is matched 1: Timer stop is valid when the condition is matched	R/W
b28	HSTAB	Hardware triggered start B	Condition: Internal hardware trigger event is valid 0: Timer start is invalid when condition is matched 1: Timer start is valid when the conditions are matched	R/W
b27	Reserved	-	Read "1", write "0" when writing	R/W
b26	ASYNCLKB	Channel B asynchronous counting clock source selection	0: LRC 1: XTAL32 Note: When the timeout function of the corresponding UART channel is enabled, the clock source is no longer XTAL32, but the clock tick generated by the UART baud rate generator	R/W
b25	SYNCLKB	Channel B Synchronous Counting Clock Source Selection	0: PCLK1 1: Internal hardware triggered events	R/W
b24	SYNSB	Channel B counting method selection	0: Synchronous counting method 1: Asynchronous counting method	R/W

b23~b20	CKDIVB[3:0]	Channel B count clock division selection	Channel B counting clock division selection: 0000: clock source 0001: Clock <small>src2</small> 0010: Clock <small>src4</small> 0011: Clock <small>src8</small> 0100: Clock source / 16 0101: Clock Source/32 0110: Clock Source/64 0111: Clock Source/128 1000: Clock source/256 1001: Clock Source/512 1010: Clock source/1024 Please do not set other values Note: The clock source to be divided can be various clock sources for asynchronous counting, synchronous counting PCLK1 at the time of	R/W
---------	-------------	--	--	-----

b19	Reserved	-	Read "0", write "0" when writing	R/W
b18	INTENB	Count Match Interrupt Enable B	0: The interrupt is invalid when the CMPBR register is equal to the count value (CNTBR), or when a capture input event occurs 1: When the CMPBR register is equal to the count value (CNTBR), or when a capture input occurs The interrupt is enabled when the event	R/W
b17	CAPMDB	Function mode selection B	0: Comparison output function 1: Capture input function	R/W
b16	CSTB	Timer start	0: Channel B timer off 1: Channel B timer start Note: This bit will automatically change to 0 when the hardware triggered stop condition is valid	R/W
b15	HICPA	Hardware Trigger Input Capture A	Condition: internal hardware trigger event is valid 0: capture input is invalid when condition is matched 1: Capture input is valid when the conditions match	R/W
b14	HCLEA	Hardware trigger clear A	Condition: Internal hardware trigger event is valid 0: Timer clear is invalid when condition is matched 1: Timer clearing is valid when the condition is matched	R/W
b13	HSTPA	Hardware Trigger Stop A	Condition: Internal hardware trigger event is valid 0: Timer stop is invalid when condition is matched 1: Timer stop is valid when the condition is matched	R/W
b12	HSTAA	Hardware triggered start A	Condition: Internal hardware trigger event is valid 0: Timer start is invalid when condition is matched 1: Timer start is valid when the conditions are matched	R/W
b11	Reserved	-	Read "0", write "0" when writing	R/W
b10	ASYNCLKA	Channel A asynchronous counting clock source selection	0: LRC 1: XTAL32	R/W
b9	SYNCLKA	Channel A synchronous counting clock source selection	0: PCLK1 1: Internal hardware triggered events	R/W
b8	SYNSA	Channel A counting method selection	0: Synchronous counting method 1: Asynchronous counting method	R/W

b7~b4	CKDIVA [3:0]	Channel A count clock division selection	Channel A count clock division selection: 0000: clock source 0001: Clock _{source2} 0010: Clock _{source4} 0011: Clock _{source8} 0100: Clock source / 16 0101: Clock Source/32 0110: Clock Source/64 0111: Clock Source/128 1000: Clock source/256 1001: Clock Source/512 1010: Clock source/1024 Please do not set other values Note: The clock source to be divided can be various clock sources for asynchronous counting, synchronous counting PCLK1 at the time of	R/W
b3	Reserved	-	Read "1", write "0" when writing	R/W
b2	INTENA	Count Match Interrupt Enable A	0: The interrupt is invalid when the CMPAR register is equal to the count value (CNTBR), or when a capture input event occurs 1: When the CMPAR register is equal to the count value (CNTBR), or when a capture input occurs	R/W

The interrupt is enabled when the event				
b1	CAPMDA	Function mode selection A	0: Comparison output function 1: Capture input function	R/W
b0	CSTA	Timer start	0: Channel A timer off 1: Channel A timer start Note: This bit will automatically change to 0 when the hardware triggered stop condition is valid	R/W

Caution:

- The internal hardware trigger events (bit31~bit28 and bit15~bit12) and XTAL32 clock sources (bit26 and bit10) mentioned in this register are provided as input by the USART module when the TIMEOUT function of the USART module is active, please refer to the introduction of USART chapter for details.

22.5.4 Status Flag Register (TMR0_STFLR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
Reserved															CMBF	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
Reserved															CMAF	

position	Marker	Place Name	Function	Reading and writing
b31~b17	Reserved	-	Read "0", write "0" when writing	R
b16	CMBF	Counting Match B	0: The value of CMPBR register is not equal to the count value and no capture input action has occurred 1: The value of CMPBR register is equal to the count value or capture input action occurs	R/W
b15~b1	Reserved	-	Read "0", write "0" when writing	R
b0	CMAF	Counting Match A	0: The value of the CMPAR register is not equal to the count value and no capture input action has occurred 1: The value of CMPAR register is equal to the count value or capture input action occurs	R/W

22.6 Precautions for use

- 1) ASYNCLKA bit to select the asynchronous clock source, and then set the BCONR_SYNSA bit to select the asynchronous counting method before starting Timer0.
- 2) If asynchronous counting is selected, the count value (CNTAR), the reference value (CMPAR) the start bit (BCONR_CSTA) status bit (STFLR.CMAF) for write action, Timer0 writes the modified value into the corresponding register only after 3 asynchronous count clocks from the time the write action is received.

23 Real Time Clock (RTC)

23.1 Introduction

The Real Time Clock (RTC) is a counter that stores time information in BCD code format. It records the specific calendar time from year 00 to year 99. It supports both 12/24 hour time systems and automatically calculates the days 28, 29 (leap year) 30 and 31 based on the month and year. The basic features are shown in Table 23-1.

Table 23-1 Basic Specifications of the RTC

Counting clock source	External low-speed oscillator (32.768KHz) RTC internal low-speed oscillator (32.768KHz)
Basic Functions	— BCD code for seconds, minutes, hours, days, weeks, months, years time
	— Software start or stop
	— 12/24 hour system selectable, leap year automatic recognition
	— — Programmable Alarm Clock
	— Distributed/uniformly compensated 1Hz clock output
	— — Clock error compensation function
Interruptions	Cycle interruption
	Alarm clock interruption

23.2 Basic Block Diagram

The basic block diagram of the RTC is shown in Figure 23-1.

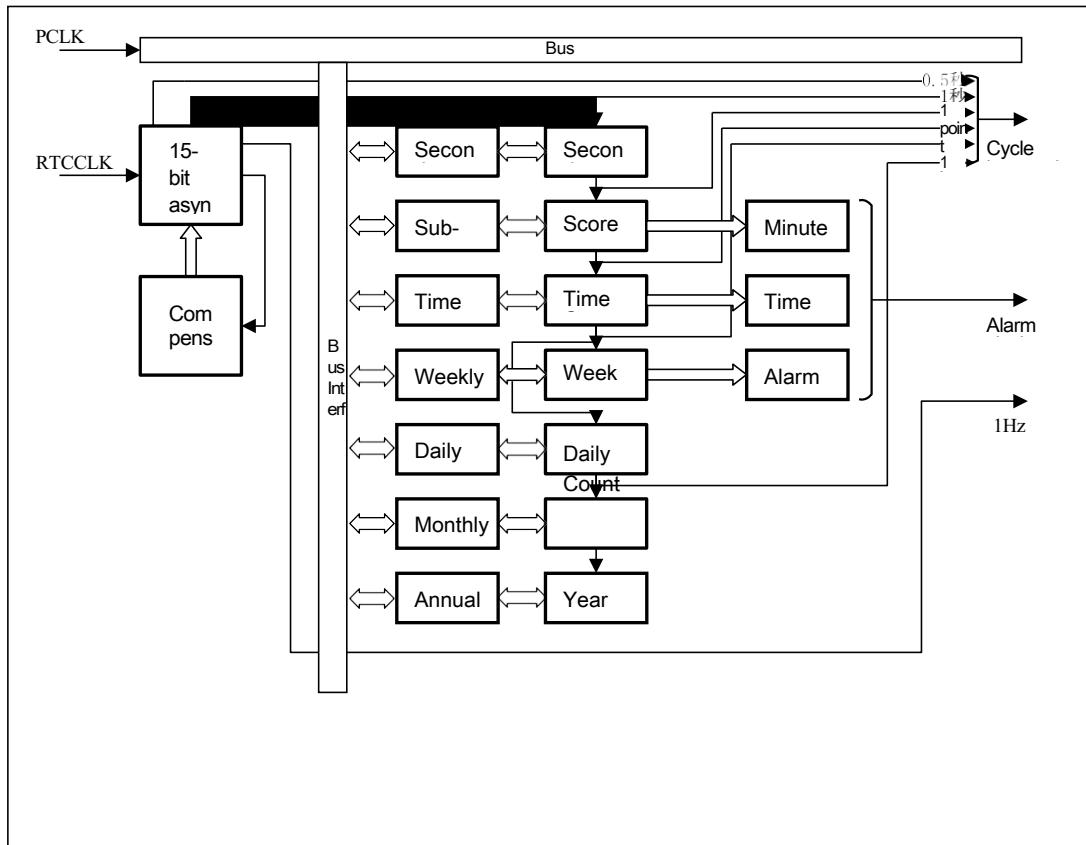


Figure 23-1 Basic block diagram of RTC

23.3 Function Description

23.3.1 Power-up settings

The RTC register can be reset by power-on reset or by setting RTC_CR0.RESET to start the RTC after setting the control register, calendar initial value, alarm setting, etc. After the RTC is started, all other external reset requests cannot reset the RTC and the RTC will remain in operation. The RTC can be stopped by setting the START bit of the control register to "0", and the clock source must be stable when the RTC is set.

23.3.2 RTC Counting Start Setting

- 1) Reset all registers except RTC_CR0 after power-on; you can also set RTC_CR0.RESET=0, confirm RESET bit is "0", then set RTC_CR0.RESET=1 to reset all registers;
- 2) Set RTC_CR1.START=0 to stop counting;
- 3) Set the system clock register, turn on the sub oscillator, then set RTC_CR3 and select the RTC count clock source;
- 4) Set RTC_CR1 to set the time system, period, and 1 Hz clock output;
- 5) Setting the calendar count register for seconds, minutes, hours, days, weeks, months and years;
- 6) When clock error compensation is required, set the count clock error compensation registers RTC_ERRCRL, RTC_ERRCRH;
- 7) Clear the flag register bits in register RTC_CR1 and enable the interrupt;
- 8) Set RTC_CR1.START=1, counting starts.

23.3.3 System low-power mode switching

If the system switches to low power mode immediately after the RTC count starts, perform one of the following checks before switching modes.

- 1) After RTC_CR1.START=1 is set, mode switching is performed after more than 2 RTC count clocks.
- 2) After RTC_CR1.START=1 is set, set RTC_CR2.RWREQ=1 and query RTC_CR2.RWEN=1. Set the calendar count register, then set RTC_CR2.RWREQ=0 and query RTC_CR2.RWEN=0 for mode switching.

23.3.4 Read out the count register

- 1) RTC_CR2.RWREQ=1 for calendar register read request after more than 2 RTC count clocks have been set after RTC_CR1.START=1 has been set;
- 2) Query until RTC_CR2.RWEN=1;

-
- 3) Read all or part of the seconds, minutes, hours, days, weeks, months and years count register values;
 - 4) Set RTC_CR2.RWREQ=0;
 - 5) Query until RTC_CR2.RWEN=0.

23.3.5 Write to count register

- 1) RTC_CR2.RWREQ=1 for calendar register write request after RTC_CR1.START=1 has been set **and after** more than 2 RTC count clocks;
- 2) Query until RTC_CR2.RWEN=1;
- 3) Write all or part of the seconds, minutes, hours, days, weeks, months and years count register values;
- 4) Set RTC_CR2.RWREQ=0. Note that all write operations must be completed within 1 second;
- 5) Query until RTC_CR2.RWEN=0.

23.3.6 Alarm settings

- 1) Set RTC_CR2.ALME=0, alarm disable;
- 2) Set RTC_CR2.ALMIE=1, alarm interrupt permitted;
- 3) minute alarm RTC_ALMMIN, hour alarm RTC_ALMHOUR, week alarm RTC_ALMWEEK Setting;
- 4) Set RTC_CR2.ALME=1, alarm permit;
- 5) Waiting for an interruption to occur;
- 6) When RTC_CR2.ALMF=1, the alarm clock interrupt processing is entered.

23.3.7 Clock error compensation

Since there are errors in the operation of the external sub oscillator crystal under various temperature conditions, it is necessary to compensate for the errors when high precision counting results are required. For the compensation method, refer to 23.5.15 Clock error compensation registers (RTC_ERRCRH, RTC_ERRCRL)

23.3.8 1Hz output

The RTC can output a 1Hz clock with three types of accuracy output, the first, normal accuracy 1Hz output without clock error compensation, the second, distributed compensation 1Hz output with average compensation every 32 seconds and the third, uniform compensation 1Hz output with compensation every second. When the clock error compensation function is valid RTC_ERRCRH.COMPEN=1, the distributed compensated 1Hz output and the uniform compensated 1Hz output can be selected. Of these, the

The 1 Hz output setting for normal accuracy is as follows:

- 1) Set RTC_CR0.RESET=0, confirm RESET bit is "1", then set RTC_CR0.RESET=1 to reset Calendar count register;
- 2) Set RTC_CR1.START=0, counting stops;
- 3) 1Hz output pin setting;
- 4) RTC_CR1.ONEHZOE=1, clock output permission;

-
- 5) Set RTC_CR1.START=1, counting starts;
 - 6) Waiting for more than 2 count cycles;
 - 7) 1Hz output start.

Distributed compensation 1Hz output is set as follows:

- 1) Set RTC_CR0.RESET=0, confirm RESET bit is "1", then set RTC_CR0.RESET=1 to reset Calendar count register;
- 2) Set RTC_CR1.START=0 to stop counting;
- 3) 1Hz output pin setting;
- 4) RTC_CR1.ONEHZOE=1, clock output permission;
- 5) Clock error compensation register RTC_ERRCRL.COMP[7:0] and RTC_ERRCRH.COMP[8] Compensation number Settings;
- 6) clock error compensation register RTC_ERRCRH.COMPEN=1, error compensation is valid;
- 7) Set RTC_CR1.START=1, counting starts;
- 8) Waiting for more than 2 count cycles;
- 9) 1Hz output start.

Uniform compensation 1Hz output is set as follows:

- 1) Set RTC_CR0.RESET=0, confirm RESET bit is "1", then set RTC_CR0.RESET=1 to reset Calendar count register;
- 2) Set RTC_CR1.START=0 to stop counting;
- 3) RTC output pin setting;
- 4) RTC_CR1.ONEHZOE=1, clock output permission;
- 5) RTC_CR1.ONEHZSEL=1 to select the output uniformly compensated 1Hz clock;
- 6) Clock error compensation register RTC_ERRCRL.COMP[7:0] and RTC_ERRCRH.COMP[8] Compensation number Settings;
- 7) clock error compensation register RTC_ERRCRH.COMPEN=1, accuracy compensation is valid;
- 8) Set RTC_CR1.START=1, counting starts;
- 9) Waiting for more than 2 count cycles;
- 10) 1Hz output start.

23.4 Interrupt description

The RTC supports 2 types of interrupts. Timed alarm interrupt and fixed cycle interrupt.

23.4.1 Alarm clock interruption

The alarm interrupt RTC_ALM is triggered when ALMIE=1 in control register 2 (RTC_CR2) and ALME=1 in control register 2 (RTC_CR2), if the current calendar time is equal to the minute alarm register (RTC_ALMMIN), hour alarm register (RTC_ALMHOUR), week alarm register (RTC_ALMWEEK). The alarm interrupt is triggered when the time is equal to the minute alarm register (RTC_ALMMIN), the hour alarm register (RTC_ALMHOUR), and the week alarm register (RTC_ALMWEEK). ALMF. Write "1" to RTC_CR1.ALMFCLR bit to clear the alarm flag. INT_SELx (x=0~31, 44~49) can only be configured to the corresponding NVIC vector.

23.4.2 Fixed-cycle interruptions

The fixed-cycle interrupt RTC_PRD, control register 2 (RTC_CR2) with PRDIE=1, triggers the fixed-cycle wake-up interrupt when the selected cycle occurs. There is no independent flag bit configured for the fixed-cycle interrupt, you can select RTC_PRD interrupt to NVIC vector by INTC.INT_SELx(x=0~31, 44~49)and query the NVIC corresponding flag bit.

23.5 Register Description

Table 23-2 shows the register list of the RTC

module. Register base address:

0x4004_C000

Table 23-2 Register List

Register Name	Symbols	Offset	Bit width	Reset value
Control register 0	RTC_CR0	0x0000	8	Indeterminate
Control register 1	RTC_CR1	0x0004	8	0x00
Control register 2	RTC_CR2	0x0008	8	0x00
Control register 3	RTC_CR3	0x000C	8	0x00
Second counter register	RTC_SEC	0x0010	8	0x00
Sub-count register	RTC_MIN	0x0014	8	0x00
Time counter register	RTC_HOUR	0x0018	8	0x12
Week Count Register	RTC_WEEK	0x001C	8	0x00
Day Count Register	RTC_DAY	0x0020	8	0x00
Monthly Count Register	RTC_MON	0x0024	8	0x00
Year Count Register	RTC_YEAR	0x0028	8	0x00
Minute Alarm Register	RTC_ALMMIN	0x002C	8	0x00
Time alarm clock register	RTC_ALMHOUR	0x0030	8	0x12
Weekly alarm clock register	RTC_ALMWEEK	0x0034	8	0x00
Clock error compensation register	RTC_ERRCRH	0x0038	8	0x00
Clock error compensation register	RTC_ERRCRL	0x003C	8	0x20

23.5.1 Control register 0 (RTC_CR0)

Reset value: Variable

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															

position	Marker	Place Name	Function	Reading and writing
b31~b1	Reserved	-	Read "0", write "0" when writing	R/W
b0	RESET	RTC calendar counter reset	Write status 0: Initialization register is invalid 1: Initialization register is valid Initialize all RTC registers. Read out status 0: Normal count state or end of RTC software reset 1: RTC is in reset state Note: Please make sure the bit is "0" before writing "1", otherwise it will not be initialized.	R/W

23.5.2 Control register 1 (RTC_CR1)

Reset value: 0x00

b3	b3	b2	b2	b2	b2	b2	b23	b22	b21	b20	b1	b18	b17	b16	
Reserved															
b1	b1	b1	b1	b1	b1	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved							STAR	ONEHZSE	ONEHZO	ALMFCL	AMP	PRDS[2]	PRDS[1]	PRDS[0]	

position	Marker	Place Name	Function	Reading and writing																																								
b31~b8	Reserved	-	Read '0', write "0" when writing	R/W																																								
b7	START	RTC counting starts	0: RTC count stops 1: RTC counting work	R/W																																								
b6	ONEHZSEL	1Hz output selection	0: Distributed compensation 1Hz output 1: Uniformly compensated 1Hz output Note: This bit setting is valid when RTC_ERRCRH.COMPEN=1.	R/W																																								
b5	ONEHZOE	1Hz output enable	0: 1Hz output disable 1: 1Hz output license	R/W																																								
b4	ALMFCLR	ALMF flag cleared	0: Clear RTC_CR2.ALMF flag register 1: Invalid	R/W																																								
b3	AMPM	Time system selection	0: 12-hour time system 1: 24-hour time system	R/W																																								
b2~0	PRDS[2:0]	Cycle break selection	Cycle selection setting: <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>PRDS[2]</td><td>PRDS[1]</td><td>PRDS[0]</td><td>Periodic selection</td></tr> <tr> <td>0</td><td>0</td><td>0</td><td>No selection</td></tr> <tr> <td>0</td><td>0</td><td>1</td><td>Every 0.5 second cycle</td></tr> <tr> <td>0</td><td>1</td><td>0</td><td>per 1 second cycle</td></tr> <tr> <td>0</td><td>1</td><td>1</td><td>per 1 minute cycle</td></tr> <tr> <td>1</td><td>0</td><td>0</td><td>per 1 time period</td></tr> <tr> <td>1</td><td>0</td><td>1</td><td>per 1 day cycle (00: 00 hours)</td></tr> <tr> <td>daily)</td><td></td><td></td><td>(00 seconds)</td></tr> <tr> <td>1</td><td>1</td><td>X</td><td>per 1 month cycle (0000 hours on the 1st of each month)</td></tr> <tr> <td></td><td></td><td></td><td>Minutes 00 seconds</td></tr> </table> Note: When writing cycle selection during START=1 counting, please turn off the cycle interrupt permit to prevent misoperation. And the relevant flag bit should be cleared after writing.	PRDS[2]	PRDS[1]	PRDS[0]	Periodic selection	0	0	0	No selection	0	0	1	Every 0.5 second cycle	0	1	0	per 1 second cycle	0	1	1	per 1 minute cycle	1	0	0	per 1 time period	1	0	1	per 1 day cycle (00: 00 hours)	daily)			(00 seconds)	1	1	X	per 1 month cycle (0000 hours on the 1st of each month)				Minutes 00 seconds	R/W
PRDS[2]	PRDS[1]	PRDS[0]	Periodic selection																																									
0	0	0	No selection																																									
0	0	1	Every 0.5 second cycle																																									
0	1	0	per 1 second cycle																																									
0	1	1	per 1 minute cycle																																									
1	0	0	per 1 time period																																									
1	0	1	per 1 day cycle (00: 00 hours)																																									
daily)			(00 seconds)																																									
1	1	X	per 1 month cycle (0000 hours on the 1st of each month)																																									
			Minutes 00 seconds																																									

23.5.3 Control register 2 (RTC_CR2)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								ALME	ALMIE	PRDIE	-	ALMF	-	RWEN	RWREQ

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Read "0", write "0" when writing	R/W
b7	ALME	Alarm Clock Enabling	0: Alarm clock function is disabled 1: Alarm clock function license	R/W
b6	ALMIE	Alarm clock interrupt enable	0: Alarm clock interruption disable 1: Alarm clock interruption permit	R/W
b5	PRDIE	Periodic interrupt enable	0: Cycle interrupt disable 1: Cycle interruption license	R/W
b4	Reserved	-	Read "0", write "0" when writing	R/W
b3	ALMF	Alarm clock logo	0: Alarm clock mismatch 1: Alarm clock matching	R
b2	Reserved	-	Read with variable, write with "0" when writing	R/W
b1	RWEN	Read/write allowed	0: read/write disable 1: Read/write allowed Note: Calendar register read/write allowed flag. Please check if this bit is "1" before reading/writing. The calendar register includes seconds, minutes, hours, week, day, month, and year count registers.	R/W
b0	RWREQ	Read/Write Requests	0: Normal counting mode 1: Read/Write requests Note: When reading/writing the calendar register, please set this bit to "1" and request reading/writing. Since the counter is counting continuously, please complete the reading/writing operation and clear this bit to "0" within 1 second.	R/W

23.5.4 Control register 3 (RTC_CR3)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								RCKSEL	-	-	LCREN	-	-	-	-

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Read "1", write "0" when writing	R/W
b7	RCKSEL	RTC count clock selection	0: Select sub oscillator 1: Select LOCO	R/W
b6	Reserved	-	Read "1", write "0" when writing	R/W
b5	Reserved	-	Read "1", write "0" when writing	R/W
b4	LCREN	Low-speed oscillator enable	0: Low speed oscillator stop 1: Low-speed oscillator work Note: The operation and stop of the low-speed oscillator is determined by the clock control circuit and any of the LRCEN settings. Please set the LRCEN bit to enable when the low speed oscillator is used as the RTC clock source.	R/W
b3~b1	Reserved	-	Read "1", write "0" when writing	R/W
b0	Reserved	-	Read "1", write "0" when writing	R/W

23.5.5 Second Count Register (RTC_SEC)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								-	SECD[2:0]	SECU[3:0]					

position	Marker	Place Name	Function	Reading and writing
b31~b7	Reserved	-	Read "1", write "0" when writing	R/W
b6~b4	SECD[2:0]	second tenth place	Seconds decimeter value	R/W
b3~b0	SECU[3:0]	second single digit	Second digit count value	R/W

Indicates 0-59 seconds, using decimal counting. Please write the BCD code of decimal 0-59, and the written value will be ignored when writing the wrong value.

23.5.6 Sub-count register (RTC_MIN)

Reset value: Variable

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								-	MIND[2:0]			MINU[3:0]			

position	Marker	Place Name	Function	Reading and writing
b31~b7	Reserved	-	Read '1', write "0" when writing	R/W
b6~b4	MIND[2:0]	Decimal	Decimal count value	R/W
b3~b0	MINU[3:0]	Sub-digit	Sub-digit count value	R/W

Indicates 0-59 points, using decimal counting. Please write the BCD code of decimal 0-59, and the written value will be ignored when you write the wrong value.

23.5.7 Time Count Register (RTC_HOUR)

Reset value: 0x12

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								-	-	HOURD[1:0]	HOURU[3:0]				

position	Marker	Place Name	Function	Reading and writing
b31~b6	Reserved	-	Read "1", write "0" when writing	R/W
b5~b4	HOURD[1:0]	Time 10	Time decimal meter value	R/W
b3~b0	HOURU[3:0]	Time Digit	Hourly digit count value	R/W

For 24-hour time system, it means 0~23 hours; for 12-hour time system, b5=0 means AM, then 01~12 means AM; b5=1 means PM, then 21~32 means PM.

Please set the BCD code of 0~23 or 01~12,21~32 in correct decimal according to the value of control bit AMPM. Writing values out of range will be ignored.

Refer to the following table for specific time indications:

24-hour time system	AMPM=1	12-hour hourly system	AMPM=0
Time	Register representation	Time	Register representation
00 hours	00H	12:00 AM	12H
01 hour	01H	AM 01 hour	01H
02 hours	02H	AM 02 hours	02H
03 hours	03H	AM 03 hours	03H
04 hours	04H	AM 04 hours	04H
05 hours	05H	AM 05 hours	05H
06 hours	06H	AM 06 hours	06H
07 hours	07H	AM 07 hours	07H
08 hours	08H	AM 0800 hours	08H
09 hours	09H	AM 0900 hours	09H
10 o'clock	10H	AM 10:00 am	10H
11 pm	11H	11:00 AM	11H
12 o'clock	12H	PM 12:00 pm	32H
13:00	13H	PM 01 hour	21H
14h	14H	PM 02 hours	22H
15 hours	15H	PM 03 hours	23H
16h00	16H	PM 04 hours	24H
17h00	17H	PM 05 hours	25H
18h00	18H	PM 06 hours	26H
19h	19H	PM 07 hours	27H
20 hours	20H	PM 0800 hours	28H
21h00	21H	PM 0900 hours	29H
22 hours	22H	PM 10 when	30H
2300 hours	23H	PM 11:00 pm	31H

23.5.8 Day Count Register (RTC_DAY)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								-	-	DAYD[1:0]	DAYU[3:0]				

position	Marker	Place Name	Function	Reading and writing
b31~b6	Reserved	-	Read "1", write "0" when writing	R/W
b5~b4	DAYD[1:0]	Day 10	Daily decimeter value	R/W
b3~b0	DAYU[3:0]	Daily Digits	Daily single digit meter value	R/W

The decimal representation of 1~31 days and the automatic calculation of leap years and months. The specific representation is as follows:

Month	Day count representation
February (ordinary year)	01~28
February (leap year)	01~29
April, June, September, November	01~30
January, March, May, July, August, October, December	01~31

23.5.9 Week Count Register (RTC_WEEK)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								-	-	-	-	-	-	WEEK[2:0]	

position	Marker	Place Name	Function	Reading and writing
b31~b3	Reserved	-	Read "1", write "0" when writing	R/W
b2~b0	WEEK[2:0]	Week	Weekly count value	R/W

Decimal 0~6 means Sunday~Saturday. Please write the correct BCD code of decimal 0~6, write other values, they will be ignored. The weekly count values correspond to the following:

Week	The weekly count indicates
Sunday	00H
Monday	01H
Tuesday	02H
Wednesday	03H
Thursday	04H
Friday	05H
Saturday	06H

23.5.10 Monthly Count Register (RTC_MON)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								-	-	-	MON[4:0]				

position	Marker	Place Name	Function	Reading and writing
b31~b5	Reserved	-	Read "1", write "0" when writing	R/W
b4~b0	MON[4:0]	Month	Monthly Value	R/W

Decimal 1~12 means 1~12 months. Please write the correct decimal 1~12 BCD code, write other value, it will be ignored.

23.5.11 Year Count Register (RTC_YEAR)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								YEARD[3:0]				YEARU[3:0]			

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Read "1", write "0" when writing	R/W
b7~b4	YEARD[3:0]	Year 10	Annual decimal values	R/W
b3~b0	YEARU[3:0]	Year Single Digit	Annual single digit values	R/W

Decimal 0~99 means 0~99 years. Counting according to month progression. Automatic calculation of leap years such as 00, 04, 08, ..., 92, 96, etc. Please write the correct decimal year count value, writing wrong value will be ignored.

23.5.12 Sub-alarm register (RTC_ALMMIN)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				-	ALMMIND[2:0]				ALMMINU[3:0]						

position	Marker	Place Name	Function	Reading and writing
b31~b7	Reserved	-	Read "0", write "0" when writing	R/W
b6~b4	ALMMIND[2:0]	Minute alarm clock ten matching values		R/W
b3~b0	ALMMINU[3:0]	Minute alarm clock single digit matching value		R/W

Please set the BCD code of decimal 0~59. Write other values, no alarm match will occur.

23.5.13 Time Alarm Clock Register (RTC_ALMHOUR)

Reset value: 0x12

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				-	-	ALMHOURD[1:0]				ALMHOURU [3:0]					

position	Marker	Place Name	Function	Reading and writing
b31~b6	Reserved	-	Read "0", write "0" when writing	R/W
b5~b4	ALMHOURD[1:0]	Time alarm clock decimal	Hourly alarm clock 10-bit matching value	R/W
b3~b0	ALMHOURU [3:0]	Time alarm clock digits	Time alarm clock digit matching value	R/W

Please set the correct alarm matching value according to the time system, otherwise the time alarm matching will not occur.

23.5.14 Weekly alarm clock register (RTC_ALMWEEK)

Reset value: 0x00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								-	ALMWEEK[6:0]						

position	Marker	Place Name	Function	Reading and writing
b31~b7	Reserved	-	Read "1", write "0" when writing	R/W
b6~b0	ALMWEEK[6:0] 1	Alarm Clock of the Week	Weekly alarm clock matching values. b0~b6 corresponds to Sunday~Saturday respectively, when the corresponding setting is "1", it means the alarm clock is valid on that day of the week. For example, b0=1, b5=1 means the alarm is valid on Sunday and Friday.	R/W

Please set the correct alarm matching value according to the time system, otherwise the time alarm matching will not occur.

23.5.15 Clock error compensation registers (RTC_ERRCRH, RTC_ERRCRL)

Reset value: 0x0000_0020

RTC_ERRCRH

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								COMPEN	-	-	-	-	-	-	COMP [8]

RTC_ERRCRL

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								COMP[7:0]							

RTC_ERRCRH

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Read "0", write "0" when writing	R/W
b7	COMPEN	Compensation Enable	0: Clock error compensation is invalid 1: Clock error compensation is effective	R/W
B6~b1	Reserved	-	Read "0", write "0" when writing	R/W
b0	COMP [8]	Compensation value	Set the compensation value together with COMP[7:0]	R/W

RTC_ERRCRL

position	Marker	Place Name	Function	Reading and writing																																																																																																																																																																																													
b31~b8	Reserved	-	Read "0", write "0" when writing	R/W																																																																																																																																																																																													
b7~b0	COMP[7:0]	Compensation value	The compensation value setting enables the accuracy compensation of +/-0.96ppm per second. The compensation value is 9 digits with decimal point 2 The last 5 digits are the decimal part of the complement. Compensable range -275.5ppm~+212.9ppm. minimum resolution 0.96ppm. please refer to the following table for details of compensation accuracy:	R/W																																																																																																																																																																																													
			<table border="1"> <thead> <tr> <th colspan="10">Compensation value setting</th> <th rowspan="2">Number of compensation</th> </tr> <tr> <th>COMPEN</th> <th colspan="9">COMP[8:0]</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> <td>-275.5ppm</td> </tr> <tr> <td></td> <td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td></td> <td>-274.6ppm</td> </tr> <tr> <td></td> <td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td> <td>~</td> </tr> <tr> <td></td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> <td>-30.5ppm</td> </tr> <tr> <td></td> <td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td> <td>~</td> </tr> <tr> <td></td> <td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td> <td>-0.96ppm</td> </tr> <tr> <td></td> <td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> <td>0ppm</td> </tr> <tr> <td></td> <td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td> <td>+0.96ppm</td> </tr> <tr> <td></td> <td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td> <td>~</td> </tr> <tr> <td></td> <td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> <td>+30.5ppm</td> </tr> <tr> <td></td> <td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td><td>~</td> <td>~</td> </tr> <tr> <td></td> <td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td> <td>+212.0ppm</td> </tr> <tr> <td></td> <td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td> <td>+212.9ppm</td> </tr> <tr> <td>0</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td> <td>Invalid</td> </tr> </tbody> </table>	Compensation value setting										Number of compensation	COMPEN	COMP[8:0]									1	1	0	0	0	0	0	0	0	0	0	-275.5ppm		1	0	0	0	0	0	0	0	1		-274.6ppm		~	~	~	~	~	~	~	~	~	~	~		0	0	0	0	0	0	0	0	0	0	-30.5ppm		~	~	~	~	~	~	~	~	~	~	~		0	0	0	0	1	1	1	1	1	1	-0.96ppm		0	0	0	1	0	0	0	0	0	0	0ppm		0	0	0	1	0	0	0	0	0	1	+0.96ppm		~	~	~	~	~	~	~	~	~	~	~		0	0	1	0	0	0	0	0	0	0	+30.5ppm		~	~	~	~	~	~	~	~	~	~	~		0	1	1	1	1	1	1	1	1	0	+212.0ppm		0	1	1	1	1	1	1	1	1	1	+212.9ppm	0	X	X	X	X	X	X	X	X	X	X	Invalid	
Compensation value setting										Number of compensation																																																																																																																																																																																							
COMPEN	COMP[8:0]																																																																																																																																																																																																
1	1	0	0	0	0	0	0	0	0	0	-275.5ppm																																																																																																																																																																																						
	1	0	0	0	0	0	0	0	1		-274.6ppm																																																																																																																																																																																						
	~	~	~	~	~	~	~	~	~	~	~																																																																																																																																																																																						
	0	0	0	0	0	0	0	0	0	0	-30.5ppm																																																																																																																																																																																						
	~	~	~	~	~	~	~	~	~	~	~																																																																																																																																																																																						
	0	0	0	0	1	1	1	1	1	1	-0.96ppm																																																																																																																																																																																						
	0	0	0	1	0	0	0	0	0	0	0ppm																																																																																																																																																																																						
	0	0	0	1	0	0	0	0	0	1	+0.96ppm																																																																																																																																																																																						
	~	~	~	~	~	~	~	~	~	~	~																																																																																																																																																																																						
	0	0	1	0	0	0	0	0	0	0	+30.5ppm																																																																																																																																																																																						
	~	~	~	~	~	~	~	~	~	~	~																																																																																																																																																																																						
	0	1	1	1	1	1	1	1	1	0	+212.0ppm																																																																																																																																																																																						
	0	1	1	1	1	1	1	1	1	1	+212.9ppm																																																																																																																																																																																						
0	X	X	X	X	X	X	X	X	X	X	Invalid																																																																																																																																																																																						

Description of compensation calculation:

When the 1Hz clock is output directly in the default state, the compensation target value is calculated by measuring the accuracy of this clock. Assuming that the actual measured value is 0.9999888Hz, the

$$\text{Actual oscillation frequency} = 32768 \times 0.9999888 \approx 32767.63$$

$$\text{Compensation target value} = (\text{actual oscillation frequency} - \text{target frequency}) / \text{target frequency} \times 10^6$$

$$= (32767.96 - 32768) / 32768 \times 10^6 \\ = -11.29 \text{ppm}$$

setpoint calculated:

$$\text{COMP}[8:0] = \left(\frac{\text{Compensation target value [ppm]} \times 2^{15}}{10^6} \right) + 0001.00000B$$

Take the 2's complement

If the compensation target value is +20.3 ppm, calculate the corresponding register value as follows:

$$\text{COMP[8:0]} = (20.3 \times 215/106) \text{ taken as 2's complement} + 0001.00000\text{B}$$

$$\begin{aligned} &= (0.6651904) \quad \text{Take 2's complement + 0001.00000B} \\ &= 0000.10101B + 0001.00000B \\ &= 0001.10101B \end{aligned}$$

If the compensation target value is -20.3 ppm, calculate the corresponding register value as follows:

$$\begin{aligned} \text{COMP[8:0]} &= (-20.3 \times 215/106) \text{ taken as 2's complement + 0001.00000B} \\ &= (-0.6651904) \text{ taken as 2's complement + 0001.00000B} \\ &= 1111.01011B + 0001.00000B \\ &= 0000.01011B \end{aligned}$$

24 Watchdog counter (WDT/SWDT)

24.1 Introduction

There are two watchdog counters, one is a dedicated watchdog counter where the count clock source is a dedicated internal RC (SWDTRC:10KHz)

(SWDT) and the other is a general-purpose watchdog counter (WDT) with count clock source of PCLK3.

The dedicated watchdog and general-purpose watchdog are 16-bit decrementing counters used to monitor software faults resulting from deviations from normal operation of the application due to external disturbances or unforeseen logic conditions.

Both watchdogs support the window function. The window interval can be preset before the count starts, and when the count value is in the window interval, the counter can be refreshed and the count starts again. The basic features are shown in Table 24-1.

Table 24-1 Basic Characteristics of the Watchdog Counter

Counting Clock	SWDT: 1/16/32/64/128/256/2048 divisions of SWDTRC WDT: 4/64/128/256/512/1024/2048/8192 divisions of PCLK3
Maximum overflow time	SWDT: 3.72hour(max) WDT: 10.7s (PCLK3=50MHz)
Counting Mode	Decreasing count
Window Functions	Window interval can be set to define the allowed interval for refresh action
Start-up method	1) Hardware start-up 2) Software startup (SWDT does not support software startup)
Stop conditions	1) Resetting in progress 2) Underflow, refresh error occurs when Start again: In hardware start mode, automatically start after reset or interrupt request output In software start mode, set refresh register again
Interrupt/reset conditions	1) Counting down overflow 2) Refresh Error

24.2 Function Description

24.2.1 Start the watchdog

The watchdog counter can be started in two ways: hardware start-up and software start-up. (SWDT only supports hardware start-up)

The hardware start-up method means that the watchdog counter setting information (ICG0 register) is read from the main flash memory area at start-up, and the counter starts counting automatically; the software start-up method means that after setting the control register, the counter starts counting after writing the flush register to complete the flush action.

24.2.2 Hardware boot method

When bit 16 (WDTAUTS) and bit 0 (SWDTAUTS) of ICG0 register are 0, it is hardware boot method. When hardware start mode is selected, the setting information of WDT_CR register is invalid.

In the hardware start-up method, the WDT/SWDT-related settings (count clock, window setting value, count period, etc.) in the ICG0 register are loaded into the WDT/SWDT module during reset, and the counter automatically starts counting according to the settings after reset. Figure 24-1 shows an example of the hardware start-up method.

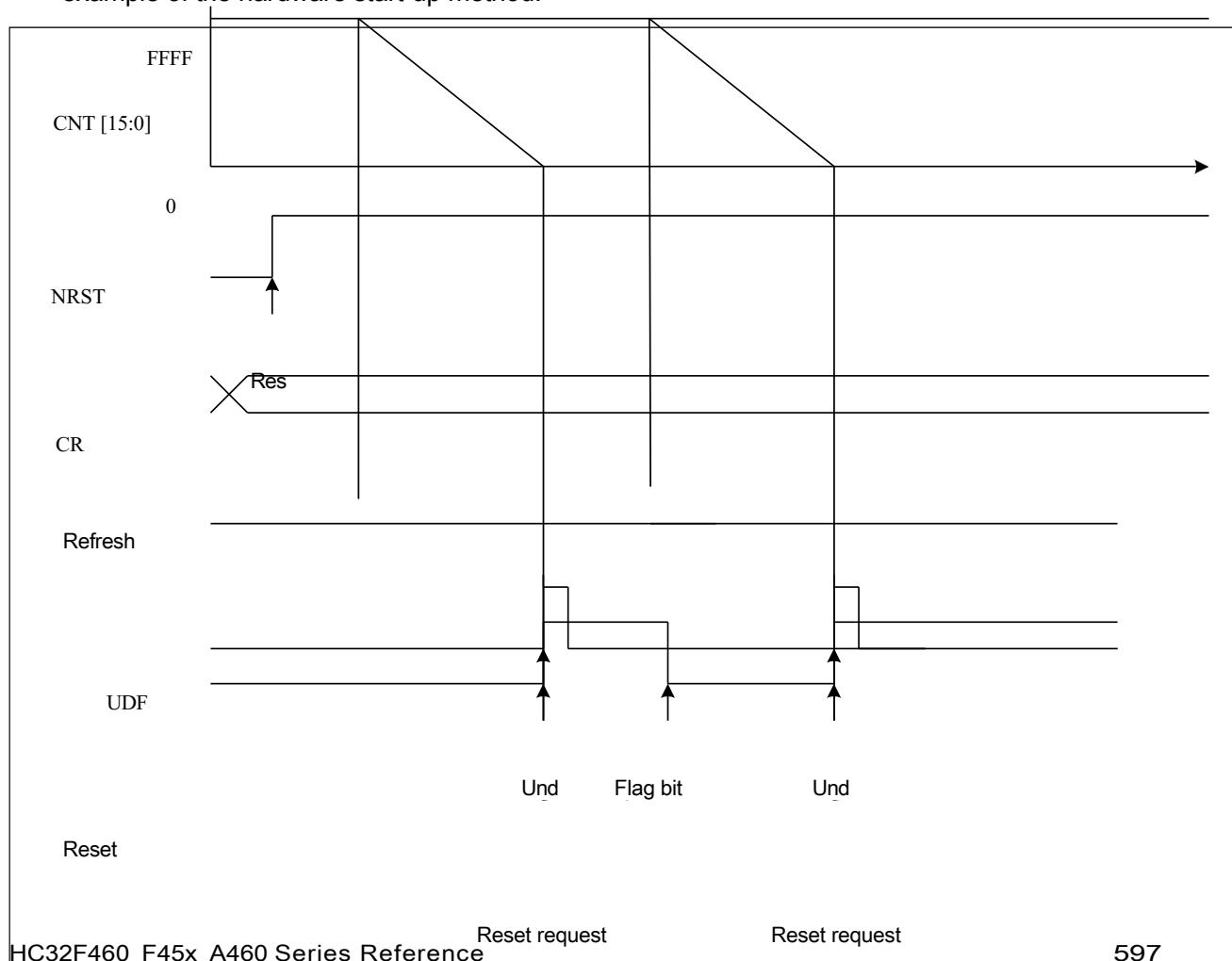


Figure 24-1 Hardware Startup Example

24.2.3 Software startup method

When bit 16 (WDTAUTS) of ICG0 register is 1, the WDT is started by setting the refresh register as the software start method. After resetting, set the count clock, window setting value, count period, etc. in the WDT_CR register, and then execute the refresh action, and the counter will start counting. Figure 24-2 shows the example of the software start-up method.

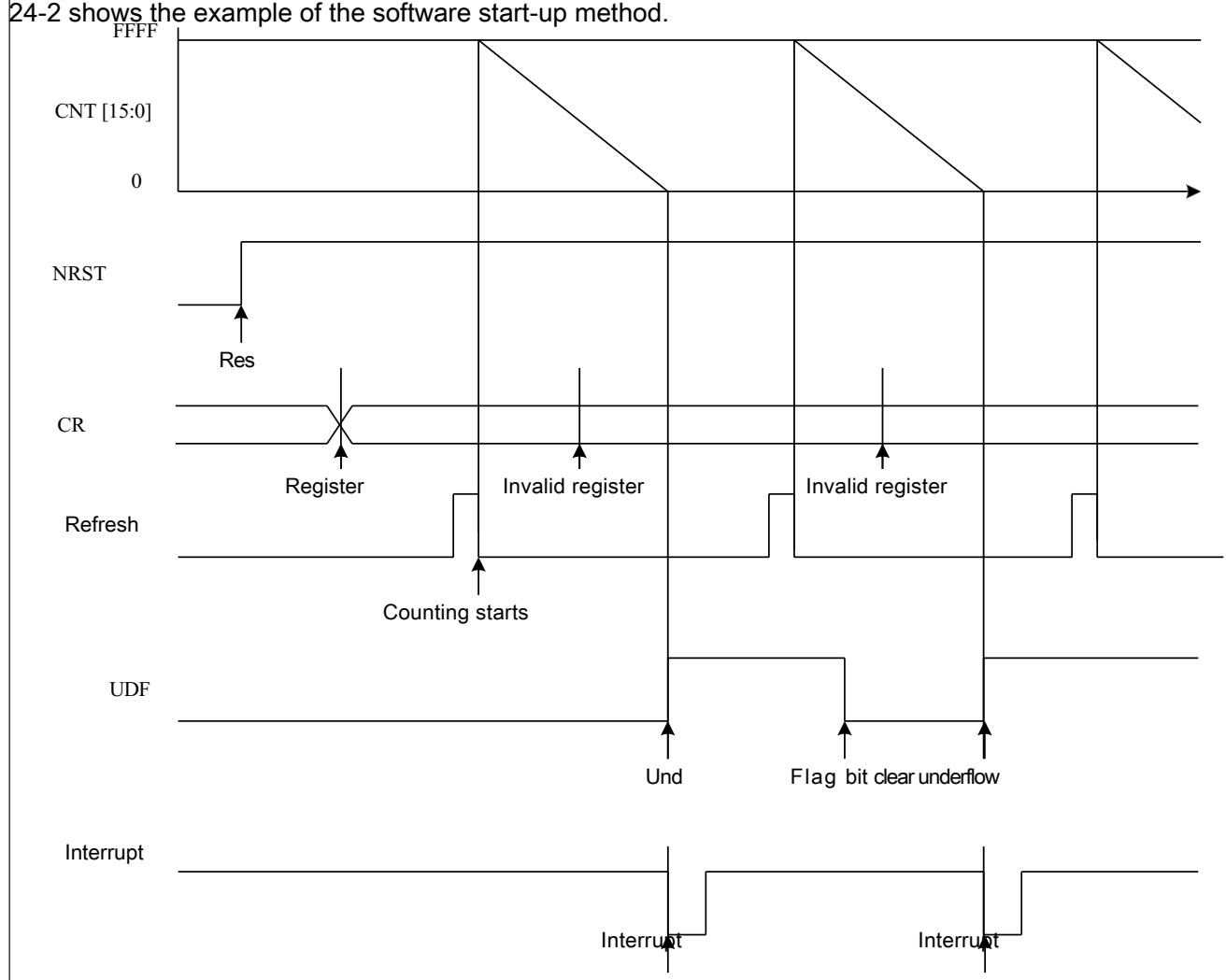


Figure 24-2 Software startup example

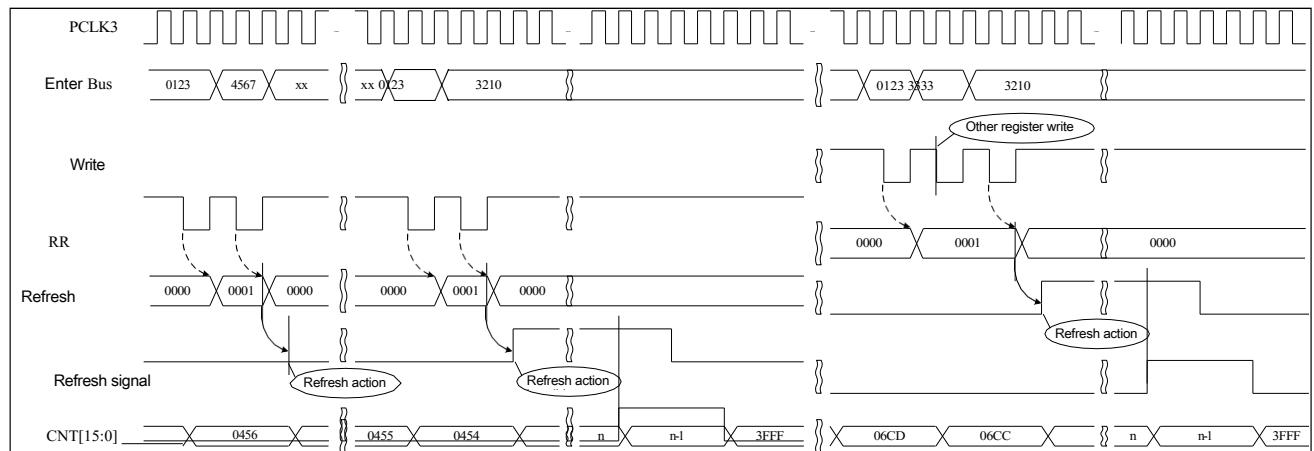
24.2.4 Refresh action

(S) Write 0x0123 and then 0x3210 in the WDT_RR register to complete a refresh action, and the counter of WDT/SWDT will start counting (software start) or restart counting.

(S) WDT_RR register between write 0x0123, 0x3210, if to occur to other register access or read

(S) WDT_RR register, etc., does not affect the normal refresh action.

An example of the action is shown in Figure 24-3.



**Figure 24-3 Example of various refresh action timings
(action confirmation, falling edge of refresh request signal, etc.)**

The refresh action requires 4 count cycles to update the count value, so please write the refresh register 4 count values before refreshing the lower window and lower overflow position. Please read the status register to confirm the count value.

24.2.5 Logo position

The refresh error flag bit and the count underflow flag bit are held in the case of both reset requests and interrupt requests. When a reset is released or an interrupt is entered, the cause of the reset or interrupt can be confirmed by querying the flag bits.

Flag bit clear: read "1" and then write "0".

The hardware start mode watchdog count does not stop when the refresh error or count underflow flag bit is set. For SWDT, when writing "0" to the flag bit, it takes up to 3 SWDTLRC and 2 PCLK3 times for the register bit to be cleared. For WDT, when writing "**1**" to the flag bit, it takes up to 5 PCLK3 times for the register bit to be cleared. On the other hand, a read "**1**" to the flag bit is not valid for a certain period of time after a refresh error or underflow error, which is 1 count cycle + 2 SWDTLRC (SWDT module); 1 count cycle + 2 PCLK3 (WDT module).

24.2.6 Interrupt reset

SWDT has the option of generating an interrupt request or a reset request in case of counter count

overflow or refresh error. The SWDTITS bit of ICG0 is set to determine whether to generate an interrupt request or a reset request.

The WDT can choose to generate an interrupt request or a reset request in case of counter count overflow or refresh error. The decision to generate an interrupt request or a reset request is made by setting the WDTITS bit of ICG0 during hardware startup and the WDT_CR register ITS bit during software startup.

There are two conditions for interrupt reset of WDT/SWDT. One is when the counter count generates an overflow; the other is when a refresh action is performed outside the refresh allowed interval, generating a refresh error.

24.2.7 Counting down overflow

As in the example in Figure 24-4, the underflow is generated when the decrement count reaches zero.

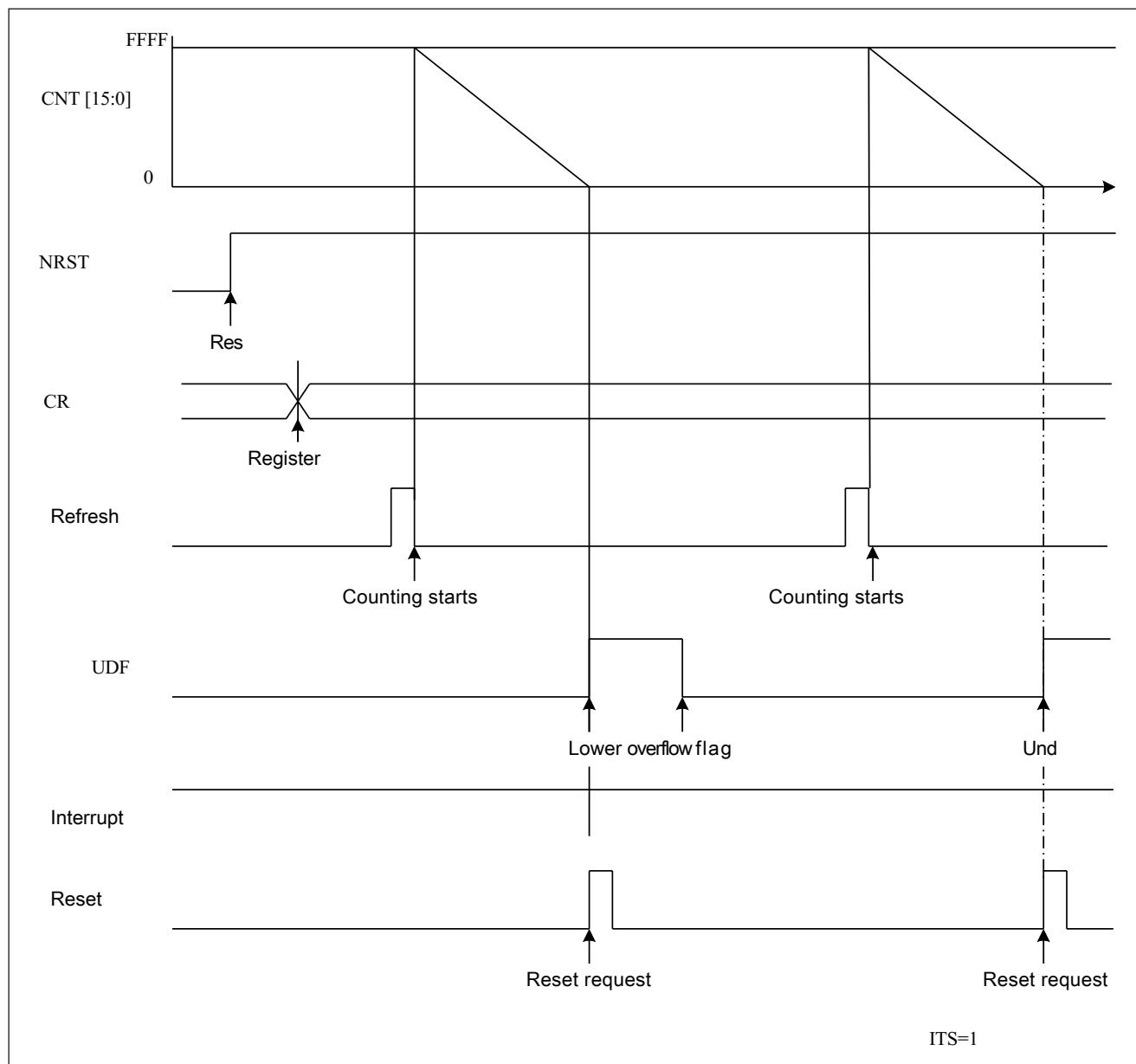


Figure 24-4 Example of counter underflow action

24.2.8 Refresh Error

After setting the window interval, the counter will be refreshed and restarted only when the refresh action is performed inside the window interval, and a refresh error will occur when the refresh action is performed outside the window interval. Figure 24-5 shows an example of a refresh action.

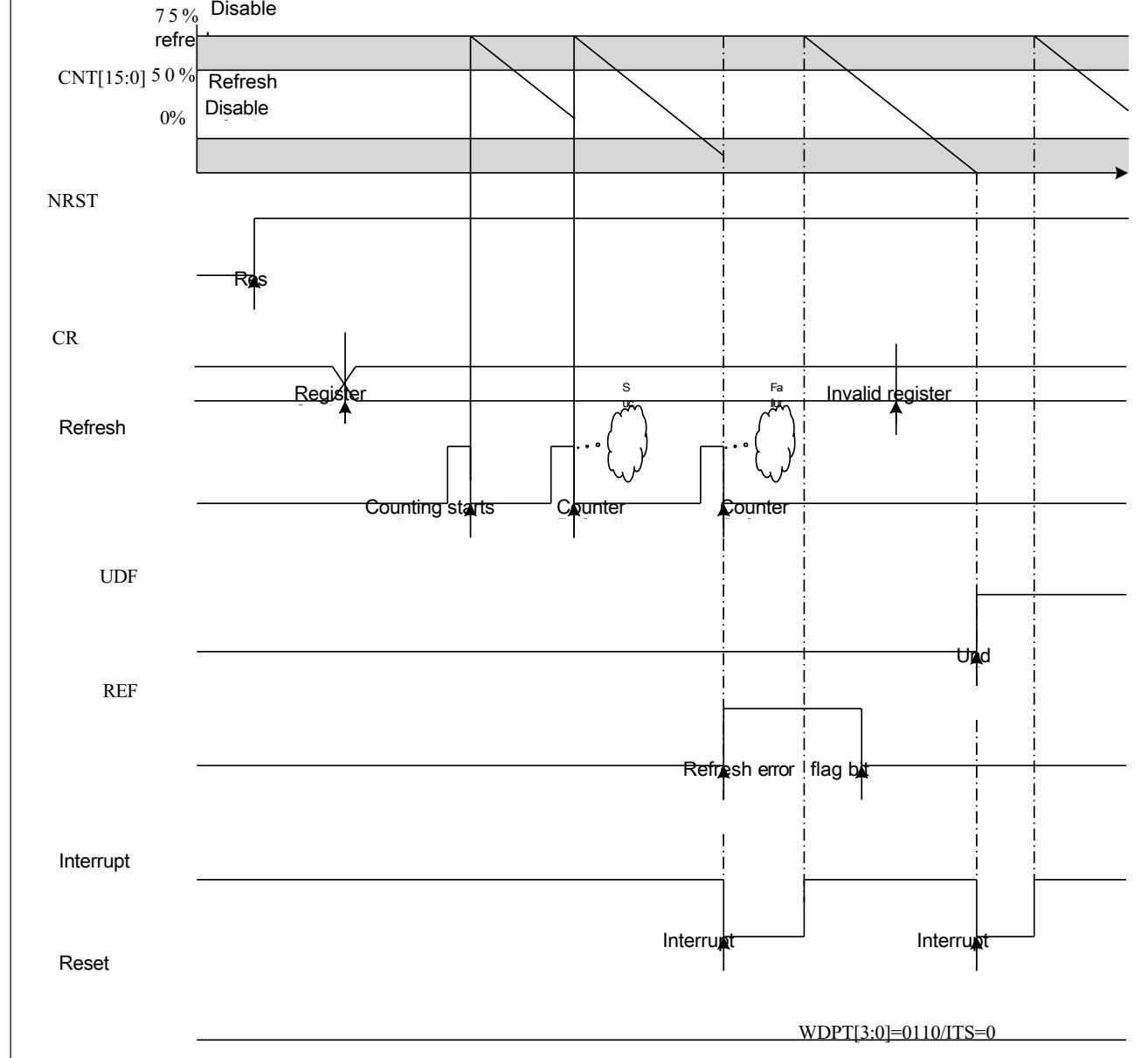


Figure 24-5 Example of Counter Refresh Action

24.3 Register Description

Table 24-2 shows the register list of SWDT and WDT

modules. WDT_BASE_ADDR: 0x4004_9000

SWDT_BASE_ADDR:0x4004_9400

Table 24-2 Register List

Register Name	Sym bols	Offset Address	Bit width	Reset value
SWDT status register	SWDT_SR	0x04	32	0x0000_0000
SWDT refresh register	SWDT_RR	0x08	32	0x0000_0000
WDT control register	WDT_CR	0x00	32	0x8001_0FF3
WDT Status Register	WDT_SR	0x04	32	0x0000_0000
WDT Refresh Register	WDT_RR	0x08	32	0x0000_0000

24.3.1 Control register (WDT_CR)

Reset value: 0x8001_0FF3

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
ITS	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SLPOFF
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	WDPT[3:0]		CKS[3:0]		-	-	PERI[1:0]					

position	Marker	Place Name	Function	Reading and writing
b31	ITS	Refresh error/overflow interrupt/reset selection	0: Interrupt request 1: Reset request	R/W
b30~b17	Reserved	-	Read "0", write "0" when writing	R
b16	SLPOFF	WDT count disable in Sleep mode	0: WDT counting permission in sleep mode 1: WDT in sleep mode count disable	R/W
b15~b13	Reserved	-	Read "0", write "0" when writing	R
b11~b8	WDPT[3:0]	Refresh Allowed Area Count Value Percentage	0000 : 0%~100% 0001 : 0%~25% 0010: 25%~50% 0011 : 0%~50% 0100: 50%~75% 0101 : 0%~25%, 50%~75% 0110: 25%~75% 0111 : 0%~75% 1000: 75%~100% 1001 : 0%~25%, 75%~100% 1010 : 25%~50%, 75%~100% 1011 : 0%~50%, 75%~100% 1100: 50%~100% 1101 : 0%~25%, 50%~100% 1110: 25%~100% 1111 : 0%~100%	R/W
b7~b4	CKS[3:0]	Counting Clock	0010: PCLK3/4 0110: PCLK3/64 0111: PCLK3/128 1000: PCLK3/256 1001: PCLK3/512 1010: PCLK3/1024 1011: PCLK3/2048 1101: PCLK3/8192 Remaining values: Reserved functions	R/W
b3~b2	Reserved	-	Read "0", write "0" when writing	R

b1~b0	PERI[1:0]	Counting Period	00:256 cycle 01:4096 cycle 10:16384 cycle 11:65536 cycle	R/W
-------	-----------	-----------------	--	-----

24.3.2 Status registers (SWDT_SR, WDT_SR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	REF	UDF
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CNT [15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b18	Reserved	-	Read "1", write "0" when writing	R
b17	REF	Refresh error flag	0: No refresh error 1: Refresh error occurred Read a 1 to this bit and write a 0 to clear the bit.	R/W
b16	UDF	Counting down overflow flag	0: No count underflow 1: Counting underflow occurs Read a 1 to this bit and write a 0 to clear the bit.	R/W
b15~b0	CNT [15:0]	Counting value	Current count value of the counter	R

24.3.3 Refresh registers (SWDT_RR, WDT_RR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RF[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Read "1", write "0" when writing	R
b15~b0	RF[15:0]	Refresh value	After writing 0x0123 and 0x3210 in turn, the refresh action is completed. When the register is written to 0x0123, the readout register is 0x0000_0001; the rest of the cases readout value is 0x0000_0000.	R/W

24.4 Precautions for use

When SWDT operates, the peripheral clock PCLK3 must operate at a frequency greater than or equal to 4 times the count clock (ICG0_SWDTCKS[3:0] sets the SWDTLRC divided clock), i.e. PCLK3 frequency \geq count clock frequency $\times 4$.

25 Universal Synchronous Asynchronous Transceiver (USART)

25.1 Introduction

This product is equipped with 4 units of Universal Serial Transceiver Module (USART). The Universal Serial Transceiver Module (USART) enables flexible full-duplex data exchange with external devices; this USART supports Universal Asynchronous Serial Communication Interface (UART) Clock Synchronous Communication Interface, Smart Card Interface (ISO/IEC7816-3). Supports modem operation (CTS/RTS operation) multi-processor operation. Supports UART receive TIMEOUT function in conjunction with Timer0 module.

USART Key Features:

- Support full duplex asynchronous communication, full duplex clock synchronous communication
- Transmitter and receiver have independent enable bits
- Built-in double buffer
- LSB/MSB optional
- Modem operation support (CTS/RTS)
- Transmission flags: send data register empty, send data complete, receive data register full, receive error flag, UART receive timeout flag

UART main features:

- Programmable data length: 8-bit/9-bit
- Check function can be configured: odd check/even check/no check
- Stop bit configurable: 1 bit / 2 bits
- Clock source selectable: internal clock source (clock generated by internal baud rate generator)/external clock source (clock input from USARTn_CK pin)
- Receiving errors: checksum errors, frame errors, overflow errors
- Supports inter-processor communication
- Built-in digital filter
- Support receiving data TIMEOUT function
- Unit 1 supports stop mode wake-up function

Clock synchronization mode main features:

- Data length: 8 bits
- Receiving error: overflow error
- Clock source: Internal clock source (clock generated by internal baud rate generator)/External clock source (clock input from USARTn_CK pin)

Key features of the smart card interface:

- Data length: 8 bits
- Automatically sends out error signals when calibration errors are detected
- Support data retransmission

25.2 USART System Block Diagram

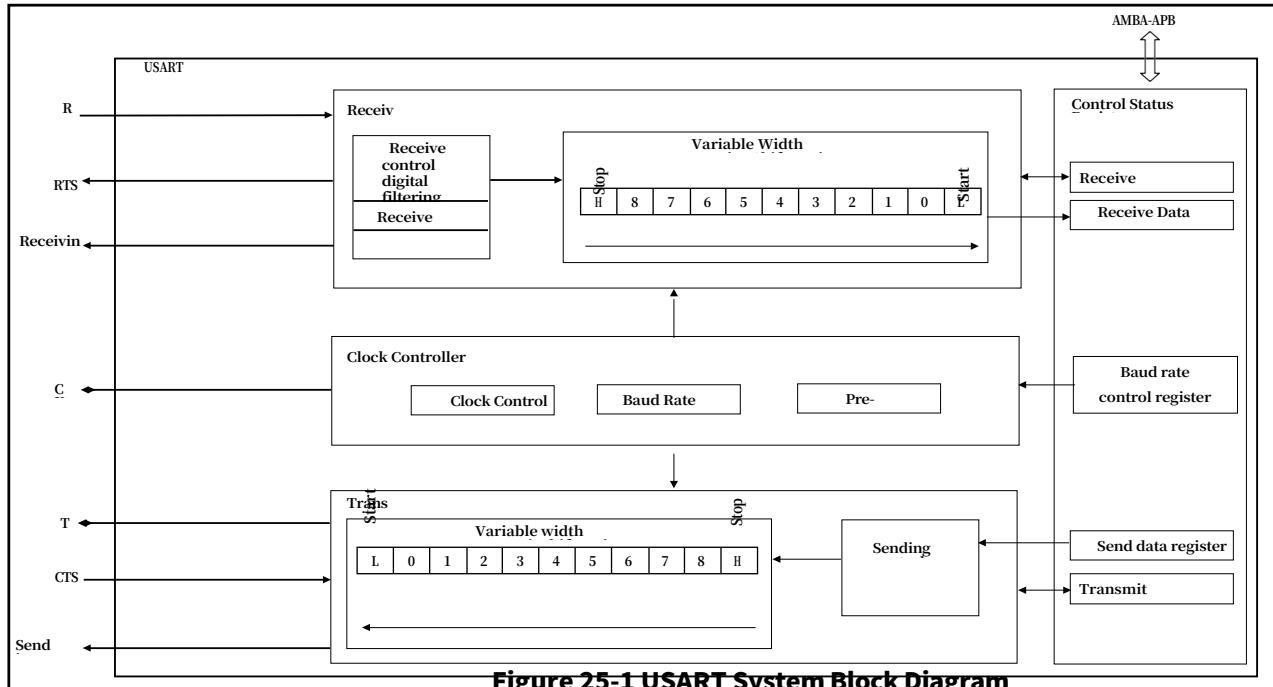


Figure 25-1 USART System Block Diagram

25.3 Pin

Descript ion

Table 25-1 USART Pin Descriptions

Pin Name	Direction	Function Description
USARTn_CK	Input and Output	Clock
USARTn_TX	Output	Send data pin
USARTn_RX	Input	Receive data pin
USARTn_CTS	Input	Modem operation pin clears the transmit pin
USARTn_RTS	Output	Modem operation pin request send pin

n:1~4

25.4 Function Description

This chapter will explain the functions of UART, multi-processor communication, smart card, and clock synchronization mode in detail.

25.4.1 UART

25.4.1.1 Clock

The UART can select either the clock generated by the internal baud rate generator (internal clock source) or the clock input from the USARTn_CK pin (external clock source) as the clock source for communication.

Internal clock source

CLKC[1:0] bits are set to 00b or 01b to select the clock source as the internal clock source, i.e. the clock generated by the internal baud rate generator.

CLKC[1:0]=00b, the USARTn_CK pin is not used as a clock pin and can be used as a normal IO.

CLKC[1:0]=01b when USARTn_CR2.CLKC[1:0]=01b outputs a clock of the same frequency as the communication baud rate from the USARTn_CK pin.

The clock source for the internal baud rate generator is selected by the setting of the USARTn_PR.PSC[1:0] bits as PCLK, PCLK/4, PCLK/16, PCLK/64.

External clock source

CLKC[1:0] bits are set to 10b or 11b to select the clock source as an external clock input from the USARTn_CK pin at 16 times the baud rate (USARTn_CR1.OVER8=0) or 8 times the baud rate (USARTn_CR1.OVER8=1).

Maximum Baud Rate

When the internal clock source is used, the baud rate generated by the internal baud rate generator is calculated using the formula

$$B = \frac{C}{8 \times (2 - \square\square\square\square 8) \times (\text{DIV_Integer} + 1)}$$

B: Baud rate Unit: MBps

C: Clock set by USARTn_PR.PSC[1:0] bits (PCLK,PCLK/4,PCLK/16,PCLK/64) Unit: MHz

OVER8:USARTn_CR1.OVER8 Set value

DIV_Integer:USARTn_BRR.DIV_Integer Set value

maximum baud rate is PCLK/8(MBps).

When the external clock source is external, the maximum frequency required for the external input UART clock is PCLK(MHz)/4, so the maximum baud rate is PCLK/64 (Mbps) (when USARTn_CR1.OVER8=0) or PCLK/32 (Mbps) (when USARTn_CR1.OVER8=1) when the clock source is external input clock.

Note that in addition to the PCLK-based calculation method described above, the UART maximum communication baud rate also needs to refer to the maximum communication baud rate specified in the Electrical Characteristics section.

25.4.1.2 Data Format

In UART mode, a frame of data consists of a start bit, a data bit, a check bit and a stop bit.

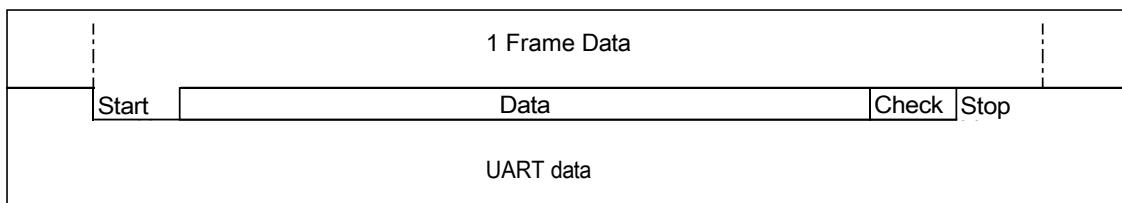


Figure 25-2 UART Data Format

Start position

The start bit is composed of a low level with one bit fixed.

Data bits

The data bits can be configured as 8 or 9 bits.

Check digit

The parity bit can be configured as 1-bit even parity or 1-bit odd parity or no parity bit.

Stop bit

The stop bit is fixed high and can be configured as 1 or 2 bits.

25.4.1.3 Modem operation

The modem operation includes CTS function and RTS function, CTS function and RTS function can only be used at the same time. The RTS function is valid when USARTn_CR3.CTSE=0, and the CTS function is valid when USARTn_CR3.CTSE=1.

CTS Features

The CTS function is to control the data transmission by the input of USARTn_CTS pin. The data can be sent only when the input of USARTn_CTS pin is low, and the data being sent will not be affected if the input of USARTn_CTS is high during the data transmission. **RTS Function**

The RTS function is to request the other party to send data by outputting

low on the USARTn_RTS pin. All of the following conditions need to be met for the USARTn_RTS pin to go low:

- Receive enable (**USARTn_CR1.RE=1**), and no data is being received
- No unread receive data in **USARTn_DR.RDR** register
- No reception errors, including frame errors, checksum errors and overflow errors

25.4.1.4 Transmitter

The transmitter can send 8 or 9 bits of data, depending on the setting of the **USARTn_CR1**.

The transmitter enable bit (**USARTn_CR1.TE**) is set to 1. After writing the transmit data, the transmit data is output serially on the TX pin; the corresponding clock pulse can be output or not on the **USARTn_CK** pin.

The order of sending data is: start bit -> data bit (MSB/LSB) -> check bit (yes or no) -> stop bit. The transmit data register TDR and the internal transmit shift register form a double buffer structure, which can send data continuously.

When writing transmit data through the transmit data register air break or DMA, only one data can be written in one request to ensure the correctness of the transmission.

Sending data setting procedure

1. Set the **USARTn_CR1** register to the reset value
2. Set the pins to be used by the UART
3. Select clock source by **USARTn_CR2.CLKC[1:0]** bits
4. Set **USARTn_CR1**, **USARTn_CR2**, **USARTn_CR3** registers
5. Set **USARTn_PR** to select the prescaler value, and **USARTn_BRR** register to set the communication baud rate (not necessary when the clock source is external)
6. Enables the transmitter (**USARTn_CR1.TE=1**) or set **USARTn_CR1.TXEIE=1** if you need to use the transmit data register air break (TE and TXEIE bits are written to 1 at the same time)
7. Wait for the send data register to be empty, write communication data to **USARTn_DR.TDR**, data transfer to the send shift register, send start

(When the CTS function is active, the data is transferred to the transmit shift register when the **USARTn_CTS** input is low, and transmission starts)

8. If you need to send data continuously, repeat step 7
9. Confirm that the send is complete by acknowledging the **USARTn_SR.TC** bit. In case of sending data continuously and using the send interrupt, the last send data can be written through the TI interrupt and the send completion interrupt is generated after the last frame of data is sent by writing 0 to **USARTn_CR1.TXEIE** and 1 to **USARTn_CR1.TCIE**.

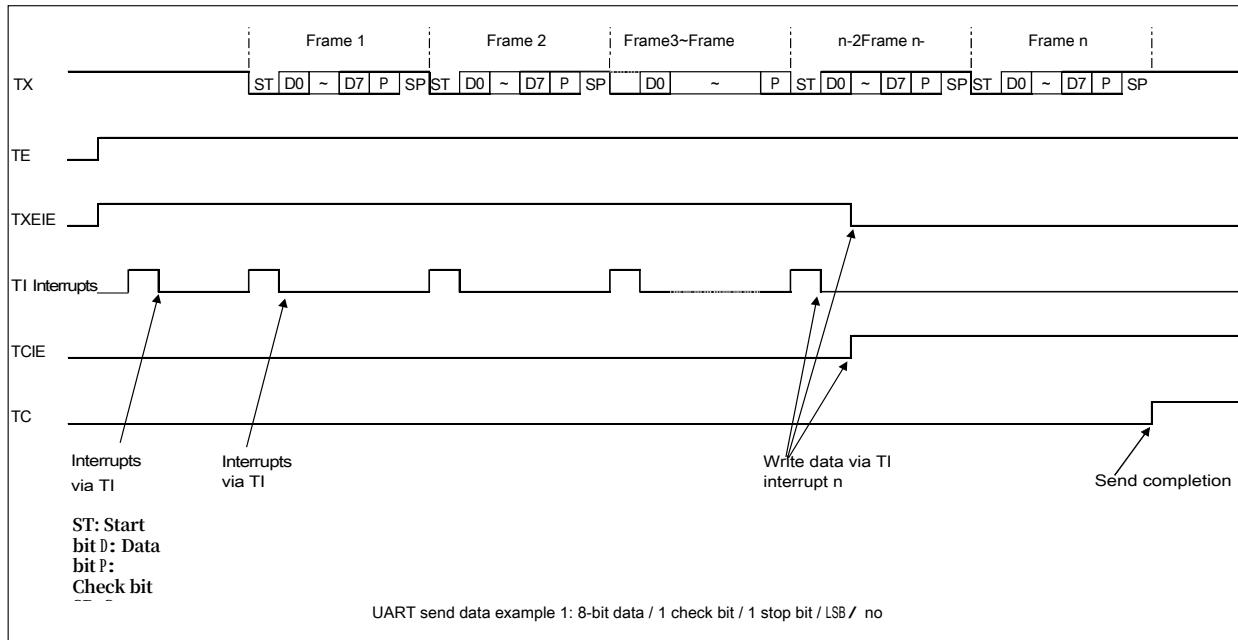


Figure 25-3 UART Transmit Data Legend 1

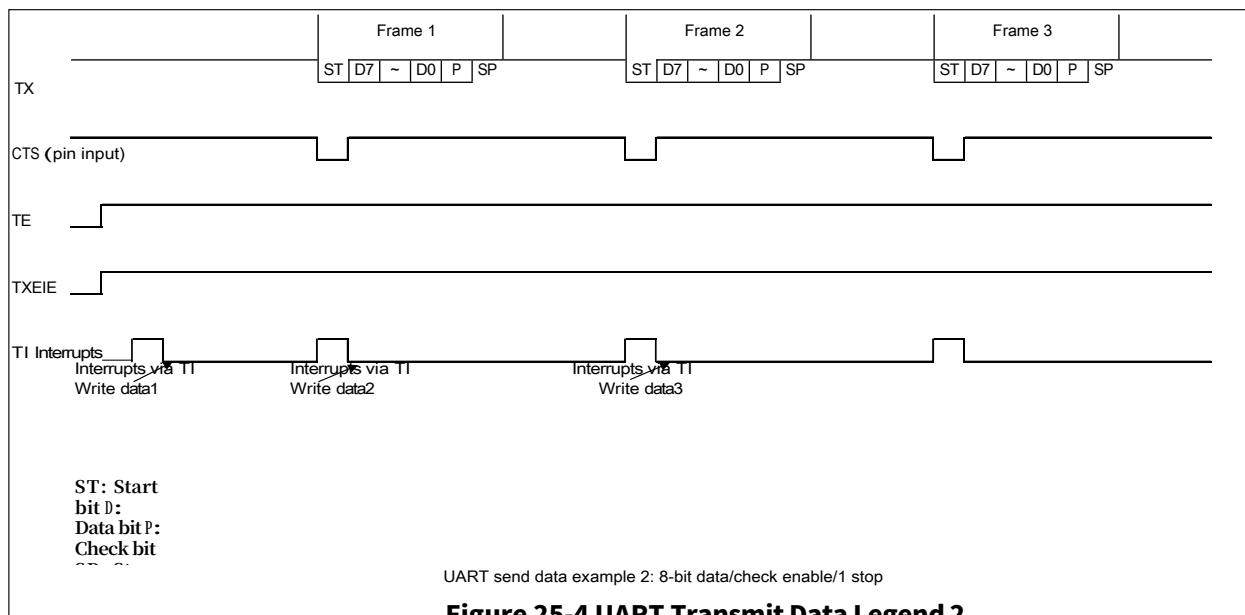


Figure 25-4 UART Transmit Data Legend 2

Transmitter interrupts

The UART mode transmitter supports two types of interrupts, the send data register air interrupt **TI** and the send completion interrupt **TCI**. **TXEIE=1**, **TI** interrupt occurs when the value of **USARTn_DR**. **TCIE=1**, the **TCI** interrupt occurs when the last bit of the data is sent and the **USARTn_DR**.

25.4.1.5 Receiver

The receiver can receive 8-bit or 9-bit data, depending on the setting of the USARTn_CR1. After the receiver enable bit (USARTn_CR.RE) is set to 1 and the start bit is detected, data on the RX pin is received into the receive shift register, a full frame of data is received, and the data is transferred from the receive shift register to the receive data register USARTn_DR.

The order of receiving data is: Start bit -> Data bit (MSB/LSB) -> Checksum bit (yes or no) -> Stop bit.

The receive data register USARTn_DR, RDR register and the internal receive shift register form a double buffer structure, which can receive data continuously.

When reading receive data via receive data register full interrupt or DMA, only one data can be read at one request.

Start bit detection

SBS=0 for low level detection and USARTn_CR1.SBS=1 for falling edge detection.

Sampling and reception tolerances

When a start condition (low or falling edge) is detected, the USART will clock synchronize the received data based on the internal base clock to start data reception.

The data is sampled in the data center, USARTn_CR1.OVER8=0 at the 8th internal base clock and USARTn_MR.OVER8=1 at the 4th internal base clock.

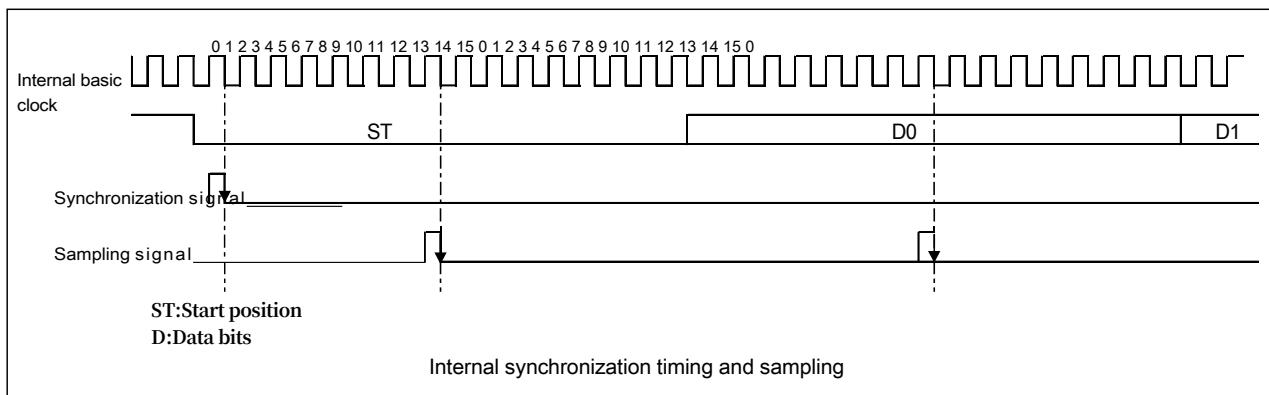


Figure 25-5 UART Internal Synchronization and Sampling Timing

The UART asynchronous receiver will only function properly if the total clock system deviation is less than the tolerance of the UART receiver. Factors that affect the total deviation include:

- Deviations caused by transmitter errors (which also include deviations in the local oscillator of the transmitter)
- Error caused by baud rate quantization of the receiver
- Deviation of the receiver local oscillator

- Deviations caused by transmission lines

- The maximum deviation allowed by the UART asynchronous receiver for correct data reception depends specifically on the following options:
- The data length FL. FL is determined by the 8 or 9 data bits defined by the M bit in the USART_CR1 register and the parity enable bit defined by the PCE bit
- 8x or 16x oversampling as defined by the OVER8 bit in the USART_CR1 register
- Whether to use the fractional baud rate as defined by the FBME bit in the USART_CR1 register

Table 25-2 UART Receiver Tolerance when DIV_Fraction is 0

FL	OVER8 bits = 0	OVER8 bits = 1
10	4.375%	3.75%
11	3.97%	3.41%
12	3.646%	3.125%

Table 25-3 UART Receiver Tolerance when DIV_Fraction is not 0

FL	OVER8 bits = 0	OVER8 bits = 1
10	3.88%	3%
11	3.53%	2.73%
12	3.23%	2.5%

In the special case when the received frame contains idle frames of 10 bits or

11 bits or 12 bits of time, Table 25-2 and

The data specified in Table 25-3 may be slightly different.

Receiving data setting procedure

1. Set the USARTn_CR1 register to the reset value
2. Set the pins to be used by the UART
3. Select clock source by USARTn_CR2.CLKC[1:0] bits
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3 registers
5. Set USARTn_PR to select the prescaler value, and USARTn_BRR register to set the communication baud rate (not necessary when the clock source is external)
6. Enables the receiver (USARTn_CR1. RE=1) or set USARTn_CR1. RIE=1 if a receive interrupt is required
7. When the start bit is detected, the receiver receives the data into the receive shift register and checks the parity and stop bits
 - 1) In case of a calibration error, the received data is transferred to the USARTn_DR.RDR register and the USARTn_SR.PE flag is set

-
- 2) When the stop bit is not high, a frame error occurs and the received data is transferred to the USARTn_DR.RDR register and set

Bit USARTn_SR.FE Flag

- 3) Data is lost and the USARTn_SR.ORE flag is set when an overflow error occurs
- 4) When no error occurs, the received data is transferred to the USARTn_DR.RDR memory and the USARTn_SR.RXNE flag is set, the received data is read and the data is received continuously by repeating step 7.

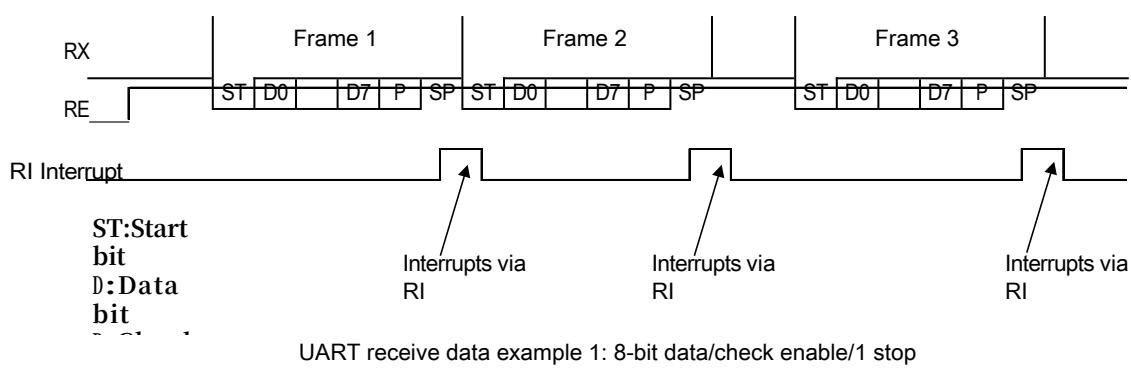


Figure 25-6 UART Receive Data Legend 1

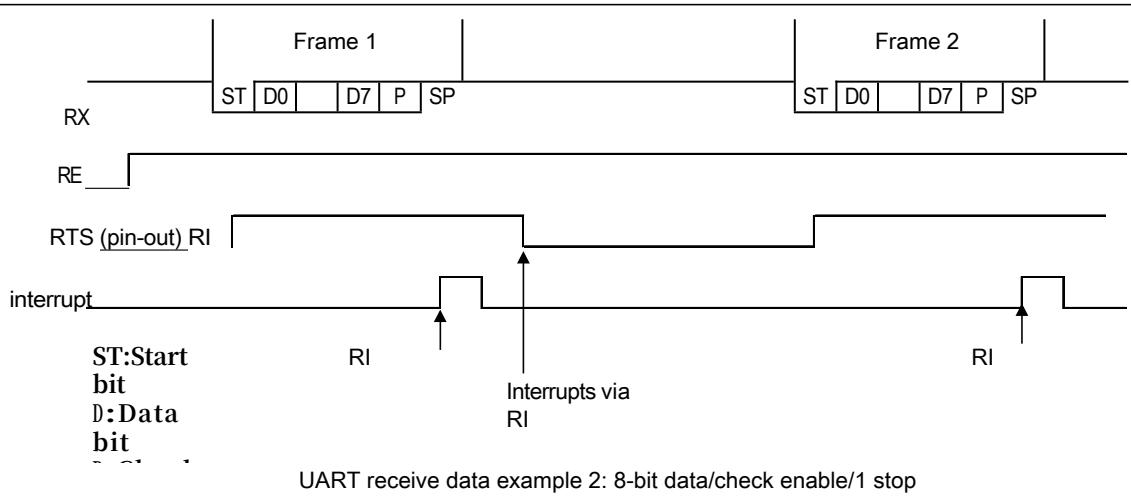


Figure 25-7 UART Receive Data Legend 2

Error

Handling

g

There are three types of receive errors when receiving data, namely overflow error (USARTn_SR.ORE), checksum error (USARTn_SR.PE) and frame error (USARTn_SR.FE). No further data reception can be performed if any of the reception errors occur. Data reception can be restarted by clearing all error flags by writing the corresponding clear register.

The overflow error occurs when a new frame of data is received without the USARTn_DR.RDR

register value being read, so the previous frame of data should be read before the last bit of the current frame is received.

A parity error occurs when a parity error has occurred.

The frame error occurs when the stop bit is low, and only the first stop bit is checked in the case of 2 stop bits. The received data is lost when an overflow error occurs, and the RI interrupt does not occur.

The data received in case of a checksum error is transmitted to USARTn_DR.RDR, and the RI interrupt does not occur. The received data is transmitted to USARTn_DR.RDR when a framing error occurs, and the RI interrupt does not occur. **Receiver interrupt**

The UART mode receiver supports two types of interrupts, receive data register full interrupt RI and receive error interrupt **REI**.

USARTn_CR.RIE=1, no receive error occurred, RI interrupt occurs when data is transferred from receive shift register to receive data register.

USARTn_CR.RIE=1, **REI** interrupt occurs in case of overflow error, checksum error or frame error during reception.

25.4.1.6 UART receive TIMEOUT function

The TIMEOUT counter starts when the UART receive data stop bit is detected, and TIMEOUT occurs when the next received data frame is not detected after the set TIMEOUT time (the set unit is the number of received bits). RE=0, then wait for USARTn_CR1.RE=1 and then the TIMEOUT status bit USARTn_SR.

The TIMEOUT counter uses the Timer0 module counter, which corresponds to the following: USART1: Timer0 Unit1 A channel

USART2: **Timer0 Unit1 B channel**

USART3: Timer0 Unit2 A channel

USART4: Timer0 Unit2 B channel

TIMEOUT Function Timer0 Compare

counter value setting

Timer0 is a 16-bit counter with a maximum selectable count clock of 1024 divisions. The TMR0_CMPAR value is calculated as follows:

$$\text{CMPA } < \text{B} > \text{R} = \frac{\text{RTB}}{2^{\text{CKDIVA } < \text{B} >}} - 4 \text{ (count clock not divided)} \\ \text{CMPA } < \text{B} > \underline{\text{RTB}}_{2^{\text{CKDIVA } < \text{B} >}} - 2 \text{ (count clock 2 divided)}$$

CMPA R = $\frac{1}{2 \text{CKDIVA}_{}}$ - 1 (count clock 4 divisions and
above) CMPAR : TMR0_CMPAR register value

RTB: Receive Timeout Bits

CKDIRA : TMR0.BCONR.CKDIVA bit register value

Timeout function setting procedure

1. Set the USARTn_CR1 register to the reset value
2. Set the pins to be used by the UART
3. Select the clock source via USARTn_CR2.CLK[1:0] bits to select the clock source (CR2.CLKC[0]=1 if internal clock source is selected)
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3 registers
5. Set USARTn_PR to select the prescaler value, and USARTn_BRR register to set the communication baud rate (not necessary when the clock source is external)
6. Clear the USARTn_SR.RTOF flag, set USARTn_CR1.RTOE=1, and set USARTn_CR1.RTOIE=1 if interrupt is needed
7. Set TMR0.BCONR.CSTA=0
8. Set TMR0_CNTAR to 0, set the TMR0_CMPAR register and the TMR0.BCONR.CKDIVA register to determine the TIMETOUI time, and set the TMR0_CMPAR register to 0.
Set TMR0.BCONR.HCLEA=1 and TMR0.BCONR.HSTAA=1
BCONR.ASYNCLKA=1, TMR0.BCONR.SYNSA=1
9. Enables the receiver (USARTn_CR1.RE=1) or set USARTn_CR1.RIE=1 if a receive interrupt is required
10. Set TMR0.BCONR.CSTA=0 after TIMEOUT is detected.

Set TMR0.BCONR.CSTA=0 to clear USARTn_SR.RTOF state by writing USARTn_CR1.CRTOF Bit.

25.4.1.7 RX line wake-up stop mode function

When the UART communication is idle, the system can be put into stop mode to save current consumption. UART Unit 1 can wake up the system in stop mode through the RX line without changing the UART PORT setting. The specific steps are as follows:

1. When UART communication is idle, set the **USART_1_WUPI** interrupt vector and INT_WUPEN. RXWUEN bit to enable the UART receive signal line wakeup stop mode function.
2. The system enters stop mode.
3. When the system detects the falling edge of the RX line, it returns from the stop mode and turns off the function in the **USART_1_WUPI** interrupt handler.

It should be noted that when the communicating party needs to wake up the system, it needs to send a frame of wake-up data (0x00 is recommended), **which will** not be received by

the UART and will not set the relevant flags. And the communicating party needs to pass the time required for the system to wake up in stop mode before transmitting the UART communication data.

25.4.1.8 UART Interrupts

and Events

Table 25-4 UART interrupt/event table

Function Name	Marker	Enable bit (interrupt only)	Logic position	Can it be used as an event source
Receiving error interrupts	REI	RIE	ORE,FE,PE	may
Receive data register full interrupt	RI	RIE	None	may
Send data register air break	TI	TXEIE	None	may
Send completion interrupt	TCI	TCIE	TC	may
TIMEOUT interrupt	RTOI	RTOIE	RTOF	may
RX line wake-up stop mode interrupt	WUPI	INT_WUPEN.RXWUEN	—	No

25.4.2 Multi-processor communication

25.4.2.1 Function Introduction

The multi-processor communication mode is a communication mode in which multiple processors share communication lines, and the processors are divided into sending and receiving stations, each of which has its own unique ID, and the sending station sends two types of data: receiving station ID and communication data. The MPB bit is added to the data format to distinguish whether the current frame is the ID of the receiving station or the communication data; if the MPB bit is 0, the current frame is the communication data, and if the MPB bit is 1, the current frame is the ID of the receiving station. If not, it enters silent mode (neither data is received nor the receive-related flag is set) until the ID is received again.

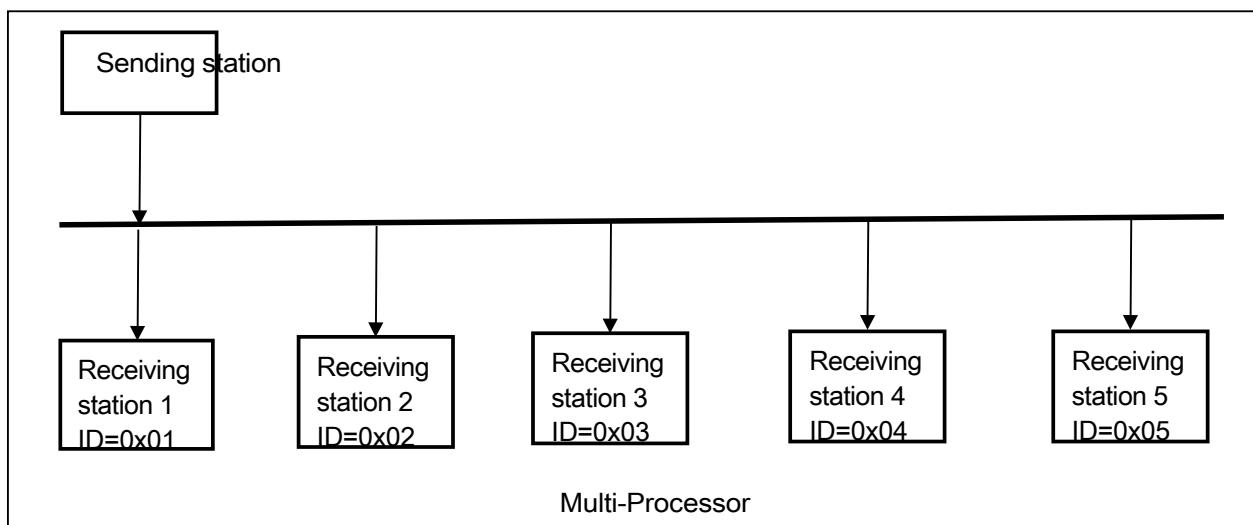


Figure 25-8 Multiprocessor Communication Legend

25.4.2.2 Data Format

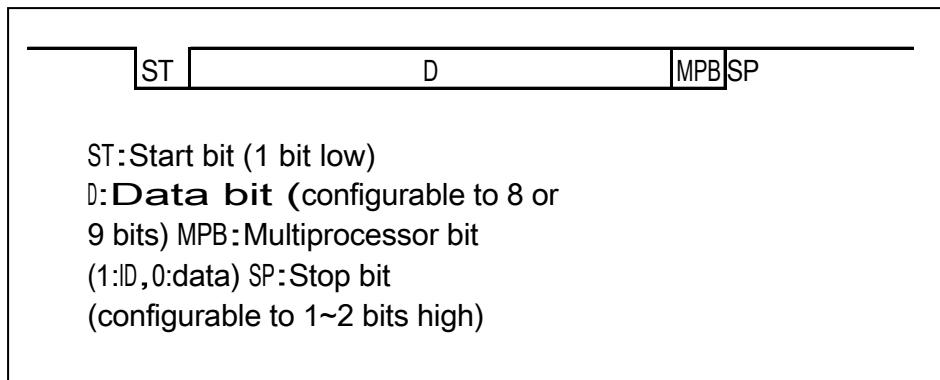


Figure 25-9 Multiprocessor Mode Data Format

25.4.2.3 Action

Description

In multi-processor mode, the parity bit function is invalid and the multi-processor bit function is added. The rest of the functions such as clock and interrupt are the same as UART mode.

Sending station action

1. Set the USARTn_CR1 register to the reset value
2. Set the pins to be used
3. Select clock source by USARTn_CR2.CLKC[1:0] bits
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3 registers
5. Set USARTn_PR to select the prescaler value, and USARTn_BRR register to set the communication baud rate (not necessary when the clock source is external)
6. Enables the transmitter (USARTn_CR1. TE=1) or set USARTn_CR1. TXEIE=1 if you need to use the transmit data register air break (TE and TXEIE bits are written to 1 at the same time)
7. Wait for the transmit data register to be empty, set the USARTn_DR.MPID bit to 1 (transmit ID), write the ID value to USARTn_DR, transmit ID
(When the CTS function is active, the data is transferred to the transmit shift register when the USARTn_CTS input is low, and transmission starts)
8. Set USARTn_DR.MPID bit to 0 (send data), write data to USARTn_DR, send data
(When CTS function is active, data is transferred to send shift register when USARTn_CTS input is low, and send starts)
9. Repeat step 8 if you need to send data continuously, and repeat 7 and 8 if you need to change the ID before sending data.
10. Confirm that the send is complete by acknowledging the USARTn_SR.TC bit. In the case of

sending data continuously and using the send interrupt, the last send data can be written through the TI interrupt and USARTn_CR1. TXEIE is written to 0 in USARTn_CR1. TCIE is written to 1. The send completion interrupt is generated after the last frame of data is sent.

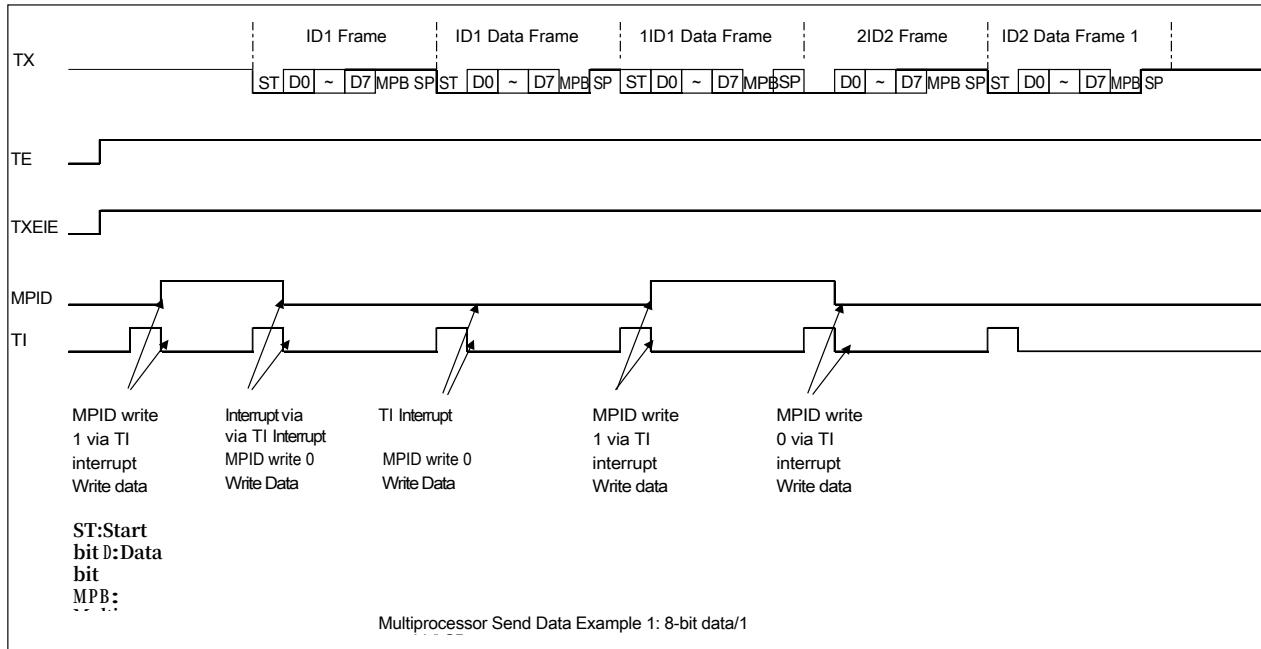


Figure 25-10 Example of multi-processor mode sending data

Receiving

station

action

In multiprocessor mode, the receiving station must ensure that it can receive each ID data, compare it with its own ID, and if it agrees, receive the data, and if not, enter silent mode (no data is received and no receive-related flags are set) until the next ID data is received. This is achieved through the USARTn_CR1.SLME bit.

Data is received normally when USARTn_CR1.SLME=0.

When USARTn_CR1.SLME=1, no data is received, no RI interrupt occurs, and the error flag FE, ORE is not set unless data with MPB bit 1 (D1) is received. When data with MPB bit 1 is received (D1), USARTn_CR1.SLME bit is automatically cleared, data is received normally and interrupt occurs.

Action steps:

1. Set the USARTn_CR1 register to the reset value
2. Set the pins to be used
3. Select clock source by USARTn_CR1.CLKC[1:0] bits
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3 registers
5. Set USARTn_PR to select the prescaler value, and USARTn_BRR register to set the communication baud rate (not necessary when the clock source is external)
6. USARTn_CR1.RE=1, USARTn_CR1.SLME=1 (wait for receive ID), and set USARTn_CR1.RIE=1 if receive interrupt is used

-
7. When the start bit is detected, the receiver receives the data into the receive shift register and checks the USARTn_SR.MPB bit
 8. If USARTn_SR.MPB=1, the USARTn_CR1.SLME bit is automatically cleared, data is received normally, and the software compares

Received ID and own ID

- 1) If the ID is consistent, the data is received normally, and an interrupt occurs for error detection, the same as for UART received data
- 2) If the IDs do not match, the software writes 1 to the USARTn_CR1.SLME bit again and repeats the action of 8

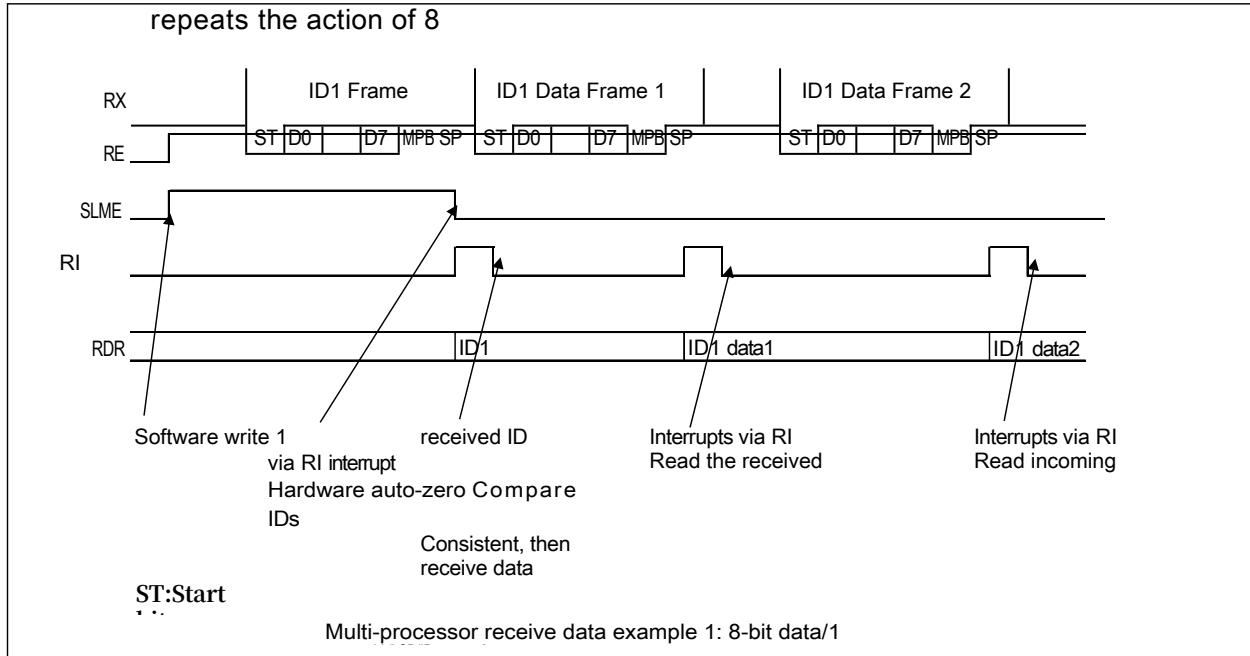


Figure 25-11 Multiprocessor Mode Receive Data Legend 1

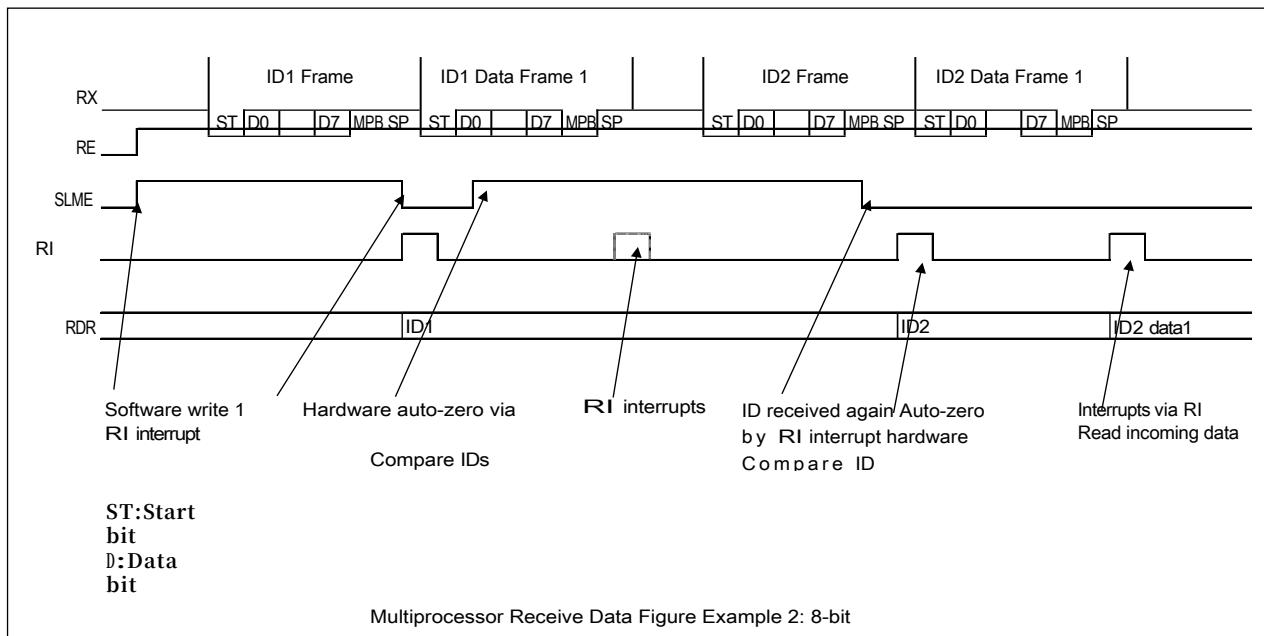


Figure 25-12 Multiprocessor Mode Received Data Legend 2

25.4.2.4 Interrupts and events

Multiprocessor mode has the same interrupt handling as UART mode except no checksum errors.

Table 25-5 Multiprocessor Mode Interrupt/Event Table

Function Name	Marker	Enable bit (interrupt only)	Log o	Can it be used as an event source
Receiving error interrupts	REI	RIE	ORE,FE	No
Receive data register full interrupt	RI	RIE	RXNE	may
Send data register air break	TI	TXEIE	TXE	may
Send completion interrupt	TCI	TCIE	TC	No

25.4.3 Smart Card

25.4.3.1 Connection schematic

The smart card communication protocols specified in ISO/IEC 7816-3 are supported. The diagram below shows the connection diagram for smart card mode.

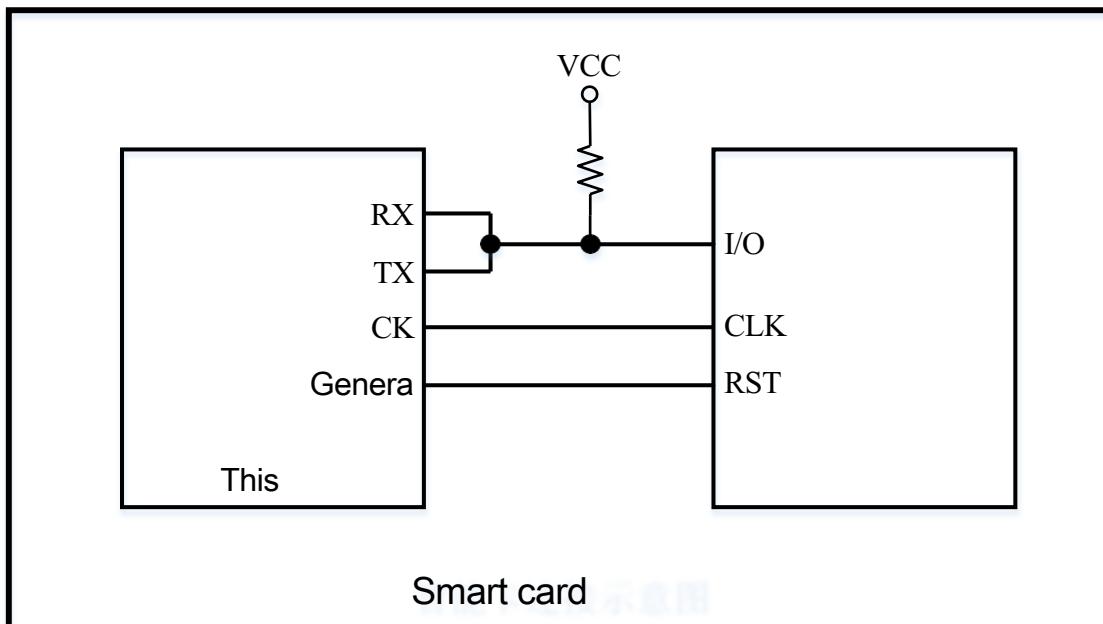


Figure 25-13 Smart Card Connection Diagram

25.4.3.2 Clock

Internal clock source

Only the clock generated by the internal baud rate generator can

be used as the clock source during smart card mode. The base

clock number for one data transmission is the

USARTn_CR3.BCN[2:0] setting.

The clock output of smart card mode is controlled by setting the register USARTn_CR2.CLKC[1:0] bits.

Maximum Baud Rate

When the internal clock source is used, the baud rate generated by the internal baud rate generator is calculated using the formula

$$B = \frac{C}{2 \times BCN \times (DIV_Integer + 1)}$$

B: Baud rate Unit: MBps

C: Clock set by USARTn_PR.PSC[1:0] bits (PCLK,PCLK/4 ,PCLK/16,PCLK/64) Unit:
MHz

DIV_Integer:USARTn_BRR.DIV_Integer set

value BCN:USARTn_CR3.BCN register set value

When C is PCLK, DIV_Integer=0, and BCN=0, the baud rate is PCLK/64(MBps), the highest baud rate.

Sampling and reception tolerances

After the falling edge of RX is detected, the USART will clock synchronize the received data based on the internal base clock to start data reception. The receive data will be sampled at the center of the data.

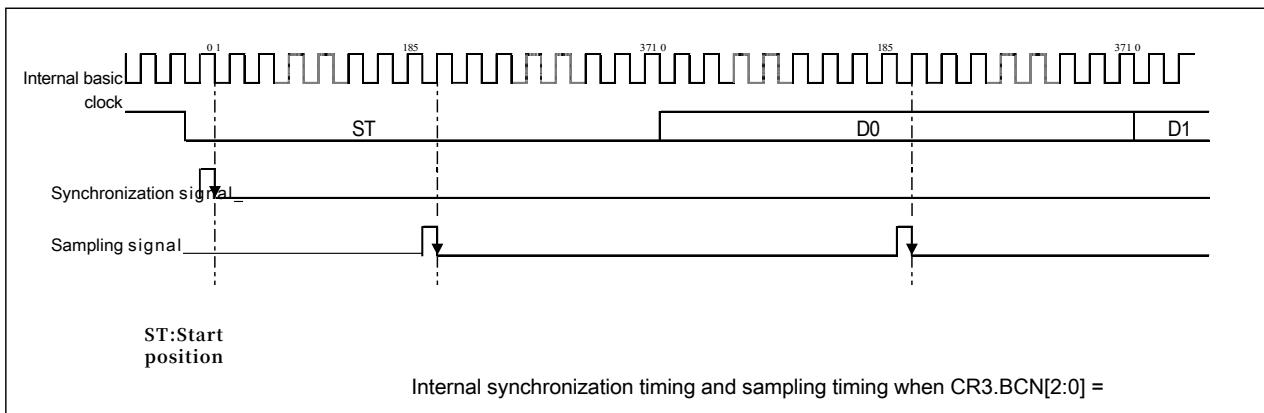


Figure 25-14 Smart Card Mode Synchronization Timing and Sampling Timing Diagram

The receiving tolerance is calculated as follows:

$$RM[\%] = |0.5 \times (1 - \frac{1}{BCN}) - 9.5| \times 100$$

RM: Receiving

□

Tolerance

BCN: Number of clocks required for one bit data transmission

(USARTn.CR3.BCN[2:0] setting value) CFD: Clock frequency

deviation

25.4.3.3 Data Format

In smart card mode, a frame of data consists of a start bit, a data bit and a check bit.

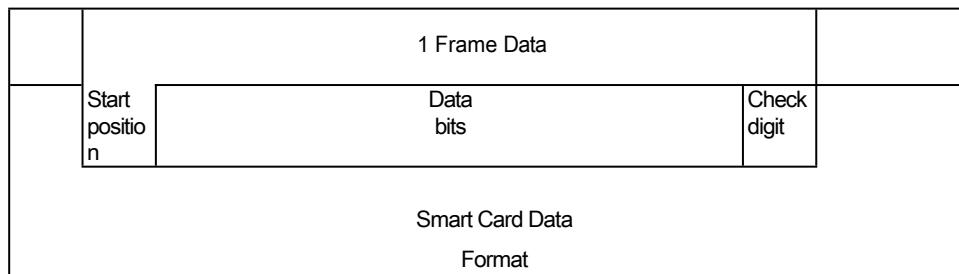


Figure 25-15 Smart Card Mode Synchronization Timing and Sampling Timing Diagram

**Starti
ng
positi
on**

The start bit is fixed with one low level.

Data bits

The data bits are fixed to 8-bit data.

Check digit

The parity bit needs to be configured as 1-bit even parity.

25.4.3.4 Smart card initialization setting steps

1. Set the USARTn_CR1 register to the reset value
2. Set the pins to be used
3. Status register confirmation, USARTn_SR register set to reset value
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3 registers
5. Set USARTn_PR to select the prescaler value and USARTn_BRR register to set the communication baud rate
6. USARTn_CR2.CLKC[1:0] bits set clock control
7. USARTn_CR1 register (TE, RE, RIE, TXEIE bits) setting, except for self-test, TE and RE should not be set to 1 at the same time

When switching from transmit mode to receive mode, or when switching from receive mode to transmit mode, you need to reset steps 1 to 7 above.

25.4.3.5 Smart Card Mode Action Description

TC=1 and USARTn_CR1.TXEIE=1.

Function Overview

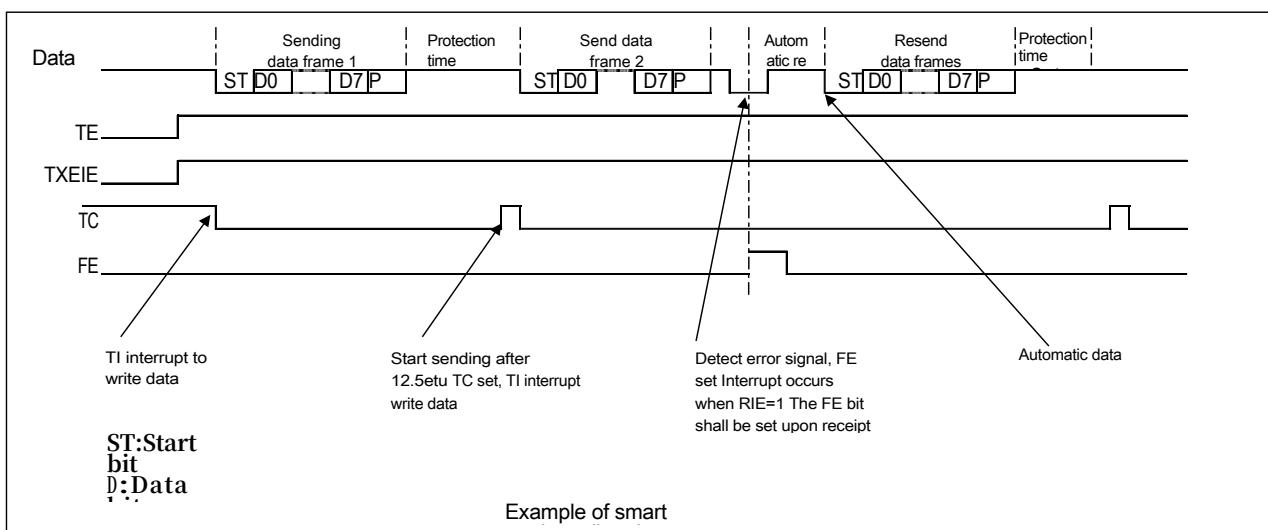
When sending data between two frames of data
(from the end of the parity bit to the start of the next frame) there is 2etu (Elementary Time Unit) or more.

If an error signal from the receiver is detected when sending data, the data is automatically retransmitted after 2etu.

When a check error occurs in the received data, a low level of 1etu is sent, which is the error signal, and the timing of the error signal is sent after 10.5etu at the beginning of reception.

Sending instructions

1. After a frame is sent, if an error signal from the receiver is detected, USARTn_SR.FE is set to 1 (if USARTn_CR.RIE=1, an error interrupt occurs) the USARTn_SR_TC flag is not set to 1, and the data is automatically resent. The USARTn_SR.FE bit must be cleared before the next frame check bit is accepted.
2. After a frame of data is sent without errors, the USARTn_SR.TC flag is set and a TI interrupt occurs when USARTn_CR1.TXEIE=1. The data is written again, and then the



data can be sent continuously.

Figure 25-16 Example of smart card mode sending data

Receiving Instructions

1. When receiving data, if a checksum error is detected, USARTn_SR.PE is set and a REI interrupt occurs when the interrupt is enabled. the USARTn_SR.PE bit needs to be cleared before the next frame of checksum bits is received.
2. When a checksum error occurs, a low level of 1etu, the error signal, is sent, asking the sender to resend the data.
3. Normal reception of data can be received continuously by reading the received data through the RI interrupt.
4. Overflow error detection when receiving data.

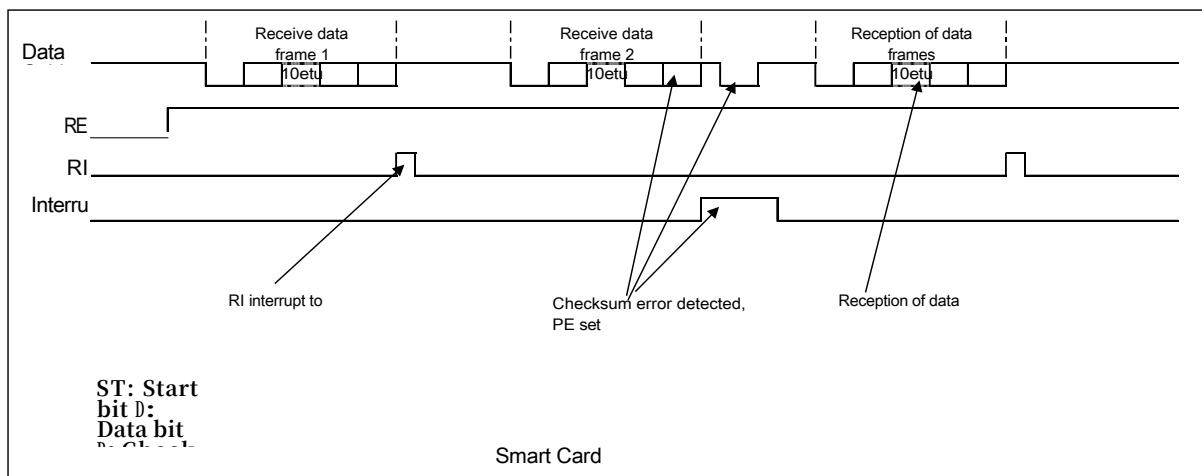


Figure 25-17 Example of smart card mode receiving data

25.4.3.6 Interrupts

and
events

Table 25-6 Smart Card Mode Interrupt/Event Table

Function Name	Marker	Enable bit (interrupt only)	Logo position	Can it be used as an event source
Error interruption	REI	RIE	ORE,PE,FE	may
Receive data full interrupt	RI	RIE	RXNE	may
Sending data over the air	TI	TXEIE	TC	may

25.4.4 Clock synchronization mode

25.4.4.1 Clock

The clock synchronization mode allows you to select either the clock generated by the internal baud rate generator (internal clock source) or the clock input from the USARTn_CK pin (external clock source) as the clock source for communication.

Internal clock source

The synchronous clock is output from the USARTn_CK pin. 8 clock pulses are output for one frame of data, and the clock output is fixed high when neither data is sent nor received.

External clock source

The external clock source, i.e., the input clock from the USARTn_CK pin, is used as the communication clock.

Maximum Baud Rate

When the internal clock source is used, the baud rate generated by the internal baud rate generator is calculated using the formula

$$B = \frac{C}{4 \times (\text{DIV_Integer} + 1)}$$

B: Baud rate Unit: MBps

C: Clock set by USARTn_PR.PSC[1:0] bits (PCLK,PCLK/4 ,PCLK/16,PCLK/64) Unit:
MHz

DIV_Integer:USARTn_BRR.DIV_Integer Set value

The maximum baud rate is PCLK/8(MBps) when C is PCLK and DIV_Integer=1 for internal clock source. Note that **DIV_Integer** is not allowed to be set to 0 in synchronous mode.

For external clock source, the maximum frequency of external input clock is required to be PCLK(MHz)/6, so the maximum baud rate is PCLK/6(MBps).

Note that in addition to the PCLK-based calculation described above, the maximum communication baud rate for synchronous mode should also be referred to the maximum communication baud rate specified in the Electrical Characteristics section.

25.4.4.2 Data Format

Clock synchronization mode has a fixed 8-bit data frame, and 8 synchronized clock pulses are required to send and receive a frame of data. When sending data, the data is sent on the falling edge of the synchronous clock, and when receiving data, the data is sampled on the rising edge of the synchronous clock.

The synchronous clock is fixed high when no data is transmitted and the communication line holds the value of the last bit after the last bit is sent.

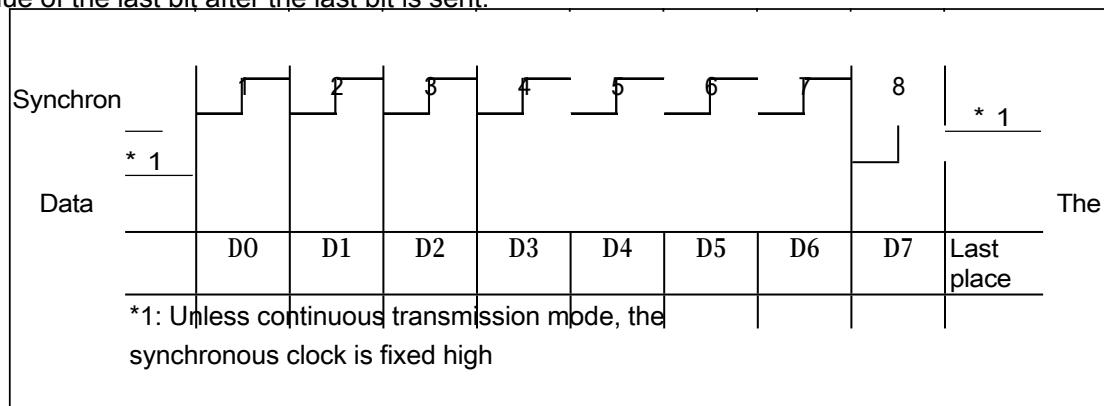


Figure 25-18 Clock Synchronization Mode Data Format

25.4.4.3 Modem operation

The modem operation includes CTS function and RTS function; CTS function and RTS function can only be selected one at a time, not both. The RTS function is valid when USARTn_CR3.CTSE=0, and the CTS function is valid when USARTn_CR3.CTSE=1.

CTS Features

The CTS function is to control the data transmission by the input of USARTn_CTS pin. The data can be sent only when the input of USARTn_CTS pin is low, and the data being sent will not be affected if the input of USARTn_CTS is high during the data transmission. **RTS Function**

The RTS function is to request the other party to send data by outputting

a low level on the USARTn_RTS pin. All of the following conditions need

to be met for the USARTn_RTS pin to go low:

- Receive enable (USARTn_CR1.RE=1), and no data is being received
- No unread data in USARTn_DR.RDR register (when USARTn_CR1.RE=1)
- USARTn_DR.TDR update completed (when USARTn_CR1.TE=1)
- No reception errors

If all the above conditions are not met at the same time, USARTn_RTS outputs high.

25.4.4.4 Transmitter

When the transmitter enable bit (USARTn_CR1.TE) is set to 1, the data in the transmit shift register is output serially on the USARTn_TX pin and the corresponding clock pulse is output on the USARTn_CK pin.

The transmit data register USARTn_DR.TDR and the internal transmit shift register form a double buffer structure, which can send data continuously.

When writing transmit data through the transmit data register air break or DMA, only one data can be written in one request to ensure the correctness of the transmission.

Sending data setting procedure

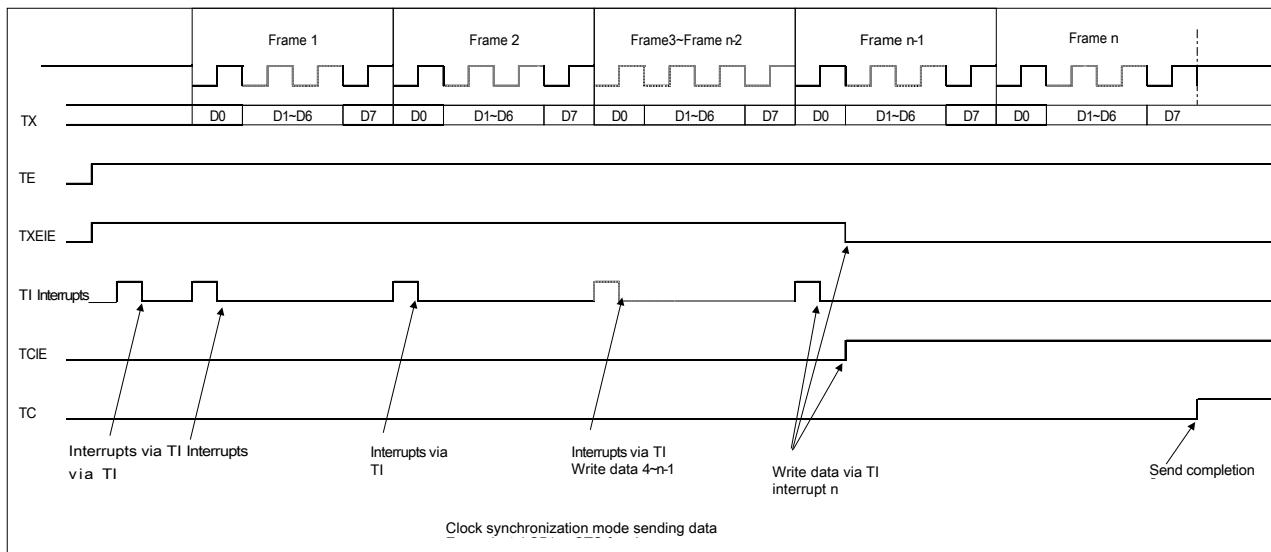
1. Set the USARTn_CR1, USARTn_SR1 registers to reset values
2. Set the pins to be used
3. Select clock source by USARTn_CR2.CLKC[1:0] bits
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3 registers
5. Set USARTn_PR to select the prescaler value, and USARTn_BRR register to set the communication baud rate (not necessary when the clock source is external)
6. Enable the transmitter (USARTn_CR1.TE=1) and set USARTn_CR1.

TXEIE=1 (TE and TXEIE bits are written to 1 at the same time) if you need to use the transmit data register air break

7. Wait for the send data register to be empty, write communication data to USARTn_DR.TDR, data transfer to the send shift register, send start

(When the CTS function is active, the data is transferred to the transmit shift register when the USARTn_CTS input is low, and transmission starts)

8. If you need to send data continuously, repeat step 7
9. Confirm that the send is complete by acknowledging the USARTn_SR.TC bit. In case of sending data continuously and using the send interrupt, the last send data can be written through the TI interrupt and USARTn_CR1_. TXEIE is written to 0 and USARTn_CR1. TCIE is



written to 1. The send completion interrupt is generated after the last data is sent.

Figure 25-19 Clock Synchronization Mode Sending Data

Example 1

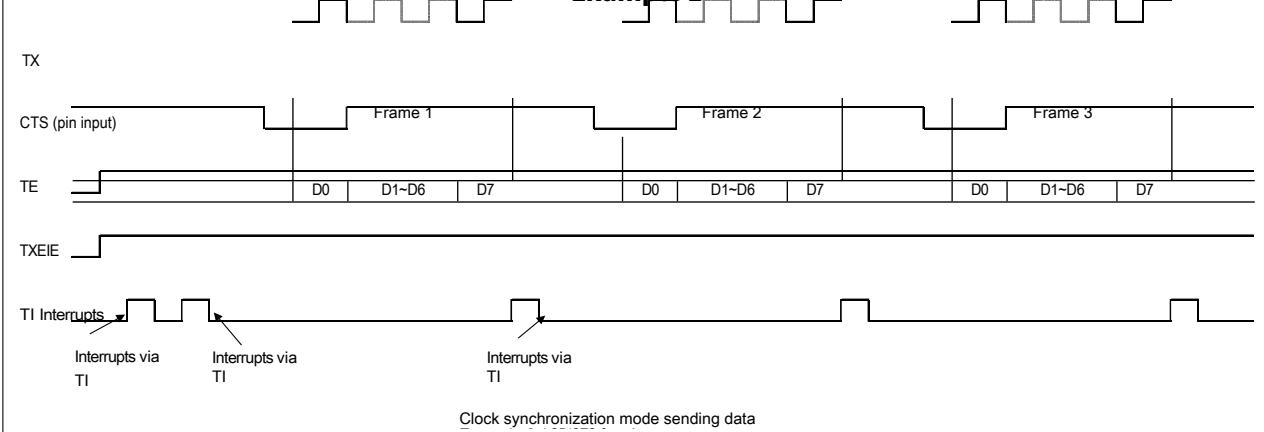


Figure 25-20 Clock synchronization mode sending data example 2

Transmitter interrupts

The clock synchronous mode transmitter supports two interrupts, the send data register

air interrupt TI and the send completion interrupt TCI. TXEIE=1, the TI interrupt occurs

when the value of USARTn_DR. TCIE=1, the **TCI** interrupt occurs when the last bit of the data is sent and the USARTn_DR..

25.4.4.5 Receiver

Receiving data setting procedure

1. Set USARTn_CR1, USARTn_SR registers to reset value
2. Set the pins to be used
3. Select clock source by USARTn_CR2.CLKC[1:0] bits
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3 registers
5. Set USARTn_PR to select the prescaler value, and USARTn_BRR register to set the communication baud rate (not necessary when the clock source is external)
6. Enables the receiver (USARTn_CR1.RE=1) and if a receive interrupt is required, set USARTn_CR1.RIE=1
(When using the RTS function, the USARTn_RTS output goes low after RE=1)
7. Synchronized to the input synchronous clock or internally generated synchronous clock to start receiving data, receiving data to the receive shift register.
 - 1) Data is lost and the USARTn_SR.ORE flag is set when an overflow error occurs
 - 2) When no error occurs, the received data is transferred to the USARTn_DR.RDR register, the USARTn_SR.RXNE flag is set, and the current received data is read before the last bit of the received data in the next frame, and then step 7 is repeated to achieve continuous data reception.

(When using the RTS function, the USARTn_RTS output goes low after the data is read)

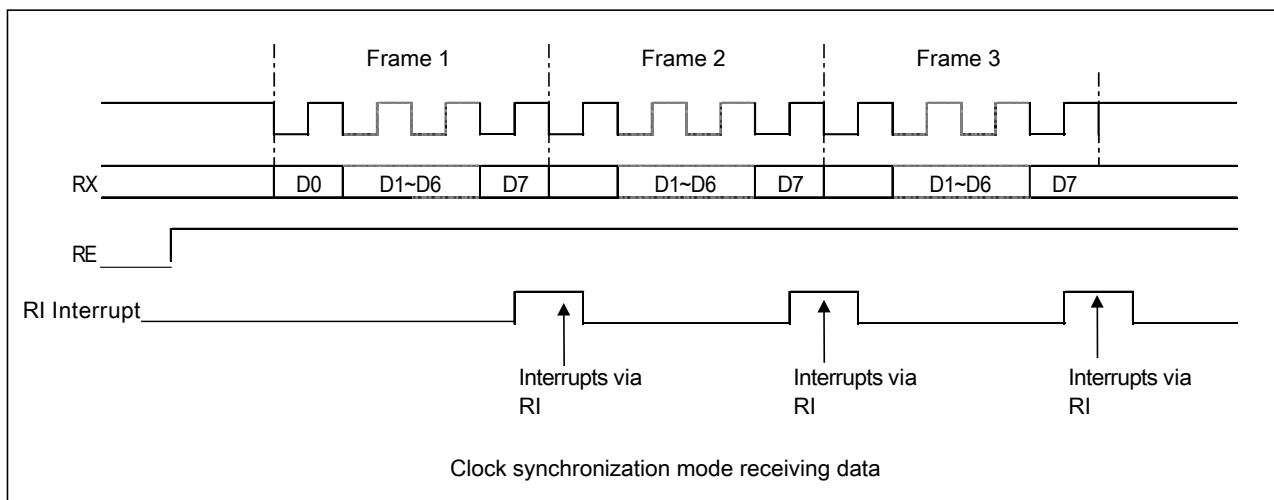
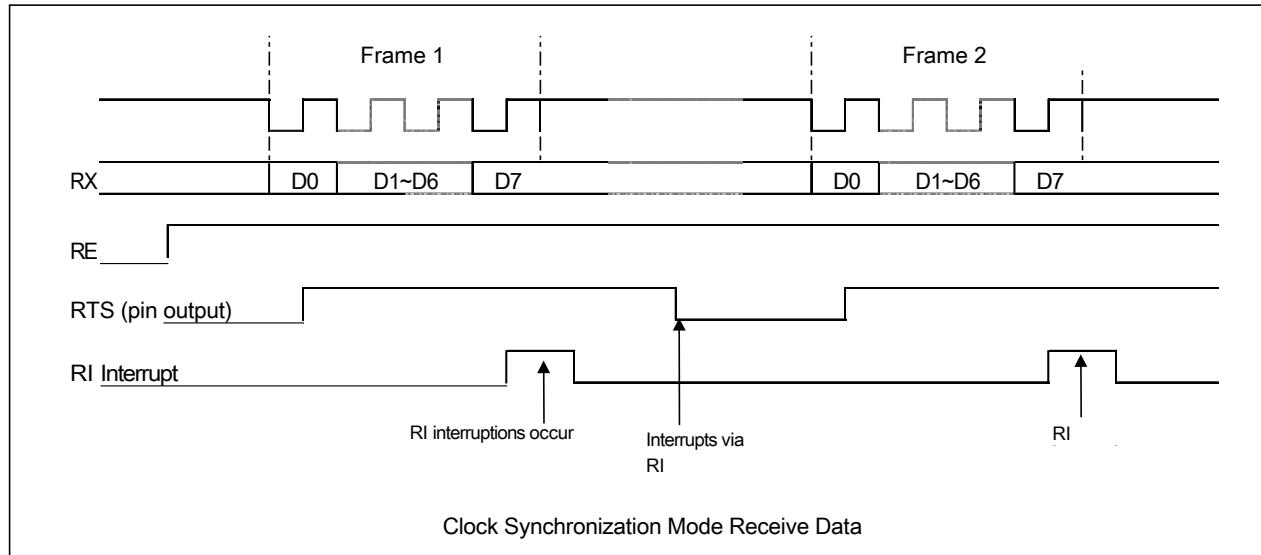


Figure 25-21 Clock Synchronization Mode Receive Data

Example 1

**Figure 25-22 Clock Synchronization Mode Received Data Example 2****Error****Handling****g**

The receive error is an overflow error (USARTn_SR.ORE) when data is received in clock synchronous mode. A receive error occurs and no more data can be received or sent. Data transmission can be restarted by clearing the error flag by writing the corresponding clear register.

RDR value is not read, so the data received in the previous frame should be read before the last bit of the current frame is received. The received data is lost when the overflow error occurs, and the

RI interrupt does not occur. Receiver interrupts

The clock synchronization mode receiver supports two interrupts, receive data

register full interrupt RI and receive error interrupt REI. RIE=1, RI interrupt

occurs when data is transferred from receive shift register to receive data register.

RIE=1, REI interrupt occurs when an error occurs in the received data (overflow error).

25.4.4.6 Simultaneous sending and receiving of data

USART clock synchronization mode supports full duplex operation to send and receive data at the same time. A command is required to write 1 to RE, TE, RIE, TXEIE when sending and receiving data at the same time, other setting process is the same as transmitter and receiver.

25.4.4.7 Clock synchronization mode interrupts and events**Table 25-7 Clock synchronization mode interrupt/event table**

Interrupt name	Marker	Enable bit (interrupt only)	Logo position	Can be used as an event source
----------------	--------	-----------------------------	---------------	--------------------------------

Receiving error interrupts	REI	RIE	ORE	No
Receive data register full interrupt	RI	RIE	RXNE	may
Send data register air break	TI	TXEIE	TXE	may
Send completion interrupt	TCI	TCIE	TC	No

25.4.5 Digital filtering function

When USARTn_CR1.NFE=1, the built-in digital filter function is effective. The digital filter is effective only in UART mode to remove the noise on the RX of the receive data line.

The built-in digital filter can filter out noise smaller than 3/16 (USARTn_CR1.OVER8=0) width of one bit of data or 3/8 width (USARTn_CR1._OVER8=1).

If the digital filter is started again after the clock is stopped, the digital filter continues to operate from the state it held when the clock was stopped. USARTn_CR.TE=0 and USARTn_CR.RE=0 resets the Flip-Flop state inside the digital filter to 1.

25.5 Register Description

USART1_BASE_ADDR:0x4001_D000

USART2_BASE_ADDR:0x4001_D400

USART3_BASE_ADDR:0x4002_1000

USART4_BASE_ADDR:0x4002_1400

Table 25-8 USART Register List

Register Name	Offset Address	Reset value
Status register (USART_SR)	0x00	0x0000_00C0
Data register (USART_DR)	0x04	0x0000_01FF
Baud Rate Register (USART_BRR)	0x08	0x0000_FFFF
Control register 1 (USART_CR1)	0x0C	0x8000_0000
Control register 2 (USART_CR2)	0x10	0x0000_0000
Control register 3 (USART_CR3)	0x14	0x0000_0000
Prescaler register (USART_PR)	0x18	0x0000_0000

25.5.1 Status register (USART_SR)

USART Status Register

Offset Address: 0x00

Reset value: 0x0000_00C0

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MPB
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	RTO			RXN		ORE	-	FE	PE

position	Marker	Place Name	Function	Reading and writing
b31~b17	Reserved	-	Read <code>^</code> , <code>write "0"</code> when writing	R
b16	MPB	Multiprocessor bits	Multiprocessor bit flag 0: Current received data is communication data 1: The current received data is ID Note: MPB bits are only valid in multi-processor mode	R
b15~b9	Reserved	-	Read <code>^</code> , <code>write "0"</code> when writing	R
b8	RTOF	UART receives the TIMEOUT flag bit	UART receive TIMEOUT flag bit 0: No UART receive TIMEOUT 1: UART receive TIMEOUT occurs RTOF setting conditions <ul style="list-style-type: none"> No new received data is detected after a set period of time from the STOP bit of the last data frame detected RTOF clear condition Clear register CR1.CRTOF bit write Note: RTOF is set to 1 by hardware and only when CR1.RE=1 and CR1.RTOE=1. When CR1.RE=0, TIMEOUT function is valid, but RTOF is not set to 1. RTOF is valid only when USART_CR1.RTOE=1, otherwise, please ignore this value. bit, when using the UART receive <code>timeout</code> function, this bit needs to be cleared by software before setting RTOE=1.	R
b7	TXE	Send data register empty	Send data register empty flag TXE bit is valid during UART and clock synchronization mode. 0: Data is not transferred to shift register, send data register is non-empty 1: Data transfer to shift register, send data register empty Note: TXE bit is set to 1 and cleared to 0 by hardware, hardware will clear TXE to 0 when data is not transferred to the shift register, hardware will set TXE to 1 when data is transferred to the shift register	R

position	Marker	Place Name	Function	Reading and writing
b6	TC	Send completion flag	<p>Send completion flag</p> <p>bit 0: sending data in progress</p> <p>1: Send data completion</p> <p>UART mode, clock synchronization mode TC setting condition</p> <ul style="list-style-type: none"> • TE=0 when sending forbidden • The value of the transmit data register is not updated when the last bit of a frame of data is sent TC clear condition • When TE=1, write send data to the send data register <p>Smart card mode TC setting condition</p> <ul style="list-style-type: none"> • TE=0 when sending forbidden • After a certain time has elapsed since the last 1 byte of data was sent, FE=0 and the value of the transmit data register has not been updated. <p>The specific timing of the TC set is: 2.5 bits of time elapsed after the check bit is sent TC clear condition</p> <ul style="list-style-type: none"> • When TE=1, write send data to the send data register <p>Note: The TE bit changes from 0 to 1 when TC is held to 1</p>	R
b5	RXNE	Receive data register is not empty	<p>Receive data register is not empty flag 0: No data received</p> <p>1: Prepare to read the received data</p> <p>Note: The RXNE bit is set to 1 and cleared to 0 by hardware, and the hardware sets the RXNE when it is ready to read the received data</p> <p>1, the hardware clears RXNE to 0 after reading the received data</p>	R
b4	Reserved	-	Read "1", write "0" when writing	R
b3	ORE	Receive overflow error	<p>Receive overflow error flag bit 0: No receive overflow error</p> <p>1: Receive overflow error occurred ORE set condition</p> <ul style="list-style-type: none"> • A new frame of data is received when the receive data register is not read ORE clear condition • Clear register CR1.CORE bit to write 1 Note: RE=0 does not reset the ORE bit <p>Data received before ORE=1 will be held, data received when ORE=1 will be dropped</p> <p>No further data reception after ORE=1, and no data transmission in clock synchronization mode</p>	R
b2	Reserved	-	Read "1", write "0" when writing	R

position	Marker	Place Name	Function	Reading and writing
b1	FE	Receive frame error	<p>Receive frame</p> <p>error flag bit 0: No receive frame</p> <p>error</p> <p>1: Receive frame error occurred</p> <p>UART mode</p> <p>FE placement conditions</p> <ul style="list-style-type: none"> The stop bit of the received data frame is low, and in the case of two stop bits only the first stop bit is checked FE clear condition Clear register CR1.CFE bit to write 1 <p>Note: RE=0 does not reset the FE bit when in UART mode</p> <p>Data received when FE=1 will be retained but RI interrupts will not occur, and data cannot be received after FE=1</p> <p>Smart card mode FE setting condition</p> <ul style="list-style-type: none"> Sample to low error signal flag FE clear condition Clear register CR1.CFE bit to write 1 <p>Note: In smart card mode, RE=0 does not reset the FE bit</p>	R
b0	PE	Receive data checksum	<p>Receive Data</p> <p>Checksum Error Flag 0: No receive data checksum error</p> <p>1: Receive data verification error occurs PE set condition</p> <ul style="list-style-type: none"> PE clear condition when parity error occurs in received data Clear register CR1.CPE bit write 1 Note: RE=0 does not reset the PE bit <p>The data received when PE=1 will be retained but the RI interrupt will not occur, and the received data cannot continue after PE=1.</p>	R

According to

25.5.2 Data register (USART_DR)

USART Data Register

Offset Address: 0x04

Reset value: 0x0000_01FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16							
-	-	-	-	-	-	-		RDR[8:0]														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0							
-	-	-	-	-	-	MPID		TDR[8:0]														

position	Marker	Place Name	Function	Reading and writing
b31~b25	Reserved	-	Read "", write "0" when writing	R
b24~b16	RDR[8:0]	Receive Data Register	Receive Data Register Note: The highest bit RDR[8] is valid only in UART mode and the data length is set to 9 bits	R
b15~b10	Reserved	-	Read "", write "0" when writing	R
b9	MPID	Multi-processor mode ID bits	Select bit for sending communication data or sending ID in multiprocessor mode 0: Send data 1: Send ID Note: MPID bit is only valid in multi-processor mode, other modes must be set to reset value	R/W
b8~b0	TDR[8:0]	Send data register	Send data register Note: The highest bit TDR[8] is valid only in UART mode and the data length is set to 9 bits	R/W

25.5.3 Baud Rate Register (USART_BRR)

USART Bit Rate

Register Offset

Address: 0x08

Reset value: 0x0000_FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DIV_Integer [7:0]										-	DIV_Fraction[6:0]				

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Read "0", write "0" when writing	R
b15~b8	DIV_Integer[7:0]	Integer divider register	Integer divider register Note: DIV_Integer[7:0] can only be set when TE=0&RE=0 (transmit/receive disabled)	R/W
b7	Reserved	-	Read "0", write "0" when writing	R
b6~b0	DIV_Fraction[6:0]	Fractional Frequency Division Register Frequency Division Register	Fractional Frequency Division Register Note: DIV_Fraction[6:0] can only be set when TE=0&RE=0 (transmit/receive disabled), and the value is valid only when FBME=1	R/W

Table 25-9 Baud rate calculation formula (fractional baud rate invalid FBME=0)

Mode	Baud rate calculation formula	Error E(%) calculation formula
UART mode Multi-processor mode	$B = \frac{C}{8 \times (2 - \square\square\square\square) \times (\text{DIV_Integer} + 1)}$	$\square(\%) = \frac{C}{8 \times (2 - \square\square\square\square) \times (\text{DIV_Integer} + 1)} \times \frac{100}{B}$
Clock synchronization mode	$B = \frac{C}{4 \times (\text{DIV_Integer} + 1)}$	-
Smart Card Mode	$B = \frac{C}{2 \times \text{BCN} \times (\text{DIV_Integer} + 1)}$	$\square(\%) = \frac{C}{2 \times \text{BCN} \times (\text{DIV_Integer} + 1)} \times \frac{100}{B}$

B: Baud rate Unit: Mbps C: Clock set by

PR.PSC[1:0] bits Unit: MHz BCN:

CR3.BCN register setting value

Table 25-10 Baud rate calculation formula (fractional baud rate valid FBME=1)

Mode	Baud rate calculation formula	Error E(%) calculation formula
------	-------------------------------	--------------------------------

UART mode Multi-processor mode	$B = \frac{C \times (128 + DIV_Fraction)}{8 \times (2 - \square \square \square \square 8) \times (DIV_Integer + 1) \times 256}$	$\square(\%) = \frac{C \times (128 + DIV_Fraction)}{8 \times (2 - \square \square \square \square 8) \times (DIV_Integer + 1) \times 256 \times B} \times 100$
Clock synchronization mode	$B = \frac{C \times (128 + DIV_Fraction)}{4 \times (DIV_Integer + 1) \times 256}$	
Smart Card Mode	$B = \frac{C \times (128 + DIV_Fraction)}{2 \times BCN \times (DIV_Integer + 1) \times 256}$	$\square(\%) = \frac{C \times (128 + DIV_Fraction)}{2 \times BCN \times (DIV_Integer + 1) \times 256 \times B} \times 100$

B: Baud rate Unit: Mbps C: Clock set by

PR.PSC[1:0] bits Unit: MHz BCN:

CR3.BCN register setting value

25.5.4 Control register 1 (USART_CR1)

USART Control

Register 1 Offset Address:

0x0C

Reset value: 0x8000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
SBS	NFE	FBME	ML	-	-	-	MS	-	-	-	CRTO F	COR E	-	CFE	CPE
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
OVER 8	-	-	M	-	PCE	PS	-	TXE IE	TCI E	RIE	SLME	TE	RE	RTO IE	RTO E

position	Marker	Place Name	Function	Reading and writing
b31	SBS	UART mode	When receiving data in UART mode, the start bit detection mode is set to 0: the start bit	R/W
		receive data start	detection mode is low on the RX pin	
		bit detection	1: Start bit detection method for the falling edge of the RX pin	
		method set	Note: The SBS bit must hold the reset value when in non-UART mode	
		positioning	SBS bit can only be set when TE=0 & RE=0 (transmit/receive disable)	
b30	NFE	Digital filter enable	Digital filter enable bit	R/W
		bit	0: Disable digital filter	
			function	
			1: Enabling data filtering function	
			Note: The NFE bit must hold the reset value for non-UART modes	
			NFE bit can only be set when TE=0 & RE=0 (transmit/receive disable)	
b29	FBME	Fractional baud	Fractional Baud Rate	R/W
		rate function	Function Enable Bit 0:	
		enable	Disable	
			1: Enabling	
			Note: FBME bit can only be set when TE=0 & RE=0 (transmit/receive disable)	
b28	ML	MSB/LSB	MSB/LSB mode selection bit in UART mode/clock synchronization	R/W
		selection bit	mode/smart card mode 0: LSB mode	
			1: MSB method	
			Note: ML bit can only be set when TE=0 & RE=0 (transmit/receive disable)	
b27~b25	Reserved	-	Read "0", write "0" when writing	R
b24	MS	Communication	Communication	R/W
		mode selection bit	mode selection	
			bit 0: UART	
			mode	
			1: Clock synchronization mode	
			Note: MS bit can only be set when TE=0 & RE=0 (transmit/receive disable), smart card mode MS needs to write reset value	
b23~b21	Reserved	-	Read "0", write "0" when writing	R

position	Marker	Place Name	Function	Reading and writing
b20	CRTOF	RTOF clear bit	RTOF clear bit 0:Do not clear the RTOF flag 1:Clear the RTOF flag Note: CRTOF bit write 1 clears the RTOF flag and returns 0 when read	R/W
b19	CORE	ORE flag clear bit	ORE flag clear bit 0: Do not clear the ORE flag 1: Clear the ORE logo Note: The CORE bit clears the ORE flag by writing 1, and returns 0 when read	R/W
b18	Reserved	-	Read <code>"/"</code> , <code>write "0"</code> when writing	R
b17	CFE	FE flag clear bit	FE flag clear bit 0: Do not clear the FE flag 1:Clear the FE flag Note: The CFE bit clears the FE flag by writing 1 and returns 0 when read	R/W
b16	CPE	PE flag clear bit	PE flag clear bit 0:Do not clear the PE flag 1:Clear the PE logo Note: CPE bit write 1 clears the PE flag and returns 0 when read	R/W
b15	OVER8	UART oversampling mode setting, i.e. the number of base clocks during one bit data transfer	UART oversampling mode setting, i.e. the number of base clocks during one bit data transfer 0: 16 bits 1: 8 bits Note: The OVER8 bit must hold the reset value when in non-UART mode The OVER8 bit can only be set when TE=0 & RE=0 (transmit/receive disable)	R/W
b14~b13	Reserved	-	Read <code>"/"</code> , <code>write "0"</code> when writing	R/W
b12	M	Data length set position	In UART mode, the transmit/receive data length is set to 0: 8 bits 1: 9 bits Note: The M bit must hold the reset value when in non-UART mode M bit can only be set when TE=0 & RE=0 (transmit/receive disable)	R/W
b11	Reserved	-	Read <code>"/"</code> , <code>write "0"</code> when writing	R
b10	PCE	Checksum enable bit	Parity enable bit 0: no parity when in UART mode 1: Calibration Note: The PCE bit must be 1 in smart card mode, and the PCE bit must hold the reset value in clock synchronization mode The PCE bit can only be set when TE=0 & RE=0 (transmit/receive disabled)	R/W
b9	PS	Check digit	In UART mode, parity selection bit 0: even parity 1: Odd calibration Note: PS bit can only be set when TE=0 & RE=0 (transmit/receive disable), PS bit is only valid when PCE=1	R/W
b8	Reserved	-	Read <code>"/"</code> , <code>write "0"</code> when writing	R

position	Marker	Place Name	Function	Reading and writing
b7	TXEIE	Send data register air break enable bit 0: request is invalid, TI interrupt does not occur 1: TI interrupt request is valid, TI interrupt occurs	Send data register air break enable bit 0: TI interrupt request is invalid, TI interrupt does not occur not occur 1: TI interrupt request is valid, TI interrupt occurs	R/W
		Note: TXEIE bit and TE bit should be written to 1 at the same time		
b6	TCIE	Send completion interrupt enable bit	Request enable bit in send completion 0: TCI interrupt request is invalid, TCI interrupt does not occur 1: TCI interrupt request is enabled, TCI interrupt occurs	R/W
b5	RIE	Receive interrupt enable bit	Receive interrupt enable bit 0: Receive interrupt request is invalid, RI and REI interrupts do not occur 1: Receive interrupt request is valid, RI and REI interrupts occur	R/W
b4	SLME	Silent mode enable bit	Silent mode enable bit 0 for multiprocessor operation: normal mode 1: Silent mode When SLME=1, the communication data with MPB bit 0 is not read from the receive shift register to the receive data register, and the error flag ORE and FE bits are not set. When ID data with MPB of 1 is received, SLME is automatically cleared and normal data reception action starts. Note: The SLME bit is only valid in UART multiprocessor mode, this bit must remain reset in other modes Value.	R/W
b3	TE	Transmitter enable bit	Transmitter enable bit 0: Transmitter disable 1: Transmitter enable Note: The TE bit can only be written 1 when TE=0 & RE=0 (transmit/receive disable) during clock synchronization mode.	R/W
b2	RE	Receiver enable bit	Receiver enable bit 0: Receiver disable 1: Receiver enable Note: The RE bit in clock synchronization mode can only be written 1 when TE=0 & RE=0 (transmit/receive disabled)	R/W
b1	RTOIE	UART TIMEOUT interrupt enable bit	UART TIMEOUT interrupt enable bit 0: UART TIMEOUT interrupt request is invalid, RTOI interrupt does not occur 1: UART TIMEOUT interrupt request is valid, RTOI interrupt occurs Note: RTOIE needs to be set to 1 at the same time as RTOE, or set to 1 after RTOE=1	R/W
b0	RTOE	UART TIMEOUT function enable bit	UART TIMEOUT function enable bit 0: UART TIMEOUT function is disabled 1: UART TIMEOUT function is enabled Note: Before setting RTOE=1 to 1, you need to clear the USARTn.SR.RTOF flag and stop the TimerO counting function of the	R/W

corresponding channel

25.5.5 Control register 2 (USART_CR2)

USART Control

Register 2 Offset Address:

0x10

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	STOP	CLKC[1:0]	-	-	-	-	-	-	-	-	-	-	-	MPE

position	Marker	Place Name	Function	Reading and writing
b31~b14	Reserved	-	Read "0", write "0" when writing	R
b13	STOP	Stop bit set positioning	In UART mode, the stop bit length is set to 0:1 stop bit 1:2 stop bits Note: The STOP bit must hold the reset value when in non-UART mode STOP bit can only be set when TE=0 & RE=0 (transmit/receive disable)	R/W
b12~b11	CLKC[1:0]	Clock control bits	UART mode 00b: The clock source is the clock generated by the internal baud rate generator, the clock is not output to the USARTTn_CK pin, the USARTtN_CK pin can be used as ordinary IO 01b: The clock source is the clock generated by the internal baud rate generator, the clock output to USARTtN_CK pin, the output clock frequency and baud rate is the same 10b or 11b: The clock source is an external input clock with a frequency of 16 times the baud rate (OVER8=0) or 8 times the baud rate (OVER8=1) Clock synchronization mode 00b or 01b: The clock source is the clock generated by the internal baud rate generator and output to the USARTtN_CK pin 10b or 11b: The clock source is an external input clock with the same frequency and baud rate as the input clock Smart Card Mode 00b: The clock source is the clock generated by the internal baud rate generator, the clock is not output to the CK pin, the CK pin can be used as ordinary IO 01b: Clock source is the clock generated by internal baud rate generator, clock output to CK pin 10b or 11b: set disable Note: CLKC[1:0] bits can only be set when TE=0 & RE=0 (transmit/receive disable)	R/W
b10~b1	Reserved	-	Read "0", write "0" when writing	R

b0	MPE	Multi-processor function enable bit	Multiprocessor function enable bit 0: Disable when in UART mode 1: Enabling Note: The MPE bit must hold the reset value when in non-UART mode MP bit can only be set when TE=0 & RE=0 (transmit/receive disable)	R/W
----	-----	-------------------------------------	--	-----

25.5.6 Control register 3 (USART_CR3)

USART Control

Register 3 Offset Address:

0x14

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	BCN[2:0]			-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b15	b14
-	-	-	-	-	-	CTS E	-	-	-	SCE N	-	-	-	-	-

position	Marker	Place Name	Function	Reading and writing
b31~b24	Reserved	-	Read "0", write "0" when writing	R
b23~b21	BCN[2:0]	Basic clock number	When in smart card mode, set the base clock number during one-bit data transmission BCN[2:0] setting value Number of base clocks during one-bit data transmission 000b 32 001b 64 010b Set Prohibit 011b 128 100b Set Prohibit 101b 256 110b 372 111b Set Prohibit Note: BCN[2:0] bits must remain reset when not in smart card mode BCN[2:0] bits can only be set when TE=0 & RE=0 (transmit/receive disabled)	R/W
b20~b10	Reserved	-	Read "0", write "0" when writing	R
b9	CTSE	CTS function enable bit	CTS function enable bit 0: RTS function 1: CTS function Note: CTSE bit can only be set when TE=0 & RE=0 (transmit/receive disable)	R/W
b8~b6	Reserved	-	Read "0", write "0" when writing	R
b5	SCEN	Smart card mode enable bit	Smart card mode enable bit 0: Disable smart card mode 1: Enabling smart card mode Note: The SCEN bit must hold the reset value when not in smart card mode SCEN bit can only be set when TE=0 & RE=0 (transmit/receive disable)	R/W
b4~b0	Reserved	-	Read "0", write "0" when writing	R

25.5.7 Prescaler register (USART_PR)

USART prescaler register

offset address: 0x18

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PSC[1:0]

position	Marker	Place Name	Function	Reading and writing
b31~b2	Reserved	-	Read "0", write "0" when writing	R
b1~b0	PSC[1:0]	Prescaler value	When internal clock source, prescaler divider value selection bit 00: PCLK 01: PCLK/4 10: PCLK/16 11: PCLK/64 Note: The PSC[1:0] bits can only be set when TE=0 & RE=0 (transmit/receive disable)	R/W

25.6 Precautions for use

25.6.1 UART Caution

Transmitter

When the UART mode transmitter transmits forbidden (**USARTn_CR1.TE=0**), the TX pin can be used as a normal IO and the output value and direction can be set. If 0 is output, it causes the receiver to generate a frame error, which interrupts the data transmission. If 1 is output, the receiver does not detect the start bit and thus cannot start data transmission.

Receiver

When the UART mode generates a frame error, the software can detect if the subsequent RX line is low and thus determine if the sender wants to interrupt the transmission.

If the receive data start bit detection method is low level detection, continue to receive all low level data after clearing the error flag and the receive error will occur again.

25.6.2 Notes on Clock Synchronization Mode

- 1) When sending data using an external input clock, the update of **USARTn_DR** - TDR needs to be completed before the clock is input, and after writing data, it needs to wait at least one bit of data time before inputting the clock.
- 2) When sending data continuously, the next frame of data needs to be updated before the last bit of the current frame is sent.

25.6.3 Other Notes

- 1) To prevent a Hi-Z state of the TX communication line when transmit is disabled, the following method can be used:
 - Communication line pull up
 - Set the TX pin to normal IO output before **USARTn_CR1.TE=0** at the end of sending data
 - Set IO to TX function after **USARTn_CR1.TE=1** before the start of sending data

26 Integrated Circuit Bus (I2C)

26.1 Introduction

I2C (Integrated Circuit Bus) is used as an interface between the microcontroller and the I2C serial bus. Provides multi-master mode function to control all I2C bus protocols, arbitration. Standard mode, fast mode are supported. SMBus bus is also supported.

I2C main features.

- 1) I2C bus mode and SMBUS bus mode are selectable. Master mode, slave mode selectable.
Automatically ensures various ready times, hold times and bus idle times corresponding to the transfer rate.
- 2) Standard mode up to 100Kbps, fast mode up to 400Kbps.
- 3) Automatic generation of start conditions, restart conditions and stop conditions with the ability to detect bus start conditions, restart conditions and stop conditions
- 4) 2 slave mode addresses can be set. 7-bit address format and 10-bit address format can be set at the same time. Broadcast call address, SMBus host address, SMBus device default address, and SMBus alarm address can be detected.
- 5) The answer bit can be determined automatically when sending. The answer bit can be sent automatically when receiving.
- 6) Handshake function.
- 7) Arbitration function.
- 8) A timeout function that detects when the SCL clock has been stopped for a long time.
- 9) SCL input and SDA input have built-in digital filters with programmable filtering capability.
- 10) Communication error, receive data full, send data empty, end of one frame send, address match consistent interrupt.

26.2 I2C System Block Diagram

26.2.1 System Block Diagram

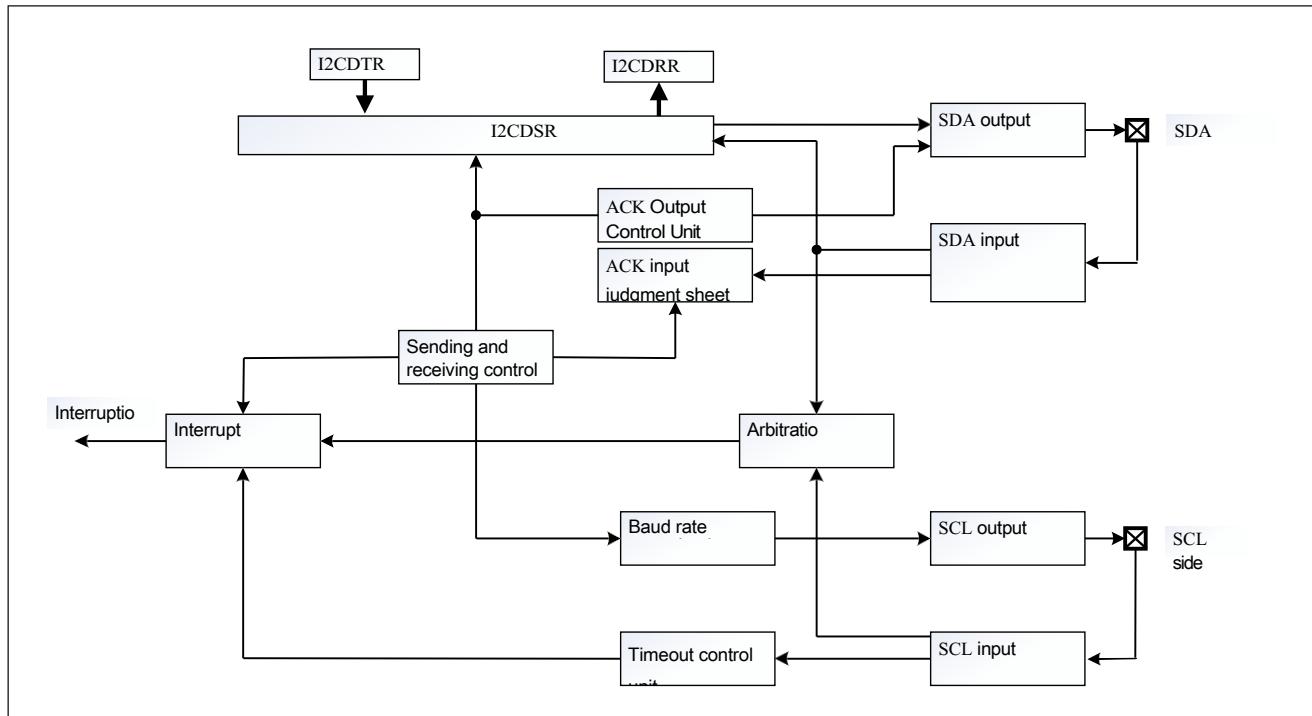


Figure 26-1 I2C System Block Diagram



26.2.2 Structure Diagram

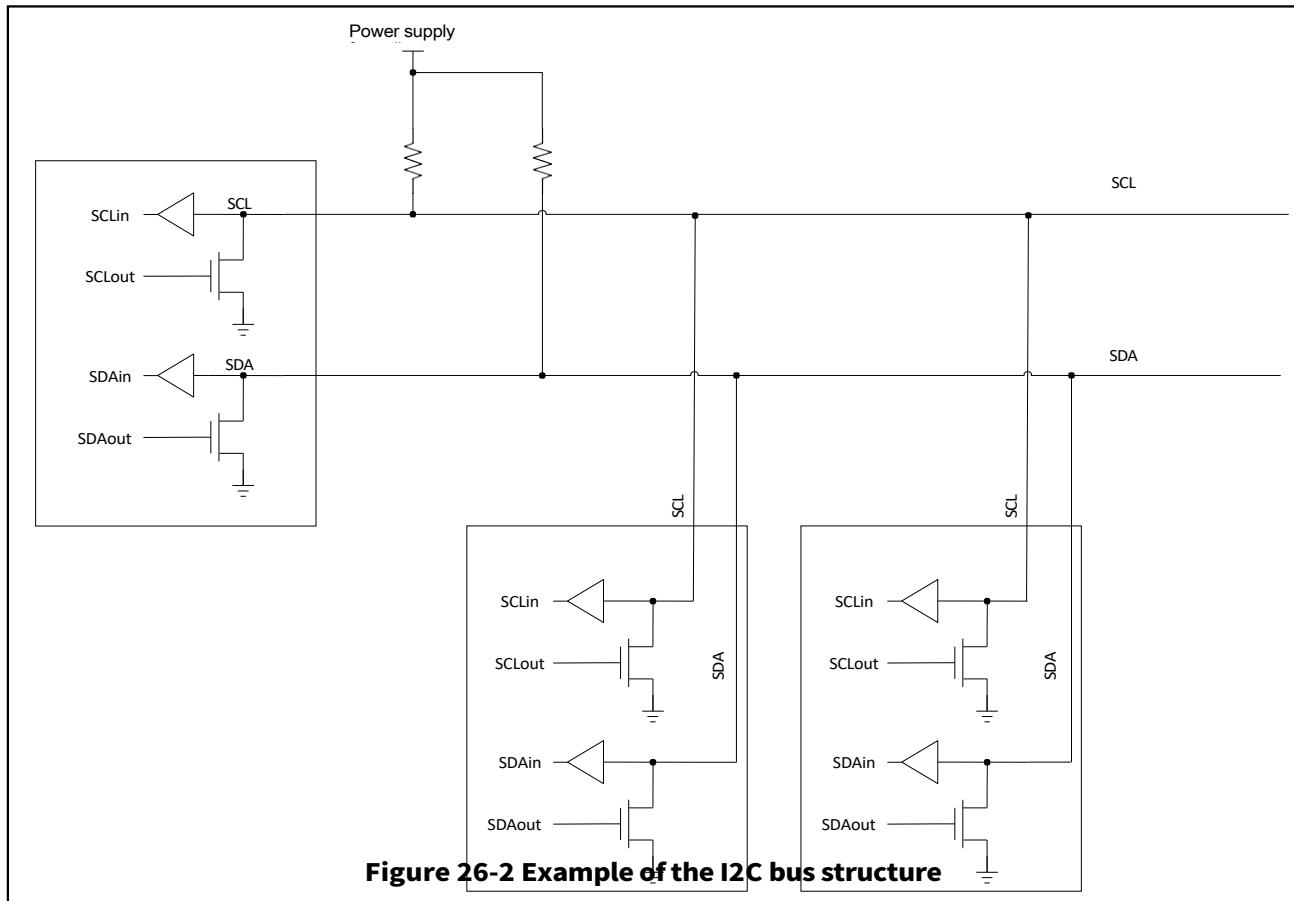


Table 26-1 Input/Output Pins

Pin Name	Input/output	Function
SCL	Input/output	Serial clock input/output pins
SDA	Input/output	Serial clock input/output pins

When I2C bus is selected, the SCL/SDA input level is CMOS level. When SMBus is selected, the SCL/SDA input level is a TTL level.

26.3 Action Description

This section provides a description of the I2C module functions.

26.3.1 I2C Protocol

The I2C bus consists of a clock line (SCL) and data line (SDA). All connected devices must be open-drain outputs, and the SCL and SDA lines have external pull-up resistors. The resistor value depends on the system application.

Typically, a complete communication process consists of the following 4 parts:

1. Start conditions
2. Address Delivery
3. Data Transfer
4. Stop conditions

The following figure shows the timing diagram of the I2C bus.

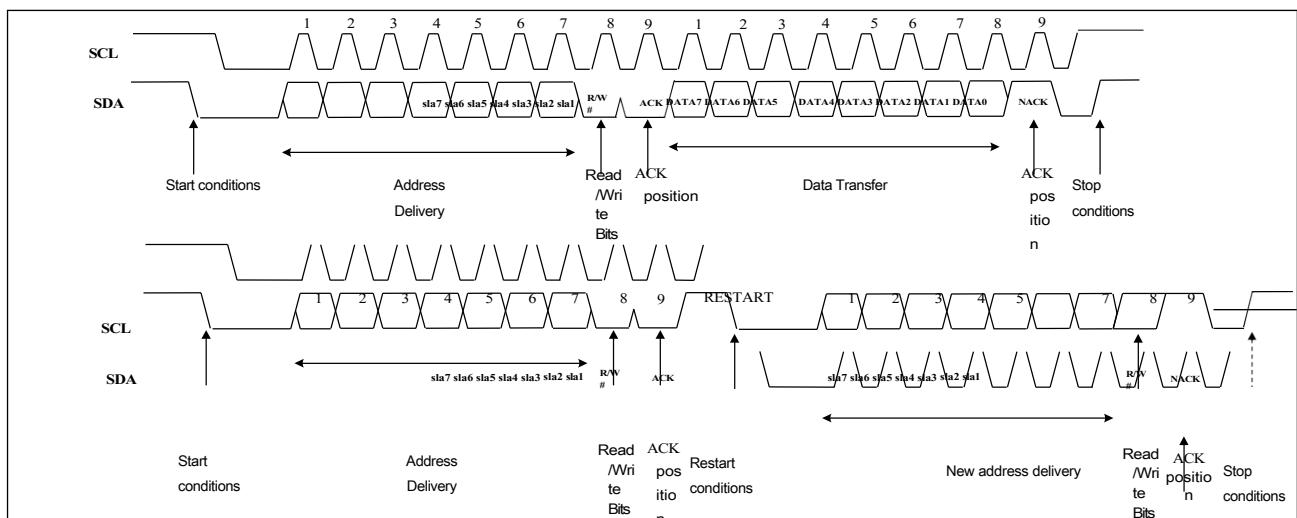


Figure 26-3 Timing diagram of the I2C bus

26.3.1.1 Start condition

When none of the hosts on the bus are driving the bus, the bus enters the idle state, and both SCL and SDA are high. When the bus is idle, all devices on the bus can start communication by sending a start condition.

If the START position is "1" in the state of I2C_SR.BUSY flag "0"(bus idle), the start condition is issued. If a start condition is detected, the I2C_SR.BUSY flag and I2C_SR.STARTF flag are automatically set to "1". The START position is automatically set to "0". In this case, if the SDA signal sent with START bit "1" has the same signal state as the SDA line, and the start condition is

detected, the start condition is considered to be issued correctly by the START bit, and the I2C_SR.MSL bit and I2C_SR.TRA bit are automatically set to "1". TRA bits to "1" automatically and then change to master transmit mode. In addition, I2C_SR.TEMPTYF is automatically changed to "1" because the TRA bit is "1". Next, write the slave address to the I2C_DTR register and send the address.

26.3.1.2 Address Delivery

The frame following the start condition or restart condition is an address frame that specifies the address of the object to which the master is communicating. The specified slave remains in effect until the stop condition is sent.

The high 7 bits of the address frame are the slave address. Bit 8 of the address frame determines the direction of the data frame transmission.

- 1) The 7-bit addressing mode is shown in the following figure [7-bit address format]

Host transmit mode, host transmit address frame bit 8 is 0

Host receive mode, the host sends the address frame with bit 8 as 1

- 2) The 10-bit addressing mode is shown in the following figure [10-bit address format]

In host transmit mode, the host sends the header sequence (11110XX0, where XX indicates the high two bits of the 10-bit address) and then sends the low eight bits of the slave address in the second frame.

In host accept mode, the host sends a header sequence (11110XX0, where XX indicates the high two bits of the 10-bit address) and then sends the low eight bits of the slave address for the second frame. Next, a restart condition is sent, followed by another frame of the header sequence

(11110XX1, where XX indicates the high two bits of the 10-bit address)

7-bit address format

S	SLA (7 digits)	R/W #	ACK/NACK	DATA(8 bits)	AC K		DATA(8 bits)	ACK/NACK	P
---	----------------	-------	----------	--------------	------	--	--------------	----------	---

10-bit address format

S	11110b+SLA (2 places)	W#	AC K	SLA (8-bit)	AC K	DATA(8 bits)	AC K		DATA(8 bits)	ACK/NACK	P
---	-----------------------	----	------	-------------	------	--------------	------	--	--------------	----------	---

S	11110b+SLA (2 places)	W#	AC K	SLA (8-bit)	AC K	Sr	11110b+SLA (2 places)	R	AC K	DATA(8 bits)	AC K		DATA(8 bits)	ACK/NACK	P
---	-----------------------	----	------	-------------	------	----	-----------------------	---	------	--------------	------	--	--------------	----------	---

Figure 26-4 Data Format of I2C Bus

S : Indicates the

start condition. SLA :

Indicates the slave

address.

R/W# : Indicates the direction of sending and receiving. When R/W# is "1", the data is sent

from the host to the master; when R/W# is "0", the data is sent from the master to the slave.

Sr : Indicates the restart

condition. DATA : Indicates the

sent and received data P :

Indicates the stop condition.

26.3.1.3 Data Transfer

After the address match is agreed, the hosts on the bus transmit data frame by frame according to the direction defined by R/W.

All data transmitted after an address frame is considered a data frame. Even the lower 8 bits of a 10-bit address format are considered data frames.

The length of the data frame is 8 bits, the low SDA of SCL changes, the high SDA of SCL holds, and one bit of data is sent per clock cycle. The 9th clock after the data frame is the answer bit, which is the handshake signal transmitted by the receiver to the sender.

If the slave on the bus receives data and does not respond to the host on the 9th clock cycle, the slave must send a NACK. If the host on the bus receives data and sends a NACK on the 9th cycle, the slave receives the NACK and the slave stops sending data.

The data transfer is terminated whether the host or the slave sends a NACK. The host can do either of the following:

- 1) Send a stop condition to release the bus
- 2) Send a restart condition to start a new communication.

Host sending data

In host transmit mode, the host outputs the SCL clock and sends data, and the slave receives the data and returns an answer. An example of the host send data runtime is shown in the following figure.

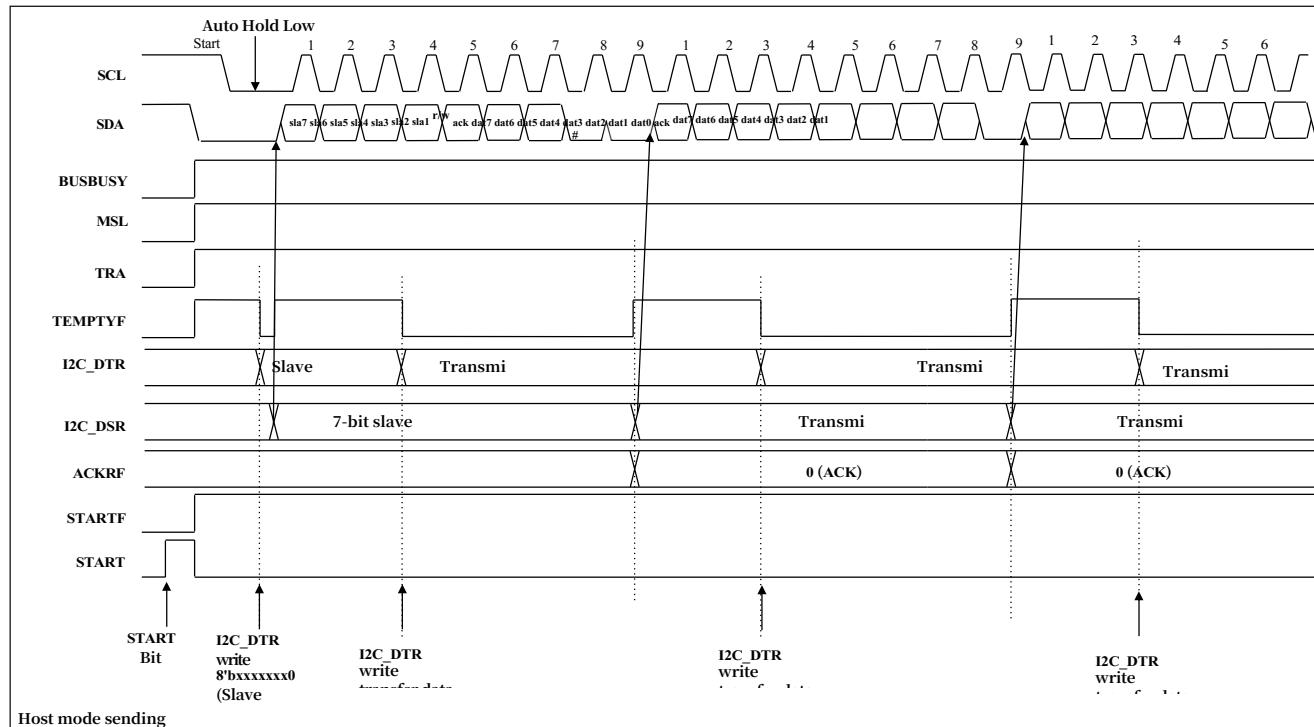


Figure 26-5 Timing diagram for sending data from the host in 7-bit address format (example)

Host receiving data

In the host receive mode, the host outputs the SCL clock, receives the slave data and returns an answer. An example of the host receive data runtime is shown in the following figure.

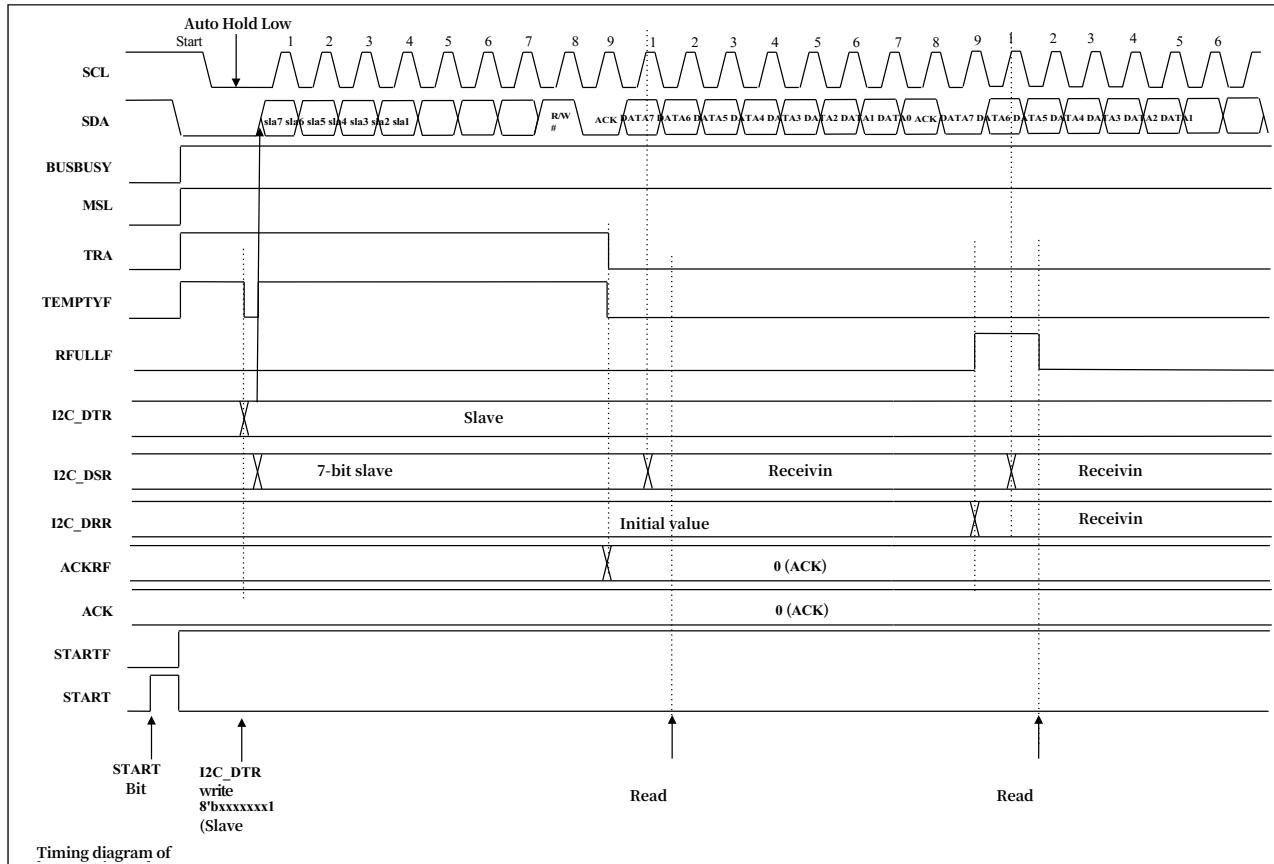


Figure 26-6 Timing diagram for receiving data from a host with 7-bit address format (example)

Slave sends data

In the slave transmit mode, the SCL clock from the host is received, the product sends data for the slave, and the host returns an answer. The operation timing of the slave sending data is shown in the following figure.

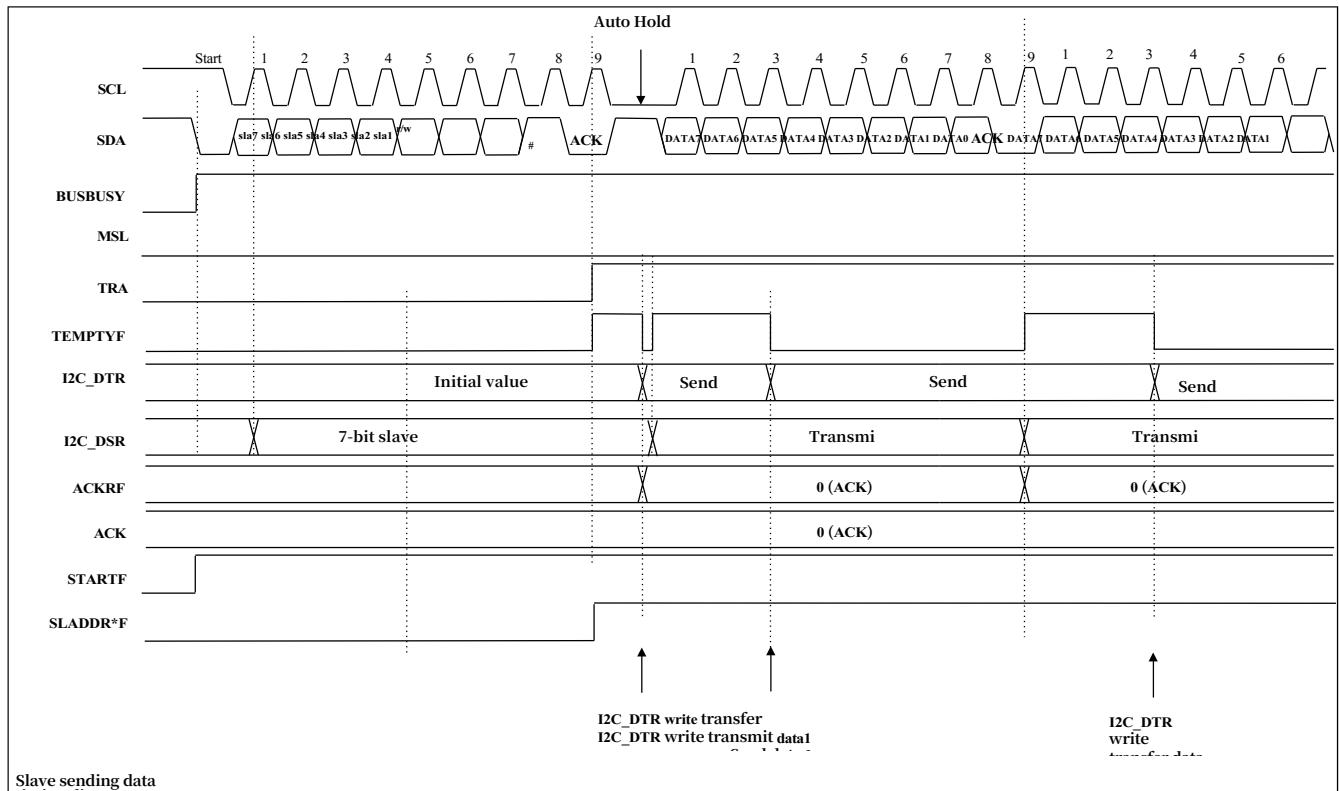


Figure 26-7 Slave transmit mode timing diagram for 7-bit address format (example)

Slave receiving data

In the slave receive mode, the SCL clock and data from the host are received, and an answer is returned after the data is received. An example of the slave receive data operation timing is shown in the following figure.

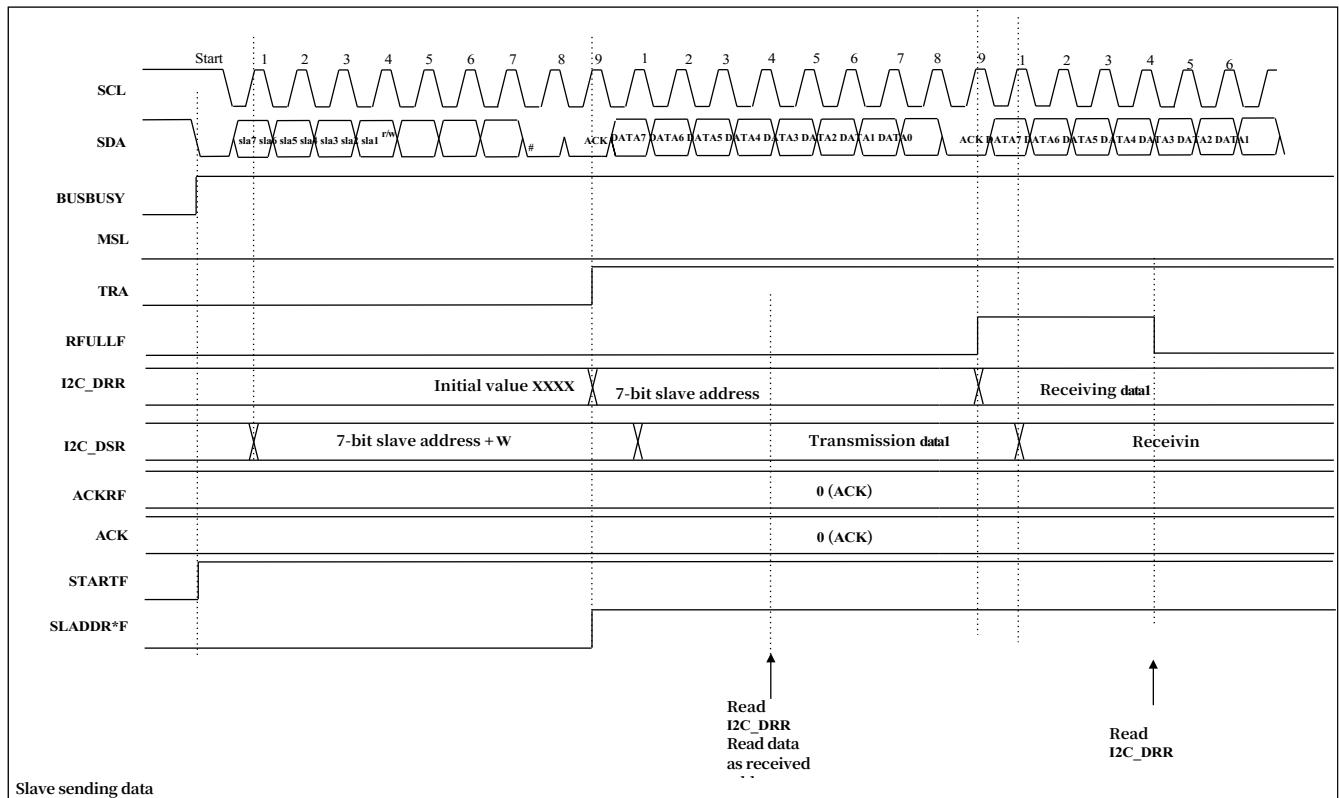


Figure 26-8 7-bit address format slave receive mode timing diagram (example)

26.3.1.4 Stop condition

Issue a stop condition via the I2C_CR1.STOP bit.

BUSY flag is "1" (bus busy) and the I2C_SR.MSL bit is "1" (host mode), set the STOP position "1", the stop condition is issued.

26.3.1.5 Restart conditions

The restart condition is generated via the I2C_CR1.RSTART bit.

BUSY flag is "1" (bus busy) and the I2C_SR.MSL bit is "1" (host mode), RSTART position "1", the line restart condition is generated.

The restart condition allows the host to switch between transmit/receive modes without releasing the BUS rights. It is also possible to establish communication with another slave without releasing the BUS right.

26.3.1.6 SCL clock synchronization

When using the I2C bus in multi-host mode, it is possible that the SCL clock and the SCL clock may conflict due to competition with other hosts. If the SCL clock is in conflict, the host needs to synchronize with the SCL clock and synchronize the SCL clock bit by bit. If the SCL line drops due to the SCL clock output of another host while the rising edge of the SCL line is detected and the high level set by the I2C_CCR.SHIGHW register is being counted, the incremental counting of the high level width is aborted when the falling edge of the SCL line is detected and the counting of the low level width set by the I2C_CCR.SLOWWW is started while the SCL line is driven low. SLOWWW sets the incremental count of the low level width, ends the low level drive of the SCL line when the count of the low level width ends, and releases the SCL line. At this time, if the low level width of the SCL clock of other hosts is greater than the low level width set by SLOWWW, the low level width of the SCL clock is extended. When the other host ends the low output, the SCL line is released and the SCL clock rises. Therefore, in the event of a SCL clock output conflict, the high level width of the SCL clock is synchronized with the short clock and the low level width is synchronized with the long clock.

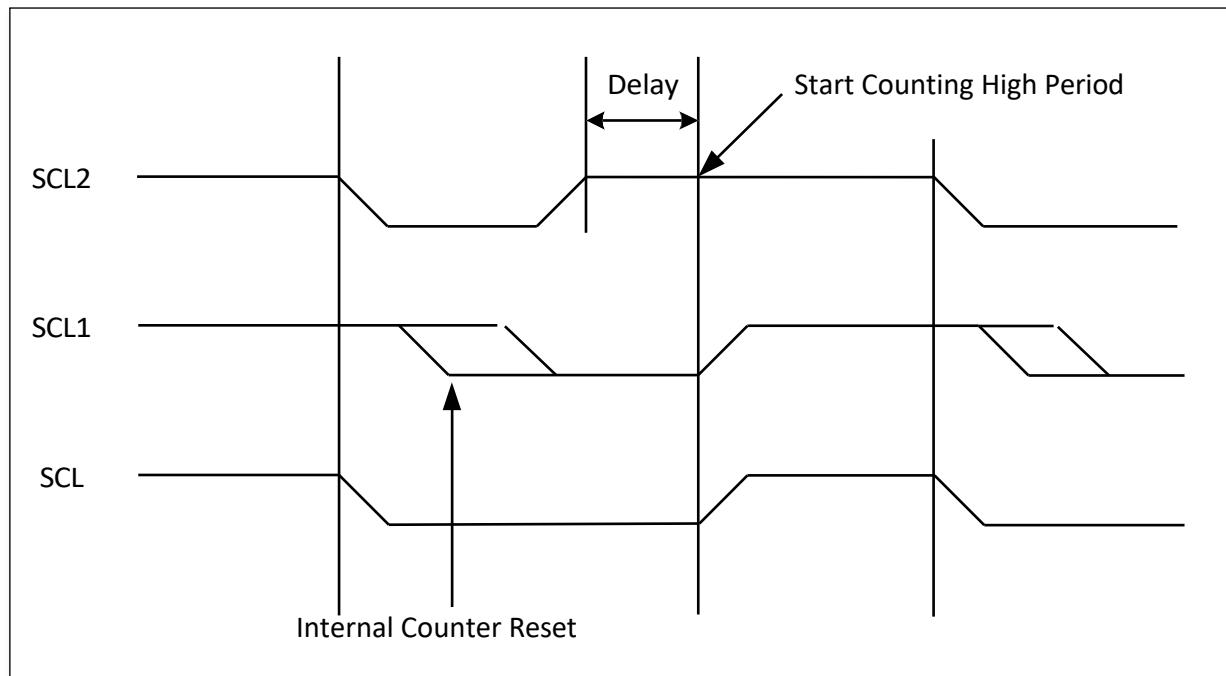


Figure 26-9 SCL Synchronization Timing

26.3.1.7 Arbitration on

The I2C bus is a true multi-host bus, allowing multiple hosts to be connected.

If two or more hosts try to control the bus at the same time, the SCL clock synchronization process determines the bus clock. The low period of the bus clock depends on the longest low level clock and the high period depends on the shortest high level clock. The data collected at high level determines the

arbitration result. Arbitration failure occurs when the transmitted SDA output is high (SDA pin is in high impedance state) and the SDA line is detected as low. If a host arbitration failure occurs, it immediately shifts to slave receive mode. At this point, if the slave address, including the broadcast address, matches, operation in slave mode continues.

26.3.1.8 Handshake

Handshaking during data transfer is achieved through the SCL clock synchronization mechanism. The slave holds the SCL clock line low after transmitting a frame of data (containing ACK bits) in this case, the low level of the SCL clock puts the host into a wait state until the slave releases the SCL line.

[Slave transmit mode]

- 1) In transmit mode (I2C_SR.TRA bit = 1) if the shift register (I2C_DSR register) is empty and the transmit data (I2CDTR register) is not written, the SCL line is automatically held low between the 9th clock and the low level of the first clock of the next transmission.

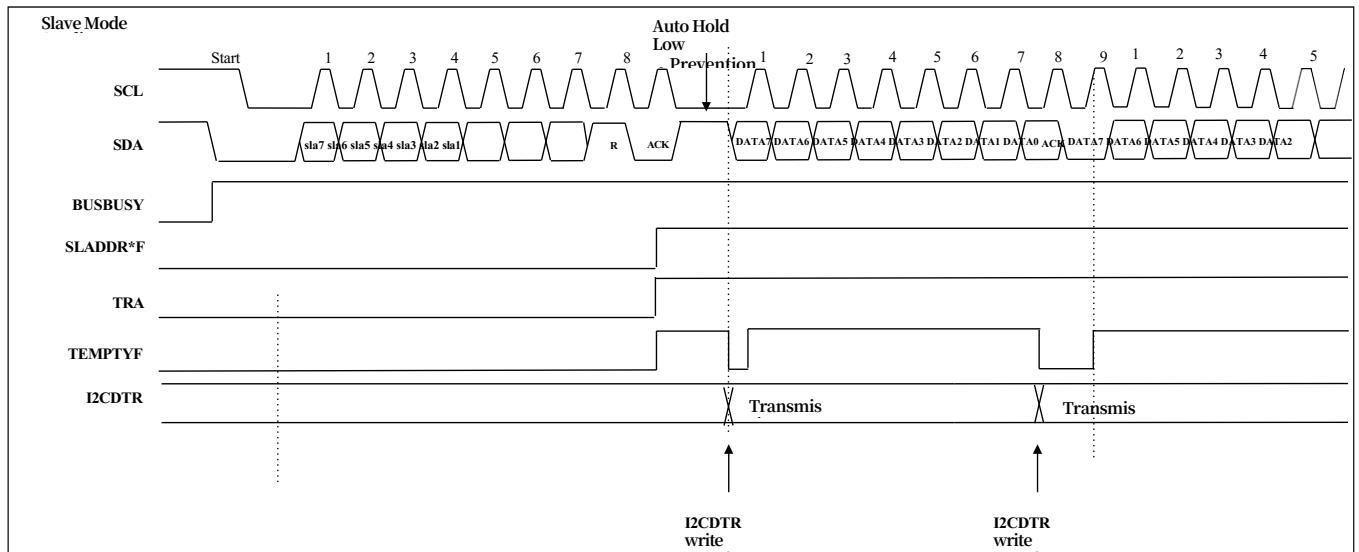


Figure 26-10 Slave transmit timing diagram (1)

- 2) After the I2C_SR.NACKF flag changes to "1" or the last transmit data is written to the I2C_DTR register, wait until the I2C_SR.TEMPTYF flag is "1" until the I2C_SR.TENDF flag changes to "1". When the IC_SR_NACKF flag or TENDF flag is "1", the SCL line is held low after the 9th clock drop. In this case, the communication must be terminated by reading the I2C_DR register to release the SCL line.

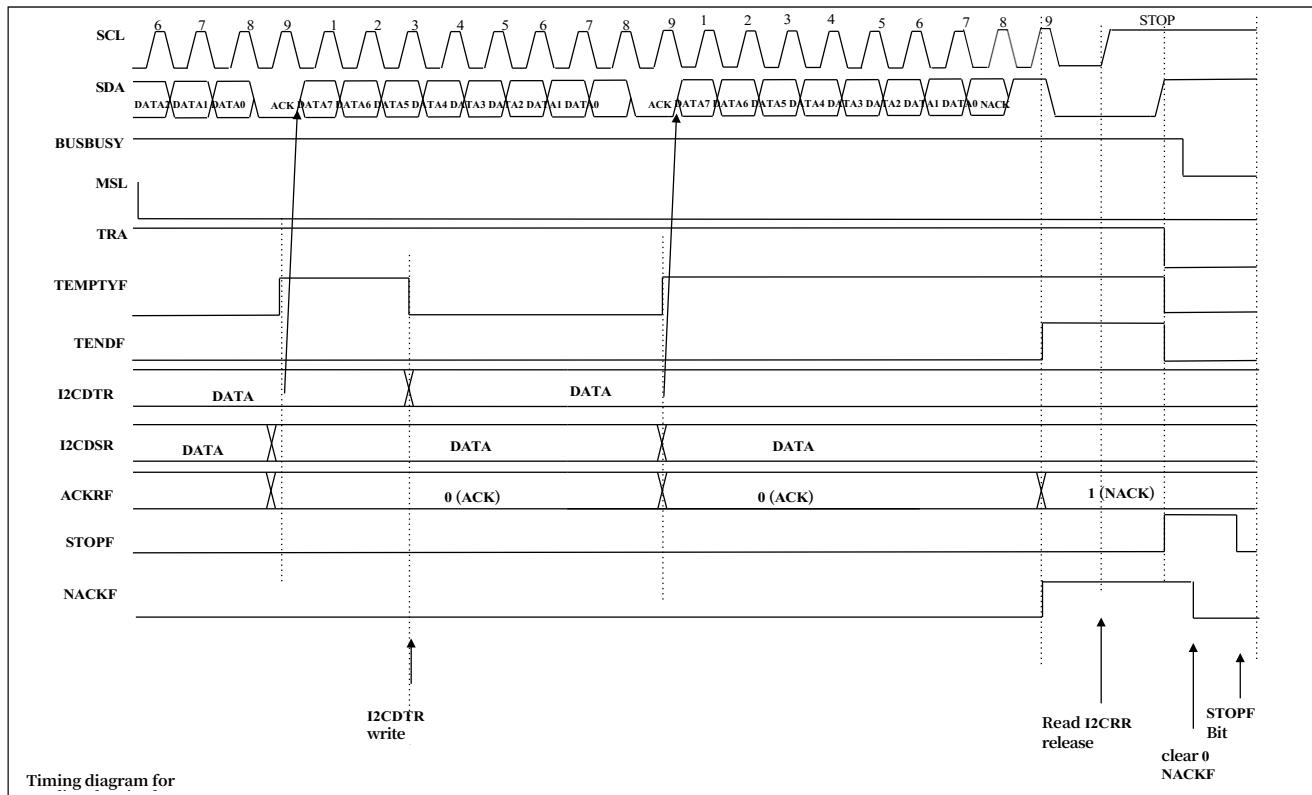
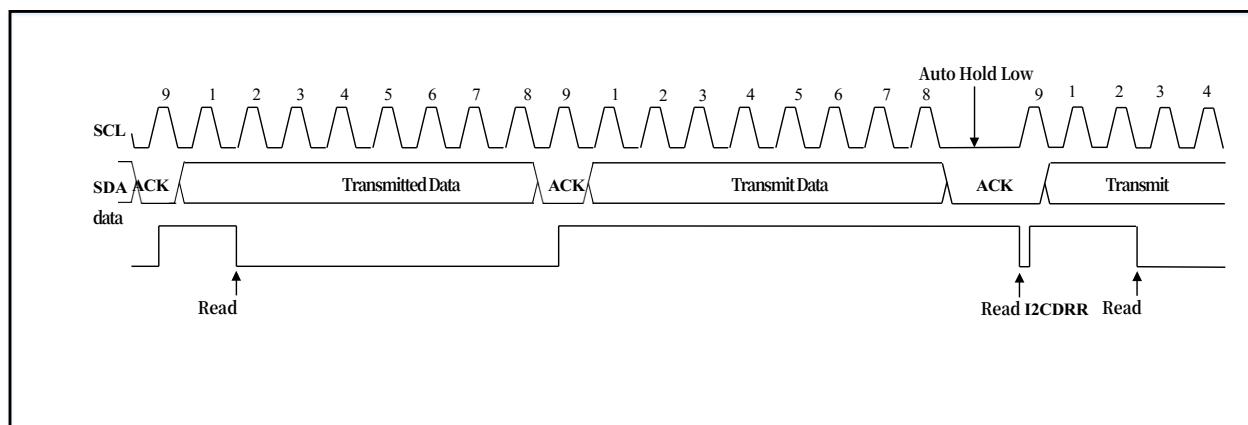


Figure 26-11 Slave Transmit Timing Diagram (2)

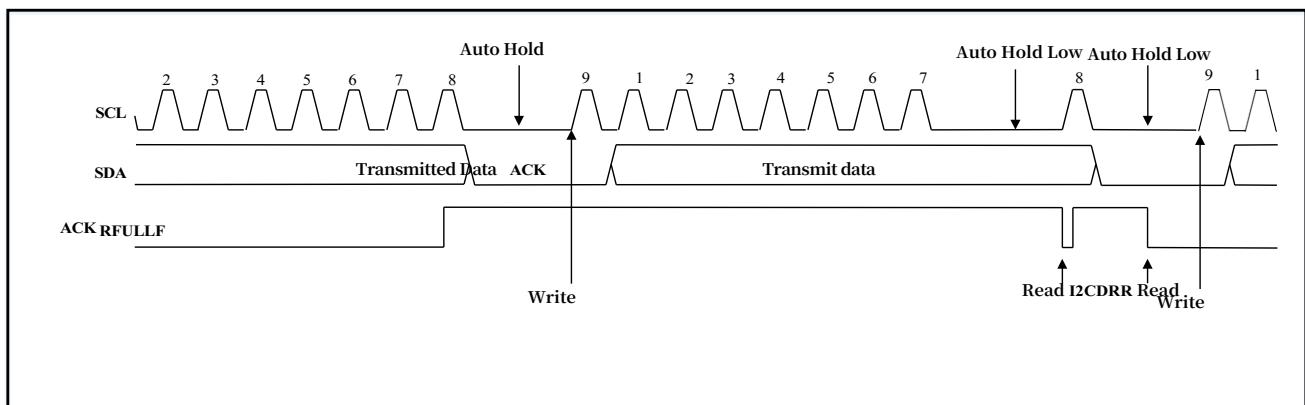
[Slave Receive Mode]

If a response processing delay occurs in the receive mode (I2C_SR.TRA bit = 0) and the receive data is full (I2C_SR.RFULLF flag = 1) due to delaying the read of the receive data (I2C_DRR register) by at least one transmission frame, etc., the SCL line is automatically held low between the 8th SCL and 9th SCL clocks before starting the next data reception. line low between the 8th SCL and 9th SCL clock before starting the next data reception, and the action timing is shown below.



[Fast ACK/NACK].

In SMBUS communication, the system's built-in CRC operator is used to calculate the packet error code (PEC) for SMBUS or to check the received data. In the process of checking the PEC code, ACK or NACK is sent at the last byte depending on whether it matches or not. This necessitates holding SCL low at the falling edge of the 8th clock of the received last byte of SCL. This is done to meet the software processing time. Based on the calculation, the software writes the I2C_CR1.ACK bit to unlock SCL low. Fast ACK/NACK is controlled by the I2C_CR3.FACKEN bit and the action timing is shown in the figure below.



26.3.2 Address Matching

As a slave, you can set two types of addresses in addition to the broadcast address and the host notification address, and the slave address can be set in 7-bit address or 10-bit address format.

26.3.2.1 Slave address matching

This I2C bus can set two types of slave addresses and has a slave address detection function for each.

When SLADDR1EN and SLADDR0EN are "1", the slave address set by I2C_SLR1 and I2C_SLR0 registers can be detected.

If the set slave address matches, the corresponding SLADDR1F and SLADDR0F are set to "1" on the falling edge of the 9th clock of the SCL clock, and then the I2C_SR.RFULLF flag or I2C_SR.TEMPTYF flag is set to "1" according to the subsequent R/W# bit. TEMPTYF flag is set to "1" according to the subsequent R/W# bits. This generates a receive data full interrupt or a send data interrupt, and determines which slave address is specified by checking the I2C_SR.SLADDR1F and SLADDR0F flags.

The timing of the SLADDR1F and SLADDR0F flags changing to "1" is shown in the figure below.

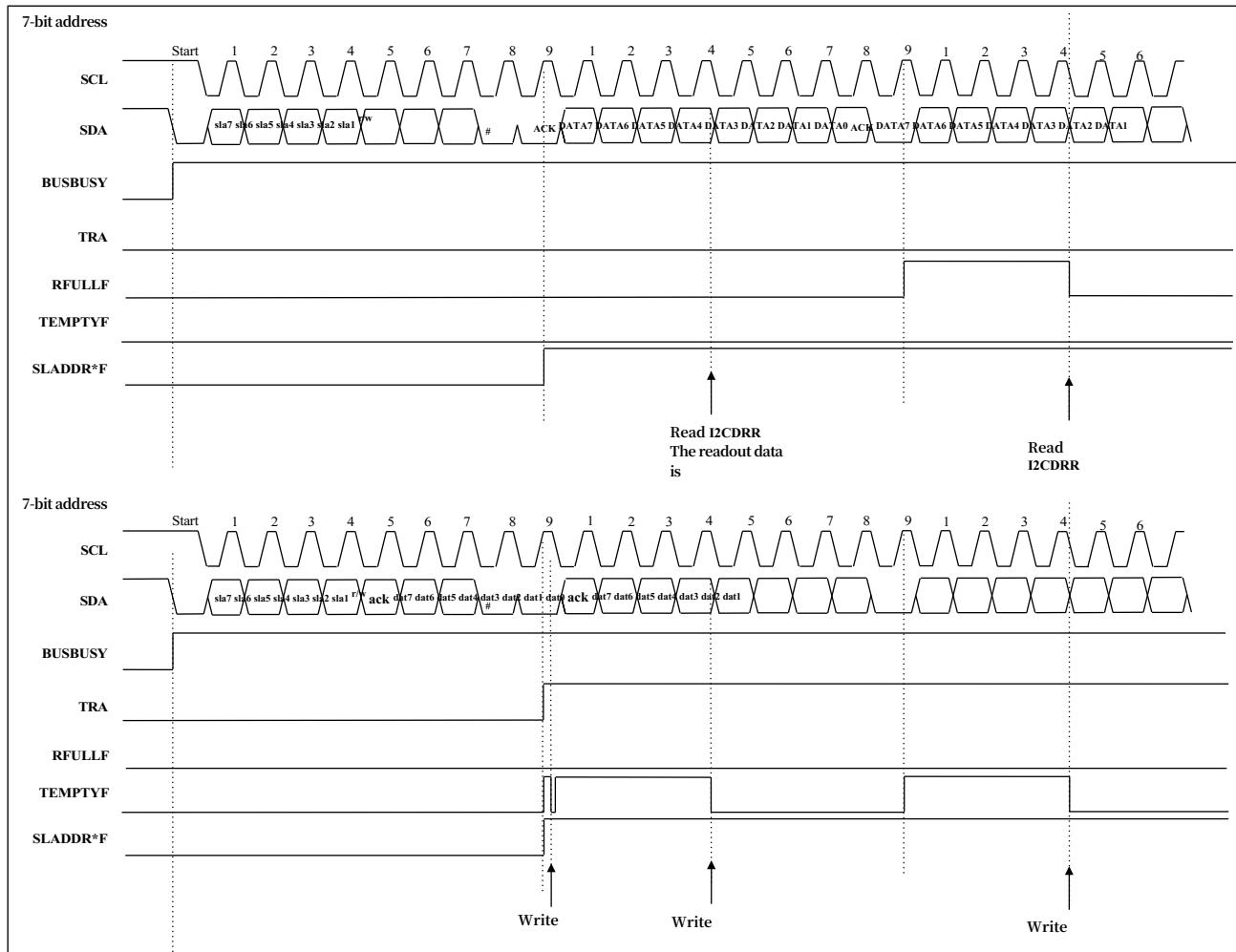


Figure 26-12 Timing when 7-bit address format is selected

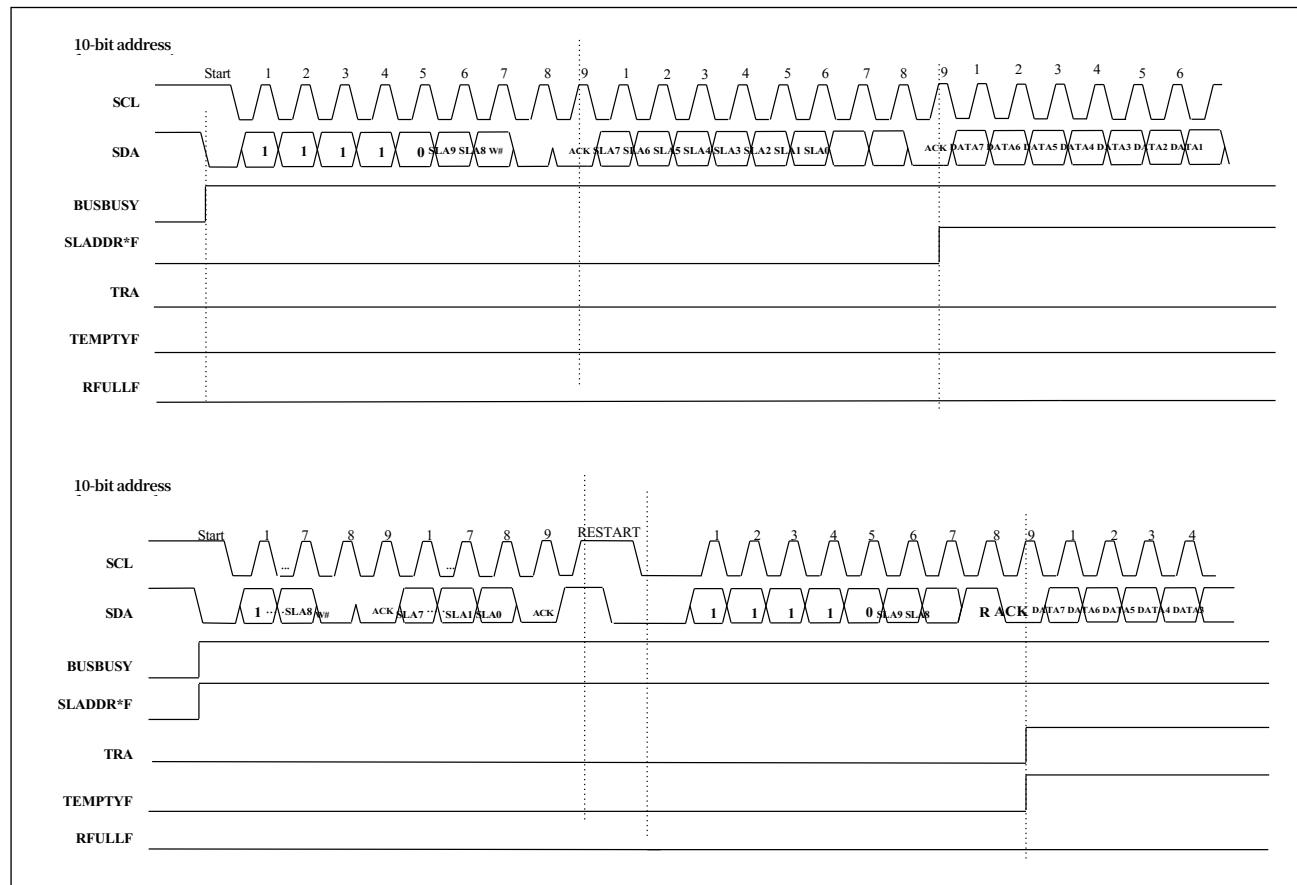


Figure 26-13 Timing when 10-bit address format is selected

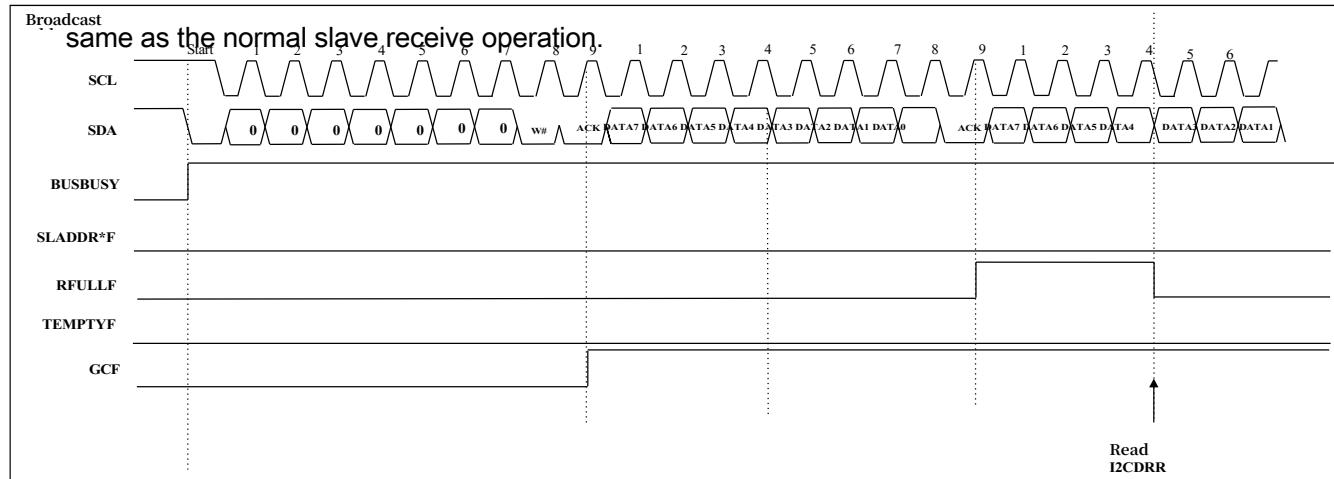
26.3.2.2 Broadcast address matching

When the I2C_CR1.GCEN bit is "1", the broadcast address (0000 000b+0[W]) can be detected.

However, if the address after the start condition or restart condition is 0000 000b+1[R] (start byte) this address is considered as the slave address of All "" and not as the broadcast address.

If the broadcast address is matched, the I2C_SR.GCF flag is set to "1" at the falling edge of the

9th clock of the SCL clock. The operation after a consistent broadcast address match is the

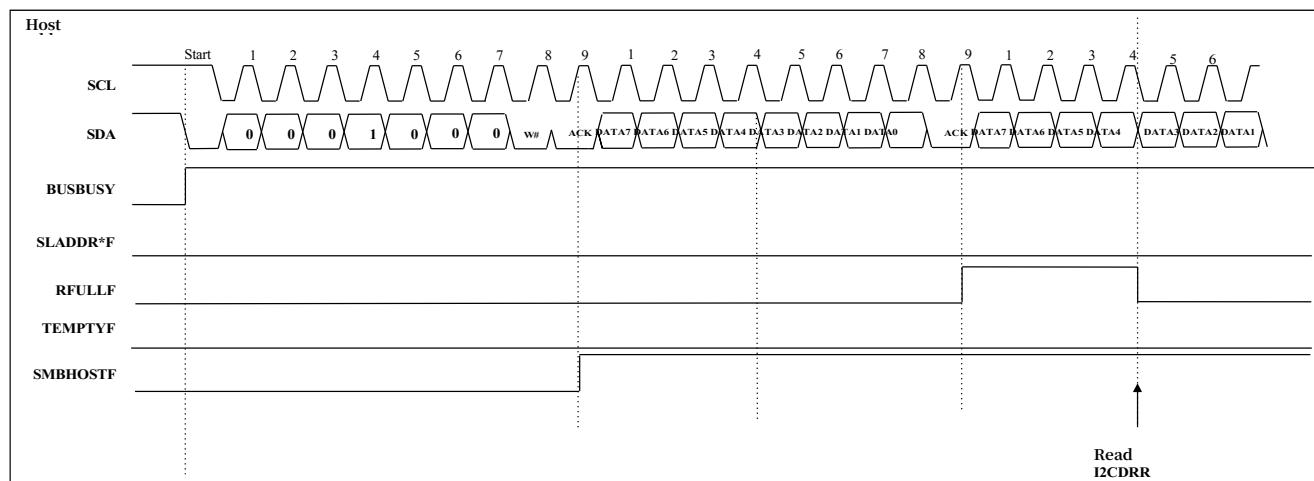


26.3.2.3 SMBus Host Address Matching

This product has a host address detection function during SMBus operation. If the I2C_CR1.SMBHOSTEN position "1" is set when the I2C_CR1.SMBUS bit is "1", the host address (0001 000b) can be detected in the slave receive mode (I2C_CR1.MSL bit TRA bit is "00b"). The host address (0001 000b) is detected in the slave receive mode (I2C_CR1.MSL bit "00 b")

If the SMBUS host address is detected, set the I2C_SR_SMBHOSTF flag to "1" at the falling edge of the 9th clock of the SCL clock.

The SMBUS host address is detected even if the bit following the SMBUS host address (0001 000b) is a Rd bit (R/W# bit receives a "1") operation after SMBUS host address detection is the same as normal slave mode operation.



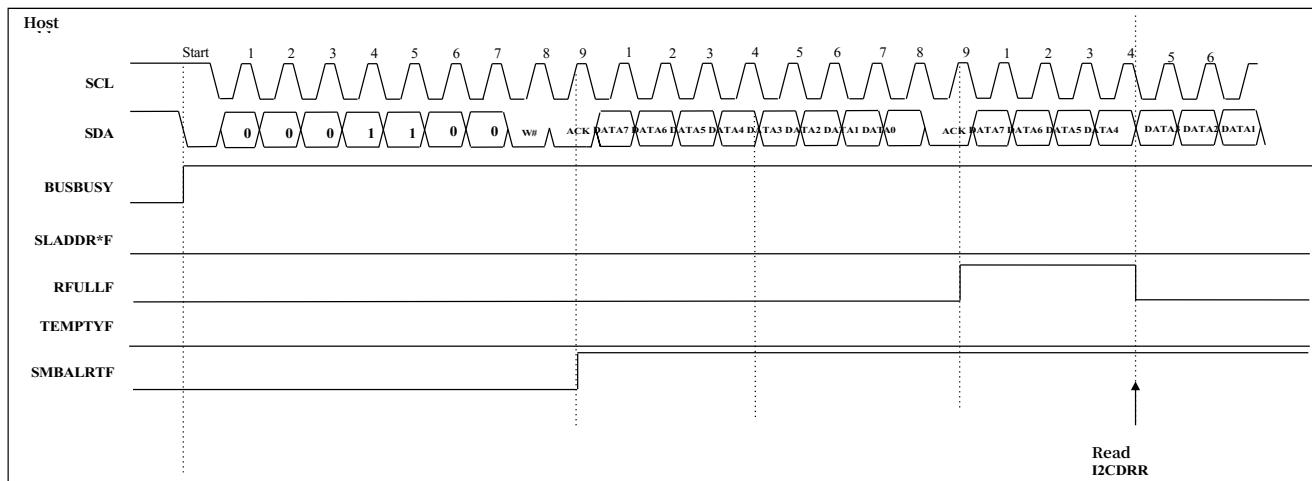
26.3.2.4 SMBus Alarm Response Address Matching

This product has an alarm response address detection function during SMBus operation. If the I2C_CR1.SMBUS

SMBARLERTEN = 1 when the I2C_CR1.SMBARLERTEN bit = 1, it is possible to detect the SMBUS alarm response address (0001 100b) in slave receive mode. The SMBUS alarm response address (0001 100b) is detected in the slave receive mode (I2C_CR1.MSL bit TRA bit = "00b")

If the SMBUS alarm response address is detected, the SMBUS alarm response address is set at the ninth digit of the SCL clock. The SMBARLERTF flag is set to "1" at the fall of the 9th clock of SCL.

SMBUS alarm response operation after address detection is the same as normal slave mode operation.

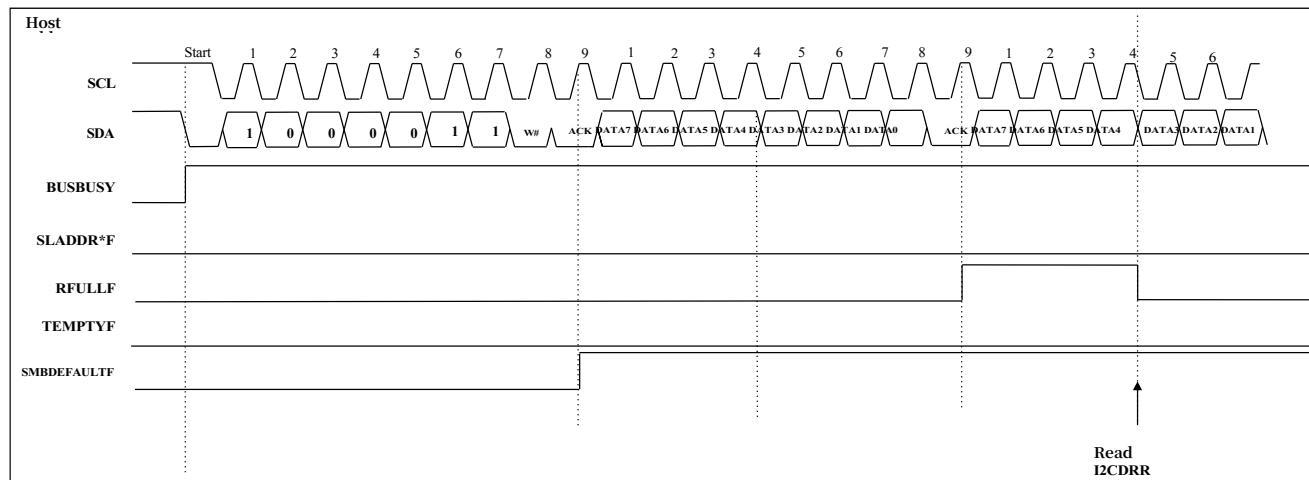


26.3.2.5 SMBus Default Address Matching

This product has a default address detection function for SMBus operation. If the I2C_CR1.SMBDEFAULTEN bit is set to "1" when the I2C_CR1.SMBUS bit is "1" can be detected in slave receive mode (I2C_CR1.MSL bit TRA bit "00b"). The SMBUS default address (1100 001b) is detected in slave receive mode (I2C_CR1.MSL bit TRA is "00b")

If the SMBUS default address is detected, the I2C_SR.SMBDEFAULTF flag is set to "1" at the falling edge of the 9th clock of the SCL clock.

SMBUS runs the same as normal slave mode operation after default address detection.



26.3.3 SMBus Action

This I2C interface is capable of SMBus (Ver.2.0) based communication. For SMBus communication, the I2C_CR1.SMBUS position must be set. CKDIV[2:0] bits and the I2CCCR register must be set to a transmission speed in the range of 10kbps to 100kbps of the SMBus specification.

26.3.3.1 SMBus Timeout Measurement

1) SCL level timeout measurement

The bus busy state can be detected by detecting that the SCL line has been fixed low or high for more than a certain period of time, and by detecting an abnormal bus state.

The timeout detection function monitors the state of the SCL line and counts the time spent high or low by the internal counter. If there is a change in the SCL line (rise/fall), it is reset, otherwise it continues to count. If the internal counter counts to the TOUTHIGH/TOUTLOW setting while the SCL line is not changing, a timeout is detected and the bus is notified of the abnormal status.

For internal counter counting, you can select whether to count at a low or high state of the SCL line, or both low and high states by setting the HTMOUT and LTMOUT bits. If both HTMOUT and LTMOUT bits are set to "0", no internal counting is performed.

2) Timeout measurement for slaves

The slave device for SMBus communication needs to measure the interval shown below (timeout interval: TLOW:SEXT)

- Start condition to stop condition interval

For timeout measurement via the slave device, the time from the detection of the start condition to the detection of the stop condition is measured using the start condition detection interrupt and the stop condition detection interrupt and via the chip timer. This timeout measurement time must be within 25ms (max) of the cumulative time of the SMBus specification clock low [slave] TLOW:SEXT.

If the timer measurement time exceeds the timeout for clock Low level detection of the SMBus specification TTIMEOUT: 25ms

(min) the slave will need to release the bus.

3) Timeout measurement of the host

The master device for SMBus communication needs to measure the interval shown below (timeout interval: TLOW:MEXT)

- Start condition to answer bit interval
- The interval from the answer bit to the next answer bit
- The interval from the answer bit to the stop condition

When the timeout measurement is performed by the host, the time of each interval is measured by the chip timer using the start condition detection interrupt, the stop condition detection interrupt, and the end-of-send interrupt or the receive data full interrupt. The measurement time of this timeout must be within the cumulative time of the clock low of the SMBus specification [host] TLOW:MEXT: 10ms (max), and the cumulative result of all TLOW:MEXT from the start condition to the stop condition must be within TLOW:SEXT: 25ms (max).

If the timer measurement time exceeds the cumulative time of the clock low level of the SMBus specification [Master] TLOW:MEXT: 10ms (max) or if the cumulative result of each measurement time exceeds the cumulative time of the SMBus The host is required to abort processing if the cumulative result of each measurement time exceeds the timeout TTIMEOUT: 25ms (min) detection of the SMBus specification. The transmit must be aborted immediately when the host transmits (write I2C_DTR register) Abort host processing by issuing a stop condition.

26.3.3.2 Packet Error Code (PEC)

In communication, the CPU is used to compute CRC, send packet error codes (PEC) for SMBus or check the received data.

26.3.4 Reset

There is a function to reset the communication module. There are two types of resets: one is a reset to initialize all registers including the ICCR2.BBSY flag, and the other is an internal reset to release the slave address matching status and initialize the internal counter while maintaining various set values.

SWRST position "0" must I2C_CR1.

It can also be used to release the bus from an unexpected stop state since either reset will release the output state of the SCL pin/SDA pin and change to a high impedance state.

Resets in slave mode can cause desynchronization with the master device, so try to avoid using them. It must be noted that the bus status of the start condition etc. cannot be monitored during a reset (I2C_CR1.PE bit and I2C_CR1.SWRST bit "01b").

26.3.5 Interrupt and event signal output

I2C has four interrupts and event outputs for triggering the start of other peripheral circuits for user selection. These include: occurrence of a communication error (arbitration failure detection, NACK detection, timeout detection, start condition detection, stop condition detection) end of receive, send data null, and end of send.

The list of interrupts is shown below.

Name	Interrupt source	Interrupt flags	Interrupt conditions
I2C_EEI	Communication errors/communication events	ARLOF	ARLOF=1&ARLOIE=1
		SLADDR0F	SLADDR0F=1& SLADDR0IE=1
		SLADDR1F	SLADDR1F=1& SLADDR1IE=1
		SMBALRTF	SMBALRTF =1& SMBALRTIE=1
		SMBHOSTF	SMBHOSTF =1& SMBHOSTFIE=1
		SMBDEFAULTF	SMBDEFAULTF =1& SMBDEFAULTIE=1
		GENCALLF	GENCALLF =1& GENCALLIE=1
		NACKF	NACKF=1&NACKIE-1
		TMOUTF	TMOUTF=1&TMOUTIE=1
		STARTF	STARTF=1&STARTIE=1
		STOPF	STOPF=1&STOPIE=1
I2C_RXI	Receiving data full	RFULLF	RFULLF=1&RFULLIE=1
I2C_TXI	Send data empty	TEMPTYF	TEMPTYF=1&TEMPTYIE=1
I2C_TEI	End of sending	TENDF	TENDF=1&TENDIE=1

The event signal output list is shown below.

Name	Event Source	Event Conditions
I2C_EEI	Communication error/communication time	ARLOF=1
		SLADDR0F=1
		SLADDR1F=1
		SMBALRTF=1
		SMBHOSTF=1
		SMBDEFAULTF=1
		GENCALLF=1
		NACKF=1
		TMOUTF=1
		STARTF=1

		STOPF=1
I2C_RXI	Receiving data full	RFULLF=1
I2C_TXI	Send data empty	TEMPTYF=1
I2C_TEI	End of sending	TENDF=1

26.3.6 Programmable digital filtering

The states of the SCL and SDA pins are internalized through the analog filter circuit and the digital filter. The block diagram of the digital filter circuit is shown in the following figure.

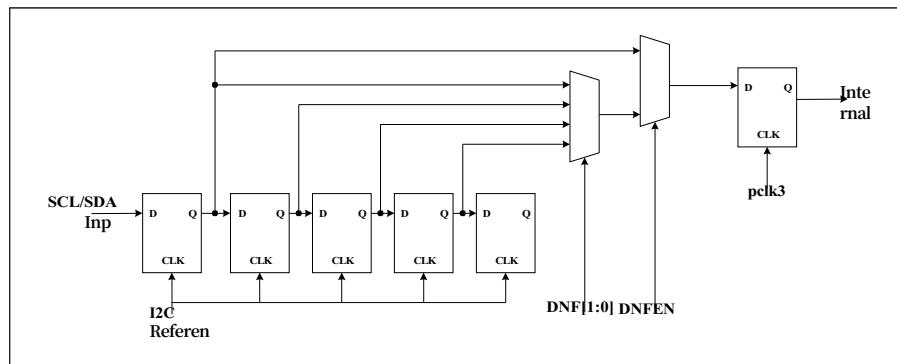


Figure 26-14 Block diagram of digital filtering circuit

The internal digital filter circuit consists of a 4-segment series trigger circuit and a match detection circuit.

The number of active segments of the digital filter is selected via the I2C_FLTR.DNF bit. Depending on the number of active segments selected, the noise cancellation capability is 1 to 4 I2C cycles.

The input signal of the SCL pin (or the input signal of the SDA pin) is sampled on the falling edge of the I2C internal clock, and if the trigger circuit output of the number of valid segments set by the I2C_FLTR.DNF bit matches all, the level is transmitted as an internal signal, otherwise the original value is maintained.

26.4 Application software sets up the I2C initialization process

When you start sending or receiving data, you must initialize it with the steps shown in the figure below.

1. The PE bit is set to 0.
2. SWRST set to 1, communication reset
3. PE bit set to 1, internal status reset
4. Set the slave address format and address
5. Set baud rate
6. Set control register functions and interrupts as required
7. The SWRST bit is set to 0 to release the internal status reset.
8. Initialization is finished. Data can be sent and received.

26.5 Register Description

I2C1 Base Address:

0x4004_E000 I2C2 Base

Address: 0x4004_E400 I2C3

Base Address: 0x4004_E800

Table 26-2 List of Registers

Register Name	Sym bols	Offset Address	Bit width	Reset value
I2C control register 1	I2C_CR1	0x00	32	0x0000_0040
I2C control register 2	I2C_CR2	0x04	32	0x0000_0000
I2C control register 3	I2C_CR3	0x08	32	0x0000_0006
I2C control register 4	I2C_CR4	0x0C	32	0x0030_0307
I2C slave address register 0	I2C_SLR0	0x10	32	0x0000_1000
I2C slave address register 1	I2C_SLR1	0x14	32	0x0000_0000
I2C Status Register	I2C_SLTR	0x18	32	0xFFFF_FFFF
I2C Status Register	I2C_SR	0x1C	32	0x0000_0000
I2C Status Register	I2C_CLR	0x20	32	0x0000_0000
I2C data transmission register	I2C_DTR	0x24	8	0xFF
I2C Data Receive Register	I2C_DR	0x28	8	0x00
I2C Baud Rate Control Register	I2C_CCR	0x2C	32	0x0000_1F1F
I2C Baud Rate Control Register	I2C_FLTR	0x30	32	0x0000_0010

26.5.1 I2C control register 1 (I2C_CR1)

Reset value: 0x0000_0040

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SWR ST	-	-	-	-	ACK	STOP	STA RT	RES TART	ENG C	-	SMB HOST EN	SMBD EFAU LTEN	SMB ALRT EN	SMB US	PE

position	Marker	Place Name	Function	Reading and writing									
b31-16	Reserved	-	Read "0", write "0" when writing	R/W									
b15	SWRST	Software Reset	0: Release reset 1: Software reset This bit is combined with PE bit to select internal status reset or communication reset <table border="1" style="margin-left: 20px;"> <tr> <th>SWRST</th><th>PE</th><th>Reset content</th></tr> <tr> <td>1</td><td>0</td><td>Communication reset: All registers and internal status reset inside I2C.</td></tr> <tr> <td>1</td><td>1</td><td>Internal state reset: I2C_SR, I2C_DSR registers and internal state machine for reset</td></tr> </table>	SWRST	PE	Reset content	1	0	Communication reset: All registers and internal status reset inside I2C.	1	1	Internal state reset: I2C_SR, I2C_DSR registers and internal state machine for reset	R/W
SWRST	PE	Reset content											
1	0	Communication reset: All registers and internal status reset inside I2C.											
1	1	Internal state reset: I2C_SR, I2C_DSR registers and internal state machine for reset											
b14-11	Reserved	-	Read "0", write "0" when writing	R/W									
b10	ACK	Send an answer	0: Answer bit sends "0" (sends ACK) 1: Answer bit sent "1" (NACK sent)	R/W									
b9	STOP	Stop condition generation bit	0: No stop condition is generated 1: Generate stop conditions This bit can be software set to 1 and cleared to 0. Hardware clear 0 conditions: Stop condition detected Failed arbitration Start condition detected Communication Reset	R/W									
b8	START	Start condition generation bit	0: No start condition is generated 1: Generate starting conditions This bit can be software set to 1 and cleared to 0. Hardware clear 0 conditions: Start condition detected When arbitration fails Communication Reset	R/W									
b7	RESTART	Repeat start condition generation bit	0: No duplicate start condition is generated 1: Generate duplicate start conditions	R/W									

This bit can be software set to 1 and cleared to 0.

Hardware clear 0 conditions:

- 1) Start condition is detected
 - 2) When arbitration fails
 - 3) Communication reset
-

b6	ENGC	Broadcast call enable	0: Broadcast address detection is invalid 1: Broadcast address detection is valid	R/W
b5	Reserved	-	Read '1', write "0" when writing	R/W
b4	SMBHOSTEN	Allow matching SMBUS host address bits	0: Matching of SMBUS host addresses is prohibited 1: Allow matching SMBUS host address	R/W
b3	SMBDEFAULTEN	Allow matching SMBUS default address bits	0: Disable matching SMBUS default address 1: Allow matching SMBUS default address	R/W
b2	SMBALRTEN	Allows matching SMBUS alarm response address bits	0: Disable SMBUS alarm response address 1: Allow SMBUS alarm response address	
b1	SMBUS	SMBUS/I2C bus mode selection bit	0: I2C bus mode 1: SMBUS bus mode	R/W
b0	PE	I2C function enable	0: I2C function is disabled 1: I2C function allowed This bit is combined with SWRST bit to select internal status reset or communication reset	R/W

26.5.2 I2C control register 1(I2C_CR2)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	SMB ALRT IE	SMB HOST IE	SMB DEFA ULTI	GEN CALL IE	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	TMO UTIE	-	NAC KIE	-	-	ARL OIE	-	TEMP TYIE	RFUL LIE	-	STO PIE	TEN DIE	SLA DDR1 IE	SLA DDR0 IE	STA RTIE

position	Marker	Place Name	Function	Reading and writing
b31~b24	Reserved	-	Read 0, write "0" when writing	R/W
b23	SMBALRTIE	SMBUS Alarm Response Address Match Consistent Interrupt Allowed	0: SMBUS alarm response address match consistent interrupt disable 1: SMBUS alarm response address match consistent interrupt allowed	R/W
b22	SMBHOSTIE	SMBUS Host Address Match Consistent Interrupt Allowed	0: SMBUS host address match consistent interrupt disable 1: SMBUS host address match consistent interrupt allowed	R/W
b21	SMBDEFAULTIE	SMBUS Default Address Match Consistent Interrupt Allowed	0: SMBUS default address match consistent interrupt disable 1: SMBUS default address match consistent interrupt allowed	R/W
b20	GENCALLIE	Broadcast call address match consistent interrupt allowed	0: Broadcast call address match consistent interrupt disable 1: Broadcast call address match consistent interrupt allowed	R/W
b19~b15	Reserved	-	Read 0, write "0" when writing	R/W
b14	TMOUTIE	Timeout interrupt allowed	0: Timeout interrupt disable 1: Timeout interrupt allowed	R/W
b13	Reserved	-	Read 0, write "0" when writing	R/W
b12	NACKIE	NACK interrupt allowed	0: Received NACK interrupt disable 1: Received NACK interrupt allowed	R/W
b11~b10	Reserved	-	Read 0, write "0" when writing	R/W
b9	ARLOIE	Arbitration failure interruption allowed	0: Arbitration failed to interrupt the bus 1: Arbitration failure interruption allowed	R/W
b8	Reserved	-	Read 0, write "0" when writing	R/W
b7	EMPTYIE	Send data air break allow bit	0: Sending data over the air is prohibited 1: Sending data air break allowed	R/W
b6	RFULLIE	Receive data full interrupt allow bit	0: Receive data full interrupt disable 1: Receive data full interrupt allowed	R/W
b5	Reserved	-	Read 0, write "0" when writing	R/W
b4	STOPIE	Stop condition interrupt allowed	0: Bus detects stop condition interrupt disable 1: Bus detects stop condition interrupt allowed	R/W

b3	TENDIE	Send a frame of data end interrupt allow bit	0: Send a frame of data end interrupt disable 1: Send a frame of data end interrupt allowed	R/W
b2	SLADDR1IE	Slave Address 1 Match Consistent Interrupt Allowed	0: Slave address 1 match consistent interrupt disable 1: Slave address 1 match consistent interrupt allowed	R/W
b1	SLADDR0IE	Slave address 0 match consistent interrupt allowed	0: Slave address 0 match consistent interrupt disable 1: Slave address 0 match consistent interrupt allowed	R/W
b0	STARTIE	Start condition/restart bar	0: Bus detects start condition interrupt disable	R/W

Interrupt Allowance

1: Bus detects start condition interrupt allowed

26.5.3 I2C control register 1(I2C_CR3)

Reset value: 0x0000_0006

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	FACK EN	-	-	-	-	HTM OUT	LTM OUT	TMOUT TEN

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Read "", write "0" when writing	R/W
b7	FACKEN	RFULLF flag position bit time point selection	0: This bit is "1" after clock up of the SCL clock. (SCL line is not held low at the falling edge of the 8th clock) 1: This bit is "1" after clock up of the SCL clock. (SCL line remains low at the falling edge of the 8th clock) The held low is released by writing the ACK bit.	R/W
b6~b3	Reserved	-	Read "", write "0" when writing	R/W
b2	HTMOUT	High level timeout detection allowed	0: Timeout detection is disabled when the SCL line is high. 1: Timeout detection is allowed when the SCL line is high.	R/W
b1	LTMOUT	Low level timeout detection allowed	0: Timeout detection is disabled when the SCL line is low. 1: Timeout detection is allowed when the SCL line is low.	R/W
b0	TMOUTEN	Timeout function allowed bit	0: Detection of SCL level timeout function is disabled 1: Detect SCL level timeout function allows	R/W

26.5.4 I2C control register 1(I2C_CR4)

Reset value: 0x0030_0307

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	BUS WAI	-	-	-	-	-	-	-	-	-	-

position	Marker	Place Name	Function	Reading and writing
b31~b22	Reserved	-	Read "", write "0" when writing	R/W
b21~b20	Reserved	-	Read "", write "" when writing	R/W
b19~b11	Reserved	-	Read "", write "0" when writing	R/W
b10	BUSWAIT	Bus waiting position	0: When I2C_DRR is full and I2C_DSR is empty, it does not stay low between the 9th clock and the 1st clock of the next transmission, and continues to receive the next data. 1: When I2C_DRR is full and I2C_DSR is empty, it remains low between the 9th clock and the 1st clock of the next transmission, and continues to receive the next data by reading the I2C_DRR register. Release the held low level.	R/W
b9~b8	Reserved	-	Read "", write "" when writing	R/W
b7~b3	Reserved	-	Read "", write "0" when writing	R/W
b2~b0	Reserved	-	Read "", write "" when writing	R/W

26.5.5 I2C slave address register 0 (I2C_SLR0)

Reset value: 0x0000_1000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADD	-	-	SLA	-	-	SLADDR0[9:0]									
RMO	-	-	DDR0	-	-										
D0	-	-	EN	-	-										

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Read "/" , write "0" when writing	R/W
b15	ADDRMOD0	7-bit/10-bit address format selection bit	0: Select 7-bit address format 1: Select 10-bit address format	R/W
b14~b13	Reserved	-	Read "/" , write "0" when writing	R/W
b12	SLADDR0EN	Slave address 0 valid bits	0: Slave address register 0 setting value is invalid 1: Slave address register 0 setting value is valid	R/W
b11~b10	Reserved	-	Read "/" , write "0" when writing	R/W
b9~b8	SLADDR0[9:8]	High bit of 10-bit slave address	Set the slave address. When the ADDRMOD0 bit is "0", this bit setting is invalid. When the ADDRMOD0 bit is " "1 ", this bit is used as the high two bits of the 10-bit slave address.	R/W
b7~b0	SLADDR0[7:0]	7-bit address / low bit of 10-bit address	Set the slave address. When the ADDRMOD0 bit is "0", SLADDR0[7:1] is the 7-bit slave ground The SLADDR0[0] bit is invalid. When the ADDRMOD0 bit is " "1 ", SLADDR0[7:0] is the low 8-bit address of the 10-bit slave address.	R/W

26.5.6 I2C slave address register 1 (I2C_SLR1)

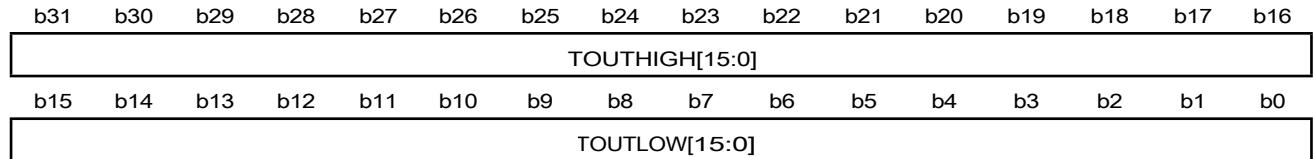
Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADD	-	-	SLA	-	-	SLADDR1[9:0]									
RMO	-	-	DDR1	-	-										
D1			EN												

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Read <code>r</code> , write "0" when writing	R/W
b15	ADDRMOD1	7-bit/10-bit address format selection bit	0: Select 7-bit address format 1: Select 10-bit address format	R/W
b14~b13	Reserved	-	Read <code>r</code> , write "0" when writing	R/W
b12	SLADDR1EN	Slave address 1 valid bit	0: Slave address register 1 setting value is invalid 1: Slave address register 1 setting value is valid	R/W
b11~b10	Reserved	-	Read <code>r</code> , write "0" when writing	R/W
b9~b8	SLADDR1[9:8]	High bit of 10-bit slave address	Set the slave address. When the ADDRMOD1 bit is "0", this bit setting is invalid. When the ADDRMOD1 bit is "1", this bit is used as the high two bits of the 10-bit slave address.	R/W
b7~b0	SLADDR1[7:0]	7-bit address / low bit of 10-bit address	Set the slave address. When the ADDRMOD1 bit is "0", SLADDR1[7:1] is the 7-bit slave ground The SLADDR1[0] bit is invalid. When the ADDRMOD1 bit is "1", SLADDR1[7:0] is the low 8 bits of the 10-bit slave address.	R/W

26.5.7 I2C SCL Level Timeout Control Register (I2C_SLTR)

Reset value: 0xFFFF_FFFF



position	Marker	Place Name	Function	Reading and writing
b31~b16	TOUTHIGH	SCL high timeout period	TOUTHIGH sets the SCL high timeout period. SCL high timeout = TOUTHIGH × I2C reference clock period	R/W
b15~b0	TOUTLOW	SCL low timeout period	TOUTLOW sets the SCL low timeout period. SCL low timeout = TOUTLOW × I2C reference clock period	R/W

26.5.8 I2C Status Register (I2C_SR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	SMB ALR TF	SMB HOS TF	SMBD EFAU LTF	GENC ALLF	-	TRA	BUSY	MSL
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	TMO UTF	-	NAC KF	-	ACK RF	ARL OF	-	TEM PTYF	RFU LLF	-	STO PF	TEN DF	SLAD DR1F	SLAD DR0F	STA RTF

position	Marker	Place Name	Function	Reading and writing
b31~b21	Reserved	-	Read "0", write "0" when writing	R
b23	SMBALRTF	SMBUS Alarm Response Address Match Consistency Flag Bit	0: No match to SMBUS alarm response address 1: Host address set to "1" is detected: The received address matches 0001 100b with a consistent clear "0" condition: SMBALRTFCLR writes "0" Stop condition communication reset detected	R
b22	SMBHOSTF	SMBUS host address match consistency flag bit	0: No match to SMBUS host address 1: Match to the SMBUS host address address match consistent conditions as follows: Set "1" condition: The received address and 0001 000b match the same clear "1" condition: SMBHOSTFCLR write "0" Stop condition communication reset detected	R
b21	SMBDEFAULTF	SMBUS Default Address Match Consistency Flag Bit	0: No match to SMBUS default address 1: Match to SMBUS default address set to "1" condition: The received address matches the 1100 001b with a consistent clear "0" condition: SMBDEFAULTFCLR writes "0" Stop condition communication	R

reset detected

b20	GENCALLF	Broadcast call address match consistency flag	0: No match to broadcast call address 1: Match to broadcast call address set to "1** condition: Clear the "0" condition when the received slave address matches the broadcast call address (All "0"): GENCALLFCLR writes "1" Stop condition communication reset detected	R
b19	Reserved	-	Read "1", write "0" when writing	R
b18	TRA	Send Receive Select Bit	This bit indicates whether to choose to send data or receive data. 0: Receive data 1: Send data This bit can be software set to 1 and cleared to 0. Hardware set "1" condition Start condition detected The R/W bit sent in host mode is 0 In slave mode, the address matches and the received R/W bit is 1 hardware clear "0" condition Stop condition detected The R/W bit sent in host mode is 1 In slave mode, the address matches and the received R/W bit is 0 communication reset	R/W
b17	BUSY	Bus busy flag bit	0: Idle state, no communication on the bus 1: Possession status, bus is communicating Set "1" condition Start condition clear "0" condition detected on the bus: Bus stop condition detected Communication Reset	R

b16	MSL	Master-slave selection bit	This bit indicates whether the master or the slave. 0: Slave mode 1: Host mode The combination of the MSL and TRA bits indicates the I2C operation mode. <table border="0"> <thead> <tr> <th>MSL mode</th><th>TRA</th><th>I2C operation</th></tr> </thead> <tbody> <tr> <td>0 mode</td><td>0</td><td>Slave receive</td></tr> <tr> <td>0 mode</td><td>1</td><td>Slave transmit</td></tr> <tr> <td>1 mode</td><td>0</td><td>Host receive</td></tr> <tr> <td>1 mode</td><td>1</td><td>Host transmit</td></tr> </tbody> </table> mode This bit can be software set to 1 and cleared to 0. Hardware set "1" condition If the START bit is 1, the start condition hardware clear "0" condition is detected. 1) Stop condition detected 2) Failed arbitration 3) Communication Reset	MSL mode	TRA	I2C operation	0 mode	0	Slave receive	0 mode	1	Slave transmit	1 mode	0	Host receive	1 mode	1	Host transmit	R/W
MSL mode	TRA	I2C operation																	
0 mode	0	Slave receive																	
0 mode	1	Slave transmit																	
1 mode	0	Host receive																	
1 mode	1	Host transmit																	
b15	Reserved	-	Read "1", write "0" when writing	R															
14	TMOUTF	Timeout flag bit	0: SCL level timeout not detected 1: SCL level timeout set to "1" condition: SCL is not flipped to clear "1" within the period set by I2C_SLR: TMOUTFCLR writes "1" Communication Reset	R															
b13	Reserved	-	Read "1", write "0" when writing	R															
b12	NACKF	NACK flag bit	0: NACK not received 1: NACK received	R															

Placement "1" condition:

In transmit mode, NACK

clear "0" condition is

received: NACKFCLR

write "1**"

Communication Reset

b11	Reserved	-	Read "0", write "0" when writing	R
b10	ACKRF	Receive answer bit	<p>0: Received answer bit is "0**" (Receive ACK)</p> <p>1: Receive answer bit "1**" (receive NACK) Set "1" condition:</p> <p>In transmit mode, the NACK clear "0"</p> <p>condition is received:</p> <p>In transmit mode, an ACK is received</p> <p>Communication Reset</p>	R
b9	ARLOF	Arbitration failure flag bit	<p>0: No arbitration failure occurred</p> <p>1: Arbitration failure set to "1"</p> <p>condition:</p> <p>Arbitration failure</p> <p>Clear "0"</p> <p>condition:</p> <p>ARLOFCLR</p> <p>writes "1**"</p> <p>Communication Reset</p>	R
b8	Reserved	-	Read "0", write "0" when writing	R
b7	TEMPTYF	Send data null flag bit	<p>0: I2C_DTR register full</p> <p>1: I2C_DTR register is vacant "1"</p> <p>condition:</p> <p>I2C_DTR data transfer to I2C_DSR</p> <p>TRA position 1</p> <p>Clear "0"</p> <p>condition:</p> <p>write I2C_DTR</p> <p>TRA bit clear 0</p> <p>Communication Reset</p>	R
b6	RFULLF	Receive data full flag bit	<p>0: I2C_DRR register empty</p> <p>1: I2C_DRR register full</p> <p>set "1" condition:</p> <p>The received data is transferred from I2C_DSR to I2C_DR clear "0"</p> <p>condition:</p> <p>Read I2C_DRR</p> <p>RFULLFCLR</p> <p>Write "1**"</p> <p>Communication Reset</p>	R
b5	Reserved	-	Read "0", write "0" when writing	R

b4	STOPF	Stop condition flag bit	0: No stop condition detected by the bus 1: The bus detects a stop condition set to "1" condition: Stop condition clear "0" condition detected: STOPFCLR writes "1" Communication Reset	R
b3	TENDF	Send data end flag bit	0: I2C_DSR register in transmit 1: I2C_DSR register send end	R
			Placement "1" condition: TEMPTYF = 1, the 9th rise of SCL along this position "1 ** Clear "0" condition: Stop condition is detected Write I2C_DTR TENDFCLR writes "1" Communication Reset	
b2	SLADDR1F	Slave Address Register 1 Match Consistency Flag	0: Slave address register 1 consistent address is not detected 1: Slave address register 1 consistent address set to "1**" condition is detected: When I2C_SLR1.ADDRMOD1 bit is "0", the received slave address and I2C_SLR1.SLADDR1[7:1] match. When the I2C_SLR1.ADDRMOD1 bit is "1", the first byte of the 10-bit slave address is received with 11110b + I2C_SLR1.SLADDR1[9:8] matches and the second byte address matches with I2C_SLR1.SLADDR1[7:0]. Clear "0" condition: Stop condition detected: SLADDR1FCLR write "1" Communication Reset	R

b1	SLADDR0F	Slave Address Register 0 Match Consistency Flag	0: Slave address register 0 consistent address is not detected 1: Slave address register 0 consistent address set to "1" condition is detected: When I2C_SLR0.ADDRMOD0 bit is "0", the received slave address and I2C_SLR0.SLADDR0[7:1] match. When I2C_SLR0.ADDRMOD0 bit is "1", the first byte address of the 10-bit slave address is received with 11110b + I2C_SLR0. SLADDR0[9:8] matches and the second byte address matches I2C_SLR0. SLADDR0[7:0]. Clear "0" condition: Stop condition detected SLADDR0FCLR write "1" Communication Reset	R
b0	STARTF	Start condition/restart condition flag bit	0: Start condition not detected by the bus 1: The bus detects the start condition and sets the "1" condition 1) The start condition is detected Clear "0" condition 1) Stop condition is detected 2) STARTFCLR writes "1" 3) Communication Reset	R

26.5.9 I2C Status Clear Register (I2C_CLR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	SMB ALRT FCLR	SMB HOST FCLR	SMB DEFA ULTF CLR	GEN CALL FCLR	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	TMO UTFC LR	-	NAC KFCLR	-	-	ARLO FCLR	-	TEM PTYF CLR	RFU LLFC LR	-	STO PFC LR	TEN DFC LR	SLA DDR1 FCLR	SLAD DR0F CLR	STA RTFC LR

position	Marker	Place Name	Function	Reading and writing
b31~b24	Reserved	-	Write "0" when writing in	W
b23	SMBALRTFCLR	SMBUS Alarm Response Address Match Consistency Flag Clear Bit	Write "1" to clear the SMBALRTF flag bit	W
b22	SMBHOSTFCLR	SMBUS host address match consistency flag clear bit	Write "1" to clear the SMBHOSTF flag bit	W
b21	SMBDEFAULTFCLR	SMBUS Default Address Match Consistency Flag Clear Bit	Write "1" to clear the SMBDEFAULTF flag bit	W
b20	GENCALLFCLR	Broadcast call address match consistency flag bit	Write "1" to clear the GENCALLF flag bit	W
b19~b15	Reserved	-	Write "0" when writing in	W
b14	TMOUTFCLR	Timeout flag bit	Write "1" to clear the TMOUTF flag bit	W
b13	Reserved	-	Write "0" when writing in	W
b12	NACKFCLR	NACK flag bit	Write "1" to clear the NACKF flag bit	W
b11~b10	Reserved	-	Write "0" when writing in	W
b9	ARLOFCLR	Arbitration failure flag bit	Write "1" to clear the ARLOF flag bit	W
b8	Reserved	-	Write "0" when writing in	W
b7	TEMPTYFCLR	Send data null flag bit	Write "1" to clear the TEMPTYF flag bit	W
b6	RFULLFCLR	Receive data full flag bit	Write "1" to clear the RFULLF flag bit	W
b5	Reserved	-	Write "0" when writing in	W
b4	STOPFCLR	Stop condition flag bit	Write "1" to clear the STOPF flag bit	W
b3	TENDFCLR	Send data end flag bit	Write "1" to clear the TENDF flag bit	W
b2	SLADDR1FCLR	Slave Address Register 1 Match Consistency Flag Clear Bit	Write "1" to clear the SLADDR1F flag bit	W
b1	SLADDR0FCLR	Slave Address Register 1 Match Consistency Flag Clear Bit	Write "1" to clear the SLADDR0F flag bit	W
b0	STARTFCLR	Start condition/restart condition flag clear bit	Write "1" to clear the STARTF flag bit	W

26.5.10 I2C Data Transmit Register (I2C_DTR)

Reset value: 0xFF

b7	b6	b5	b4	b3	b2	b1	b0
DT[7:0]							

If the I2C_DSR register is empty, the transmit data written in the I2C_DTR register is transferred to the I2C_DSR register and the transmit mode starts sending data to the SDA.

The I2C_DSR register and the I2C_DTR register are double buffered structures. If the data of the I2C_DTR register is pre-written during the I2C_DSR register data transmission, continuous data transmission is possible.

The I2C_DTR register is readable and writable. Please write the I2C_DTR register only once when the transmit data air break request occurs.

26.5.11 I2C Data Receive Register (I2C_DRR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
DR[7:0]							

If 1 frame of data is received, the received data can be dumped from the shift register (I2C_DSR) to the I2C_DRR register and can then move on to the next data reception state.

The I2C_DSR register and the I2C_DRR register are double buffered structures. During the I2C_DSR register data reception, if the data in the I2C_DRR register is read, continuous data reception can be performed.

Write to the I2C_DRR register is prohibited. Please read the I2C_DRR register only once when the receive data full interrupt request occurs.

RFULLF flag bit is "1", if the next data is received immediately instead of reading the data in the I2C_DRR register, the SCL clock is automatically held low for one SCL clock before the next RFULLF flag bit becomes "1". **The SCL clock is** automatically held low for one SCL clock before the next RFULLF flag bit becomes "1".

26.5.12 I2C Data Shift Register (I2C_DSR)

b7	b6	b5	b4	b3	b2	b1	b0
DSR							

T h e I2C_DSR register **i s** used as a shift register for transmitting and receiving data. the I2C_DSR register is neither readable nor writable.

When data is sent, the send data is transferred from the I2C_DTR register to the I2C_DSR register, and the data is sent from the SDA pin. During data reception, once 1 frame of data reception is finished, the data is transferred from the I2C_DSR register to the I2C_DR register.

26.5.13 I2C clock control register (I2C_CCR)

Reset value: 0x0000_1F1F

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	FREQ[2:0]
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	SHIGHW[4:0]				-	-	-	SLOWW[4:0]					

position	Marker	Place Name	Function	Reading and writing
b31~b23	Reserved	-	Read "0", write "0" when writing	R/W
b22~b19	Reserved	-	Read "0", write "1" when writing	R/W
b18-b16	FREQ[2:0]	I2C reference clock frequency setting position	0 0 0: I2C reference clock frequency = PCLK3/1 0 0 1: I2C reference clock frequency = PCLK3/2 0 1 0: I2C reference clock frequency = PCLK3/4 0 1 1: I2C reference clock frequency = PCLK3/8 1 0 0: I2C reference clock frequency = PCLK3/16 1 0 1: I2C reference clock frequency = PCLK3/32 1 1 0: I2C reference clock frequency = PCLK3/64 1 1 1: I2C reference clock frequency = PCLK3/128	R/W
b15~b13	Reserved	-	Read "0", write "1" when writing	R/W
b12~b8	SHIGHW[4:0]	Set SCL high level width bit	Set the high level width of the SCL clock	R/W
b7~b5	Reserved	-	Read "0", write "1" when writing	R/W
b4~b0	SLOWW[4:0]	Set SCL low level width bit	Set the low level width of the SCL clock	R/W

SHIGHW bit (sets the SCL high level width bit)

In master mode, SHIGHW is used to set the high level width of the SCL clock. In slave mode, the setting is invalid.

SLOWW bit (sets the SCL low level width bit)

SLOWW is used to set the low level width of the SCL clock. In slave mode, the setting value should be greater than the data ready time. Data ready time (tSU:DAT) 250ns (~100kbps: standard mode)
100ns (~400kbps: fast mode)

Baud rate:

DNFE=0,FREQ=000

Baud rate = $1 / \{ [(SHIGHW+3)+(SLOWW+3)] / \Phi I2C + SCL \text{ rise time} + SCL \text{ fall time} \}$

DNFE=1,FREQ=000

Baud rate = $1 / \{ [(SHIGHW+3+\text{filtering capability})+(SLOWW+3+\text{filtering capability})] / \Phi I2C + SCL \text{ Rise time} + SCL \text{ drop time} \}$

DNFE=0,FREQ!=000

Baud **rate**=1/{[(SHIGHW+2)+(SLOWW+2)]/ Φ I2C+SCL rise
time+SCL fall time} DNFE=1,FREQ!=000

Baud rate = $1/\{[(S\text{HIGHW}+2+\text{filtering capability})+(S\text{LOWW}+2+\text{filtering capability})]/\Phi\text{I}2\text{C}+\text{SCL Rise time}\}$

Cautio +SCL drop time}

n:

- SCL **The rise time [tr] of the line** and fall time **[tf]** depends on the total capacitance of the bus [Cb] and pull-up resistor [Rp], please refer to NXP's I2C bus specification for details.

26.5.14 I2C Filter Control Register (I2C_FLTR)

Reset value: 0x0000_0010

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	ANF EN	DNF EN	-	-	DNF[1:0]	

position	Marker	Place Name	Function	Reading and writing
b31~b6	Reserved	-	Read "0", write "0" when writing	R/W
b5	ANFEN	Analog filtering function allows bit	0: Analog filtering function is disabled 1: Analog filtering function allows	R/W
b4	DNFEN	Digital filtering function allows bit	0: Digital filtering function is disabled 1: Digital filtering function allows	R/W
b3~b2	Reserved	-	Read "0", write "0" when writing	R/W
b1-b0	DNF[1:0]	Digital filter filter capacity selection	00: Filtering capability 1 I2C reference clock cycle 01: Filtering capability 2 I2C reference clock cycles 10: Filtering capability 3 I2C reference clock cycles 11: Filtering capability 4 I2C reference clock cycles	R/W

27 Serial Peripheral Interface (SPI)

27.1 Introduction

This product is equipped with a 4-channel serial peripheral interface **SPI**, which supports high-speed full-duplex serial synchronous transmission for easy data exchange with peripheral devices. The user can set the 3-wire/4-wire, master/slave and baud rate range as required.

SPI Key Features:

Table 27-1 Highlights of SPI Characteristics

Key Points	Description
Number of channels	4 Channels
Serial communication function	<ul style="list-style-type: none">- Supports 4-wire SPI mode and 3-wire clock synchronous operation mode- Supports both full-duplex and transmission-only communication methods- Adjustable polarity and phase of communication clock SCK
Data Format	<ul style="list-style-type: none">- Selectable data shift order: MSB start/LSB start- Selectable data width: 4/5/6/7/8/9/10/11/12/13/14/15/16/20/24/32 bits- Transmit or receive up to 4 frames of 32-bit width data in a single pass
Baud rate	<ul style="list-style-type: none">- The baud rate can be adjusted in host mode by the built-in special baud rate generator, and the baud rate range is 2 divisions of PCLK ~256 crossover frequencies- The maximum baud rate allowed in slave mode is 6 divisions of PCLK
Data buffering	<ul style="list-style-type: none">- With 16-byte data buffer area- Support double buffering
Error Monitoring	<ul style="list-style-type: none">- Mode fault error monitoring- Data overload error monitoring- Data underload error monitoring- Parity error monitoring
Chip selection signal control	<ul style="list-style-type: none">- Four chip select signal lines per channel- The relative timing relationship between the chip select signal and the communication clock can be adjusted- Adjustable chip select signal invalidation time between two consecutive communications- Adjustable polarity
Transmission control in host mode	<ul style="list-style-type: none">- Initiate transfer by writing data to the data register- Communication auto-hang-up function

Interrupt/AOS source	- Receiving data area is full - Send data area is empty - SPI errors (mode/overload/underload/parity) - SPI Idle - Transmission completed
Low power control	Settable module stop
Other Functions	— SPI initialization function

Caution:

- When using the communication auto-hang function in the master receive mode, no overload error will occur because the communication clock is stopped. Refer to 27.8.2 Overload Errors for details.

27.2 SPI System Block Diagram

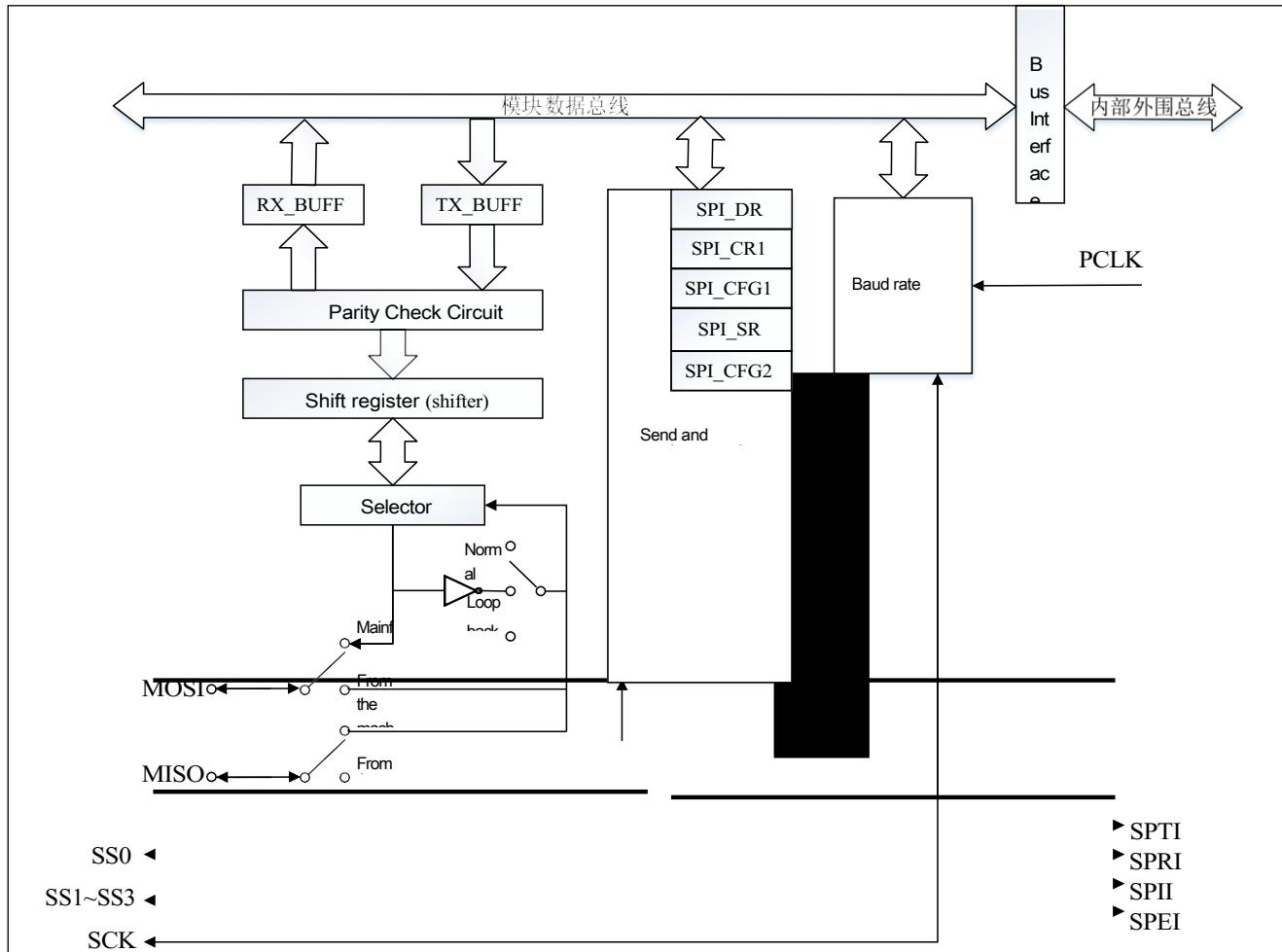


Figure 27-1 System Block Diagram

27.3 Pin

**Descript
ion**

Table 27-2 Pin Descriptions

Pin Name	Port Direction	Function
SCK	Input/output	Communication clock pin
MOSI	Input/output	Host data transmission pins
MISO	Input/output	Slave data transmission pins
SS0	Input/output	Slave select input/output pins

SS1	Output	Slave select output pin
SS2	Output	Slave select output pin
SS3	Output	Slave select output pin

27.4 SPI Action System Description

27.4.1 Host mode pin status

When the **SPI** is operating in host mode, the status of each pin is shown in the following table.

Table 27-3 SPI Pin Status Descriptions in Host Mode

Mode	Foot er Name	Pin Status (PFS.ODS=0)	Pin Status (PFS.ODS=1)
SPI Action (SPIMDS=0)	Host Mode (MSTR=1, MODFE=0)	SCK	CMOS output
		SS0~SS3	CMOS output
		MOSI	CMOS output
		MISO	Input
Clock synchronized operation (SPIMDS=1)	Host Mode (MSTR=1)	SCK	CMOS output
		SS0~SS3 (Do not make (Use)	Hi-Z (available as universal) (I/O)
		MOSI	CMOS output
		MISO	Input

Caution:

- When a valid level is input to SS0, the **SPI** relinquishes bus control and the pin state is Hi-Z.

27.4.2 Slave mode pin status

When the **SPI** is operating in slave mode, the status of each pin is shown in the following table.

Table 27-4 SPI Pin Status Descriptions in Slave Mode

Mode	Pin Name	Pin Status (PFS.ODS=0)	Pin Status (PFS.ODS=1)
SPI Action (SPIMDS=0)	Slave Mode (MSTR=0, MODFE=0)	SCK	Input
		SS0	Input
		SS1~SS3 (not used)	Hi-Z (available as general purpose I/O)
		MOSI	Input
		MISO (Note 2)	CMOS output / Hi-Z
Clock synchronized operation	Slave Mode (MSTR=0)	SCK	Input
		SS0~SS3 (Do not make (Use)	Hi-Z (available as general purpose I/O)

(SPI MDS=1)		MOSI	Input	Input
		MISO	CMOS output	OD Output

27.4.3 SPI System Connection Example

Host Mode

In the SPI system architecture of host-multi-slave mode, the host drives SCK, **MOSI**, and SS0 to SS3. In **SPI** slave devices 0 to 3, when the SS input of a slave is a valid level, the slave device drives MISO.

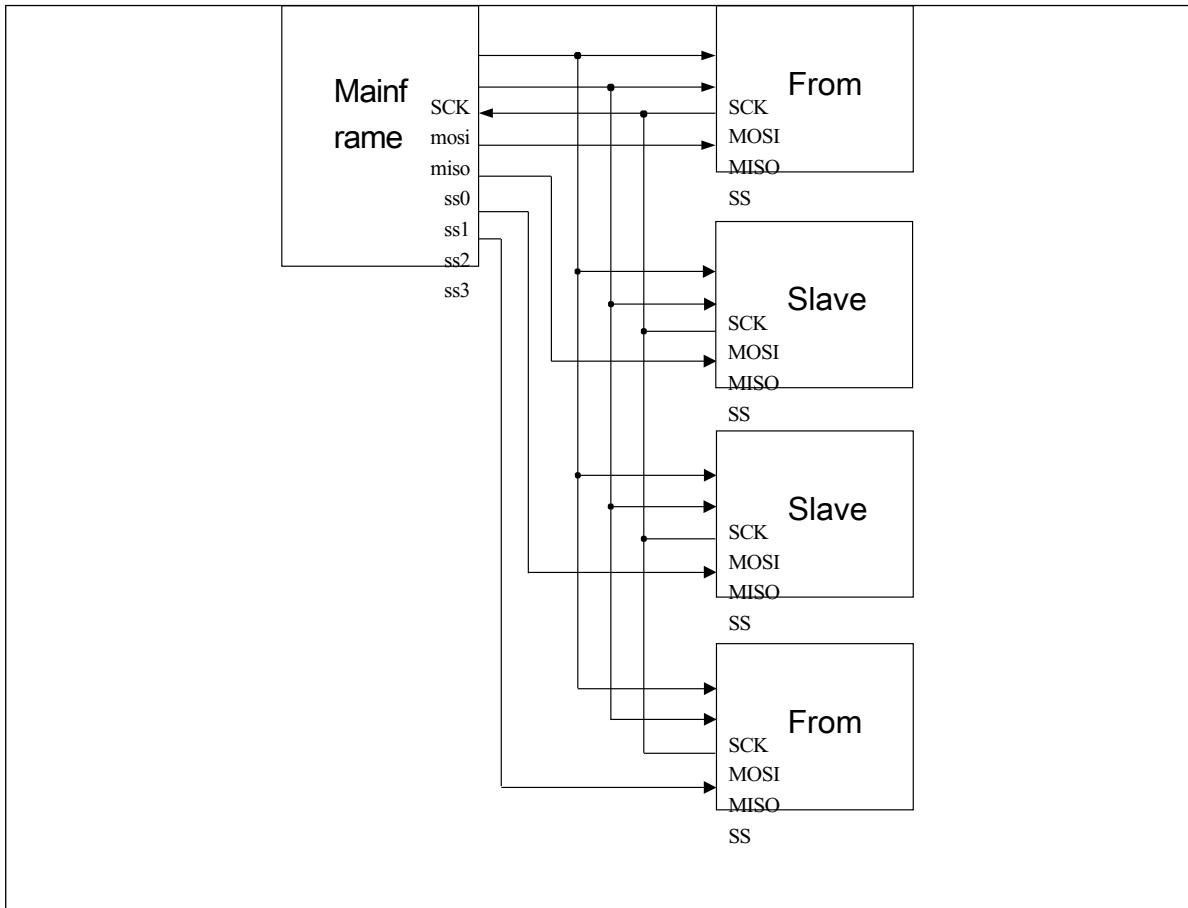


Figure 27-2 Host Mode Structure

Clock

synchronize

d operation

In an **SPI** system architecture used as a clock synchronization operation, the host device drives SCK and MOSI, and the slave device drives MISO. The SS pin is not used.

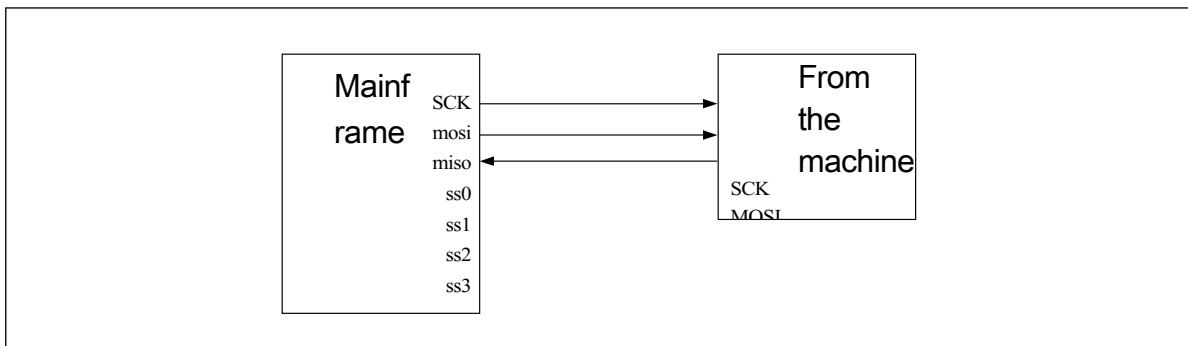


Figure 27-3 Three-Wire Clock Synchronization Operation

27.5 Description of data communication

27.5.1 Baud rate

In host mode, the **SPI** clock is provided by the internal baud rate generator; in slave mode, the clock is input from the SCK pin.

The baud rate depends on the SPI_CFG2.MBR[2:0] bit setting. It is calculated as shown in the following formula, where N is the setting of MBR[2:0] bits in the range of 0 to 7.

$$\text{Baud rate} = \frac{\text{PCLK}}{2^N + 1}$$

Table 27-5 Partial setpoint lower speed

MBR[2:0] bits of the Set value	Crossover Ratio	Baud rate			
		PCLK=5MHz	PCLK=10MHz	PCLK=20MHz	PCLK=40MHz
0	2	2.50Mbps	5.00Mbps	10.0Mbps	20.0Mbps
1	4	1.25Mbps	2.50Mbps	5.00Mbps	10.0Mbps
2	8	625kbps	1.25Mbps	2.50Mbps	5.00Mbps
3	16	313kbps	625kbps	1.25Mbps	2.50Mbps
4	32	156kbps	313kbps	625kbps	1.25Mbps
5	64	78kbps	156kbps	313kbps	625kbps
6	128	39kbps	78kbps	156kbps	313kbps
7	256	20kbps	39kbps	78kbps	156kbps

27.5.2 Data Format

The data format of the **SPI** depends on the value of the parity permission bit PAE in **SPI** command register SPI_CFG2 and **SPI** control register SPI_CR1. The **SPI** treats the data of a certain length (the data length is set by the DSIZE[3:0] bits in register SPI_CFG2) from the LSB bits in data register SPI_DR as the object of transfer. The data is processed regardless of the MSB/LSB shift order.

DSIZE[3:0] determines the bit width of the data, which ranges from 4 to 32 bits, and SPI_CR1.PAE determines the last bit of the data, which is used as the parity bit when PAE is 1 and the lowest bit of the data itself when it is 0. As shown in Figure 28-4.

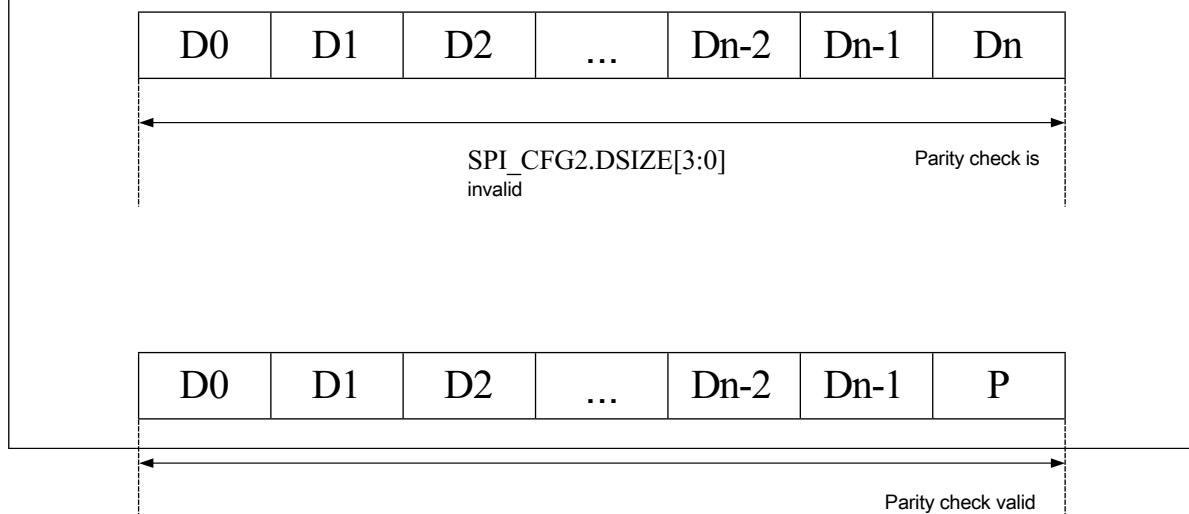


Figure 27-4 Data Format

When SPI data is sent, the transmitted data enters the transmit buffer (TX_BUFF), and then copies the data from TX_BUFF to the shift register (shifter) and the shifter sends out the data in turn; when SPI data is received, the data is shifted in from the shifter in turn, and then copies the data from the shifter to the receive buffer (RX_BUFF) after the shift is completed.

During data transfer, there are four cases depending on the setting of the shift order control bit SPI_CFG2.LSBF and the parity control bit SPI_CR1.PAE:

- 1) MSB pass first, parity invalid

When sending, data d31~d0 are copied from TX_BUFF to the shifter in order, and out of the highest shift of the shifter in the order of d31~d0;
When receiving, the data d31~d0 is shifted in from the lowest shift of the shifter, and then copied to RX_BUFF after all the data is shifted in.

2) LSB first, parity invalid

When sending, data d31~d0 are copied from TX_BUFF to **shifter** in the order of d0~d31 ~~but~~ of the highest shift of **shifter** in the order of d0~d31;

When receiving, the data d0~d31 is **shifted in** from the lowest shift of the shifter, and when all the data is shifted in, the data is shifted in accordance with d31~d0

The order is copied from shifter to RX_BUFF.

3) MSB first, when parity is valid

When sending, first calculate the value of the parity bit P based on the values of d31~d1, then replace d 0 with Pcopy it to the shifter in the order of d31~d1, P, and shift out from the highest bit of the shifter in the order of d31~P; When receiving, the data d31~P is shifted in from the lowest bit of the shifter, and parity checking is performed while the data is copied to the shifter. Finally, the data is copied to RX_BUFF again.

4) LSB first, when parity is valid

When sending, first calculate the value of parity bit P based on the values of d30~d0, then replace d31 with P, copy from TX_BUFF to shifter in the order of d0~P, and shift out from the highest bit of shifter in the order of d0~P; when receiving, data d0~P shift in from the lowest bit of shifter, and when data is copied to shifter Parity check is performed when the data is copied to the shifter. The data d0~P are rearranged during copying and copied to RX_BUFF in the order of P~d0.

27.5.3 Transmission format

1) The case of CPHA=0

When the SPI_CFG2.CPHA bit is "0", the SPI samples data on the odd edge of SCK and updates data on the even edge. Figure 27-5 shows the SPI transmission timing diagram when CPHA=0. When the input level of the SS signal becomes active, MOSI/MISO starts to update the transferred data. The first data sample is taken at the first SCK signal edge after the SS signal becomes active, and after that, the data is sampled every SCK cycle. The setting of the CPOL bit does not affect the timing of the SCK signal, but only the polarity of the signal.

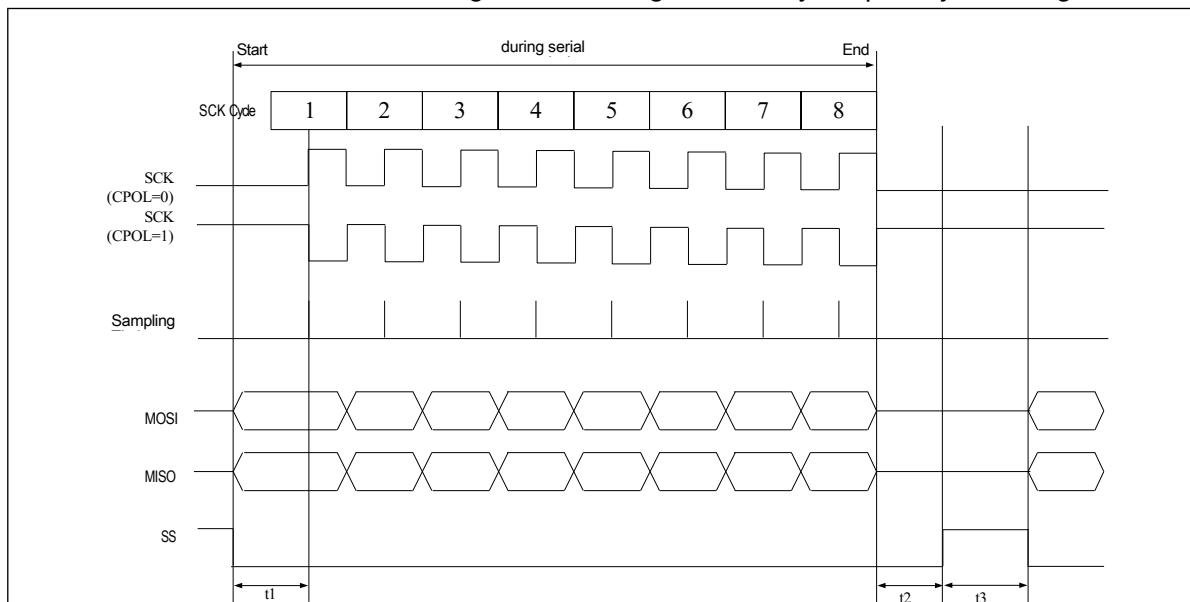


Figure 27-5 Data transfer format diagram (CPHA=0)

In the above figure, t1 indicates the interval from SSI signal valid to SCK oscillation (SCK delay time) t2 indicates the interval from SCK

The interval between when the oscillation stops and when the SS signal becomes invalid (SS invalidation delay time) and t3 indicates the minimum wait time between the end of the serial transmission and the start of the next transmission (next access delay) t1, t2, and t3 are controlled by the host device on the **SPI** system. Refer to Section 27.6.2 for details.

2) The case of CPHA=1

When the SPI_CFG2.CPHA bit is "1", the **SPI** performs data updates on the odd edges of SCK and data sampling on the even edges. Figure 27-6 shows the **SPI** transmission timing when CPHA=1. MOSI/MISO begins transmitting data updates on the first SCK signal edge after the **SSI** signal goes active. After that, the data is updated every SCK cycle. The setting of the SPI_CFG2.CPOL bit does

not affect the timing of the SCK signal but only the polarity of the signal.

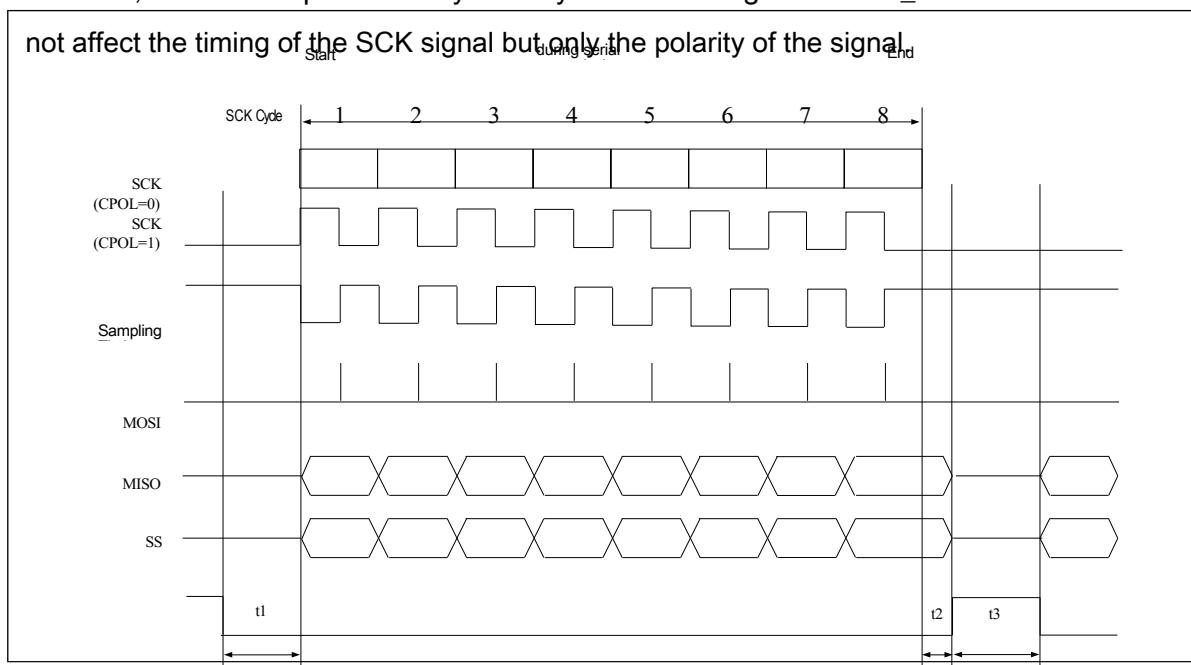


Figure 27-6 Data Transfer Format (CPHA=1)

The same for t1, t2, t3 and CPHA bit 0.

27.5.4 Communication method

This SPI has two communication modes, full-duplex synchronous serial communication and transmit-only serial communication, which can be selected by the TXMDS bit in the **SPI** control register (SPI_CR1).

1) Full duplex synchronous serial communication method

When the SPI_CR1.TXMDS bit is "0", the **SPI** is running in full-duplex synchronous serial communication mode. As shown in Figure 27-7, the SPI_CFG1.FTHLV[1:0] bit is "00b" and the SPI_CFG2.CPHA bit is "1" and

and

The SPI performs an 8-bit serial transfer when the SPI_CFG2.CPOL bit is "0".

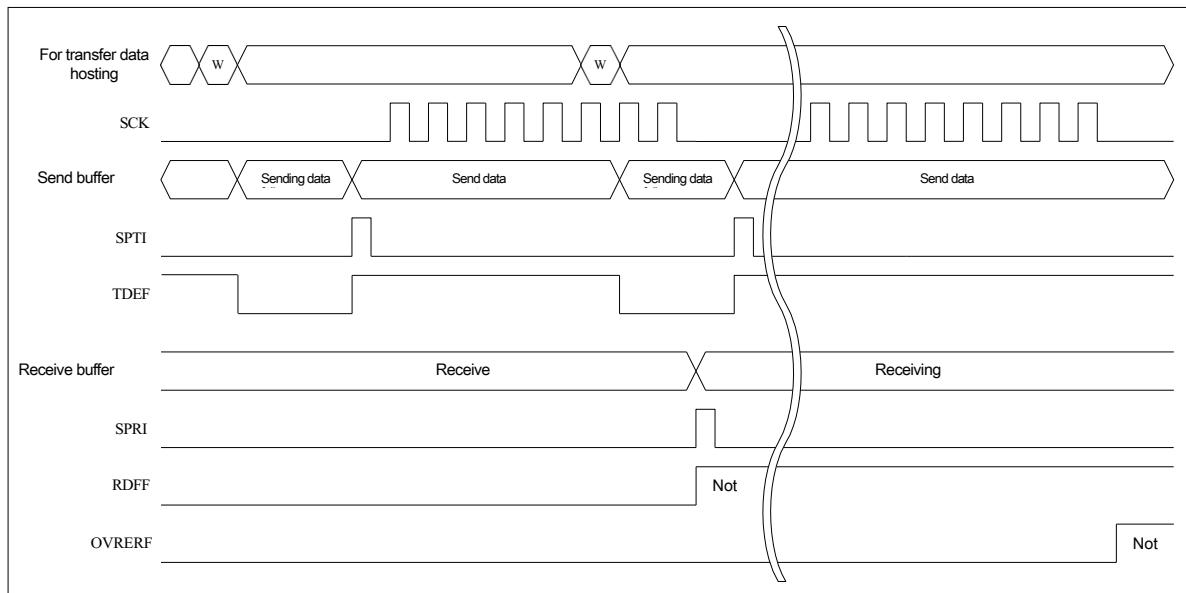


Figure 27-7 Full Duplex Synchronous Serial Communication

Notes:

1. At the end of this serial transmission, if the receive data buffer register is empty, the **SPI** will copy the received data from the shift register to the receive data buffer register, the receive data buffer register full flag bit is set to 1 (RDFF) and a receive data full interrupt request (SPRI) generated.
2. When this serial transmission ends, if the last received data is still held in the receive data buffer register and not read by the system, the **SPI** will set the data overload flag to 1, this data reception is invalid, and the data in the receive shift register will be discarded.

2) Sending communication method only

When the SPI_CR1.TXMDS bit is "1" the SPI is running in transmit-only communication mode. As shown in Figure 27-8, SPI_CFG1.FTHLV[1:0] bits are "00b" and SPI_CFG2.CPHA bit is "1" and The SPI performs an 8-bit serial transfer when the SPI_CFG2.CPOL bit is "0".

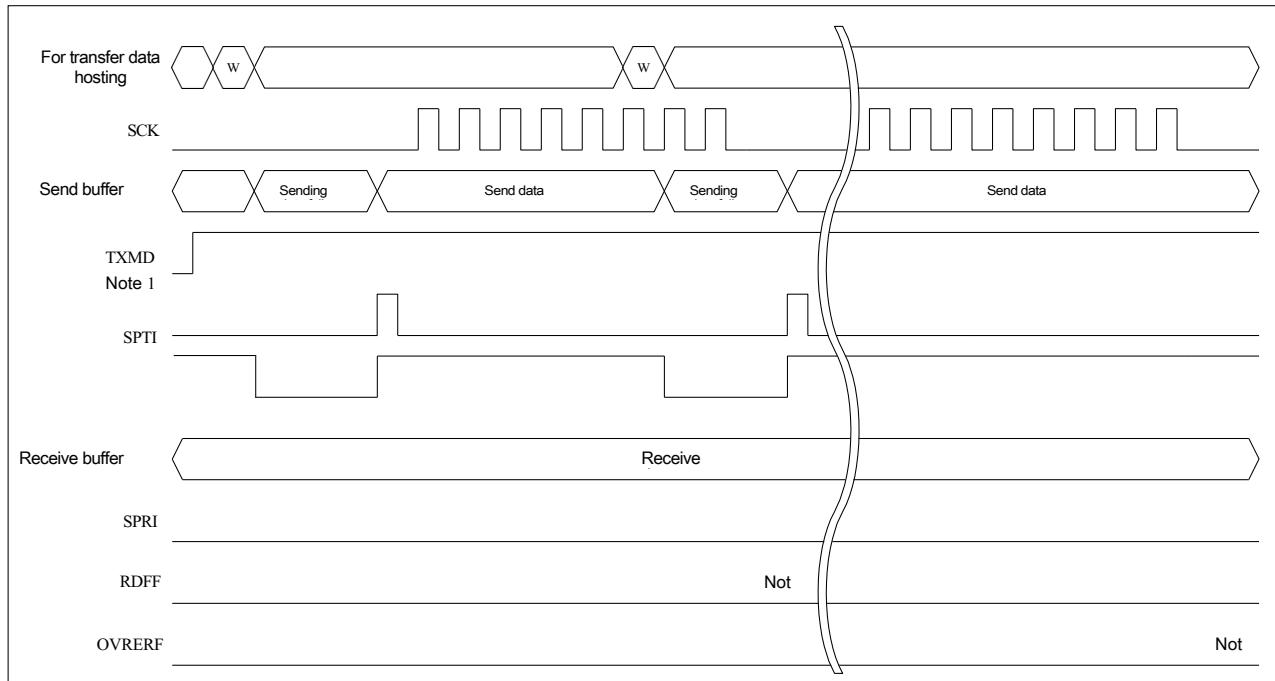


Figure 27-8 Transmit-only communication

Notes:

1. Before setting into the transmit-only communication mode, make sure that there is no unread data in the receive buffer register (i.e., the RDFF is 0) and no data overload errors (i.e., OVRERF is 0)
2. In the transmit-only communication method, when this serial transmission ends, no data will be accepted even if the receive data buffer register is empty, and the RDFF will always remain at 0.
3. In the transmit-only communication mode, no data overload error occurs because the receive data buffer register is always empty, and the OVRERF flag bit is always kept at 0.

27.6 Running Instructions

27.6.1 Outline of operation mode

This **SPI** supports 4-wire **SPI** mode and 3-wire clock synchronous operation mode.

Each mode of operation can be used for serial communication as a host or a slave. Set the MSTR and SPIMDS bits in **SPI** control register SPI_CR1 to set the mode of the SPI.

Table 27-6 SPI Mode and Register Setting Relationships

Mode	Mainframe (SPI operation)	From the machine (SPI operation)	Mainframe (clock synchronized operation)	From the machine (clock synchronized operation)
Setting of MSTR bit	1	0	1	0
Setting of SPIMDS bit	0	0	1	1
SCK signal	Output	Input	Output	Input
MOSI signal	Output	Input	Output	Input
MISO signal	Input	Output / Hi-Z	Input	Output
SS0 signal	Output	Input	Hi-Z (not used)	Hi-Z (not used)
SS1~SS3 signals	Output	Hi-Z	Hi-Z (not used)	Hi-Z (not used)
SS polarity change function	There are	There are	–	–
Maximum transmission rate	~PCLK/2	~PCLK/6	~PCLK/2	~PCLK/6
Clock source	Internal baud rate generator	SCK input	Internal baud rate generator	SCK input
Clock polarity	2 kinds	2 kinds	2 kinds	2 kinds
Clock Phases	2 kinds	2 kinds	2 kinds	1 kind (CHPA=1)
Start transmission bit	MSB/LSB	MSB/LSB	MSB/LSB	MSB/LSB
Transmitted data length	4~32 bits	4~32 bits	4~32 bits	4~32 bits
SCK delay control	There are	None	There are	None
SS invalid delay control	There are	None	There are	None
Next access delay control	There are	None	There are	None
Transmission start method	Write send buffer by sending buffer null interrupt request	SS input valid or SCK clock edge	Write send buffer by sending buffer null interrupt request	SCK Oscillation
Send buffer empty detection	There are	There are	There are	There are

Receive buffer full detection	Yes (Note 1)	Yes (Note 1)	Yes (Note 1)	Yes (Note 1)
Overload error detection	Yes (Note 1)	Yes (Note 1)	Yes (Note 1)	Yes (Note 1)
Parity error detection	Yes (Note 1, Note 2)			
Underload error detection	There are	None	There are	None

Notes:

1. When the SPI_CR1.TXMDS bit is "1", receive buffer full detection, overload error detection, and parity error detection are not performed.
2. When the SPI_CR1.PAE bit is "0", parity error detection is not performed.

27.6.2 Host action during SPI operation mode

1) Description of action when acting as a host

If the SPI data transmit buffer register (TX_BUFF) is empty (the TDEF flag bit in the status register SPI_SR is 1) the **SPI** will update the SPI_DR data to TX_BUFF after writing the data of the frame length set by the FTHLV[1:0] bits of the format control register SPI_CFG1 to the **SPI** data register (SPI_DR). If the shift register (**shifter**) is empty at this point, the **SPI** copies the data from TX_BUFF to the shift register to start the serial transfer.

When the transmit data is copied to the shifter, the **SPI** changes the shifter state to full; when the serial transfer ends, it changes to empty. The shifter state cannot be read.

This serial transfer ends when the SPI has sent the SCK edge required for the final sample timing, independent of the SPI_CFG2_CPHA bit. If the receive buffer (RX_BUFF) is empty, the **SPI** copies the data in the **shifter** to RX_BUFF after the serial transfer ends, which can be read through the data register SPI_DR.

The final sampling sequence depends on the bit length of the transmitted data. The SPI data length of the master mode depends on the setting of the SPI_CFG2_DSIZE[3:0] bits, and the polarity of the SS output pins depends on the setting of the SPI_CFG1 register. For details on the SPI transfer format, refer to 27.5.3 Transfer Format.

2) Initialization of **SPI** host mode

- ① Set the communication configuration register 1 (SPI_CFG1) including the baud rate setting, frame rate setting, and various delay time settings.
- ② Set the communication configuration register 2 (SPI_CFG2), including SS level setting, data shift order setting, various delay allowable bits setting, data format and clock polarity phase setting, etc.
- ③ If you need to use interrupt, please set the interrupt register of the system.
- ④ If you need to use DMA, please set the relevant registers of DMA.
- ⑤ Set the input and output pins.
- (6) Set **SPI** control register SPI_CR1, including the setting of mode and operation mode, setting of self-diagnostic function, setting of parity, etc.
- (7) Confirm the setting of SPI_CR1 register.
 - ⑧ Clears various flag bits.
 - ⑨ Set the interrupt permit bit.
- ⑩ Set the SPE bit of control register SPI_CR1 to 1, and the action starts.

27.6.3 Slave action during SPI operation mode

1) Description of the action when SPI acts as a slave

When the SPI_CFG2.CPHA bit is 0, the SPI needs to start driving valid data to the MISO output signal if it detects that the SS0 input signal has changed to a valid level. Therefore, with the CPHA bit at 0, the SS0 input signal level is changed from invalid to

is the trigger signal that is effectively considered to start serial transmission.

When the CPHA bit is "1", if the SPI detects the first SCK edge while the SS0 input signal is at a valid level, it is necessary to start driving valid data to the MISO output signal. Therefore, **with the** CPHA bit "1", the first SCK edge with the SS0 signal at a valid level is considered the trigger signal to start serial transmission.

If the SPI detects the start of a serial transfer while the shifter is empty, it changes the shifter to full and cannot transfer data from TX_BUFF to the shifter during the serial transfer. If the **shifter** is full before the serial transfer is started, the **SPI** keeps **the shifter** full.

If the SPI detects the SCK edge of the last sample timing, this serial transfer ends, independent of the SPI_CFG2.CPHA bit. If RX_BUFF is empty, the SPI copies the **shifter**'s accept data to RX_BUFF after the serial transfer ends. This data can be read by accessing SPI_DR. The **SPI** changes the **shifter** to the empty state after the serial transfer ends, and this state is independent of the state of RX_BUFF.

A mode fault error occurs if the **SPI** detects an invalid SS0 input signal during a serial transfer.

The final sampling timing depends on the bit length of the transmitted data, the data length of the SPI in slave mode depends on the value set by the SPI_CFG2.DSIZE[3:0] bits, and the polarity of the SS0 input signal depends on the value set by the SPI_CFG1.SS0PV bits. For details on the SPI transfer format, refer to 27.5.3 Transfer Format.

Caution:

- When the SPI_CFG2.CPHA bit is "0", the SS0 input signal level is changed from invalid to valid as the trigger signal to start serial transmission. If the SS0 input signal is fixed as valid in a single slave mode structure, the SPI will not start serial transmission properly. Therefore, in a structure where the SS0 input signal is fixed as valid, the CPHA bit must be set to "1" **in order** for the **SPI in slave mode to** transmit and receive properly. If the CPHA bit needs to be set to "0", **the** SS0 input signal cannot be fixed.

2) Initialization of SPI Slave Mode

- ① Set the communication configuration register 1 (SPI_CFG1) which mainly includes the setting of the number of frames to be used.
- ② Set the communication configuration register 2(SPI_CFG2) including the transmission rate, data format and clock polarity phase settings.
- ③ If you need to use interrupt, please set the interrupt register of the system.
- ④ If you need to use DMA, please set the relevant registers of DMA.

-
- ⑤ Set the input and output pins.
 - (6) Set **SPI** control register SPI_CR1, including the setting of mode and operation mode, setting of self-diagnostic function, setting of parity, etc.
 - (7) Confirm the setting of SPI_CR1 register.
 - ⑧ Clears various flag bits.
 - ⑨ Set the interrupt permit bit.
 - ⑩ Set the SPE bit of control register SPI_CR1 to 1, and the action starts.

27.6.4 Host action in clock synchronous operation mode

When the SPI_MDS bit in SPI control register SPI_CR1 is 1, the **SPI** is in clock synchronous operation mode. In this mode of operation, the SPI uses only three pins, SCK, **MOSI**, and MISO, for communication, and the SSi pin is released for normal I/O functions.

Although the SSi pin is not used during clock-synchronous operation mode, the internal operation of the module is the same as the SPI operation mode. However, since there is no input from SSi, no mode fault error is detected.

1) Description of the action when SPI acts as a host

If the **SPI data** transmit buffer register (TX_BUFF) is empty (the TDEF flag bit in the status register SPI_SR is 0) the **SPI** will update the SPI_DR data to TX_BUFF after writing the data of the frame length set by the FTHLV[1:0] bits of the format control register SPI_CFG1 to the **SPI** data register (SPI_DR). If the shift register (**shifter**) is empty at this point, the SPI copies the data from TX_BUFF to the shift register to start the serial transfer.

When the transmit data is copied to the shifter, the **SPI** changes the shifter state to full; when the serial transfer ends, it changes to empty. the shifter state cannot be read.

This serial transfer ends when the SPI has sent the SCK edge required for the final sample timing, independent of the SPI_CFG2.CPHA bit. If the receive buffer (RX_BUFF) is empty, the **SPI** copies the data in the **shifter** to RX_BUFF after the serial transfer ends, which can be read through the data register SPI_DR.

The final sampling sequence depends on the bit length of the transmitted data, the SPI data length of the master mode depends on the setting of the SPI_CFG2.DSIZE[3:0] bits, and the polarity of the SS output pins depends on the setting of the SPI_CFG1 register. For details on the SPI transfer format, refer to 27.5.3 Transfer Format.

2) Initialization settings of the host during clock synchronous operation mode

- ① Set the communication configuration register 1 (SPI_CFG1) including the baud rate setting, frame rate setting, and various delay time settings, etc.
- ② Set the communication configuration register 2(SPI_CFG2), including the data shift order setting, the setting of various delay allowable bits, the data format and the clock polarity phase setting, etc.
- ③ If you need to use interrupt, please set the interrupt register of the system.
- ④ If you need to use DMA, please set the relevant registers of DMA.
- ⑤ Set the input and output pins.

-
- (6) Set SPI control register SPI_CR1, including the setting of mode and operation mode, setting of self-diagnostic function, setting of parity, etc.
 - (7) Confirm the setting of SPI_CR1 register.
 - Ⓐ Clears various flag bits.
 - ⑥ Set the interrupt permit bit.

- ⑩ Set the SPE bit of control register SPI_CR1 to 1, and the action starts.

27.6.5 Slave operation in clock synchronous operation mode

1) Description of the action when **SPI** acts as a slave

When the SPI_CFG2.CPHA bit is 0, the **SPI** needs to detect that the SS0 input signal becomes active as the trigger signal to start serial communication. Since the SS0 pin is not used in clock-synchronous operation mode, normal communication is not possible when the CPHA bit is 0.

When the CPHA bit is "1", if the SPI detects the first SCK edge while the SS0 input signal is active, it needs to start signaling the MISO output to drive the active data. Since the SS0 pin is not available in clock synchronous operation mode, the first SCK edge is considered the trigger signal to start serial transmission **when the CPHA** bit is "1".

If the **SPI** detects the start of a serial transfer while the shifter is empty, it changes the **shifter** to full and cannot transfer data from TX_BUFF to the shifter during the serial transfer. If the **shifter** is full before the serial transfer is started, the **SPI** keeps the **shifter** full.

If the **SPI** detects the SCK edge of the last sample timing, this serial transmission ends. If RX_BUFF is empty, the **SPI** copies the received data from the **shifter** to RX_BUFF at the end of the serial transfer, which can be read by accessing SPI_DR. The **SPI** changes the **shifter** to an empty state at the end of the serial transfer, which is independent of the state of RX_BUFF. The final sampling timing depends on the bit length of the transmitted data, and the data length of the slave mode SPI depends on the value set by the SPI_CFG2.DSIZE[3:0] bits.

2) Initialization settings of the slave during clock synchronous operation mode

- ① Set the communication configuration register 1 (SPI_CFG1) which mainly includes the setting of the number of frames to be used.
- ② Set the communication configuration register 2(SPI_CFG2), including the transmission rate, data format and clock polarity phase settings.
- ③ If you need to use interrupt, please set the interrupt register of the system.
- ④ If you need to use DMA, please set the relevant registers of DMA.
- ⑤ Set the input and output pins.
- ⑥ Set SPI control register SPI_CR1, including the setting of mode and operation mode, setting of self-diagnostic function, setting of parity, etc.
- ⑦ Confirm the setting of SPI_CR1 register.
- ⑧ Clears various flag bits.

-
- ⑥ Set the interrupt permit bit.
 - ⑩ Set the SPE bit of control register SPI_CR1 to 1, and the action starts.

27.6.6 Processing flow of several SPI actions

1) Data transfer processing flow when **SPI** is the host

- ① Wait for an interrupt that the data send buffer register is empty or confirm that the data send buffer register is empty by polling.
- ② Write the data to be sent to the data register SPI_DR.
- ③ Repeat step ①② until the last data is sent.
- ④ Clear the allowable bit TXIE of the transmit data register air interrupt to zero and set the **SPI** idle state interrupt allowable bit IDIE to 1.
- ⑤ Send **SPI** idle state interrupt.
- ⑥ Set SPE to 0 to stop SPI operation and clear IDIE to zero at the same time. 2) Data reception processing flow

① Wait for the interrupt that the data receive buffer register is full or confirm that the data receive buffer register is in the full state by polling.

- ② Read data from the receive buffer register by accessing SPI_DR.
- ③ Repeat step ①② until the last received data is read.
- ④ Clear the interrupt allowable bit RXIE for the data receive buffer register full to zero. 3) Communication error handling process

① Waiting for a communication error interruption or confirming the communication error flag position by polling

- (MODFERF/OVRERF/UDRERF/PERF) is set to 1.
- ② Confirm SS0 status and troubleshoot mode fault errors.
 - ③ Clear SPE to stop **SPI** operation.
 - ④ Clear all SPI interrupt allow bits and block SPI interrupts.
 - ⑤ Determine the communication error type by the error flag bit and perform communication error processing.
 - ⑥ Clear the error flag bit to zero.
 - (7) Start **SPI** and restart communication.

27.7 Parity bit self-diagnosis

The parity check circuit consists of a parity check bit for transmitting data and an error detection part for receiving data. The parity check circuit can be troubleshooted using the self-diagnostic function according to the flow shown in the figure below.

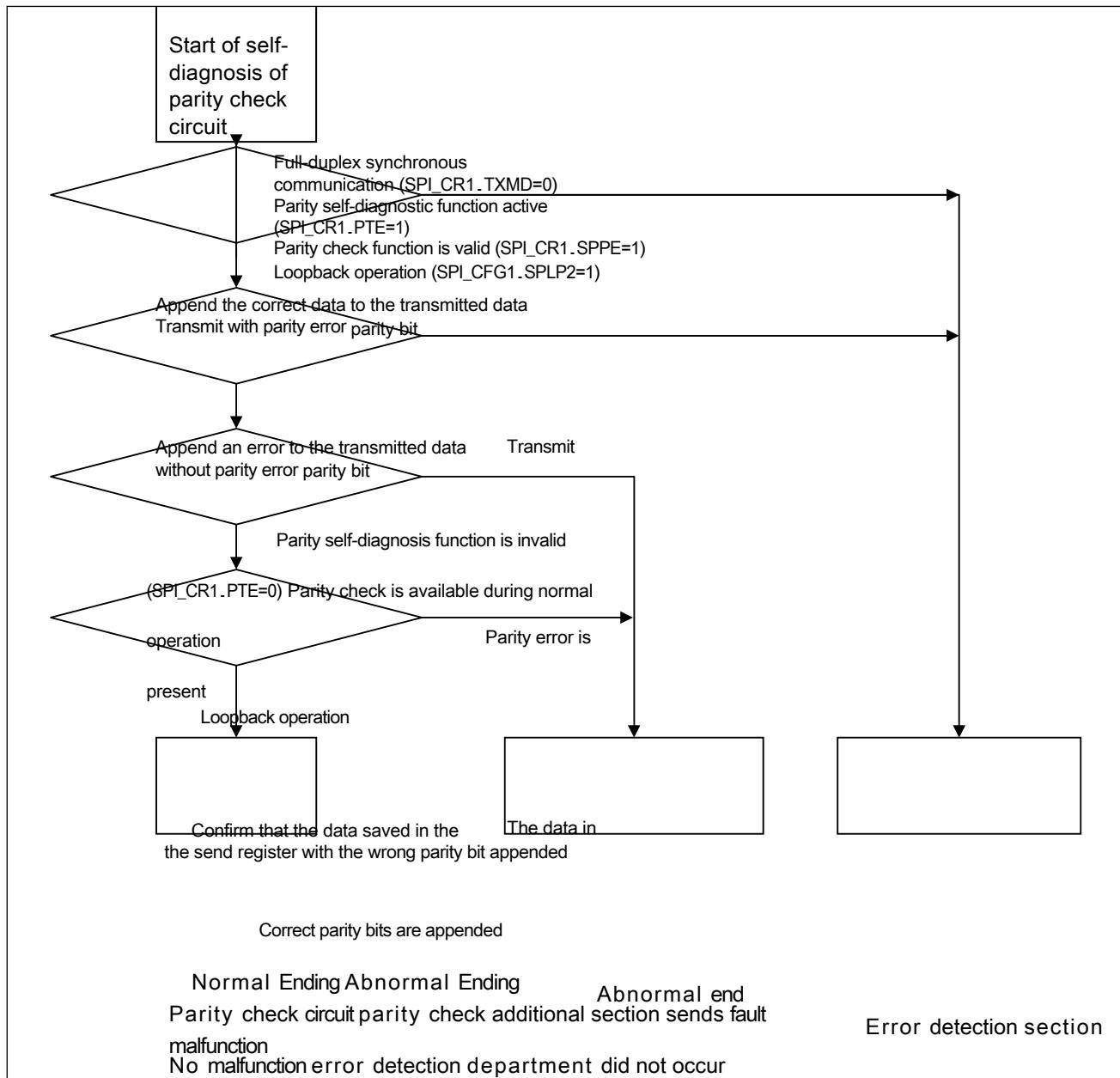


Figure 27-9 Parity Check Flow

27.8 Error detection

In a normal **SPI** serial transfer, the system sends data serially by writing to the SPI_DR register and obtains data received serially by reading from the SPI_DR register. However, due to the state of the transmit/receive buffers and the state of the **SPI at the** beginning/end of the serial transfer, abnormal transfers may occur in some cases. When an abnormal transfer occurs, the **SPI** detects the transfer as an underload error, an overload error, a parity error, or a mode fault error. The correspondence between abnormal transmission and SPI error detection is shown in Table 27-7 below.

Table 27-7 Error Detection Correspondence Table

Serial number	Occurrence conditions	SPI operation	Detection of errors
①	Write SPI_DR register in the transmit buffer full state	<ul style="list-style-type: none"> — Keep sending buffer contents — Write data loss 	None
②	Read SPI_DR register with empty receive buffer	Output the last serial reception data	
③	In slave mode, serial transmission starts in the state where the transmit data is not transferred to the shift register	<ul style="list-style-type: none"> — Aborting serial transmission — Loss of transmit and receive data -Stop driving the MISO output signal — Stop SPI function 	Underload error
④	Slave mode: The effective level width of SS0 pin does not reach the required time for data transmission.	<ul style="list-style-type: none"> - Aborting transmission - Loss of transmit and receive data - Stop SPI function 	
⑤	Ending serial transmission when the receive buffer is full	<ul style="list-style-type: none"> — Keeping the contents of the receive buffer — Receiving data loss 	
(6)	Incorrect parity bits are received when full-duplex synchronous serial communication is in progress and the parity function is active	Parity error flag valid	Parity error

In the case described in ①, no detection error occurs in SPI. To prevent data omission during writing to the SPI_DR register, the data must be written to the SPI_DR register by sending a buffer empty interrupt. Similarly, in the case of ②, **SPI** also does not generate detection errors. To prevent extraneous data from being read, data reads from SPI_DR must be performed by receiving a register-full interrupt request.

27.8.1 Underload error

When the MSTR bit is 0, the SPI operates in the slave state. If the **SPI is not**

ready to transmit data until the SS0 pin receives a valid level when SPE is set to 1, an underload error occurs in the **SPI** and the SPI_SR.MODFERF and SPI_SR.UDRERF flags are set to 1.

When an underload error is detected, the **SPI** will stop the drive signal output and set SPI_CR1.SPE to 0.

Monitoring for underload errors can be done by accessing the SPI_SR register directly, or by reading SPI_SR using an **SPI error interrupt**, etc. If you do not use the error interrupt, use the polling method to monitor for underload errors.

When SPI_SR.MODFERF is 1, the system disables writing 1 to the SPE bit. to set SPI_CR1.SPE to 1 to enable the SPI function you must first clear the MODFERF flag.

27.8.2 Mode Error

MODFERF is set to 1 and SPI_CR1.SPE is set to 0. When the SPI_SR.MODFREF is cleared to zero when transmission is required, then **SPI_CR1.SPE is set** to 1. MODFREF, then set SPI_CR1.SPE to 1.

27.8.3 Overload error

OVRERF flag is set to 1. Since **the SPI** does not copy the data from the shift register to the receive buffer when the OVRERF flag is 1, the receive buffer holds the receive data before the error occurs. To set the OVRERF flag to 0, the SPI_SR register must be read with the OVRERF flag at 1 before writing "0" to the OVRERF flag.

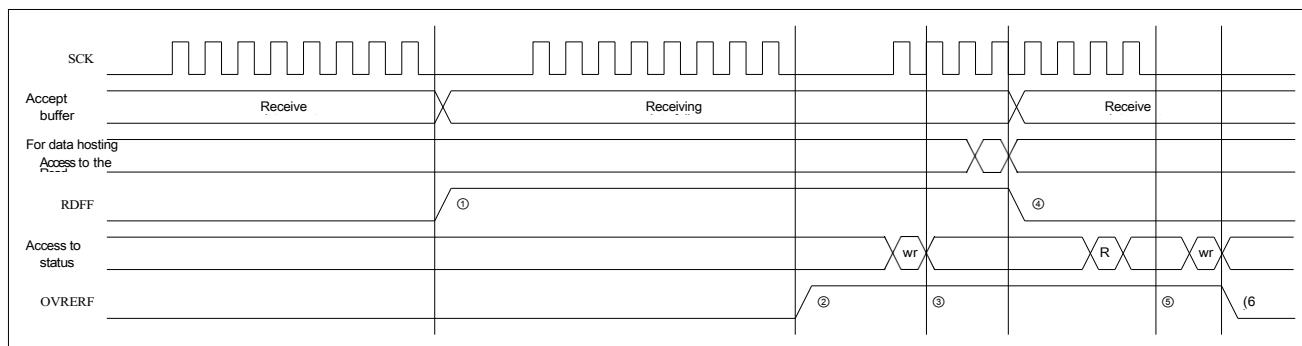


Figure 27-10 Overload Error Handling

The following illustrates the operation of the flags in the timing sequence shown in ① to ⑥ of the figure:

- ① End the serial transfer in the accept buffer empty state, the SPI operates normally, copies the data from the shift register to the accept buffer register, and sets the RDFF flag to 1.
- ② When the serial transfer ends with the receive buffer full, the SPI detects an overload error and sets the OVRERF flag to 1. The **SPI** will not copy data from the shift register to the receive buffer. Parity errors are not detected even **when the PAE bit is "1"**.
- ③ Without reading the SPI_SR register, the OVRERF bit cannot be written and the clearing fails.
- ④ RDFF flag becomes 0. The OVRERF flag does not become 0 even if the receive buffer state is empty at this time.
- ⑤ The SPI will not copy data from the shift register to the receive buffer register and will not generate a receive buffer full interrupt if the OVRERF flag is "1" (overload error), and the RDFF flag will remain at 0. Even if the PAE bit is "1" **Even if the PAE bit is "1"**, no parity error is detected. In the event of an overload error, if the serial transfer ends without copying the receive data from the shift register to the receive buffer, the **SPI** determines that the shift register is empty and allows the data to be transferred from the transmit buffer register to the shift register.
- ⑥ The SPI sets the OVRERF flag to 0 after reading the SPI_SR register with the OVRERF flag at 1 and then writing 0 to the OVRERF flag.

Monitoring for overload errors can be done either by accessing the SPI_SR register directly or by using the SPI error interrupt to access the SPI_SR register. During serial transfers, the occurrence of an overload error must be detected early by, for example, reading the SPI_SR register immediately after reading the SPI_DR register.

Normal receive operation is only possible after the OVRERF flag has been changed to 0.

If the communication auto-hang feature is enabled in host mode (SPI_CR1.CSUSPE bit set to 1) the SPI suspends the communication clock for the last sample period before an overload error occurs, at which point the SPI remains in a normal communication state and an overload error does not occur because the shift register has not yet completed receiving the last bit. During the communication clock pause, the receive buffer register can be read, and after the read, the receive buffer register status becomes empty and the SPI restarts the communication clock to complete the last bit of data reception. Refer to Figure 27-11 and Figure 27-12 below for detailed actions.

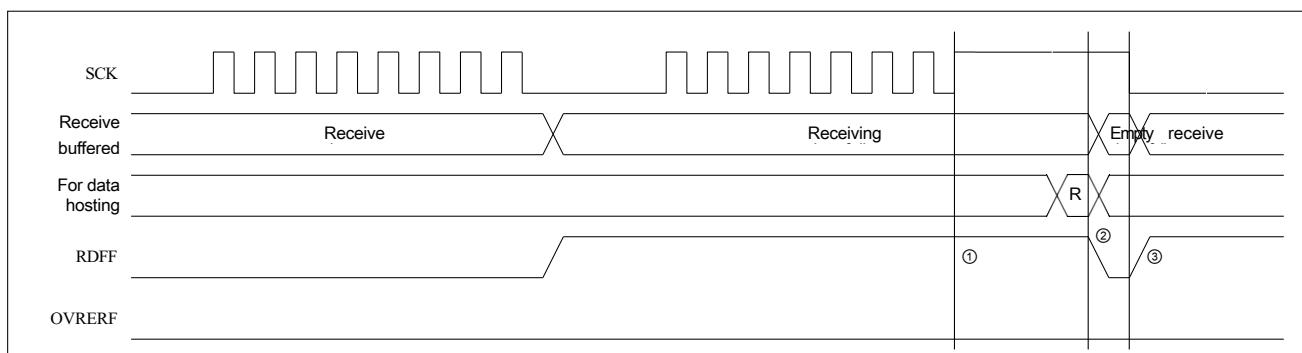


Figure 27-11 Diagram of the action when the clock auto-stop function is enabled (CPHA=1)

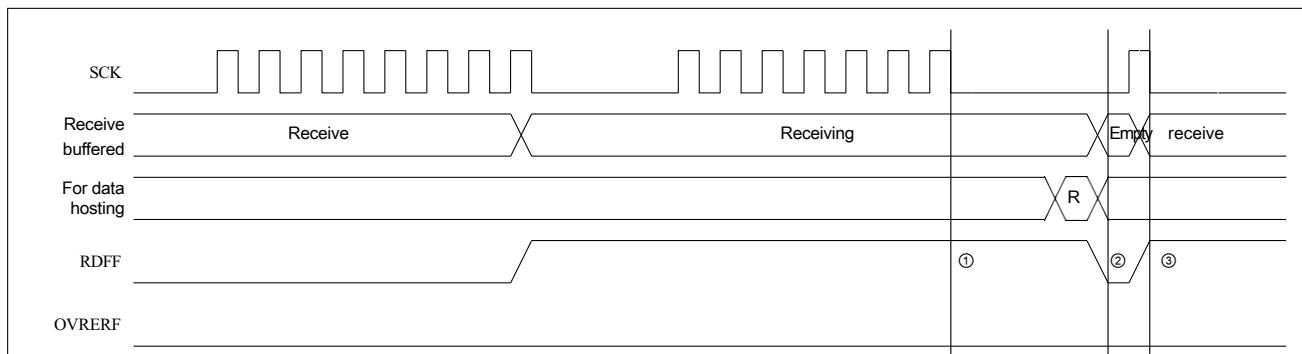


Figure 27-12 Diagram of the action when the clock auto-stop function is enabled (CPHA=0)

The following illustrates the operation of the flags in the timing sequence shown in ① to ③ of the figure:

- ① When the receive buffer register is full, the SPI suspends the communication clock before the last bit of data is received. No overload error will occur at this time.
- ② When the data in the receive buffer register is read by accessing SPI_DR, the receive buffer register becomes empty, the RDFF flag is cleared, and the SPI restarts the

communication clock to complete the last bit of data communication.

- ③ When the last bit of data communication is completed, the receive buffer register becomes full again, the RDFF flag is set to 1, and the received data can be read by accessing SPI_DR.

27.8.4 Parity error

TXMDS bit is "0" and SPI_CR1.PAE bit is "1", SPI will perform parity check at the end of full duplex synchronous serial communication. When the SPI detects a parity error in the received data, it sets the SPI_SR.PERF flag to 1. In the SPI_SR.OVRERF bit "1" state, the SPI does not detect a parity error in the received data because it does not copy the shift register data to the receive buffer. To clear the PERF flag, read the SPI_SR register with the PERF flag at 1 and then write 0 to the PERF flag.

An example of the operation of the OVRERF flag and PERF flag is shown in Figure 27-13 below. In the example in the figure, the SPI performs an 8-bit serial transfer for full-duplex synchronous serial communication with the SPI_CR1.TXMDS bit at 0 and the SPI_CR1.PAE bit at 1.

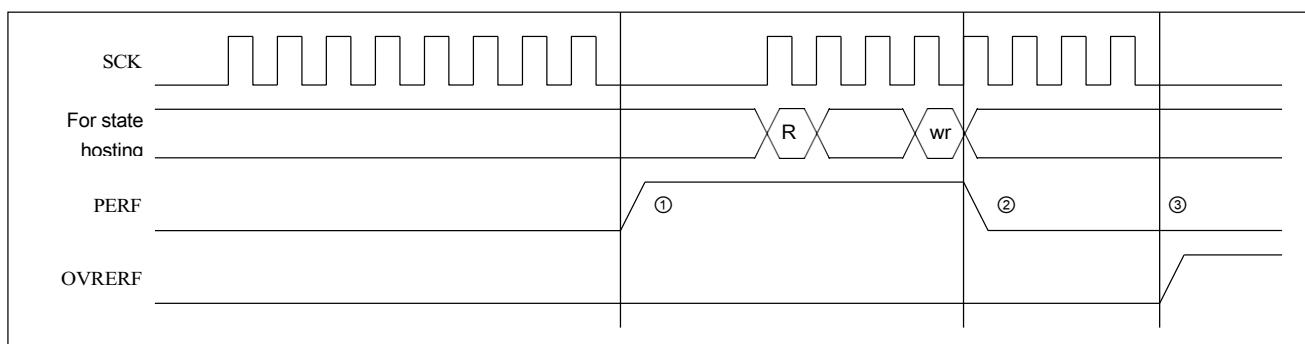


Figure 27-13 Parity error

The following illustrates the operation of the flags in the timing sequence shown in ① to ③ in the figure:

- ① The SPI does not detect an overload error and the serial transfer ends normally. the SPI copies the data from the shift register to the receive buffer. At this point, the SPI performs parity check on the received data. If a parity error is detected, the PERF flag is set to 1.
- ② After reading the SPI_SR register with PERF flag 1, write 0 to the PERF flag and clear the RERF flag to zero.
- ③ SPI detects an overload error, at this time SPI will not copy the data from the shift register to the receive buffer, SPI will not parity the data, and no parity error will occur.

The occurrence of a parity error can be monitored by accessing the SPI_SR register directly or by reading the SPI_SR register via the SPI error interrupt. When performing serial transfers, the occurrence of parity errors must be monitored as early as possible by, for example, accessing the status register SPI_SR.

27.9 Initialization of SPI

The SPI function can be disabled and some **SPI functions** initialized by clearing the SPE bit through a write operation or mode fault error detection. If a system reset occurs, the full **SPI function is** initialized.

27.9.1 Clear SPE bit for initialization

When the SPI_CR1.SPE bit is 0, the SPI performs the following initialization operations:

- Aborts an ongoing serial transmission.
- Stop driving the output signal if it is in the slave state (the state changes to Hi-Z)
- Initializes the internal state of the **SPI**.
- The transmit buffer register is cleared and the SPI_SR.TDEF flag is set to 1.

When initializing by clearing the SPE bit to zero, the SPI's control bits are not initialized. Therefore, by resetting the SPE position to 1, the **SPI** can be started in the same transfer mode as before initialization.

Clearing the SPE bit does not initialize the error flag bit and sequence state. Therefore, it is possible to confirm the occurrence of an error during **SPI** transmission by reading the data in the receive buffer even after the SPE is cleared.

Since clearing the SPE bit clears the transmit buffer register and sets the SPI_SR.TDEF flag to 1. Therefore, if the SPI_CR1.TXIE bit is set to 1 after initialization, an interrupt with an empty SPI transmit buffer register is generated. To avoid this interrupt, you must set the TXIE bit to 0 at the same time you clear the SPE bit.

27.9.2 System reset initialization

Initialization via system reset will initialize all control bits, status bits, and data registers of the SPI.

27.10 Interrupt source

The **SPI** interrupt sources are receive buffer full, transmit buffer empty, mode fault, overload, underload, parity error, and SPI idle. The receive buffer full and transmit buffer empty interrupts can be used to initiate DMA for data transfer.

Interrupts for overload, underload, and parity errors are set as SPI error interrupts SPEI, so it is necessary to determine the actual source of the interrupt occurring by flags. the SPI interrupt sources are specified as shown in Table 27-8. Once the interrupt condition is established, the corresponding interrupt request is generated.

For interrupt sources where the receive buffer is full and the transmit buffer is empty, you need to clear them by changing the buffer state through data transfer. When using DMA for transmit or receive, the DMA must be configured first, and the **SPI** must be set after the DMA is set to the action-permitted state.

Table 27-8 SPI Interrupt Source Descriptions

Interrupt source	abbreviated as	Interrupt conditions	Start DMA
Receiver buffer full	SPRI	When the receive buffer becomes full while the SPI_CR1.RXIE bit is "1"	Can
Occurrence of buffer empty	SPTI	When the transmit buffer becomes empty in the state of SPI_CR1.TXIE bit "1"	Can
SPI errors (overload, underload, parity errors)	SPEI	OVRERF, SPI_SR.PERF or SPI_SR.MODFERF and SPI_SR.UDRERF flags change when the SPI_CR1.EIE bit is "1". When "1" is used	Cannot
SPI Idle Interrupt	SPIII	When the IDLF flag becomes "0" in the state of "1" in SPI_CR1.IDIE bit	Cannot

27.11 Available event trigger sources

The main event trigger sources generated by **SPI that** can be used are as follows:

- Data send buffer register empty
- Data receive buffer register full
- **SPI** communication errors (including overload, underload, parity errors, etc.)
- **SPI** in idle state
- End of **SPI** communication

Users can write the vectors corresponding to the above event trigger sources into different

trigger object registers to achieve various event trigger functions. Please refer to [Interrupt

Controller (INTC) for the vectors corresponding to the above event trigger sources.

27.12 Register Description

Register List

Register base address: SPI1_BASE:0x4001_C000.

SPI2_BASE:0x4001_C400 SPI3_BASE:0x4002_0000.

Register Name	Offset Address	Reset value
SPI data register SPI_DR	0x00	0x0000_0000
SPI control register SPI_CR1	0x04	0x0000_0000
SPI communication configuration register 1 SPI_CFG1	0x0C	0x0000_0010

27.12.1 SPI Data Register (SPI_DR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
SPD [31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SPD [15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b0	SPD [31:0]	Serial Data	SPI Data Storage	R/W

27.12.2 SPI control register (SPI_CR1)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PAE	PAOE	PATE	MODFE	IDIE	RXIE	TXIE	EIE	CSUSPE	SPE	SPLPBK2	SPLPBK	MSTR	-	TXMDS	SPIMDS

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Read "0", write "0" when writing	R/W
b15	PAE	Parity check allowed	0: send data without parity bit, receive data without parity check 1: Send data with additional parity bits, receive data for parity check (SPI_CR1.TXMDS=0); send data with additional parity bits, receive data without parity (SPI_CR1.TXMDS=1)	R/W
b14	PAOE	Parity check mode selection	0: Select even parity for transmitting and receiving 1: Select odd checksum for sending and receiving	R/W
b13	PATE	Parity check self-diagnosis	0: Parity check self-diagnosis function is invalid 1: Parity check self-diagnosis function is effective	R/W
b12	MODFE	Mode fault error detection allows	0: Disable mode fault error detection 1: Allow mode fault error detection	R/W
b11	IDIE	SPI idle interrupt allowed	0: Disable idle interrupt request generation 1: Allow idle interrupt requests to be generated	R/W
b10	RXIE	SPI receive interrupt allowed	0: SPI receive interrupt request generation is disabled 1: Allow SPI receive interrupt request generation	R/W
b9	TXIE	SPI transmit interrupt allowed	0: SPI transmit interrupt request generation is disabled 1: Allow SPI to send interrupt request generation	R/W
b8	EIE	SPI error interrupt allowed	0: SPI error interrupt request generation is disabled 1: Allow SPI error interrupt request generation	R/W
b7	CSUSPE	Communication auto-hang function allows	0: Communication auto-hang function is invalid 1: The communication auto-hang function is effective	R/W
b6	SPE	SPI function allows	0: SPI function is invalid 1: SPI function is valid	R/W
b5	SPLPBK2	SPI loopback 2-bit	0: Normal mode 1: Loopback mode (send data = receive data)	R/W
b4	SPLPBK	SPI loopback bit	0: Normal mode 1: Loopback mode (Inverted phase of sent data = received data)	R/W
b3	MSTR	SPI Master-Slave Mode Selection	0: Slave mode 1: Host mode	R/W
b2	Reserved	-	Read "0", write "0" when writing	R/W
b1	TXMDS	Communication mode selection	0: Full duplex synchronous serial communication 1: Transmit serial communication only	R/W
b0	SPIMDS	SPI Mode Selection	0: SPI operation (4-wire type) 1: Clock synchronous operation (3-wire type)	R/W

27.12.3 SPI communication configuration register 1 (SPI_CFG1)

Reset value: 0x0000_0010

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	MIDI[2:0]			-	MSSDL[2:0]			-	MSSI[2:0]			-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	SS3 PV	SS2 PV	SS1 PV	SS0 PV	-	SPR DTD	-	-	-	-	FTHLV[1:0]]	

position	Marker	Place Name	Function	Reading and writing
b31	Reserved	-	Read "0", write "0" when writing	R/W
b30~b28	MIDI[2:0]	Host next access	b30~b28	R/W
		data interval idle	0 0 0 0: 1 SCK + 2 PCLK	
		time set position	0 0 1: 2 SCK + 2 PCLK	
			0 1 0: 3 SCK + 2 PCLK	
			0 1 1: 4 SCKs + 2 PCLKs	
			1 0 0 : 5 SCK + 2 PCLK	
			1 0 1: 6 SCK + 2 PCLK	
			1 1 0: 7 SCK + 2 PCLK	
			1 1 1: 8 SCKs + 2 PCLKs	
b27	Reserved	-	Read "0", write "0" when writing	R/W
b26~b24	MSSDL[2:0]	Host SS invalid delay	b26~b24	R/W
		set position	0 0 0 0: 1 SCK	
			0 0 1: 2 SCKs	
			0 1 0: 3 SCKs	
			0 1 1: 4 SCKs	
			1 0 0 : 5 SCKs	
			1 0 1: 6 SCKs	
			1 1 0: 7 SCKs	
			1 1 1: 8 SCKs	
b23	Reserved	-	Read "0", write "0" when writing	R/W
b22~b20	MSSI[2:0]	Host SS idle time set	b22~b20	R/W
		position	0 0 0 0: 1 SCK	
			0 0 1: 2 SCKs	
			0 1 0: 3 SCKs	
			0 1 1: 4 SCKs	
			1 0 0 : 5 SCKs	
			1 0 1: 6 SCKs	
			1 1 0: 7 SCKs	
			1 1 1: 8 SCKs	
b19~b12	Reserved	-	Read "0", write "0" when writing	R/W
b11	SS3PV	SS3	0: Low level of SS3 signal is valid	R/W
		signal	1: High level of SS3 signal is valid	
		polarity		
		setting		
b10	SS2PV	SS2	0: Low level of SS2 signal is valid	R/W
		signal	1: High level of SS2 signal is valid	
		polarity		
		setting		

b9	SS1PV	SS1 signal polarity setting	0: Low level of SS1 signal is valid 1: High level of SS1 signal is valid	R/W
----	-------	--------------------------------------	---	-----

b8	SS0PV	SS0 signal polarity setting	0: Low level of SS0 signal is valid 1: High level of SS0 signal is valid	R/W
b7	Reserved	-	Read "0", write "0" when writing	R/W
b6	SPRDTD	Data register read object selection	0: SPI_DR read receive buffer 1: SPI_DR Read transmit buffer (must be read when TDEF=1)	R/W
b5	Reserved	-	Read "0", write "0" when writing	R/W
b4	Reserved	-	Read "0", write "1" when writing	R/W
b3~b2	Reserved	-	Read "0", write "0" when writing	R/W
b1~b0	FTHLV[1:0]	Frame rate set positioning	b1~b0 0 0: 1 frame 0 1:2 frames 1 0:3 frames 1 1:4 frames	R/W

27.12.4 SPI Status Register (SPI_SR)

Reset value: 0x0000_0020

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	RDF F	-	TDE F	UDR ERF	PER F	MOD FER F	IDL NF	OVR ERF

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Read "1", write "0" when writing	R/W
b7	RDFF	Receive buffer full flag	0: No data in SPI_DR register 1: SPI_DR register has data Hardware set, clear, and write "1" when writing	R
b6	Reserved	-	Read "1", write "0" when writing	R/W
b5	TDEF	Send buffer empty flag	0: Send buffer with data 1: No data in the occurrence buffer Hardware set, clear, and write "1" when writing	R
b4	UDRERF	Underload error flag	0: Mode fault error occurs (MODFERF=1) 1: Underload error occurs (MODFERF=1) When MODFERF=0, this bit is initialized After hardware set, read 1 write 0, status bit clear	R/W
b3	PERF	Parity error flag	0: No parity error occurred 1: Parity error occurred After hardware set, read 1 write 0, status bit clear	R/W
b2	MODFERF	Mode fault error flag	0: Failure to send mode fault error 1: A mode fault error occurred After hardware set, read 1 write 0, status bit clear	R/W
b1	IDLNF	SPI idle flag	0: SPI is idle 1: SPI is the transmission status hardware set, clear zero	R
b0	OVRERF	Overload error flag	0: No overload error occurred 1: Overload error occurred After hardware set, read 1 write 0, status bit clear	R/W

27.12.5 SPI communication configuration register 2 (SPI_CFG2)

Reset value: 0x0000_0F1D

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
MSS IE	MSS DLE	MID IE	LSB F	DSIZE[3:0]				SSA[2:0]			MBR[2:0]			CPO L	CPH A

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Read "0", write "0" when writing	R/W
b15	MSSIE	SCK delay allowed	0: SCK delayed by 1 SCK 1: SCK Delay is the set value of MSSI	R/W
b14	MSSDLE	SS invalid delay allowed	0: SS Invalid delay of 1 SCK 1: SS Invalid delay is the setting of MSSDL	R/W
b13	MIDIE	SPI next access delay allowed	0: Next access delay is 1 SCK + 2 PCLKs 1: Next access delay is the set value of MIDI	R/W
b12	LSBF	SPI LSB first bit	0: MSB first 1: LSB first	R/W
b11~b8	DSIZE[3:0]	SPI data length set position	b11~b8 0 0 0 0: 4 bits 0 0 0 1: 5 bits 0 0 1 0: 6 bits 0 0 1 1: 7 bits 0 1 0 0: 8 bits 0 1 0 1: 9 bits 0 1 1 0: 10 bits 0 1 1 1: 11 bits 1 0 0 0 0: 12 bits 1 0 0 1: 13 positions 1 0 1 0: 14 bits 1 0 1 1: 15 bits 1 1 0 0: 16 bits 1 1 0 1: 20 positions 1 1 1 1 0: 24 bits 1 1 1 1 : 32 bits	R/W
b7~b5	SSA [2:0]	SS signal effective set positioning	b7~b5 0 0 0 0: SS0 0 0 1: SS1 0 1 0: SS2 0 1 1: SS3 1 x x: Prohibit setting	R/W
b4~b2	MBR [2:0]	Bit rate crossover setting positioning	b4~b2 0 0 0 0: Selects 2 divisions of the basic bit rate 0 0 1: Selects 4 divisions of the basic bit rate 0 1 0: Selects the basic bit rate of 8 divisions 0 1 1: Selects 16 divisions of the basic bit rate	R/W

1 0 0: Selects 32 divisions of the basic bit rate

1 0 1: Select 64 divisions of the basic bit rate

1 1 0: Select 128 divisions of the basic bit rate

1 1 1: Selects 256 divisions of the basic bit rate

b1	CPOL	SCK polarity set position	0: SCK is Low level when idle 1: SCK is High level when idle	R/W
b0	CPHA	SCK phase setting position	0: Data sampling at odd edges, data change at even edges 1: Data change at odd edges, data sampling at even edges	R/W

28 Quad Wire Serial Peripheral Interface (QSPI)

28.1 Introduction

The Quad Wire Serial Peripheral Interface (QSPI) is a memory control module used to communicate with serial ROMs with SPI-compatible interfaces. The targets include serial Flash, serial EEPROM and serial FeRAM.

Table 28-1 QSPI Main Specifications

Par ame ters	Spe cific atio n
Number of channels	1 channel
SPI	— Supports multiple protocols such as extended SPI, 2-wire SPI and 4-wire SPI
	— Supports SPI mode 0 and SPI mode 3
	— Address line width selectable 8-bit/16-bit/24-bit/32-bit
Timing Adjustment	Timing can be adjusted to support various serial flash memories
Flash Memory Read	- Support multiple reading methods a. Standard reading/fast reading b. 2-wire output fast readout / 2-wire input and output fast readout c. Four-wire output fast readout / four-wire input and output fast readout
	— Free setup command
	— Adjustable number of virtual cycles
	-16-byte pre-read function
	— Status Inquiry Function
	-SPI bus cycle extension function
	— XIP control function
	—
	—
	—
Direct communication function	Flexible and extensive support for a wide range of serial flash software control commands, including erase, write, ID read and power-down control, etc.
Interrupt source	Hardware error interrupts
Module stop function	Power consumption can be reduced by module stop

QSPI

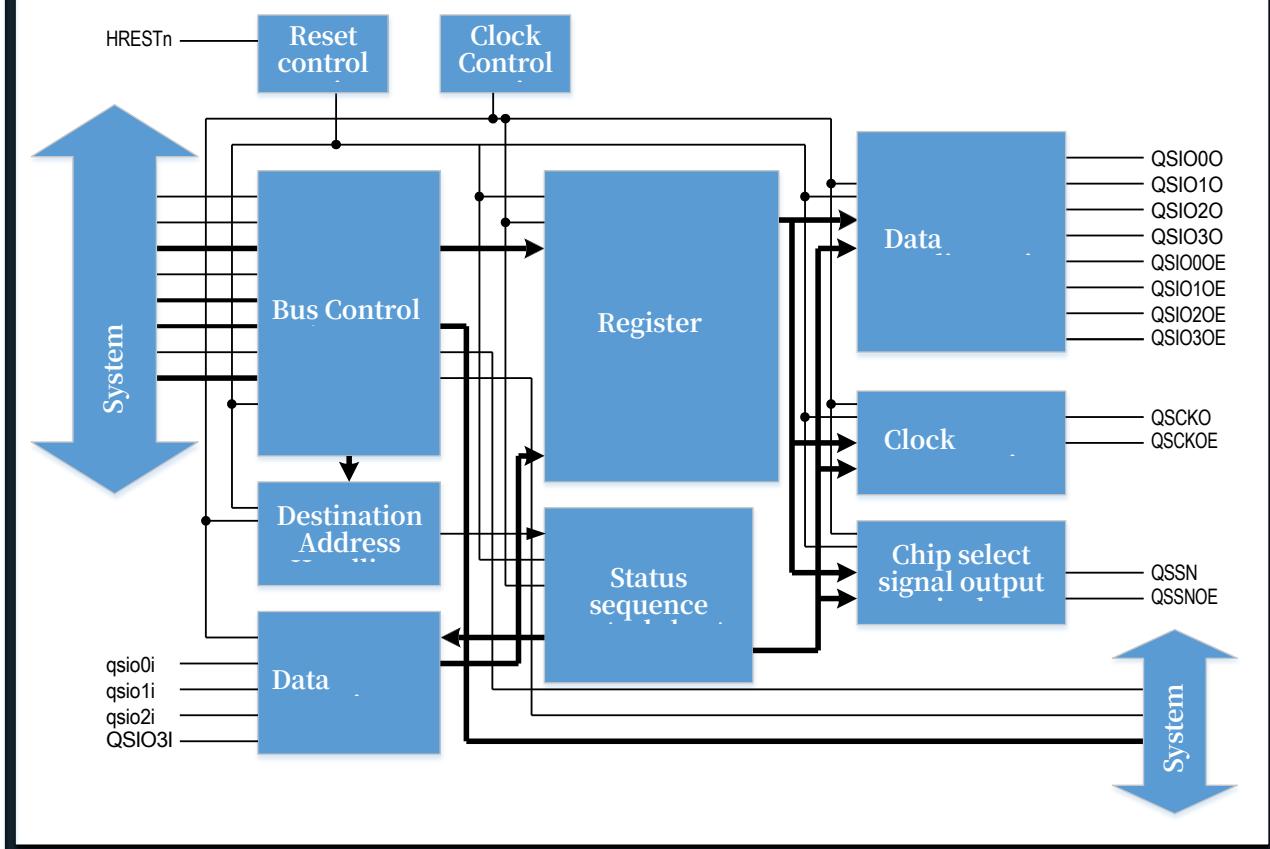


Figure 28-1 Module Composition of QSPI

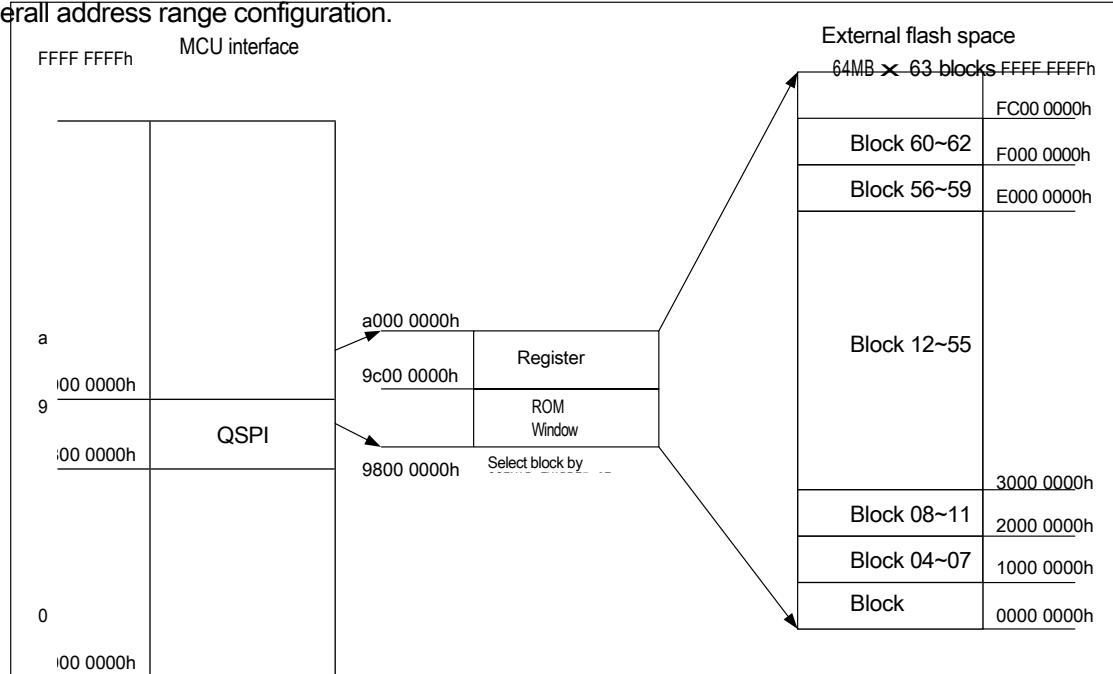
Table 28-2 QSPI Pins

Footnote Name	Input/output	Function Description
QSCK	Output	QSPI Clock Output Pins
QSSN	Output	QSPI Slave Selection Pins
QSI00	Input/output	Data line 0
QSI01	Input/output	Data cable1
QSI02	Input/output	Data cable 2
QSI03	Input/output	Data Cable 3

28.2 Memory Mapping

28.2.1 Internal bus space

The location of the serial flash and associated control registers in the AHB bus space is determined by the overall address range configuration.



**Figure 28-2 Default area setting and AHB bus space
memory mapping relationship**

28.2.2 ROM space and address width of the bus

The QSPI has a 32-bit address bus width to match the serial flash memory. Whenever a read access is made to the ROM space of the **QSPI**, the QSPI bus automatically starts working to transfer the data read from the serial flash memory.

Instead of only using 32-bit address bus widths, QSPI can also choose to use 8-bit/16-bit/24-bit address bus widths by setting AWSL[1:0] in the QSFCR register.

If the 8-bit/16-bit/24-bit address bus width is selected, only the low bit space with the matching address can be accessed normally, i.e., accessing the high bit of the serial flash mirror space in the

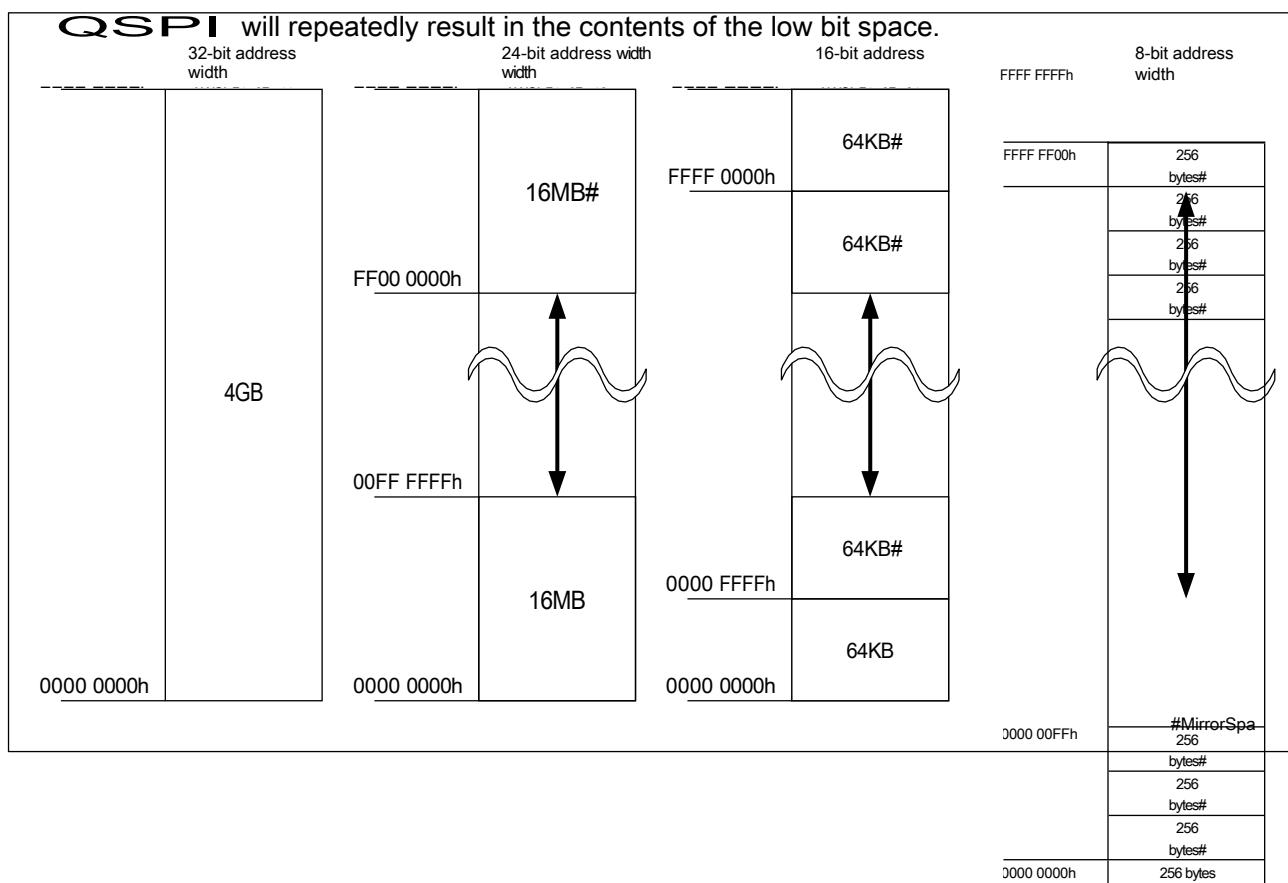


Figure 28-3 QSPI-ROM Space Memory Image Map

Cautio

n:

- The address bus width can be selected by setting AWSL[1:0] in the QSFCR register to use 8-bit/16-bit/24-bit /32-bit.

28.3 QSPI Bus

28.3.1 SPI Protocol

This **QSPI** supports three protocols: Extended SPI, 2-wire **SPI**, and 4-wire SPI. The initial default protocol is the extended SPI protocol. The protocol can be configured separately for each stage by setting the DPRSL[1:0]/APRSL[1:0]/IPRSL[1:0] bits in the QSCR register. The extended **SPI** protocol uses only the single line of the QSIO0 pin for instruction output, and the subsequent address and data are output using single **line**/ two **line**/ four line depending on the specific read mode instruction.

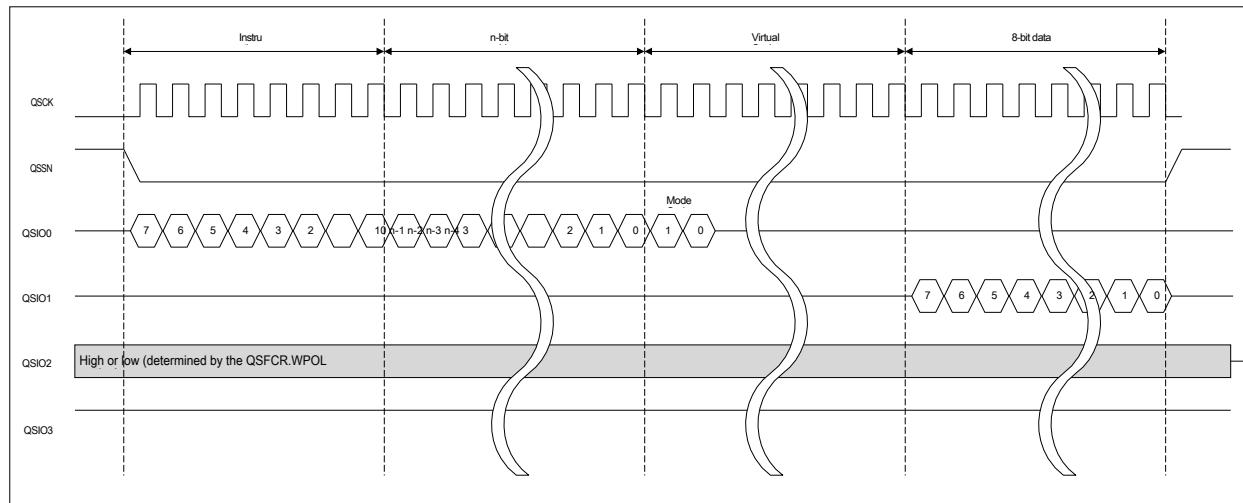


Figure 28-4 Extended SPI Protocol Action Schematic 1 (Fast Read Mode)

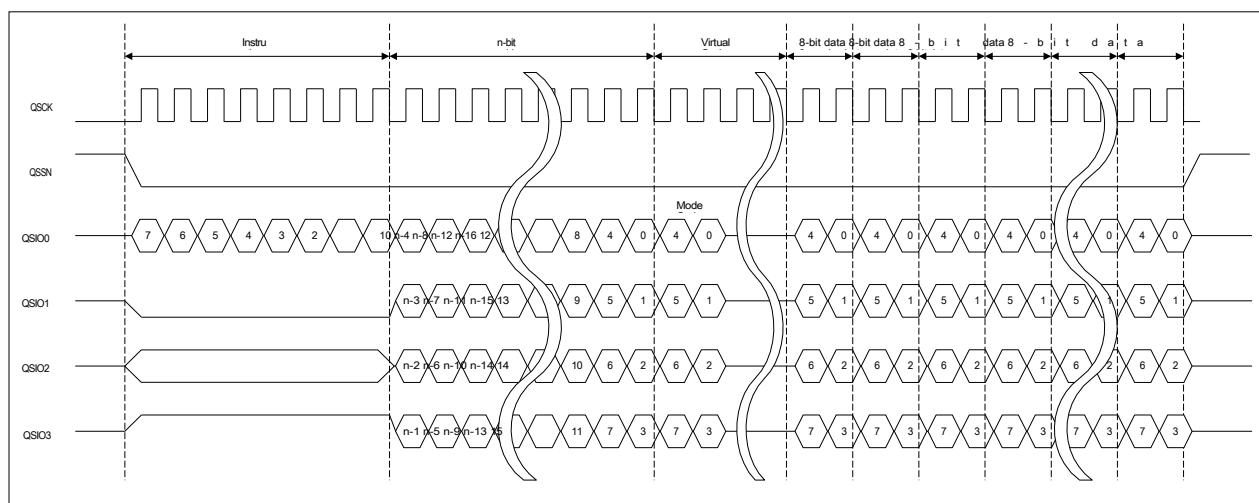


Figure 28-5 Extended SPI Protocol Action Diagram 2 (4-Wire Input/Output Fast Read Mode)

The two-wire **SPI** protocol uses the QSIO0 and QSIO1 pins to perform the appropriate operations, including issuing commands, addressing, receiving data, and so on.

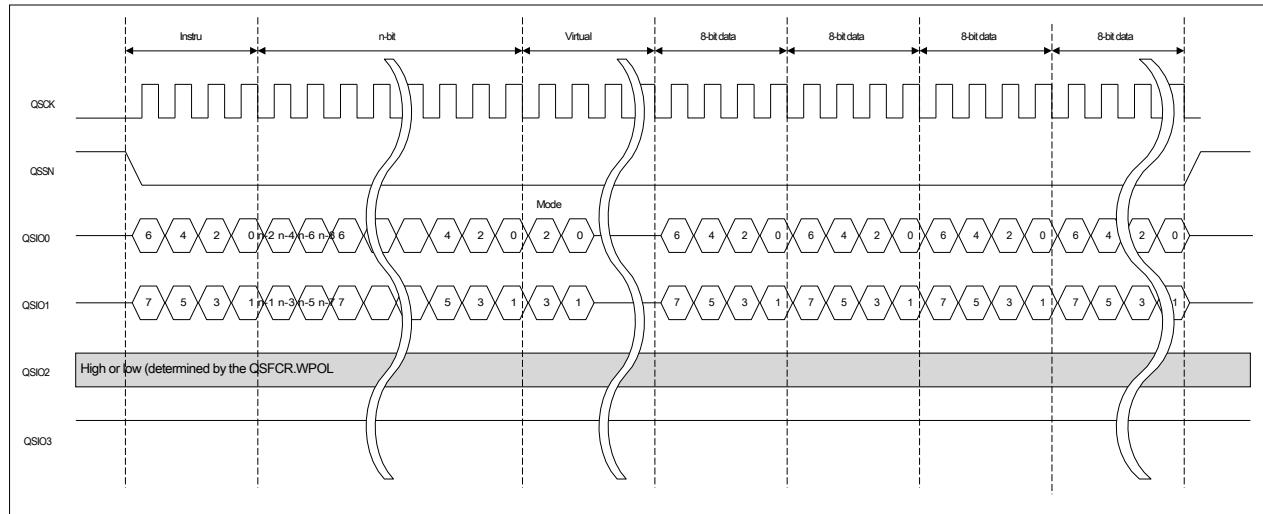


Figure 28-6 2-wire SPI protocol action diagram (fast read mode)

The four-wire **SPI** protocol uses four pins, QSIO0, QSIO1, QSIO2, and QSIO3, to perform all related operations such as issuing commands, addressing, and receiving data.

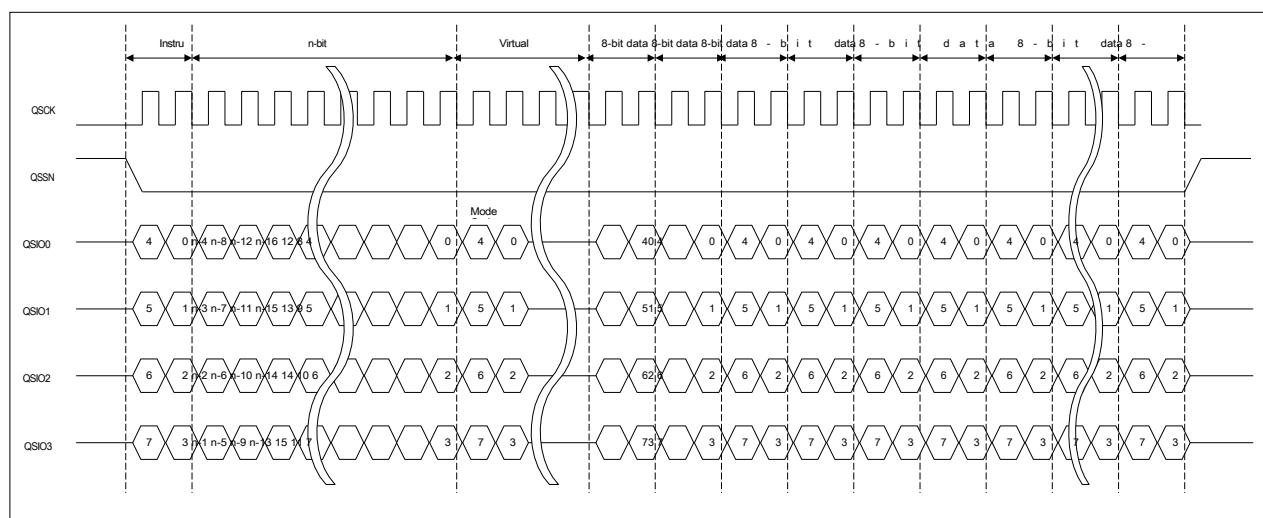


Figure 28-7 Four-wire SPI Protocol Action Schematic (Fast Read Mode)

28.3.2 SPI mode

SPI mode 0 and mode 3 are available and can be switched by setting the SPIMD3 bit in the QSCR register. The difference between SPI mode 0 and mode 3 is the difference in the standby level of QSCK during the standby state. In **SPI mode 0, the standby level of QSCK is low, while in mode 3, the standby level is high.**

The serial data is output from the QSPI on the falling edge of the serial clock and read into the external flash on the rising edge. The external flash memory outputs serial data on the falling edge of the serial clock and is read in by the **QSPI on the** falling edge of the next clock.

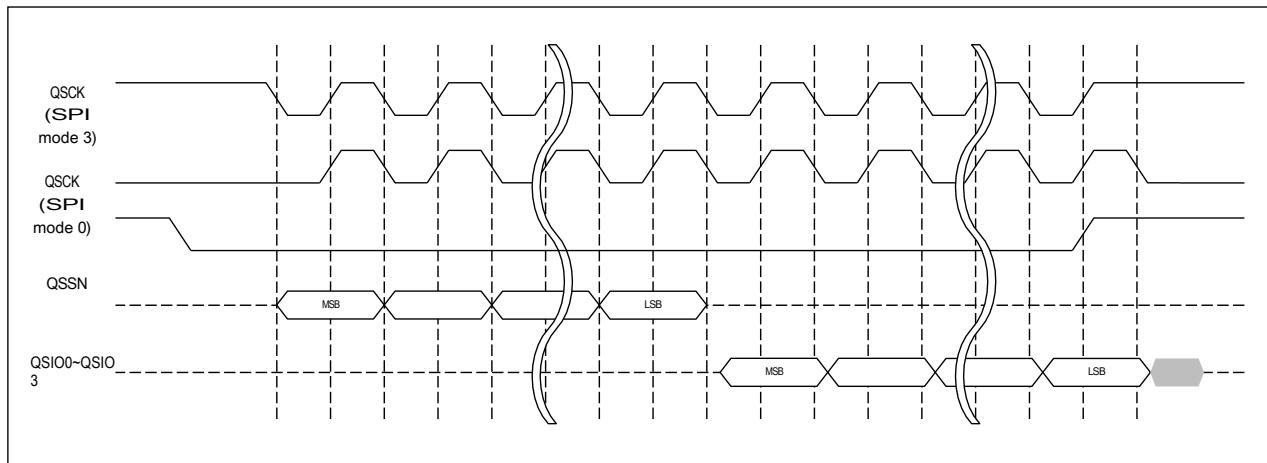


Figure 28-8 Basic timing diagram of the serial interface

28.4 Timing Adjustment of **QSPI** Bus

The QSPI bus signals can be fine-tuned in timing via registers, and the fine-tuned timing can be better used for various QSPI bus accesses, either in ROM access mode or direct communication mode.

28.4.1 **QSPI** Bus Reference Clock

The reference clock for the QSPI bus is obtained by dividing HCLK. By setting the DIV[5:0] bits of the QSCR register, you can select a variety of clock sources from 2 to 64 divisions of HCLK as the reference clock for the **QSPI** bus.

Table 28-3 QSPI Bus Reference Clock Selection List

DIV[5:0]	Crossover Ratio	Actual movement frequency	DIV[5:0]	Crossover Ratio	Actual movement frequency
		(HCLK=200MHz)			(HCLK=200MHz)
6'b000000	2	100.00	6'b100000	33	6.06
6'b000001	2	100.00	6'b100001	34	5.88
6'b000010	3	66.67	6'b100010	35	5.71
6'b000011	4	50.00	6'b100011	36	5.56
6'b000100	5	40.00	6'b100100	37	5.41
6'b000101	6	33.33	6'b100101	38	5.26
6'b000110	7	28.57	6'b100110	39	5.13
6'b000111	8	25.00	6'b100111	40	5.00
6'b001000	9	22.22	6'b101000	41	4.88
6'b001001	10	20.00	6'b101001	42	4.76
6'b001010	11	18.18	6'b101010	43	4.65
6'b001011	12	16.67	6'b101011	44	4.55
6'b001100	13	15.38	6'b101100	45	4.44
6'b001101	14	14.29	6'b101101	46	4.35
6'b001110	15	13.33	6'b101110	47	4.26
6'b001111	16	12.50	6'b101111	48	4.17
6'b010000	17	11.76	6'b110000	49	4.08
6'b010001	18	11.11	6'b110001	50	4.00
6'b010010	19	10.53	6'b110010	51	3.92
6'b010011	20	10.00	6'b110011	52	3.85
6'b010100	21	9.52	6'b110100	53	3.77
6'b010101	22	9.09	6'b110101	54	3.70
6'b010110	23	8.70	6'b110110	55	3.64
6'b010111	24	8.33	6'b110111	56	3.57
6'b011000	25	8.00	6'b111000	57	3.51
6'b011001	26	7.69	6'b111001	58	3.45

6'b011010	27	7.41	6'b111010	59	3.39
-----------	----	------	-----------	----	------

DIV[5:0]	Crossover Ratio	Actual movement frequency	DIV[5:0]	Crossover Ratio	Actual movement frequency
		(HCLK=200MHz)			(HCLK=200MHz)
6'b011011	28	7.14	6'b111011	60	3.33
6'b011100	29	6.90	6'b111100	61	3.28
6'b011101	30	6.67	6'b111101	62	3.23
6'b011110	31	6.45	6'b111110	63	3.17
6'b011111	32	6.25	6'b111111	64	3.13

28.4.2 SPI Bus Reference Clock

When an even division of HCLK is selected for the reference clock, the high and low times of the QSCK signal are the same, and if an odd division is selected, the high time of the QSCK signal will be one HCLK cycle longer than the low.

If you want the QSCK to output a clock signal with a duty cycle of about 50% even when selecting an odd division, set the DUTY bit in the QSFCR register to 1. With this setting, the output time of the rising edge of the QSCK signal will be half an HCLK cycle later than before the adjustment, and the output time of the falling edge will remain unchanged. This results in a QSCK signal with a duty cycle of 50%. The DUTY bit is set to 0 when an even division of HCLK is selected for the reference clock, and the DUTY bit defaults to 1 in the initial state.

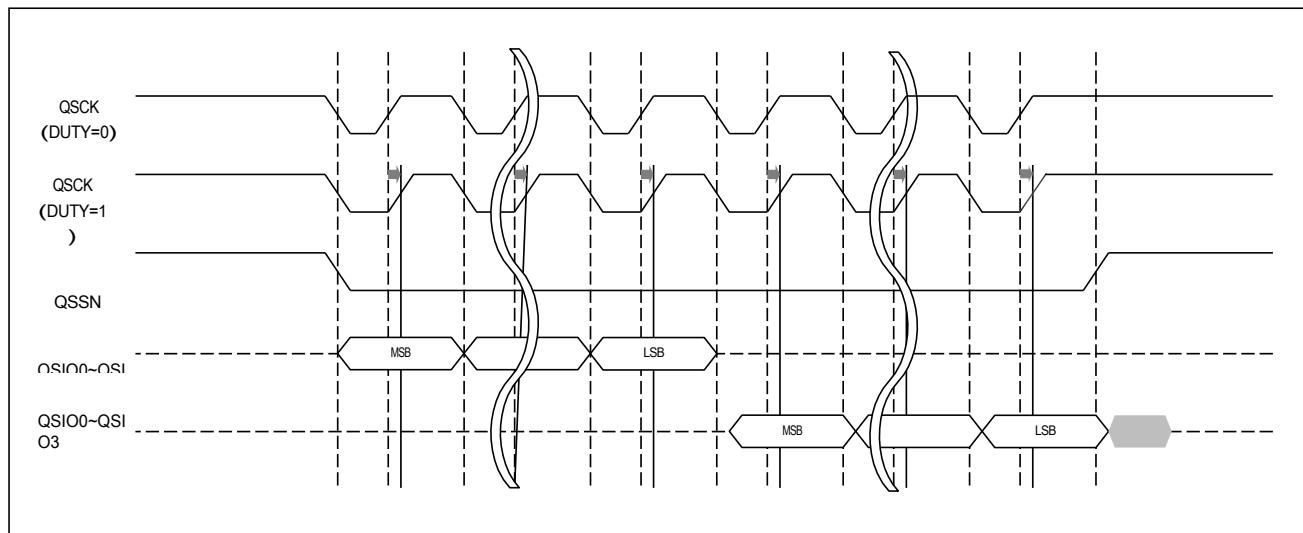


Figure 28-9 Schematic diagram of output clock duty cycle correction when HCLK three-division is selected for the base clock

28.4.3 QSSN signal minimum high level width

To meet the cancellation time requirements required by the serial flash, the QSSN signal must remain high (i.e., idle) long enough between adjacent **QSPI** bus cycles. The minimum high width of QSSN can be selected by setting the SSHW[3:0] bits in the QSCSCR register from 1

to 16 QSPI baseline clock cycles.

28.4.4 QSSN build time

The time between when the QSSN signal starts to go low (i.e., becomes active) and the first rising edge of the QSCK signal output is called the QSSN build time, which can be configured via a register setting to meet the requirements of the external serial flash. The SSNLD bit of the QSFCR register is set to select whether the QSSN build time is 0.5 or 1.5 QSPI reference clock cycles. This setting can also be used to configure the build time of the data pin from the data output permit to the first rising edge of the QSCK signal output, as required.

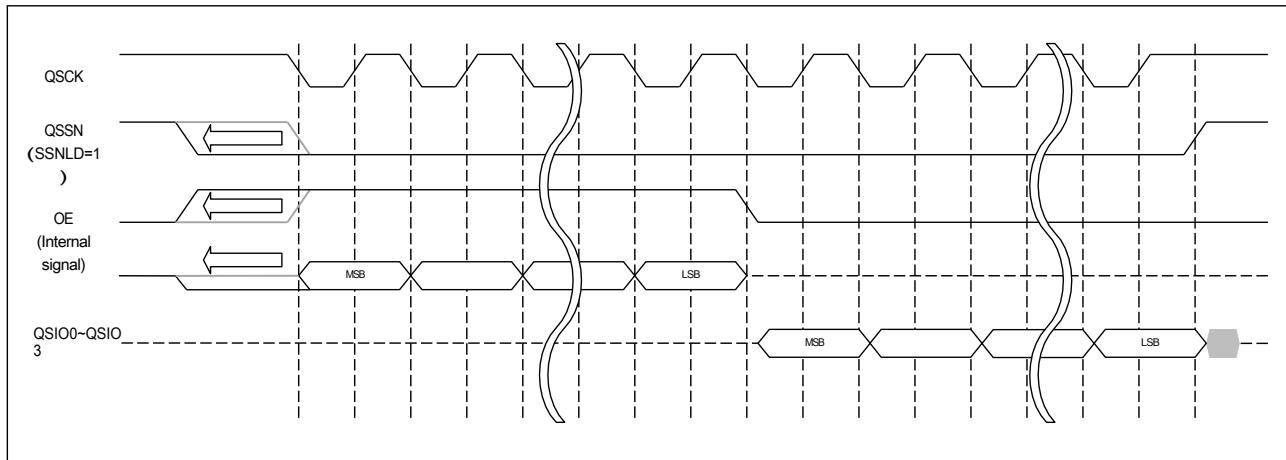


Figure 28-10 QSSL Build Time Configuration Diagram

28.4.5 QSSN hold time

The time between the last rising edge of the QSCK output and the start of the QSSN signal going high (i.e., becoming idle) is called the QSSN hold time, which can be configured via a register setting to meet the requirements of the external device. Set the SSNHD bit of the QSFCR register to select whether the QSSN hold time is 0.5 or 1.5 QSPI reference clock cycles.

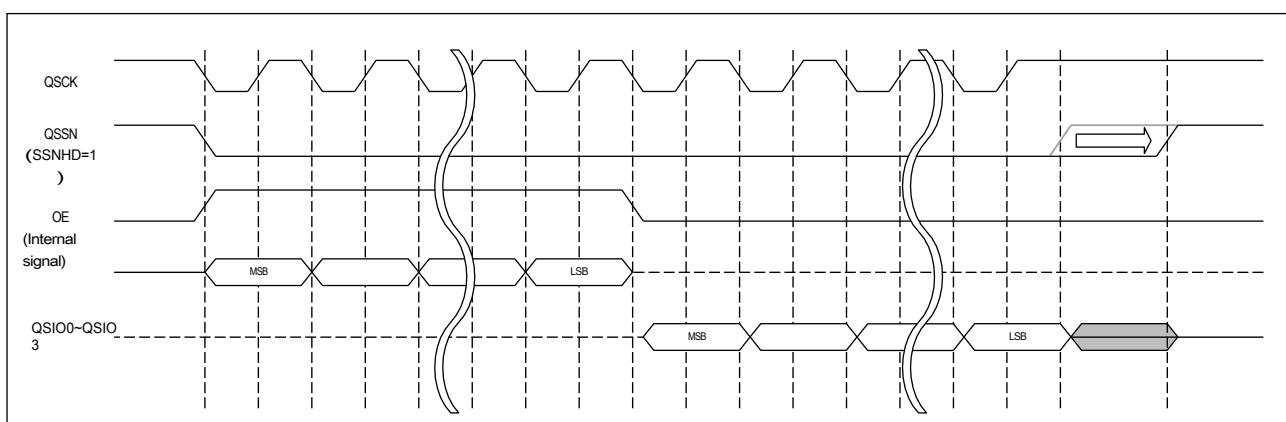


Figure 28-11 Diagram of QSSN hold time configuration

28.4.6 Serial data reception delay

The data from the serial flash is output synchronously with the falling edge of QSCK, and the QSPI receives that data on the falling edge of the next QSCK. The time between when the serial flash starts outputting data and when that data is received by the QSPI is called the receive delay, and the QSPI adds a delay adjustment period before the first data receive cycle. From the serial flash perspective, this period can be considered as an increase in the adversary's data reception period. This delay adjustment cycle is only generated during the data receive action.

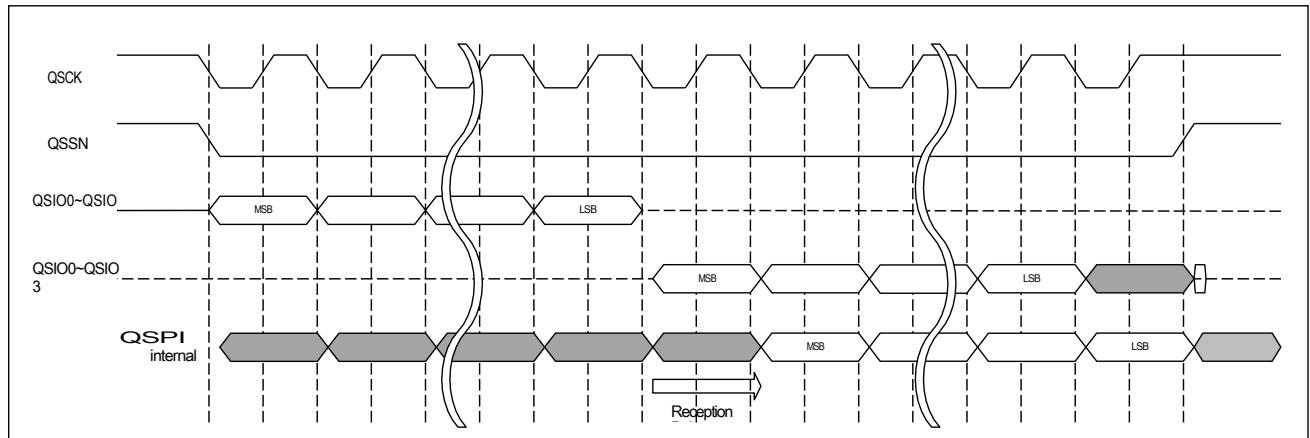


Figure 28-12 Data reception delay diagram



28.5 Introduction of SPI Instructions for ROM Access

28.5.1 Existing QSPI-ROM Instruction Reference

Table 28-4 List of Reference Instructions

Model Name	Command Code	Description	
4-byte instruction mode	Standard Reading	8'h13	
	Quick Read	8'h0C	
	2-wire output quick read	8'h3C	
	2-wire input/output quick read	8'hbC	
	Four-wire output quick read	8'h6C	
	Four-wire input/output quick read	8'hEC	
	Exit 4-byte command mode	8'HB7	
3-byte instruction mode	Standard Reading	8'h03	
	Standard Reading	8'h0B	When 8-bit address is selected and A8=1
	Quick Read	8'h0B	-
	2-wire output quick read	8'h3B	-
	2-wire input/output quick read	8'hBB	-
	Four-wire output quick read	8'h6B	-
	Four-wire input/output quick read	8'hEB	-
	Enter 4-byte command mode	8'hE9	-
-	Write mode	8'h06	-

To access the serial flash memory, the instructions are set via the instruction register QSCCMD.

28.5.2 Standard Read Instructions

The standard read instruction is a common read instruction supported by most serial flash memories. At the beginning of a serial bus cycle, the serial flash select signal is set to active, and the **QSPI** outputs the instruction code (03h/13h)*1 for that instruction, followed by the destination address, the width of which can be set by the **AWSL[1:0]** bits in the QSFCR register. The data is then ready to be received. The instruction selected for the initial state of the **QSPI** is the standard read instruction.

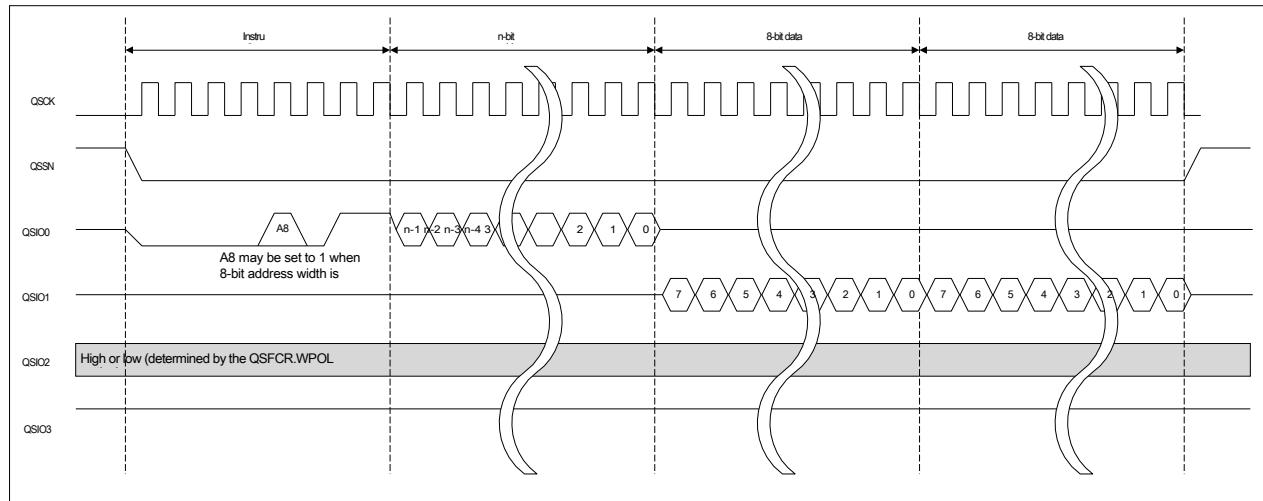


Figure 28-13 Standard Read Bus Cycle Diagram

28.5.3 Quick read command

The fast read instruction is a read instruction that supports a faster communication clock. At the beginning of a serial bus cycle, the serial flash select signal is set to active and the **QSPI** outputs the instruction code (0Bh/0Ch)*1 for the instruction, followed by the target address, the width of which can be set by the **AWSL[1:0]** bits in the QSFCR register. The address output is followed by a certain number of virtual cycles, the exact number of which is determined by **DMCYCN[3:0]** in the QSFCR register. This is immediately followed by the reception of data.

The first two cycles of the virtual cycle are used to determine whether XIP mode is selected. When XIP mode is selected, the instruction used in this transfer will be applied to the next SPI bus cycle, and the instruction transfer portion will be omitted on the next SPI bus cycle. For details, refer to

28.7 XIP control.

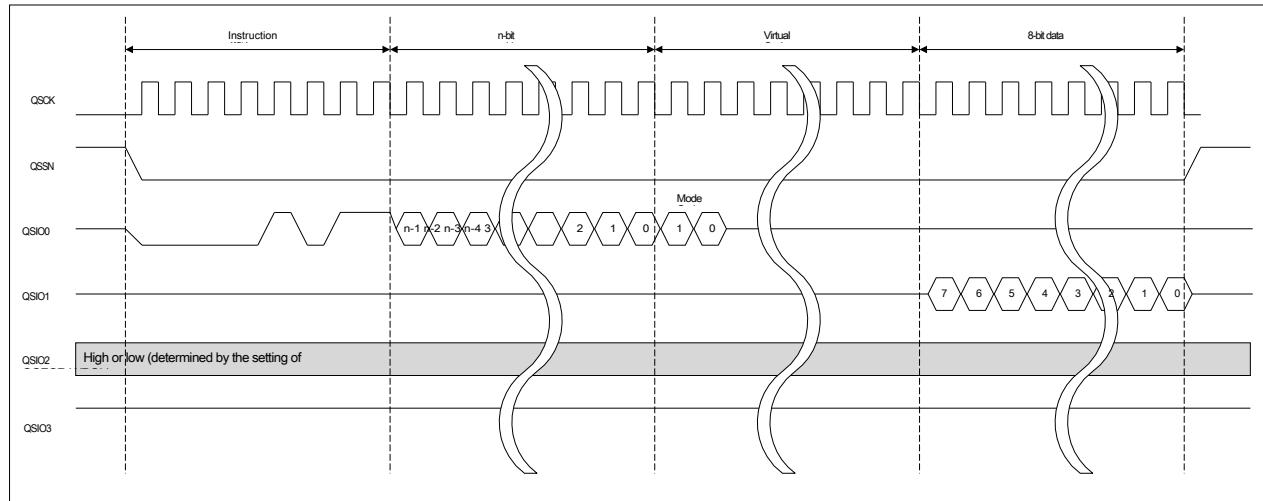


Figure 28-14 Fast Read Bus Cycle Diagram

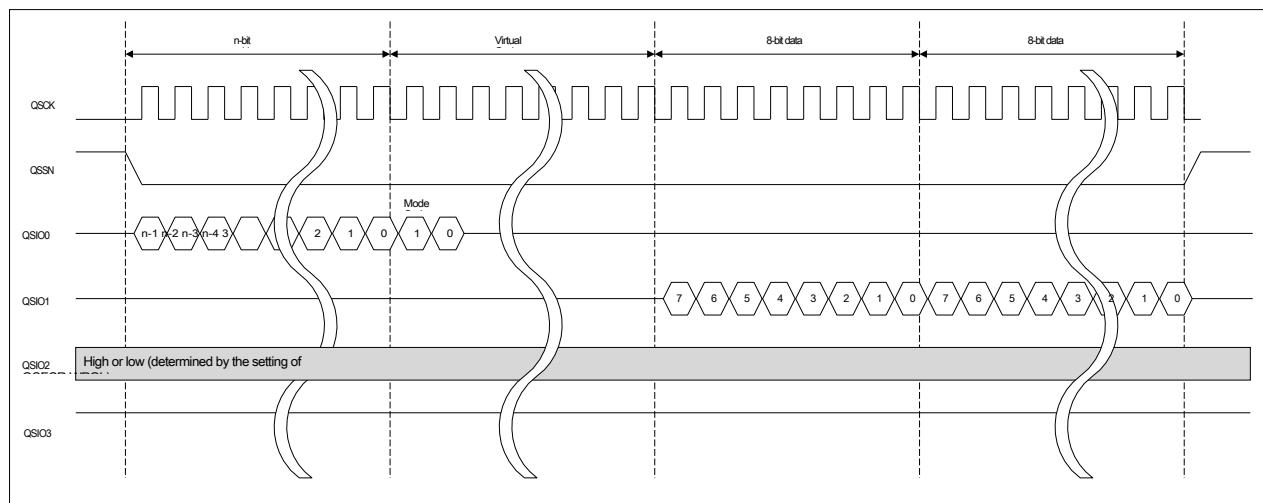


Figure 28-15 Diagram of fast read bus cycle with XIP mode selected

Cautio

n:

- - To use the fast read command make sure to use a serial flash that supports the fast read feature.

28.5.4 2-wire output fast read command

A 2-wire output fast read is a read instruction that uses two signal lines for data reception. At the beginning of a serial bus cycle, the serial flash select signal is set to active and the QSPI begins to output the instruction code (3Bh/3Ch) and the destination address for that instruction from the QSIO0 pin, the address width of which can be set by the AWSL[1:0] bits in the QSFCR register. This is followed by a certain number of virtual cycles, the exact number of which is determined by DMCYCN[3:0] in the QSFCR register. Data reception then begins via the QSIO0 and QSIO1 pins. The even-bit data is received at QSIO0 and the odd-bit at QSIO1.

The first two cycles of the virtual cycle are used to determine whether XIP mode is selected. When XIP mode is selected, the instruction used for this transfer will be applied to the next QSPI bus cycle, and the instruction transfer portion will be omitted on the next QSPI bus cycle. Refer to 28.7 XIP Control for details.

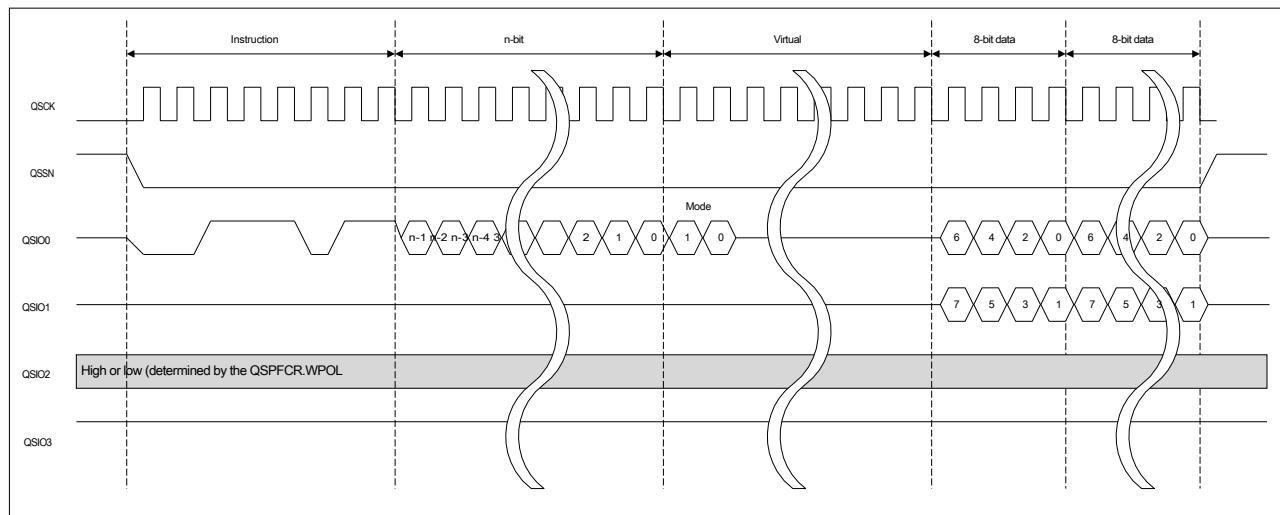
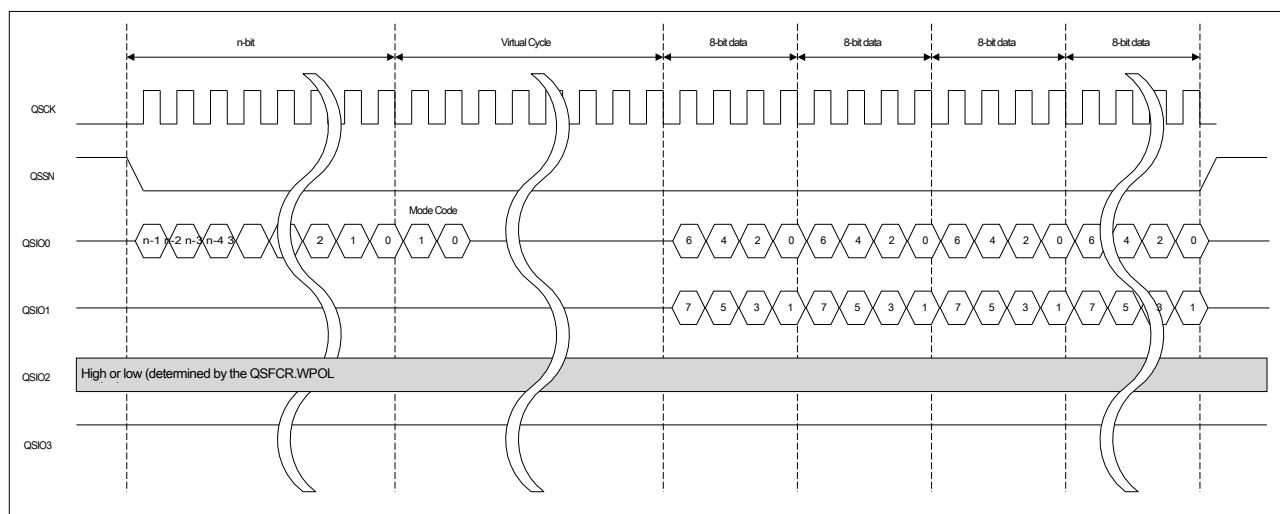


Figure 28-16 Schematic diagram of a 2-wire output fast read bus cycle



Caution:

Figure**28-17 Diagram of fast read bus cycle with 2-wire output for XIP mode selection**

-- To use the 2-wire output fast read command make sure to use a serial flash memory that supports this feature.

28.5.5 2-wire input/output quick read command

A 2-wire input-output fast read is a read instruction that uses two signal lines for address sending and data receiving. When a serial bus cycle begins, the serial flash select signal is set to active and the QSPI begins to output the instruction code for this instruction from the QSIO0 pin

(BBh/BCh) After this the QSPI outputs the target address from pins QSIO0 and QSIO1, and the address width can be set by the AWSL[1:0] bits in the QSFCR register. This is followed by a certain number of virtual cycles, the exact number of which is determined by DMCYCN[3:0] in the QSFCR register. Data reception then begins via the QSIO0 and QSIO1 pins. The QSIO0 pin is used for address and virtual cycle (including XIP mode selection information) transfer and data reception for even bits, and the QSIO1 pin is used for odd bits.

The first two cycles of the virtual cycle are used to determine whether XIP mode is selected. When XIP mode is selected, the instruction used for this transfer will be applied to the next QSPI bus cycle, and the instruction transfer portion will be omitted on the next QSPI bus cycle. Refer to 28.7 XIP Control for details.

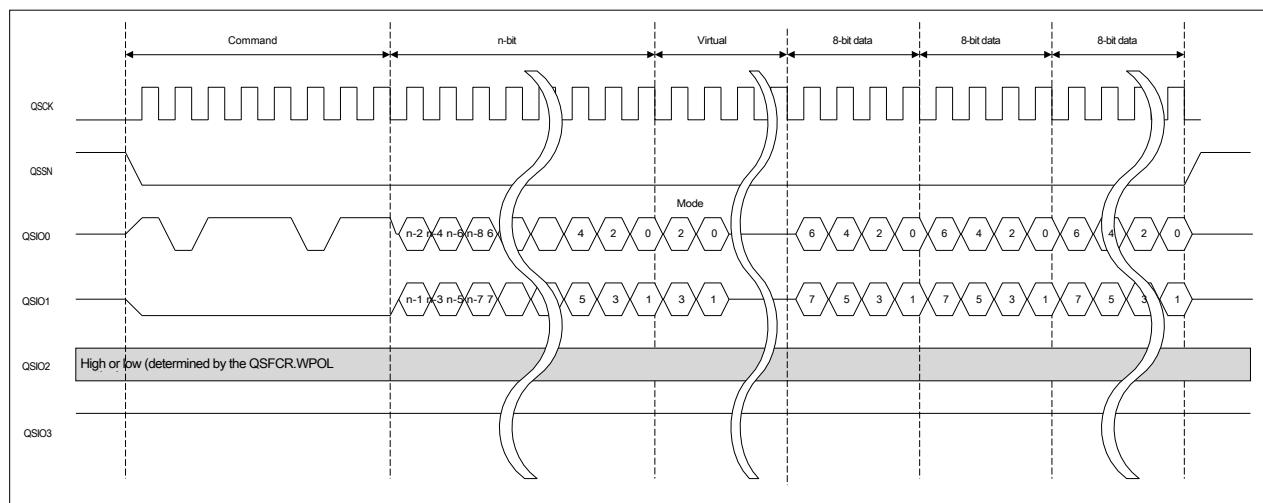
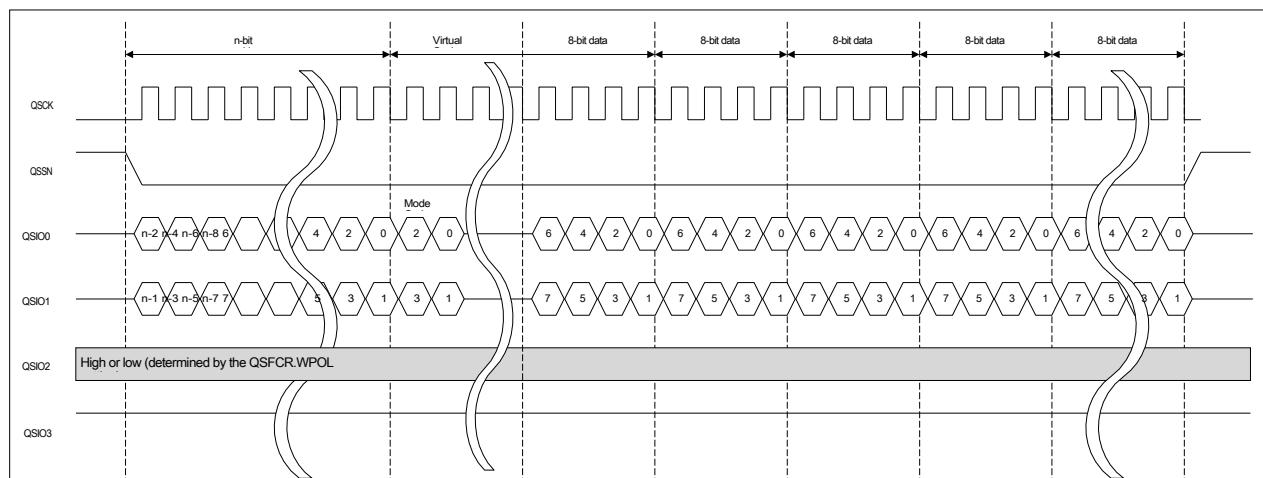


Figure 28-18 2-wire input/output fast read bus cycle diagram



**Figure 28-19 Diagram of 2-wire input/output fast read bus
cycle with XIP mode selected**

Caution:

- - To use the 2-wire input-output fast read command make sure you use a serial flash memory that supports this feature.

28.5.6 Four-line output fast read command

The Quad Output Fast Read is a read instruction that uses four signal lines for data reception. At the beginning of a serial bus cycle, the serial flash select signal is set to active and the QSPI starts outputting the instruction code (6Bh/6Ch) and the destination address for that instruction from the QSIO0 pin, the address width of which can be set by the AWSL[1:0] bits in the QSFCR register. This is followed by a certain number of virtual cycles, the exact number of which is determined by DMCYCN[3:0] in the QSFCR register. Then the data reception starts through the four pins QSIO0, QSIO1, QSIO2 and QSIO3.

The first two cycles of the virtual cycle are used to determine whether XIP mode is selected.

When XIP mode is selected, the instruction used in this transfer will be applied to the next SPI bus cycle, and the instruction transfer portion will be omitted on the next SPI bus cycle. For details, refer to

28.7 XIP control.

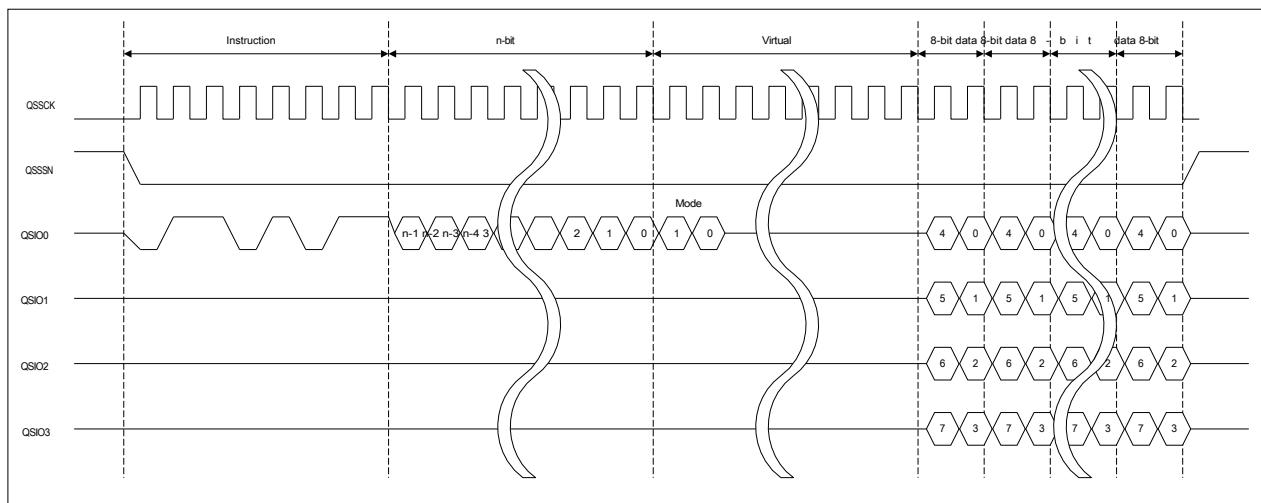


Figure 28-20 Four-wire output fast read bus cycle diagram

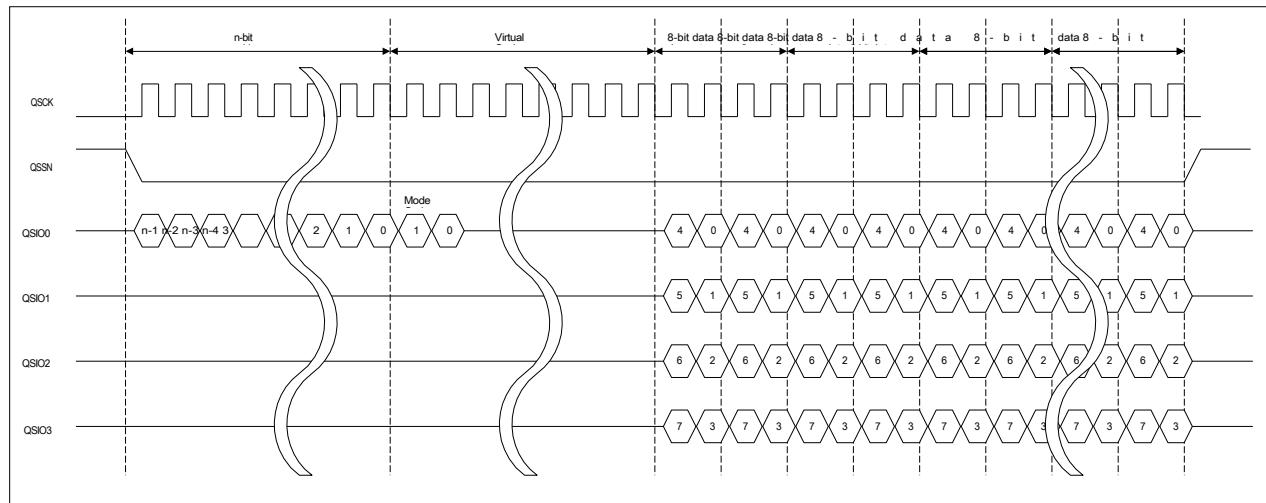


Figure 28-21 Schematic diagram of a four-wire output fast read bus cycle with XIP mode selected

Caution:

- - To use the four-line output fast read command make sure you use a serial flash memory that supports this feature.

28.5.7 Four-wire input/output quick read command

A quad-wire input-output fast read is a read instruction that uses four signal lines for address sending and data receiving. When a serial bus cycle begins, the serial flash select signal is set to active and the QSPI begins to output the instruction code for this instruction from the QSIO0 pin (EBh/ECh). After this the QSPI outputs the destination address from the four pins QSIO0, QSIO1, QSIO2 and QSIO3. The address width can be set by the AWSL[1:0] bits in the QSFCR register. This is followed by a certain number of virtual cycles, the exact number of which is determined by DMCYCN[3:0] in the QSFCR register. Then the data reception starts through the four pins QSIO0, QSIO1, QSIO2 and QSIO3.

The first two cycles of the virtual cycle are used to determine whether XIP mode is selected. When XIP mode is selected, the instruction used for this transfer will be applied to the next QSPI bus cycle, and the instruction transfer portion will be omitted on the next QSPI bus cycle. Refer to 28.7 XIP Control for details.

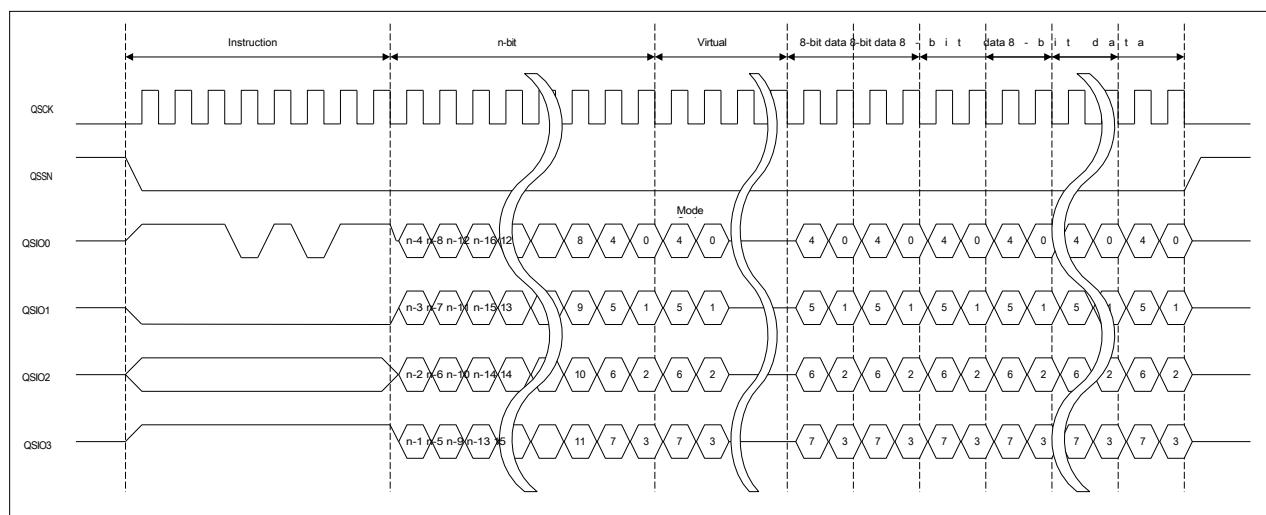


Figure 28-22 Four-wire input/output fast read bus cycle diagram

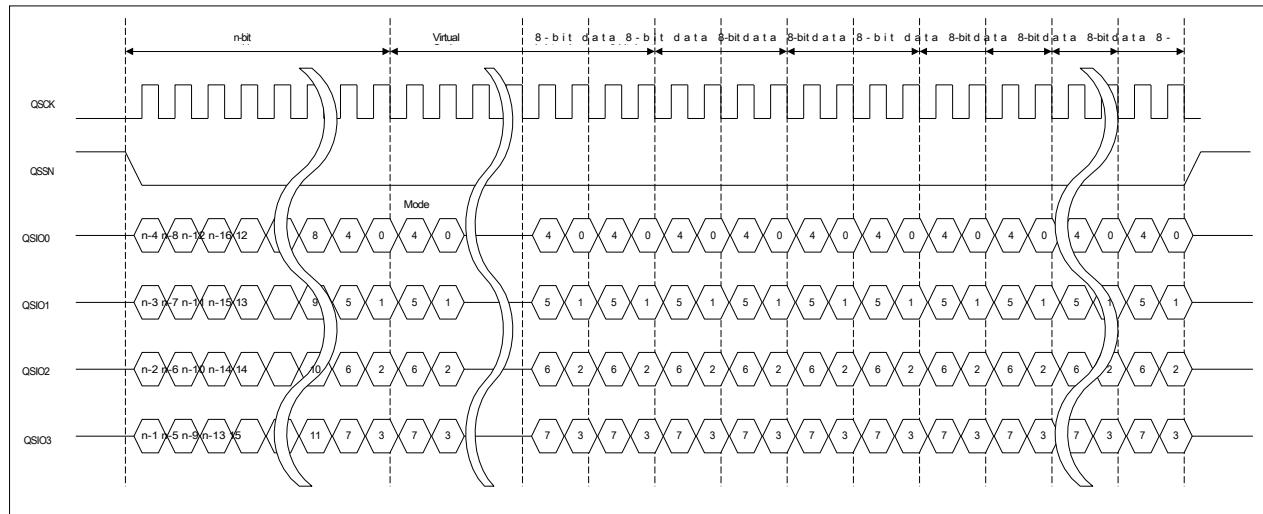


Figure 28-23 Four-wire input/output fast read bus cycle diagram with XIP mode selected

Cautio

n:

-- To use the four-wire input-output fast read command make sure you use a serial flash memory that supports this feature.

28.5.8 Enter 4-Byte mode command

The Enter 4-Byte Mode instruction sets the address width of the serial flash to 4 bytes. When a serial bus cycle begins, the Serial Flash Select signal is set to active and the **QSPI** begins to output the instruction code (B7h) from the QSIO0 pin.

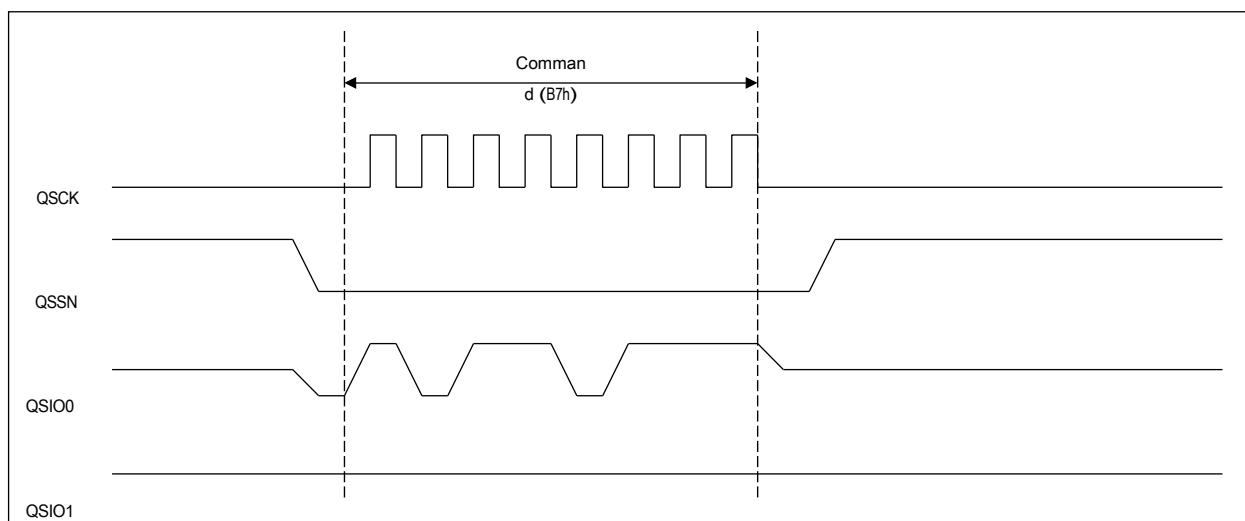


Figure 28-24 Diagram of the instruction bus cycle into 4-Byte mode

Cautio

n:

This command is issued regardless of whether the serial flash is in 4-Byte or 3-Byte mode. This command can be issued regardless of whether the serial flash is in 4-Byte or 3-Byte mode.

28.5.9 Exit 4-Byte mode command

The Exit 4-Byte mode instruction sets the address width of the serial flash to 3 bytes. When a serial bus cycle begins, the Serial Flash Select signal is set to active and the **QSPI** begins to output the instruction code (E9h) from the QSIO0 pin.

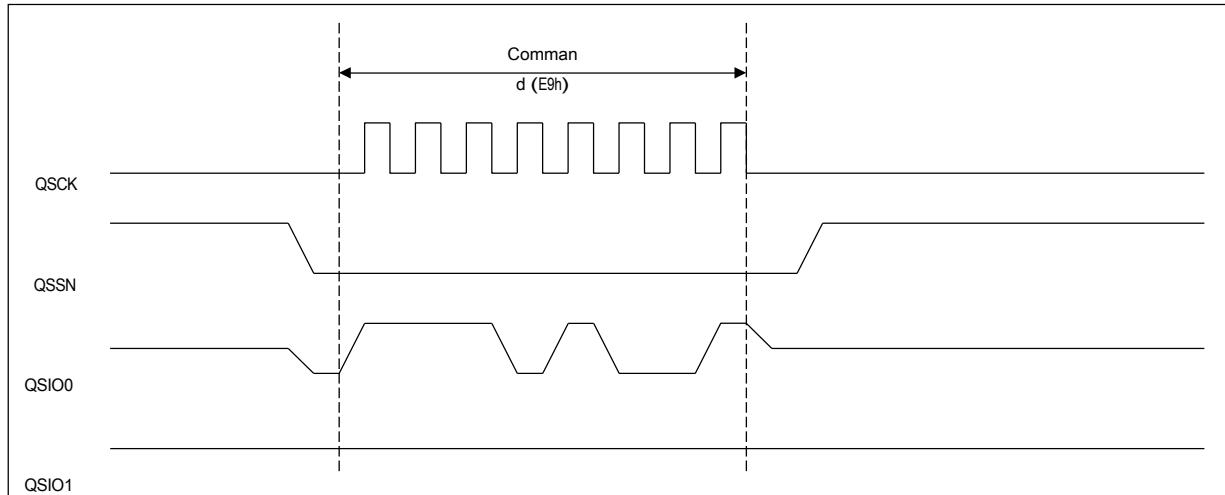


Figure 28-25 Exit 4-Byte Mode Instruction Bus Cycle Diagram

Cautio

n:

This command is issued regardless of whether the serial flash is in 4-Byte or 3-Byte mode. This command can be issued regardless of whether the serial flash is in 4-Byte or 3-Byte mode.

28.5.10 Write permission command

The write permit instruction allows the address width of the serial flash to be changed. When a serial bus cycle begins, the Serial Flash Select signal is set to active and the QSPI begins to output the instruction code (06h) from the QSIO0 pin.

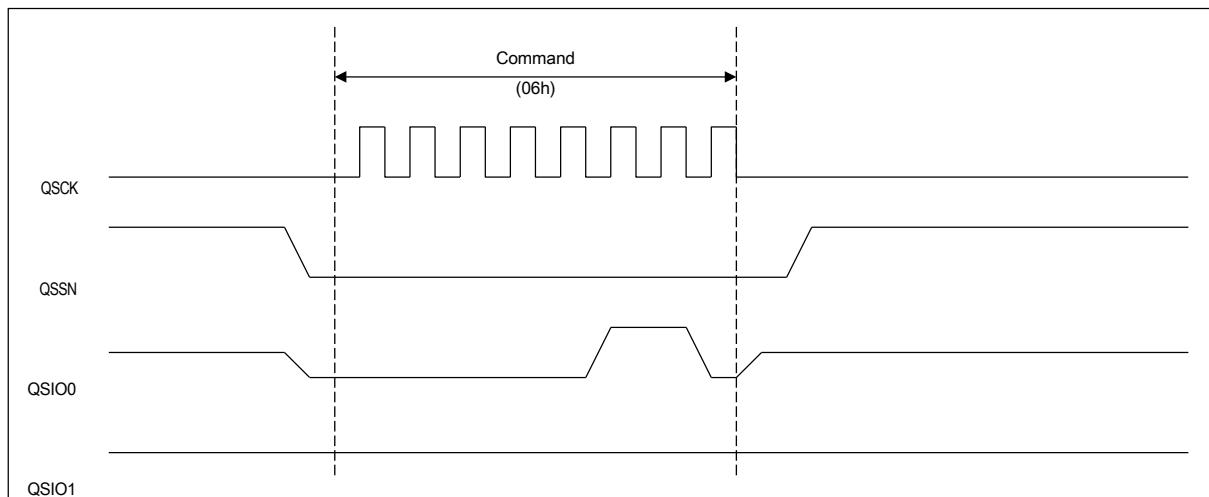


Figure 28-26 Write License Instruction Bus Cycle
Diagram

28.6 QSPI Bus Cycle Scheduling

28.6.1 Single flash read with independent conversion

A single read instruction for a ROM is independently converted one-to-one from the chip's internal bus cycle to a **QSPI** bus cycle. When a ROM read bus cycle is detected, the QSSN signal is set to the active state, initiating a QSPI bus cycle. When the data from the serial flash is received, the QSSN signal becomes invalid and the **QSPI** bus cycle is declared complete.

When another ROM read bus cycle is detected, the QSSN signal is set to active again after ensuring that the invalid hold time has exceeded the minimum invalid hold width, and a new QSPI bus cycle begins.

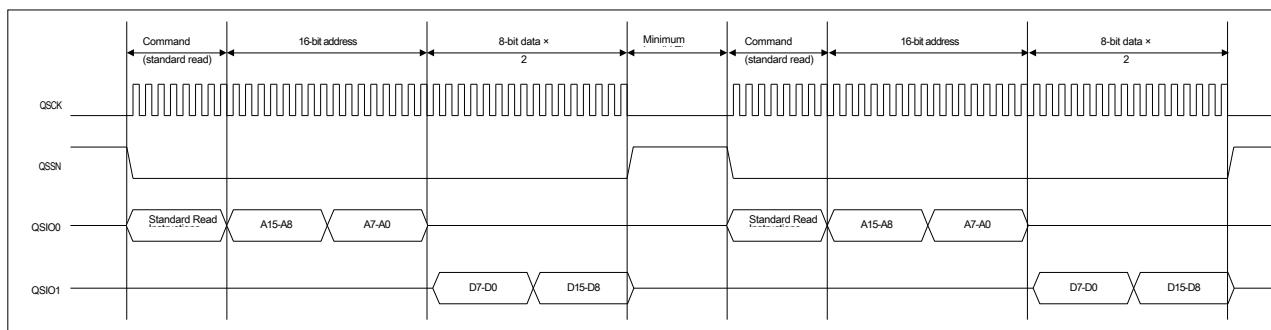


Figure 28-27 Schematic of a single flash data read operation with independent conversion

28.6.2 Flash memory reading using the pre-read function

For transfers such as CPU instructions or data blocks, the system typically reads the data in an incremental sequence of flash addresses. Serial flash memory has the ability to transfer data continuously without the need to retransmit instruction codes and addresses. However, if the internal bus cycles issued by the MCU are converted independently, the QSPI bus cycles are also divided into separate units, making it impossible to effectively take advantage of this continuous data transfer capability of serial flash. In response, QSPI provides a pre-read feature for continuous data reception.

Activate the Pre-Read function by setting the PFE bit in the QSCR register to 1. When this function is active, data is continuously received and stored in the buffer without waiting for another flash read request. When the MCU issues a flash read operation, the QSPI will match the access address. If the match is successful, the data in the buffer at the corresponding location is passed to the MCU; if the match fails, the data in the buffer is discarded and a new QSPI bus cycle is reissued.

The pre-read buffer can store up to 16 bytes of data, in addition to the 2-byte data receive buffer that can also store the pre-read data, and the QSPI bus cycle ends when all buffers are full. When all buffers are full, the QSPI bus cycle ends. When a new buffer space is created after the buffer data is read, the QSPI automatically starts a new QSPI bus cycle to resume the

pre-reading action.

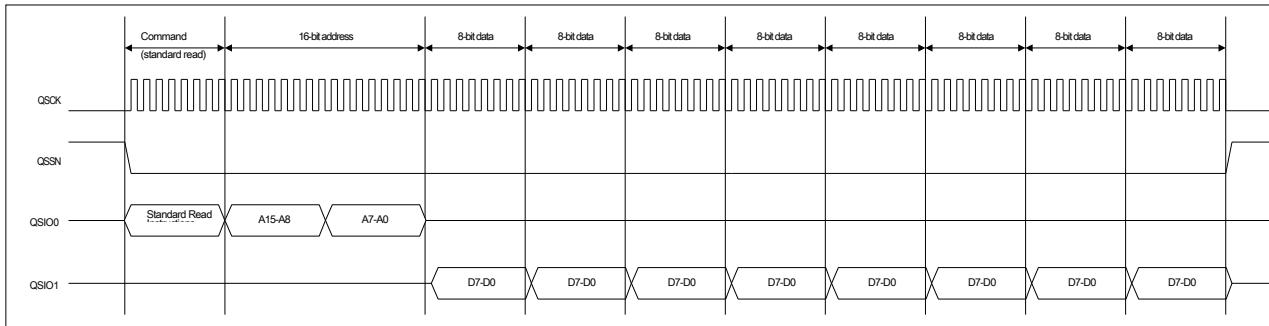


Figure 28-28 Diagram of data reading operation when the pre-read function is active

28.6.3 Pre-read termination

If a ROM read bus cycle to another address in the serial flash occurs during a preread transfer, the original preread action will be terminated and a new QSPI bus cycle will begin. Normally the pre-reading action is terminated after the current byte transfer is complete, but if the PFSAE bit in the QSCR register is set to 1, the **QSPI** will stop the pre-reading action instantly instead of waiting until the current byte transfer is complete. Using this feature requires that the serial flash device supports the immediate stop action feature.

28.6.4 Pre-read status monitoring

QSPI provides pre-read status monitoring to reduce this load.

In the Pre-Read Status Register QSSR, the PFAN bits show the current pre-reading status, the PFFUL bits indicate that the pre-reading data buffer is full, and PFNUM[4:0] shows the number of bytes of data that have been read into the buffer so far. These status bits can be easily used to determine the current pre-reading status by a CPU instruction.

Caution:

- When executing a pre-reading status monitoring program, please place the program code outside the object serial flash area or enable the instruction cache. Otherwise, the pre-reading object will frequently switch between the object data area and the instruction area, losing the meaning of pre-reading, and the monitoring program will enter an infinite loop because the pre-reading never completes.

28.6.5 Flash reads using **QSPI** bus cycle extension

If SSNW[1:0] in the QSCSCR register is set to a value other than 00, the QSPI bus cycle will be held after receiving data and waiting for the next data to be read. The QSSN signal will remain active low and QSCK will be stopped. If the next flash read instruction arrives, the QSPI will restart the QSCK to accept data directly if the address of the read object is sequentially incremented immediately after the current address. If the address is

non-sequential, the QSSN signal is set to an invalid high state to end the previously held QSPI bus cycle and restart a new SPI bus cycle.

This feature improves read efficiency by preventing the system from repeatedly sending instruction codes and addresses when discontinuous reads are performed on successively incrementing addresses.

The **QSPI bus cycle** extension time can be set via SSNW[1:0].

When the next read does not occur after the set time, QSSN will automatically set to a high invalid state, ending the **QSPI bus cycle**. If SSNW[1:0] is set to 11, QSSN will be extended indefinitely and the **QSPI bus cycle** will always be in the hold state, but this will increase the power consumption of the serial flash.

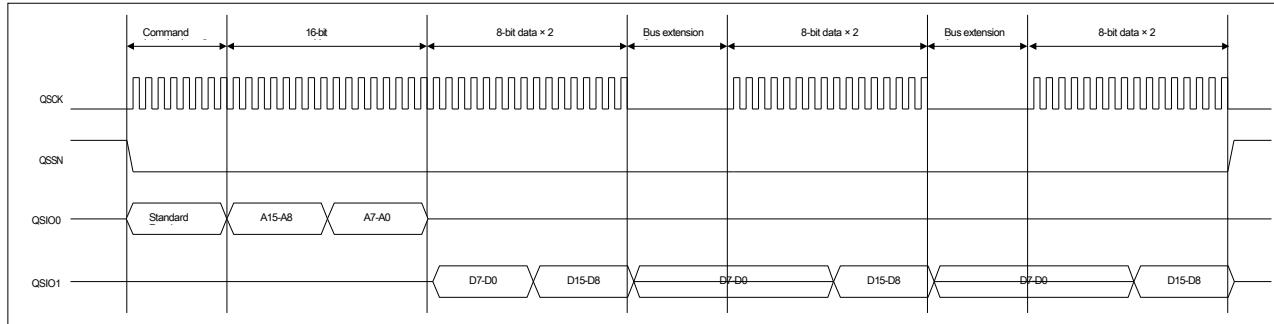


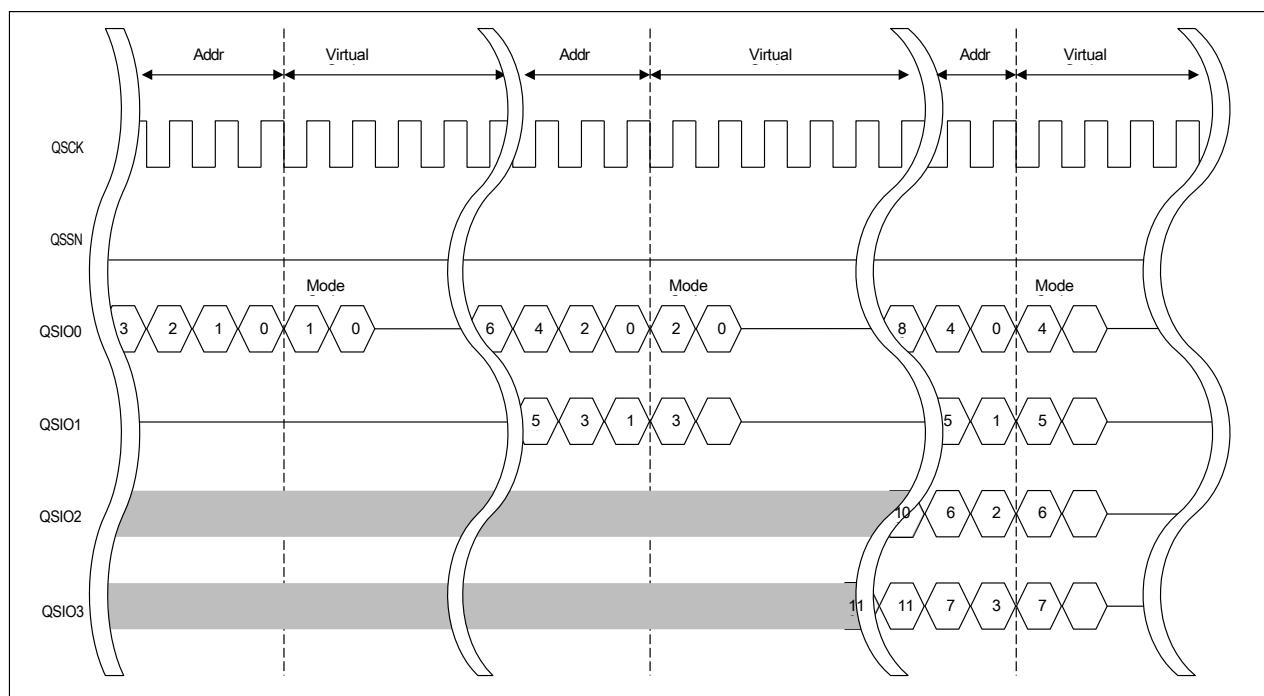
Figure 28-29 Schematic of Data Read Operation Using QSPI Bus Cycle Extension Function

28.7 XIP

Control

Some serial flash devices can reduce latency time by omitting the receive read instruction. This capability can be selected by the mode code sent during the virtual cycle.

The QSPI controls the **XIP** mode of the serial flash by sending the **XIP** mode code during the first two cycles of the virtual cycle during fast speed instructions. The **XIP mode code** varies from serial flash to serial flash and can be set using the XIPMC[7:0] bits of the QSXCMD register. Refer to Figure 28-30 below for details.



① Extending the SPI protocol with fast read instructions
②Extended SPI protocol with 2-wire

①2-wire SPI protocol
②Extended SPI protocol with 2-wire

①Four-wire SPI protocol
②Extended SPI protocol with four-wire

Figure 28-30 XIP mode control schematic

28.7.1 XIP mode settings

If the XIP mode code corresponding to the serial flash is written to XIPMC[7:0] of register QSXCMD and the XIPE bit of register QSCR is set to 1, when the next fast read instruction occurs, the set mode code is transmitted to the object serial flash during the first two cycles of the virtual cycle, and after the mode code is received, the serial flash and its control section start the XIP mode. You can verify that XIP mode has been entered by accessing the XIPF bit of the QSSR.

Caution:

- Starting the XIP mode of the serial flash requires setting the corresponding mode code in QSXCMD[7:0]. The XIP mode of the control section only requires setting the XIPE position to 1, independent of the value of QSXCMD[7:0].

28.7.2 XIP mode exit

If the code corresponding to the exit XIP mode of the serial flash is written to XIPMC[7:0] of register QSXCMD and the XIPE bit of register QSCR is set to 0, when the next fast read instruction occurs, the set exit mode code is transmitted to the object serial flash during the first two cycles of the virtual cycle, and after the exit mode code is received, the XIP mode of the serial flash and its control section is terminated. It is possible to verify that the XIP mode has been exited by accessing the XIPF bit of the QSSR.

Caution:

- Exiting the XIP mode of the serial flash requires setting the appropriate exit mode code in QSXCMD[7:0]. The XIP mode of the control section only requires clearing the XIPE bits to zero, independent of the value of QSXCMD[7:0].

28.8 QSIO2 and QSIO3 pin status

The QSIO2 and QSIO3 pin states depend on the serial read mode set by the MDSEL[2:0] bits in the QSCR register.

Table 28-5 Pin Status of QIO2 and QIO3

QSCR Register MDSEL[2:0] bits	QSIO2 Status	QSIO3 Status	Re mar ks
000			Standard read (initial state)
001			Quick Read
010			2-wire output quick read
011	Output status, the output level is determined by the WPOL bit of QSFCR register, the initial output is low	Output high level	2-wire input/output quick read
100			Four-wire output quick read
101	As the third data line for input or output action, the standby state is Hi-Z	As the fourth data line for input or output action, the standby state is Hi-Z	Four-wire input/output quick read
110			Custom Protocol Standard Read
111	Refer to the specific protocol settings for each phase	Refer to the specific protocol settings for each phase	Custom Protocol Quick Read

Caution:

- The QSIO2 pin can also be used as the WP# function for serial flash.
- The QSIO3 pin can also be used as a serial flash HOLD# or RESET# function.

28.9 Direct communication mode

28.9.1 About direct communication mode

QSPI can read serial flash by automatically converting the MCU's external ROM read bus cycle to a QSPI bus cycle. However, serial flash has many different additional functions, such as ID reading, erase, write and status reading. There is no standard set of instructions for setting these functions, and as new functions are rapidly added to the Serial Flash, it becomes increasingly difficult to cope with them at the hardware level.

For this case, QSPI provides a direct communication mode that allows the user to control the serial flash directly through software. This mode allows the software to generate as many QSPI bus cycles as desired.

28.9.2 Setting of direct communication mode

Setting the DCOME bit of the QSCR register to 1 allows you to enter the direct communication mode. Once in direct communication mode, normal flash read operations will not be possible, and to perform regular flash reads, you will need to clear the DCOME bit to exit direct communication mode.

Caution:

- If you are in XIP mode, you need to exit XIP mode before starting direct communication mode.

28.9.3 Generation of **QSPI** bus cycles in direct communication mode

A complete **QSPI** bus cycle in direct communication mode starts with the first operation on DCOM[7:0] of register QSDCOM and ends with a write operation on the QSCR register. Multiple operations can be performed on DCOM[7:0] during this period, with a write to DCOM[7:0] **converting to** a single byte of data transfer on the **QSPI** bus, and a read to DCOM[7:0] converting to a single byte of data reception on the QSPI bus.

The QSSN signal remains active low from the first operation on DCOM[7:0] of register QSDCOM until the last write operation on the QSCR register.

Direct communication mode does

not support multilinear actions.

Caution:

- Write operations to registers other than QSCR and QSDCOM are not available in direct communication mode. For the other registers

The write operation will exit the direct communication mode. Using this method of exit may lead to unpredictable situations so it is not recommended.

28.10 Interruptions

When a read access to ROM is detected in direct communication mode, the RAER bit of the QSSR register is set to 1. The QSPI will generate a bus hardware error interrupt. This interrupt request will be held until the RAER bit is cleared. Please refer to [Interrupt Controller (INTC) details].

28.11 Precautions for use

28.11.1 QSPI Register Setting Order

The QSPI control registers can be dynamically set or changed during system operation. However, not paying attention to the order in which the registers are set may cause the QSPI bus cycle to start before the registers are fully set, so please configure the register setting order carefully to avoid such situations.

28.11.2 Setting of module stop signal

The QSPI is in the module stop state after system reset, and the register can only be set if the **QSPI** module stop signal in the module stop control register is cleared to zero. For details, please refer to 5.5 Ways to Reduce Power Consumption.

28.12 Register Description

Register base address: 0x9C00_0000

Table 28-6 QSPI Register List

Register Name	Offset Address	Reset value
Control register QSCR	0x0000	0x003F_0000
Chip Select Control Register QSCSCR	0x0004	0x0000_000F
Format Control Register QSFCR	0x0008	0x0000_8033
Status Register QSSR	0x000C	0x0000_8000
Direct communication command register QSDCOM	0x0010	-
Instruction code register QSCCMD	0x0014	0x0000_0000
XIP mode code register QSXCMD	0x0018	0x0000_00FF
Flag clear register QSSR2 (write only)	0x0024	-
External address register QSEXAR	0x0804	0x0000_0000

28.12.1 QSPI Control Register (QSCR)

Reset value: 0x003F_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DIV[5:0]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	DPRSL[1: 0]	APRSL[1: 0]	IPRSL[1: 0]	SPIMD3	XIPE	DCOME	PFSAE	PFE	MDSEL[2: 0]					

position	Marker	Place Name	Function	Read ding and writin g
b31~b21	Reserved	-	Read "0", write "0" when writing	R/W

b20~b16	DIV[5:0]	Base clock selection bit	Serial interface reference clock selection b5 b4 b3 b2 b1 b0	R/W
			0 0 0 0 0 0 0: 2 HCLK cycles	
			0 0 0 0 0 0 1: 2 HCLK cycles*	
			0 0 0 0 0 1 0: 3 HCLK cycles	
			0 0 0 0 0 1 1: 4 HCLK cycles*	
			0 0 0 0 1 0 0: 5 HCLK cycles	
			0 0 0 0 1 0 1: 6 HCLK cycles*	
			0 0 0 0 1 1 0: 7 HCLK cycles	
			0 0 0 0 1 1 1: 8 HCLK cycles*	
			0 0 1 0 0 0 0: 9 HCLK cycles	
			0 0 1 0 0 0 1: 10 HCLK cycles*	
			0 0 1 0 1 0: 11 HCLK cycles	
			0 0 1 0 1 1: 12 HCLK cycles*	
			0 0 1 1 0 0: 13 HCLK cycles	
			0 0 1 1 0 1: 14 HCLK cycles*	
			0 0 1 1 1 0: 15 HCLK cycles	
			0 0 1 1 1 1: 16 HCLK cycles*	
			0 1 0 0 0 0 0: 17 HCLK cycles	
			0 1 0 0 0 0 1: 18 HCLK cycles	
			0 1 0 0 1 0: 19 HCLK cycles	
			0 1 0 0 1 1: 20 HCLK cycles	
			0 1 0 1 0 0: 21 HCLK cycles	
			0 1 0 1 0 1: 22 HCLK cycles	
			0 1 0 1 1 0: 23 HCLK cycles	
			0 1 0 1 1 1: 24 HCLK cycles	
			0 1 1 0 0 0 0: 25 HCLK cycles	
			0 1 1 0 0 0 1: 26 HCLK cycles	
			0 1 1 0 1 0: 27 HCLK cycles	
			0 1 1 0 1 1: 28 HCLK cycles	
			0 1 1 1 1 0 0: 29 HCLK cycles	
			0 1 1 1 1 0 1: 30 HCLK cycles	
			0 1 1 1 1 1 0: 31 HCLK cycles	
			0 1 1 1 1 1 1: 32 HCLK cycles	
			1 0 0 0 0 0 0 0: 33 HCLK cycles	
			1 0 0 0 0 0 0 1: 34 HCLK cycles*	
			1 0 0 0 0 1 0: 35 HCLK cycles	

1 0 0 0 0 1 1: 36 HCLK cycles*
 1 0 0 1 0 0: 37 HCLK cycles
 1 0 0 1 0 1: 38 HCLK cycles*
 1 0 0 1 1 0: 39 HCLK cycles
 1 0 0 1 1 1: 40 HCLK cycles*
 1 0 1 0 0 0: 41 HCLK cycles
 1 0 1 0 0 1: 42 HCLK cycles*
 1 0 1 0 1 0: 43 HCLK cycles
 1 0 1 0 1 1 1: 44 HCLK cycles*
 1 0 1 1 1 0 0: 45 HCLK cycles
 1 0 1 1 1 0 1: 46 HCLK cycles*
 1 0 1 1 1 1 0: 47 HCLK cycles
 1 0 1 1 1 1 1: 48 HCLK cycles*
 1 1 0 0 0 0 0: 49 HCLK cycles
 1 1 0 0 0 0 1: 50 HCLK cycles
 1 1 0 0 1 0: 51 HCLK cycles
 1 1 0 0 1 1: 52 HCLK cycles
 1 1 0 1 0 0: 53 HCLK cycles
 1 1 0 1 0 1: 54 HCLK cycles
 1 1 0 1 1 1 0: 55 HCLK cycles
 1 1 0 1 1 1 1: 56 HCLK cycles
 1 1 1 1 0 0 0 0: 57 HCLK cycles
 1 1 1 1 0 0 1: 58 HCLK cycles
 1 1 1 1 0 1 0: 59 HCLK cycles
 1 1 1 1 0 1 1 1: 60 HCLK cycles
 1 1 1 1 1 0 0: 61 HCLK cycles
 1 1 1 1 1 0 1: 62 HCLK cycles
 1 1 1 1 1 1 0: 63 HCLK cycles
 1 1 1 1 1 1 1: 64 HCLK cycles

b15~b14	Reserved	-	Read "0", write "0" when writing	R/W
b13~b12	DPRSL[1:0]	Data reception phase SPI protocol selection	Data reception phase SPI protocol selection. b1 b0 0 0: Extended SPI protocol 0 1: 2-wire SPI protocol 1 0: Four-wire SPI protocol 1 1: Set the ban	R/W
b11~b10	APRSL[1:0]	Address sending phase SPI protocol selection	Address sending phase SPI protocol selection. b1 b0 0 0: Extended SPI protocol 0 1: 2-wire SPI protocol 1 0: Four-wire SPI protocol 1 1: Set the ban	R/W
b9~b8	IPRSL[1:0]	Command sending phase SPI protocol selection	Command send phase SPI protocol selection. b1 b0 0 0: Extended SPI protocol 0 1: 2-wire SPI protocol 1 0: Four-wire SPI protocol 1 1: Set the ban	R/W

b7	SPIMD3	SPI Mode Selection	SPI mode selection 0: 0: SPI mode 0 1: SPI mode 3	R/W
b6	XIPE	XIP mode licensing	0: XIP mode disabled 1: XIP mode license	R/W
b5	DCOME	Direct Communication License	QSPI bus communication mode selection 0: ROM access mode 1: Direct communication mode	R/W
b4	PFSAE	Pre-read instant stop license	Select the position to reset the pre-read action 0: The current pre-read action is aborted at the byte boundary 1: The current pre-read action is aborted instantly	R/W
b3	PFE	Pre-Read License	Pre-read function valid/invalid selection 0: 0: Pre-read function is valid 1: Pre-read function is effective	R/W
b2~b0	MDSEL[2:0]	QSPI Read Mode Selection	Serial interface read mode selection b2 b1 b0 0 0 0: Standard reading 0 0 1: Quick Read 0 1 0: 2-wire output quick read 0 1 1: 2-wire input/output quick read b1 1 0 0: Four-wire output fast read 1 0 1: Four-wire input and output quick read 1 1 0: Custom standard read 1 1 1: Custom Quick Read	R/W

28.12.2 QSPI Chip Select Control Register (QSCSCR)

Reset value: 0x0000_000F

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	SSNW[1:0]	SSHW[3:0]				

position	Marker	Place Name	Function	Reading and writing
b31~b6	Reserved	-	Read "1", write "0" when writing	R/W
b5~b4	SSNW[1:0]	QSSN valid time extension setting	QSSN valid time extension function selection after QSPI bus access b5 b4 0 0: No extension of QSSN effective time 0 1: Extend the QSSN valid time by 32 QSCK cycles 1 0: Extend the QSSN valid time by 128 QSCK cycles 1 1: Extend QSSN valid time	R/W

b3	b2	b1	b0
0	0	0	0: 1 QSCK cycle
0	0	0	1: 2 QSCK cycles
0	0	1	0: 3 QSCK cycles
0	0	1	1: 4 QSCK cycles
0	1	0	0: 5 QSCK cycles
0	1	0	1: 6 QSCK cycles
0	1	1	0: 7 QSCK cycles
0	1	1	1: 8 QSCK cycles
1	0	0	0: 9 QSCK cycles
1	0	0	1: 10 QSCK cycles
1	0	1	0: 11 QSCK cycles
1	0	1	1: 12 QSCK cycles
1	1	0	0: 13 QSCK cycles
1	1	0	1: 14 QSCK cycles
1	1	1	0: 15 QSCK cycles
1	1	1	1: 16 QSCK cycles

28.12.3 QSPI Format Control Register (QSFCR)

Reset value: 0x0000_8033

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DUTY	-	-	-	DMCYCN [3:0]	-	WPOL	SSNLD	SSNHD	-	Four_BIC	AWSL[1:0]				

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Read "0", write "0" when writing	R/W
b15	DUTY	Duty Cycle Correction	QSCK output waveform duty cycle correction 0: No duty cycle correction 1: Hysteresis the rising edge of QSCK for 0.5 HCLK cycles (valid when QSCK selects a frequency that is an odd multiple of HCLK)	R/W
b14~b12	Reserved	-	Read "0", write "0" when writing	R/W
b11~b8	DMCYCN [3:0]	Virtual Cycle Setup	Number of virtual cycles selected when using fast read instructions b3 b2 b1 b0 0 0 0 0: 3 QSCK cycles*1 0 0 0 1: 4 QSCK cycles 0 0 1 0: 5 QSCK cycles 0 0 1 1: 6 QSCK cycles 0 1 0 0: 7 QSCK cycles 0 1 0 1: 8 QSCK cycles 0 1 1 0: 9 QSCK cycles 0 1 1 1: 10 QSCK cycles 1 0 0 0 0: 11 QSCK cycles 1 0 0 1: 12 QSCK cycles 1 0 1 0: 13 QSCK cycles 1 0 1 1: 14 QSCK cycles 1 1 0 0: 15 QSCK cycles 1 1 0 1: 16 QSCK cycles 1 1 1 0: 17 QSCK cycles 1 1 1 1: 18 QSCK cycles	R/W
b7	Reserved	-	Read "0", write "0" when writing	R/W
b6	WPOL	WP pin output level setting	WP pin (QIO2) level setting 0: low 1: High level	R/W
b5	SSNLD	QSSN signal output time delay setting	QSSN Signal Output Timing Selection 0: 0.5 QSCK output QSSN ahead of the first rising edge of QSCK 1: QSSN output 1.5 QSCKs ahead of the first rising edge of QSCK	R/W

b4	SSNHD	QSSN signal release time delay setting	QSSN Signal Release Timing Selection 0: 0.5 QSCK lags behind the last rising edge of QSCK to release QSSN 1: Release QSSN 1.5 QSCK lags behind the last rising edge of QSCK	R/W
b3	Reserved	-	Read "1", write "0" when writing	R/W
b2	Four_BIC	4-byte address read instruction code selection	Read instruction code selection when address width is 4 bytes 0: Do not use 4-byte address to read instruction code	R/W

1: Read instruction code using 4-byte address				
b1~b0	AWSL[1:0]	Address width selection	Serial interface address width selection b1 b0	R/W
			0 0: 1 byte	
			0 1: 2 bytes	
			1 0: 3 bytes	
			1 1: 4 bytes	

*1: To avoid conflicts when the QIO0 terminal switches input and output states, select more than 4 QSPICK cycles as virtual cycles if the QSSMD_QSOEX bit is set to 1 (processing permission signal is extended by 1 cycle)

28.12.4 QSPI Status Register (QSSR)

Reset value: 0x0000_8000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PFAN	PFFUL	-	PFNUM[4:0]				RAER	XIPF	-	-	-	-	-	-	BUSY

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	Read "0", write "0" when writing	R/W
b15	PFAN	Pre-read action status	Pre-read action status signal 0: Pre-read in action status 1: Pre-reading is stopped	R
b14	PFFUL	Pre-read buffer status	Pre-read buffer status signal 0: Pre-read buffer has space left 1: Pre-read buffer data is full	R
b13	Reserved	-	Read "0", write "0" when writing	R/W
b12~b8	PFNUM[4:0]	Number of bytes of data already stored in the pre-read buffer	The number of bytes of data stored in the pre-read buffer is displayed b4 b3 b2 b1 b0 0 0 0 0 0 0: 0 bytes 0 0 0 0 0 1: 1 byte 0 0 0 0 1 0: 2 bytes 0 0 0 0 1 1: 3 bytes 0 0 1 0 0 0: 4 bytes 0 0 1 0 1: 5 bytes 0 0 1 1 0: 6 bytes 0 0 1 1 1: 7 bytes 0 1 0 0 0: 8 bytes 0 1 0 0 1: 9 bytes 0 1 0 1 0: 10 bytes 0 1 0 1 1: 11 bytes 0 1 1 0 0: 12 bytes 0 1 1 0 1: 13 bytes 0 1 1 1 0: 14 bytes 0 1 1 1 1: 15 bytes 1 0 0 0 0 0: 16 bytes 1 0 0 0 0 1: 17 bytes 1 0 0 1 0: 18 bytes the rest of the setting is invalid	R
b7	RAER*1	ROM access error flag	Error flag bit for ROM access occurring in direct communication mode 0: ROM access not detected 1: ROM access is detected to occur	R/W

b6	XIPF	XIP mode logo	XIP mode status signal 0: Non- XIP mode 1: XIP mode	R
b5~b1	Reserved	-	Read "1", write "0" when writing	R/W

b0	BUSY	Bus busy sign	QSPI bus operating status flag bit in direct communication mode 0: bus idle, no serial transmission process 1: Bus busy, serial transfer process in progress	R
----	------	---------------	---	---

*1: RAER needs to be cleared by the RAERCLR bit of QSSR2

28.12.5 QSPI Command Code Register (QSCCMD)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RIC [7:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Read "0", write "0" when writing	R/W
b7~b0	RIC[7:0]	Replace command code	Serial flash instruction code to replace the default instruction	R/W

28.12.6 QSPI Direct Communication Command Register (QSDCOM)

Reset value: None

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DCOM [7:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Read "0", write "0" when writing	R/W
b7~b0	DCOM [7:0]	Direct communication mode command	The interface in direct communication mode communicates directly over the QSPI bus. Read and write accesses to this interface are translated into a corresponding QSPI bus cycle. This interface is only valid in direct communication mode, and access to this interface is disabled in ROM access mode.	R/W

28.12.7 QSPI XIP Mode Code Register (QSXCMD)

Reset value: 0x0000_00FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
XIPMC [7:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Read "0", write "0" when writing	R/W
b7~b0	XIPMC[7:0]	XIP mode code	Mode code for serial flash memory. (Set XIP mode)	R/W

28.12.8 QSPI System Configuration Register (QSSR2)

Reset value: None

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	RAERCLR	-	-	-	-	-	-	-

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Write "1" when writing in	W
b7	RAERCLR	RAER Clearance	Clear the RAER bit in QSSR when writing 1	W
b6~b0	Reserved	-	Write "1" when writing in	W

28.12.9 QSPI External Extended Address Register (QSEXAR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EXADR[5: 0]								-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

position	Marker	Place Name	Function	Reading and writing
b31~b26	EXADR[5:0]	External extension address code	QSPI external address high 6-bit setting, with QSPI ROM access window address can access a maximum of 64MB x 63 blocks of external ROM space	R/W
b25~b0	Reserved	-	Read "1", write "1" when writing	R/W

29 IC built-in audio bus module (I2S)

29.1 Introduction

I2S (Inter_IC Sound Bus) is the integrated circuit's built-in audio bus, which is dedicated to data transfer between audio devices.

Table 29-1 I2S main characteristics

Function	Main Features
Communication method	<ul style="list-style-type: none">- Supports full-duplex and half-duplex communications- Supports master mode or slave mode operation
Data Format	<ul style="list-style-type: none">- Optional channel length: 16/32 bit- Optional transmission data length: 16/24/32 bits- Data shift order: MSB start
Baud rate	<ul style="list-style-type: none">- 8-bit programmable linear prescaler for accurate audio sampling frequency- Support sampling frequency 192k, 96k, 48k, 44.1k, 32k, 22.05k, 16k, 8k- Can output drive clock to drive external audio components with a fixed ratio of 256 x Fs (Fs is the audio sampling frequency)
I2S protocol support	<ul style="list-style-type: none">- I2S Philips standard- MSB alignment standards- LSB alignment standards- PCM Standard
Data buffering	<ul style="list-style-type: none">-With 2-word deep, 32-bit wide input and output FIFO buffer areas
Clock source	<ul style="list-style-type: none">- Internal I2SCLK (UPLL/R/UPLL/Q/UPLL/P/MPLL/R/MPLL/Q/MPLL/P) can be used; it can also be used byThe external clock on the I2S_EXCK pin provides
Interruptions	<ul style="list-style-type: none">- Generate an interrupt when the effective space in the transmit buffer reaches the alarm threshold- Generate an interrupt when the effective space in the receive buffer reaches the alarm threshold- The receive data area is full and there is still a write data request, receive overflow- Sending data area is empty and there are still sending requests, send the next overflow- Send data area is full and still has write data request, send overflow

29.2 I2S System Block Diagram

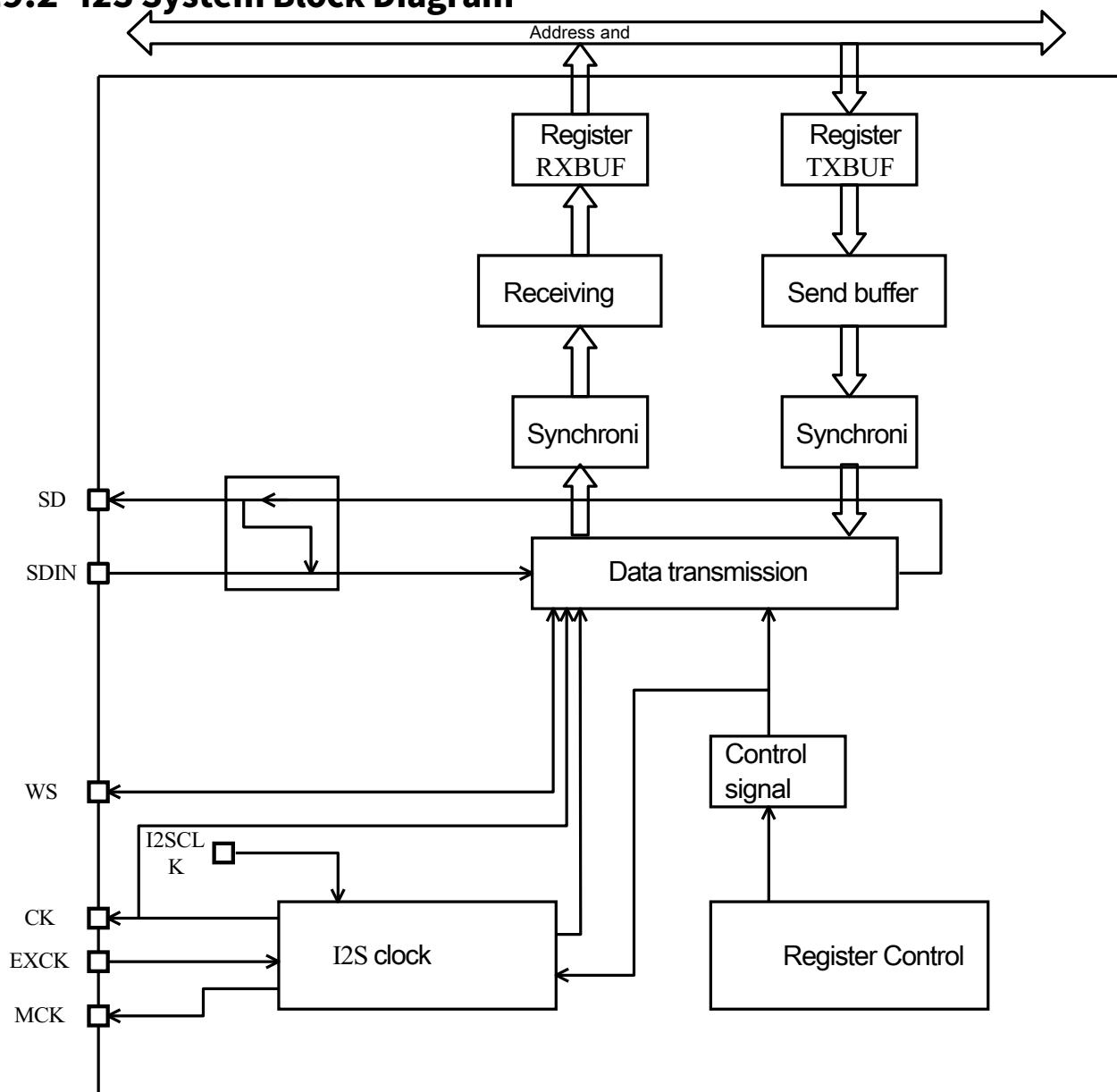


Figure 29-1 I2S system block diagram

29.3 Pin

Descript

Table 29-2 I2S Pin Descriptions

ion

Foot r Name	Dire ctio n	Func n Descript ion
I2Sn_CK	Input and Output	Communication Clock
I2Sn_WS	Input and Output	Word selection
I2Sn_SD	Input and Output	Serial Data
I2Sn_SDIN	Input	Full duplex audio data input
I2Sn_EXCK	Input	External clock source pin
I2Sn_MCK	Output	Drive Clock

n:1~4

29.4 Function Description

This chapter will explain the functions of I2S in detail.

29.4.1 I2S General Description

I2S Pin Function

- I2Sn_SD: Serial data for half duplex mode data input, or half/full duplex mode data output.
- I2Sn_WS: word selection, which is the data control signal output in master mode and the data control signal input in slave mode.
- I2Sn_CK: Serial clock, which is the serial clock output in master mode and the serial clock input in slave mode.
- I2Sn_EXCK: external clock source, is the main mode clock generator to select the external clock as the frequency division clock source.
- I2Sn_SDIN: Pin for serial data input in I2S full duplex mode.
- I2Sn_MCK: When I2S is configured as master mode (and MCKOE position 1), this additional clock is output using the drive clock (individually mapped) at $256 \times F_s$, where F_s is the audio signal sampling frequency.

The I2S uses its own clock generator to generate the communication clock in master mode. This clock generator is also the source that drives the clock output.

29.4.2 Communication method

Both half-duplex and full-duplex communication methods are supported and can be selected via the DUPLEX bit of the I2S control register (I2S_CTRL).

When the I2S_CTRL.DUPLEX bit is 0, the I2S operates in half-duplex communication mode, using the I2Sn_SD pin as the output data pin for transmit only and the I2Sn_SD pin as the input data pin for receive only.

When the I2S_CTRL.DUPLEX bit is 1, the I2S operates in full-duplex communication mode, where the I2Sn_SDIN pin is used as the input data pin and the I2Sn_SD pin is used as the output data pin to achieve full-duplex communication.

29.4.3 Supported audio protocols

Four combinations of data and frame formats are available, and data can be sent in the following formats:

- Encapsulating 16-bit data in 16-bit frames
- Encapsulating 16-bit data in 32-bit frames
- Encapsulating 24-bit data in 32-bit frames
- Encapsulating 32-bit data in 32-bit frames

When using 16-bit data from a 32-bit packet, the first 16 bits (MSB) are valid and the 16-bit LSB is forced to zero without any software operation (only one read/write operation is required)

When using 24 bits of data from a 32-bit packet, the first 24 bits (MSB) are valid and the 8-bit LSB is forced to zero without any software operation (only one read/write operation is required)

For all data formats and communication standards, the highest significant bit will always be sent first (MSB priority)

The I2S interface supports four audio protocols, which can be configured using the I2SSTD[1:0] and PCMSYNC bits in the I2S_CFGR register.

29.4.3.1 I2S Philips Standard

The WS signal is used to indicate the channel to which the data is currently being sent. This signal is valid from one clock before the first bit of the current channel data (MSB).

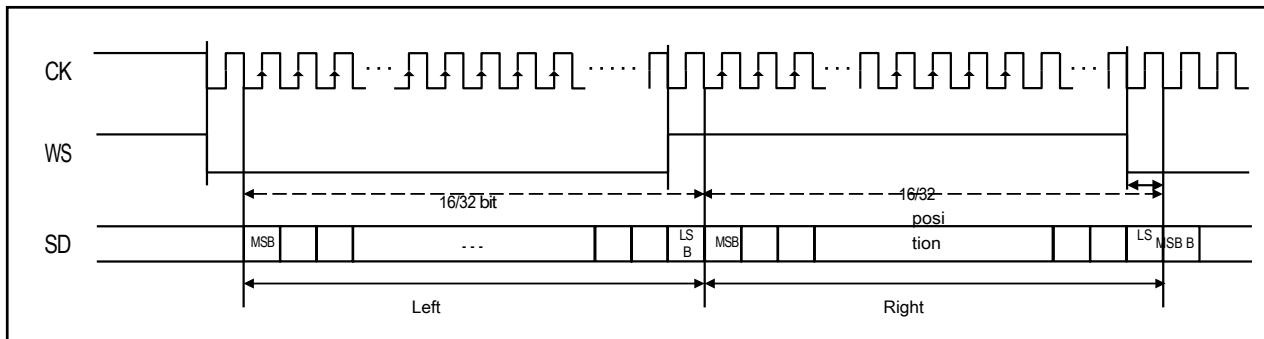


Figure 29-2 I2S Philips protocol waveform (16/32-bit full precision)

16-bit loaded in 16-bit frame, in transmit mode, write to I2S_TXBUF register 0xXXXX_3344, SD output serial data 0x3344 in receive mode\$ D input serial data 0xEEDD|I2S_RXBUF register read data 0x0000_EEDD.

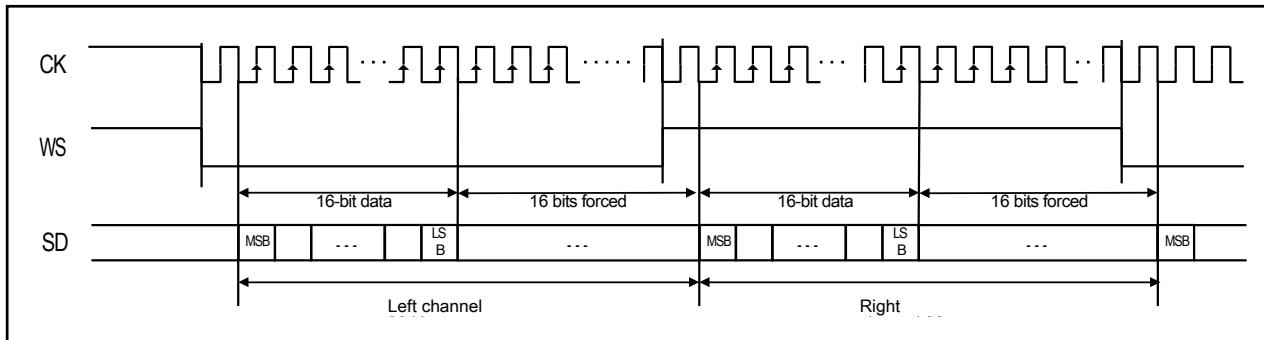


Figure 29-3 I2S Philips protocol waveform (16-bit data encapsulated in a 32-bit frame)

16-bit loaded in 32-bit frame, in transmit mode, write to I2S_TXBUF register 0xXXXX_3344, SD output serial data 0x3344_0000; in receive mode, SD input serial data 0xEEDD_XXXX,I2S_RXBUF register read data 0x0000_EEDD.

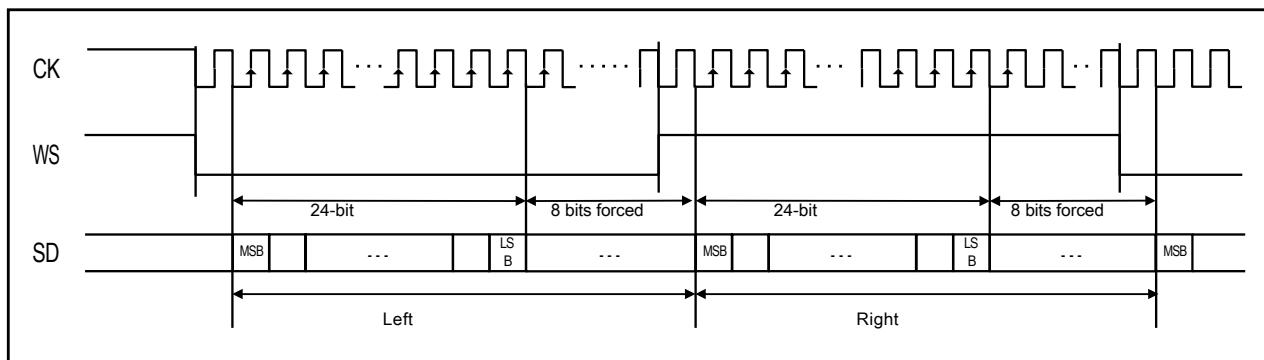


Figure 29-4 I2S Philips protocol waveform (24-bit data encapsulated in a 32-bit frame)

24-bit loaded in 32-bit frame, in transmit mode, written to I2S_TXBUF register 0xXX22_3344, SD output serial data 0x2233_4400; in receive mode, SD input serial data 0xEEDD_11XX,I2S_RXBUF register read out data 0x00EE_DD11.

29.4.3.2 MSB Alignment Standards

This standard generates both the WS signal and the first data bit (i.e. **MSB**it)

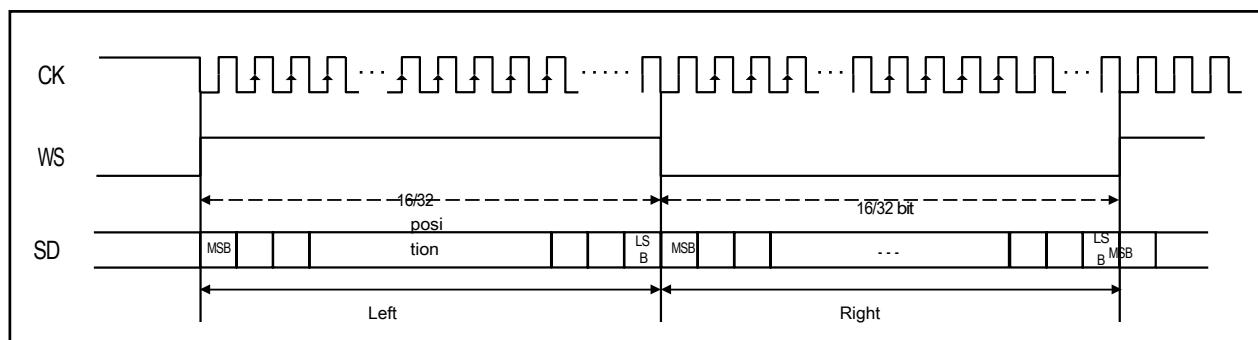


Figure 29-5 I2S MSB protocol waveform (16/32-bit full precision)

The sender changes the data on the falling edge of the clock signal; the receiver reads the data on the

rising edge.

16-bit loaded in 16-bit frame, in transmit mode, write to I2S_TXBUF register 0xXXXX_3344, SD output serial data 0x3344 in receive mode\$D input serial data 0xEEDD, I2S_RXBUF register read data 0x0000_EEDD.

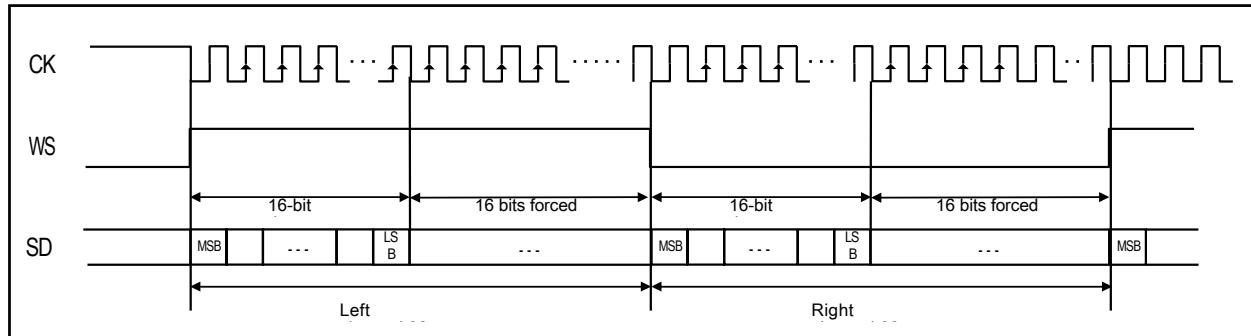


Figure 29-6 I2S MSB protocol waveform (16-bit data encapsulated in a 32-bit frame)

16-bit loaded in 32-bit frame, in transmit mode, write to I2S_TXBUF register 0xXXXX_3344, SD output serial data 0x3344_0000; in receive mode, SD input serial data 0xEEDD_XXXX, I2S_RXBUF register read data 0x0000_EEDD.

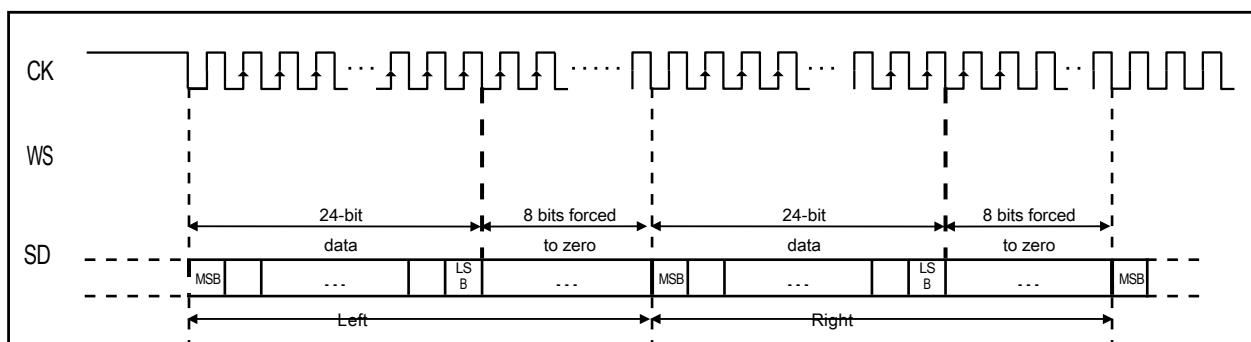


Figure 29-7 I2S MSB protocol waveform (24-bit data encapsulated in a 32-bit frame)

24-bit loaded in 32-bit frame, in transmit mode, written to I2S_TXBUF register 0xXX22_3344, SD output serial data 0x2233_4400; in receive mode, SD input serial data 0xEEDD_11XX, I2S_RXBUF register read out data 0x00EE_DD11.

29.4.3.3 LSB Alignment Standards

The standard is similar to the MSB alignment standard (there is no difference between 16-bit and 32-bit full-precision frame formats)

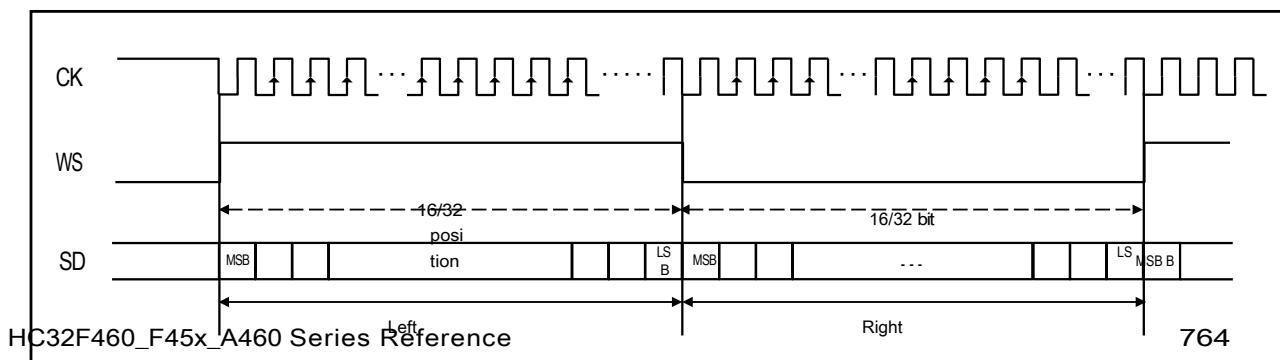


Figure 29-8 I2S LSB protocol waveform (16/32-bit full precision)

16-bit loaded in 16-bit frame, in transmit mode, write to I2S_TXBUF register 0xXXXX_3344, SD output serial data 0x3344; in receive mode, SD input serial data 0xEEDD; I2S_RXBUF register read data 0x0000_EEDD.

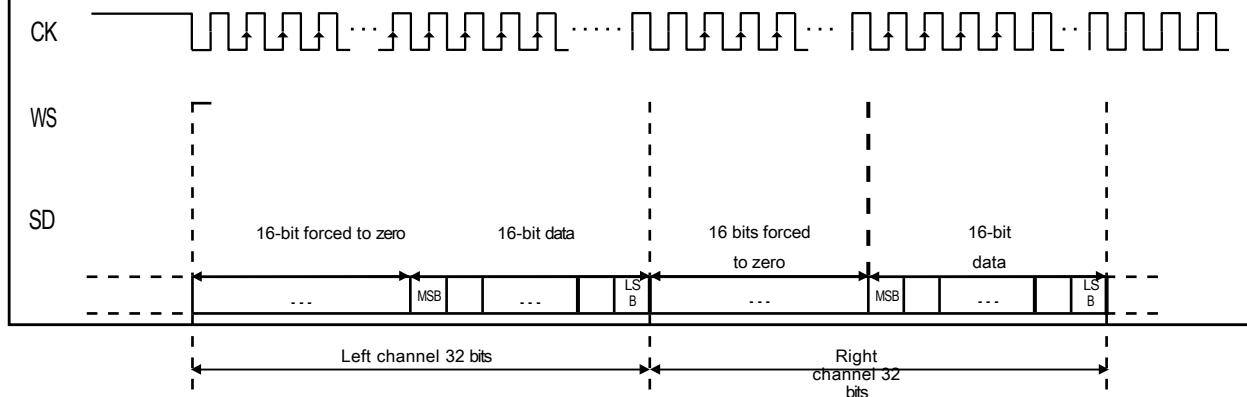


Figure 29-9 I2S LSB protocol waveform (16-bit data encapsulated in a 32-bit frame)

16-bit loaded in 32-bit frame, in transmit mode, write to I2S_TXBUF register 0xXXXX_3344, SD output serial data 0x0000_3344; in receive mode, SD input serial data 0xXXXX_1122, I2S_RXBUF register read out data 0x0000_1122.

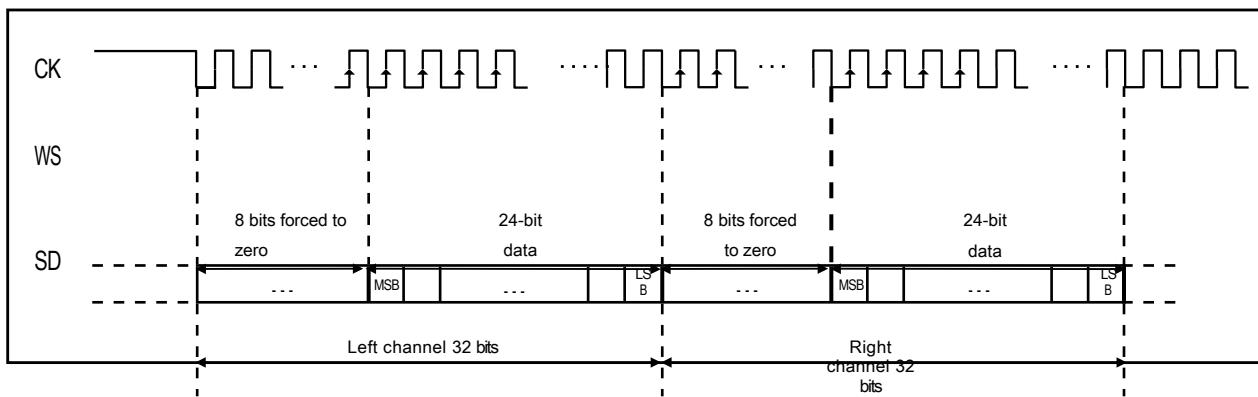


Figure 29-10 I2S LSB protocol waveform (24-bit data encapsulated in a 32-bit frame)

24-bit loaded in 32-bit frame, in transmit mode, write to I2S_TXBUF register 0xXX22_3344, SD output serial data 0x0022_3344; in receive mode, SD input serial data 0XXDD_1122, I2S_RXBUF register read out data 0x00DD_1122.

29.4.3.4 PCM Standards

For the PCM standard, no channel information is required. Two PCM modes (short frame and long frame) are available and can be configured using the PCMSYNC bit in I2S_CFGR.

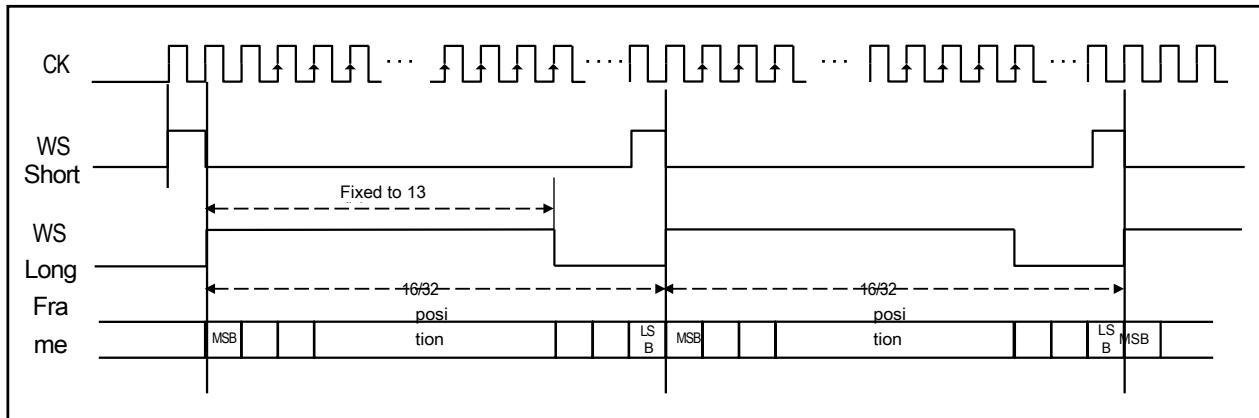


Figure 29-11 I2S PCM protocol waveform (16/32-bit full precision)

For long frame synchronization, the WS signal lasts for 13

cycles in master mode. For short frame synchronization,

the WS sync signal lasts for only one cycle.

16-bit loaded in 16-bit frame, in transmit mode, write to I2S_TXBUF register 0xXXXX_3344, SD output serial data 0x3344 in receive mode\$D input serial data 0xEEDDJ2S_RXBUF register read data 0x0000_EEDD.

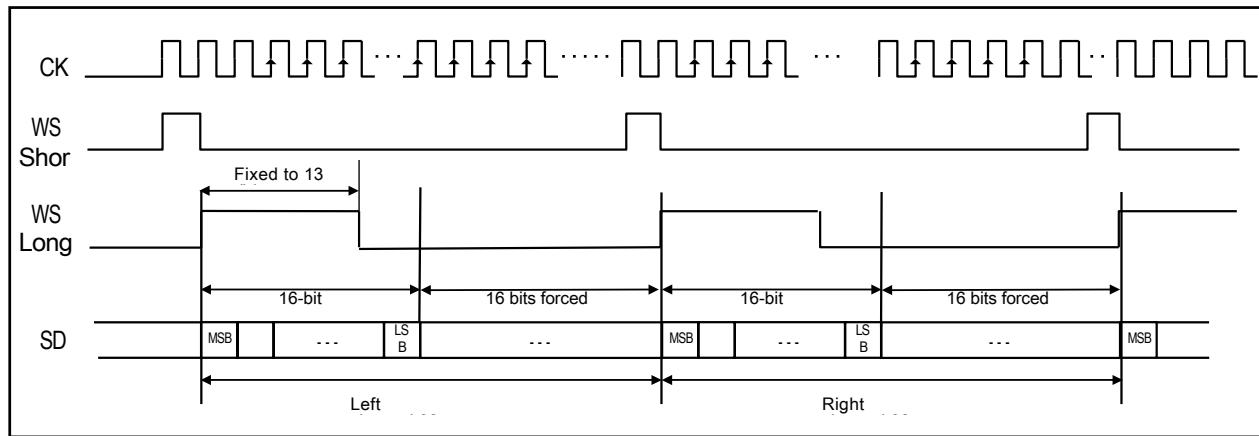


Figure 29-12 I2S PCM protocol waveform (16-bit data encapsulated in a 32-bit frame)

16-bit loaded in 32-bit frame, in transmit mode, write to I2S_TXBUF register 0xXXXX_3344, SD output serial data 0x0000_3344; in receive mode, SD input serial data 0xEEDD_XXXX,I2S_RXBUF register read out data 0x0000_EEDD.

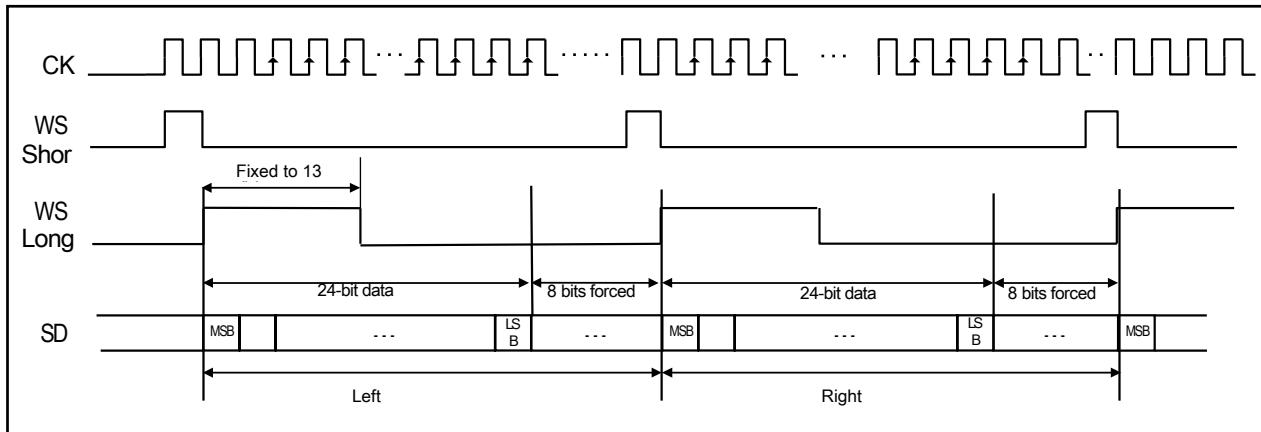


Figure 29-13 I2S PCM protocol waveform (24-bit data encapsulated in a 32-bit frame)

24-bit loaded in 32-bit frame, in transmit mode, written to I2S_TXBUF register 0xXX22_3344, SD output serial data 0x0022_3344; in receive mode, SD input serial data 0xEEDD_11XX,I2S_RXBUF register read out data 0x00EE_DD11.

Caution:

- For both modes (master/slave) and both synchronizations (short/long synchronization), the number of bits between two consecutive data sets (and two synchronization signals) needs to be specified (DATLEN and CHLEN bits in the I2S_CFGR register), even in slave mode.

29.4.4 Clock Generator

The I2S bit rate is used to determine the data stream on the I2S data line and the I2S clock signal frequency. I2S bit rate = number of bits per channel x number of channels x audio sampling frequency

For 16-bit dual-channel audio, the I2S bit rate is calculated as follows: I2S bit rate = $16 \times 2 \times F_s$. If the packet is 32 bits wide, the I2S bit rate = $32 \times 2 \times F_s$.

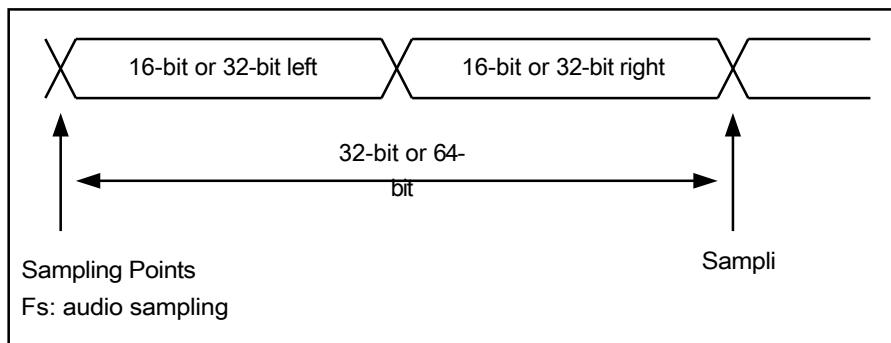


Figure 29-14 Audio Sampling Frequency Definition

When configuring the master mode, the linear divider needs to be set correctly so that the desired audio frequency is used for communication.

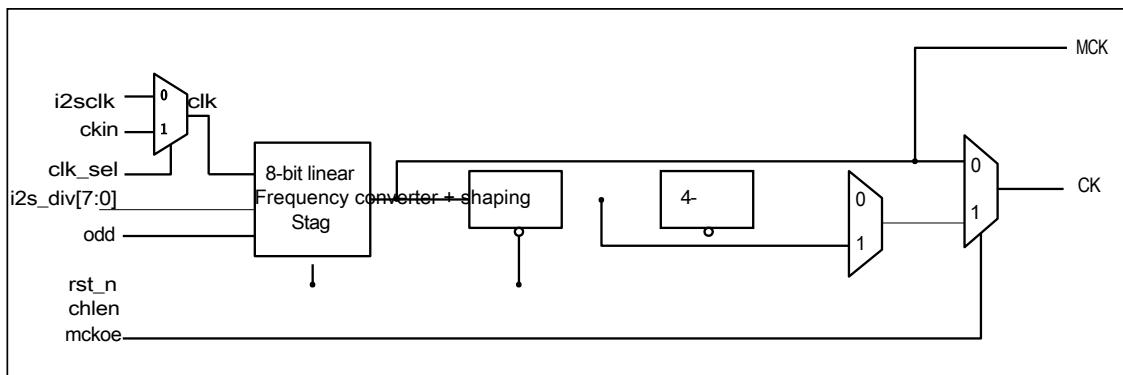


Figure 29-15 Clock Generator Architecture

The clk clock source can be either the I2SCLK output or an external clock.

The audio sampling frequency may be 192kHz, 96kHz or 48kHz, etc. To achieve the desired frequency, the current crossover needs to be programmed according to the following formula:

When the output drive clock (mckoe is set to 1):

$F_s = \text{clk} / [(16 \times 2) \times ((2 \times \text{I2SDIV} + \text{ODD}) \times 8)]$ (when the channel frame width is 16 bits)
 $F_s = \text{clk} / [(32 \times 2) \times ((2 \times \text{I2SDIV} + \text{ODD}) \times 4)]$ (when the channel frame width is 32 bits)

When turning off the drive clock (mckoe clear):

$F_s = \text{clk} / [(16 \times 2) \times (2 \times \text{I2SDIV} + \text{ODD})]$ (when the channel frame width is 16 bits)

$F_s = \text{clk} / [(32 \times 2) \times (2 \times \text{I2SDIV} + \text{ODD})]$ (when the channel frame width is 32 bits)

The following table provides sample accuracy values for different clock configurations. Other configurations are available to achieve better clock accuracy.

Table 29-3 Audio Frequency Accuracy (for VCO input frequency = 1MHz)

Drive Clock	Objectives fS(Hz)	Data Format	Frequency multiplication factor	Output Crossover Ratio	I2SDIV	ODD	Real-time fS(Hz)	Error
Output off	8000	16-bit	288	3	187	1	8000	0.0000%
		32-bit	256	4	62	1	8000	0.0000%
	16000	16-bit	256	4	62	1	16000	0.0000%
		32-bit	256	2	62	1	16000	0.0000%
	32000	16-bit	256	2	62	1	32000	0.0000%
		32-bit	256	5	12	1	32000	0.0000%
	48000	16-bit	384	10	12	1	48000	0.0000%
		32-bit	384	5	12	1	48000	0.0000%
	96000	16-bit	384	5	12	1	96000	0.0000%
		32-bit	424	3	11	1	96014.49219	0.0151%
	22050	16-bit	290	3	68	1	22049.87695	0.0006%
		32-bit	302	2	53	1	22050.23438	0.0011%
	44100	16-bit	302	2	53	1	44100.46875	0.0011%
		32-bit	429	4	19	0	44099.50781	0.0011%
	192000	16-bit	424	3	11	1	192028.9844	0.0151%
		32-bit	258	3	3	1	191964.2813	0.0186%
Output Enable	8000	Irrelevant	256	5	12	1	8000	0.0000%
	16000	Irrelevant	426	4	13	0	16000.60059	0.0038%
	32000	Irrelevant	426	4	6	1	32001.20117	0.0038%
	48000	Irrelevant	258	3	3	1	47991.07031	0.0186%
	96000	Irrelevant	344	2	3	1	95982.14063	0.0186%
	22050	Irrelevant	429	4	9	1	22049.75291	0.0011%
	44100	Irrelevant	271	2	6	0	44108.07422	0.0183%

Caution:

-The octave factor can be UPLLN/MPLLN. The octave factor can be UPLLN/MPLLN, the output division ratio can be UPLLP/UPLLQ/UPLLR when the octave factor is UPLLN. Output division ratio can be MPLLP/MPLLQ/MPLLR when the octave factor is MPLLN. Refer to the CMU UPLL Configuration Register and CMU MPLL Configuration Register of the clock controller (CMU) for specific settings. The CMU UPLL configuration register and CMU MPLL configuration register are referenced for details.

29.4.5 I2S Master Mode

I2S can be configured as follows:

- Send master or receive master (half duplex mode using I2S)
- Simultaneous transmitting and receiving of master devices (full duplex mode using I2S)

The I2S operates in master mode with the serial clock output from pin CK and the word select signal generated from pin WS. The MCKOE bit of register I2S_CTRL can be set to output or not to output the drive clock (MCK).

Steps

1. Set the pins to be used for I2S.
2. The clock source is selected via the I2S_CTRL_CLKSEL, **I2S_CTRL.I2SPLLSEL** bits.
3. Set the I2S_PR.I2SDIV[7:0] bits and I2S_CTRL_ODD bits to define the serial data baud rate to achieve the appropriate audio sampling frequency.
4. Set the I2S configuration register (**I2S_CFGR**) Select the I2S standard via the **I2SSTD[1:0]** and PCMSYNC bits, the data length via the **DATLEN[1:0]** bits and the number of bits per channel by configuring the CHLEN bits.
5. If you need to use interrupt, please set the system's interrupt register.
6. If you want to use DMA, please set the DMA related registers.
7. Set the I2S control register (**I2S_CTRL**), including the operating mode setting, communication mode setting, clock output license setting, data output license setting, FIFO reset setting, transmit/receive buffer threshold setting, etc. The operating mode WMS bit selects the I2S master mode.
8. Set the interrupt permit bit.
9. Set I2S_CTRL_TXE and I2S_CTRL_RXE, action starts.

Caution:

- When writing communication data to TXBUF using TXIRQOUT interrupt, if writing two data per interrupt, first turn off the transmit interrupt enable flag bit TXIE after the interrupt starts, and then turn on TXIE after writing data. or write only one data per interrupt.

Sending Sequence

TXE position 1 in the I2S_CTRL register allows transmitting. The transmit sequence starts as soon as the data is written to the transmit buffer. A complete frame means that the left channel data is sent first and then the right channel data is sent. There are no partial frames where only the left channel is sent. During the first bit send, the data is loaded into the shift register in parallel, then shifted serially and output to the SD pin (MSB first) For more detailed information on write operations in the various I2S standard modes,

see 29.4.3 Supported audio protocols.

Receiving Sequence

This mode of operation is essentially the same as the transmit mode, differing only in the transmit-receive setting of the I2S_CTRL register, which sets RXE to position 1 to allow reception. For more detailed information on read operations in the various I^SS standard modes, see 29.4.3 Supported audio protocols.

If new data is received while previously received data has not been read, an overflow error will be generated and the I2S_ER_RXERR flag will be set to 1. If I2S_CTRL.EIE position 1, an interrupt will be generated to indicate the error.

29.4.6 I2S Slave Mode

I2S can be configured as follows:

- Send slave or receive slave (half duplex mode using I2S)
- Slave devices that transmit and receive simultaneously (full duplex mode using I2S)

This mode of operation follows essentially the same rules as the I2S master mode. In slave mode, the I2S interface does not generate a clock. The clock and WS signals are fed from the external master device to which the I2S interface is connected. This way, the user does not need to configure the clock.

Steps

1. Set the pins to be used for I2S.
2. Set the I2S configuration register (I2S_CGFR) to select the I2S standard via the I2SSSTD[1:0] and PCMSYNC bits, the data length via the DATLEN[1:0] bits and the number of bits per channel by configuring the CHLEN bits.
3. If you need to use interrupt, please set the system's interrupt register.
4. If you want to use DMA, please set the DMA related registers.
5. To send data, 1~4 data to be sent should be pre-written to I2S_TXBUF first.
6. Set the I2S control register (I2S_CTRL), including the operating mode setting, communication mode setting, clock output license setting, data output license setting, FIFO reset setting, transmit/receive buffer threshold setting, etc. The operating mode WMS bit selects the I2S slave mode.
7. Set the interrupt permit bit.
8. Set I2S_CTRL.TXE and I2S_CTRL.RXE, action starts.

Sending Sequence

The transmit sequence starts when WMS is set to 1, TXE is set to 1, the external master device sends the clock and requests data transfer via the WS signal. At the start of communication, data is transferred from the transmit buffer to the shift register. During the first bit send, data is loaded from the internal bus into the shift register in parallel, then shifted serially and output to the SD pin (MSB first). Each time the transmit buffer FIFO space is larger than the set threshold, an interrupt will be generated if I2S_CTRL.TXIE is positioned 1. For more detailed information on write operations in the various I2S standard modes, see 29.4.3 Supported audio protocols.

To ensure continuous audio data transmission, the next data to be sent must be written to the TX

FIFO before the end of the current data transmission. If the first clock edge of the next data communication arrives while the data is not yet written to the TX FIFO, a transmit underflow will occur and the I2S_ER.TXERR flag will be set to 1 and an interrupt may be generated. If I2S_CTRL.EIE position 1 is generated when the I2S_ER.TXERR flag changes to 1.

Receiving Sequence

This mode of operation is essentially the same as the transmit mode, differing only in the transmit-receive setting of the I2S_CTRL register, which sets RXE to position 1 to allow reception. For more detailed information on read operations in the various I2S standard modes, see 29.4.3 Supported audio protocols. If new data is received while previously received data has not been read, a receive overflow error will be generated and the RXERR flag will be set to 1. If I2S_CTRL.EIE is set to 1, an interrupt will be generated to indicate the error.

29.4.7 I2S interrupt

I2S interrupt sources are transmit buffer valid space greater than the alarm threshold, receive buffer valid space less than the alarm threshold, receive overflow, transmit underflow and transmit overflow. Send underflow and send overflow are set as I2S send error interrupt TXERR, so it is necessary to determine the actual interrupt source by the flag. I2S interrupt sources are specified in Table 29-4. Once the interrupt condition is established, the corresponding interrupt request is generated.

Users can write the vectors corresponding to the above event trigger sources into different trigger object registers to achieve various event trigger functions. Please refer to [Interrupt Controller (INTC)] for the vectors corresponding to the above event trigger sources.

Table 29-4 shows the list of I2S interrupts.

Table 29-4 I2S interrupt requests

Interruption events	Event Flags	Enable control bit
Send buffer valid space is larger than the alarm threshold	TXBA	TXIE
Receive buffer effective space is less than the alarm threshold	RXBA	RXIE
The receive data area is full and there is still a write data request, receive overflow	RXERR	EIE
Sending data area is empty and there is still a request to send, send the next overflow	TXERR	EIE
Send data area is full and still has write data request, send overflow	TXERR	EIE

29.4.8 Precautions for use

29.4.8.1 Precautions when using as a host

- 1) When the I2S is acting as a host for a data-only transmit action, if all data in I2S_TXBUF has been sent and no new data is written, the I2S will pause after the last data has been sent, at

which point the I2S will no longer generate a communication clock and the transmit error flag bit TXERR will be set to 1. At this point the user can choose to turn off the I2S by writing the I2S_CTRL.**TXE bit** to 0 to turn off the I2S, or write new transmit data to I2S_TXBUF to continue the transmit action. WS will start from the left channel again when sending continues (Philips, MSB/LSB mode)

If the I2S_CTRL.TXE bit is written to 0 directly during the transmit action, the I2S will be shut down immediately and the current data transmission is terminated. This practice will result in the slave's state not being grasped and will cause confusion when communication is restarted without resetting the slave, so it is recommended that the user write the I2S_CTRL.TXE bit to 0 to turn off the I2S when it is in a suspended state.

- 2) When I2S acts as a host for data-only reception actions, two dummy frames can be written in advance if you want to temporarily stop the reception action

TXE will be set to 1 when the data needs to be paused and counted to the corresponding position, 4 data TXE will be set to 1 when the baud rate is 8k~96k, 5 data TXE will be set to 1 when the baud rate is 192k, when the two frames of dummy data are sent, I2S will pause the action, then I2S will no longer generate the communication clock, the transmit error flag bit TXERR will be set to TXE and I2S_CTRL.RXE can be reset to zero to turn off I2S, or write another frame of dummy data to I2S_TXBUF to restart the communication action, turn off TXE when the first data is received again after the pause,

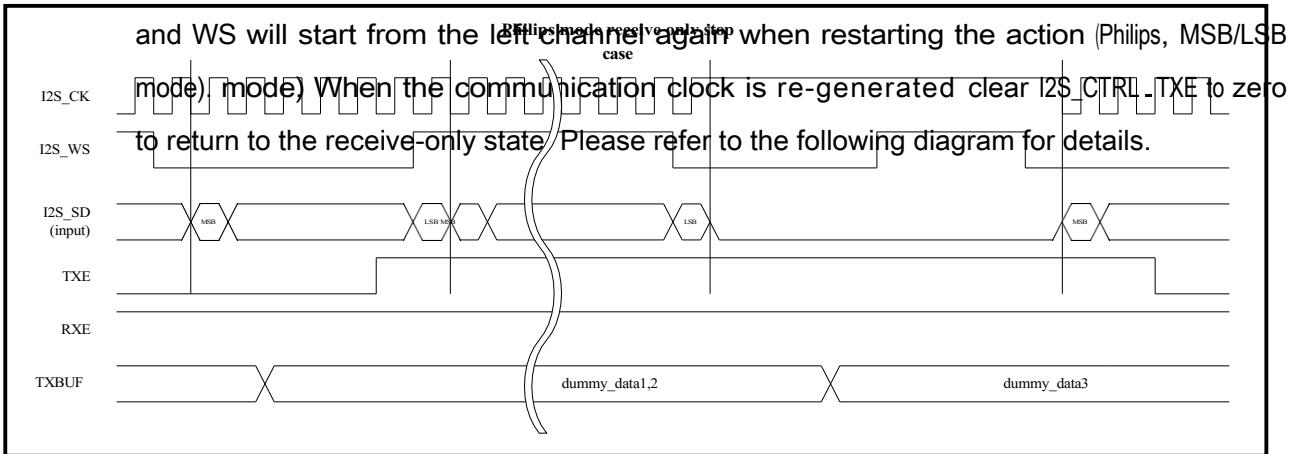


Figure 29-16 Host only receives temporary stop reception

If the I2S_CTRL.RXE bit is written to 0 directly during the receive action, the I2S will be shut down immediately and the current data reception will be terminated. This will cause confusion when the slave sends data when communication is restarted without resetting the slave, so it is recommended that the user turn off the I2S by clearing the I2S_CTRL.TXE and I2S_CTRL.RXE bits when the I2S is suspended.

3) When the I2S acts as a host for full duplex operation, if all data in I2S_TXBUF has been sent and no new data is written, the I2S will pause after the last data has been sent, at which point the I2S will no longer generate a communication clock and the transmit error flag bit TXERR will be set to 1. TXE and I2S_CTRL.RXE to shut down the I2S or write new transmit data to I2S_TXBUF to continue the transmit and receive actions. WS will start from the left channel again when continuing to transmit (Philips, MSB/LSB mode)

If the I2S_CTRL.TXE and I2S_CTRL.RXE bits are written to 0 directly during full duplex operation, the I2S will be shut down immediately and the current data transmission and reception will be terminated. This practice will result in the slave's status not being grasped and will cause confusion when communication is restarted without resetting the slave, so it is

recommended that users turn off the **I2S** by clearing the I2S_CTRL.TXE and I2S_CTRL.RXE bits to zero when the I2S is in a suspended state.

- 4) When I2S acts as a host for PCM short-frame data transmission action, there are two settings to restart transmission when I2S pauses because there is no new data write action for I2S_TXBUF, which one to choose depends on the data reception specification of the slave. If the slave has to check the status of WS every time it receives data, it needs to set I2S_CTRL.TXE to 0 after I2S pause, then write new transmit data to I2S_TXBUF and then set I2S_CTRL.TXE to 1 to restart the transmission. Specific

The action is shown in the figure below.

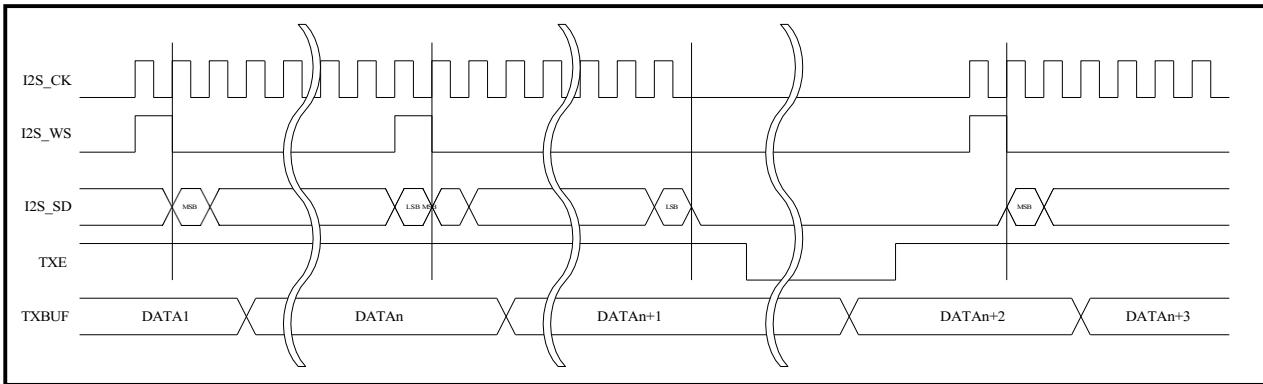


Figure 29-17 PCM short frame host sending pause and resend method one

If the slave will only detect the WS state when the first frame of data is received, the I2S can restart the transmission by writing new transmit data directly to I2S_TXBUF after the I2S

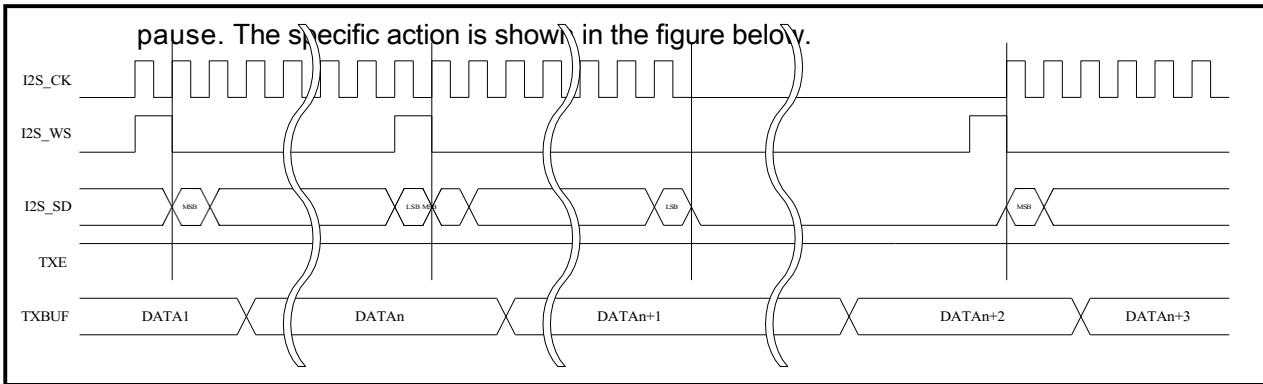


Figure 29-18 PCM short-frame host sending pause and retransmission method II

29.4.8.2 Precautions when using as a slave

- 1) When I2S is acting as a slave, you need to make sure that all registers are configured before finally turning on I2S_CTRL.TXE or I2S_CTRL.RXE to start the slave operation.
- 2) When I2S starts slave operation in Philips, MSB/LSB mode, it is necessary to ensure that the WS signal is at the right channel level when starting, and when I2S starts slave operation in PCM mode, it is necessary to ensure that the WS signal is at low level when starting.
- 3) When the I2S acts as a slave for data reception, each received data will not be read until the next frame of data reception begins, so when communication is paused or terminated, the last frame of data received by the I2S will not be read until the next communication begins.
- 4) When the I2S acts as a slave for data reception in Philips, MSB/LSB mode, the WS is checked for left channel level before each frame of left channel data reception. When the I2S

is receiving data as a slave in PCM mode, the WS is verified to be at a valid level per standard communication protocol before each frame is received.

29.5 Register Description

I2S1 Base Address:

0x4001_E000 I2S2 Base

Address: 0x4001_E400 I2S3

Base Address: 0x4002_2000

I2S4 Base Address:

0x4002_2400

Table 29-5 List of I2S registers

Register Name	Symbols	Offset Address	Bit width	Reset value
I2S control register	I2S_CTRL	0x000	32	0x0000_2200
I2S status register	I2S_SR	0x004	32	0x0000_0014
I2S error status register	I2S_ER	0x008	32	0x0000_0000
I2S configuration register	I2S_CFGR	0x00C	32	0x0000_0000
I2S transmit buffer FIFO data register	I2S_TXBUF	0x010	32	0x0000_0000
I2S receive buffer FIFO data register	I2S_RXBUF	0x014	32	0x0000_0000
I2S prescaler register	I2S_PR	0x018	32	0x0000_0002

Note: Only 32-bit write registers are

supported CMU_BASE_ADDR2 :

Register Name	Symbols	Offset Address	Bit width	Reset value

Note: This register is detailed in the CMU chapter. This register is used to configure the clock source when I2SPLL is selected as the I2S master mode clock source and can be configured as UPLL/R/UPLL/Q/UPLL/P/MPLL/R/MPLL/Q/MPLL/P.

29.5.1 I2S control register (I2S_CTRL)

I2S Control

Register Offset

Address: 0x000

Reset value: 0x0000_2200

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	CLKSEL L	DUPLEX X	CKOE	LRCKOE E	SDOE	I2SPL LSEL	CODEC RC	FIFOR
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	RXBIRQWL			-	TXBIRQWL		MCKOE	ODD	WMS	EIE	RXIE	RXE	TXIE	TXE	

position	Marker	Place Name	Function	Reading and writing
b31~b27	Reserved	-	Read "0", write "0" when writing	R
b23	CLKSEL	Clock source selection	0: Select I2SPLL 1: Select external clock	R/W
b22	DUPLEX	Communication method selection	0: Half Duplex 1: Full duplex	R/W
b21	CKOE	Communication clock output license	0: Output disabled 1: Output licensing	R/W
b20	LRCKOE	Channel clock output license	0: Output disabled 1: Output licensing	R/W
b19	SDOE	Data export license	0: Output disabled 1: Output licensing	R/W
b18	I2SPLLSEL	I2SPLL input selection	0: Input disabled 1: Enter the license	R/W
b17	CODECRC	Codec reset control	0: Software reset 1: Release reset	R/W
b16	FIFOR	fifo reset	0: Release reset 1: Software reset	R/W
b15	Reserved	-	Read "0", write "0" when writing	R
b14~b12	RXBIRQWL[2:0]	Receive buffered interrupt request level	Interrupt request triggered when the available space is less than the set value Note: can only be set to 0/1/2, because the fifo space is 2	R/W
b11	Reserved	-	Read "0", write "0" when writing	R
b10~b8	TXBIRQWL[2:0]	Send buffered interrupt request level	Interrupt request triggered when more space is available than the set value Note: Can only be set to 0/1/2, because the fifo space is 2	R/W

b7	MCKOE	Drive clock output enable	0: Disable driving clock output 1: Enables the drive clock output Note: This bit is only used when in I2S master mode	R/W
----	-------	---------------------------	---	-----

b6	ODD	Prescaler Odd Factor	0: Actual crossover value = I2SDIV×2 1: Actual crossover value = I2SDIV×2+1 Note: This bit is used only in I2S master mode. To set ODD to 1, the I2S_PR register should be set first and then the I2S_CTRL register.	R/W
b5	WMS	I2S operating mode selection	0: I2S host mode 1: I2S slave mode	R/W
b4	EIE	Communication error interrupt enable	0: Communication error interrupt invalid 1: Communication error interruption is valid	R/W
b3	RXIE	Receive interrupt enable	0: Receive interrupt is invalid 1: Receive interrupt valid	R/W
b2	RXE	Receive Enable	0: Forbidden to receive 1: Allowed to receive	R/W
b1	TXIE	Send interrupt enable	0: Send interrupt is invalid 1: Send interrupt is valid	R/W
b0	TXE	Send Enable	0: Forbidden to send 1: Allow sending	R/W

29.5.2 I2S Status Register (I2S_SR)

I2S Status Register

offset address: 0x004

Reset value: 0x0000_0014

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	RXBF	RXBE	TXBF	TXBE	RXBA	TXBA

position	Marker	Place Name	Function	Reading and writing
b31~b6	Reserved	-	Read "0", write "0" when writing	R
b5	RXBF	Receive buffer full	0: Receive buffer is not full 1: Receive buffer full	R
b4	RXBE	Receive buffer empty	0: Receive buffer is not empty 1: Receive buffer empty	R
b3	TXBF	Send buffer full	0: Send buffer is not full 1: Send buffer full	R
b2	TXBE	Send buffer empty	0: Send buffer is not empty 1: Send buffer empty	R
b1	RXBA	Receive buffered alarms (related to water level)	0: No alarm in the receive buffer 1: Receive buffer alarm	R
b0	TXBA	Send buffered alarms (related to water level)	0: No alarm in the send buffer 1: Send buffer alarm	R

29.5.3 I2S Error Status Register (I2S_ER)

I2S Error Status Register

Offset Address: 0x008

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	RXERR	TXERR

position	Marker	Place Name	Function	Reading and writing
b31~b2	Reserved	-	Read "0", write "0" when writing	R
b1	RXERR	Receiving error	0: Do not clear the flag bit 1: Clear flag bit	R/W
b0	TXERR	Sending error	0: Do not clear the flag bit 1: Clear flag bit	R/W

TXERR=1 when transmit overflow/underflow occurs and RXERR=1 when receive overflow occurs.

Writing a 1 to the TXERR bit clears the flag bit when a transmit overflow/underflow occurs, and writing a 1 to the RXERR bit clears the flag bit when a receive overflow occurs.

29.5.4 I2S Configuration Register (I2S_CFGR)

I2S Configuration Register

offset address: 0x00C

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	PCMSY NC	CHLEN	DATLEN[1:0]	I2SSTD[1:0]		

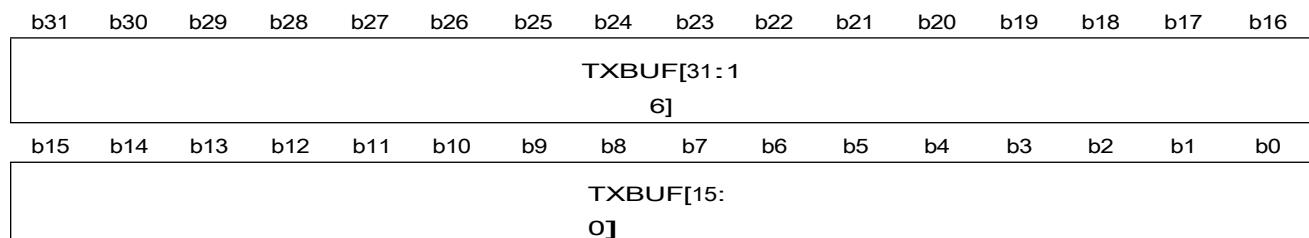
position	Marker	Place Name	Function	Reading and writing
b31~b6	Reserved	-	Read "0", write "0" when writing	R
b5	PCMSYNC	PCM frame synchronization	PCM frame synchronization on 0: short frame synchronization on 1: Long frame synchronization Note: This bit is meaningful only if I2SSTD=11 (using PCM standard)	R/W
b4	CHLEN	Channel length	Mono one frame data length selection 0: 16bit 1: 32bit	R/W
b3~b2	DATLEN[1:0]	Transmission data length selection	Transmitted data length selection 00: 16bit 01: 24bit 1X: 32bit	R/W
b1~b0	I2SSTD[1:0]	Communication protocol selection	Communication protocol selection 00: Philips protocol 01: MSB justified protocol (left-aligned) 10: LSB justified protocol (right-aligned) 11: PCM protocol	R/W

29.5.5 I2S transmit buffer FIFO data register (I2S_TXBUF)

I2S Transmit Buffer FIFO Data Register

Offset Address: 0x010

Reset value: 0x0000_0000



position	Marker	Place Name	Function	Reading and writing
b31~b0	TXBUF[31:0]	Sending data	Store and send data	W

Notes:

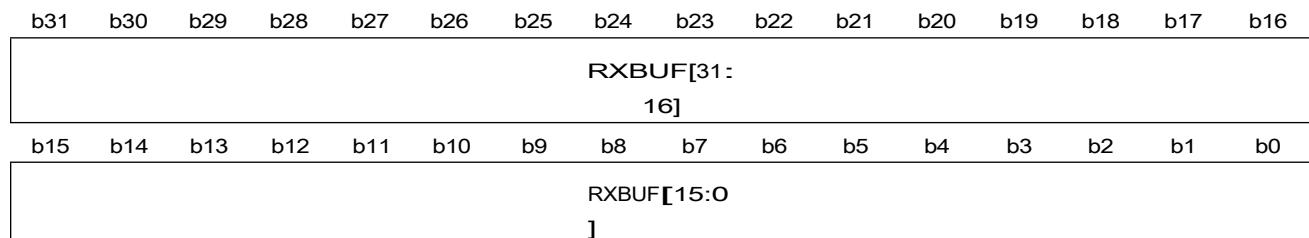
- For 16-bit frames, TXBUF[15:0] stores one frame of left channel or one frame of right channel transmit data.
- For 32-bit frames, TXBUF[31:0] stores one frame of left channel or one frame of right channel transmit data.

29.5.6 I2S Receive Buffer FIFO Data Register (I2S_RXBUF)

I2S Receive Buffer FIFO Data Register

Offset Address: 0x014

Reset value: 0x0000_0000



position	Marker	Place Name	Function	Reading and writing
b31~b0	RXBUF [31:0]	Receiving data	Storage of incoming data	R

Notes:

- For 16-bit frames, RXBUF[15:0] stores one frame of left channel or one frame of right channel receive data.

-
- For 32-bit frames, RXBUF[31:0] stores one frame of left channel or one frame of right channel receive data.

29.5.7 I2S divider register (I2S_PR)

I2S Prescaler Register

Offset Address: 0x018

Reset value: 0x0000_0002

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-								I2SDIV[[7:0]]

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	Read '0', write "0" when writing	R
b7~b0	I2SDIV[7:0]	Frequency division factor	I2SDIV[7:0]=0 or I2SDIV[7:0]=1 is the disabled value See 4.4 Clock Generator Module Actual crossover value = I2SDIV×2+I2S_CTRL.ODD 00000010: 2 crossover frequencies 00000011: 3-way frequency 00000100: 4-way frequency 11111101: 253 crossover 11111110: 254 crossover 11111111: 255 crossover Note: This bit is only used when in I2S master mode	R/W

30 Controller Area Network (CAN)

30.1 Introduction

The CAN (**Controller Area Network**) bus is a bus standard that allows microprocessors or devices to communicate with each other without a host. The module follows CAN bus protocols 2.0A and 2.0B and is upward compatible with CAN-FD. The CAN bus controller handles the sending and receiving of data on the bus, and in this product, CAN has eight filter groups. The filters are used to select the messages to be received for the application.

The application sends transmit data to the bus through one high priority **Primary Transmit Buffer** (PTB) and four **Secondary Transmit Buffers** (STB), and the transmit scheduler determines the order in which mailboxes are sent. The 4 STBs and 10 RBs can be interpreted as a 4-stage **FIFO** and a 10-stage FIFO, which are fully controlled by hardware.

The CAN bus controller can also support **time-trigger** communication.

CAN Key Features:

- CAN2.0A/CAN2.0B protocol is fully supported.
- Upward compatible with CAN-FD protocol.
- Supports maximum communication baud rate **1Mbit/s**
- Support 1~1/256 baud rate pre-scaling, flexible baud rate configuration.
- 10 receive buffers
 - FIFO method
 - Errors or non-received data do not overwrite stored messages
- 1 high priority master transmit buffer PTB
- 4 sub-send buffers STB
 - FIFO method
 - Priority Arbitration Method
- 8 independent groups of filters
 - Supports 11-bit standard ID and 29-bit extended ID
 - Programmable ID CODE bits and MASK bits
- PTB/STB both support single transmit mode
- Silent mode support
- Supports loopback mode
- Support for capturing the type of errors transmitted and locating the location of arbitration failures
- Programmable error warning values

- Supports ISO 11898-4 time triggered CAN and receive timestamps

30.2 CAN System Block Diagram

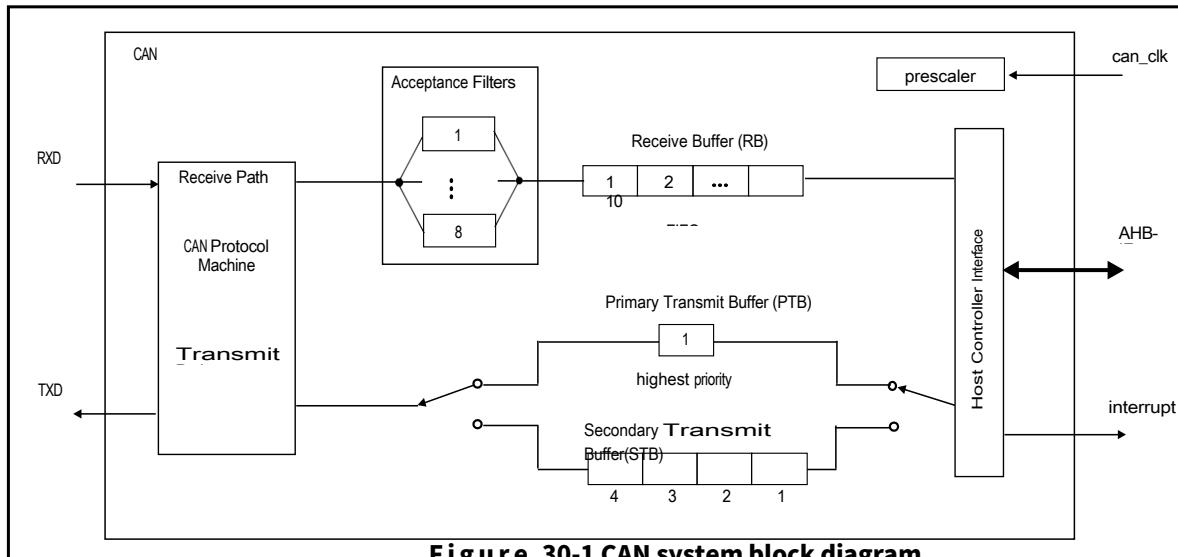


Figure 30-1 CAN system block diagram

30.3 Pin Description

Description

Table 30-1 CAN Pin Descriptions

Pin

Footnote r Name	Dire ctio n	Function Descript ion
CAN_RxD	Input	CAN receive data signal
CAN_TxD	Output	CAN send data signal

30.4 Function Description

The CAN function is described in detail in this chapter.

30.4.1 Baud rate setting

The clock source of can_clk for CAN communication is an external high-speed oscillator. Before using the CAN module, the CAN communication clock needs to be set in the CMU chapter. The EXCLK (CAN control logic clock) must be 1.5 times or more the can_clk (CAN communication clock) when selecting the clock.

The following figure shows the definition of the CAN bit time. The upper part of the dotted line is the bit time defined by the CAN protocol and the lower part of the dotted line is the bit time defined by this CAN controller CAN-CTRL. The segment1 and segment2 can be set via register BT, which can only be set when CFG_STAT.RESET=1, i.e. when the CAN software is reset.

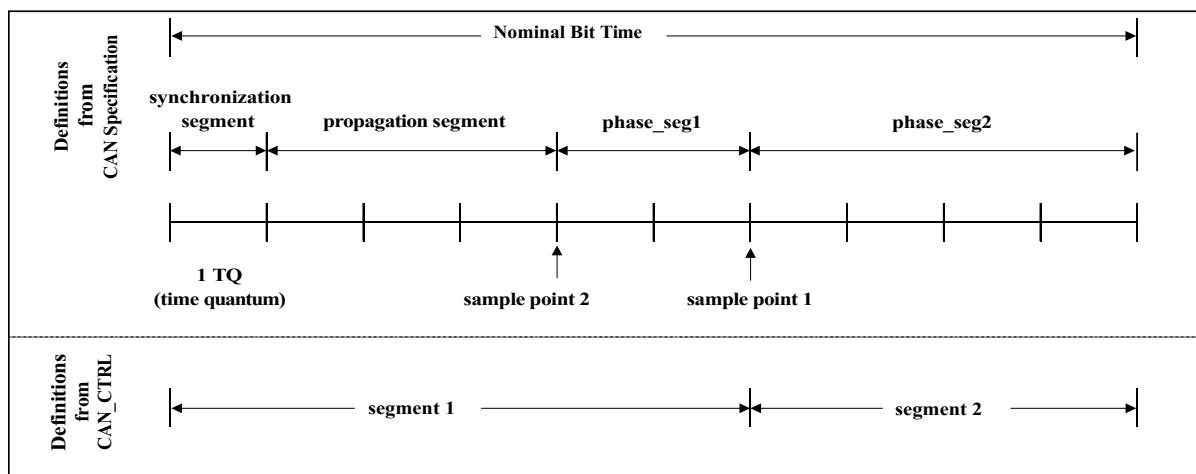


Figure 30-2 CAN bit time definition diagram

Refer to the following equation for TQ calculation, where PRESC is set via the PRESC bit of the BT register. $\square\square\square_ \square\square$ is the CAN communication clock frequency.

$$TQ = \frac{\square\square\square\square}{\square\square\square_ \square\square} + 1$$

Please refer to the following instructions for the calculation of sampling points.

- 1) If the PRESC bit of the BT register is set ≥ 1 , the sample point is located at the demarcation point of segment1 and segment2 as shown in sample point 1 of Figure 30-2.
- 2) If the PRESC bit of the BT register is set to 0, the sample point is 2 TQs ahead of the segment1 and segment2 demarcation points as shown in Figure 30-2. point2, the sample point is 2 TQs ahead of the segment1 and segment2 demarcation points.

It is recommended that the PRESC bit of the BT register be set to a value ≥ 1 .

Please refer to the following formula for bit time calculation, where SEG_1 and SEG_2 are set by the SEG_1 and SEG_2 bits of the BT register.

$$BT = t_{SEG1} + t_{SEG2} = ((SEG_1 + 2) + (SEG_2 + 1)) \times TQ$$

Table 30-2 CAN Bit Time Setting Rules

position	Setting range	Rules
SEG_1 bit of BT register	0~63	SEG_1 ≥ SEG_2 + 1 SEG_2 ≥ SJW
SEG_2 bit of BT register	0~7	
SJW bit of BT register	0~7	

30.4.2 Send buffer

CAN_CTRL provides two transmit buffers for sending data, the primary transmit buffer PTB and the secondary transmit buffer STB. PTB has the highest priority but can only buffer one frame of data, STB has a lower priority than PTB but can buffer 4 frames of data, and the 4 frames of data in STB can work in FIFO mode or priority arbitration mode.

All 4 frames of data in the STB can be sent by setting the TSALL bit in the TCMD register to 1. In FIFO mode, the first data written is sent first, and in priority mode, the data with the smallest ID is sent first.

The data in the PTB has the highest priority, so a PTB send can defer an STB send, but an STB that has already won arbitration and started sending cannot be deferred by a PTB send.

A frame of data in the PTB and STB takes up 4 words and can be accessed through the TBUF register. The next SLOT in the STB is selected by the TBSEL bit of the TCMD register, TBSEL=0 for PTB, TBSEL=1 for STB, and by the TSNEXT bit of the TCTRL register:

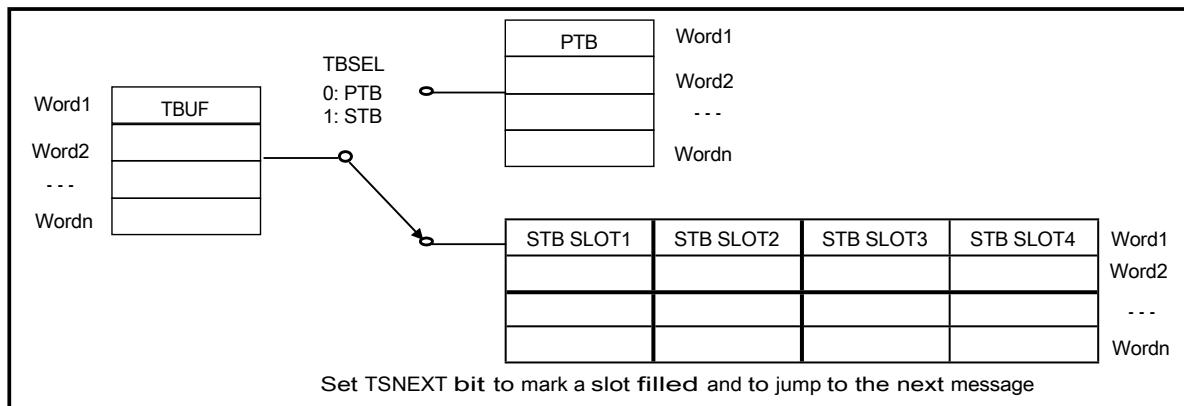


Figure 30-3 CAN TBUF register write send buffer and diagram

30.4.3 Receiving buffer

CAN_CTRL provides 10 SLOT receive buffers for storing the received data, the 10 SLOT receive

buffers operate in FIFO mode. Each RB SLOT takes up 4 words and reads the received data through the RBUF register, always the first

Reads the earliest received data and releases the already read setting RREL of RCTRL register to 1 the next RB SLOT.

RBSLOT by and points to

The diagram of RB SLOT reading through RBUF is shown below.

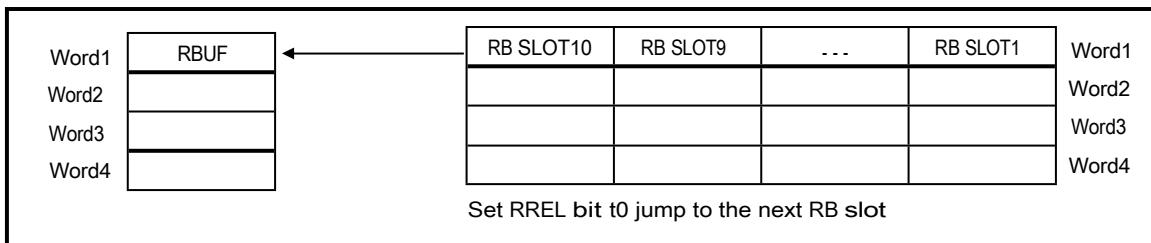


Figure 30-4 CAN RBUF register read-receive buffer diagram

30.4.4 Receive filter register set

CAN_CTRL provides 8 groups of 32-bit filters for filtering the received data to reduce the CPU load, the filters can support standard format 11-bit IDs or extended format 29-bit IDs. The ID CODE register is used to compare the received CAN IDs and the ID MASK register is used to select the CAN bits for comparison. When the corresponding ID MASK bit is 1, the ID CODE of that bit is not compared.

The received data is received as long as it passes any of the 8 sets of filters and the received data is stored in RB, otherwise the data is not received and not stored.

Each group of filters is enabled or disabled by the ACFEN register. IDCODE and ID MASK are set by the SELMASK bit of the ACFCTRL register, with SELMASK=0 pointing to ID CODE and SELMASK=1 pointing to ID MASK. filters are selected by the ACFADR bit of the ACFCTRL register. CODE and ID MASK are accessed through the ACF register and can only be set when CFG_STAT_RESET=1, i.e., when the CAN software is reset. refer to the following figure for the ACF register access to the filter register set.

ACFADR=7

SELMASK=0

ACF_8

ID CODE

www.xhsc.com.c

ID MASK

Figure 30-5 Diagram of CAN ACF register access filter

group

30.4.5 Data sending

You must ensure that the data to be sent by the PTB or STB has been filled before starting the transmission, and then start the PTB or STB transmission. No re-filling of data is allowed during transmission.

The procedure for sending data settings is as follows:

1. Set TBSEL to select from PTB and STB to send BUF
2. Write the data to be sent via the TBUF register
3. If STB is selected, set TSNEXT=1 to complete the loading of all STB SLOTS
4. Send Enable
 - PTB sends using TPE
 - STB sends using TSALL or TSONE
5. Send completion status confirmation
 - PTB send completion using TPIF, TPIE is used to enable TPIF
 - STB uses TSONE to use TSIF when sending is complete and TSIE to enable TSIF
 - uses TSALL to finish sending TSIF is set after all STB SLOTdata is sent, and TSIE is used to enable TSIF.

30.4.6 Single data transmission

The TPSS bit of the CFG_STAT register is used to set the single send mode for PTB and the TSSS bit is used to set the single send mode for STB when the automatic retransmission function is not required. When the data is successfully sent, the single send operation is the same as the normal send mode. However, the following result occurs when the data is not sent successfully:

- TPIF is set (TPIE=1) the corresponding BUF SLOT data will be cleared.
- When an error is sent, KOER is updated and BEIF is set (BEIE=1)
- Arbitration failed and ALIF is set (ALIE=1)

In single send mode, you cannot rely on TPIF alone to determine whether the send is complete, but need to determine whether the send is complete together with BEIF and ALIF.

30.4.7 Cancel data sending

You can cancel a data send that has been requested but not yet executed via TPA or TSA.

Cancellation of a data send can occur in the following ways:

- In Arbitration
 - If the node arbitration fails, the data transmission is cancelled.
 - If the node arbitration is successful, it continues to send.
- Data being sent
 - The data is successfully sent and ACK is received, and the corresponding flag and status are

set normally. Data transmission is not canceled.

- The data was successfully sent but no ACK was received, the data send is cancelled and the error counter is increased.

- TSALL=1 The transmit data set, the ST SLOTdata being sent normally, and the STB SLOT that did not start sending is cancelled.

Cancellation of data delivery results in either:

- The TPA releases the PTB and makes TPE=0.
- TSA releases one STB SLOT or all STB SLOTS depending on whether it is a TSONE or TSALL enabled transmission.

30.4.8 Data reception

The receive filter group reduces the CPU load by filtering out unwanted receive data and reducing the occurrence of interrupts and RB reads. The receive data setup procedure is as follows:

1. Sets the filter group.
2. Set RFIE, RAFIE and AFWL.
3. Wait for RFIF or RAIF.
4. Reads the earliest received data from the RB FIFO via RBUF.
5. Set RREL=1 to select the next RB SLOT.
6. Repeat 4, 5 until RB is confirmed empty by RSTAT.

30.4.9 Error Handling

CAN_CTRL on the one hand automatically handles some of the errors, e.g. by automatically retransmitting data or discarding received frames containing errors, and on the other hand reports the errors to the CPU via interrupts.

CAN nodes have the following three error states:

- Error active: Node automatically sends active error flags when it detects an error.
- Error Passive: The node automatically sends a passive error flag when it detects an error.
- Node Off: In the off state this node no longer affects the entire CAN network.

The CAN_CTRL provides two counters, TECNT and RECNT, for error counting; the TECNT and RECNT counters are incremented and decremented according to the rules specified in the CAN2.0B protocol. A programmable CAN error warning LIMIT register is also provided to generate an error interrupt to notify the CPU.

There are five error types in the CAN communication process, which can be identified by the KOER bit of the EALCAP register.

- Bit error
- Wrong form
- Filling error
- Response error

- CRC error

30.4.10 Node closure

When the number of transmit errors is greater than 255, the CAN node automatically enters the Node Off state and thus does not participate in CAN communication until it returns to the Error Active state. The CAN node shutdown state can be confirmed by the BUSOFF bit in the CFG_STAT register, and an EIF interrupt is generated while BUSOFF is set.

There are two ways for CAN to recover from the node off state to the error active state:

- Power-on reset
- Receive 128 consecutive 11-bit invisible bit sequences (recovery sequences)

In the node off state, the TECNT value remains unchanged and RECNT is used to count the recovery sequence. After recovering from the node shutdown state, TECNT and RECNT are reset to 0.

The RESET bit of the CFG_STAT register is set at the same time as the node off flag BUSOFF is set.

30.4.11 Arbitration Failure Location Capture

CAN_CTRL captures the exact location of the arbitration failure bit and reflects it in the ALC register, which holds the location of the most recent arbitration failure bit and does not update the ALC bit if the node wins the arbitration.

ALC values are defined as follows:

After the SOF bit, the first ID data bit ALC is 0, the second ID data bit ALC is 1, and so on. Since arbitration occurs only in the arbitration field, the maximum value of ALC is 31. For example, if a standard format remote frame and an extended frame arbitrate, and the extended frame fails at the IDE bit, ALC = 12.

30.4.12 Loopback mode

CAN_CTRL supports the following two loopback modes:

- Internal loopback
- External loopback

Both loopback modes can receive their own outgoing data frames and are mainly used for testing purposes.

In internal loopback mode, the module internally connects the receive data line to the transmit data line, and the transmit data is not output. In internal loopback mode, the node generates a self-response signal to avoid ACK errors.

The external loopback mode maintains the connection to the transceiver so that the sent data can still appear on the CAN bus and the CAN can receive its own sent data with the help of the transceiver. The external loopback mode can be determined by the SACK bit in the RCTRL register to generate

a self-response signal or not; when SACK=0, no self-response signal is generated, and when SACK=1, a self-response signal is generated.

External loopback mode, with SACK=0, results in either

- Other nodes also receive the data frame sent by this node and send an answer signal, in which case this node is able to send and receive data successfully.
- If no other node returns an answer signal, an answer error will be generated and the data will be resent and the error counter will be increased. Single send mode is recommended in this case.

When returning from loopback mode to normal mode, a software reset of CAN_CTRL is required in addition to clearing the mode bit.

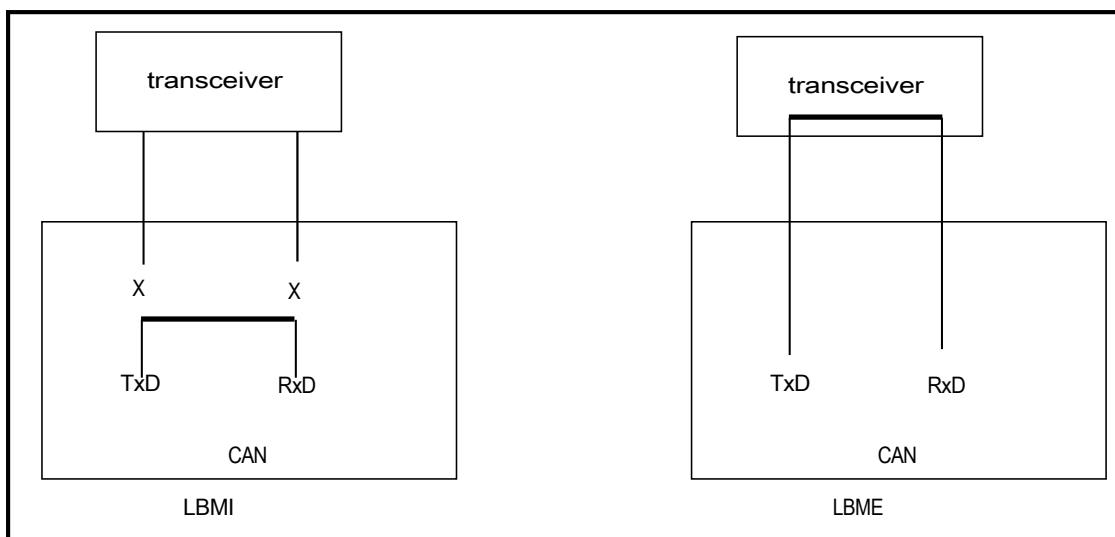


Figure 30-6 CAN LBMI and LBME Schematic

30.4.13 Silent

mode

Silent mode can be used to monitor CAN network data. In silent mode, you can receive data from the CAN bus and do not send any data to the bus. Set the LOM in the TCMD register to 1 to put the CAN bus controller into silent mode and clear it to 0 to leave silent mode.

The external loopback mode can be combined with the silent mode to form an external loopback silent mode, when the CAN can be considered a quiet receiver but can send data when necessary. In external loopback silent mode, frames containing self-answering signals are allowed to be sent, but the node does not generate error flags and overloaded frames.

30.4.14 Software reset function

The software reset function is implemented by setting the RESET bit of the CFG_STAT register to 1.

The reset range of the software reset function is shown in the table below.

Table 30-3 Software Reset Range Table

Regist er Place Name	Softwa re Reset	Re mar ks	Regist er Place Name	Soft ware Reset	Re mar ks
ACFADR	No	–	EWL	Yes	–
ACODE	No	Can only be written during a software reset	KOER	Yes	–
AE_x	No	–	LBME	Yes	–
AFWL	No	–	LBMI	Yes	–
AIF	Yes	–	RACTIVE	Yes	Receiving stops immediately and no ACK is generated
ALC	Yes	–	RAFIE	No	–
ALIE	No	–	RAFIF	Yes	–
ALIF	Yes	–	RBALL	Yes	–
AMASK	No	Can only be written during a software reset	RBUF	Yes	All RBs are marked as empty, with variable values
BEIE	No	–	REF_ID	No	–
BEIF	Yes	–	REF_IDE	No	–
BUSOFF	No	Clear by writing 1	RFIE	No	–
EIE_F	No	–	RFIF	Yes	–
EIF	No	–	RIE	No	–
EPASS	No	–	RIF	Yes	–
EPIE	No	–	ROIE	No	–
EPIF	Yes	–	ROIF	Yes	–
EWARN	No	–	ROM	No	–

Regist er Place Name	Softw are Reset	Re mar ks	Regist er Place Name	Softw are Reset	Re mar ks
ROV	Yes	-	TSNEXT	Yes	-
RREL	Yes	-	TSONE	Yes	-
PRESC	No	Can only be written during a software reset	TPIE	No	-
RSTAT	Yes	-	TPIF	Yes	-
SACK	Yes	-	TPSS	Yes	-
SELMASK	No	-	TSFF	Yes	All STB SLOTS are marked as empty
SEG_1	No	Can only be written during a software reset	TSIE	No	-
SEG_2	No	Can only be written during a software reset	TSIF	Yes	-
SJW	No	Can only be written during a software reset	TSSS	Yes	-
TACTIVE	Yes	Sending stops immediately	TSSTAT	Yes	All STB SLOTS are marked as empty
TBE	Yes	-	TTEN	Y es Y es	-
TBF	No	-	TTIF	Yes	-
TBPTR	No	-	TTIE	No	-
TBSEL	Yes	-	TTPTR	No	-
TBUF	Yes	All STBs are marked as empty and point to PTB	TTTBM	No	-
TECNT	No	Cleared by BUSOFF=1	TTYPE	No	-
TEIF	Yes	-	TT_TRIG	No	-
TPA	Yes	-	TT_WTRIG	No	-
TPE	Yes	-	T_PRESC	No	-
TSA	Yes	-	WTIE	No	-
TSALL	Yes	-	WTIF	Yes	
TSMODE	No	-			

30.4.15 Upward compatible with CAN-FD function

CAN-CTRL Even if CAN-FD frames are received in a network containing CAN-FD, the receiver automatically ignores these frames, does not return the ACK and waits until the bus is free to send or receive the next CAN2.0B frame.

30.4.16 Time-triggered TTCAN

CAN-CTRL provides partial (lever 1) hardware support for the time-triggered communication method specified in ISO 11898-4. This section describes the TTCAN functionality in the following 5 sections.

30.4.16.1 TBUF behavior in TTCAN

mode TTTBM=1

When TTTBM=1, PTB and STB SLOT form a TB SLOT, which is designated to send BUF through TBPTR register, where TBPTR=0 points to PTB, TBPTR=1 points to STB SLOT1, and so on. The host can mark the transmit BUF SLOT by using the TPE and TPF registers, where the TBSEL and TSNEXT registers have no meaning and can be ignored.

With TTTBM=1, the PTB does not have any special properties and, like the STB SLOT, the transmission completion flag uses TSIF.

In TTCAN mode, there is no FIFO mode or priority arbitration mode for sending BUFS, and only one selected SLOT can send data.

In TTCAN mode, the start of transmission needs to be time-triggered and TPE, TSONE, TSALL, TPSS and TPA are fixed to 0 and ignored.

TTTB_M=0

When TTTBM=0, a combination of event-driven communication and receive timestamp functions are used. In this mode, PTB and STB functions are the same as when TTEN=0, so the PTB always has the highest priority, while the STB can work in FIFO mode or arbitration mode.

30.4.16.2 TTCAN Functional Description

After power-up, the Time **Master** needs to be initialized according to the ISO 11898-4 protocol. There can be up to 8 potential Time Masters in a CAN network, each with its own reference message ID (last 3 digits of the ID). These potential Time **Masters** send their own reference messages according to their own priority.

After TTEN=1, the 16-bit counter starts working and when the reference message is successfully received or the Time **Master** successfully sends a reference message, the CAN controller copies the Sync_Mark to the Ref_Mark, which sets the **cycle** time to 0. A

successfully received reference message sets the RIF flag and a successfully sent reference message sets the TPIF flag or TSIF flag. TPIF flag or TSIF flag. At this point the host needs to prepare the trigger condition for the next action.

The trigger condition can be a receive trigger. This trigger only triggers the interrupt can be used to detect that the expected message was not received.

The trigger condition can also be a send trigger. This trigger starts sending the data in the TBUF SLOT specified by the TTPTR register. If the selected TBUF SLOT is marked empty, sending does not begin, but the interrupt flag is set.

30.4.16.3 TTCAN Timing

CAN_CTRL supports ISO11898-4 level 1. A 16-bit counter is included that operates at the bit times defined for PRESC, SEG_1, SET_2. If TTEN=1, there is an additional prescaler T_PRESC.

If the frame is a reference message, the Sync_Mark is copied to the Ref_Mark. cycle time is equal to the counter value minus the Ref_Mark. this time is used as the timestamp for the received message or as the trigger time reference for the sent message.

30.4.16.4 TTCAN trigger method

The TTYPE register defines the trigger method of the TTCAN, the TTPTR register specifies the send SLOT, and TT_TRIG specifies the cycle time of the trigger.

The following five triggers are included:

- Immediate trigger
- Time Trigger
- Single send trigger
- Send start trigger
- Send Stop Trigger

All triggers use the TTIF flag except for the immediate trigger method. with TTTBM=1, only the time trigger method is supported.

Immediate trigger

The trigger is started by writing the high bit of TT_TRIG (not caring about the value written) in this mode, the data in the TBUF SLOT selected by TTPTR is sent immediately. TTIF is not set.

Time Trigger

The time-triggered method generates an interrupt only by setting the TTIF flag and has no other function. If a node expects to receive the expected data within a specific time window, the time-triggered method can be used. If the TT_TRIG value is less than the actual cycle time then TEIF is set and no other action is taken.

Single send trigger

The single-send trigger method is used to send data within the execution time window. In this case, the TSSS bit is ignored.

If the data does not start sending within the specified transmit enable time window, the frame is discarded. The corresponding transmit BUF SLOT is marked empty and the AIF is set. The data in the corresponding transmit BUF is not rewritten, because it can be sent again by setting the TPF.

If the TT_TRIG value is less than the actual cycle time, TEIF is set and no other action is taken.

Send start trigger

The send start trigger method is used within the arbitration time window to participate in arbitration. tsss is used to determine whether to automatically resend or single send mode. If the specified message is not sent successfully, the send stop trigger can be used to stop the send.

If the TT_TRIG value is less than the actual cycle time, TEIF is set and no other action is taken.

Send Stop Trigger

The send stop trigger method is used to stop a send that has been started by the send start trigger method. If the transmission is stopped, the transmission frame is discarded, AIF is set and the selected TBUF SLOT is marked empty, but the data in the TBUF SLOT is not rewritten and can be sent again by setting the TPF.

If the TT_TRIG value is less than the actual cycle time, TEIF is set and execution stops.

30.4.16.5 TTCAN Trigger Watch Time

The TTCAN trigger watchdog time function is similar to the watchdog function and is used when TTTBM=1. It is used to see how long the watchdog has been in operation since the last successful reception of a reference message. The reference message can be received during the cycle time or after an event, and the application should set the appropriate watchdog time for each case.

If the cycle **count is** equal to TT_WTRIG, WTIF is set. write 0 via WTIE to turn off the watchdog trigger. If TT_WTRIG is smaller than the actual cycle time, then TEIF is set.

30.4.17 Interrupts

rrup

Table 30-4 CAN Interrupt Table

tion

s

Interruption flags	Description
RIF	Receiving interruptions
ROIF	Receive overflow interrupt
RFIF	Receive BUF full interrupt
RAFIF	Receive BUF will be full interrupt
TPIF	PTB transmit interrupt
TSIF	STB send interrupt
EIF	Error interruption
AIF	Cancel send interrupt
EPIE	Error Passive Interruption
ALIF	Failed arbitration interruption
BEIF	Bus error interrupt
WTIF	Trigger watchdog interrupt
TEIF	Trigger an error interrupt
TTIF	Time-triggered interrupts

30.5 Register Description

CAN_BASE_ADDR:0x40070400

Table 30-5 List of CAN registers

Register Name	Sym bols	Offset Address	Bit width	Reset value
CAN receive BUF register	CAN_RBUF	0x00~0x0F	128	0xFFFF XXXX
CAN transmit BUF register	CAN_TBUF	0x50~0x5F	128	0xFFFF XXXX
CAN configuration and status registers	CAN_CFG_STAT	0xA0	8	0x80
CAN Command Register	CAN_TCMD	0xA1	8	0x00
CAN transmit control register	CAN_TCTRL	0xA2	8	0x90
CAN Receive Control Register	CAN_RCTRL	0xA3	8	0x00
CAN receive and transmit interrupt enable registers	CAN_RTIE	0xA4	8	0xFE
CAN receive and transmit interrupt flag registers	CAN_RTIF	0xA5	8	0x00
CAN error interrupt enable and flag registers	CAN_ERRINT	0xA6	8	0x00
CAN warning qualifying register	CAN_LIMIT	0xA7	8	0x1B
CAN Bit Timing Register	CAN_BT	0xA8	32	0x0102 0203
CAN error and arbitration failure catch register	CAN_EALCAP	0xB0	8	0x00
CAN receive error counter register	CAN_RECNT	0xB2	8	0x00
CAN transmit error counter register	CAN_TECNT	0xB3	8	0x00
CAN filter group control register	CAN_ACFCTRL	0xB4	8	0x00
CAN filter group enable register	CAN_ACFEN	0xB6	8	0x01
CAN filter group code and mask registers	CAN_ACF	0xB8	32	0xFFFF XXXX
TTCAN TB slot pointer register	CAN_TBSLOT	0xBE	8	0x00
TTCAN Time Trigger Configuration Register	CAN_TTCFG	0xBF	8	0x90
TTCAN Reference Message Register	CAN_REF_MSG	0xC0	32	0xFFFF XXXX
TTCAN Trigger Configuration Register	CAN_TRG_CFG	0xC4	16	0x0000
TTCAN Trigger Time Register	CAN_TT_TRIG	0xC6	16	0x0000
TTCAN Trigger Watch Time Register	CAN_TT_WTRIG	0xC8	16	0x0000

Table 30-6 CAN register BYTE/HALFWORD/WORD access layout

Address	BYTE Access	HALFWORD Access	WORD Access

0x00~0x0F	CAN_RBUF	CAN_RBUF		CAN_RBUF			
0x50~0x5F	CAN_TBUF	CAN_TBUF		CAN_TBUF			
0xA0	CAN_CFG_STAT	CAN_TCM_D	CAN_CFG_S_TAT	CAN_RCT_RL	CAN_TCTR_L	CAN_TCM_DL	CAN_CFG_ST_AT

Address	BYTE Access	HALFWORD Access		WORD Access					
0xA1	CAN_TCMD	–		–					
0xA2	CAN_TCTRL	CAN_RCTL RL	CAN_TCTRL	–					
0xA3	CAN_RCTRL	–		–					
0xA4	CAN_RTIE	CAN_RTIF F	CAN_RTIE	CAN_LIM IT	CAN_ERRI NT	CAN_RTIF F	CAN_RTIE		
0xA5	CAN_RTIF	–		–					
0xA6	CAN_ERRINT	CAN_LIM IT	CAN_ERRIN T	–					
0xA7	CAN_LIMIT	–		–					
0xA8	CAN_BT[7:0]	CAN_BT[15:0]		CAN_BT					
0xA9	CAN_BT[15:8]	–		–					
0xAA	CAN_BT[23:16]	–		–					
0xAB	CAN_BT[31:24]	CAN_BT[31:16]		–					
0xB0	–	–		CAN_TE NT	CAN_REC NT	–			
0xB1	–	–		–					
0xB2	CAN_RECNT	CAN_TE NT	CAN_RECNT	–					
0xB3	CAN_TECNT	–		–					
0xB4	CAN_ACFCTRL[7: :0]	CAN_ACFCTRL		CAN_ACFEN		CAN_ACFCTRL			
0xB5	CAN_ACFCTRL[1 5:8]	–		–					
0xB6	CAN_ACFEN[7:0 1]	CAN_ACFEN		–					
0xB7	CAN_ACFEN[15: 8]	–		–					
0xB8	CAN_ACF	CAN_ACF		CAN_ACF					
0xBC	–	–		CAN_TTC FG	CAN_TBSL OT	–			
0xBD	–	–		–					
0xBE	CAN_TBSLOT	CAN_TTC FG	CAN_TBSLO T	–					
0xBF	CAN_TTCFG	–		–					
0xC0	CAN_REF_MSG[7: :0]	CAN_REF_MSG[15:0]		CAN_REF_MSG					
0xC1	CAN_REF_MSG[1]	–		–					

Address	BYTE Access	HALFWORD Access	WORD Access	
	5:8]			
0xC2	CAN_REF_MSG[2:16]	CAN_REF_MSG[31:16]	–	
0xC3	CAN_REF_MSG[3:24]	–	–	
0xC4	CAN_TRG_CFG[7:0]	CAN_TRG_CFG	CAN_TT_TRIG	CAN_TRG_CFG
0xC5	CAN_TRG_CFG[1:8]	–	–	
0xC6	CAN_TT_TRIG[7:0]	CAN_TT_TRIG	–	
0xC7	CAN_TT_TRIG[1:8]	–	–	
0xC8	CAN_TT_WTRIG[7:0]	CAN_TT_WTRIG		CAN_TT_WTRIG
0xC9	CAN_TT_WTRIG[15:8]			

30.5.1 CAN Receive BUF register (CAN_RBUF)

CAN Receive Buffer Registers

Offset Address: 0x00

Reset value: 0XXXXX XXXXX

register points to the RB

SLOTaddress of the earliest received CAN

mailboxthe RBUF register can be read in any order. The KOER bit is the register EALCAP_KOER and is meaningful only when RBALL=1.

The TX bit indicates that a message sent by itself was received in loopback mode.

The CYCLE_TIME bit is valid only in TTCAN mode and indicates the cycle time at the beginning of the SOF. The data format of the CAN receive mailbox is as follows:

Table 30-7 Standard Format CAN Receive Mailbox Format

Address	b7	b6	b5	b4	b3	b2	b1	b0	Function
RBUF	ID[7:0]								ID
RBUF+1	–				ID[10:8]				ID
RBUF+2	–								ID
RBUF+3	–								ID
RBUF+4	IDE=0	RTR	0	0	DLC [3:0]				Control
RBUF+5	KOER [2:0]			TX	–				Status
RBUF+6	CYCLE_TIME[7:0]								TTCAN
RBUF+7	CYCLE_TIME[15:8]								TTCAN
RBUF+8	DATA1								Data
RBUF+9	DATA2								Data
RBUF+10	DATA3								Data
RBUF+11	DATA4								Data
RBUF+12	DATA5								Data
RBUF+13	DATA6								Data
RBUF+14	DATA7								Data
RBUF+15	DATA8								Data

Table 30-8 Extended Format CAN Receive Mailbox Format

Address	b7	b6	b5	b4	b3	b2	b1	b0	Function	
RBUF	ID[7:0]								ID	
RBUF+1	ID[15:8] ID[10:8]								ID	
RBUF+2	ID[23:16]								ID	
RBUF+3	-			ID[28:24]					ID	
RBUF+4	IDE=1	RTR	0	0	DLC [3:0]				Control	
RBUF+5	KOER[2:0] TX			TX	-					Status
RBUF+6	CYCLE_TIME[7:0]								TTCAN	
RBUF+7	CYCLE_TIME[15:8]								TTCAN	
RBUF+8	DATA1								Data	
RBUF+9	DATA2								Data	
RBUF+10	DATA3								Data	
RBUF+11	DATA4								Data	
RBUF+12	DATA5								Data	
RBUF+13	DATA6								Data	
RBUF+14	DATA7								Data	
RBUF+15	DATA8								Data	

The control bits mean the following:

IDE (IDentifier Extension).

0: Standard format

1: Extended format

RTR (Remote Transmission Request) 0:

Data frame

1: Remote frames

DLC(Data Length Code).

Data length code, set in the range of 0~8, corresponding to the data length of 0Byte~8Byte

30.5.2 CAN transmit BUF register (CAN_TBUF)

CAN Transmit Buffer Registers

Offset Address: 0x50

Reset value: 0XXXX XXXX

next empty CAN transmit BUF

the TBUF registers can be read in any order. The

corresponding TBUF SLOT is marked with a software write 1 to TSNEXT to point to the next TBUF SLOT.

TBUF can only be accessed by WORD.

The data format of the CAN sending mailbox is as follows:

Table 30-9 Standard Format CAN Sending Mailbox Format

Address	b7	b6	b5	b4	b3	b2	b1	b0	Function
TBUF	ID[7:0]								ID
TBUF+1	–				ID[10:8]				ID
TBUF+2	–								ID
TBUF+3	–								ID
TBUF+4	IDE=0	RTR	0	0	DLC [3:0]				Control
TBUF+5	–								–
TBUF+6	–								–
TBUF+7	–								–
TBUF+8	DATA1								Data
TBUF+9	DATA2								Data
TBUF+10	DATA3								Data
TBUF+11	DATA4								Data
TBUF+12	DATA5								Data
TBUF+13	DATA6								Data
TBUF+14	DATA7								Data
TBUF+15	DATA8								Data

Table 30-10 Extended Format CAN Sending Mailbox
Format

Address	b7	b6	b5	b4	b3	b2	b1	b0	Function
TBUF	ID[7:0]								ID
TBUF+1	ID[15:8]								ID
TBUF+2	ID[23:16]								ID
TBUF+3	-			ID[28:24]					ID
TBUF+4	IDE=0	RTR	0	0	DLC [3:0]				Control
TBUF+5	-								-
TBUF+6	-								-
TBUF+7	-								-
TBUF+8	DATA1								Data
TBUF+9	DATA2								Data
TBUF+10	DATA3								Data
TBUF+11	DATA4								Data
TBUF+12	DATA5								Data
TBUF+13	DATA6								Data
TBUF+14	DATA7								Data
TBUF+15	DATA8								Data

The control bits mean the following:

IDE (IDentifier Extension).

0: Standard format

1: Extended format

RTR (Remote Transmission Request) 0:

Data frame

1: Remote frames

DLC(Data Length Code).

Data length code, set in the range of 0~8, corresponding to the data length of 0Byte~8Byte

30.5.3 CAN Configuration and Status Register (CAN_CFG_STAT)

CAN Configuration and Status Register Offset

Address: 0xA0

Reset value: 0x80

b7	b6	b5	b4	b3	b2	b1	b0
RESET	LBME	LBMI	TPSS	TSSS	RACTIVE	TACTIVE	BUSOFF
<hr/>							
positi on	Marker	Place Name	Function				Reading and writing
<hr/>							
b7	RESET	Reset request	Reset request bit 0: No partial reset is requested 1: Request for partial reset Some registers can only be written when RESET=1, please refer to the software reset function, when the node enters BUS OFF state, the hardware will automatically RESET position 1. Please note that when RESET=0 it takes 11 CAN bit time for the node to participate in communication.				R/W
<hr/>							
b6	LBME	External loopback mode enable bit	External loopback mode enable bit 0: Disable external loopback mode 1: Enabling external loopback mode Note: Setting this bit is prohibited in communication.				R/W
<hr/>							
b5	LBMI	Internal loopback mode enable bit	Internal loopback mode enable bit 0: Disable internal loopback mode 1: Enabling internal loopback mode Note: Setting this bit is prohibited in communication.				R/W
<hr/>							
b4	TPSS	PTB single transmission mode	PTB single transmission mode 0: Disable PTB single transmission mode 1: Enabling PTB single transmission mode				R/W
<hr/>							
b3	TSSS	STB single transmission mode	STB single transmission mode 0: STB single transmission mode is disabled 1: Enabling STB single transmission mode				R/W
<hr/>							
b2	RACTIVE	Receiving status signals	Receiving status signal 0: Not receiving 1: Receiving in progress				R
<hr/>							
b1	TACTIVE	Transmitting status signals	Transmitting status signal 0: Not transmitting 1 : Sending in progress				R
<hr/>							

b0	BUSOFF	Bus off status	Bus off state 0: Bus active state 1: Bus off state Note: Writing 1 will clear the TECNT and RECNT registers, but only for debugging purposes.	R/W
----	--------	----------------	--	-----

30.5.4 CAN Command Register (CAN_TCMD)

CAN Command

Register Offset

Address: 0xA1

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
TBSEL	LOM	-	TPE	TPA	TSONE	TSALL	TSA

position	Marker	Place Name	Function	Reading and writing
b7	TBSEL	Send BUF selection bit	Transmit Buffer Select bit 0: PTB 1: STB When TTEN=1 & TTTBM=1, TBSEL is reset to the reset value. Note: When writing the TBUF register or TSNEXT bit, this bit needs to hold a fixed value.	R/W
b6	LOM	Silent mode enable bit	Silent mode enable bit (Listen Only Mode) 0: Silent mode is disabled 1: Enable silent mode Sending is disabled when LOM=1&LBME=0. When LOM=1&LBME=1, answering the corresponding received frames and error frames is prohibited, but data can be sent. Note: Setting this bit is prohibited in communication.	R/W
b5	Reserved	-	The reset value must be maintained.	R
b4	TPE	PTB transmit enable bit	PTB transmit enable bit (Transmit Primary Enable) 0: Disable PTB transmit 1: Enables PTB transmission When this bit is enabled, the Mailbox in the PTB will be sent at the next available location. The STB transmission already started will continue, but the next waiting STB transmission will be delayed until the PTB transmission is completed. This bit will remain 1 after a write 1 until the PTB transmission is complete or until the transmission is cancelled via TPA. Software cannot clear this bit by writing a 0. The TPE is reset to the reset value by hardware in the following cases: – RESET=1 – BUSOFF=1 – LOM=1&LBME=0 – TTEN=1&TTTB=1	R/W

position	Marker	Place Name	Function	Reading and writing
b3	TPA	PTB send cancel bit	PTB Transmit Primary Abort bit 0: No cancellation 1: Cancel PTB transmission that has been requested by TPE Set 1 but not yet started This bit is written 1 by software but cleared by hardware. The TPE bit can be cleared by writing 1, so it is written 1 at the same time as TPE. the following cases TPE is reset to reset value by hardware: – RESET=1 – BUOFF=1 – TTEN=1&TTTB=1	R/W
b2	TSONE	Send a frame of STB data to set the position	Transmit Secondary ONE frame 0: Do not send 1: Send a frame of STB data In FIFO mode, the earliest written data is sent, and the highest priority data is sent in priority mode This bit will remain 1 after a write 1 until STB transmission is complete or until the transmission is cancelled via TSA. Software cannot clear this bit by writing a 0. TSONE is reset to the reset value by hardware in the following cases: – RESET=1 – BUOFF=1 – LOM=1&LBME=0 – TTEN=1&TTTB=1	R/W
b1	TSALL	Send all STB data to set location	Transmit Secondary ALL frame 0: Do not send 1: Send all data in STB This bit will remain 1 after a write 1 until STB transmission is complete or until the transmission is cancelled via TSA. Software cannot clear this bit by writing a 0. TSALL is reset to the reset value by hardware in the following cases: – RESET=1 – BUOFF=1 – LOM=1&LBME=0 – TTEN=1&TTTB=1	R/W
b0	TSA	STB send cancel bit	STB Transmit Secondary Abort bit 0: No cancellation 1: Cancel STB transmission that has been requested by TSONE or TSALL set to 1 but not yet started This bit is written 1 by software but cleared by hardware. Writing 1 clears the TSONE or TSALL bit. TSA is reset to the reset value by hardware in the following cases: – RESET=1 – BUOFF=1	R/W

30.5.5 CAN Transmit Control Register (CAN_TCTRL)

CAN Transmit Control

Register Offset Address: 0xA2

Reset value: 0x90

b7	b6	b5	b4	b3	b2	b1	b0
-	TSNEXT	TSMODE	TTTBM	-	-	TSSTAT[1:0]	
<hr/>							
position	Marker	Place Name	Function				Reading and writing
b7	Reserved	-	The reset value must be maintained.				R
b6	TSNEXT	Next STB SLOT	Next STB (Transmit buffer Secondary NEXT) 0: No action 1: Current STB SLOT is filled, pointing to the next SLOT After the application writes the data in the TBUF, the application identifies that the current STB SLOT has been filled by setting the TSNEXT bit, and thus the hardware points the TBUF to the next STB SLOT. The data in the STB SLOT identified by the TSNEXT bit can be sent via the TSONE or TSALL bit. This bit is cleared in hardware by writing 1 by the application. After all STB SLOTS are filled, TSNEXT is held at 1 until an STB SLOT is released. Note: This bit is fixed to 0 in TTCAN mode.				R/W
b5	TSMODE	STB sending mode	STB transmit mode (Transmit buffer Secondary operation MODE) 0: FIFO mode 1: Priority mode FIFO mode sends data frames in the order they are written. The priority mode is automatically determined based on the ID; the smaller the ID, the higher the priority. Regardless of the mode, PTB has the highest priority. Note: The TSMODE bit can only be set when the STB is empty.				R/W
b4	TTTBM	TTCAN BUF mold style	TTCAN BUF Mode (TTCAN Transmit Buffer Mode) TTTBM is ignored when TTEN=0. 0: TSMODE decision, PTB and STB 1: Set by TBPTR and TTPTR In TTCAN mode, this bit can be set to 0 when only the timestamp function is required to be received, and TSMODE determines whether to use PTB or STB. Note: The TSMODE bit can only be set when the STB is empty.				R/W
b3~b2	Reserved	-	The reset value must be maintained.				R

position	Marker	Place Name	Function	Reading and writing
b1~b0	TSSTAT	STB Status	STB status (Transmission Secondary STATus bits) TTEN=0 or TTTBM=0 00: STB empty 01: STB less than or equal to half full 10: STB is greater than half full 11: STB full TTEN=1 and TTTBM=1 00: PTB and STB empty 01: PTB and STB non-full 10: Reserved 11: PTB and STB full	R

30.5.6 CAN Receive Control Register (CAN_RCTRL)

CAN Receive Control

Register Offset Address: 0xA3

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
SACK	ROM	ROV	RREL	RBALL	-		RSTAT[1:0]

position	Marker	Place Name	Function	Reading and writing
b7	SACK	self-answering	Self-ACKnowledge 0: No self-ACKnowledge 1: When LBME=1, enable self-answer function	R/W
b6	ROM	Receive BUF overflow mode set position	Receive buffer overflow mode (Receive buffer overflow mode) 0: The earliest received data is overwritten 1: Newly received data is not stored	R/W
b5	ROV	Receive BUF overflow flag bit	Receive buffer overflow flag bit (Receive buffer OVerflow) 0: No overflow 1: Upper overflow, at least one data loss Clear the zero by writing RREL to 1.	R
b4	RREL	Release receive BUF	Receive buffer RELease 0: No release 1: indicates that this receive BUF has been read and the RBUF register points to the next RB SLOT.	R/W
b3	RBALL	Receive BUF data to store all data frames	Receive Buffer stores ALL data frames 0: Normal mode 1: Store all data including those with errors.	R/W
b2	Reserved	-	The reset value must be maintained.	R
b1~b0	RSTAT	Receive BUF status	Receive buffer STATus 00: RBUF empty 01: RBUF is non-empty but less than the AFWL programmed value 10: RBUF is greater than or equal to the AFWL programmed value but not full 11: Full (keep this value when overflowing)	R

30.5.7 CAN Receive and Transmit Interrupt Enable Register (CAN_RTIE)

CAN Receive and **Transmit** Interrupt Enable

Register Offset Address: 0xA4

Reset value: 0xFE

b7	b6	b5	b4	b3	b2	b1	b0
RIE	ROIE	RFIE	RAFIE	TPIE	TSIE	EIE	TSFF

position	Marker	Place Name	Function	Reading and writing
b7	RIE	Receive interrupt enable	Receive Interrupt Enable 0: Disable 1: Enabling	R/W
b6	ROIE	Receive overflow interrupt enable	Receive Overrun Interrupt Enable 0: Disable 1: Enabling	R/W
b5	RFIE	Receive BUF full interrupt enable	Receive BUF Full Interrupt Enable (RB Full Interrupt Enable) 0: Disable 1: Enabling	R/W
b4	RAFIE	Receive BUF will be full interrupt enable	Receive BUF Almost Full Interrupt Enable (RB Almost Full Interrupt Enable) 0: Disable 1: Enabling	R/W
b3	TPIE	PTB transmit interrupt enable	PTB Transmission Primary Interrupt Enable 0: Disable 1: Enabling	R/W
b2	TSIE	STB transmit interrupt enable	STB Transmission Secondary Interrupt Enable 0: Disable 1: Enabling	R/W
b1	EIE	Error interrupt enable	Error Interrupt Enable 0: Disable 1: Enabling	R/W
b0	TSFF	Send BUF full flag	TTEN=0 or TTTBM=0: STB full flag (Transmit Secondary buffer Full Flag) 0: STB SLOT is not fully filled 1: STB SLOT is fully filled TTEN=1 and TTTBM=1: TB full flag (Transmit buffer Full Flag) 0: TBPTR selected transmit BUF is not fully filled 1: TBPTR selected send BUF is fully filled	R



30.5.8 CAN Receive and Transmit Interrupt Status Register (CAN_RTIF)

CAN Receive and **Transmit** Interrupt Status

Register Offset Address: 0xA5

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
RIF	ROIF	RFIF	RAFIF	TPIF	TSIF	EIF	AIF

position	Marker	Place Name	Function	Reading and writing
b7	RIF	Receive interrupt flag	Receive Interrupt Flag 0: No data frame received 1: A valid data frame or remote frame is received Clear 0 by writing 1 through the application.	R/W
b6	ROIF	Receive overflow interrupt flag	Receive Overrun Interrupt Flag 0: No RB is overwritten (overwrite) 1: RB has at least one covered ROIF and RFIF are set to 1 at the same time when the overflow occurs. 0 is cleared by writing 1 to the application.	R/W
b5	RFIF	Receive BUF full interrupt flag	Receive BUF Full Interrupt Flag (RB Full Interrupt Flag) 0: RB FIFO is not full 1 : RB FIFO full Clear 0 by writing 1 through the application.	R/W
b4	RAFIF	Receive BUF will be full interrupt flag	Receive BUF Almost Full Interrupt Flag 0: The number of filled RB SLOTS is less than the AFWL setting 1: The number of RB SLOTS being filled is greater than or equal to the AFWL setting Clear 0 by writing 1 through the application.	R/W
b3	TPIF	PTB send interrupt flag	PTB Transmission Primary Interrupt Flag 0: No PTB transmission completed 1: The requested PTB is sent successfully to complete clear 0 by application write 1. Note: TPIF is not valid in TTCAN mode, only TSIF flags are applicable	R/W
b2	TSIF	STB send interrupt flag	STB Transmission Secondary Interrupt Flag 0: No STB transmission completed 1: The requested STB is sent successfully to complete clear 0 by writing 1 through the application. Note: TPIF is not valid in TTCAN mode, only the TSIF flag is used	R/W

positi on	Marker	Place Name	Function	Reading and writing
b1	EIF	Error interrupt flag	Error Interrupt Flag 0: The BUSOFF bit has not changed, or the value of the error counter has not changed relative to the ERROR warning limit setting. 1: The BUSOFF bit changes, or the value of the error counter changes in relation to the ERROR warning limit setting. For example, the value of the error counter changes from less than the set value to more than the set value, or from more than the set value to less than the set value. Clear 0 by writing 1 through the application.	R/W
b0	AIF	Cancel the send interrupt flag	Abort Interrupt Flag 0: No send data canceled 1: Send messages requested via TPA and TSA are successfully cancelled. Clear 0 by writing 1 through the application.	R/W

30.5.9 CAN Error Interrupt Enable and Flag Register

(CAN_ERRINT) CAN ERROR INTERRUPT Enable and

Flag Register Offset Address: 0xA6

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
EWARN	EPASS	EPIE	EPIF	ALIE	ALIF	BEIE	BEIF

position	Marker	Place Name	Function	Reading and writing
b7	EWARN	Reach the set ERROR WARNING limit reached 0: RECNT or TECNT is less than the EWL setting	The set ERROR WARNING limit reached 0: RECNT or TECNT is less than the EWL setting 1: RECNT or TECNT is greater than or equal to the EWL setting WARNING Clear 0 by writing 1 through the application.	R
b6	EPASS	Error Passive	Error Passive mode active 0: Node is active error node 1: The node is a passive error node	R
b5	EPIE	Error Passive Interrupt Enable	Error Passive Interrupt Enable 0: Disable 1: Enabling	R/W
b4	EPIF	Error Passive Interrupt Flag	Error Passive Interrupt Flag 0: No error active to error passive or error passive to error active change has occurred 1: Change from error active to error passive or error passive to error active occurs Clear 0 by writing 1 through the application.	R/W
b3	ALIE	Arbitration failure interrupt enable	Arbitration Lost Interrupt Enable 0: Disable 1: Enabling	R/W
b2	ALIF	Arbitration failure interrupt flag	Arbitration Lost Interrupt Flag 0: Arbitration successful 1: Arbitration failure Clear 0 by writing 1 through the application.	R/W
b1	BEIE	Bus error interrupt enable	Bus Error Interrupt Enable 0: Disable 1: Enabling	R/W
b0	BEIF	Bus error interrupt flag	Bus Error Interrupt Flag 0: No bus error 1: Bus error Clear 0 by writing 1 through the application.	R/W

30.5.10 CAN Bit Timing Register (CAN_BT)

CAN Bit Timing

Register Offset Address:

0xA8

Reset value: 0x0102 0203

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
PRESC[7:0]								-	SJW[6:0]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
-	SEG_2[6:0]								SEG_1[7:0]							

position	Marker	Place Name	Function	Reading and writing
b31~b24	PRESC	Prescaler setting	Prescaler setting (Prescaler) This register sets the value when the module BCLK is set to position (PRESC+1) division as the clock of TQ.	R/W
b23	Reserved	-	The reset value must be maintained.	R
b22~b16	SJW	Resynchronization compensation width time setting	Resynchronization Compensation Width Time Setting (Bit Timing Segment 2) Resynchronization Compensation Width Time = (SJW+1)*TQ	R/W
b15	Reserved	-	The reset value must be maintained.	R
b14~b8	SEG_2	Bit 2 time setting	Bit Timing Segment 2 time unit setting (Bit Timing Segment 2) Bit Timing Segment 2 time = (SEG_2+1)*TQ	R/W
b7~b0	SEG_1	Bit 1 time setting	Bit Timing Segment 1 Time = (SEG_1+2)*TQ	R/W

30.5.11 CAN Error and Arbitration Lost Capture

Register (CAN_EALCAP) CAN Error and

Arbitration Lost Capture Register Offset

Address: 0xB0

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
KOER [2:0]				ALC[4:0]			

position	Marker	Place Name	Function	Reading and writing
b7~b5	KOER	Error category	Kind Of Error 000: No error 001: Bit error 010: Form error 011: Filling error 100: Wrong answer 101: CRC error 110: Other errors 111: Reserved The KOER bit is updated when there is an error, and the KOER bit remains unchanged when sending and receiving normally.	R
b4~b0	ALC	Arbitration Failure Location Capture	Arbitration Lost Capture (ALC) records the position in a frame of data at the time of an arbitration failure.	R

30.5.12 CAN Warning Qualification Register (CAN_LIMIT)

CAN Warning Limits Register

Offset Address: 0xA7

Reset value: 0x1B

b7	b6	b5	b4	b3	b2	b1	b0
AFWL[3:0]						EWL[3:0]	

position	Marker	Place Name	Function	Reading and writing
b7~b4	AFWL	Receive BUF will be full Warning Limit	Receive BUF will be full Warning Limit (receive buffer Almost Full Warning Limit) The set value range is 1~10. AFWL=0 is meaningless and is treated as AFWL=1.	R/W
b3~b0	EWL	Error Waring Limit programmin g value	Error Waring Limit Programmed Value (Programmable Error Waring Limit) Error Waring Limit=(EWL+1)*8. The value of this register setting affects the EIF flag.	R/W

30.5.13 CAN Receive Error Counter Register (CAN_RECNT)

CAN Receive Error CouNT Register

Offset Address: 0xB2

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
RECNT[7:0]							

position	Marker	Place Name	Function	Reading and writing
b7~b0	RECNT	Receive error counter	Receive Error CouNT The receive error counter increases or decreases according to the error count specified by the CAN protocol. This counter does not have an upper Overflow, 255 is the maximum value.	R

30.5.14 CAN Transmit Error Counter Register (CAN_TECNT)

CAN Transmit Error CouNT Register

Offset Address: 0xB3

Reset value: 0x00



position	Marker	Place Name	Function	Reading and writing
b7~b0	TECNT	Send error counter	Transmit Error CouNT The transmit error counter increases or decreases according to the error count specified by the CAN protocol. This counter does not have an overflow and 255 is the maximum value.	R

30.5.15 CAN Filter Group Control Register (CAN_ACFCTRL)

CAN Acceptance Filter Control

Register Offset Address: 0xB4

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
-		SELMASK	-				ACFADR

position	Marker	Place Name	Function	Reading and writing
b7~b6	Reserved	-	The reset value must be maintained.	R
b5	SELMASK	Selecting the filter's mask register filter's mask register	Select filter's mask register (SElect acceptance MASK) 0: ACF points to the filter ID register 1: ACF points to the filter MASK register Select specific filter register groups via ACFADR	R/W
b4	Reserved	-	The reset value must be maintained.	R
b3~b0	ACFADR	Filter Address	filter address (acceptance filter address) ACFADR points to a specific filter to distinguish between ID and MASK by SELMASK. 0000: points to ACF_1 0001: Points to ACF_2 0010: Points to ACF_3 0011: Points to ACF_4 0100: Points to ACF_5 0101: Point to ACF_6 0110: Points to ACF_7 0111~1111: point to ACF_8	R/W

30.5.16 CAN Acceptance Filter Group Enable

Register (**CAN_ACFEN**) CAN Acceptance Filter

Enable Register Offset Address: 0xB6

Reset value: 0x01

b7	b6	b5	b4	b3	b2	b1	b0
AE_8	AE_7	AE_6	AE_5	AE_4	AE_3	AE_2	AE_1

position	Marker	Place Name	Function	Reading and writing
b7	AE_8	ACF_8 enable	ACF_8 Enable (Acceptance Filter 8 Enable) 0: Disable 1: Enabling	R/W
b6	AE_7	ACF_7 enable	ACF_7 Enable (Acceptance Filter 7 Enable) 0: Disable 1: Enabling	R/W
b5	AE_6	ACF_6 enable	ACF_6 Enable (Acceptance Filter 6 Enable) 0: Disable 1: Enabling	R/W
b4	AE_5	ACF_5 enable	ACF_5 Enable (Acceptance Filter 5 Enable) 0: Disable 1: Enabling	R/W
b3	AE_4	ACF_4 enable	ACF_4 Enable (Acceptance Filter 4 Enable) 0: Disable 1: Enabling	R/W
b2	AE_3	ACF_3 enable	ACF_3 Enable (Acceptance Filter 3 Enable) 0: Disable 1: Enabling	R/W
b1	AE_2	ACF_2 enable	ACF_2 Enable (Acceptance Filter 2 Enable) 0: Disable 1: Enabling	R/W
b0	AE_1	ACF_1 enable	ACF_1 Enable (Acceptance Filter 1 Enable) 0: Disable 1: Enabling	R/W

30.5.17 CAN Acceptance Filter code and mask register

(CAN_ACF) CAN Acceptance Filter code and mask

Register Offset address: 0xB8

Reset value: 0xXXXX XXXX

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	AIDEE	AID E	ACODE[28:16] or AMASK[28:16]												
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ACODE[15:0] or AMASK[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31	Reserved	-	The readout value is variable.	R
b30	AIDEE	IDE bit comparison enable	IDE bit check enable (Acceptance mask IDE bit check enable) is valid when SELMASK=1 0: the filter receives standard and extended format frames 1: The filter receives standard format or extended format frames defined by the AIDE bit	R/W
b29	AIDE	IDE bit MASK	IDE bit MASK 0: The filter only receives the standard format 1: The filter only accepts extended formats	R/W
b28~b0	ACODE/ AMASK	Screener CODE/Scree ner MASK	The filter CODE (acceptance filter code) points to a specific filter via ACFADR. When SELMASK=0, it means the CODE of the filter. Bit 10~bit 0 is used for standard format, and bit 28~bit 0 is used for extended format. The filter CODE (acceptance filter mask) points to a specific filter via ACFADR. When SELMASK=1, it indicates the MASK of the filter. Bits 10~Bits 0 are used in standard format and bits 28~Bits 0 are used in extended format.	R/W

30.5.18 TTCAN TB slot pointer register (CAN_TBSLOT)

TTCAN TB Slot Pointer Register

Offset Address: 0xBE

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
TBE	TBF	-	-	-		TBPTR[2:0]	1

position	Marker	Place Name	Function	Reading and writing
b7	TBE	Set TB to empty	Set TB slot to "empty" 0: No operation 1: The SLOT selected by TBPTR is marked as empty TBE is automatically reset to 0 when SLOT is marked as empty and TSFF=0. If this bit is set to 1, TBE=1 if there is data being sent in the selected SLOT, then TBE is reset to 0 when the sending is completed, sending is wrong or sending is canceled. TBE has higher priority than TBF.	R/W
b6	TBF	Set TB as filled	Set TB slot to "Filled" 0: No operation 1: The SLOT selected by TBPTR is marked as filled When SLOT is marked as filled and TSFF=1, TBE is automatically reset to 0.	R/W
b5~b3	Reserved	-	The reset value must be maintained.	R
b2~b0	TBPTR	TB SLOT Pointer	TB SLOT pointer (Pointer to a TB message slot) 000: points to PTB 001: Point to STB SLOT1 010: Point to STB SLOT2 011: Pointing to STB SLOT3 100: Points to STB SLOT4 Others: Set the ban The pointed TB SLOT can be accessed read or write via TBUF and can be marked as populated or not by TBE and TBF. The TBSEL and TSNEXT registers are invalid in TTCAN mode. Note: Writes to this bit can only be performed when TSFF=0.	R/W

30.5.19 TTCAN Time Trigger Configuration Register (CAN_TTCFG)

TTCAN TB Slot Pointer Register

Offset Address: 0xBF

Reset value: 0x90

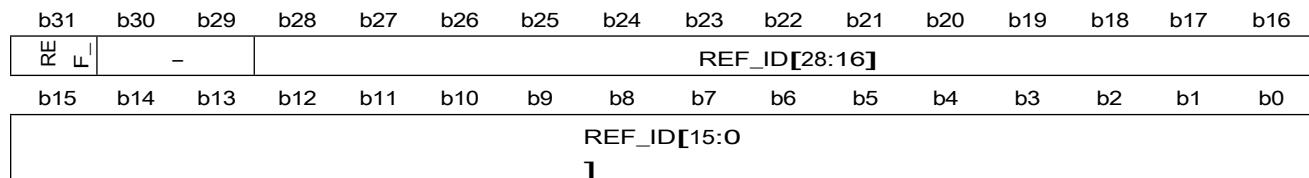
b7	b6	b5	b4	b3	b2	b1	b0
WTIE	WTIF	TEIF	TTIE	TTIF	T_PRESC[1 : 0]		TTEN
<hr/>							
position	Marker	Place Name	Function			Reading and writing	
b7	WTIE	Trigger watchdog interrupt enable	Watch Trigger Interrupt Enable 0: Disable 1: Enabling				R/W
b6	WTIF	Trigger the watchdog interrupt flag	Trigger Watch Trigger Interrupt Flag (Watch Trigger Interrupt Flag) WTIF is set when CYCLE COUNT value = TT_WTRIG set value and WTIE = 1. Clear 0 by writing 1 through the application.				R/W
b5	TEIF	Trigger the error interrupt flag	Trigger Error Interrupt Flag TEIF is set when the TT_TTIG setting is less than the actual CYCLE_TIME. Clear 0 by application write 1.				R/W
b4	TTIE	Time-triggered interrupt enable	Time Trigger Interrupt Enable 0: Disable 1: Enabling				R/W
b3	TTIF	Time-triggered interrupt flag	Time Trigger Interrupt Flag TTIF is set when CYCLE COUNT value = TT_TRIG set value and TTIE = 1. If TT_TRIG is not updated, TTIF is only set 1 time and the next basic CYCLE is not set. Clear 0 by writing 1 through the application.				R/W
b2~b1	T_PRESC	TTCAN counter prescaler	TTCAN counter prescaler (TTCAN Timer Prescaler) 00: 1 division of the bit time set in the BT register 01: 2 divisions of the bit time set by the BT register 10: 4 divisions of the bit time set by the BT register 11: 8 divisions of the bit time set by the BT register Note: T_PRESC can be operated at TTEN=0 for write operation or at the same time for write TTEN=1.				R/W
b0	TTEN	TTCAN Enable	TTCAN Enable (Time Trigger Enable) 0: Disable 1: Enable TTCAN, the counter starts counting.				R/W

30.5.20 TTCAN Reference Message Register (CAN_REF_MSG)

TTCAN Reference Message Register

Offset Address: 0xC0

Reset value: 0x0000 0000



position	Marker	Place Name	Function	Reading and writing
b31	REF_IDE	Reference news of IDE bit	REFERENCE message IDE bit 0: Standard format 1: Extended format	R/W
b30~b29	Reserved	-	The readout value is variable.	R
b28~b0	REF_ID	ID bit of the reference message	ID bits of the reference message (REFERENCE message IDentifier) REF_IDE=0: REF_ID[10:0] is valid REF_IDE=1: REF_ID[28:0] is valid REF_ID is used to detect reference messages and is applicable for both sending and receiving. When a reference message is detected, the Sync_Mark of the current frame becomes Ref_Mark. REF_ID[2:0] is fixed to 0 and its value is not checked, so that up to 8 potential time masters can be supported. When the highest byte of REF_MSG is written, then it is necessary to wait for 6 CAN clock cycles to complete the write operation of REF_MSG to CAN clock domain transfer.	R/W

30.5.21 TTCAN Trigger Configuration Register (CAN_TRG_CFG)

TTCAN Reference Message Register

Offset Address: 0xC4

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TEW [3:0]				–	TTYPE[2:0]			–			TTPTR[2:0]				

position	Marker	Place Name	Function	Reading and writing
b15~b12	TEW	Send enable window	Transmit Enable Window Single Shot Transmit Trigger for TTCAN, you can set the TEW+1 cycle time window, the transmission is only allowed within this window.	R/W
b11	Reserved	–	The reset value must be maintained.	R
b10~b8	TTYPE	Trigger Type	Trigger Type (Trigger Type) 000: Immediate Trigger for immediate transmission 001: Time Trigger for receive triggers 010: Single Shot Transmit Trigger for exclusive time windows 011: Transmit Start Trigger for merged arbitrating time windows 100: Transmit Stop Trigger for merged arbitrating time windows Others: Reserved The trigger time is set by TT_TRIG register and TB Slot is selected by TTPTR.	R/W
b7~b3	Reserved	–	The reset value must be maintained.	R
b2~b0	TTPTR	Send trigger TB slot pointer	Transmit Trigger TB slot Pointer (Transmit Trigger TB slot Pointer) 000: points to PTB 001: Point to STB SLOT1 010: Point to STB SLOT2 011: Pointing to STB SLOT3 100: Points to STB SLOT4 Others: Set the ban If the pointed TB SLOT is marked empty, TEIF is set when the trigger time is reached.	R/W

30.5.22 TTCAN Trigger Time Register (CAN_TT_TRIG)

TTCAN Reference Message Register

Offset Address: 0xC6

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TT_TRIG[15:0]															

position	Marker	Place Name	Function	Reading and writing
b15~b0	TT_TRIG	Trigger time	Trigger Time (Trigger Time) Used to specify the cycle time of the trigger, for send trigger send SOF time is about TT_TRIG setting + 1 After the highest byte of TT_TRIG is written, the TT_TRIG value starts to pass to the CAN clock domain. Therefore as If BYTE operation, you need to write the low byte first and then the high byte.	R/W

30.5.23 TTCAN Trigger Watchdog Time Register (CAN_TT_WTRIG)

TTCAN Watch Trigger Time Register

Offset Address: 0xC8

Reset value: 0xFFFF

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TT_WTRIG[15:0]															

position	Marker	Place Name	Function	Reading and writing
b15~b0	TT_WTRIG	Trigger time	Trigger Time (Trigger Time) Used to specify the cycle time of the watchdog trigger. When the highest byte of TT_WTRIG is written, the TT_WTRIG value starts to pass to the CAN clock domain. Therefore, if BYTE operation, the low byte needs to be written first before the high byte.	R/W

30.6 Precautions for use

30.6.1 CAN bus anti-interference measures

CAN bus is widely used in automotive, industrial control and other industries. If the electromagnetic environment of CAN application site is harsh, there are circuit imbalance, space electromagnetic field, power grid incoming and other factors, it will cause CAN bus to generate a lot of communication noise due to radiation and conduction interference, resulting in the increase of bus error frames, frequent retransmission, and the correct data cannot arrive in time, which seriously affects the data communication quality. Therefore, the actual application should be devoted to eliminate noise interference and ensure the stable operation of CAN bus network.

The following are several common types of CAN bus immunity measures (including but not limited to)

- 1) Increased electrical isolation of the CAN bus interface
- 2) Signal ground of the common transceiver
- 3) Use shielded twisted-pair cable and ground it properly
- 4) Improved twinning of CAN transmission lines
- 5) Adding signal protector
- 6) Improved network topology
- 7) Application layer software anti-interference mechanism

30.6.2 CAN controller noise constraints

In the CAN bus network should ensure that the bit time of communication meets the requirements of the standard protocol, if the introduction of noise interference that does not meet the bit time width may cause abnormal operation of the CAN controller.

31USB2.0 Full Speed Module (USBFS)

31.1 USBFS Introduction

The USB Full Speed (USBFS) controller provides a USB communication solution for portable devices. the USBFS controller supports both host and device modes and has an integrated full-speed PHY inside the chip. the USBFS controller supports full-speed (FS, 12Mb/s) and low-speed (LS, 1.5Mb/s) transceivers in host mode and only full-speed (FS, 12Mb/s) transceivers in device mode. The USBFS controller supports all transfer modes (control, bulk, interrupt, and synchronous) defined by the USB 2.0 protocol.)

31.2 USBFS Key Features

There are three main categories: generic features, host mode features, and device mode features.

31.2.1 General Features

- Built-in on-chip USB2.0 full-speed PHY
- Supports host mode and device mode
- FS SOF and low-speed "Keep-alive" tokens are supported and have the following features:
 - SOF Pulse pin output function
 - SOF pulses can be used as internal event sources to trigger TIMER, DMA and other modules
 - Configurable frame period
 - Configurable end-of-frame interrupts
- Module with embedded DMA and software configurable AHB burst transfer type
- Power saving features such as USB hang, stop RAM clock, stop PHY domain clock
- 1.25KB of dedicated RAM with advanced FIFO control
 - RAM space can be divided into different FIFOs for flexible and efficient use of RAM
 - Multiple packets can be stored per FIFO
 - Dynamic allocation of storage areas
 - The size of the FIFO can be configured to a non-power-of-2 value to allow continuous use of the memory cells
- Maximum USB bandwidth can be achieved within one frame without application intervention
- Host mode or device mode can be determined automatically based on the level of the ID line

31.2.2 Host Mode Features

- Host mode supports USB2.0 Full Speed (FS, 12Mb/s) and Low Speed (LS, 1.5Mb/s) transfers
- Requires VBUS voltage generation via external power chip
- Up to 12 host channels (pipes) each channel can be dynamically reconfigured to support any type of USB transfer
- Built-in hardware scheduler for:
 - Store up to 8 interrupt plus synchronous transfer requests in a periodic hardware queue
 - Store up to 8 control plus bulk transfer requests in an acyclic hardware queue
- Manage a shared RX FIFO, a cyclic TX FIFO, and an acyclic TX FIFO for efficient use of USB data RAM

31.2.3 Device Mode Features

- Slave mode supports USB2.0 Full Speed (FS, 12Mb/s) transfers.
- 1 bi-directional control endpoint 0
- 5 OUT endpoints that can be configured to support bulk, interrupted, or simultaneous transmission
- 5 IN endpoints that can be configured to support bulk, interrupt or synchronous transmission
- Contains 6 transmit FIFOs (one transmit FIFO per IN endpoint) and one receive FIFO (shared by all OUT endpoints)
- Support remote wake-up function.
- Support soft disconnect function
- VBUS PIN supports 5V withstand voltage.

31.3 USBFS System Block Diagram

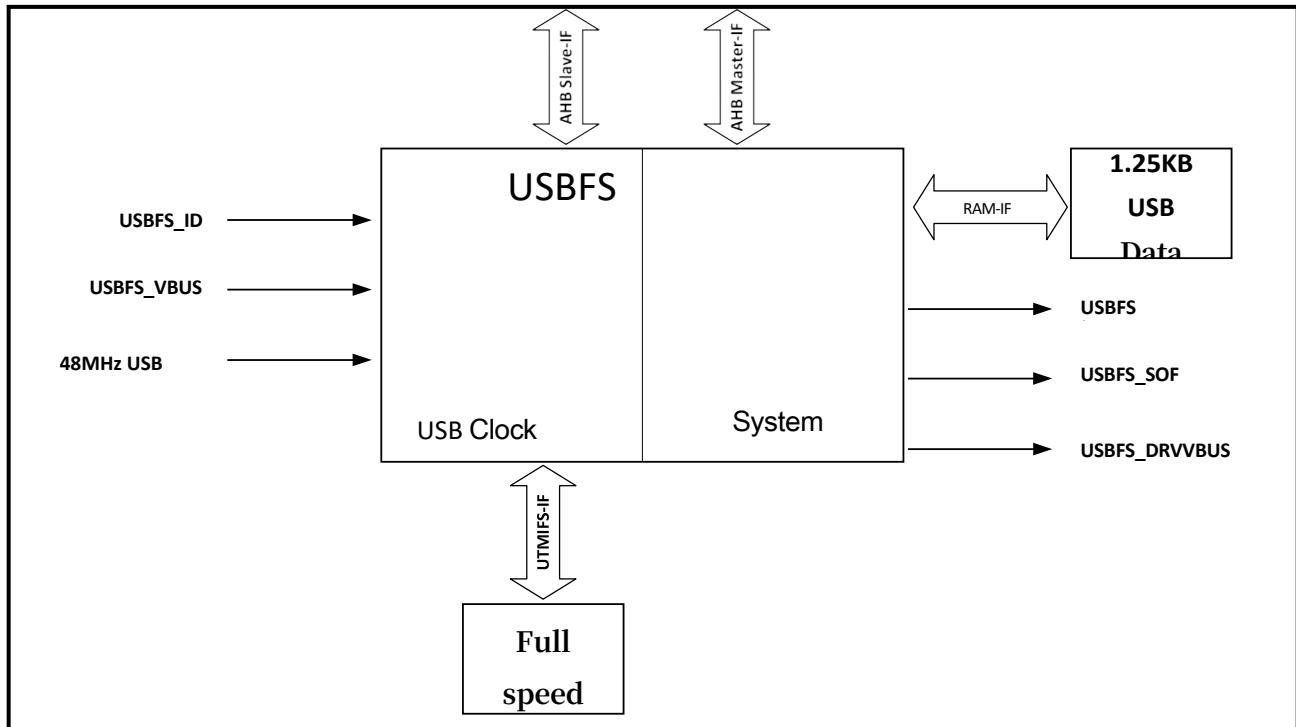


Figure 31-1 USBFS System Block Diagram

31.4 USBFS Pin

Description

Table 31-1 USBFS Pin Descriptions

Foot e Name	Dire ctio n	Functio n Descript ion
USBFS_VBUS	Input	Power port, 5V withstand voltage
USBFS_DP	Input/output	Differential data D+ signal
USBFS_DM	Input/output	Differential data D-signal
USBFS_DRVVBUS	Output	External power chip enable signal
USBFS_ID	Input	USB A-B device identification signal
USBFS_SOF	Output	SOF output pulse signal

Since the USBFS_DP and USBFS_DM pins are multiplexed with general purpose GPIOs, it is recommended to turn off the digital function of their corresponding pins when using USBFS, as described in the General Purpose GPIO chapter. Also when the USBFS function is not in use, additional current consumption is generated when the digital function pins corresponding to the USBFS_DP and USBFS_DM pins are flipped.

31.5 USBFS Function Description

31.5.1 USBFS clock and operating modes

The clock used by USBFS needs to be configured to 48MHz, which is generated by the internal PLL circuit, and the PLL clock source needs to be selected from an external high-speed oscillator.

USBFS can be used as a host or device and includes an on-chip full-speed PHY.

The on-chip full-speed PHY has integrated internal pull-up and pull-down resistors, and the

USBFS can be automatically selected based on the current mode and connection status. The

USBFS operates with a VCC voltage range of 3.0 to 3.6V.

31.5.2 USBFS mode decision

USBFS determines the current operating mode in two ways:

Method 1: The module works in device mode when the USBFS_ID line is detected high and in host mode when the USBFS_ID line is detected low.

Method 2: Force host/device mode by setting the FDMOD or FHMOD bit of register USBFS_GUSBCFG to 1 to force the module to operate in device or host mode by ignoring the level of the USBFS_ID line.

31.5.3 USBFS Host Features

31.5.3.1 Host Function Introduction

When USBFS is operating in host mode, VBUS is the 5V power supply pin specified by the USB protocol. USBFS_DRVVBUS is used to enable the external USB power supply chip, and the over-current detection of the external power supply chip can be achieved by the external interrupt IRQ of this MCU. USB_VBUS can be used as GPIO in host mode.

A typical USB host mode system build diagram is as follows:

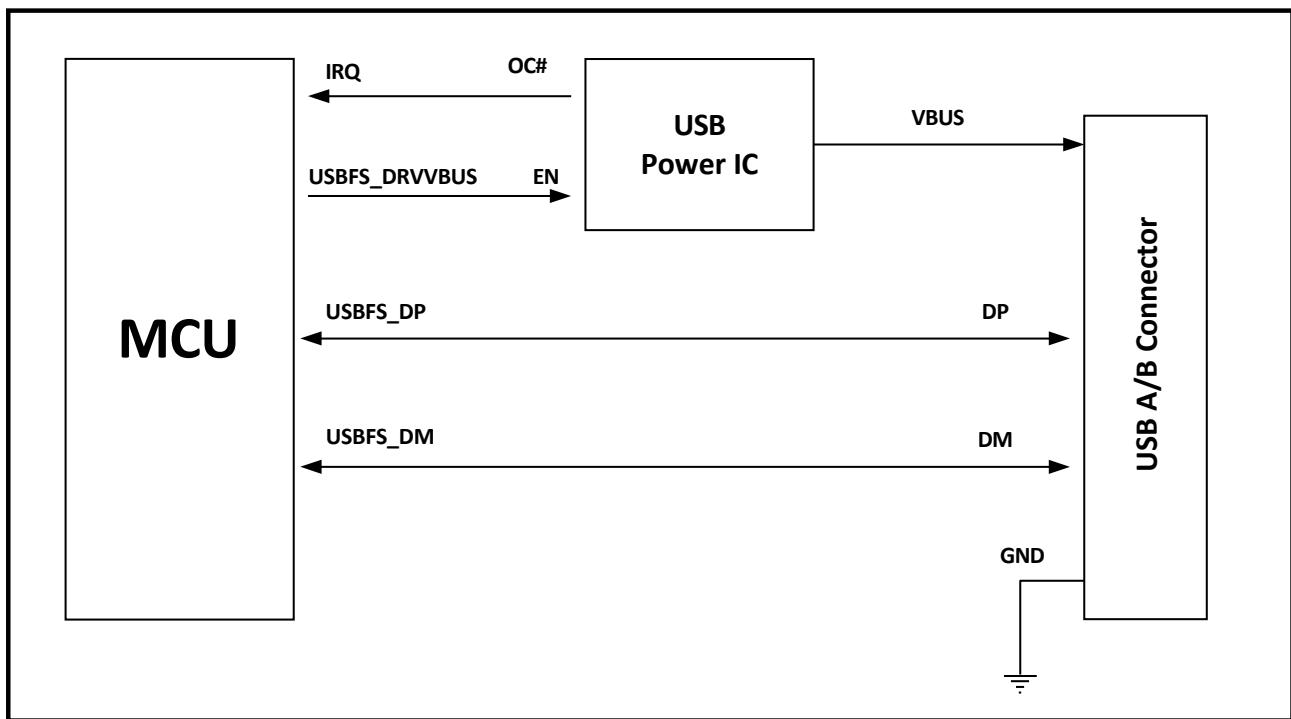


Figure 31-2 USBFS host mode system build diagram

31.5.3.2 Host port

**power
supply**

This MCU cannot output 5V to provide VBUS; for this, a USB power chip or basic power switch must be added to the microcontroller

(if the application board provides 5V power) to drive the 5V VBUS line. The external power chip can be driven through any GPIO output or USBFS_DRVVBUS. When the application determines to use GPIO to control the external device to provide VBUS, the Port Power bit in the Host Port Control and Status Register (PPWR bit in USBFS_HPRT) must still be set to 1.

31.5.3.3 Host detects device connection and disconnection

The USB device will be detected immediately after connection. the USBFS module will signal a host port interrupt, which is triggered by the Device Connection bit in the Host Port Control and Status register

(PCDET bit in USBFS_HPRT).

A device disconnect event will trigger a disconnect detection interrupt (DISCINT bit in USBFS_GINTSTS)

31.5.3.4 Host Enumeration

After a device connection is detected, if a new device is connected, the host must initiate the enumeration process by sending the USB reset and configuration commands to the new device.

The application drives the USB reset signal (single-ended zero) through USB by setting the Port Reset bit in the Host Port Control and Status Register (PRST bit in USBFS_HPRT) to 1, causing the process to last a minimum of 10 ms and a maximum of 20 ms. The application calculates the duration of this process and then clears the port reset bit to zero.

Immediately upon completion of the USB reset sequence, the Port Enable/Disable Change bit (PENCHNG bit in USBFS_HPRT) triggers a host port interrupt, which in turn notifies the application that a port speed field from the Host Port Control and Status registers is available

(PSPD in USBFS_HPRT) reads the speed of the enumerated device and that the host has started driving the SOF (FS) or Keep-alive token (LS). At this point the host is ready to complete the enumeration of the device by sending commands to the device.

31.5.3.5 Host hang up

The application hangs the USB bus by setting the port hang bit in the Host Port Control and Status Register (PSUSP in USBFS_HPRT) to 1. The USBFS module stops sending SOF and enters the hang state.

The bus can be brought out of the hang state by autonomous activity of the remote device (remote wakeup). In this case, the remote wakeup signal will trigger a remote wakeup interrupt (WKUPINT bit in USBFS_GINTSTS). The hardware puts the port recovery bit in the host port control and status registers (PRES bit in USBFS_HPRT) resets itself and automatically drives the recovery signal through USB. The application must time the recovery window, then clear the port recovery bit to exit the hang state and restart the SOF.

If the exit from the hang state is initiated by the host, the application must restore the port to position 1 to initiate the recovery signal on the host port, timing the recovery window and eventually clearing the port recovery bit to zero.

31.5.3.6 Host channel

The USBFS module implements 12 host channels. Each host channel can be used for USB host transfers (USB pipes). The host can handle up to 8 transfer requests simultaneously. If an application has more than 8 transfer requests pending, the Host Controller Driver (HCD) must reassign the channel for the unprocessed transfer requests after the channel is released from the previous task (i.e., after a transfer completion and channel stop interrupt is received).

Each host channel can be configured to support input/output and cyclic/non-cyclic transactions. Each host channel uses a dedicated control (HCCHARx) register, a transfer configuration (HCTSIZx) register/interrupt (HCINTx) **register**, and its associated interrupt mask register (HCINTMSKx).

Host channel control

Applications can control the host channel through the Host Channel x Characteristic Register (HCCHARx) as follows:

- Channel enable/disable
- Set the speed of the target USB device: FS/LS

- Set the address of the target USB device
- Set the number of the endpoint on the target USB device that communicates with this channel
- Set the transmission direction on this channel: IN/OUT
- Set the type of USB transmission on this channel: control/batch/interrupt/sync
- Set the maximum packet length of the device endpoints communicating with this channel
- Set the frame to be transmitted periodically: odd/even

Host channel transmission

The Host Channel Transfer Size Register (**HCTSIZx**) allows the application to program the transfer size parameters and read the transfer status. This register must be set prior to channel enable position 1 in the Host Channel Characteristics register. Immediately after enabling the endpoint, the packet count field becomes read-only while the USBFS module updates the field with the current transmission status.

The following transmission parameters can be programmed:

- Transfer size in bytes
- The number of packets that make up the entire transmission size
- Initial data PID

Host channel status/interrupts

The Host Channel x Interrupt register (**HCINTx**) indicates the status of the endpoint in the event of USB and AHB related events. When the host channel interrupt bit in the interrupt register (**HCINT** bit in **USBFS_GINTSTS**) is set to 1, the application must read these registers to obtain detailed information. Before reading these registers, the application must first read the Host All Channel Interrupt (**HCAINT**) register to obtain the channel number of the Host Channel x Interrupt register. The application must clear the corresponding bits in this register before it can clear the corresponding bits in the **HAINT** and **GINTSTS** registers. The **USBFS_HCINTMSK x** register also provides the mask bits for each interrupt source for each channel.

The host module provides the following status checking and interrupt generation functions:

- Transfer completion interrupt, indicating that both the application (AHB) and USB side have completed data transfer
- Channel stops due to transfer completion, USB transaction error, or application disable command
- The associated transmit FIFO is half empty or completely empty (IN endpoint)
- ACK response received
- NAK response received
- Received STALL response

-
- USB transaction errors due to CRC checksum failures, timeouts, bit fill errors, and incorrect EOPs
 - Crosstalk error
 - Overflow on frames
 - Flipped bit error for data synchronization

31.5.3.7 Host Scheduler

The host module has a built-in hardware scheduler that autonomously reorders and manages the USB transaction requests issued by the application. At the beginning of each frame, the host executes periodic (synchronous and interrupt) transactions first, followed by acyclic (control and bulk) transactions to comply with the USB specification's guarantee of high priority for synchronous and interrupt transfers.

The host processes USB transactions through request queues (a periodic request queue and an acyclic request queue). Each request queue can store up to 8 entries. The order in which USB transaction requests are written to the queue determines the order in which the transactions are executed on the USB interface.

At the beginning of each frame, the host processes the periodic request queue first and then the acyclic request queue. The host issues an outstanding periodic transmit interrupt (IPXFR bit in USBFS_GINTSTS) if a synchronous or interrupt type USB transmit transaction request scheduled to execute on the current frame is still pending at the end of the current frame. The USBFS module is responsible for the management of the periodic and non-periodic request queues. The periodic transmit FIFO and queue status registers (HPTXSTS) and the non-periodic transmit FIFO and queue status registers (HNPTXSTS) are read-only registers that applications can use to read the status of each request queue, including:

- Number of currently available free entries in the periodic (non-periodic) request queue (up to 8)
- Currently available free space in the periodic (non-periodic) TxFIFO (OUT transaction)
- IN/OUT tokens, host channel numbers, and other status information

Since each request queue can store up to 8 USB transaction requests, applications can send host USB transaction requests to the scheduler in advance; the actual communication will occur on the USB bus at the latest after the scheduler has finished processing the 8 pending periodic transactions and the 8 non-periodic transactions.

To issue a transaction request to the host scheduler (queue), application must read the PTXQSAV bit in the USBFS_HNPTXSTS register or the NPTQXSAV bit in the USBFS_HNPTXSTS register to ensure that there is at least one available space in the periodic (non-periodic) request queue to store the current request.

31.5.4 USBFS Device Features

31.5.4.1 Equipment Function Introduction

When the USBFS is operating in device mode, VBUS is the 5V power pin specified by the USB protocol and is a 5V voltage tolerant pin. This module always detects the level status of the VBUS line to connect or disconnect the device.

A typical USB device mode system build diagram is as follows:

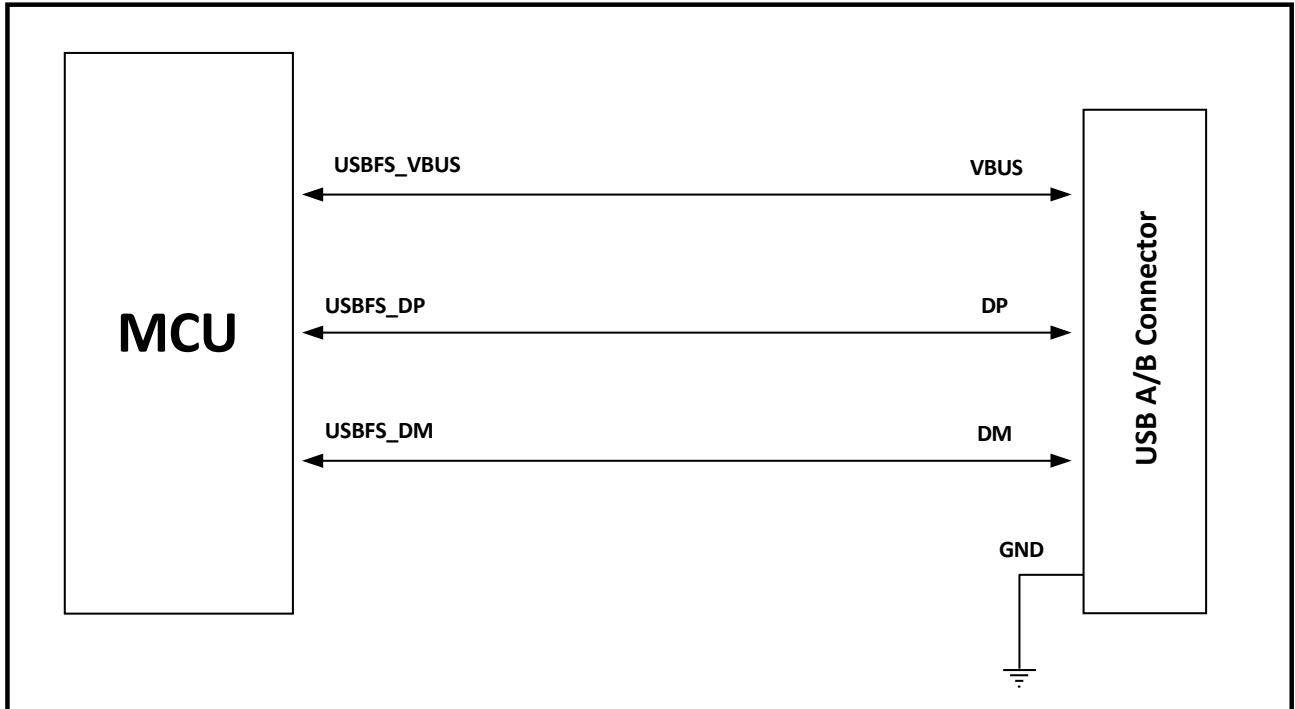


Figure 31-3 USBFS device mode system build diagram

31.5.4.2 Equipment

power

supply

status

When the module detects that USBFS_VBUS is high, it puts the USB device into a powered state. The USBFS then automatically connects the DP pull-up resistor, signals a full device to host connection and generates a session request interrupt (VBUSVINT bit in USBFS_GINTSTS) to indicate the power state is entered.

In addition, the USBFS_VBUS input ensures that the host provides a valid VBUS level during USB operation. If a low VBUS level is detected (e.g., triggered by power interference or host port shutdown) the USBFS will automatically disconnect.

When powered, the USBFS expects to receive a reset signal from the host. Other USB operations are not performed. A detected reset interrupt (USRST in USBFS_GINTSTS) is generated immediately after the reset signal is received. After the reset signal, an enumeration completion interrupt is generated

(ENUMDNE bit in USBFS_GINTSTS) the USBFS then enters the default state.

31.5.4.3 Device default state

By default, USBFS expects to receive the SET_ADDRESS command from the host. Other USB operations are not performed. When a valid SET_ADDRESS command is decoded on the USB, the application writes the corresponding address value to the device address word in the device configuration register

The USBF then enters the address state and is ready to answer to the host transaction with the configured USB address.

31.5.4.4 Device pending status

The USBFS device continuously monitors USB activity. After 3ms of USB idle time, an early suspend interrupt is issued (ESUSP bit in USBFS_GINTSTS) and the device is acknowledged to be in the pending state by the pending interrupt (USBSUSP bit in USBFS_GINTSTS) after 3ms. Then, the device hang bit (SUSPSTS bit in USBFS_DSTS) in the device status register is automatically set to 1 and the USBFS then enters the hang state.

The pending state can be exited via the device itself. In this case, the application will set the remote wake-up signal bit in the device control register (RWUSIG bit in USBFS_DCTL) is set to 1 and cleared to zero within 1ms to 15m.

However, if the device detects a recovery signal from the host, a recovery interrupt is generated (WKUPINT bit in USBFS_GINTSTS) and the device hang bit is automatically cleared to zero.

31.5.4.5 Device Soft Disconnect

The power supply state can be exited via software with the soft disconnect feature. Setting the soft disconnect bit in the Device Control Register (SDIS bit in USBFS_DCTL) to 1 removes the DP pull-up resistor, which causes a device disconnect detection interrupt on the host side, even though the USB cable is not actually unplugged from the host port.

31.5.4.6 Device

Endpoint

Endpoint

Category

The USBFS module implements the following USB endpoints:

- Control endpoint 0:
 - Two-way and only control messages are processed
 - Use a separate set of storages for IN and OUT transactions
 - Dedicated control
 - (USBFS_DIEPCTL0/USBFS_DOEPCTL0) Memory,
 - transmission configuration (USBFS_DIEPTSIZ0/USBFS_DOEPTSIZ0) Send
 - memory and and state state Interrupt Break
 - (USBFS_DIEPINT0/USBFS_DOEPINT0) registers. The bit groups available in the control and transfer size registers are slightly different from those in the other endpoints

- 5 IN endpoints
 - Each endpoint can be configured to support synchronous transmission, bulk transmission or interrupted transmission types
 - Each endpoint has a dedicated control (USBFS_DIEPCTLx) register, a transfer configuration (USBFS_DIEPTSIZx) register, and a status interrupt (USBFS_DIEPINTx) register
 - Device IN Endpoint General Interrupt Mask Register (USBFS_DIEPMSK) Can be used to enable/disable all IN endpoints (including EP0) on the same type of endpoint interrupt source

- Supports the Incomplete Synchronous IN Transfer interrupt (IISOIXFR bit in USBFS_GINTSTS) which will trigger when there is an incomplete transfer on at least one synchronous IN endpoint in the current frame. This interrupt is triggered together with the periodic frame interrupt (USBFS_GINTSTS/EOPF)
- 5 OUT endpoints
 - Each endpoint can be configured to support synchronous transmission, bulk transmission or interrupted transmission types
 - Each endpoint has a dedicated control (USBFS_DOEPCTLx) register, a transfer configuration (USBFS_DOEPTSIZx) register, and a status interrupt (USBFS_DOEPINTx) register
 - Device OUT Endpoint General Interrupt Mask Register (USBFS_DOEPMSK) can be used to enable/disable the same type of endpoint interrupt source on all OUT endpoints (including EP0)
 - Supports an incomplete synchronous OUT transfer interrupt (INCOMPISOOUT bit in USBFS_GINTSTS) which will trigger when there is an incomplete transfer on at least one synchronous OUT endpoint in the current frame. This interrupt is triggered together with the periodic frame interrupt (USBFS_GINTSTS/EOPF)

Endpoint Control

The application can take the following control over the endpoint via the Device Endpoint \times IN/OUT control register (DIEPCTLx/DOEPCTLx):

- Endpoint enable/ disable
- Activate the endpoint in the current configuration
- Set the USB transfer type (synchronous, bulk and interrupt)
- Set the supported packet size
- Set the Tx-FIFO number associated with the IN endpoint
- Set the data0/data1 PID to be received or used when sending (bulk/interrupt transfers only)
- Set the odd/even frame corresponding to the received or sent transaction (synchronous transmission only)
- The NAK bit can be set to reply to host requests with NAK regardless of the state of the FIFO at that time
- The STALL bit can be set so that all host tokens for this endpoint are replied to by hardware STALL
- The OUT endpoint can be set to listen mode, i.e. no CRC check is performed on the received data

Endpoint Transmission

The Device Endpoint \times Transmission Length Register (DIEPTSIx/DOEPTSIx) allows

the application to program the transmission length parameter and read the transmission status. This register must be set before endpoint enable position 1 in the Endpoint Control Register. Immediately after the endpoint is enabled, these fields become read-only while the USBFS module updates these fields based on the current transfer status.

The following transmission parameters need to be configured:

- Length of a single transmission in bytes
- The number of packets that make up the entire transmission

Endpoint Status/Status

The Device Endpoint x interrupt register (DIEPINT x /DOPEPINT x) indicates the status of the endpoint in the event of USB and AHB related events. When the OUT endpoint interrupt bit or IN endpoint interrupt bit (OEPINT bit in USBFS_GINTSTS or IEPINT bit in USBFS_GINTSTS, respectively) in the module interrupt register is set to 1, the application must read these registers to obtain detailed information. Before the application can read these registers, the Device All Endpoint Interrupt (USBFS_DAINT) register must be read to obtain the endpoint number of the Device Endpoint x Interrupt register. The application must clear the corresponding bits in this register before it can clear the corresponding bits in the DAINT and GINTSTS registers.

- The module provides the following status checking and interrupt functions:
- Transfer completion interrupt, indicating that both the AHB and USB sides of the application have completed data transfer
- Setup phase completed (for OUT endpoints of control transmission type only)
- The associated transmit FIFO is half empty or completely empty (IN endpoint)
- NAK answer has been sent to the host (IN endpoint for synchronous transmission only)
- IN token received when TxFIFO is empty (IN endpoint for bulk and interrupt transfer types only)
- OUT token received when endpoint is not yet enabled
- babble error detected
- Application closure endpoints take effect
- Application sets NAK in effect for endpoints (IN endpoints for synchronous transfer types only)
- More than 3 consecutive setup packets received (for control type OUT endpoints only)
- Timeout condition detected (IN endpoint for control transport type only)
- Packets of the synchronous transmission type are lost without generating interruptions

31.5.5 USBFS SOF pulse pin output function

The USBFS can monitor, track, and configure SOF frames in both host and device modes and also features SOF pulse output, which is output via the USBFS_SOF pin with an output width of 16 system clock cycles.

31.5.5.1 Host SOF

In host mode, the number of PHY clocks that occur during the two consecutive SOF (FS) or keep-alive (LS) tokens generated can be programmed in the Host Frame Interval Register (HFIR), which in turn allows the application to control the SOF frame period. An interrupt is generated at the start of the frame (SOF bit in USBFS_GINTSTS). The current frame number and the time remaining before the next SOF occurs can be tracked by the application in the Host Frame

Number Register (HFNUM).

Use the SOFEN bit in the USBFS system control register USBFS_SYCTLREG to enable a simultaneously generated SOF pulse signal of 16 system clock cycles in width from any SOF token to be output from the USBFS_SOF pin.

In addition, SOF pulses can be used as internal events to trigger external modules such as DMA transfers, TIMER counts, etc.

31.5.5.2 Equipment SOF

In device mode, the USB will trigger a Start of Frame interrupt (SOF bit in USBFS_GINTSTS) every time a SOF token is received. The corresponding frame number can be read from the device status register (FNSOF bit in USBFS_DSTS). Using the SOFEN bit in the USBFS system control register USBFS_SYCTLREG, a SOF pulse signal with a width of 16 system clock cycles can also be generated and made available externally by outputting the signal at the USBFS_SOF pin.

In addition, SOF pulses can be used as internal events to trigger external modules such as DMA transfers, TIMER counts, etc.

The periodic end-of-frame interrupt (GINTSTS/EOPF) is used to notify the application when 80%, 85%, 90 %, or 95% of the frame interval time has elapsed, depending on the periodic frame interval field in the device configuration register (PFIVL bit in USBFS_DCFG). This function can be used to determine if all synchronous communication for the frame is complete.

31.5.6 USBFS power control

When the USBFS module is not in use, the HCLK and PHY clocks of the USBFS module can be stopped via the CMU module, thereby reducing power consumption. When using the USB module, but the device USB session is not started or the device is not connected, you can use the power reduction technique in the USB hang state.

- Stop PHY clock (STPPCLK bit in USBFS_GCCTL)

With the Stop PHY clock position 1 in the Clock Gating Control Register, most of the 48 MHz internal clock domain of the USBFS full-speed module is turned off by clock gating. Even if the application still provides clock input, the dynamic power consumption of the module due to clock signal flipping will be saved.

- HCLK gating (GATEHCLK bit in USBFS_GCCTL)

By setting GATEHCLK in the Clock Gating Control Register to position 1, most of the system clock domains inside the USBFS module are turned off by clock gating. Only the register read and write interfaces remain active. The dynamic power consumption of the module due to clock signal flip-flops is saved even if the application still provides clock input.

To save dynamic power, the USB data FIFO is only clocked when it is accessed by the USBFS module.

31.5.7 USBFS dynamically updates the USBFS_HFIR register

In host mode, the USB module has the ability to dynamically fine tune the frame period to synchronize external devices with SOF frames. If the USBFS_HFIR register is changed within the current SOF frame, the SOF cycle will be corrected accordingly in the next frame, as described in Figure 31-4.

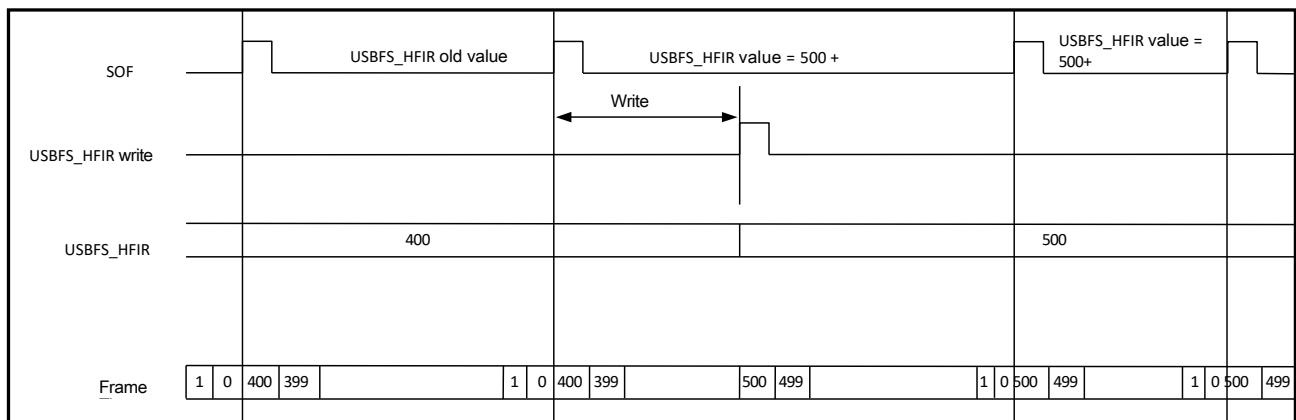


Figure 31-4 USBFS dynamic update USBFS_HFIR register schematic

31.5.8 USBFS Data FIFO

The USBFS system has 1.25KB of dedicated RAM and uses an efficient FIFO control mechanism. the packet FIFO controller module in the USBFS module divides the RAM space into multiple TxFIFOs (into which the application presses data for transient storage before the USB transfer) and a single RxFIFO (into which data received from the USB is transiently stored before it is read by the application). (where data received from USB is stored briefly before it is read by the application)

The number and organization of the FIFOs built in RAM depends on the role of the device. In device mode, one TxFIFO is configured for each active IN endpoint. the size of the FIFOs are all configured by software to better meet the application requirements.

31.5.9 USBFS Host FIFO Architecture

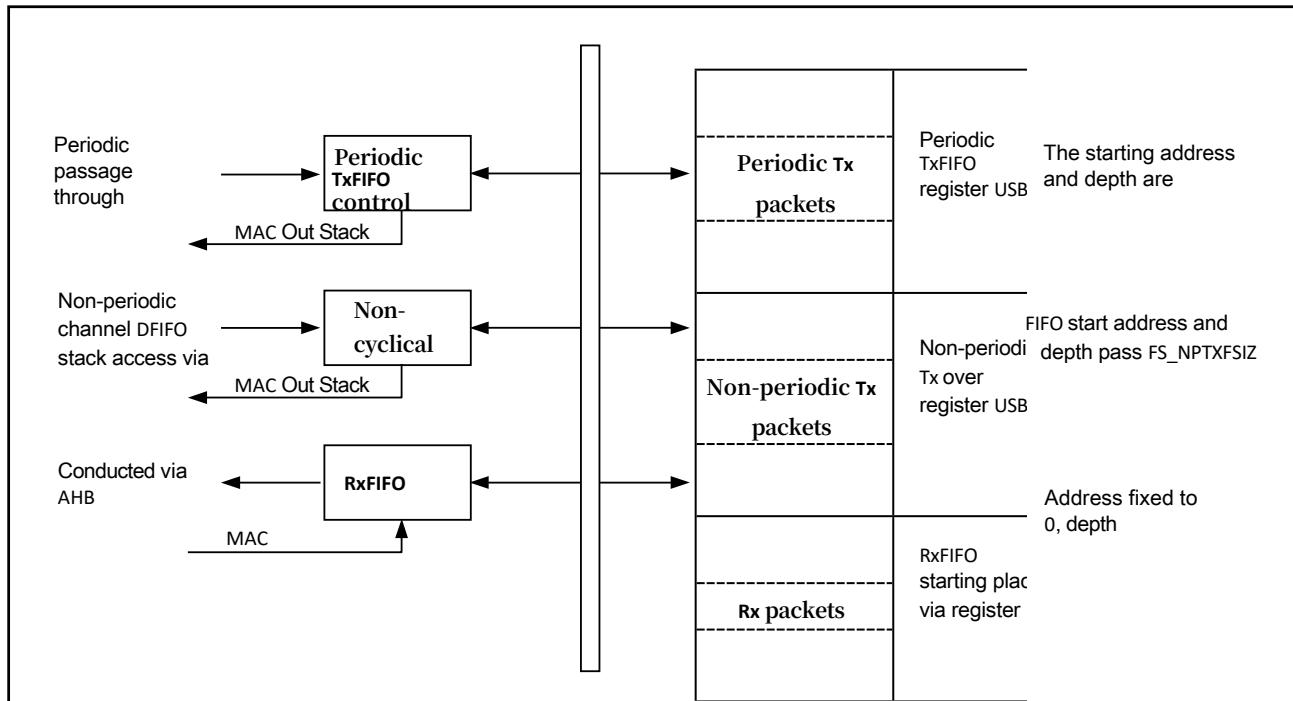


Figure 31-5 Diagram of FIFO architecture in USBFS host mode

31.5.9.1 Host RxFIFO

The host uses a receive FIFO to handle all cyclic and acyclic transactions. the FIFO is used as a receive buffer to hold the data received from the USB (the data portion of the received packet) until it is transferred to system memory. Packets from the device IN endpoint are received and stored one by one as long as there is space in the FIFO. The status of each received packet (containing the host destination channel, number of bytes, data PID, and checksum of the received data) is also stored in the FIFO. The size of the receive FIFO is configured in the Receive FIFO Size Register (GRXFSIZ).

The single receive FIFO architecture allows the USB host to efficiently fill the receive data buffer:

- All IN configuration host channels share the same RAM buffer (shared FIFO)
- For any sequence of IN tokens driven by the host software, the USBFS module can fill the receive FIFO to the limit

The application receives Rx FIFO non-air breaks as long as at least one packet is available for reading in the RxFIFO. The application reads the packet information from the receive status read and out stack registers and finally reads the data from the RxFIFO.

31.5.9.2 Host TxFIFO

The host uses one transmit FIFO for all non-periodic (control and bulk) OUT transactions and another transmit FIFO for all periodic (synchronous and interrupt) OUT transactions. the FIFO is used as a transmit buffer to hold the data to be sent over USB (transmit packets) The size of the periodic(non-

periodic) TxFIFO is configured in the Host Periodic (non-periodic) Send FIFO Size (HPTXFSIZ/HNPTXFSIZ) register.

The two Tx FIFOs operate on a priority basis, with cyclic communication having higher priority, so cyclic communication occurs first within the time of a USB frame. At the start of the frame, the built-in host scheduler processes the periodic request queue first and then the non-periodic request queue.

The two transmit FIFO architecture allows the USB host to optimize the management of the cyclic and acyclic transmit data buffers separately:

- All host channels configured to support periodic (non-periodic) OUT transactions share the same RAM buffer (shared FIFO)
- For any sequence of OUT tokens driven by the host software, the USBFS module can fill the periodic (non-periodic) transmit FIFO to the limit

The USBFS module issues a periodic TxFIFO null break whenever the periodic TxFIFO is half or full null.

(PTXFE bit in USBFS_GINTSTS) depending on the value of the periodic Tx-FIFO empty level bit (PTXFELVL bit in USBFS_GAHBCFG) in the AHB configuration register. As long as there is free space in both the periodic TxFIFO and the periodic request queue, the application can write transmit data in advance. You can find out how much space is available in both the Host Periodic Send FIFO and the Queue Status Register (HPTXSTS) by reading them.

The USBFS module issues a non-periodic TxFIFO null break whenever the non-periodic TxFIFO is half-empty or fully empty

(NPTXFE bit in USBFS_GINTSTS) depending on the non-periodic TxFIFO free level bit in the AHB configuration register (TXFELVL bit in USBFS_GAHBCFG) The application can write transmit data as long as there is free space in both the acyclic TxFIFO and acyclic request queues. You can find out how much space is available in both the host acyclic send FIFO and the queue status register (HNPTXSTS) by reading them.

31.5.10 USBFS Device FIFO Architecture

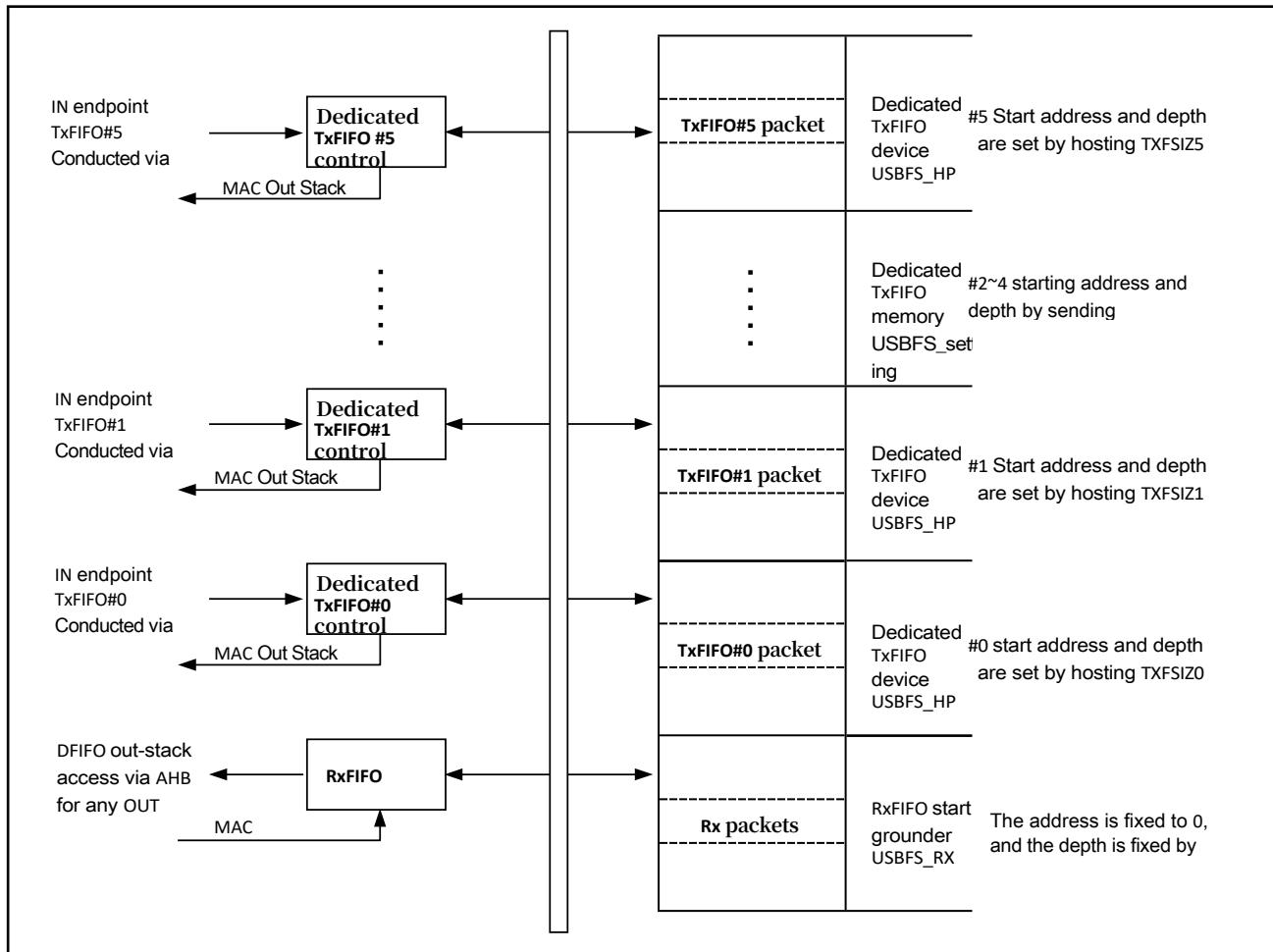


Figure 31-6 Diagram of FIFO architecture in USBFS device mode

31.5.10.1 Device RxFIFO

USBFS devices use a single receive FIFO to receive data sent to all OUT endpoints. In addition to valid data, the status of the received packets (including OUT endpoint target number, byte count, data PID, and validation of the received data) is stored by the module. When there is no space available, the device replies with a host transaction NAK answer and triggers an interrupt on the addressed endpoint. The size of the receive FIFO is configured in the Receive FIFO Size Register (GRXFSIZ).

The single receive FIFO architecture allows USB devices to fill the receive RAM buffer more efficiently:

- All OUT endpoints share the same RAM buffer (shared FIFO)
- For any host sequence OUT token, the USBFS module can fill the receive FIFO to the limit

The application will always receive Rx FIFO non-air breaks as long as there is at least one packet in the Rx FIFO available for reading

(RXFNE bit in USBFS_GINTSTS) The application reads the packet information from the receive status read and out stack registers (GRXSTSP) and finally reads the corresponding data from the receive FIFO by reading the out stack address associated with the endpoint.

31.5.10.2 Device TxFIFO

The module provides a dedicated FIFO for each IN endpoint. The application configures the FIFO size for IN endpoint 0 via the Off-Cycle Send FIFO Size register (USBFS_DIEPTSI0) and for IN endpoint x via the Device IN Endpoint Send FIFOx register (DIEPTSIx).

31.5.11 USBFS FIFO RAM allocation

31.5.11.1 Host

mode receives FIFO

RAM allocation

Status information is written to the FIFO with each received packet, so at least (max packet size / 4) + 1 space must be allocated for the received packet. If multiple synchronization channels are enabled, the space allocated for receiving consecutive packets must be at least (max packet size / 4) + 1. Typically, the recommended space is twice (max packets/4 + 1), so that the USB can receive subsequent packets at the same time as the previous packet is transmitted to the CPU.

Transmission completion status information is written to the FIFO along with the last packet received by the endpoint, so a location must be allocated for this purpose.

When running in DMA mode, the DMA address registers for each host channel will be stored in the FIFO, so it is necessary to reserve a place in the FIFO for each channel to store its address registers.

Send FIFO RAM allocation

The minimum RAM required by the host to send FIFOs acyclically is the size of the largest packet transmitted on all supported acyclic OUT channels.

Typically, the recommended space is twice the maximum packet size, so that while the USB is sending the current packet, the AHB can fill the sending FIFO with the next packet.

The minimum RAM required for a host to send a FIFO periodically is the size of the largest packet transmitted on all supported periodic OUT channels. If there is at least one synchronous OUT endpoint, the space must be at least twice the size of the largest packet in that channel.

When running in DMA mode, the DMA address registers for each host channel will be stored in the FIFO, so it is necessary to reserve a place in the FIFO for each channel to store its address registers.

31.5.11.2 Device

Mode Receive FIFO

RAM Allocation

The application should allocate RAM for SETUP packets. 11 locations must be reserved in the receive FIFO to receive SETUP packets on the control endpoint. the USBFS module will not write any other data to these locations reserved for SETUP packets. A location will be assigned for the global OUT NAK. Status information is written to the FIFO with each receive packet, so at least (maximum packet size/4) + 1 space must be allocated for the receive packet. If multiple synchronization endpoints are enabled, the space allocated for receiving consecutive packets must be at least

(In general, the recommended space is twice (max packet size/4 + 1) so that the USB can receive subsequent packets at the same time as the previous packet is transmitted to the CPU.

Transmission completion status information is pushed into the FIFO along with the last packet received by that endpoint. It is usually recommended to allocate a location for each OUT endpoint.

Send FIFO RAM allocation

The minimum RAM space required for each IN endpoint to send a FIFO is the maximum packet size for that particular IN endpoint.

31.5.12 USBFS System Performance

Optimal USB and system performance is achieved with large RAM buffers, highly configurable FIFO sizes, fast 32-bit FIFO access via AHB stack-up/stack-down registers, and especially the advanced FIFO control mechanism. In fact, USBFS can efficiently fill the available RAM space through this mechanism regardless of the current USB sequence. With these features:

- The application has sufficient margin to calculate and correct the CPU load to optimize CPU bandwidth utilization:
 - Applications can accumulate a large amount of send data before sending it out via USB
 - Enables sufficient time margin to read data from the receive FIFO
- The USB module is capable of maintaining full speed operation, i.e. providing maximum bandwidth at full speed (as much hardware autonomy as possible, with as little software involvement as possible)
 - The USB module can accumulate a large amount of transmit data at its disposal in advance, allowing for autonomous management of USB data transmission
 - Large amount of empty space in the receive buffer, which can be filled automatically with data from the USB

Because the USBFS module can efficiently fill a 1.25KB RAM buffer and 1.25KB of send/receive data is sufficient for a full-speed frame, USB systems can reach maximum USB bandwidth in a single frame without application intervention.

31.5.13 USBFS Interrupts and Events

Global interrupts are the main interrupts to be handled by the software. The flag bits of the global interrupts can be read in the USBFS_GINTSTS register.

Table 31-2 USBFS Interrupt Events Table

Interruption flags	Description	Operation mode	Internal event sources
WKUPINT	Resume/remote wake-up interrupt	Host or device	–
VBUSVINT	VBUS valid interrupt	Equipment	–
DISCINT	Disconnection interruption	Mainframe	–
CIDSCHG	Connector ID line status change interrupt	Host or device	–
PTXFE	Periodic TxFIFO Air Breaks	Mainframe	–
HCINT	Host channel interruption	Mainframe	–
HPRTINT	Host port interruption	Mainframe	–
DATAFSUSP	Data acquisition hangs	Equipment	–
IPXFR/INCOMPISOOUT	Incomplete cyclic transmission / Incomplete OUT synchronous transmission	Equipment	–
IISOIXFR	Incomplete IN sync transfer	Equipment	–
OEPINT	OUT endpoint interrupt	Equipment	–
IEPINT	IN endpoint interrupt	Equipment	–
EOPF	Periodic end-of-frame interruptions	Equipment	–
ISOODRP	Dropping synchronous OUT packet interrupts	Equipment	–
ENUMDNE	Enumeration completed	Equipment	–
USBRST	USB reset interrupt	Equipment	–
USBSUSP	USB hang interrupt	Equipment	–
ESUSP	Early hang-up interruption	Equipment	–
GONAKEFF	Global OUT NAK valid interrupt	Equipment	–
GINAKEFF	Global non-cyclic IN NAK valid interrupt	Equipment	–
NPTXFE	Non-periodic TxFIFO air break	Mainframe	–
RXFNE	RxFIFO non-air break	Host or device	–
SOF	Frame start interrupt	Host or device	Yes
MMIS	Pattern mismatch interrupt	Host or device	–

31.6 USBFS Programming Model

31.6.1 USBFS module initialization

The application must execute the module initialization sequence.

Please refer to 31.5.2 USBFS Mode Decision for the mode decision method.

This section describes the initialization process after the USBFS controller is powered up. The initialization sequence must be followed by the application, whether it is working in host or device mode. Initialize all module global registers according to the module configuration:

1. Program the following fields in the USBFS_GAHBCFG register:
 - Global interrupt mask bit GINTMSK = 1
 - RxFIFO non-empty (RXFNE bit in USBFS_GINTSTS)
 - Periodicity TxFIFO Empty Threshold
2. Program the following fields in the USBFS_GUSBCFG register:
 - FS Timeout calibration field
 - USB Turnaround Time Field
3. The software must unmask the following bits in the USBFS_GINTMSK register:
 - Pattern mismatch interrupt mask
4. By reading the CMOD bit in USBFS_GINTSTS, the software can determine whether the USBFS controller is operating in host mode or device mode.

31.6.2 USBFS Host Initialization

To initialize the module as a host, the application must perform the following steps:

1. Program HPRTINT in the USBFS_GINTMSK register to unmask.
2. Program the USBFS_HCFG register to select the full-speed host.
3. Program the PPWR bit in USBFS_HPRT to 1 to provide VBUS to the USB bus.
4. Wait for the PCDET interrupt in USBFS_HPRT. This indicates that a device is connected to the host port.
5. Program the PRST bit in USBFS_HPRT to 1 to signal a reset on the USB bus.
6. Wait at least 10ms for the reset process to complete.
7. Program the PRST bit in USBFS_HPRT to 0.
8. Wait for the PENCHNG interrupt in USBFS_HPRT.
9. Read the PSPD bit in USBFS_HPRT to get the enumeration speed.
10. Using the selected PHY clock, set the HFIR register accordingly.
11. Program the FSLSPCS field in the USBFS_HCFG register based on the device speed detected in step 9. If the FSLSPCS is changed, a port reset must be performed.

12.Program the USBFS_GRXFSIZ register to select the size of the receive FIFO.

13. Program the USBFS_HNPTXFSIZ register to select the size and starting address of the non-periodic transmit FIFO used for non-periodic communication transactions.

14. Program the USBFS_HPTXFSIZ register to select the size and starting address of the periodic communication send FIFO for periodic transactions.

To communicate with the device, the system software must initialize and enable at least one channel.

31.6.3 USBFS device initialization

During power-up or after switching from host mode to device mode, the application must perform the following steps to initialize the module as a device.

1. Program the following fields in the USBFS_DCFG register:
 - Equipment speed
 - Non-zero length state OUT Handshake signal
2. Program the USBFS_GINTMSK register to unmask the following interrupts:
 - USB reset
 - Enumeration completed
 - Hang up early
 - USB Hang Up
 - SOF
3. Wait for VBUSVINT in USBFS_GINTSTS to interrupt, indicating that it enters the power supply state.
4. Wait for the USBRST interrupt in USBFS_GINTSTS. This indicates that a reset signal has been detected on the USB, and the reset process lasts approximately 10ms from the time this interrupt is received.
5. Wait for the ENUMDNE interrupt in USBFS_GINTSTS. This interrupt indicates the end of the reset process on the USB. When this interrupt is received, the application must read the USBFS_DSTS register to determine the enumeration rate and perform the steps listed in Endpoint Initialization at Enumeration Completion.

At this point, the device is ready to accept SOF packets and perform a control transmission on control endpoint 0.

31.6.4 USBFS DMA Mode

USB uses the AHB host interface to obtain transmit packet data (AHB to USB) and receive data updates (USB to AHB). AHB host interface uses programmed DMA addresses (HCDMAx registers in host mode and DIEPDMAx/DOEPDMAx registers in device mode) to access the data buffers.

31.6.5 USBFS Host Programming Model

31.6.5.1 Channel initialization

The application must initialize one or more channels before it can communicate with the connected device.

To initialize and enable the channel, the application must perform the following steps:

1. Program the USBFS_GINTMSK register to unmask interrupts for the following bits:
 - The non-periodic transmit FIFO for OUT transactions is empty (applies when working at the pipeline transaction level and the packet count field is programmed with a value greater than 1)
 - The non-periodic transmit FIFO for OUT transactions is half empty (applies when working at the pipeline transaction level and the packet count field is programmed with a value greater than 1)
2. Program the USBFS_HAINTMSK register to enable the selected channel interrupt.
3. Program the USBFS_HCINTMSK register to enable the interrupts related to communication transactions reflected in the Host Channel Interrupt Register.
4. Program the USBFS_HCINTSIZx register for the selected channel to specify the total transfer size in bytes and the expected number of packets, including short packets. The application must program the PID field using the initial data PID (for the first OUT transaction or expected from the first IN transaction).
5. Program the USBFS_HCCHARx register of the selected channel to specify the endpoint characteristics of the device, such as type, speed, direction, ~~enable~~ by setting channel enable position 1 only when the application is ready to send or receive packets)

31.6.5.2 Channel stop

Applications can disable any channel by programming the USBFS_HCCHARx register to CHDIS and CHENA position 1. This causes the USBFS host to clear previous requests (if any) made on the channel and generate a channel stop interrupt. The application must wait for a CHH interrupt in USBFS_HCINTx before reassigning the channel to another communication transaction. the USBFS host will not interrupt a communication transaction that has been started on the USB.

Before disabling a channel, the application must ensure that there is at least one free space in either the non-periodic request queue (when disabling non-periodic channels) or the periodic request queue (when disabling periodic channels). The application can clear the request queue when it is full (before disabling the channel) by programming the USBFS_HCCHARx register to CHDIS position 1 and clearing the CHENA bit to zero. The application will disable the channel when either of the following conditions occurs:

1. STALL, TXERR, BBERR, or DTERR interrupts are received in USBFS_HCINTx for the IN or OUT channel. The application must be able to receive other interrupts for the same channel (DTERR, Nak, Data, TXERR) receiving the channel stop signal.
2. Receives a DISCINT(Disconnect Device) interrupt in USBFS_GINTSTS. (The application will disable all enabled channels)

3. The application will stop it before the transfer completes properly.

In DMA mode, the application cannot stop an inseparable periodic transfer with a write register.

31.6.6 USBFS Device Programming Model

31.6.6.1 USB reset with endpoint endpoint initialization

1. Set NAK position 1 for all OUT endpoints
 - SNAK = 1 in USBFS_DOEPCTLx (for all OUT endpoints)
2. Unblock the following interrupt bits
 - In USBFS_DAIINTMSK, INEP0=1 (control 0 IN endpoint)
 - In USBFS_DAIINTMSK, OUTEP0=1 (control 0 OUT endpoint)
 - In DOEPMSK, STUP=1
 - XFRC=1 in DOEPMSK
 - In DIEPMSK, XFRC=1
 - In DIEPMSK, TOC=1
3. Set data FIFO RAM for each FIFO
 - Program the USBFS_GRXFSIZ register to be able to receive OUT data and SETUP data for control transmissions. This register must be equal to at least 1 maximum packet size for control endpoint 0 + 2 words (for controlling the status of OUT packets) + 10 words (for SETUP packets)
 - Program the USBFS_TX0FSIZ register (depending on the selected FIFO number) to be able to send control IN data. This register must be equal to at least 1 maximum packet size for control endpoint 0.
4. Program the following fields in the endpoint-related registers to enable control of OUT endpoint 0 to receive SETUP packets
 - STUPCNT=3 in USBFS_DOEPTSIZ0 (receive up to 3 consecutive SETUP packets)At this point, all initialization required to receive SETUP packets is complete.

31.6.6.2 Endpoint initialization during USB reset

1. In the enumeration completion interrupt (ENUMDNE in USBFS_GINTSTS), the USBFS_DSTS register is read to determine the enumeration speed of the device.
2. Program the MPSIZ field in USBFS_DIEPCTL0 to set the maximum packet size. This step configures control endpoint 0. The maximum packet size for the control endpoint depends on the enumeration speed.

At this point, the device is ready to receive SOF packets and is configured to perform control transmissions at control endpoint 0.

31.6.6.3 Endpoint initialization upon receipt of SetAddress command

This section describes the actions that an application must perform when it receives a SetAddress command in a SETUP packet.

-
1. Use the device address received in the SetAddress command to program the USBFS_DCFG register
 2. Programming the module to send IN packets for the status phase

31.6.6.4 Endpoint initialization upon receipt of SetConfiguration/SetInterface command

This section describes the actions that an application must perform when receiving a SetConfiguration or SetInterface command in the SETUP package.

1. When the SetConfiguration command is received, the application must program the endpoint registers to configure them using the characteristics of the valid endpoints in the new configuration.
2. When the SetInterface command is received, the application must program the endpoint register of the endpoint specified by the command.
3. Endpoints that were valid in the previous configuration or other settings are not valid in the new configuration or other settings. These invalid endpoints must be deactivated.
4. Use the USBFS_DAINTMSK register to enable interrupts for valid endpoints and mask interrupts for invalid endpoints.
5. Set the data FIFO RAM for each FIFO.
6. After configuring all the required endpoints, the application must program the module to send IN packets in the status phase. At this point, the device module is ready to receive and send any type of packet

31.6.6.5 Endpoint Activation

This section describes the steps required to activate a device endpoint or to configure an existing device endpoint to a new type.

1. Program the characteristics of the desired endpoint in the following fields of the USBFS_DIEPCTLx register (for IN or bidirectional endpoints) or the USBFS_DOEPCTLx register (for OUT or bidirectional endpoints).
 - Maximum packet size
 - USB Active Endpoint Location 1
 - Endpoint initial data synchronization bits (for interrupt and bulk endpoints)
 - Endpoint Type
 - TxFIFO Number
2. Once the endpoint is activated, the module starts decoding the token sent to that endpoint and replies with a valid handshake signal if the received token is valid.

31.6.6.6 Endpoint deactivation

This section describes the steps required to deactivate an existing endpoint.

1. In the endpoint to be deactivated, clear the USBFS_DIEPCTLx register (for IN or bi-directional endpoints) or the USBFS_DOEPCTLx register (for

OUT or bi-directional endpoints) to clear the USB active endpoint bit in the USBFS_DIEPCTLx register.

2. When an endpoint is disabled, the module ignores the token sent to that endpoint, resulting in a USB timeout.

31.6.7 USBFS Operating Model

31.6.7.1 SETUP and OUT data transfer

This section describes the internal data flow and application operation steps during the data OUT transfer and SETUP transactions.

Packet Reading

This section describes how to read packets (OUT data and SETUP packets) from the receive FIFO.

1. When the RXFNE interrupt (USBFS_GINTSTS register) is captured, the application must read the receive status pop-up register (USBFS_GRXSTSP).
2. The application can mask the RXFNE interrupt (in USBFS_GINTMSK) by writing RXFNE=0 (in USBFS_GINTSTS) until it reads the packet out of the receive FIFO.
3. If the byte count of the received packet is not 0, the data is ejected from the receive data FIFO and stored in memory. If the received packet byte count is 0, no data is ejected from the receive data FIFO.
4. The status of the packet read from the receive FIFO has the following states:
 - Global OUT NAK:
PKTSTS=Global OUT NAK, BCNT=0x000, the values of EPNUM and DPID are irrelevant.
These data indicate that the global OUT NAK bit is in effect.
 - SETUP packet:
PKTSTS=SETUP, BCNT=0x008, EPNUM=control EP number, DPID=D0. These data indicate that the SETUP packet received on the specified endpoint is now readable from the receive FIFO.
 - Establishment phase completed:
PKTSTS=Build phase complete, BCNT=0x0, EPNUM=Control EP number, DPID value is irrelevant.
These data indicate that the build phase of the specified endpoint is complete and the data phase has been initiated. After this status entry is ejected from the receive FIFO, the module will generate an establishment interrupt on this control OUT endpoint.
 - OUT packets:
PKTSTS=DataOUT, BCNT=size of received OUT packet (BCNT:0~1024),
EPNUM=endpoint number of received packet, DPID=actual data PID.
 - Data transfer completed:
PKTSTS=OUT data transfer complete, BCNT=0x0, EPNUM=Out EP number of completed data transfer, DPID value is irrelevant.
These data indicate the completion of the OUT data transfer for the specified OUT endpoint.
After this status entry is ejected from the receive FIFO, the module will raise a "transmission

"complete" interrupt at the specified OUT endpoint.

5. After ejecting data from the receive FIFO, the RXFNE interrupt must be unmasked (USBFS_GINTSTS).
6. Steps 1 through 5 will be repeated each time the application detects an RXFNE interrupt in USBFS_GINTSTS. reading an empty receive FIFO may result in undefined module behavior.

SETUP transaction

This section describes the way the module handles SETUP packets and the order in which the application handles SETUP transactions. Application requirements:

1. To receive SETUP packets, the control OUT endpoint STUPCNT field (USBFS_DOEPTSIZE0) must be programmed with a non-zero value. If the application programs the STUPCNT field to a non-zero value, the module receives the SETUP packet and writes it to the receive FIFO, regardless of the NAK status and the EPENA bit setting in USBFS_DOEPCTL0. The STUPCNT field is decremented for each SETUP packet received by the control endpoint. If the STUPCNT field is not programmed to the appropriate value prior to receiving a SETUP packet, the module can still receive SETUP packets and decrement the STUPCNT field, but the application may not be able to determine the correct number of SETUP packets to receive during the build phase of the control transfer.
 - In USBFS_DOEPTSIZE0, STUPCNT=3
2. The application must always allocate some extra space in the receive data FIFO to be able to receive up to three consecutive SETUP packets on the control endpoint.
- Space is reserved for 10 words. The first SETUP packet requires 3 words, "build phase complete" status double word requires 1 word, and 6 words are needed to store two additional SETUP packets.
- Each SETUP packet requires 3 words to store 8 bytes of SETUP data and 4 bytes of SETUP status. The module will reserve these spaces in the receive FIFO.
- This FIFO is only used for storing SETUP packets and will never use that space for packets.
3. The application must read the 2 words of the SETUP packet from the receive FIFO.
4. The application must read and discard the "build phase complete" status word from the receive FIFO

Internal data flow:

1. When a SETUP packet is received, the module writes the received data to the receive FIFO without checking the available space in the receive FIFO and without regard to the NAK and STALL bit settings of the endpoint.
- The module will internally place the IN NAK and OUTNAK position 1 of the control IN/OUT endpoint where the SETUP packet is received.
2. For each SETUP packet received on the USB, the module writes 3 words of data to the receive FIFO and decrements the STUPCNT field by 1.
 - The first word contains the internal control information used by the module
 - The second word contains the first 4 bytes of the SETUP command

- The third word contains the last 4 bytes of the SETUP command
3. When the build phase ends and the data IN/OUT phase begins, the module writes a status entry ("build phase complete" word) to the receive FIFO, indicating that the build phase is complete.
 4. On the AHB side, the SETUP packet is read by the application.
 5. The module will use the STUP interrupt when the application pops up the word "build phase complete" from the receive FIFO

(USBFS_DOEPINTx) to interrupt the application and indicate that it can process the received SETUP packet.

- The module will clear the endpoint enable bit of the control OUT endpoint.

Application programming sequence:

1. Program the USBFS_DOEPTSIZ0 register.
- STUPCNT=3
2. Wait for the RXFNE interrupt (USBFS_GINTSTS) and read the packet from the receive FIFO.
3. Triggering of the STUP interrupt (USBFS_DOEPINTx) indicates successful completion of SETUP data transfer.
- When this interrupt occurs, the application must read the USBFS_DOEPTSIZx register to determine the number of SETUP packets received and process the last SETUP packet received.

To process more than three consecutive SETUP packets:

According to the USB 2.0 specification, a host will not normally send more than three consecutive SETUP packets to the same endpoint in a SETUP packet error. However, the USB 2.0 specification does not limit the number of consecutive SETUP packets that a host can send to the same endpoint. When this happens, the USBFS controller will generate an interrupt (B2BSTUP in USBFS_DOEPINTx)

Set global OUT NAK to 1

Internal data flow:

1. If the application sets the global OUT NAK (SGONAK bit in USBFS_DCTL) to 1, the module will stop writing data other than SETUP packets to the receive FIFO. Regardless of the amount of space available in the receive FIFO, the device replies NAK to unsynchronized OUT tokens sent by the host, and simply ignores synchronized OUT packets.
2. The module writes the global OUT NAK to the receive FIFO. the application must leave enough space for this.
3. The module will set the GONAKEFF interrupt (USBFS_GINTSTS) to 1 when the application ejects the global OUT NAK word from the receive FIFO.
4. When the application detects this interrupt, it assumes that the module is in global OUT NAK mode. The application can clear this interrupt by clearing the SGONAK bit in USBFS_DCTL to zero.

Application programming sequence:

1. To stop receiving any type of data into the receive FIFO, the application must program the following fields to set the global OUT NAK to position 1.
 - In USBFS_DCTL, SGONAK =1
2. Wait for the GONAKEFF interrupt in USBFS_GINTST. Once triggered, this interrupt indicates that the module has stopped receiving any type of data other than SETUP packets.

- If the application has set the SGONAK position in USBFS_DCTL application can receive valid OUT packets before the module raises GONAKEFF interrupt (USBFS_GINTSTS).
3. The application can temporarily mask this interrupt by performing a write operation to the GINAKEFFM bit in the USBFS_GINTMSK register.
- In the USBFS_GINTMSK register, GINAKEFFM=0
4. SGONAK bit in USBFS_DCTL must be cleared to zero when the application is ready to exit the global OUT NAK mode. This operation also clears the GONAKEFF interrupt (USBFS_GINTSTS).
- In USBFS_DCTL, CGONAK=1
5. If the application has previously masked this interrupt, it must unmask it as follows:
- In GINTMSK, GINAKEFFM=1

Set global OUT NAK to 1

The application must disable the enabled OUT endpoints using the following sequence. Application programming sequence:

1. Before disabling any OUT endpoints, the application must enable global OUT NAK mode in the module.
 - In USBFS_DCTL, SGONAK=1
2. Wait for GONAKEFF interrupt (USBFS_GINTSTS)
3. Disable the OUT endpoint by programming the following fields:
 - In USBFS_DOEPCTLx, EPDIS=1
 - In USBFS_DOEPCTLx, SNAK=1
4. Wait for the EPDISD interrupt (USBFS_DOEPINTx), which indicates that the OUT endpoint has been completely disabled. When the EPDISD interrupt is raised, the module also clears the following bits to zero:
 - In USBFS_DOEPCTLx, EPDIS=0
 - In USBFS_DOEPCTLx, EPENA=0
5. The application must clear the global OUT NAK bit to start receiving data from other OUT endpoints that are not disabled.
 - In USBFS_DCTL, SGONAK=0

Universal asynchronous OUT data transfer

This section describes a conventional asynchronous OUT data

transfer (control, bulk or interrupt) Application requirements:

1. Before establishing an OUT transfer, the application must allocate a buffer in memory to hold all the data to be received as part of the OUT transfer.
2. For OUT transfers, the transfer size field in the transfer size register of the endpoint must be a multiple of the maximum packet size of the endpoint

Number (and aligned by word)

- Transfer size [EPNUM] = $n \times (\text{MPSIZ}[EPNUM] + 4 - (\text{MPSIZ}[EPNUM] \bmod 4))$
 - Packet Count [EPNUM] = n
 - $n > 0$
3. When an OUT endpoint interrupt occurs, the application must read the transfer size register of the endpoint to calculate the amount of valid data in memory. The amount of valid data received may be less than the programmed transfer size.
 - Effective amount of data in memory = initial transfer set by the application - remaining transfer after module update
 - Number of received USB packets = Initial number of packets set by the application - Remaining number of packets after module update

Internal data flow:

1. The application must set the transmit size and packet count fields in the endpoint-related registers, clear the NAK bit to zero, and enable the endpoint to receive data.
2. Once the NAK bit is cleared, the module starts receiving data and writing it to the receive FIFO (as long as there is space in the receive FIFO) For each packet received on the USB, the packet and its status are written to the receive FIFO, and each packet written to the receive FIFO (a packet with the maximum packet size or a short packet) ~~decrements~~ the packet count field at that endpoint by 1.
 - Received packets with invalid CRC are automatically cleared from the receive FIFO.
 - After replying an ACK for a packet on the USB, the module will discard any asynchronous OUT packets that the host has resent because it could not detect the ACK. The application will not detect multiple consecutive OUT packets on the same endpoint with the same data PID. In this case, the packet count does not decrement.
 - If there is no space in the receive FIFO, synchronous or asynchronous packets will be ignored and not written to the receive FIFO. in addition, the asynchronous OUT token will receive a NAK handshake answer.
 - In all three cases above, the packet count is not decremented because no data is written to the receiving FIFO.
3. When the packet count goes to 0 or a short packet is received on the endpoint, the NAK bit on that endpoint will be set to 1. With NAK position 1, synchronous or asynchronous packets will be ignored and will not be written to the receive FIFO, while the asynchronous OUT token will receive a NAK handshake answer.
4. After the data is written to the receive FIFO, the application will read the data from the receive FIFO and write it to external memory, one packet at a time, coming one endpoint

at a time.

5. After each packet is written to external memory on the AHB, the transfer size of the endpoint is automatically subtracted from the size of that packet.
 6. The OUT data transfer completion status of the OUT endpoint is written to the receive FIFO when
 - Transfer size is 0 and packet count is 0
 - The last OUT packet written to the receive FIFO is a short packet
- (Packet size:0~max packet size -1)
7. When the application pops up this status entry (OUT data transfer complete) and generates a transfer complete interrupt for that endpoint, it also clears

Endpoint enable bit.

Application programming sequence:

1. Program the USBFS_DOEPTSIZx register using the transfer size and the corresponding number of packets.
2. Use the endpoint feature to program the USBFS_DOEPCTLx register with EPENA and CNAK position 1.
 - In USBFS_DOEPCTLx, EPENA=1
 - In USBFS_DOEPCTLx, CNAK =1
3. Wait for the RXFNE interrupt (in USBFS_GINTSTS) and read the packet from the receive FIFO.
 - This step can be repeated several times, depending on the transfer size.
4. Triggers the XFRC interrupt (USBFS_DOEPINTx) to indicate the successful completion of an unsynchronized OUT data transfer.
5. Read the USBFS_DOEPTSIZx register to determine the amount of valid data.

Universal synchronization OUT data transfer

This section describes the regular synchronous OUT data transfer.

Application requirements:

1. All application requirements for non-synchronous OUT data transfer apply to synchronous OUT data transfer.
2. For the transmission size and packet count fields in synchronous OUT data transmissions, they must always be set to the number of packets of the maximum packet size that can be received in a single frame. The synchronous type of OUT data transmission transaction must be completed within a single frame.
3. Before the end of the periodic frame (EOPF interrupt in USBFS_GINTSTS), the application must read all synchronous OUT packets (data entries and status entries) from the receive FIFO.
4. To receive the data in the next frame, a synchronous OUT endpoint must be enabled before SOF(USBFS_GINTSTS) after EOPF(USBFS_GINTSTS).

Internal data flow:

1. The internal data flow of the synchronous OUT endpoint is essentially the same as that of the non-synchronous OUT endpoint, but with slight differences.
2. When the synchronous OUT endpoint is enabled by setting the endpoint enable position 1

and clearing the NAK bit to zero, the even/odd frame position 1 must be set accordingly. the module will receive data in a specific frame on the synchronous OUT endpoint only if the following conditions are met:

-EONUM (in USBFS_DOEPCTLx) = FNSOF [0] (in USBFS_DSTS)

3. When the application reads a synchronous OUT packet (data and status) from the receive FIFO in its entirety, the module updates the RXDPID field in USBFS_DOEPTSIz with the data PID of the last synchronous OUT packet read from the receive FIFO.

Application programming sequence:

1. Program the USBFS_DOEPTSIz register using the transfer size and the corresponding packet count

2. Use the endpoint feature to program the USBFS_DOEPCTLx register with the endpoint enable bit, clear NAK bit, and odd/even frame position 1.
 - EPENA=1
 - CNAK=1
 - EONUM = (0: even / 1: odd)
3. Wait for the RXFNE interrupt (in USBFS_GINTSTS) and read the packet from the receive FIFO.
 - This step can be repeated several times, depending on the transfer size.
4. The XFRC interrupt (in USBFS_DOEPINTx) indicates that the synchronous OUT data transfer is complete. This interrupt does not necessarily mean that the data in memory is valid.
5. For synchronous OUT transfers, the application may not always detect this interrupt. Instead, the application may detect the IISOOXFRM interrupt in USBFS_GINTSTS.
6. Read the USBFS_DOEPTSIZx register to determine the size of the received transmission and to determine the validity of the data received in the frame. The application must consider data received in memory as valid only if one of the following conditions is met:
 - RXDPID=D0 (in USBFS_DOEPTSIZx) and the number of USB packets receiving this valid data = 1
 - RXDPID=D1 (in USBFS_DOEPTSIZx) and the number of USB packets receiving this valid data = 2
 - RXDPID=D2 (in USBFS_DOEPTSIZx) and the number of USB packets receiving this valid data = 3 The number of USB packets receiving this valid data = the initial number of packets programmed by the application - the number of packets remaining after the module update.

Applications can discard invalid packets.

Incomplete synchronous OUT data transfer

This section describes the application programming sequence

in case of packet loss in synchronous OUT packets. Internal

data flow:

1. For synchronous OUT endpoints, an XFRC interrupt (in USBFS_DOEPINTx) may not always be raised. If the module drops synchronous OUT packets, the application may not detect an XFRC interrupt (in USBFS_DOEPINTx) if
 - The module will discard the received ISO OUT data if the receive FIFO cannot accommodate the full ISO OUT packet
 - Received synchronous OUT packet with CRC error

- Module received a corrupted synchronous OUT token
 - Applications are very slow to read data from the receiving FIFO
2. If the module detects the end of a periodic frame before the transmission of all synchronous OUT endpoints is complete, an Incomplete Synchronous OUT Data interrupt (IISO0XFRM in USBFS_GINTSTS)s triggered, indicating that the XFRC interrupt (in USBFS_DOEPINTx) is not triggered on at least one synchronous OUT endpoint. At this point, the endpoint that did not complete the transfer remains enabled, but there is no valid transfer in progress on that endpoint of the USB.

Application programming sequence:

1. Hardware triggered `IISO0XFRM` interrupt (`USBFS_GINTSTS`) indicates that at least one synchronous OUT endpoint in the current frame has an incomplete transmission.
2. If this happens because the synchronous OUT data is not fully read from the endpoint, the application must ensure that all synchronous OUT data (including data entries and status entries) is first read away from the receiving FIFO before continuing processing.
 - The application can detect the `XFRC` interrupt (`USBFS_DOEPINTx`) once all data has been read from the receive FIFO. In this case, the application must re-enable the endpoint to receive the synchronous OUT data in the next frame.
3. When the application receives an `IISO0XFRM` interrupt (in `USBFS_GINTSTS`), the application must read the control registers of all synchronous OUT endpoints (`USBFS_DOEPCTLx`) to determine which endpoints have incomplete transmissions in the current frame. An incomplete endpoint transmission is indicated when both of the following conditions are met:
 - `EONUM` bit (in `USBFS_DOEPCTLx`) = `FNSOF [0]` (in `USBFS_DSTS`)
 - `EPENA=1` (in `USBFS_DOEPCTLx`)
4. Before a SOF interrupt is detected (in `USBFS_GINTSTS`), the completion of the previous operation must be performed to ensure that the current frame number has not changed.
5. For synchronous OUT endpoints with incomplete transfers, the application must discard the data in memory and disable the endpoint by setting `EPDIS` position 1 in `USBFS_DOEPCTLx`.
6. Wait for the `EPDIS` interrupt (in `USBFS_DOEPINTx`) and enable the endpoint to receive new data in the next frame.
 - Since it may take some time for the module to disable the endpoint, the application may not be able to receive the data in the next frame after receiving invalid synchronization data.

Stop Asynchronous OUT Endpoint

This section describes how an application can stop a non-synchronous endpoint.

1. Places the module in global OUT NAK mode.
2. Endpoints required for prohibition
 - When disabling endpoints, set `STALL=1` (in `USBFS_DOEPCTL`) instead of setting `SNAK` position 1 in `USBFS_DOEPCTL`. The `STALL` bit always has higher priority than the `NAK` bit.
3. The `STALL` bit (in `USBFS_DOEPCTLx`) must be cleared to zero when the application no longer needs the endpoint to reply to the `STALL` handshake signal.

-
4. If the application receives a **SetFeature.Endpoint Halt** or **ClearFeature.Endpoint Halt** command from the host to set or clear the STALL state of an endpoint, it must set STALL position 1 or clear it to zero before the state phase on that control endpoint is transmitted.

31.6.7.2 IN Data Transfer

Packet Writing

This section describes how the application writes packets to the endpoint FIFO if the dedicated transmit FIFO is enabled.

1. The application can choose either polling mode or interrupt mode.
 - In polling mode, the application monitors the status of the endpoint transmit data FIFO by reading the USBFS_DTXFSTSx register to determine if there is enough space in the data FIFO.
 - In interrupt mode, the application waits for a TXFE interrupt (in USBFS_DIEPINTx) and then reads the USBFS_DTXFSTSx register to determine if there is enough space in the data FIFO.
 - To write a single non-zero length packet, there must be enough space in the data FIFO to accommodate the entire packet.
 - To write a zero-length packet, the application cannot look into the FIFO space.
2. If one of the above methods is used, when the application determines that there is enough space to write the transmit packet, the application must first perform the appropriate write operation to the endpoint control register before writing the data to the data FIFO. Typically, the application must perform a read-modify-write operation to the USBFS_DIEPCTLx register to avoid modifying the The application must normally perform a read-modify-write operation on the USBFS_DIEPCTLx register to avoid modifying the rest of the register while enabling endpoint position 1.

The application can write multiple packets from the same endpoint to the transmit FIFO if there is enough space; for periodic IN endpoints, the application can only write multiple packets within a frame at a time. The application writes all packets to be sent in the next frame only after the communication transaction for the previous frame has been transmitted.

Set IN endpoint NAK to 1

Internal data flow:

1. When the application sets IN NAK to 1 for a specific endpoint, the module stops data transmission on the endpoint, regardless of the availability of data in the endpoint's transmit FIFO.
2. The non-synchronous endpoint receives the IN token and replies with a NAK handshake answer.
 - Synchronous endpoint receives IN token and returns zero-length packet
3. The module triggers the INEPNE (IN endpoint NAK) interrupt in

USBFS_DIEPINTx in response to the SNAK bit in USBFS_DIEPCTLx.) interrupt in
USBFS_DIEPCTLx in response to the SNAK bit in USBFS_DIEPCTLx.

4. Once the application detects the interrupt, it assumes the endpoint is
inIN NAKmode. The application
can clear this interrupt by setting the CNAK in USBFS_DIEPCTLx to position 1.

Application programming sequence:

1. To stop sending any data on a specific IN endpoint, the application must set IN NAK to position 1. To set this position 1, the following fields must be programmed.
 - SNAK=1 in USBFS_DIEPCTLx

2. Wait for the INEPNE interrupt in USBFS_DIEPINTx to trigger. This interrupt indicates that the module has stopped sending data at the endpoint.
3. The module can send valid IN data on the endpoint while the application has NAK position 1 but the "NAK valid" interrupt is not yet triggered.
4. The application can temporarily mask this interrupt by writing the INEPNEM bit in DIEPMSK.
 - In DIEPMSK, INEPNEM = 0
5. To exit the endpoint NAK mode, the application must clear the NAK status bit (NAKSTS) in USBFS_DIEPCTLx to zero. This operation also clears the INEPNE interrupt (in USBFS_DIEPINTx)
 - In USBFS_DIEPCTLx, CNAK=1
6. If the application has masked the interrupt, it must be unmasked as follows:
 - In DIEPMSK, INEPNEM=1

Prohibit IN endpoints

Use the following sequence to disable a specific IN endpoint that has been previously enabled.

Application programming sequence:

1. The application must stop writing data to the AHB before the IN endpoint can be disabled.
2. The application must have the endpoint set to NAK mode.
 - SNAK=1 in USBFS_DIEPCTLx
3. Wait for the INEPNE interrupt in USBFS_DIEPINTx.
4. Place the following location 1 in the USBFS_DIEPCTLx register of the endpoint that must be disabled.
 - EPDIS=1 in USBFS_DIEPCTLx
 - SNAK=1 in USBFS_DIEPCTLx
5. The triggering of the EPDISD interrupt in USBFS_DIEPINTx indicates that the module has completely disabled the specified endpoint. While triggering the interrupt, the module also clears the following bits to zero:
 - In USBFS_DIEPCTLx, EPENA=0
 - In USBFS_DIEPCTLx, EPDIS=0
6. The application must read the USBFS_DIEPTSIZx register for the periodic IN EP to calculate how much data on the endpoint is sent on the USB.
7. The application must clear the data in the endpoint transmit FIFO by setting the following field in the USBFS_GRSTCTL register to 1:
 - TXFNUM (in USBFS_GRSTCTL) = endpoint transmit FIFO number

- TXFFLSH (in USBFS_GRSTCTL) = 1

The application must poll the USBFS_GRSTCTL register until the module clears the TXFFLSH bit to zero, which indicates the end of the FIFO clear operation. To send new data on this endpoint, the application can re-enable the endpoint at a later time.

Universal non-cyclic IN data transfer

Application requirements:

1. Before establishing an IN transmission, the application must ensure that each packet that makes up an IN transmission can fit in a single buffer.
2. For IN transfers, the Transfer Size field in the Endpoint Transfer Size register indicates the effective amount of data for this transfer, which consists of multiple maximum packet sizes and a single short packet. This short packet is sent at the end of the transmission.
 - To send multiple packets of the maximum packet size and add a short packet at the end of the transmission: Transmission Size[EPNUM] = $x \times \text{MPSIZ}[EPNUM] + sp$
If ($sp > 0$), packet count[EPNUM] = $x + 1$.
Otherwise, packet count[EPNUM] = x
 - To send a single zero-length packet:
transmission size [EPNUM] = 0
Packet Count [EPNUM] = 1
 - To send multiple packets of the maximum packet size and add a zero-length packet at the end of the transmission, the application must split the transmission into two parts.
The first part sends packets of the maximum packet size, and the second part sends only zero-length packets.
First transmission: transmission size [EPNUM] = $x \times \text{MPSIZ}[epnum]$; packet count = n; second transmission: transmission size [EPNUM] = 0; packet count = 1;
3. When an endpoint is enabled for data transfer, the module updates the transfer size register. At the end of an IN transfer, the application must read the transfer size register to determine how much of the data fed into the transmit FIFO has been sent out via USB.
4. Amount of data fed into the transmit FIFO = initial transfer size programmed by the application - final transfer size after module update
 - Amount of data already sent via USB = (initial packet count programmed by the application - final packet count after module update) $\times \text{MPSIZ}[EPNUM]$
 - Remaining amount of data to be sent via USB = (initial transfer size programmed by the application - amount of data already sent via USB)

Internal data flow:

-
1. The application must set the transmission size and packet count fields in the registers of a particular endpoint and enable that endpoint to send data.
 2. The application must also write the required data to the transmit FIFO of the endpoint.
 3. Each time the application writes a packet to the transmit FIFO, the packet size is automatically subtracted from the transfer size of that endpoint. **"packets** in FIFO" count is incremented as data is written to the FIFO (this is a 3-bit count that is maintained internally by the module, one for each IN endpoint transmit FIFO). The maximum number of packets maintained by the module in the IN endpoint FIFO

always eight) For zero-length packets, each FIFO has a separate flag and there is no data in the FIFO.

4. When data is written to the transmit FIFO, the module sends that data out when it receives an IN token. After each packet is sent and a reply ACK handshake signal is received, the packet count at that endpoint is decremented by 1 until the packet count becomes 0. The packet count is not decremented when a timeout occurs.
5. For zero-length packets (indicated by the internal zero-length flag), the module issues a zero-length packet for the IN token and decrements the value of the packet count field.
6. If there is no data in the FIFO corresponding to the endpoint where the IN token was received, and the packet count field for that endpoint is zero, the module generates an "IN token received when TxFIFO is empty" (ITTXFE) interrupt for that endpoint (provided the NAK bit for that endpoint is not set to 1) The module replies with a NAK handshake signal on this asynchronous endpoint.
7. The module will internally return the FIFO pointer to the beginning and will not generate a timeout interrupt.
8. When the transmission size is 0 and the packet count is 0, a transmission complete (XFRC) interrupt is generated for that endpoint, and the endpoint enable is cleared to zero.

Application programming sequence:

1. Program the USBFS_DIEPTSIZx register using the transfer size and the corresponding packet count.
2. Program the USBFS_DIEPCTLx register using the endpoint feature and set CNAK and EPENA (Endpoint Enable) to position 1.
3. When sending a non-zero length packet, the application must poll the USBFS_DTXFSTSx register (where x is the FIFO number associated with that endpoint) to determine if there is enough space in the data FIFO. The application can also select the TXFE bit (in USBFS_DIEPINTx) before writing the data.

Universal periodic IN data transfer

This section describes a typical periodic IN data transfer. Application requirements:

1. Application requirements 1, 2, 3, and 4 for generic non-periodic IN data transfers apply equally to periodic IN data transfers (with minor modifications to requirement 2)

-
- The application can only send a number of packets of the maximum packet size or a number of packets of the maximum packet size plus a short packet at the end of the transmission.
- To send multiple packets of the maximum packet size and add a short packet at the end of the transmission, the following condition must be satisfied: Transmission Size[EPNUM] = $x \times \text{MPSIZ}[EPNUM]$ + sp
(where x is an integer greater than 0 and sp ranges from 0 ~
MPSIZ[EPNUM]-1) If (sp > 0), packet count[EPNUM] = $x + 1$

Otherwise, packet count[EPNUM] = x;

MCNT[EPNUM] = packet count[EPNUM]

- The application cannot send a zero-length packet at the end of a transmission. Applications can send a zero-length packet on its own.

- To send a single zero-

length packet:

transmission size

[EPNUM]=0 packet count

[EPNUM]=1

MCNT[EPNUM]=Packet Count[EPNUM]

- 2. Applications can only schedule data transfers one frame at a time.

– $(MCNT - 1) \times MPSIZ < XFERSIZ \leq MCNT \times MPSIZ$

– PKTCNT = MCNT (in USBFS_DIEPTSIZx)

– If XFERSIZ < MCNT × MPSIZ, the last packet transmitted is a short packet

– Please note MCNT is located in USBFS_DIEPTSIZx, MPSIZ is located in USBFS_DIEPCTLx, PKTCNT is located in USBFS_DIEPTSIZx, XFERSIZ is located in USBFS_DIEPTSIZx

- 3. Before receiving an IN token, the application must write the complete data to be sent in the frame to the transmit FIFO. When the IN token is received, the module performs the operation when the FIFO is empty, even if only one double word of the data to be sent in the frame is not written in the transmit FIFO. When the transmit FIFO is empty:

- Zero-length packets will be replied on the sync endpoint
- The NAK handshake signal will be replied on the interrupt endpoint

Internal data flow:

1. The application must set the transmission size and packet count fields in the registers of a particular endpoint and enable that endpoint to send data.
2. The application must also write the required data to the transmit FIFO associated with the endpoint.
3. Each time the application writes a packet to the transmit FIFO, the packet size is automatically subtracted from the transfer size of that endpoint. The application continues to fetch data from memory to write to the transmit FIFO until the transfer size of the endpoint becomes zero.
4. When the periodic endpoint receives an IN token, the module will start sending the data in the FIFO (if there is data in the FIFO). If the FIFO does not contain a complete packet of data for the frame to be sent, the module will generate a "IN token received when

TxFIFO is empty^{**} interrupt for the endpoint.

- Zero-length packets will be replied on the sync endpoint
 - The NAK handshake signal will be replied on the interrupt endpoint
5. The endpoint's packet count decrements by 1 when
- For synchronous endpoints, when sending a zero-length or non-zero-length packet
 - For interrupt endpoints, decrement when sending the ACK handshake signal
 - When both the transmission size and packet count are 0, a transmission complete interrupt is generated for that endpoint and the endpoint enable bit is cleared to zero.

6. During the Periodic Frame Interval controlled by the PFIVL bit in USBFS_DCFG), the module generates an IISOIXFR interrupt in USBFS_GINTSTS when it finds any data in the synchronous IN endpoint FIFO that should be empty during the current frame that has not yet been sent.

Application programming sequence:

1. Use the endpoint feature to program the USBFS_DIEPCTLx register with CNAK and EPENA position 1.
2. Write the data that needs to be sent in the next frame to the transmit FIFO.
3. The hardware triggered ITTXFE interrupt (in USBFS_DIEPINTx) indicates that the application has not yet written all the data it needs to send to the transmit FIFO.
4. If the interrupt endpoint is enabled before the interrupt is detected, the interrupt will be ignored. If the interrupt endpoint is not enabled, this endpoint is enabled so that data can be sent out when the next IN token is received.
5. If no ITTXFE interrupt is generated in USBFS_DIEPINTx when the XFRC interrupt is triggered in hardware (in USBFS_DIEPINTx), it means that the synchronous IN transfer is successfully completed. If the USBFS_DIEPTSIZx register is always read with transfer size = 0 and packet count = 0, then all data has been sent via USB.
6. Setting the XFRC interrupt (in USBFS_DIEPINTx) indicates successful completion of the interrupt IN transfer regardless of whether the ITTXFE interrupt (in USBFS_DIEPINTx) is generated. If the USBFS_DIEPTSIZx register is always read with transfer size = 0 and packet count = 0, then all data has been sent via USB.
7. If any of the preceding interrupts are not generated when the Incomplete Synchronous IN Transmission (IISOIXFR) interrupt is set in USBFS_GINTSTS, it means that the module has not received at least 1 periodic IN token in the current frame.

Incomplete synchronization IN data transfer

This section describes the actions that an application must perform for an incomplete synchronous IN data transfer.

Internal data flow:

1. A synchronous IN transmission is considered incomplete when one of the following conditions is met:
 - a) The module is receiving a corrupted Sync IN token on at least one of the Sync IN endpoints. At this point, the application detects an incomplete synchronous IN transfer interrupt (IISOIXFR bit in USBFS_GINTSTS)

-
- b) The application is too slow to write data to the transmit FIFO and receives an IN token before the complete data is written to the FIFO. In this case, the application detects the "IN token received when TxFIFO is empty" interrupt in USBFS_DIEPINTx. The application can ignore this interrupt because eventually this will generate an incomplete synchronous IN transfer interrupt at the end of the periodic frame (ISOIXFR bit in USBFS_GINTSTS). The module will send a zero-length packet via USB in response to the received IN token.
 - 2. The application must stop writing data to the transmit FIFO as soon as possible.
 - 3. The application must set the NAK bit of the endpoint and disable position 1.

4. The module will disable the endpoint, clear the disable bit and trigger an "endpoint disable" interrupt for the endpoint.

Application programming sequence:

1. Applications can ignore the "IN token received when TxFIFO is empty" interrupt in USBFS_DIEPINTx on any sync IN endpoint, as this will eventually generate an incomplete sync IN transfer interrupt (in USBFS_GINTSTS)
2. A hardware-triggered Incomplete Sync IN Transfer interrupt (in USBFS_GINTSTS) indicates the presence of an incomplete Sync IN transfer on at least one Sync IN endpoint.
3. The application must read the Endpoint Control registers of all synchronous IN endpoints to detect the presence of endpoints that have not completed IN data transfer.
4. The application must stop writing data to the "periodic transmit FIFO" associated with these endpoints.
5. Program the following fields in the USBFS_DIEPCTLx register to disable endpoints:
 - SNAK=1 in USBFS_DIEPCTLx
 - EPDIS=1 in USBFS_DIEPCTLx
6. Hardware triggering of the "endpoint disable" interrupt in USBFS_DIEPINTx indicates that the module has disabled the endpoint.
 - At this point, the application must either clear the data in the associated transmit FIFO or overwrite the existing data in the FIFO by enabling the newly transmitted endpoint in the next frame. To refresh the data, the application must use the USBFS_GRSTCTL register.

Stop Asynchronous IN Endpoint

This section describes how an application can stop a non-synchronous endpoint. Application programming sequence:

1. Disable the IN endpoint to be stopped. Also set STALL position 1.
2. EPDIS=1 in USBFS_DIEPCTLx (when endpoint is enabled)
 - STALL=1 in USBFS_DIEPCTLx
 - The STALL bit always has a higher priority than the NAK bit
3. The hardware-triggered "endpoint disable" interrupt (in USBFS_DIEPINTx) lets the application know that the module has disabled the specified endpoint.
4. Depending on the endpoint type, the application must clear the non-periodic or periodic transmit FIFO. for non-periodic endpoints, the application must re-enable another non-periodic

endpoint that does not need to be stopped to send data.

5. The STALL bit of USBFS_DIEPCTLx

must be cleared to zero when the application is ready to end the STALL handshake signal for that endpoint.

6. If the application receives a **SetFeature.Endpoint Halt**

command or **ClearFeature.Endpoint Halt** command from the host to set or clear the STALL state of the endpoint, the

Set STALL to position 1 or clear before the status phase of the control endpoint is transmitted. Example: Stopping control of OUT endpoints

In this case, the application must enable the ITTXFE interrupt of **USBFS_DIEPINTx** and the OTEPDIS interrupt of **USBFS_DOEPINTx** during the data phase of the control transfer (after the module has finished transferring the amount of data specified in the SETUP packet). OTEPDIS interrupt of **USBFS_DOEPINTx** (after the module has finished transmitting the amount of data specified in the SETUP packet) Subsequently, when the application receives this interrupt, it must set STALL in the corresponding endpoint control register to position 1 and clear this interrupt.

31.7 Register Description

The application controls the USBFS module by reading and writing to the control and status registers through the AHB slave interface. all registers in the USBFS module are 32-bit registers and their addresses are aligned to 32 bits, so they can only be accessed in a 32-bit manner.

The control and status registers are divided into the following categories:

- USBFS System Control Register
- Module Global Register
- Host Mode Register
- Device Mode Register
- Power and clock gating control registers

The USBFS System Control register is different from the other registers in that it is independent of the USBFS module and controls the settings associated with the USBFS module.

Only the module global registers, power and clock gating control registers can be accessed in host and device modes. When the USBFS module is in one mode (host or device), the application must not access the registers in another role mode, such as accessing the device mode registers when in host mode. If an illegal access occurs, a mode mismatch interrupt will be generated and reflected in the USBFS_GINTSTS.NMIS bit of the module interrupt register. When the module switches from one role mode to another, the registers in the new operating mode must be reprogrammed to the state after power-on reset.

Refer to Table 31-3~Table 31-4 for the USBFS module register list and base address. USBFS system control register base address: 0x40055400

Table 31-3 USBFS System Control Register List

(USBFS system control register) Register name	Offset Address	Reset value
USBFS system control hosting (USBFS_SYCTLREG)	0x00	0x0000 0000

USBFS module register base address: 0x400C0000

Table 31-4 USBFS System Control Register List

(USBFS global register) Register name	Offset Address	Reset value
USBFS VBUS control register (USBFS_GVBUSCFG)	0x00	0x0000 0000
USBFS AHB control register (USBFS_GAHBCFG)	0x08	0x0000 0000
USBFS USB configuration register (USBFS_GUSBCFG)	0x0c	0x0000 0A00
USBFS reset register (USBFS_GRSTCTL)	0x10	0x8000 0000
USBFS Global Interrupt Register (USBFS_GINTSTS)	0x14	0x1400 0020
USBFS Global Interrupt Mask Register (USBFS_GINTMSK)	0x18	0x0000 0000
USBFS Receive Status Debug Read Register (USBFS_GRXSTSR)	0x1c	0x0000 0000
USBFS Receive Status Read and Out Stack Register (USBFS_GRXSTSP)	0x20	0x0000 0000
USBFS Receive FIFO Size Register (USBFS_GRXFSIZ)	0x24	0x0000 0140
USBFS host non-periodic send FIFO size register (USBFS_HNPTXFSIZ)/endpoint 0 send FIFO size register (USBFS_DIEPRXF0)	0x28	0x0200 0140
USBFS Non-periodic Send FIFO/queue status register (USBFS_HNPTXSTS)	0x2c	0x0008 0100
USBFS module ID register (USBFS_CID)	0x3c	0x1234 5678
USBFS Host Periodic Send FIFO Size Register (USBFS_HPTXFSIZ)	0x100	0x0000 0000
USBFS Device IN Endpoint x Send FIFO Size Register (USBFS_DIEPTXFx)	0x100+x*4(x=1~5)	0x0100 0240+(x-1)*0x100

(USBFS host control and status registers)		Offset Address	Reset value
Register name			
USBFS Host Configuration Register (USBFS_HCFG)	0x400	0x0000 0000	
USBFS Host Frame Interval Register (USBFS_HFIR)	0x404	0x0000 EA60	
USBFS host frame number/frame remaining interval register (USBFS_HFNUM)	0x408	0x0000 3FFF	
USBFS Host Periodic Send FIFO/Queue Status Register (USBFS_HPTXSTS)	0x410	0x0008 0100	
USBFS Host All Channel Interrupt Register (USBFS_HAINT)	0x414	0x0000 0000	
USBFS Host All Channel Interrupt Mask Register (USBFS_HAINTMSK)	0x418	0x0000 0000	
USBFS Host Port Control and Status Register (USBFS_HPRT)	0x440	0x0000 0000	
USBFS Host Channel x Feature Register (USBFS_HCCHARx)	0x500+x*0x20(x=0~11)	0x0000 0000	
USBFS host channel x interrupt register (USBFS_HCINTx)	0x508+x*0x20(x=0~11)	0x0000 0000	
USBFS host channel x interrupt mask register (USBFS_HCINTx)	0x50c+x*0x20(x=0~11)	0x0000 0000	
USBFS Host Channel x Transfer Size Register (USBFS_HCTSIZx)	0x510+x*0x20(x=0~11)	0x0000 0000	
USBFS host channel x DMA address register (USBFS_HCDMAx)	0x514+x*0x20(x=0~11)	0xFFFF XXXX	

(USBFS device control and status registers)	Offset Address	Reset value
Register name		
USBFS Device Configuration Register (USBFS_DCFG)	0x800	0x0820 0000
USBFS Device Control Register (USBFS_DCTL)	0x804	0x0000 0000
USBFS Device Status Register (USBFS_DSTS)	0x808	0x0000 0002
USBFS Device IN Endpoint Universal Interrupt Mask Hosting (USBFS_DIEPMSK)	0x810	0x1400 0000
USBFS Device OUT Endpoint General Interrupt Mask Register (USBFS_DOEPMSK)	0x814	0x0000 0000
USBFS Device All Endpoint Interrupt Register (USBFS_DAINT)	0x818	0x0000 0000
USBFS Device All Endpoint Interrupt Mask Hosting (USBFS_DAINTMSK)	0x81c	0x0000 0000
USBFS Device IN Endpoint FIFO Air Break Mask Register (USBFS_DIEPEMPMSK)	0x834	0x0000 0000
USBFS Device IN Endpoint 0 Control Register (USBFS_DIEPCTL0)	0x900	0x0000 8000
USBFS Device IN Endpoint x Control Register (USBFS_DIEPCTLx)	0x900+x*0x20(n=1~5)	0x0000 0000
USBFS Device IN endpoint x interrupt register (USBFS_DIEPINTx)	0x908+x*0x20(n=0~5)	0x0000 0000
USBFS Device IN Endpoint x Transfer Size Register (USBFS_DIEPSIZx)	0x910+x*0x20(n=0~5)	0x0000 0000
USBFS Device IN endpoint x DMA address hosting (USBFS_DIEPDMAx)	0x914+x*0x20(n=0~5)	0x0000 0000
USBFS Device IN Endpoint x Send FIFO Status Register (USBFS_DTXFSTSx)	0x918+x*0x20(n=0~5)	0x0000 0000
USBFS Device OUT Endpoint 0 Control Register (USBFS_DOEPCTL0)	0xb00	0x0000 8000
USBFS Device OUT Endpoint x Control Register (USBFS_DOEPCTLx)	0xb00+x*0x20(n=1~5)	0x0000 0000
USBFS Device OUT endpoint x interrupt register (USBFS_DOEPINTx)	0xb08+x*0x20(n=0~5)	0x0000 0000
USBFS Device OUT Endpoint x Transfer Size Hosting (USBFS_DOEPSIZx)	0xb10+x*0x20(n=0~5)	0x0000 0000
USBFS Device OUT Endpoint x DMA Address Register (USBFS_DOEPDMAx)	0xb14+x*0x20(n=0~5)	0xXXXXXXXX

(USBFS power and clock dead control register)	Offset Address	Reset value
register name		
USBFS power and gated clock control register (USBFS_PCGCCTL)	0xe00	0x0000 0000

31.7.1 USBFS System Control Register

31.7.1.1 USBFS system control register (USBFS_SYCTLREG)

USBFS System Control

Register Offset Address: 0x00

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															

position	Marker	Place Name	Function	Reading and writing
b31~b2	Reserved	-	The reset value must be maintained.	R
b1	SOFEN	SOF pulse output enable bit	When the host sends SOF or the device successfully receives SOF, a SOF pulse of 16 system clock cycles width is enabled from the PAD output 0: SOF pulse is not output 1: SOF pulse output Note: Accessible in both device mode and host mode.	R/W
b0	DFB	VBUS/ID pin internal de-jitter filter bypass enable bit	VBUS/ID pin module internal de-jitter filter bypass enable bit 0: module internal de-jitter filter active 1: Bypass module internal de-jitter filter Note: Accessible in both device mode and host mode.	R/W

31.7.2 USBFS Global Register

These registers are available in both host mode and device mode and do not need to be reprogrammed when switching between the two modes. The bit values in the register descriptions are expressed in binary unless otherwise noted.

31.7.2.1 USBFS VBUS control register (USBFS_GVBUSCFG)

VBUS Configuration Register Offset

Address: 0x00

Reset value: 0x0000 0000

This register can be used to set the VBUS value and thus ignore the status of the VBUS pin.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
Reserved																
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
Reserved								VUB SVA L	VBU SOV EN	Reserved						

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	The reset value must be maintained.	R
b7	VBUSVAL	VBUS value	VBUS Value (VBUS Value) Used to set the VBUS value of USBFS, when set to 1 and VBUSOVEN is set to 1 to finish powering up the USBFS. Note: Accessible in device mode only.	R/W
b6	VBUSOVEN	VBUS Override Enable	VBUS Override Enable (VBUS Override Enable) Used to reflect the value set by VBUSVAL to the status of USBFS CORE. The value of VBUSVAL is valid only when this bit is set to 1. Note: Accessible in device mode only.	R/W
b5~b0	Reserved	-	The reset value must be maintained.	R

31.7.2.2 USBFS AHB control register (USBFS_GAHBCFG)

AHB Configuration Register

Offset Address: 0x08

Reset value: 0x0000 0000

This register can be used to configure the module after power-up or when changing the role mode.

This register contains mainly AHB system related configuration parameters.

The application must program this register before starting any AHB or USB transactions. Do not change this register after initial programming.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
PTX	TXF	Res	DMA	HBSTLEN[3:0]				FEL	ELV	erv	EN	GIN			
VL	L	ed		TMS								K			

position	Marker	Place Name	Function	Reading and writing
b31~b9	Reserved	-	The reset value must be maintained.	R
b8	PTXFELVL	Periodic TxFIFO Empty Threshold	Periodic TxFIFO empty level indicates when to trigger the periodic TxFIFO air break bit in the module interrupt register (PTXFE bit in USBFS_GINTSTS) 0: PTXFE (located at USBFS_GINTSTS) interrupt indicates that the periodic TxFIFO is half empty 1: PTXFE (located at USBFS_GINTSTS) interrupt indicates that the periodic TxFIFO is fully empty Note: Accessible in host mode only.	R/W
b7	TXFELVL	Device TxFIFO Empty Threshold	Device TxFIFO empty level In device mode, this bit indicates when to trigger the IN endpoint to send a FIFO air break (TXFE in USBFS_DIEPINTx)0: TXFE (in USBFS_DIEPINTx) interrupt indicates that the IN endpoint TxFIFO is half empty 1: TXFE (located at USBFS_DIEPINTx) interrupt indicates that the IN endpoint TxFIFO is fully empty Note: Accessible in host mode only.	R/W
b6	Reserved	-	The reset value must be maintained.	R
b5	DMAEN	DMA Enable	DMA enable 0: module runs in slave mode 1: Module runs in DMA mode Note: Accessible in both device mode and host mode.	R/W

b4~b1	HBSTLEN	Batch length/type	Burst length/type 0000b:Single 0001b: INCR 0011b:INCR4 0101b:INCR8 0111b:INCR16 Other values: Reserved Note: Accessible in both device mode and host mode.	R
-------	---------	-------------------	--	---

b0	GINTMSK	Global interrupt masking	Global interrupt mask This bit is used to mask global interrupts or to unmask global interrupts. The interrupt status register is updated by the module, independent of the setting of this bit. 0: Mask application-triggered interrupts 1: Unblock application-triggered interrupts Note: Accessible in both device mode and host mode.	R/W
----	---------	--------------------------	--	-----

31.7.2.3 USBFS USB Configuration Register (USBFS_GUSBCFG)

USBFS USB configuration register offset

address: 0x00C

Reset value: 0x0000 0A00

This register can be used to configure the module after power-up or changing the role mode. It contains configuration parameters related to USB and USB-PHY.

The application must program this register before starting any AHB or USB transactions. Do not change this register after initial programming.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Res erv ed	FDM OD	FHM OD													Reserved
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				TRDT[3:0]			Reserved		PHY SEL		Reserved				TOCAL[2:0]

position	Marker	Place Name	Function	Reading and writing
b31	Reserved	-	The reset value must be maintained.	R
b30	FDMOD	Forced device mode	Force device mode Writing a 1 to this bit forces the module into device mode, ignoring the state of the USBFS_ID input pins. 0: Normal mode, depending on the input state of the USBFS_ID pin 1: Forced device mode After forcing position 1, the application must wait at least 25 ms for the change to take effect. Note: Accessible in both device mode and host mode.	R/W
b29	FHMOD	Forced host mode	Force host mode Writing a 1 to this bit forces the module into host mode, ignoring the state of the USBFS_ID input pin. 0: Normal mode, depending on the input state of the USBFS_ID pin 1: Forced host mode After forcing position 1, the application must wait at least 25 ms for the change to take effect. Note: Accessible in both device mode and host mode.	R/W
b28~b14	Reserved	-	The reset value must be maintained.	R

b13~b10	TRDT	USB turnaround time	USB turnaround time sets the turnaround time in PHY clock counts. To calculate the value of TRDT, use the following formula: $TRDT = 4 \times AHB\ clock + 1\ PHY\ clock$ For example: 1. If the AHB clock frequency = 84 MHz (PHY clock frequency = 48 MHz), TRDT is set to 9. 2. If the AHB clock frequency = 48 MHz (PHY clock frequency = 48 MHz), TRDT is set to 5. Note: Accessible in both device mode and host mode.	R/W
---------	------	---------------------	---	-----

b9~b7	Reserved	-	The reset value must be maintained.	R
b6	PHYSEL	Full Speed Series Transceiver Selection	Full Speed serial transceiver select This bit is a write-only bit and is always 1.	W
b5~b3	Reserved	-	The reset value must be maintained.	R
b2~b0	TOCAL	FS Timeout calibration	FS timeout calibration The additional delay introduced by the PHY includes the number of PHY clocks set by the application in this field, as well as the module's full-speed inter-packet timeout interval. The impact of the delay introduced by different PHYs on the data line state is different. The standard USB timeout value for full-speed operation is 16 to 18 (inclusive) bits of time. Applications must program this field based on the enumeration speed. The number of bits of time added per PHY clock is 0.25 bits of time. Note: Accessible in both device mode and host mode.	R/W

31.7.2.4 USBFS reset register (USBFS_GRSTCTL)

USBFS reset register

offset address: 0x10

Reset value: 0x8000 0000

The application resets each hardware feature in the module through this register.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
AHB IDL	DMA REQ	Reserved													
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved					TXFNUM[4:0]					TXF FLS H	RXF FLS H	Res erv ed	FCR ST	HSR ST	CSR ST

position	Marker	Place Name	Function	Reading and writing
b31	AHBIDL	AHB master device idle	AHB master idle indicates that the AHB master device state machine is idle. Note: Accessible in both device mode and host mode.	R
b30	DMAREQ	AHB master device idle	DMA request signal (DMA request signal) This bit indicates that a DMA request is in progress. Used for debugging. Note: Accessible in both device mode and host mode.	R
b29~b11	Reserved	-	Read "/" , write "0" when writing	R
b10~b6	TXFNUM	TxFIFO number	TxFIFO number (TxFIFO number) FIFO number of the FIFO refresh using the TxFIFO refresh bit. This field can be changed only after the module has cleared the TxFIFO refresh bit to zero. <ul style="list-style-type: none"> ● 00000: – Host mode refreshes acyclic TxFIFO – Refresh Tx FIFO in device mode 0 ● 00001: – Periodic TxFIFO refresh in host mode – Refresh TXFIFO in device mode 1 ● 00010: Device mode flush TXFIFO 2 --- ● 00101: Refresh TXFIFO in device mode 15 ● 10000: Refresh all transmit FIFOs in device mode or host mode Note: Accessible in both device and host modes.	R/W

b5	TXFFLSH	TxFIFO Refresh	TxFIFO flush This bit selectively flushes one or all of the transmit FIFOs, but cannot be performed while the module is processing a communication transaction. The application can perform a write operation to this bit only after confirming that the module is not currently performing a read or write operation to the TxFIFO. Use the following registers for confirmation: – Read: A valid NAK interrupt ensures that the module is not currently performing a read operation on the FIFO – Write: The AHBIDL bit in USBFS_GRSTCTL ensures that the module is not currently performing any write operations to the FIFO Note: Accessible in both device mode and host mode.	R/W
b4	RXFFLSH	RxFIFO Refresh	RxFIFO flush The application can use this bit to flush the entire RxFIFO, but must first ensure that the module is not currently handling communication. Only after confirming that the module is not currently performing a read or write operation to the RxFIFO can the application perform a write operation to this bit. The application must wait until this bit is cleared before performing other operations. Usually it is necessary to wait for 8 clock cycles (whichever is the slowest of the PHY or AHB clocks) Note: Accessible in both device mode and host mode.	R/W
b3	Reserved	–	The reset value must be maintained.	R
b2	FCRST	Host frame counter reset	Host frame counter reset The frame number counter in the module is reset when the application performs a write operation to this bit. After the frame counter is reset, the frame number of the next SOF sent by the module is 0. Note: Accessible in both device mode and host mode.	R/W
b1	HSRST	HCLK domain logic soft reset	HCLK domain logic soft reset (HCLK soft reset) This bit is used by the application to refresh the control logic in the AHB clock domain. Only the AHB clock domain pipeline is reset. The FIFO is not refreshed by this bit. After terminating a transaction on the AHB in compliance with the protocol, all state machines in the AHB clock domain are reset to the idle state. The CSR control bits used by the AHB clock domain state machines are cleared to zero. To clear this interrupt, clear the status mask bit generated by the AHB clock domain state machine and used to control the interrupt state. Since the interrupt status bit is not cleared to zero, the application can get all module events that occur after this position 1 Status. This bit is self-clearing and the module will clear this bit after all necessary logic in it has been reset. This process requires several clocks of time, depending on the current state of the module. Note: Accessible in both device mode and host mode.	R/W

b0	CSRST	Module soft reset	<p>The module soft reset (Core soft reset) resets the HCLK and PCLK domains as described below:</p> <p>Clear each interrupt and all CSR register bits to zero, except for the following:</p> <ul style="list-style-type: none"> – RSTPDMODL bit in USBFS_PCGCCTL – GAYEHCLK bit in USBFS_PCGCCTL – PWRCLMP bit in USBFS_PCGCCTL – STPPCLK bit in USBFS_PCGCCTL – FSLSPCS bit in USBFS_HCFG – DSPD bits in USBFS_DCFG <p>Reset all module state machines (except AHB slave devices) to the idle state and clear all transmit FIFOs and receive FIFOs.</p> <p>Terminate all transactions on the AHB master device as soon as possible after the final data phase of the AHB transfer is complete. Terminate all transactions on the USB immediately.</p> <p>The application can perform a write operation to this bit at any time when it needs to reset the module. This bit is self-clearing and the module will clear this bit after all necessary logic in it has been reset, a process that takes a number of clocks depending on the current state of the module. Once this bit is cleared, the software must wait at least 3 PHY clocks before it can access the PHY domain (synchronization delay). In addition, the software</p> <p>It must also be determined that bit 31 in this register is set to 1 (AHB master device is idle) before operation can begin.</p> <p>Software reset is typically used in two situations, either during software development or after the user dynamically changes the PHY selection bits in the USB configuration registers listed above. When the user changes the PHY, the appropriate clock will be selected for the PHY and used in the PHY domain. Once the new clock is selected, the PHY domain must be reset to ensure proper operation.</p> <p>Note: Accessible in both device mode and host mode.</p>	R/W
----	-------	-------------------	---	-----

31.7.2.5 USBFS Global Interrupt Status Register (USBFS_GINTSTS)

USBFS interrupt status register

offset address: 0x14

Reset value: 0x140000020

This register is used to interrupt the application in the current mode (device mode or host mode) with the help of system-level events.

Some bits in this register are only valid in host mode, while others are only valid in device mode. In addition, this register indicates the current mode.

FIFO status interrupts are read-only; the FIFO interrupt flag is automatically cleared if the software performs a read or write operation on the FIFO during the processing of these interrupts.

Before enabling the interrupt bit, the application must clear the USBFS_GINTSTS register to zero during initialization to avoid generating any interrupts before initialization.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
WKU INT	VBU SVI NT	DIS CIN T	CID SCH G	Res erv ed	PTX FE	HCI NT	HPR TIN T	Res erv ed	DAT AFS USP	IPX F R / INC OMP ISO UT	IIS OIX FR	OEP INT	IEP INT	Res erv ed	Rese rved
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EOPF	ISO0 DRP	ENUM DNE	USBR ST	USBS USP	ESU SP	Res erv ed	Res erv ed	GON AKE FF	GIN AKE FF	NPT XFE	RXF NE	SOF	Res erv ed	MMI S	CMO D

position	Marker	Place Name	Function	Reading and writing
b31	WKUINT	Recovery/remote wake-up interrupt detected	Resume/remote wakeup detected interrupt In device mode, this interrupt will be triggered when a recovery signal is detected on the USB bus. In host mode, this interrupt will be triggered when a remote wakeup is detected on the USB. Clear this bit by writing 1 to it via software. Note: Accessible in both device mode and host mode.	R/W
b30	VBUSVINT	VBUS valid interrupt	VBUSy valid interrupt This interrupt is triggered in device mode when the USBFS_VBUS pin is detected to change from low to high. The bit is cleared by writing 1 to it via software.	R/W

Note: Accessible in device mode only.

b29	DISCINT	Disconnection break detected	Disconnect detected interrupt Triggers this interrupt when a device disconnection is detected. Clear this bit by writing 1 to it via software. Note: Accessible in host mode only.	R/W
b28	CIDSCHG	Connector ID line status change interrupt	Connector ID line status change (Connector ID status change) When the connector ID line status is changed, the module will this position 1. Clear this bit by writing 1 to it via software. Note: Accessible in both device mode and host mode.	R/W
b27	Reserved	-	The reset value must be maintained.	R
b26	PTXFE	Periodic TxFIFO Air Breaks	Periodic TxFIFO empty interrupt is triggered when the periodic transmit FIFO is half empty or completely empty and there is space in the periodic request queue for at least one entry to be written. Whether the FIFO is half empty or full empty is determined by the periodic TxFIFO empty level bit in the USBFS_GAHBCFG register. (PTXFELVL bit in USBFS_GAHBCFG) is determined. Note: Accessible in host mode only.	R
b25	HCINT	Host channel interruption	Host channels interrupt When the module sets this position to 1, it indicates that a pending interrupt exists on one of the channels in the module (in host mode). The application must read the host USBFS_HAINT register to determine the exact number of the channel on which the interrupt occurred, and then read the corresponding USBFS_HCINTx register to determine the exact cause of the triggered interrupt. The application must first clear the corresponding status bit of the USBFS_HCINTx register before it can clear that bit. Note: Accessible in host mode only.	R
b24	HPRTINT	Host port interruption	Host port interrupt When the module sets this position 1, it indicates a change in the state of the USBFS controller port in host mode. The application must read the USBFS_HPRT register to determine the exact event that triggered this interrupt. The application must first clear the corresponding status bit of the USBFS_HPRT register before it can clear this bit to zero. Note: Accessible in host mode only.	R
b23	Reserved	-	The reset value must be maintained.	R

b22	DATAFSUSP	Data acquisition hangs	<p>Data fetch suspended</p> <p>This interrupt is valid only in DMA mode. This interrupt indicates that the module has stopped fetching data for the IN endpoint because TxFIFO space or request queue space is not available. The application uses this interrupt in the endpoint mismatch algorithm. For example, after an endpoint mismatch is detected, the application performs the following actions:</p> <ul style="list-style-type: none">— Set the global non-periodic IN NAK handshake signal to 1— Prohibit IN endpoints— Emptying the FIFO— Determine the token sequence based on the IN token sequence learning queue— Re-enabling endpoints— If the global non-periodic IN N A K is cleared but the module has not yet fetched data for the IN endpoint and an IN token has been received, the global non-periodic IN NAK handshake signal is cleared: the module generates an "IN token received when FIFO is empty" interrupt. The USBFS then sends the NAK response to the host. To avoid this, the application can check the DATAFSUSP interrupt in USBFS_GINTSTS, which ensures that the global NAK handshake signal is cleared after the FIFO is full. Alternatively, the application can mask the "IN token received when FIFO is empty interrupt" when the global IN N A K handshake signal is cleared. <p>Clear this bit by writing 1 to it via software.</p> <p>Note: Accessible in device mode only.</p>	R/W
-----	-----------	------------------------	--	-----

b21	IPXFR	Incomplete cyclic / transmission /	IPXFR:Incomplete periodic transfer In host mode, if there are outstanding periodic transactions that are still pending and they are scheduled to complete during the current frame, the module will place this interrupt in position 1.	R/W
	INCOMPISO	Incomplete OUT OUT synchronous transmission	INCOMPISOOUT: Incomplete isochronous OUT transfer In device mode, the module sets this interrupt to indicate that there is an incomplete transmission on at least one synchronous OUT endpoint in the current frame. This interrupt is triggered with the periodic end-of-frame interrupt (EOPF) bit in this register. Clear this bit by writing 1 to it via software. Note: Accessible in host mode only.	
b20	IISOIXFR	Incomplete IN sync transfer	The Incomplete isochronous IN transfer module sets this interrupt to 1 to indicate that there is an incomplete transfer on at least one synchronous IN endpoint in the current frame. This interrupt is triggered with the periodic end-of-frame interrupt (EOPF) bit in this register. This bit is cleared by writing 1 to it in software. Note: Accessible in device mode only.	R/W
b19	OEPINT	OUT endpoint interrupt	OUT endpoint interrupt The module will indicate the presence of a pending interrupt (in device mode) on one of the OUT endpoints in the module when this position is 1. The application must read the host USBFS_DAIINT register to determine the exact number of the OUT endpoint where the interrupt occurred, and then read the corresponding USBFS_DOEPINTx register to determine the exact cause of the triggered interrupt. The application must first clear the corresponding status bit of the corresponding USBFS_DOEPINTx register before it can clear that bit to zero. Note: Accessible in device mode only.	R
b18	IEPINT	IN endpoint interrupt	IN endpoint interrupt The module will indicate the presence of a pending interrupt (in device mode) on one of the IN endpoints in the module when this position is 1. The application must read the host USBFS_DAIINT register to determine the exact number of the IN endpoint where the interrupt occurred, and then read the corresponding USBFS_DIEPINTx register to determine the exact cause of the triggered interrupt. The application must first set the corresponding USBFS_DIEPINTx register to one. The corresponding status bit of the USBFS_DIEPINTx register is cleared to zero before the bit is cleared to zero. Note: Accessible in device mode only.	R
b17~b16	Reserved	-	The reset value must be maintained.	R
b15	EOPF	Periodic end-of-frame interruptions	End of periodic frame interrupt indicates that the current frame has reached the periodic frame interval field in the USBFS_DCFG register (PFIVL bit in USBFS_DCFG) The period specified by the Clear this bit by writing 1 to it in software. Note: Accessible in device mode only.	R/W

b14	ISOODRP	Dropping synchronous OUT packet interrupts	Isynchronous OUT packet dropped interrupt If the module cannot write synchronous OUT packets to the RxFIFO due to insufficient space in the RxFIFO to accommodate the maximum packets for the synchronous OUT endpoint, the module will set the location 1. Clear this bit by writing 1 to it via software. Note: Accessible in device mode only.	R/W
b13	ENUMDNE	Enumeration completion interrupt	Enumeration done interrupt The module indicates that speed enumeration is complete when it sets this position to 1. The application must read the USBFS_DSTS register to get the enumeration speed. Clear this bit by writing 1 to it via software. Note: Accessible in device mode only.	R/W
b12	USBRST	USB reset interrupt	USB reset interrupt The module indicates that a reset signal is detected on the USB when this bit is set to 1. Clear this bit by writing 1 to it via software. Note: Accessible in device mode only.	R/W
b11	USBSUSP	USB hang interrupt	USB suspend interrupt The module indicates that a pending state is detected on the USB when this position 1 is set. When the idle state on the USB bus is held for 3ms, the module will enter the pending state. This bit is cleared by writing a 1 to it via software. Note: Accessible in device mode only.	R/W
b10	ESUSP	Early hang-up interruption	Early suspend interrupt The module will indicate that the USB has been detected to be idle for 3ms when this position is 1. Note: Only accessible in device mode.	R/W
b9~b8	Reserved	-	The reset value must be maintained.	R
b7	GONAKEFF	Global OUT NAK valid interrupt	Global OUT NAK effective interrupt Indicates the "Set global OUT NAK to 1" bit in the USBFS_DCTL register set by the application (SGONAK bit in USBFS_DCTL) is already in effect in the module. This bit can be cleared by writing the "Clear Global OUT NAK to zero" bit in the USBFS_DCTL register (CGONAK bit in USBFS_DCTL). Note: Accessible in device mode only.	R
b6	GINAKEFF	Global non-cyclic IN NAK valid interrupt	Global nonperiodic NAK effective interrupt Indicates that the "Set global non-periodic IN NAK to 1" bit in the USBFS_DCTL register (SGINAK bit in USBFS_DCTL), set by the application, is in effect in the module. In other words, the module has sampled the global IN NAK bit set by the application and the result has been validated. By clearing the "Clear global non-periodic IN NAK to zero" bit in the USBFS_DCTL register (CGINAK bit in UBSFS_DCTL), you can clear this bit to zero. This interrupt does not necessarily indicate that a NAK handshake signal has been sent on the USB. the STALL bit has higher priority than the NAK bit. Note: Accessible in device mode only.	R

b5	NPTXFE	Non-periodic Tx FIFO air break	Non-periodic Tx FIFO empty interrupt This interrupt is triggered when the non-periodic Tx FIFO is half or full empty and there is space in the non-periodic send request queue for at least one entry to be written. Whether the FIFO is half empty or full empty is determined by the non-periodic Tx FIFO empty level bit in the USBFS_GAHBCFG register (TXFELVL bit in USBFS_GAHBCFG) is determined. Note: Accessible in host mode only.	R
b4	RXFNE	Rx FIFO non-air break	Rx FIFO non-empty interrupt indicates that there is at least one packet in the Rx FIFO waiting to be read. Note: Accessible in both host mode and device mode.	R
b3	SOF	Frame start interrupt	Start of frame interrupt In host mode, the module indicates that a SOF (FS) or Keep-Alive (LS) signal has been sent on the USB when this position is set to 1. The application must set this to 1 to clear the interrupt. In device mode, the module indicates that a SOF token has been received on the USB when it sets this position to 1. The application can obtain the current frame number by reading the device status register. This interrupt is only present when the module is running in FS mode. Clear this bit by writing 1 to it via software. Note: Accessible in both host mode and device mode.	R/W
b2	Reserved	-	The reset value must be maintained.	R

b1	MMIS	Pattern mismatch interrupt	<p>Mode mismatch interrupt (Mode mismatch interrupt) The module places the location 1 when the application tries to access the following registers:</p> <ul style="list-style-type: none"> — Module runs in device mode to access host mode registers — Module runs in host mode to access device mode registers <p>The register access ends with an OKAY response on the AHB, but the access is internally ignored by the module and does not affect module operation.</p> <p>Clear this bit by writing 1 to it via software.</p> <p>Note: Accessible in both host mode and device mode.</p>	R/W
b0	CMOD	Current working mode	<p>Current mode of operation indicates the current mode.</p> <p>0: Device mode 1: Host mode</p> <p>Note: Accessible in both host mode and device mode.</p>	R

31.7.2.6 USBFS Global Interrupt Mask Register (USBFS_GINTMSK)

USBFS interrupt mask register

offset address: 0x18

Reset value: 0x00000000

This register is used in conjunction with the module interrupt register to interrupt the application. If an interrupt bit is masked, no interrupt associated with that bit will be generated.

However, the module interrupt (USBFS_GINTSTS) register bit corresponding to this interrupt will still be set to 1.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
WKU	VBU	DIS CIM	CID SCH GM	Res erv ed	PTX FEM	HCI M	HPR TIM	Res erv ed	DAT AFS USP M	IPX FRM /	IIS OIX FRM	OEP IM	IEP IM	Res erv ed	Res erv ed
IM	SVI								USP M	INC OMP ISO OUT M					
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EOP FM	ISO ODR PM	ENUM DNEM	USBR STM	USBS USPM	ESU SPM	Res erv ed	Res erv ed	GON AKE FFM	GIN AKE FFM	NPT XFE M	RXF NEM	SOF M	Res erv ed	MMI SM	Res erv ed

position	Marker	Place Name	Function	Reading and writing
b31	WKUIM	Recovery/remote wake-up interrupt mask detected	Resume/remote wakeup detected interrupt mask 0: Mask interrupt 1: Enabling interrupts Note: Accessible in both host mode and device mode.	R/W
b30	VBUSVIM	VBUS valid interrupt mask	VBUS valid interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in device mode only.	R/W
b29	DISCIM	Disconnection interrupt mask detected	Disconnect detected interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in host mode only.	R/W
b28	CIDSCHGM	Interrupt connector ID line state change interrupt mask	Connector ID line status change interrupt mask (Connector ID status change interrupt mask) 0: Mask interrupt 1: Enable interrupt Note: Accessible in both device mode and host mode.	R/W
b27	Reserved	-	The reset value must be maintained.	R

b26	PTXFEM	Periodic TxFIFO air break shield	Periodic TxFIFO empty interrupt mask 0: Mask interrupt 1: Enabling interrupts Note: Accessible in host mode only.	R/W
b25	HCIM	Host channel interrupt shield	Host channels interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in host mode only.	R/W
b24	HPRTIM	Host port interrupt masking	Host port interrupt mask 0: Mask interrupt 1: Enabling interrupts Note: Accessible in host mode only.	R/W
b23	Reserved	-	The reset value must be maintained.	R
b22	DATAFSUSP M	Data acquisition hang interrupt mask	Data fetch suspended interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in device mode only.	R/W
b21	IPXFRM/ INCOMPISO OUTM	Incomplete periodic transmission interrupt mask/ Incomplete OUT synchronous transmission interrupt mask	IPXFR: Incomplete periodic transfer interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in host mode only. INCOMPISOOUT: Incomplete isochronous OUT transfer interrupt mask 0: Mask interrupt 1: Enabling interrupts Note: Accessible in device mode only.	R/W
b20	IISOIXFRM	Incomplete IN sync transmission interrupt mask	Incomplete isochronous IN transfer interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in device mode only.	R/W
b19	OEPIM	OUT endpoint interrupt mask	OUT endpoint interrupt mask 0: Mask interrupt 1: Enabling interrupts Note: Accessible in device mode only.	R/W
b18	IEPIM	IN endpoint interrupt mask	IN endpoint interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in device mode only.	R/W
b17~b16	Reserved	-	The reset value must be maintained.	R
b15	EOPFM	Periodic end-of-frame interrupt mask	End of periodic frame interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in device mode only.	R/W
b14	ISOODRPM	Discard synchronous OUT packet interrupt mask	Isochronous OUT packet dropped interrupt mask 0: Mask interrupt 1: Enabling interrupts Note: Accessible in device mode only.	R/W

b13	ENUMDNEM	Enumeration completion interrupt mask	Enumeration done interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in device mode only.	R/W
b12	USBRSTM	USB reset interrupt mask	USB reset interrupt mask 0: Mask interrupt 1: Enabling interrupts Note: Accessible in device mode only.	R/W
b11	USBSUSPM	USB hang interrupt mask	USB suspend interrupt mask 0: Mask interrupt 1: Enabling interrupts Note: Accessible in device mode only.	R/W
b10	ESUSPM	Early Hanging Interrupt Mask	Early suspend interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in device mode only.	R/W
b9~b8	Reserved	-	The reset value must be maintained.	R
b7	GONAKEFFM	Global OUT NAK valid interrupt mask	Global OUT NAK effective interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in device mode only.	R/W
b6	GINAKEFFM	Global non-periodic IN NAK valid interrupt mask	Global nonperiodic NAK effective interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in device mode only.	R/W
b5	NPTXFEM	Non-periodic TxFIFO air break shield	Non-periodic TxFIFO empty interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in host mode only.	R/W
b4	RXFNEM	RxFIFO non-air break shield	RxFIFO non-empty interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in both host mode and device mode.	R/W
b3	SOFM	Frame start interrupt mask	Start of frame interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in both host mode and device mode.	R/W
b2	Reserved	-	The reset value must be maintained.	R

b1	MMISM	Pattern mismatch interrupt interrupt mask	Mode mismatch interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in both host mode and device mode.	R/W
b0	Reserved	-	The reset value must be maintained.	R

31.7.2.7 USBFS Receive Status Debug Read/USBFS Status read and out-stack register (USBFS_GRXSTSR/USBFS_GRXSTSP)

USBFS Receive status debug read/USBFS status read and pop registers read

offset address: 0x01C

Offset address of the

out-stack: 0x020 Reset

value: 0x0000 0000

The Read Receive Status Debug Read register will return the contents of the top of the receive FIFO. The Read Receive Status Read and Out Stack registers will additionally pop up the data entry at the top of the RxFIFO. The receive status contents are interpreted differently in host mode and device mode.

When the receive FIFO is empty, the module ignores read or stack out operations to this register and returns the value 0x0000 0000. When the receive FIFO non-empty bit (RXFNE bit in USBFS_GINTSTS) of the module interrupt register is set, the application must only eject the receive status FIFO.

Host mode:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										PKTSTS[3:0]				DPI	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DPID [O]	BCNT [11:0]										CHNUM[3:0]				

position	Marker	Place Name	Function	Reading and writing
b31~b21	Reserved	-	The reset value must be maintained.	R
b20~b17	PKTSTS	Packet Status	Packet status indicates the status of the received packet 0010: IN packet received 0011: IN transmission complete (triggered interrupt) 0101: Data synchronization error (trigger interrupt) 0111: Pause channel (trigger interrupt) Other values: Reserved	R

b16~b15	DPID	Data PID	Data PID Indicates the data PID of the received packet 00: DATA0 10: DATA1 01 : DATA2 11: MDATA	R
b14~b4	BCNT	Byte Count	Byte count Indicates the number of bytes of IN packets received.	R
b3~b0	CHNUM	Channel number	Channel number Indicates the channel number to which the currently received packet belongs.	R

**Device mode:**

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										PKTSTS[3:0]			DPID [1]		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DPI 1	BCNT [11:0]										EPNUM[3:0]				

position	Marker	Place Name	Function	Reading and writing
b31~b21	Reserved	-	The reset value must be maintained.	R
b20~b17	PKTSTS	Packet Status	Packet status Indicates the status of the received packet 0001: 0010: Global OUT NAK (trigger interrupt) 0011: OUT packet received 0100: SETUP transaction completed (trigger interrupt) 0110: SETUP packet received Other values: Reserved	R
b16~b15	DPID	Data PID	Data PID (Data PID) Indicates the data PID of the received OUT packet 00: DATA0 10: DATA1 01 : DATA2 11: MDATA	R
b14~b4	BCNT	Byte Count	Byte count Indicates the number of bytes in the received packet.	R
b3~b0	EPNUM	Endpoint number	Endpoint number Indicates the endpoint number to which the currently received packet belongs.	R

31.7.2.8 USBFS Receive FIFO Size Register (USBFS_GRXFSIZ)

USBFS Receive FIFO size register

offset address: 0x024

Reset value: 0x0000 0140

This application can program the size of RAM that must be allocated to the RxFIFO.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved					RXFD[10:0]										

position	Marker	Place Name	Function	Reading and writing
b31~b11	Reserved	-	The reset value must be maintained.	R
b10~b0	RXFD	RxFifo depth	RXFIFO: RxFIFO depth (RxFIFO depth) in 32-bit word units. Minimum value is 16 Maximum value is 256 The power-on reset value is the maximum Rx data FIFO depth.	R/W

31.7.2.9 USBFS host non-periodic send FIFO size register (USBFS_HNPTXFSIZ)/endpoint 0 send FIFO size (USBFS_DIEPTXF0)

USBFS Host non-periodic transmit FIFO size register/Device endpoint0 transmit FIFO size register

Offset address: 0x028

Reset value: 0x02000140

This application can program the size of RAM that must be allocated to the TxFIFO.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
nptxfd[15:0]/tx0fd[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
nptfsa[15:0]/TX0fsa[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	NPTX	Non-periodic	Host mode: NPTXFD	R/W
	FD/	TxFIFO	Non-periodic TxFIFO depth (Non-periodic	
	TX0FD	depth/endpoint	TxFIFO depth) is measured in 32-bit words.	
		0 TxFIFO depth	Minimum value is 16	
			Maximum value is 256	
			Device mode: TX0FD	
			The Endpoint 0 TxFIFO depth (Endpoint 0 TxFIFO depth) is in 32-bit words.	
			Minimum value is 16	
			Maximum value is 256	
b15~b0	NPTXFSA/	Non-periodic send	Host mode: NPTXFSA	R/W
	TX0FSA	RAM start address / Endpoint 0 send	Non-periodic transmit RAM start address (Non-periodic transmit RAM start address)	
		RAM start address	This field contains the memory start address of the non-periodic transmit FIFO RAM.	
			Device mode: TX0FSA	
			Endpoint 0 transmit RAM start address (Endpoint 0 transmit RAM start address)	
			This field contains the memory start address of the endpoint 0 send FIFO RAM.	

31.7.2.10 USBFS non-periodic send FIFO/queue status register (USBFS_HNPTXSTS)

USBFS Host non-periodic transmit FIFO size register/Device endpoint0
transmit FIFO size register

Offset address: 0x02C

Reset value: 0x00080100

This read-only register contains free space information for the acyclic TxFIFO and acyclic send request queues. This register is valid only in host mode, not in device mode.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
Res erv ed	NPTXQTOP [6:0].								NPTQXSAR[7: 0]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
NPTXFSAR [15:0]																

position	Marker	Place Name	Function	Reading and writing
b31	Reserved	-	The reset value must be maintained.	R
b30~b24	NPTXQTOP	Acyclic send request queue top	Top of the non-periodic transmit request queue The entry in the acyclic send request queue that the MAC is currently processing. Bits 30:27: Channel/endpoint number (Channel/endpoint number) Bits 26:25: – 00: IN/OUT token – 01: Sending packets of zero length – 11: Channel stop command Bit 24: Terminate (last entry for selected channel/endpoint)	R
b23~b16	NPTQXSAR	Space available in the acyclic send request queue	Space available in the acyclic send request queue (Non-periodic transmit request queue space available) Indicates the amount of free space available in the acyclic send request queue. In host mode, this queue holds IN and OUT requests. 00: The acyclic send request queue is full 01: 1 position available 10: 2 positions available bxn: n positions available (where n range: 0~8) Other values: Reserved	R

b15~b0	NPTXFSAV	Acyclic send request queue top	Non-periodic TxFIFO space available Indicates the amount of free space available in the non-periodic TxFIFO. In 32-bit word units. 00: Acyclic TxFIFO is full 01: 1 word available 10: 2 words available 0xn: n words available (where, n range: 0~256) Other values: Reserved	R
--------	----------	--------------------------------	---	---

31.7.2.11 USBFS module ID register (USBFS_CID)

USBFS core ID register

offset address: 0x03C

Reset value: 0x12345678

This register is the programmable user configuration register.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PRODUCT_ID[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PRODUCT_ID[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b0	PRODUCT_I	Product ID field D	Product ID field (Product ID field) is an ID field that can be programmed through the application.	R/W

31.7.2.12 USBFS Host Periodic Send FIFO Size Register (USBFS_HPTXFSIZ)

USBFS Host periodic transmit FIFO size

register offset address: 0x100

Reset value: 0x01400280

This application can program the RAM size that must be allocated to the cycle TxFIF.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved					PTXFD[10:0]										
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved					PTXSA [11:0]										

position	Marker	Place Name	Function	Reading and writing
b31~b27	Reserved	-	The reset value must be maintained.	R
b26~b16	PTXFD	Host Periodicity TxFIFO Depth	The Host periodic TxFIFO depth is in 32-bit words. Minimum value is 16 Maximum value is 256	R/W
b15~b12	Reserved	-	The reset value must be maintained.	R
b11~b0	PTXSA	Host Periodicity TxFIFO Start Address	Host periodic TxFIFO start address (Host periodic TxFIFO start address) The power-on reset value is the sum of the maximum RxFIFO depth and the maximum non-periodic TxFIFO depth.	R/W

31.7.2.13 USBFS Device IN Endpoint Send FIFO Size Register (USBFS_DIEPTXF_x) ($x = 1..5$)

USBFS device IN endpoint transmit FIFO size

register offset address: 0x104+($x-1$)*0x4

Reset value: 0x01000240+ $(x-1)*0x100$

This application can program the size that must be allocated to the device TxFIFO.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved								INEPTXFD[9:0]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								INEPTXSA[11:0]							

position	Marker	Place Name	Function	Reading and writing
b31~b26	Reserved	-	The reset value must be maintained.	R
b25~b16	INEPTXFD	IN endpoint TxFIFO depth	The Device IN endpoint TxFIFO depth is in 32-bit words. Minimum value is 16 Maximum value is 256	R/W
b15~b12	Reserved	-	The reset value must be maintained.	R
b11~b0	INEPTXSA	IN endpoint TxFIFO RAM start address	IN endpoint TxFIFOx RAM start address (IN endpoint FIFOx transmit RAM start address) This field contains the memory start address of the IN endpoint transmit FIFOx. The address must be aligned with a 32-bit memory location.	R/W

31.7.3 USBFS Host Mode Register

Host mode registers affect module operations in host mode. Host mode registers must not be accessed in device mode because the resulting result is unclear.

The bit values in the register descriptions are expressed in binary unless otherwise noted.

31.7.3.1 USBFS Host Configuration Register (USBFS_HCFG)

USBFS Host configuration register

offset address: 0x400

Reset value: 0x00000000

This register will be configured for the module after power-up. Do not change this register after initializing the host.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															

position	Marker	Place Name	Function	Reading and writing
b31~b3	Reserved	-	The reset value must be maintained.	R
b2	FSLSS	FS and LS support only	The application uses this bit to control the enumeration speed of the module. Using this bit, the application can make the module work as an FS host, even if the connected device supports HS communication. Do not change this field after initial programming. 1: FS/LS only, even if the connected device can support HS	R/W
b1~b0	FSLSPCS	FS/LS PHY clock selection	FS/LS PHY clock select when the module is in FS host mode 01: PHY clock running at 48 MHz Other values: Reserved When the module is in LS host mode 00: Reserved 01: Select 48MHz PHY clock frequency 10: Select 6MHz PHY clock frequency 11: Reservation Note: When the device is connected to the host computer, the FSLSPCS must be set according to the speed of the connected device (more After changing this bit, a software reset must be performed)	R/W

31.7.3.2 USBFS Host Frame Interval Register (USBFS_HFIR)

USBFS Host frame interval

register offset address: 0x404

Reset value: 0x0000EA60

This register is used to store the frame interval information set by the USBFS controller for the current speed of the connected device.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FRIVL [15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b26	Reserved	-	Reset value must be maintained	R
b15~b0	FRIVL	Frame interval	<p>Frame interval</p> <p>The value programmed by the application in this field is used to specify the time interval between two consecutive SOF (FS) or Keep-Alive tokens (LS). This field contains the number of PHY clocks that make up the desired frame interval. The application can write a value to this register only after setting the Port Enable bit of the Host Port Control and Status register (PENA bit of USBFS_HPRT) to 1. If the value is not programmed, the module will calculate the PHY clock based on the PHY clock specified in the FS/LS PHY Clock Select field of the Host Configuration Register (FSLSPCS in USBFS_HCFG). Do not change the value of this field after the initial configuration. Set value = Frame interval (ms) × (PHY clock frequency) – 1</p> <p>Note: The FRIVL bit can be modified whenever the application needs to change the frame interval time.</p>	R/W

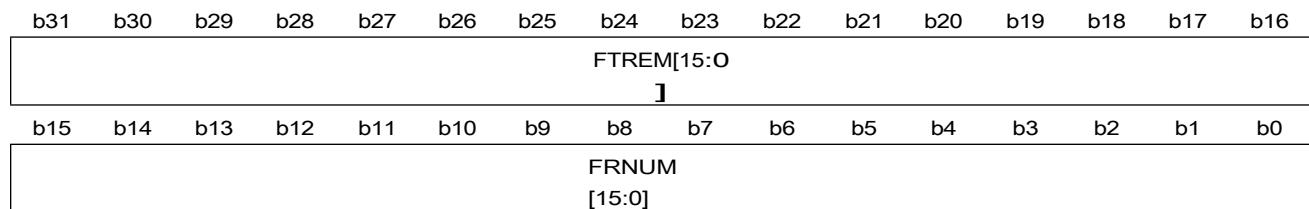
31.7.3.3 USBFS host frame number/frame remaining time register (USBFS_HFNUM)

USBFS Host frame interval

register offset address: 0x408

Reset value: 0x0000 3FFF

This register is used to indicate the current frame number. It also indicates the remaining time (in PHY clock counts).



position	Marker	Place Name	Function	Reading and writing
b31~b16	FTREM	Frame time remaining	Frame time remaining Indicates the time remaining in the current frame (in PHY clocks) This field is decremented by 1 for each PHY clock that has elapsed. When the value reaches zero, this field reloads the value in the frame interval register and is used by the module on the USB Send a new SOF.	R
b15~b0	FRNUM	Frame number	Frame number The value of this field is incremented by 1 when 1 new SOF is sent on the USB and is cleared when it reaches 0x3FFF.	R

31.7.3.4 USBFS Host periodic transmit FIFO/queue status

register (USBFS_HPTXSTS) USBFS Host periodic transmit

FIFO/queue status register Offset address: 0x410

Reset value: 0x00080100

This read-only register contains free space information for the periodic TxFIFO and periodic send request queues.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PTXQTOP[7:0]								PTXQSAV [7:0]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PTXFSAVL[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b24	PTXQTOP	Periodically send the top of the request queue	Periodically send the top of the request queue (Top of the periodic transmit request queue) Indicates the item in the periodic Tx request queue that the MAC is currently processing. This register is used for debugging. Bit 31: Odd/Even frame <ul style="list-style-type: none">– 0: send in even frames– 1: Send in odd number of frames Bits 30:27: Channel number Bits 26:25: Type <ul style="list-style-type: none">– 00: Input/output– 01: Zero Length Packets– 11: Prohibit channel command Bit 24: Terminate (last entry for the selected channel)	R
b23~b16	PTXQSAV	Periodic send request queue free space	Periodic send request queue free space (Periodic transmit request queue space available) indicates the number of free positions in the periodic transmit request queue available for writing. This queue contains both IN requests and OUT requests. 00: Periodic send request queue is full 01: 1 position available 10: 2 positions available bxn: n positions available (where, n range: 0~8) Other values: Reserved	R

b15~b0	PTXFSAVL	Periodically send data FIFO free space	Periodic send data FIFO free space (Periodic transmit data FIFO space available) indicates the number of free periodic TxFIFO locations that are available for writing. In 32-bit words 0000: Periodic TxFIFO is full 0001: 1 word available 0010: 2 words available bxn: n words available (where, n range: 0~PTXFD) Other values: Reserved	R
--------	----------	--	---	---

31.7.3.5 USBFS Host all channels

interrupt register (USBFS_HINT) USBFS

Host all channels interrupt

register Offset address: 0x414

Reset value: 0x0000 0000

When an event occurs on a channel, the host all-channel interrupt register uses the host channel interrupt bit in the module interrupt register (HCINT bit in USBFS_GINTSTS) Interrupt application. Each channel corresponds to 1 interrupt bit, with a maximum of 12 bits. When the application clears the interrupt through the corresponding host channel x interrupt register, the bits in this register are also cleared.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				HINT[11:0]											

position	Marker	Place Name	Function	Reading and writing
b31~b12	Reserved	-	The reset value must be maintained.	R
b11~b0	HINT	Channel interruption	Channel interrupt Each channel corresponds to one bit: channel 0 corresponds to bit 0 and channel 11 corresponds to bit 11.	R/W

31.7.3.6 USBFS Host all channels interrupt

mask register (**USBFS_HAINTMSK**) USBFS Host all

channels interrupt mask register Offset address:

0x418

Reset value: 0x0000 0000

The host-wide channel interrupt mask register is used in conjunction with the host-wide channel interrupt register, which in turn interrupts the application when an event occurs on the channel.

Each channel corresponds to 1 interrupt mask bit, with a maximum of 12 bits.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				HAINTM [11:0]											

position	Marker	Place Name	Function	Reading and writing
b31~b12	Reserved	-	Read "0", write "0" when writing	R
b11~b0	HAINTM	Channel interrupt masking	Channel interrupt mask 0: Mask interrupt 1: Enable interrupt Each channel corresponds to one bit: channel 0 corresponds to bit 0 and channel 11 corresponds to bit 11.	RW

31.7.3.7 USBFS Host port control and status register (USBFS_HPRT) USBFS Host port control and status register Offset

address: 0x440

Reset value: 0x0000 0000

This register is only available in host mode. Currently, the USBFS host supports only one port.

This register contains USB port related information, such as USB reset, enable, hang, resume, and connection status. The PENCHNG/PCDET bits in this register can trigger an application interrupt via the host port interrupt bit in the module interrupt register (HPRTINT bit in USBFS_GINTST). When a port interrupt occurs, the application must read this register and clear the bit that caused the interrupt to zero. The application must write a 1 to this bit to clear the interrupt.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved														PSPD[1:0]	Rese rved
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved	PWP R	PLSTS[1:0]	Res erv ed	PRS T	PSU SP	PRE S	Reserved	PEN CHN G	PEN A	PCD ET	PCST S				

position	Marker	Place Name	Function	Reading and writing
b31~b19	Reserved	-	The reset value must be maintained.	R
b18~b17	PSPD	Port Speed	Port speed Indicates the speed of the device connected to the port. 00/11: Reserved 01: Full speed 10: Low speed	R
b16~b13	Reserved	-	The reset value must be maintained.	R
b12	PPWR	Port Power	Port power The application uses this field to control the power to this port. Since the built-in PHY of this USBFS does not have power supply capability, this bit, when set to 1, enables the external USB power chip to supply power via USBFS_DRVVBUS. 0: Power down 1: Power on	R/W
b11~b10	PLSTS	Port Line Status	Indicates the current logic level of the USB data line Bit 11: Logic level of USBFS_DM Bit 10: Logic level of USBFS_DP	R
b9	Reserved	-	The reset value must be maintained.	R

b8	PRST	Port Reset	Port reset	R/W
			<p>The application initiates a reset sequence on this port when it sets this bit to 1.</p> <p>The application must time the reset cycle and clear this bit to zero when the reset sequence is complete.</p> <p>0: Port is not in reset state</p> <p>1: Port is in reset state</p> <p>The application must set this position 1 and hold it for a minimum of 10 ms to initiate a reset on the port.</p>	

b7	PSUSP	Port hangs	Port suspend The application puts this position 1 to put this port in pending mode. The module will stop sending SOF only when this position is 1. To stop the PHY clock, the application must stop the port clock in position 1, which will enable the hang input pin of the PHY. The read value of this bit reflects the current pending state of the port. The module can clear this bit when a remote wakeup signal is detected or when the application sets the port reset bit or port recovery bit in this register to 1. Or the module can clear this bit when the application sets the recovery/remote wakeup detection interrupt bit or the disconnect detection interrupt bit (WKUINT or DISCINT in USBFS_GINTSTS, respectively) in the module interrupt register to 1. . 0: The port is not in pending mode 1: The port is in hang mode	R/W
b6	PRES	Port Recovery	Port resume The application sets this bit to 1 to drive the recovery signal on this port. The module will continue to drive the recovery signal until the application clears this bit. As indicated by the Port Recovery/Remote Wakeup Detection interrupt bit in the module interrupt register (WKUINT bit in USBFS_GINTSTS), if the module detects a USB remote wakeup sequence, it starts driving the recovery signal without application intervention; if the module detects a disconnected condition, it clears this bit to zero. The read value of this bit indicates whether the module is currently driving the recovery signal. 0: No recovery signal is being driven 1: Drive recovery signal	R/W
b5~b4	Reserved	-	The reset value must be maintained.	R
b3	PENCHNG	Port enable/disable change	Port enable/disable change When the state of the port enable bit 2 in this register changes, the module clears this location 1. by writing 1 to this bit by software.	R/W
b2	PENA	Port Enable	Port enable After the port performs a reset sequence, it can only be enabled by the module and can be disabled by an overcurrent condition, a disconnect condition, or by the application clearing this bit to zero. The application cannot disable the port by performing a write operation to the register in position 1. It can only disable the port by clearing this bit. Operation of this bit will not trigger any interrupts by the application. 0: Forbidden port 1: Enabling port	R/W
b1	PCDET	Port connection detected	Port connect detected When a device connection is detected, the module sets this position 1 to trigger an application interrupt using the host port interrupt bit in the module interrupt register (the HPRTINT bit in USBFS_GINTSTS). The application must set this position 1 to clear the interrupt.	R/W
b0	PCSTS	Port connection status	Port connect status 0: The port is not connected to the device 1: The port is connected to the device	R

31.7.3.8 USBFS Host Channel x Characteristic Register (USBFS_HCCHARx) (x = 0..11)

USBFS Host channel-x characteristics register

offset address: 0x500 + (channel number × 0x20)

Reset value: 0x0000 0000

This hosting is used to set the host channel characteristics.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CHE NA	CHD IS	ODD FRM		DAD[6:0]						Reserved	EPTYP[1:0] 1	LSD EV	Rese rved		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EPD IR		EPNUM[3:0]								MPSIZ [10:0]					

position	Marker	Place Name	Function	Reading and writing
b31	CHENA	Channel Enable	Channel enable This field is set to 1 by the application software and cleared to zero by the USBFS host hardware. 0: Channel disabled 1: Enabling channel	R/W
b30	CHDIS	Channel prohibition	Channel disable The application sets this position 1 to stop sending/receiving data through the channel, and the stop operation remains in effect even if the transmission through the channel is not yet complete. The application must wait for a disable channel interrupt to confirm that the channel has been disabled.	R/W
b29	ODDFRM	Odd frames	Odd frame This field is set or reset by the application to indicate that the USBFS host must transmit an odd or even number of frames, respectively. This field is only available for periodic (synchronous and interrupt) transactions. 0: Even frames 1: Odd number of frames	R/W
b28~b22	DAD	Device Address	Device address This field is used to specify the specific device to communicate with this host.	R/W
b21~b20	Reserved	-	The reset value must be maintained.	R
b19~b18	EPTYP	Endpoint Type	The Endpoint type indicates the type of transmission selected. 00: Control 01: Synchronization 10: Batch 11: Interruption	R/W
b17	LSDEV	Low-speed equipment	Low-speed device This field is set to 1 by the application to indicate that this channel is communicating with a low-speed device.	R/W
b16	Reserved	-	The reset value must be maintained.	R

b15	EPDIR	Endpoint Direction	Endpoint direction indicates whether the direction of the communication transaction is input or output. 0: Output 1: Input	R/W
-----	-------	-----------------------	---	-----

b14-b11	EPNUM	Endpoint number	Endpoint number Indicates the endpoint number of the USB device to communicate with this host channel.	R/W
b10-b0	MPSIZ	Maximum packet size	Maximum packet size Indicates the maximum packet size of the device endpoint communicating with this host channel.	R/W

31.7.3.9 USBFS host channel x interrupt register (USBFS_HCINTx) (x = 0..11)

USBFS Host channel-x interrupt

register offset address: 0x508 + (channel

number × 0x20)

Reset value: 0x0000 0000

This register indicates the status of the channel in the event of USB and AHB related events. When the host channel interrupt bit in the module interrupt register

(The application must read this register when the HCINT bit (in USBFS_GINTSTS) is set to 1. Before performing a read operation on the register, the application must read the Host All Channel Interrupt (USBFS_HAINT) register to obtain the exact channel number of the Host Channel x Interrupt register. The application must clear the corresponding bits in this register before it can clear the corresponding bits in the USBFS_HAIN and USBFS_GINTSTS registers.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved					DTE RR	FRM OR	BBE RR	TXE RR	Res erv ed	ACK	NAK	STA LL	Res erv ed	CHH	XFRC

position	Marker	Place Name	Function	Reading and writing
b31~b11	Reserved	-	The reset value must be maintained.	R
b10	DTERR	Data switching error	Data toggle error The application needs to clear this bit by writing 1.	R/W
b9	FRMOR	Frame overflow error	Frame overrun error (Frame overrun) The application needs to clear this bit by writing 1.	R/W
b8	BBERR	Crosstalk error	Crosstalk error (Babble error) A typical cause of a crosstalk event is when an endpoint sends a packet, but the packet length exceeds the maximum packet length for the endpoint. The application needs to clear this bit by writing 1.	R/W
b7	TXERR	Communication transaction error	A communication transaction error indicates that one of the following errors has occurred on the USB: CRC checksum failure timeout Bit Fill Error Error	R/W

EOP

The application needs to clear this bit by writing 1.

b6	Reserved	-	The reset value must be maintained.	R
b5	ACK	ACK response received/issued	ACK response received/transmitted interrupt The application needs to clear this bit by writing 1.	R/W
b4	NAK	NAK response received	The NAK response received interrupt application needs to clear this bit by writing 1.	R/W
b3	STALL	STALL response received	STALL response received interrupt	R/W
b2	Reserved	-	Read "0", write "0" when writing	R

b1	CHH	Channel stop	Channel halted The transfer ended abnormally due to an arbitrary USB transaction error or in response to a forbidden request from the application. The application needs to clear this bit by writing a 1.	R/W
b0	XFRC	Transmission complete	Transfer completed There are no errors and the transfer is completed normally. The application needs to clear this bit by writing 1.	R/W

31.7.3.10 USBFS Host Channel x Interrupt Mask Register (USBFS_HCINTMSKx) (x = 0..11)

USBFS Host channel-x interrupt mask

register offset address: 0x50C + (channel number × 0x20)

Reset value: 0x0000 0000

This register is used to select to mask the host channel interrupt.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved					DTE RRM	FRM ORM	BBE RRM	TXE RRM	Res erv ed	ACK M	NAK M	STA LLM	Res erv ed	CHH M	XFRC M

position	Marker	Place Name	Function	Reading and writing
b31~b11	Reserved	-	The reset value must be maintained.	R
b10	DTERRM	Data switching error interrupt mask	Data toggle error mask 0: Mask interrupt 1: Enable interrupt	R/W
b9	FRMORM	Frame overflow error interrupt mask	Frame overrun mask 0: Mask interrupt 1: Enable interrupt	R/W
b8	BBERRM	Crosstalk error interrupt masking	Babble error mask 0: Mask interrupt 1: Enable interrupt	R/W
b7	TXERRM	Communication transaction error interrupt mask	Communication transaction error mask (Transaction error mask) 0: Mask interrupt 1: Enable interrupt	R/W
b6	Reserved	-	The reset value must be maintained.	R
b5	ACKM	ACK received/issued Response interrupt mask	ACK response receive/transmit interrupt mask (ACK response received/transmitted interrupt mask) 0: Masked interrupt 1: Enable interrupt	R/W
b4	NAKM	Receive NAK response interrupt mask	NAK response received interrupt mask 0: Mask interrupt 1: Enable interrupt	R/W
b3	STALLM	Received STALL response interrupt mask	STALL response received interrupt mask 0: Mask interrupt 1: Enable interrupt	R/W
b2	Reserved	-	The reset value must be maintained.	R

b1

CHHM

Channel stop
interrupt maskChannel halted mask 0: Mask interrupt
1: Enabling interrupts

R/W

b0	XFRCM	Transmission completion interrupt mask	Transfer completed mask 0: Mask the interrupt 1: Enable interrupt	R/W
----	-------	--	--	-----

31.7.3.11 USBFS Host Channel x Transfer Size Register (USBFS_HCTSIZx) (x = 0..11)

USBFS Host channel-x transfer size register

offset address: 0x510 + (channel number × 0x20)

Reset value: 0x0000 0000

This register is used to set the host channel transfer size and the data PID.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Res	DPID[1:0]	PKTCNT [9:0]										XFRSIZ[18:16]			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
XFRSIZ[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31	Reserved	-	The reset value must be maintained.	R
b30~b29	DPID	Data PID	Data PID (Data PID) The application sets the initial synchronization PID for data communication in this field. the host retains the setting of this field during this transmission transaction. 00: DATA0 01: Reservation 10: DATA1 11 : SETUP	R/W
b28~b19	PKTCNT	Packet Counting	Packet count The application sets the number of packets that will be sent or received in this field. The host decrements the count value once for each successful packet sent or received. When this value reaches 0, the application will be interrupted to indicate the normal completion of the operation.	R/W
b18~b0	XFRSIZ	Transfer size	Transfer size For OUT operations, this field is the number of bytes of data sent by the host during transmission. For IN operations, this field is the size of the buffer that the application reserves for the transfer. For IN transactions (periodic and acyclic), the application programs this field to be an integer multiple of the maximum packet size.	R/W

31.7.3.12 USBFS host channel xDMA address register (USBFS_HCDMAx) (x = 0..11)

USBFS Host channel-x DMA address

register offset address: 0x514 + (channel number
× 0x20)

Reset value: 0xXXXX XXXX

This register is used to set the DMA address during host DMA mode.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DMAADDR[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DMAADDR[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b0	DMAADDR	DMA Address	DMA address (DMA address) This field stores the address of the memory used by the host to obtain data from or send data to the device endpoint for DMA transfers. This register is incremented at the end of each AHB transfer.	R/W

31.7.4 USBFS Device Mode Register

Device mode registers affect module operations in device mode. Device mode registers must not be accessed in host mode because the resulting result is unclear.

The bit values in the register descriptions are expressed in binary unless otherwise noted.

31.7.4.1 USBFS Device Configuration Register (USBFS_DCFG)

USBFS Device configuration register offset

address: 0x800

Reset value: 0x0820 0000

This register configures the module to device mode after power-up, execution of certain control commands, or enumeration. Do not change this register after initial programming.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved	PFIVL[1:0]	DAD[6:0]				Reserved	NZLSOHSK	DSPD[1:0]							

position	Marker	Place Name	Function	Reading and writing
b31~b13	Reserved	-	The reset value must be maintained.	R
b12~b11	PFIVL	Periodic frame interval	Periodic frame interval Indicates the point in a frame when the application must be notified using a periodic frame break. This function can be used to determine if all synchronous communication for the frame is complete. 00: 80% frame interval 01: 85% frame interval 10: 90% frame interval 11: 95% frame interval	R/W
b10~b4	DAD	Device Address	Device address The application must set this field according to the command parameters after each SetAddress control command is executed.	R/W
b3	Reserved	-	The reset value must be maintained.	R
b2	NZLSOHSK	Non-zero length state OUT handshake signal (Non-zero-length status OUT handshake) handshake signal	Non-zero length state OUT handshake signal (Non-zero-length status OUT handshake) This field can be used by the application to select the handshake signal to be sent when the module receives a non-zero length packet during the OUT transaction of the Control Transmission Status phase. 1: When a non-zero length state OUT transaction is received, the STALL handshake signal is replied and the received OUT packets are not sent to the application. 0: Send the received OUT packets (zero length or non-zero length) to the application and based on the device	R/W
			The NAK and STALL bits of the endpoint in the endpoint control register reply	

to the handshake signal.

b1~b0	DSPD	Equipment speed	Device speed	R/W
			Indicates the speed that the application requires the module to use for enumeration, or the maximum speed supported by the application. However, the actual bus speed can only be determined after the chirp sequence is completed, and this speed is based on the speed of the USB host to which the module is connected.	
			00: Reserved	
			01: Reservation	
			10: Reserved	
			11: Full speed (USB 1.1 transceiver clocked at 48 MHz)	

31.7.4.2 USBFS Device Control Register (USBFS_DCTL)

USBFS Device control

register offset address: 0x804

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				POP RGD NE	CGO NAK	SGO NAK	CGI NAK	SGI NAK	Reserved			GON STS	GIN STS	SDI S	RWU SIG

position	Marker	Place Name	Function	Reading and writing
b31~b12	Reserved	-	The reset value must be maintained.	R
b11	POPRGDNE	Power-up programming complete	Power-on programming done This bit is used by the application to indicate that the register has been programmed after waking up from power-down mode.	R/W
b10	CGONAK	Clear global OUT NAK	Clear global OUT NAK Performing a write operation to this bit will clear the global OUT NAK to zero.	W
b9	SGONAK	Places the global OUT NAK	Set global OUT NAK (Set global OUT NAK) Performing a write operation to this bit will set the global OUT NAK to 1. The application uses this bit to send the NAK handshake signal at all OUT endpoints. The application only determines that the global OUT NAK valid bit in the module interrupt register (GONAKEFF bit in USBFS_GINTSTS) has been cleared to zero before this position 1 can be set.	W
b8	CGINAK	Zeroing out the global IN NAK	Clear global IN NAK Performing a write operation to this bit will clear the global IN NAK to zero.	W
b7	SGINAK	Placement Global IN NAK	Set global IN NAK Performing a write operation to this field sets the global acyclic IN NAK to 1. The application uses this bit to cause all acyclic IN endpoints to send NAK handshake signals. The application can only determine the global IN NAK valid bit in the module interrupt register (GINAKEFF bit in USBFS_GINTSTS) has been cleared to zero before this position 1 can be set.	W
b6~b4	Reserved	-	The reset value must be maintained.	R
b3	GONSTS	Global OUT NAK status	Global OUT NAK status (Global OUT NAK status) 0: The handshake signal will be sent based on the FIFO status and the NAK and STALL bit settings. 1: No data is received regardless of whether there is still free space in the RxFIFO. Reply to the NAK handshake signal for all received packets except	R

SETUP transactions. All synchronous type OUT packets will be
Discard.

b2	GINSTS	Global IN NAK status	Global IN NAK status (Global IN NAK status) 0: will reply to the handshake signal according to the data availability in the sending FIFO. 1: Make all non-periodic IN endpoints reply to the NAK handshake signal without considering the number in the transmit FIFO According to availability.	R
----	--------	----------------------	---	---

b1	SDIS	soft disconnection	Soft disconnect The application uses this bit to signal the USBFS module to perform a soft disconnect. With this bit in position 1, the host will not see that the device is connected and the device will not receive signals on the USB. The module will remain disconnected until the application clears this bit to zero. 0: Normal operation. This bit clears after a soft disconnect and causes the host to receive an event that the device is connected. After reconnecting the device, the USB host restarts the device enumeration. 1: Causes the host to receive an event that the device is disconnected.	R/W
b0	RWUSIG	Sending a remote wakeup signal	Sending Remote wakeup signaling When the application sets this position 1, the module initiates a remote wake-up signal to wake up the USB host. The application must set this position 1 to bring the module out of the hang state. According to the USB 2.0 specification, the application must clear this position 1 within 1 ms to 15 ms after it is set.	R/W

31.7.4.3 USBFS Device Status Register (USBFS_DSTS)

USBFS Device status register

offset address: 0x808

Reset value: 0x0000 0002

This register indicates the status of the module in the event of a USB-related event. When an interrupt occurs, the endpoint information of the interrupt must be read from the Device All Interrupts (USBFS_DAINT) register.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										FNSOF [13:8]					
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FNSOF[7: 0]										Reserved		EER R	ENUMSPD[1 : 0]	SUS PST S	

position	Marker	Place Name	Function	Reading and writing
b31~b22	Reserved	–	The reset value must be maintained.	R
b21~b8	FNSOF	Frame number of the receiving SOF	Frame number of the received SOF	R
b7~b4	Reserved	–	The reset value must be maintained.	R
b3	EERR	Indeterminate error	<p>Erratic error</p> <p>The module will position 1 to report any indeterminate errors.</p> <p>Due to an indeterminate error, the USBFS controller enters a pending state and generates an interrupt with the early pending bit of the USBFS_GINTSTS register (ESUSP bit in USBFS_GINTSTS). If the early hang interrupt is triggered by an indeterminate error, the application can only perform a soft disconnect to resume communication.</p>	R
b2~b1	ENUMSPD	Enumeration speed	<p>Enumerated speed</p> <p>Indicates the speed that the USBFS controller is enumerated to after detecting the speed by chirp sequence. 01: Reserved</p> <p>10: Reserved</p> <p>11: Full speed (PHY clock runs at 48 MHz)</p> <p>Other values: Reserved</p>	R
b0	SUSPSTS	Suspended status	<p>Suspend status</p> <p>In device mode, this bit is set to 1 whenever a pending state is detected on the USB, and the module enters the pending state when the idle state on the USB bus is held for 3ms. The module will exit the pending state when:</p> <ul style="list-style-type: none"> – USB data cable with activity on it – The application performs a write operation to the Remote Wakeup Signal bit of the USBFS_DCTL register (the RWUSIG bit in USBFS_DCTL). 	R

31.7.4.4 USBFS Device IN endpoint common interrupt mask

register (USBFS_DIEPMSK) USBFS Device IN endpoint common

interrupt mask register Offset address: 0x810

Reset value: 0x0000 0000

This register is used in conjunction with the individual USBFS_DIEPINTx registers for all endpoints to generate interrupts on each IN endpoint. The IN endpoint interrupts in the USBFS_DIEPINTx register can be masked by performing a write operation to the corresponding bit in this register. By default, status interrupts are masked.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved									INE PNE M	INE PNE M	ITT XFE MSK	TOM	Res erv ed	EPD M	XFR CM

position	Marker	Place Name	Function	Reading and writing
b31~b7	Reserved	-	The reset value must be maintained.	R
b6	INEPNEM	IN endpoint NAK valid interrupt mask	IN endpoint NAK effective mask 0: Mask interrupt 1: Enabling interrupts	R/W
b5	INEPNMM	Received IN token interrupt mask when EP does not match when EP does not match	Receive IN token interrupt mask when EP does not match (IN token received with EP mismatch mask) 0: Mask interrupt 1: Enable interrupt	R/W
b4	ITTXFEMSK	IN token interrupt mask received when TxFIFO is empty when TxFIFO is empty	IN token interrupt mask received when TxFIFO is empty (IN token received when TxFIFO empty mask) 0: Mask interrupt 1: Enable interrupt	R/W
b3	TOM	Timeout interrupt mask (non-synchronous endpoint)	Timeout interrupt mask (non-synchronous endpoint) (Timeout condition mask (Non-isochronous endpoints)) 0: Mask interrupt 1: Enable interrupt	R/W
b2	Reserved	-	The reset value must be maintained.	R
b1	EPDM	Endpoint disable interrupt mask	Endpoint disabled interrupt mask 0: Mask interrupt 1: Enable interrupt	R/W
b0	XFRM	Transmission completion interrupt mask	Transfer completed interrupt mask 0: Mask interrupt 1: Enable interrupt	R/W

31.7.4.5 USBFS Device OUT endpoint common interrupt

mask register (**USBFS_DOEPMSK**) USBFS Device OUT

endpoint common interrupt mask register Offset

address: 0x814

Reset value: 0x0000 0000

This register is used in conjunction with the individual USBFS_DOEPINTx registers for all endpoints in order to generate interrupts on each OUT endpoint. The OUT endpoint interrupts in the USBFS_DOEPINTx register can be masked by performing a write operation to the corresponding bit in this register. By default, status interrupts are masked.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
OTE	STU	Res	EPD	XFR											
PDM	PM	erv ed	M	CM											

position	Marker	Place Name	Function	Reading and writing
b31~b5	Reserved	–	The reset value must be maintained.	R
b4	OTEPDM	OUT token interrupt mask received when endpoint is disabled (OUT token received when endpoint disabled mask) is only applicable to control OUT endpoints.	OUT token interrupt mask received when endpoint is disabled (OUT token received when endpoint disabled mask) is only applicable to control OUT endpoints.	R/W
		received when endpoint is disabled	0: Mask interrupt 1: Enable interrupt	
b3	STUPM	SETUP phase completes interrupt masking	The SETUP phase done mask is only available for the control endpoint. 0: Mask interrupt 1: Enable interrupt	R/W
b2	Reserved	–	The reset value must be maintained.	R
b1	EPDM	Endpoint disable interrupt mask	Endpoint disabled interrupt mask 0: Mask interrupt 1: Enable interrupt	R/W
b0	XFRCM	Transmission completion interrupt mask	Transfer completed interrupt mask 0: Mask interrupt 1: Enable interrupt	R/W

31.7.4.6 USBFS Device All Endpoint Interrupt Register (USBFS_DAINT)

USBFS Device OUT **endpoint common interrupt mask**

register offset address: 0x818

Reset value: 0x0000 0000

When a valid event occurs on an endpoint, the USBFS_DAINT register will interrupt the application via the device OUT endpoint interrupt bit or the device IN endpoint interrupt bit (OEPINT or IEPINT bit in USBFS_GINTSTS, respectively) in the USBFS_GINTSTS register. Each endpoint corresponds to one interrupt bit, with a maximum of 16 interrupt bits for both the OUT and IN endpoints. Bidirectional endpoints will use the corresponding IN and OUT interrupt bits.

When the application sets position 1 and clears the corresponding device endpoint x interrupt register (**USBFS_DIEPINTx/USBFS_DOEPINTx**), the corresponding bits in this register will also be set to 1 and cleared to zero.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										OEPINT[5:0]					
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved										IEPINT[5:0]					

position	Marker	Place Name	Function	Reading and writing
b31~b22	Reserved	-	The reset value must be maintained.	R
b21~b16	OEPINT	OUT endpoint interrupt bit	OUT endpoint interrupt bits (OUT endpoint interrupt bits) correspond to one bit per OUT endpoint: OUT endpoint 0 corresponds to bit 16, while OUT endpoint 5 corresponds to bit 21.	R/W
b15~b6	Reserved	-	The reset value must be maintained.	R
b5~b0	IEPINT	IN endpoint interrupt bit	IN endpoint interrupt bits (IN endpoint interrupt bits) correspond to one bit per IN endpoint: IN endpoint 0 corresponds to bit 0, while IN endpoint 5 corresponds to bit 5.	R/W

31.7.4.7 USBFS Device all endpoints interrupt mask register

(USBFS_DAINTMSK) USBFS Device all endpoints interrupt

mask register Offset address: 0x81C

Reset value: 0x0000 0000

The USBFS_DAINTMSK register is used in conjunction with the Device Endpoint Interrupt register to interrupt the application when an event occurs on the device endpoint. However, the USBFS_DAINT register bit corresponding to this interrupt will still be set to 1.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										OEPINTM[5:0]					
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved										IEPINTM[5:0]					

position	Marker	Place Name	Function	Reading and writing
b31~b22	Reserved	–	The reset value must be maintained.	R
b21~b16	OEPINTM	OUT endpoint interrupt mask bit	OUT endpoint interrupt mask bits correspond to one bit per OUT endpoint: OUT endpoint 0 corresponds to bit 16, while OUT endpoint 5 corresponds to bit 21. 0: Mask interrupt 1: Enable interrupt	R/W
b15~b6	Reserved	–	The reset value must be maintained.	R
b5~b0	IEPINTM	IN endpoint interrupt mask bit	IN endpoint interrupt mask bits correspond to one bit per IN endpoint: IN endpoint 0 corresponds to bit 0, while IN endpoint 5 corresponds to bit 5. 0: Mask interrupt 1: Enable interrupt	R/W

31.7.4.8 USBFS device IN endpoint FIFO empty break mask register (USBFS_DIEPEMPMSK)

USBFS Device IN endpoint FIFO empty interrupt mask

register offset address: 0x834

Reset value: 0x0000 0000

This register is used to control the generation of the IN endpoint FIFO null break.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved										INEPTXFEM[5:0]					

position	Marker	Place Name	Function	Reading and writing
b31~b6	Reserved	-	The reset value must be maintained.	R
b5~b0	INEPTXFEM	IN EP TxFIFO Air break shield position	IN EP TxFIFO Air Break Mask Bit (IN EP Tx FIFO empty interrupt mask bits) These bits are used as mask bits for USBFS_DIEPINTx. Each bit corresponds to an IN endpoint TXFE interrupt: IN endpoint 0 corresponds to bit 0, while IN endpoint 5 corresponds to bit 5 0: Mask interrupt 1: Enable interrupt	R/W

31.7.4.9 USBFS Device Control IN Endpoint 0 Control Register (USBFS_DIEPCTL0)

USBFS Device control IN endpoint 0

control register offset address: 0x900

Reset value: 0x0000 8000

This register is used to control the control transmission endpoint 0.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPE NA	EPD IS	Reserved		SNA K	CNA K		TXFNUM[3:0]		STA LL	Res erv ed	EPTYP[1:0] 1		NAK STS	Res erv ed	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
USB AEP															MPSIZ[1:0]

position	Marker	Place Name	Function	Reading and writing
b31	EPENA	Endpoint Enable	Endpoint enable The application takes this position 1 to initiate data sending on endpoint 0. The module will clear this bit before triggering any of the following interrupts on this endpoint: – Endpoint Prohibition – Transmission completed	R/W
b30	EPDIS	Endpoint Prohibition	Endpoint disable An application can place this position 1 to stop data transmission on an endpoint even before the transmission on that endpoint is complete. The application must wait until an endpoint forbidden interrupt occurs before the endpoint can be considered forbidden. The module clears this bit before endpoint forbidden interrupt location 1. The endpoint is enabled only if the endpoint Position 1 before the application can place the position 1.	R/W
b29~b28	Reserved	–	The reset value must be maintained.	R
b27	SNAK	Set the NAK bit	Set NAK bit (Set NAK) A write operation to this bit will set the NAK position 1 of the endpoint. This bit allows the application to control the sending of the NAK handshake signal on the endpoint. The module can also set this position 1 on an endpoint after the endpoint receives a SETUP packet.	R/W
b26	CNAK	Clear the NAK bit	Clear NAK bit (Clear NAK) A write operation to this bit will clear the NAK bit of the endpoint.	R/W
b25~b22	TXFNUM	TxFIFO number	TxFIFO number (TxFIFO number) This value is set to the FIFO number assigned to IN endpoint 0. Only TX-FIFO0 can be used.	R/W
b21	STALL	STALL handshake	STALL handshake The application can only set this bit to 1. The module clears this bit when the SETUP token is received by the endpoint. If the NAK bit, global IN NAK, or global OUT NAK and this bit are all set to 1, the STALL bit takes precedence.	R/W

b20	Reserved	-	The reset value must be maintained.	R
b19~b18	EPTYP	Endpoint Type	Endpoint type The hardware is set to '00' to indicate the endpoint of the control type.	R

b17	NAKSTS	NAK Status	The NAK status (NAK status) indicates the following results: 0: The module replies with a non-NAK handshake based on the FIFO status. 1: The module replies to the NAK handshake on this endpoint. When this bit is set to 1 (either by the application or by the module), it stops sending data even if there is still data available in the TxFIFO. Regardless of this bit setting, the module always responds to SETUP packets via the ACK handshake.	R
b16	Reserved	-	The reset value must be maintained.	R
b15	USBAEP	USB Active Endpoint	USB active endpoint (USB active endpoint) This bit is always set to 1, indicating that control endpoint 0 is always active in all configurations and interfaces.	R
b14~b2	Reserved	-	The reset value must be maintained.	R
b1~b0	MPSIZ	Maximum packet size	Maximum packet size The application must program this field to be the maximum packet size for the current logical endpoint. 00: 64 bytes 01: 32 bytes 10: 16 bytes 11: 8 bytes	R/W

31.7.4.10 USBFS Device IN Endpoint x Control Register (USBFS_DIEPCTLx) (x=1..5)

USBFS Device IN endpoint x control register

offset address: 0x900 + (endpoint number × 0x20)

Reset value: 0x0000 0080

The application uses this register to control the behavior of each logical endpoint (other than endpoint 0).

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPE NA	EPD IS	SOD DFR M	SD0 PID / SEV NFR M	SNAK	CNA K	TXFNUM[3:0]				STA LL	Res erv ed	EPTYP[1:0] 1	NAK STS	EON UM/ DPI D	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
USB AEP	Reserved				MPSIZ [10:0]										

position	Marker	Place Name	Function	Reading and writing
b31	EPENA	Endpoint Enable	Endpoint enable The application takes this position 1 to initiate data sending on the endpoint. The module will clear this bit before triggering any of the following interrupts on this endpoint: – SETUP phase completed – Endpoint Prohibition – Transmission completed	R/W
b30	EPDIS	Endpoint Prohibition	Endpoint disable An application can place this position 1 to stop data transmission on an endpoint even before the transmission on that endpoint is complete. The application must wait until an endpoint forbidden interrupt occurs before the endpoint can be considered forbidden. The module clears this bit before endpoint forbidden interrupt location 1. The application can only set this bit to position 1 after endpoint enable position 1 for this endpoint.	R/W
b29	SODDFRM	Set odd number of frames	Set odd frame is only available for synchronizing IN and OUT endpoints. A write operation to this field sets the even/odd frame (EONUM) field to an odd frame.	R/W

b28	SDO PID/ SEVNFRM	Set DATA0 PID/ SEVNFRM	Set DATA0 PID (Set DATA0 PID) Only applies to interrupt/batch IN endpoints. A write operation to this field sets the endpoint data PID (DPID) field in this register to DATA0.	R/W
			SEVNFRM: Set even frame is only available for synchronous IN endpoints. A write operation to this field sets the even/odd frame (EONUM) field to an even frame.	
b27	SNAK	Set the NAK bit	Set NAK bit (Set NAK) A write operation to this bit will set the NAK of the endpoint to position 1. This bit allows the application to control the sending of the NAK handshake signal on the endpoint. The module can also set this position of the OUT endpoint when a transmission completion interrupt occurs or after a SETUP is received on the endpoint 1	R/W
b26	CNAK	Clear the NAK bit	Clear NAK bit (Clear NAK) A write operation to this bit will clear the NAK bit of the endpoint.	R/W
b25~b22	TXFNUM	TxFIFO number	TxFIFO number (TxFIFO number) These bits are used to specify the FIFO number associated with this endpoint. A separate FIFO number must be set for each valid IN endpoint. This field is valid only for IN endpoints.	R/W
b21	STALL	STALL handshake	STALL handshake The application puts this position 1 so that the device replies STALL to all tokens from the USB host. e.g. If the NAK bit, global IN NAK, or global OUT NAK is set to 1 at the same time as this bit, the STALL bit takes precedence. Only the application can clear this bit, not the module.	R/W
b20	Reserved	-	The reset value must be maintained.	R
b19~b18	EPTYP	Endpoint Type	Endpoint type The following are the types of transmissions supported by this logical endpoint. 00: Control 01: Synchronization 10: Batch 11: Interruption	R
b17	NAKSTS	NAK Status	The NAK status (NAK status) indicates the following results: 0: The module replies with a non-NAK handshake based on the FIFO status. 1: The module replies to the NAK handshake on this endpoint. When the application or module places this position 1: For non-synchronous IN endpoints: the module will stop sending any data through the IN endpoint even if there is available data in the TxFIFO. For synchronous IN endpoints: the module sends packets of zero length	R

even if there is available data in the TxFIFO.

Regardless of this bit setting, the module always responds to SETUP packets via the ACK handshake.

b16	EONUM/ DPID	Even/odd frames/endpoint data PID	Even/odd frame is only available for synchronous IN endpoints. Indicates the number of the frame in which the module sends/receives synchronized data for this endpoint. The application must program the even/odd frame numbering through the SEVNFRM and SODDFRM fields in this register for this endpoint to send/receive synchronized data. 0: Even frames 1: Odd number of frames	R/W
			DPID: Endpoint data PID (Endpoint data PID) is only applicable to interrupt/batch IN endpoints. Once the endpoint is activated, the application must program the PID of the first packet to be received or sent on this endpoint. The application programs the DATA0 or DATA1 PID using the SD0PID register field. 0: DATA0 1: DATA1	
b15	USBAEP	USB Active Endpoint	USB active endpoint (USB active endpoint) Indicates whether this endpoint is active in the current configuration and interface. When a USB reset is detected, the module clears this bit for all endpoints (except endpoint 0). After receiving the SetConfiguration and SetInterface commands, the application must program the endpoint registers accordingly and set this Position 1.	R/W
b14~b11	Reserved	-	The reset value must be maintained.	R/W

b10~b0	MPSIZ	Maximum packet size	Maximum packet size
			The application must program this field to be the maximum packet size for the current logical endpoint. This value is in bytes.

31.7.4.11 USBFS device IN endpoint x interrupt register (USBFS_DIEPINTx) (x=0..5)

USBFS Device IN endpoint x interrupt register

offset address: 0x908 + (endpoint number × 0x20)

Reset value: 0x0000 0000

This register indicates the status of the endpoint in the event of USB and AHB related events. When the IN endpoint interrupt bit in the module interrupt register

(The application must read this register when the IEPINT bit (in USBFS_GINTSTS) is set to 1. Before the application can read this register, the Device All Endpoint Interrupt (USBFS_DAINT) register must be read to obtain the exact endpoint number of the Device Endpoint x Interrupt register. The application must clear the corresponding bits in this register before it can clear the corresponding bits in the USBFS_DAINT and USBFS_GINTSTS registers.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								TXFE	INEPNE	Reserved	TTXFE	TOC	Reserved	EPDISD	XFRIC

position	Marker	Place Name	Function	Reading and writing
b31~b8	Reserved	-	The reset value must be maintained.	R
b7	TXFE	Send FIFO is empty	Transmit FIFO empty This interrupt is set when the TxFIFO at this endpoint is half-empty or fully empty. the TxFIFO half-empty or fully empty state is determined by the TxFIFO blank bit in the USBFS_GAHBCFG register (TXFELVL bit in USBFS_GAHBCFG) is determined.	R
b6	INEPNE	IN endpoint NAK valid	INEPNE: IN endpoint NAK effective (IN endpoint NAK effective) This bit can be cleared when the application clears the IN endpoint NAK by writing data to the CNAK bit in USBFS_DIEPCTLx. This interrupt indicates that the module has sampled the NAK set to 1 (by the application or module) and the result is in effect. This interrupt indicates that the IN endpoint NAK bit set to 1 by the application is now active in the module. This interrupt does not guarantee that the NAK handshake signal is sent on the USB. The STALL bit has a higher priority than the NAK bit. A software write of 1 also clears this bit.	R/W
b5	Reserved	-	The reset value must be maintained.	R

b4	TTXFE	IN token received when TxFIFO is empty	IN token received when TxFIFO is empty (IN token received when TxFIFO is empty) is only available for non-periodic IN endpoints. When the TxFIFO (periodic/non-periodic) corresponding to this endpoint is empty, an IN token is received and thus an interrupt is generated. Zeroed by software write 1.	R/W
b3	TOC	Timeout	The Timeout condition is only applicable to the control IN endpoint. Indicates that this endpoint has timed out in response to the most recently received IN token. Zeroed by software write 1.	R/W
b2	Reserved	-	The reset value must be maintained.	R

b1	EPDISD	Endpoint disable interrupt	Endpoint disabled interrupt This bit indicates that the endpoint has been disabled by the application. Zeroed by software write 1.	R/W
b0	XFRC	Transmission completion interruption	Transfer completed interrupt This field indicates that the transfer set on this endpoint has been completed on the USB and AHB. It is cleared by software write 1.	R/W

31.7.4.12 USBFS Device IN Endpoint 0 Transfer Size Register (USBFS_DIEPTSIZE0)

USBFS Device IN endpoint 0 transfer size

register offset address: 0x910

Reset value: 0x0000 0000

The application must modify this register before enabling endpoint 0. Endpoint enable bits in the Control Register via Device Control Endpoint 0

(The module modifies this register after enabling endpoint 0 (EPENA in USBFS_DIEPCTL0). The application can read this register only after the module has cleared the endpoint enable bit to zero.

Non-zero endpoints use the registers of endpoints 1~5.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										PKTCNT[1:0]	Reserved				
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								XFRSIZ[6:0]							

position	Marker	Place Name	Function	Reading and writing
b31~b21	Reserved	-	The reset value must be maintained.	R
b20~b19	PKTCNT	Packet Counting	Packet count Indicates the number of packets contained in a single data transmission for endpoint 0. This field is decremented each time a packet (maximum or short packet) is read from the TxFIFO.	R/W
b18~b7	Reserved	-	The reset value must be maintained.	R
b6~b0	XFRSIZ	Transfer size	Transfer size Indicates the amount of data, in bytes, contained in a single data transfer for endpoint 0. The module interrupts the application only when the application has finished transmitting this data. The transfer size can be set to the maximum packet size of the endpoint to interrupt at the end of each packet. The module decrements this field each time a packet from external memory is written to the TxFIFO.	R/W

31.7.4.13 USBFS Device IN Endpoint x Transfer Size Register (USBFS_DIEPTSI_x) (x=1..5)

USBFS Device IN endpoint x transfer size

register offset address: 0x910 + (endpoint number × 0x20)

Reset value: 0x0000 0000

The application must modify this register before enabling this endpoint. Via the endpoint enable bit in **t h e** USBFS_DIEPCTL_x register

(EPENA bit in USBFS_DIEPCTL_x) After enabling this endpoint, the module modifies this register.

The application can read this register only after the module has cleared the endpoint enable bit.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved				PKTCNT [9:0]								XFRSIZ[18:16]			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
XFRSIZ[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b29	Reserved	-	The reset value must be maintained.	R
b28~b19	PKTCNT	Packet Counting	Packet count Indicates the number of packets contained in a single data transmission on this endpoint. This field will be decremented each time a packet (maximum size or short packet) is read from the TxFIFO.	R/W
b18~b0	XFRSIZ	Transfer size	Transfer size This field contains the amount of data, in bytes, contained in one data transfer for the current endpoint. The module interrupts the application only when the application has finished transmitting this data. The transfer size can be set to the maximum packet size of the endpoint to interrupt at the end of each packet. The module decrements this field each time a packet from external memory is written to the TxFIFO.	R/W

31.7.4.14 USBFS Device IN Endpoint x DMA Address Register (USBFS_DIEPDMAx) (x=0..5)

USBFS Device IN endpoint x transfer size

register offset address: 0x914 + (endpoint number × 0x20)

Reset value: 0x0000 0000

This register is used to set the DMA address when the device endpoint is in DMA mode.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DMAADDR[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DMAADDR[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b0	DMAADDR	DMA Address	DMA address (DMA address) This bit contains the starting address of the external memory area when using DMA for data storage on the endpoint. Note: For control endpoints, the memory area pointed to by this field is also used to store control OUT packets as well as SETUP transaction data packets. When more than three SETUP packets are received in a row, the SETUP packets in memory are overwritten. This register is incremented each time an AHB transfer is made. The application must set a double-word aligned address.	R/W

31.7.4.15 USBFS Device IN Endpoint Send FIFO Status Register (USBFS_DTXFSTSx) (x=0..5)

USBFS Device IN endpoint transmit FIFO status

register offset address: 0x918 + (Endpoint number × 0x20)

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
INEPTFSAV[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b16	Reserved	-	The reset value must be maintained.	R
b15~b0	INEPTFSAV	IN endpoint TxFIFO free space	IN endpoint TxFIFO space available Indicates the amount of free space available in the endpoint TxFIFO. In 32-bit words: 0x0: Endpoint TxFIFO is full 0x1: 1 word available 0x2: 2 words available 0xn: n words available	R

31.7.4.16 USBFS Device Control OUT Endpoint 0 Control Register (USBFS_DOEPCTL0)

USBFS Device control OUT endpoint 0

control register offset address: 0xB00

Reset value: 0x0000 8000

This register is used to control the control transmission endpoint 0.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPE NA	EPD IS	Reserved		SNA K	CNA K	Reserved			STA LL	SNP M	EPTYP[1:0] 1		NAK STS	Reserv ed	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
USB AEP	Reserved											MPSIZ[1:0]			

position	Marker	Place Name	Function	Reading and writing
b31	EPENA	Endpoint Enable	Endpoint enable The application takes this position 1 to initiate data reception on endpoint 0. The module will clear this bit before triggering any of the following interrupts on this endpoint: – SETUP phase completed – Endpoint Prohibition – Transmission completed	R/W
b30	EPDIS	Endpoint Prohibition	Endpoint disable The application cannot disable control OUT endpoint 0.	R
b29~b28	Reserved	–	The reset value must be maintained.	R
b27	SNAK	Set the NAK bit	Set NAK bit (Set NAK) A write operation to this bit will set the NAK position 1 of the endpoint. This bit allows the application to control the sending of the NAK handshake signal on the endpoint. The module can also set this position 1 on an endpoint after the endpoint receives a SETUP packet.	R/W
b26	CNAK	Clear the NAK bit	Clear NAK bit (Clear NAK) A write operation to this bit will clear the NAK bit of the endpoint.	R/W
b25~b22	Reserved	–	The reset value must be maintained.	R
b21	STALL	STALL handshake	STALL handshake When this endpoint receives a SETUP token, the application can only set this bit to 1, and the module will clear it to zero. If the NAK bit, global OUT NAK and this bit are set to 1 at the same time, the STALL bit takes precedence. Regardless of this bit being as Whenever set, the module always responds to SETUP packets via the ACK handshake.	R/W
b20	SNPM	Listening mode	Snoop mode This bit is used to configure the endpoint to listen mode. In listening mode, the module does not check that the OUT packet is correct before transferring it to the application store.	R/W

b19~b18	EPTYP	Endpoint Type	Endpoint type	R/W
The hardware is set to '00' to indicate the endpoint of the control type.				

b17	NAKSTS	NAK Status	The NAK status (NAK status) indicates the following results: 0: The module replies with a non-NAK handshake based on the FIFO status. 1: The module replies to the NAK handshake on this endpoint. When the application or module sets this bit to 1, the module stops receiving data even if space exists in the RxFIFO to continue to accommodate the received packet. Regardless of this bit setting, the module always responds to SETUP packets via the ACK handshake.	R
b16	Reserved	-	The reset value must be maintained.	R
b15	USBAEP	USB Active Endpoint	USB active endpoint (USB active endpoint) This bit is always set to 1, indicating that control endpoint 0 is always active in all configurations and interfaces.	R
b14~b2	Reserved	-	The reset value must be maintained.	R
b1~b0	MPSIZ	Maximum packet size	Maximum packet size The maximum packet size for control OUT endpoint 0 is the same as the value programmed in control IN endpoint 0. 00: 64 bytes 01:32 bytes 10:16 Bytes 11: 8 bytes	R/W

31.7.4.17 USBFS device OUT endpoint x control register (USBFS_DOEPCtlx) (x=1..5)

USBFS Device OUT **endpoint x**

control register offset address: 0xB00 +

(endpoint number × 0x20)

Reset value: 0x0000 0000

The application uses this register to control the behavior of each logical endpoint (other than endpoint 0).

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPE NA	EPD IS	SOD DFR M/ SD1 PID	SD0 PID / SEV NFR M	SNA K	CNA K	Reserved				STA LL	SNP M	EPTYP[1:0]]		NAK STS	EON UM/ DPI D
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
USB AEP	Reserved				MPSIZ [10:0]										

position	Marker	Place Name	Function	Reading and writing
b31	EPENA	Endpoint Enable	Endpoint enable Software set, USBFS clear 0: Endpoint de- enable 1: Endpoint enable	R/W
b30	EPDIS	Endpoint Prohibition	Endpoint disable An application can set this position 1 to stop data sending/receiving on an endpoint even before the transmission on that endpoint is complete. The application must wait until an endpoint disable interrupt occurs before the endpoint can be considered a disable endpoint. The module will clear this bit before endpoint forbidden interrupt position 1. The application can only set this bit to position 1 after endpoint enable position 1 for this endpoint.	R/W
b29	SD1P ID/ SODDFRM	Set DATA1 PID/setting odd frames	Set DATA1 PID (Set DATA1 PID) For interrupt/batch OUT endpoints only. A write operation to this field sets the endpoint data PID (DPID) field in this register to DATA1. SODDFRM: Set odd frame Only applies to synchronous OUT endpoints. A write operation to this field sets the even/odd frame (EONUM) field to an odd frame.	R

b28	SDO	Set DATA0	Set DATA0 PID (Set DATA0 PID)	R
	PID/	PID/	For interrupt/batch OUT endpoints only.	
	SEVNFRM	SEVNFRM	A write operation to this field sets the endpoint data PID (DPID) field in this register to DATA0.	
			SEVNFRM: Set even frame is only available for synchronous OUT endpoints.	
			A write operation to this field sets the even/odd frame (EONUM) field to an even frame.	
b27	SNAK	Set the NAK bit	Set NAK bit (Set NAK)	R/W
			A write operation to this bit will set the NAK position 1 of the endpoint. This bit allows the application to control the sending of the NAK handshake signal on the endpoint. The module can also set this position 1 of the OUT endpoint when a transmission completion interrupt occurs or after a SETUP is received on the endpoint.	
b26	CNAK	Clear the NAK bit	Clear NAK bit (Clear NAK)	R/W
			A write operation to this bit will clear the NAK bit of the endpoint.	
b25~b22	Reserved	-	The reset value must be maintained.	R
b21	STALL	STALL handshake	STALL handshake When this endpoint receives a SETUP token, the application can only set this bit to 1, and the module will clear it to zero. If the NAK bit, global OUT NAK and this bit are set to 1 at the same time, the STALL bit takes precedence. Only the application can clear this bit to zero, while the module cannot.	R/W
b20	SNPM	Listening mode	Snoop mode This bit is used to configure the endpoint to listen mode. In listening mode, the module will no longer check the correctness of the received data.	R/W
b19~b18	EPTYP	Endpoint Type	Endpoint type The following are the types of transmissions supported by this logical endpoint. 00: Control 01: Synchronization 10: Batch 11: Interruption	R/W
b17	NAKSTS	NAK Status	The NAK status (NAK status) indicates the following results: 0: The module replies with a non-NAK handshake based on the FIFO status. 1: The module replies to the NAK handshake on this endpoint. When the application or module places this position 1: The module stops receiving any data on the OUT endpoint even if space exists in the RxFIFO to accommodate incoming packets. Regardless of this bit setting, the module always responds to SETUP packets via the ACK handshake.	R

b16	EONUM/ DPID	Even/odd frames/endpoint data PID	Even/odd frame is only available for synchronous OUT endpoints. Indicates the number of the frame in which the module sends/receives synchronized data for this endpoint. The application must program the even/odd frame numbering via the SEVNFRM and SODDFRM fields in this register for this endpoint to send/receive synchronized Data. 0: even frames 1: Odd number of frames	R/W
DPID: Endpoint data PID (Endpoint data PID) is only applicable to interrupt/batch OUT endpoints. Once the endpoint is activated, the application must program the PID of the first packet to be received or sent on this endpoint. The application programs the DATA0 or DATA1 PID using the SD0PID register field. 0: DATA0 1: DATA1				

b15	USBAEP	USB Active Endpoint	USB active endpoint (USB active endpoint) Indicates whether this endpoint is active in the current configuration and interface. When a USB reset is detected, the module clears this bit for all endpoints (except endpoint 0). Upon receipt of the SetConfiguration and SetInterface commands, the application must program the endpoint registers accordingly and set this bit to 1.	R/W
b14~b11	Reserved	-	The reset value must be maintained.	R/W
b10~b0	MPSIZ	Maximum packet size	Maximum packet size The application must program this field to be the maximum packet size for the current logical endpoint. This value is in bytes.	R/W

31.7.4.18 USBFS device OUT endpoint x interrupt register (USBFS_DOEPINTx) (x=0..5)

USBFS Device OUT **endpoint x interrupt**

register offset address: 0xb08 + (endpoint number ×

0x20)

Reset value: 0x0000 0080

This register indicates the status of the endpoint in the event of a USB and AHB related event. The application must read this register when the OUT endpoint interrupt bit in the USBFS_GINTSTS register (the OEPINT bit in USBFS_GINTSTS) is set to 1. Before the application can read this register, the USBFS_DAINT register must be read to obtain the exact endpoint number of the USBFS_DOEPINTx register. The application must clear the corresponding bit in this register before it can clear the corresponding bits in the USBFS_DAINT and USBFS_GINTSTS registers.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								B2B STUP	Res erv ed	OTE PDI S	STU P	Res erv ed	EPD ISD	XFR C	

position	Marker	Place Name	Function	Reading and writing
b31~b7	Reserved	–	The reset value must be maintained.	R
b6	B2BSTUP	Received continuous SETUP packets	Back-to-back SETUP packets received Applies to the control OUT endpoint only. This bit indicates that the endpoint has received more than three consecutive SETUP counts According to the package. Software write 1 also clears this bit.	R/W
b5	Reserved	–	The reset value must be maintained.	R
b4	OTEPDIS	OUT token received when endpoint is disabled	OUT token received when endpoint disabled Applies to control OUT endpoints only. Indicates that the OUT token is received while the endpoint is not yet enabled, thereby generating a mid Break. Zeroed by software write 1.	R/W
b3	STUP	SETUP phase completed	SETUP phase done Applies to the control OUT endpoint only. Indicates that the SETUP phase of the control endpoint is complete and that no more consecutive SETUP packets are being received on the current control transmission. On this interrupt, the application can decode the received SETUP packets. Zeroed by software write 1.	R/W
b2	Reserved	–	The reset value must be maintained.	R
b1	EPDISD	Endpoint disable interrupt	Endpoint disabled interrupt This bit indicates that the endpoint has been disabled by the application. Zeroed by software write 1.	R/W

b0	XFRC	Transmission completion interruption	Transfer completed interrupt This field indicates that the transfer set on this endpoint has been completed on the USB and AHB. It is cleared by software write 1.	R/W
----	------	--------------------------------------	--	-----

31.7.4.19 USBFS Device OUT Endpoint 0 Transfer Size Register (USBFS_DOEPTSIZ0)

USBFS Device OUT **endpoint 0 transfer size**

register offset address: 0xB10

Reset value: 0x0000 0000

The application must modify this register before enabling endpoint 0. Endpoint enable bits in the Control Register via Device Control Endpoint 0

(The module modifies this register after enabling endpoint 0 (EPENA in USBFS_DIEPCTL0). The application can read this register only after the module has cleared the endpoint enable bit to zero.

Non-zero endpoints use the registers of endpoints 1~5.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Res erv ed	STUPCNT[1: 0]								Reserved				PKT CNT	Reserved	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								XFRSIZ[6:0]							

position	Marker	Place Name	Function	Reading and writing
b31	Reserved	-	The reset value must be maintained.	R
b30~b29	STUPCNT	SETUP packet count	<p>SETUP packet count</p> <p>This field specifies the number of SETUP packets the endpoint can receive consecutively.</p> <p>01: 1 data package</p> <p>10: 2 packets</p> <p>11: 3 data packages</p>	R/W
b28~b20	Reserved	-	The reset value must be maintained.	R
b19	PKTCNT	Packet Counting	<p>Packet count</p> <p>The number of packets that should be received in one transmission.</p> <p>This bit is set by software before the endpoint is enabled, and the number of this field is set whenever a packet is received after the transmission starts</p> <p>The value is automatically reduced.</p>	R/W
b18~b7	Reserved	-	The reset value must be maintained.	R
b6~b0	XFRSIZ	Transfer size	<p>Transfer size</p> <p>Indicates the amount of data, in bytes, contained in a single data transfer for endpoint 0. Only when the application transfers</p> <p>The module will only interrupt the application after this data is finished. The transfer size can be set to the maximum number of endpoints according to the packet size to break at the end of each packet.</p> <p>The module decrements this field each time a packet is read from Rx FIFO and written to external memory.</p>	R/W

31.7.4.20 USBFS Device OUT Endpoint x Transfer Size Register (USBFS_DOEPTSIZx) (x=1..5)

USBFS Device OUT **endpoint x transfer size**

register offset address: 0xB10 + (endpoint number × 0x20)

Reset value: 0x0000 0000

The application must modify this register before enabling this endpoint. Via the endpoint enable bit in **t h e** USBFS_DOEPCTLx register

(EPENA bit in USBFS_DOEPCTLx) After enabling this endpoint, the module modifies this register.

The application can read this register only after the module has cleared the endpoint enable bit.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved			PKTCNT [9:0]				XFRSIZ[18:1 6]				XFRSIZ[15:0]				
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0

position	Marker	Place Name	Function	Reading and writing
b31~b29	Reserved	-	The reset value must be maintained.	R
b28~b19	PKTCNT	Packet Counting	Packet count (Packet count) Indicates the number of packets contained in a single data transmission on this endpoint. This field will be decremented after each packet (maximum size or short packet) written to RxFIFO.	R/W
b18~b0	XFRSIZ	Transfer size	Transfer size This field contains the amount of data, in bytes, contained in one data transfer for the current endpoint. It is only available when the application The module will only interrupt the application after the program has transferred this data. The transfer size can be set to end The maximum packet size of the point to break at the end of each packet. The module decrements this field each time a packet is read from RxFIFO and written to external memory.	R/W

31.7.4.21 USBFS device OUT endpoint x DMA address register (USBFS_DOEPDMA_x) (x=0..5)

USBFS Device OUT **endpoint** x transfer size

register offset address: 0xB14 + (endpoint number × 0x20)

Reset value: 0xFFFF XXXX

This register is used to set the DMA address when the device endpoint is in DMA mode.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DMAADDR[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DMAADDR[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b0	DMAADDR	DMA Address	DMA address (DMA address) This bit contains the starting address of the external memory area when using DMA for data transmission on the endpoint. Note: For control endpoints, the memory area pointed to by this field is also used to store control OUT packets as well as SETUP transaction data packets. When more than three SETUP packets are received in a row, the SETUP packets in memory are overwritten. This register is incremented each time an AHB transfer is made. The application must set a double-word aligned address.	R/W

31.7.5 USBFS Clock Gating Control Register

The HCLK and PHY clocks are controlled through the gated clock control registers to reduce power consumption. Bit values in the register descriptions are expressed in binary unless otherwise noted.

31.7.5.1 USBFS clock gating control register (USBFS_GCCTL)

Offset address: 0xE00

Reset value: 0x0000 0000

This register is available in both host mode and device mode.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
GAT	STP	EHC	PCL	LK	K										

position	Marker	Place Name	Function	Reading and writing
b31~b2	Reserved	–	The reset value must be maintained.	R
b1	GATEHCLK	Gate control HCLK	Gate HCLK (Gate HCLK) When USB communication hangs or a session is invalid, the application sets this bit to 1 to stop clocking modules other than the AHB bus slave, master, and wake-up logic. The application clears this bit when USB communication resumes or a new session is started.	R/W
b0	STPPCLK	Stop PHY clock	Stop PHY clock The application sets this bit to 1 to stop the PHY clock when USB communication hangs, when the session is invalid, or when the device is disconnected. When USB communication resumes, the application clears this bit to zero.	R/W

32 Cryptographic Coprocessing Module (CPM)

32.1 Introduction

The Cryptographic Co-Processing Module (CPM) consists of three sub-modules: AES Encryption and Decryption Algorithm Processor, HASH Secure Hash Algorithm, and TRNG True Random Number Generator.

The AES encryption and decryption algorithm processor follows the standard data encryption and decryption standard and can implement 128-bit key length encryption and decryption operations.

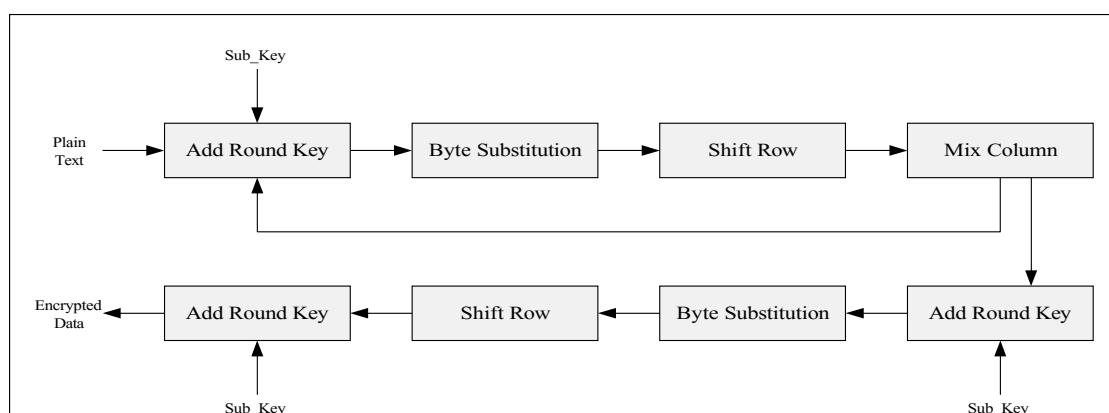
The HASH secure hash algorithm is the **SHA-2** version of the SHA-256 (Secure Hash Algorithm) which complies with the national standard "FIPS PUB 180-3" published by the National Bureau of Standards and Technology, and can generate 256-bit message digest output for messages up to 2^{64} bits in length. message digest output for messages up to 2^{64} bits in length.

TRNG True Random Number Generator is a random number generator based on continuous analog noise, providing **64bit** random numbers.

32.2 Encryption and decryption algorithm processor (AES)

32.2.1 Algorithm Introduction

The AES encryption algorithm is a key iterative grouping cipher that incorporates the repetitive effect of round transformations on the state. The round transformation of the encryption process consists of four operations: **SubBytes**, **ShiftRows**, **MixColumns**, and **AddRoundKey**, where **SubBytes** is the modulo inverse of each byte in GF(2^8) and an affine transformation; **ShiftRows** is a byte permutation, which shifts the rows in the state by different offsets. It shifts the rows in the state according to different offsets; **MixColumns** performs a linear transformation on the columns of the state; **AddRoundKey** performs a bit-by-bit dissimilarity operation on the bytes in the state and the round key. Figure 32-1 below shows a diagram of the encryption process:



**Figure 32-1 AES encryption
flow diagram**

In the above figure, Sub_Key refers to the subkey of each round. Except for the initial key used for the initial transformation, the subkey used for the subsequent rounds of transformation needs to be extended from the initial key, and the key extension process and the encryption process are carried out simultaneously.

Since the plaintext is fixed at 128bits, the number of rounds the encryption process runs depends on the length of the key. For example, if the key is 128bits, the number of rounds is 10; if the key is 192bits, the number of rounds is 12; if the key is 256bits, the number of rounds is 14

rounds. All rounds perform a complete round transformation except for the last round which lacks a MixColumn transformation. This product is equipped with AES which supports only **128bits** key length for the encryption and decryption process.

The decryption process is different from the encryption process, firstly, all the key extensions must be completed, and the decryption process starts from the last round of key extensions backward, then the four operations of the round transformation become the corresponding inverse operations: InvSubBytes, InvShiftRows, InvMixColumns, AddRoundKey. The modulo inverse operation in InvSubBytes remains, but the affine transform is changed to an inverse transform; InvShiftRows and InvMixColumns become the corresponding inverse transforms; AddRoundKey remains the same.

32.2.2 Encryption operation process

An example of the AES encryption operation is as follows:

- 1) Writes 128bits of plaintext into the data register (AES_DR).
- 2) Write the encryption key into the key register (AES_KR).
- 3) Set the bits in the control register (AES_CR), including:
 - a) Set CR.Mode to 0
 - b) Write 1 to AES_CR.START to start the module

for arithmetic Note: Steps a, b can be done simultaneously
- 4) Determines if the module operation is finished.
START is read continuously, and if its value becomes 0, the operation is finished
- 5) Read the data register (AES_DR) and get 128-bit cipher text.
- 6) If you want to continue with a new operation, go back to step 1, otherwise finish.

32.2.3 Decryption operation process

An example of the AES decryption operation is as follows:

- 1) Write 128bits of cipher text into the data register (AES_DR).
- 2) Write the decryption key into the key register (AES_KR).
- 3) Set the bits in the control register (AES_CR), including:
 - a) Set CR.Mode to 1
 - b) Write 1 to AES_CR.START to start the module

for arithmetic Note: Steps a, b can be done simultaneously
- 4) Determines if the module operation is finished.
START is read continuously, and if its value becomes 0, the operation is finished

-
- 5) Read the data register (AES_DR) to obtain 128-bit plaintext.
 - 6) If you want to continue with a new operation, go back to step 1, otherwise finish.

32.2.4 Encryption and decryption time

The time required by the AES module from the start of an operation (AES_CR.START write 1) to the end of that operation (AES_CR.START restore to 0) is as follows:

Encryption process	440 CPU clock cycles
Decryption process	580 CPU clock cycles

32.2.5 Operation Notes

- 1) After power-up, the module performs an asynchronous reset operation. The clock needs to be stable and valid until the reset is disengaged and continues to be stable during subsequent operation.
- 2) During the encryption and decryption process, the data register is changed, and if the data of the next operation is the result of the current operation, there is no need to rewrite the data.
- 3) In the case of encrypting and decrypting a large amount of data with the same key, there is no need to write the key repeatedly.
- 4) To determine the end of a module operation: read AES_CR_START continuously and if its value becomes 0, the operation is finished.

32.2.6 Register Description

BASE ADDR: 0x4000_8000

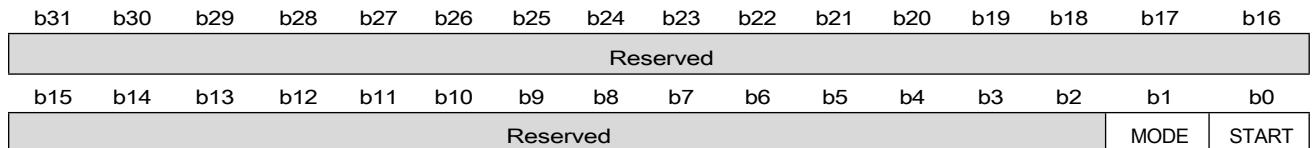
Register Name	Sym bols	Offset	Bit wid th	Reset value
AES Control Register	AES_CR	0x0000	32	0x0000_0000
AES Data Register 0	AES_DR0	0x0010	32	0x0000_0000
AES Data Register 1	AES_DR1	0x0014	32	0x0000_0000
AES Data Register 2	AES_DR2	0x0018	32	0x0000_0000
AES Data Register 3	AES_DR3	0x001C	32	0x0000_0000
AES key register 0	AES_KR0	0x0020	32	0x0000_0000
AES key register 1	AES_KR1	0x0024	32	0x0000_0000
AES Key Register 2	AES_KR2	0x0028	32	0x0000_0000
AES key register 3	AES_KR3	0x002C	32	0x0000_0000

Caution:

-The write operation to all registers is only valid when the module is idle (AES_CR.START=0), otherwise the write operation is ignored; the read operation to registers other than the control register (AES_CR) is only valid when the module is idle (AES_CR.START=0) to read out the valid data, otherwise read out the unknown data; the read operation to the control register (AES_CR) can be performed at any time, all can read out the valid data.

32.2.6.1 AES Control Register (AES_CR)

Reset value: 0x0000_0000



position	Marker	Place Name	Function	Reading and writing
b31~b2	Reserved	-	Read "0" for read write "0" for write	R/W
b1	MODE	Function Selection	0: Encryption operation 1: Decryption operations	R/W
b0	START	Start	0: This module is finished or not started 1: Start this module to perform calculations	R/W

32.2.6.2 AES Data Register (AES_DR)

The data registers consist of four 32-bit registers consisting of 128 bits of data, which are used to store the plaintext to be encrypted or the ciphertext to be decrypted before the operation of the

modu	Cryptogr	Decryptio	s
compl	aphic operation	n operation	s

The four 32-bit registers are connected together to form a 128-bit data, and the four registers need

to be operated separately when reading and writing. Number of bits : 128 bits

Offset address : 0x10 - data[31:0]

0x14 - data[63:32]

0x18 - data[95:64]

0x1C - data[127:96]

Reset value : 0x0000_0000 (each 32-bit register)

Data example: 0x00112233445566778899AABBCCDDEEFF

Offset Address	Register Name	Fill in the data
0x10	DR0	0x3322_1100
0x14	DR1	0x7766_5544
0x18	DR2	0xBBAA_9988
0x1C	DR3	0xFFEE_DDCC

32.2.6.3 AES key register (AES_KR)

The key register consists of four 32-bit registers for 128-bit keys, which are

used to store the initial key entered. Number of bits : 128 bits

Offset address : 0x20 - key[31: 0]

0x24 - key[63: 32]

0x28 - key[95: 64]

0x2C - key[127: 96]

Reset value : 0x0000_0000 (per 32-bit register)

Data example: 0x000102030405060708090A0B0C0D0E0F

Offset Address	Register Name	Fill in the data
0x20	KR0	0x0302_0100
0x24	KR1	0x0706_0504
0x28	KR2	0xB0A_0908
0x2C	KR3	0xF0E_0D0C

32.3 Secure Hash Algorithm (HASH)

32.3.1 Algorithm Introduction

The steps of the secure hashing algorithm are as follows:

The message is first padded so that its length is exactly a number that is only 64 bits smaller than a multiple of 512. The padding is done by appending a 1 to the message, followed by as many zeros as required, and then appending a 64-bit message length (before padding) so that the message length is exactly an integer multiple of 512 bits.

Next, eight 32-bit variables, A, B, C, D, E, F, G, and H, are initialized in hexadecimal. Then the main loop of the algorithm starts, processing 512-bit messages at a time, and the number of loops is the number of 512-bit groups in the message.

The main loop performs a total of 64 operations, which are called compression functions. Each operation consists of shifting, circular shifting, logical operations, modulo 2³² addition, etc. The process is shown in Figure 32-2 below, and the final output is cascaded from A, B, C, D, E, F, G, and H. Where W_t is the temporary value used in step t from the 512-bit message, and K_t is the constant value used in step t. The final output is a cascade of A, B, C, D, E, F, G, and H. W_t is the temporary value used in step t obtained from the 512-bit message, K_t is the constant value used in step t, and t (0 ≤ t ≤ 63) is a step in the 64-step loop.

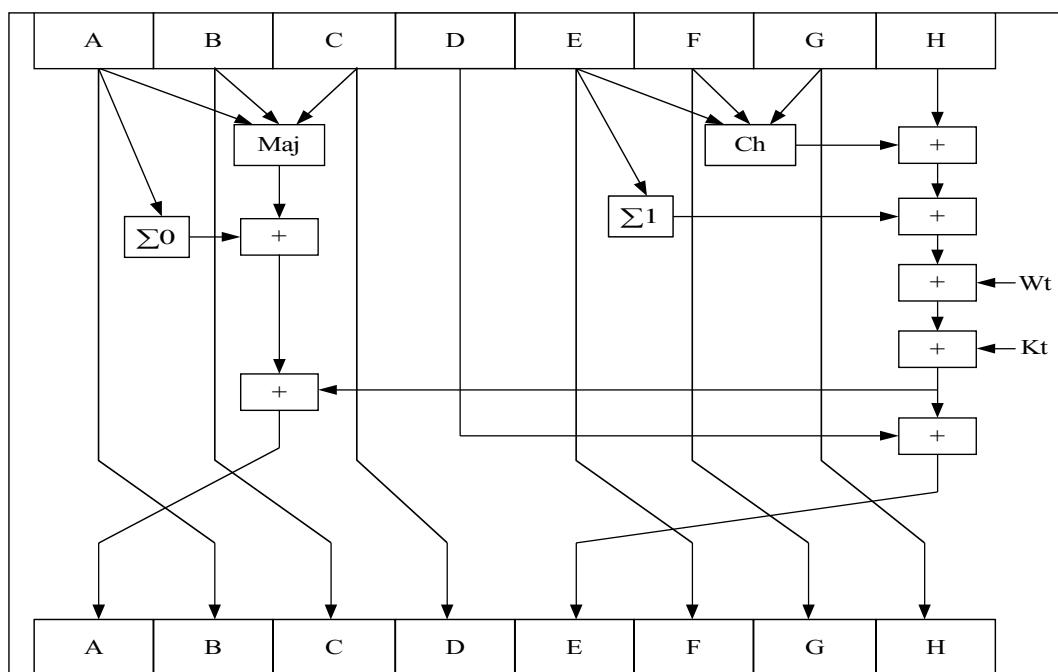


Figure 32-2 HASH
algorithm flow chart

32.3.2 Operation process

After power-up, the module performs an asynchronous reset operation. The clock needs to be stable and valid before the reset is disengaged and continues to be stable in subsequent operations. The operation flow of the HSAH module is as follows:

- 1) The software pads the raw data according to algorithmic rules and groups the padded messages by 512 bits.
- 2) Writes the data being manipulated into the data register (HASH_DR).
- 3) If this operation is the first set of data in the message grouping, write 1 to the HASH_CR . FST_GRP bit to 1.
- 4) Write 1 to HASH_CR.START to start this module to perform the operation. Note: 3 and 4 above can be done simultaneously
- 5) Determine if this module is completed for this operation by the following method:
Keep reading HASH_CR.START until the bit is 0, which means the operation is completed.
- 6) (If this operation is not the last set of data in the message grouping, return to 2)
- 7) If this operation is the last set of data in the message grouping, read the summary register (HASH_HR) to get the result of this operation. If further operations are required, return to step 1.

32.3.3 Message Filling

The padded grouping processing steps for SHA-256 are as follows:

1. Original Message Grouping

The original message is divided into L groups of size 512bit. If $I \% 512 < 448$, then the number of groups L is $I / 512$; if $I \% 512 \geq 448$, then the number of groups L is $I / 512 + 1$.

2. Add Length

① Add fill bit:

Add padding bits at the end of the $I / 512$ th group of messages: a 1 and a number of 0s, the number of 0s can be zero. If $I \% 512 < 448$, the padding makes the length of the data bits satisfy the length $448 \bmod 512$ (the last 64 bits are left as a representation of the original message length); if $I \% 512 \geq 448$, the 512bit block of the $I / 512$ group is filled with a 1 and a number of zeros, and the first 448 bits of the L ($L = I / 512 + 1$) group are filled with zeros.

② Add the original message length:

A **64bit** block representing the original message length as a **64bit** unsigned integer. The original message length is added to **the last 64 bits of the**

Lth group.

An example of the process of populating a grouping is as follows:

1) Filling Example 1:

The original message is the string "ABCDE", and its **ASCII** code is expressed as a binary bit string: "01100001 01100010 01100011 01100100 01100101", and the steps to add the length are as follows:

- A. Add "1". The populated message will be "01100001 01100010
01100011 01100100

01100101 1".

- B. Add "0". Since the original message length is 40 bits, the number of zeros to be added is $512 - 64 - 40 = 407$.

The filled message becomes (in hexadecimal)

61626364 65800000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000

- C. Add the length of the original message. The original message length of 40 is expressed in two **32bit** words (in hexadecimal): 00000000 00000028.

The filled message becomes (in hexadecimal)

61626364 65800000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000028

2) Filling Example 2:

Original	original	News	message	is	word	character
----------	----------	------	---------	----	------	-----------

String "ABCDBCDECDEFDEFGEFGFHIGHIJHIJKIJKLJKLMKLMN

LMNOMNOPNOPQ". Each character can be converted to **8bit** by its

ASCII code, so the length of the message is $1 = 56 * 8 = 448$.

- A. Add "1" and "0". The filled message (in hex) is the first message

block: 61626364 62636465 63646566 64656667
65666768 66676869 6768696A 68696A6B
696a6b6c 6a6b6c6d 6b6c6d6e 6c6d6e6f
6d6e6f70 6e6f7071 80000000 00000000

- B. Add the length of the original message. The original message length of 448 is expressed in two **32bit** words (in hexadecimal): 00000000 000001C0.

The padded message (in hexadecimal) is the second message block:

00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 000001C0

32.3.4 Register Description

BASE ADDR: 0x4000_8400

Register Name	Sym bols	Offset	Bit width	Reset value
HASH Control Register	HASH_CR	0x0000	32	0x0000_0000
HASH Summary Register 7	HASH_HR7	0x0010	32	0x0000_0000
HASH summary register 6	HASH_HR6	0x0014	32	0x0000_0000
.....
HASH summary register 1	HASH_HR1	0x0028	32	0x0000_0000
HASH summary register 0	HASH_HR0	0x002C	32	0x0000_0000
HASH data register 15	HASH_DR15	0x0040	32	0x0000_0000
HASH data register 14	HASH_DR14	0x0044	32	0x0000_0000
.....
HASH data register 1	HASH_DR1	0x0078	32	0x0000_0000
HASH data register 0	HASH_DR0	0x007C	32	0x0000_0000

32.3.4.1 HASH Control Register (HASH_CR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved														FST_GRP	START

position	Marker	Place Name	Function	Reading and writing
b31~b2	Reserved	-	Reads out as "•"	R/W
b1	FST_GRP	The first group of message grouping	0: This operation is not the first group operation of message grouping 1: This operation is the first group of operations for message grouping	R/W
b0	START	Start	0: This module is finished or not started 1: Start this module to perform calculations	R/W

Caution:

- The operation of the START bit is as follows: the module will start running after the software writes 1 to this bit; the hardware will automatically clear this bit to 0 at the end of this run; the software will query this bit to 0 to indicate the completion of this run.
- Write operations to this register can only be performed when the module is not in the computing state (i.e. when the START bit is 0), otherwise the hardware will automatically ignore the write operation. Read operations are not subject to this restriction.

32.3.4.2 HASH Summary Register (HASH_HR)

Number of bits : 256 bits

Offset address : 0x10 - hash[255:224]

- 0x14 - hash[223:192]
- 0x18 - hash[191:160]
- 0x1C - hash[159:128]
- 0x20 - hash[127:96]
- 0x24 - hash[95:64]
- 0x28 - hash[63:32]
- 0x2C - hash[31:0]

Reset value : 0x0000_0000 (each 32-bit register)

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
HASH[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
HASH[15:0]															
<hr/>															
position	Marker	Place Name	Function	Reading and writing											
b31~b0	HASH[31:0]	Summary Value	Get the message summary by reading this register after the module has completed its operation	R/W											

Caution:

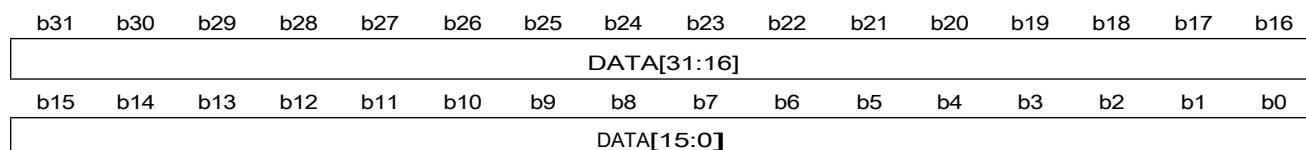
- This register consists of 8 32-bit registers stitched together. The 8 32-bit registers are accessed sequentially, with the low address. The corresponding 32-bit register holds the high word of the message digest.
- The hardware will automatically ignore write operations to this register.
- The read of this register can only be done when this module is not in the operation state (HASH_CR.START=0), otherwise the read of this register will get all zeros.

32.3.4.3 HASH Data Register (HASH_DR)

Digits : 512
 Offset Th bits - **data[511..480]**
 Address e 0x040
 :
 Th
 e
 0x044 - **data[479..448]**
 0x048 - **data[447..416]**
 0x04C - **data[415..384]**

 0x070 - **data[127..96]**
 0x074 - **data[95..64]**
 0x078 - **data[63..32]**
 0x07C - **data[31..0]**

Reset value : 0x0000_0000 (per 32-bit register)



position	Marker	Place Name	Function	Reading and writing
b31~b0	DATA[31:0]	Data Register	Used to write messages before module operations	R/W

Caution:

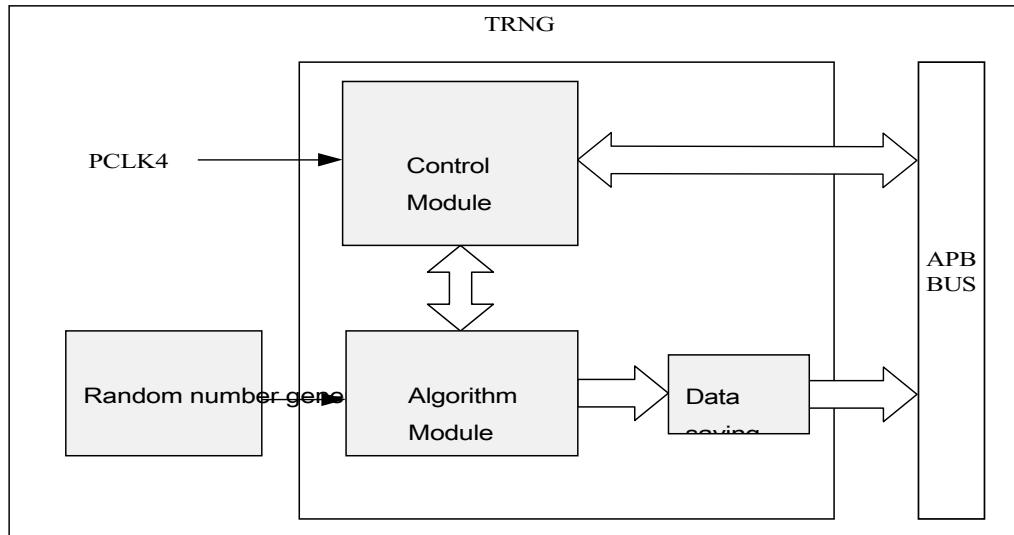
- This register consists of 16 32-bit registers stitched together. The 32-bit register corresponding to the low address holds the high word of the data.
- Writes to this register can only be performed when this module is not in the operation state (HASH_CR.START), otherwise the hardware will automatically ignore the write operation to this register.
- A read of this register will always get all zeros.

32.4 True Random Number Generator (TRNG)

32.4.1 Module Block Diagram

The TRNG module provides 1 true random number generator to generate 1 64-bit random number.

The system block diagram of the TRNG is shown in Figure 32-3 below. The algorithm module captures the random noise and saves the result to the data module and outputs it through the bus; the control module controls the mode and start-up of the TRNG.



**Figure 32-3 TRNG System
Block Diagram**

32.4.2 Operation process

The flow of true random number generation is as follows:

1. Turn on the random number generator circuit (set the EN bit of TRNG_CR to 1)
2. Configure the random number generation mode (set TRNG_MR)
3. Start random number generation (set the RUN bit of TRNG_CR to 1)
4. Read a random number (read TRNG_DR)
5. Turn off the random number generator circuit (set the EN bit of TRNG_CR to 0)

32.4.3 Interrupt and event output

RUN hardware clears and generates a random number generation completion interrupt request (TRNG_END). The end of random number generation also generates an event output, which can trigger other modules to link.

32.4.4 Operation Notes

To get a good random number, set the frequency of the peripheral clock PCLK4 to below 1MHz.

32.4.5 Register Description

BASE ADDR: 0x4004_1000

Register Name	Sym bols	Offset	Bit width	Reset value
TRNG control register	TRNG_CR	0x0000	32	0x0000_0000
TRNG mode register	TRNG_MR	0x0004	32	0x0000_0012
TRNG data register 0	TRNG_DR0	0x000C	32	0x0800_0000
TRNG data register 1	TRNG_DR1	0x0010	32	0x0800_0200

32.4.5.1 TRNG control register (TRNG_CR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved														RUN	EN

position	Marker	Place Name	Function	Reading and writing
b31~b3	Reserved	–	Read "0", write "0" when writing	R
b2	Reserved	–	Read "0", write "0" when writing	R/W
b1	RUN	Start of random number operation	0: Random number operation stops 1: Random number operation starts The software writes "1" generate a new 64-bit random number; after the run is complete, the hardware is cleared to zero.	R/W
b0	EN	Analog oscillator enable	0: Turn off the analog random number generator circuit 1: Open the analog random number generator circuit	R/W

32.4.5.2 TRNG Mode Register (TRNG_MR)

Reset value: 0x0000_0012

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved										CNT[2:0]			-	LOAD	

position	Marker	Place Name	Function	Reading and writing
b31~b5	Reserved	-	Read "0", write "0" when writing	R
b4~2	CNT[2:0]	Shift count control bit	When capturing random noise, the shift count control bit 011: shifted 32 times 100: shifted 64 times 101: Shift 128 times 110: shifted 256 times 000~010, 111: Function reserved bits	R/W
b1	Reserved	-	Read "1", write "1" when writing	R/W
b0	LOAD	Loading control position	Whether the data register is loaded with new initial values from the random number generator before random number generation 0: No new initial values are loaded 1: Load new initial values	R/W

32.4.5.3 TRNG Data Register (TRNG_DR)

Reset value: DR0: 0x0800_0000 DR1:

0x0800_0200

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DATA[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DATA[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31-b0	DATA[31:0]	Random Number	64-bit random number	R

33 Data Calculation Unit (DCU)

33.1 Introduction

The Data Computing **Unit (DCU)** is a module that simply processes data without the help of a CPU. Each DCU unit has 3 data registers, and is capable of adding, subtracting, and comparing the size of 2 data, as well as window comparison functions. This product is equipped with 4 DCU units, each of which can perform its own function independently.

Function Summary:

- 4 types of data processing are possible: addition of 2 data, subtraction, comparison and comparison of 3 data windows.
- The addition and subtraction operations are performed on the data in the DATA0 and DATA1 registers, and the results are stored in DATA0.
- Addition and subtraction can be computed either after writing the register or triggered by other peripheral circuit events.
- The result of addition and subtraction operations is automatically halved once, and the result of the halving operation and the result of the addition and subtraction operations are placed in two data registers for use by other modules.
- The comparison mode can compare 2 data between DATA0 and DATA1 registers, and between DATA0 and DATA2 registers, and can generate interrupts and flags when they are greater than, less than, or equal to, respectively.
- The comparison mode can be used to compare windows, i.e. set DATA1 and DATA2 as the upper and lower limits of the window respectively, and determine whether DATA0 is inside or outside the window based on the comparison results of DATA0 and DATA1 and DATA0 and DATA2.
- The event signals are used to start the peripheral circuits when the DCU is selected as the trigger source by other peripheral circuits with hardware trigger start function.

33.2 Function Description

33.2.1 Additive mode

The addition mode calculates the sum of DATA0 and DATA1, where DATA0 is the summed number and DATA1 is the summed number. Each time the DATA1 register is written, an operation of $(\text{DATA0} + \text{DATA1})/2$ is performed, and the result of $\text{DATA0} + \text{DATA1}$ is stored in DATA0, while (DATA0

+ The result of $\text{DATA0} + \text{DATA1})/2$ is stored in DATA2. When the result of DATA0 + DATA1 exceeds 0xFF (8bit mode) or 0xFFFF (16bit mode) or 0xFFFF_FFFF (32bit mode), a flag bit is generated and an interrupt is generated.

33.2.2 Subtractive mode

The subtraction mode calculates the difference between DATA0 and DATA1, where DATA0 is the subtracted number and DATA1 is the subtracted number. Each time the DATA1 register is written, an operation of $(\text{DATA0}-\text{DATA1})/2$ is performed, and the result of $\text{DATA0}-\text{DATA1}$ is stored in DATA0, while (DATA0

-When the result of $\text{DATA0}-\text{DATA1}$ is less than 0x0 (8bit, 16bit, 32bit mode), a flag bit is generated and an interrupt is generated.

33.2.3 Hardware triggered boot mode

The DCU is capable of triggering start-up operations based on events generated by the peripheral circuits. When using the hardware trigger start mode, it is necessary to set the Enable position of the peripheral circuit trigger function in the Function Clock Control 0 register (FCG0) to be active first. Each DCU unit can independently select the trigger start signal sent from other peripheral circuits. When selecting the start signal, write the number of the peripheral circuit start source to be selected in the trigger source selection register (DCU_TRGSELx). When this peripheral circuit event occurs, the event signal is input to the DCU and triggers the DCU to start the operation. The hardware trigger start mode includes trigger plus mode and trigger minus mode. In the trigger plus mode, the DCU will start and perform the operation of **(DATA0 + DATA1)/2 for each event** trigger, and the result of $\text{DATA0} + \text{DATA1}$ is stored in DATA0, while the result of $(\text{DATA0} + \text{DATA1})/2$ is stored in DATA2. When the result of $\text{DATA0} + \text{DATA1}$ exceeds 0xFF (8bit mode) or 0xFFFF (16bit mode) or 0xFFFF_FFFF (32bit mode), a flag bit is generated and an interrupt is generated. In trigger minus mode, the DCU will start and perform an operation of **(DATA0- DATA1)/2 for each event** trigger, and the result of $\text{DATA0}-\text{DATA1}$ is stored in DATA0, while the result of $(\text{DATA0}-\text{DATA1})/2$ is stored in DATA2. When the result of $\text{DATA0}-\text{DATA1}$ is less than 0x0 (8bit, 16bit, 32bit mode), a flag bit is generated and an interrupt is generated.

33.2.4 Comparison Mode

Compare the size of DATA0 and DATA1 and DATA0 and DATA2, optionally when DATA0 is larger than DATA1, DATA0 is smaller than DATA1, DATA0 is equal to DATA1 and when DATA0 is larger than DATA2, DATA0 is smaller than DATA2, DATA0 is equal to

The flag bit is generated on DATA2 and an interrupt is generated. The compare mode allows you to select the conditions for starting the data comparison, write DATA0 and then compare or write any data register and then compare.

33.2.5 Interrupt and event signal output

The DCU has a variety of interrupts and event outputs for triggering the start of other peripheral circuits for the user to select. The interrupt and event outputs are controlled by the interrupt condition selection register (DCU_INTEVTSEL). When an event signal needs to be output, the user needs to set the corresponding control position of the interrupt condition selection register (DCU_INTEVTSEL) to active. Each DCU unit outputs one DCU event signal, which is DCU1/DCU2/DCU3/DCU4 in the event list. position 1. Each DCU unit outputs one DCU interrupt signal in the interrupt list as DCU1/DCU/DCU3/DCU4, respectively.

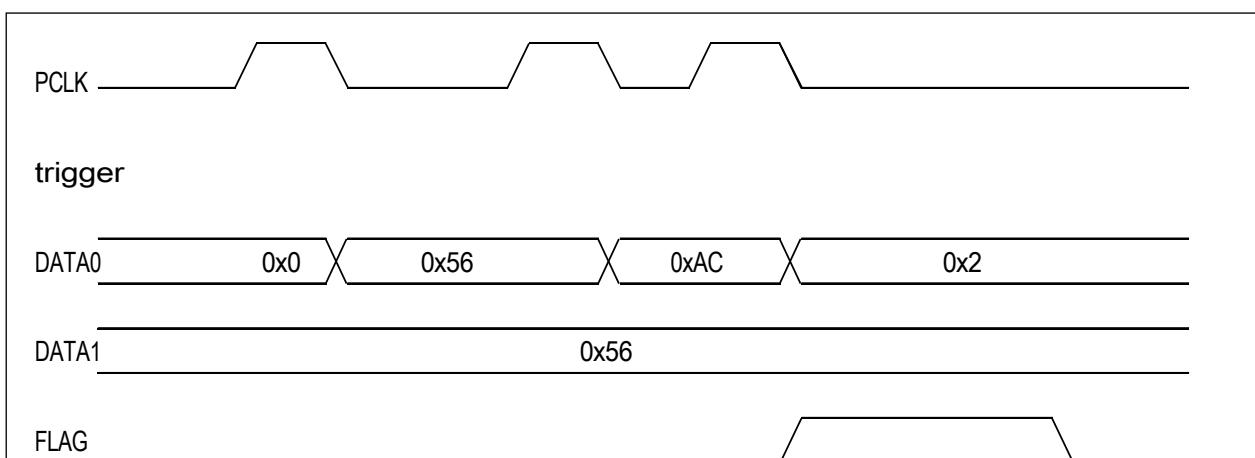
33.3 Application examples

33.3.1 Additive mode

Write 0xFF00 and 0x55 in DATA0 and DATA1 respectively, the result of calculation is 0xFF55, and the result is saved in DATA0. Write 0xFF in DATA1, the result of calculation is overflowed and the result is FLAG. FLAG.

33.3.2 Trigger plus mode

Configure the registers to achieve: select the trigger plus mode with **8bit** data width, write 0x00 and 0x56 in DATA0 and DATA1 respectively, and write the peripheral circuit event number in the trigger source selection register. The DCU is triggered by the event and performs an addition operation, and the result is 0x56, which is stored in DATA0. After 3 consecutive triggers, the calculation result overflows and the result FLAG is generated, and the software writes the FLAG register to clear the FLAG.



33.3.3 Comparison Mode

The FLAG output condition is DATA0>DATA1, and the comparison starts after writing DATA0. The FLAG output condition is DATA0 > DATA1, and the comparison starts after writing DATA0. When DATA0 is written to 0x9999, the FLAG generation condition is satisfied and the FLAG is reset by writing to the FLAG register.

33.4 Register Description

Register

List Unit 1

Name	English Abbreviations	Description	Address
DCU1 control register	DCU1_CTL	Configure the action mode of the DCU	0x4005_2000
DCU1 flag register	DCU1_FLAG	DCU's result identification	0x4005_2004
DCU1 data register 0	DCU1_DATA0	Storage of computing data	0x4005_2008
DCU1 data register 1	DCU1_DATA1	Storage of computing data	0x4005_200C
DCU1 data register 2	DCU1_DATA2	Storage of computing data	0x4005_2010

Unit 2

Name	English Abbreviations	Description	Address
DCU2 control register	DCU2_CTL	Configure the action mode of the DCU	0x4005_2400
DCU2 flag register	DCU2_FLAG	DCU's result identification	0x4005_2404
DCU2 data register 0	DCU2_DATA0	Storage of computing data	0x4005_2408
DCU2 data register 1	DCU2_DATA1	Storage of computing data	0x4005_240C
DCU2 data register 2	DCU2_DATA2	Storage of computing data	0x4005_2410
DCU2 flag reset register	DCU2_FLAGCLR	Clear DCU result markers	0x4005_2414
DCU2 interrupt condition selection register	DCU2_INTEVTSEL	Selecting the conditions for DCU generated interrupts	0x4005_2418

Module 3

Name	English Abbreviations	Description	Address
DCU3 control register	DCU3_CTL	Configure the action mode of the DCU	0x4005_2800
DCU3 flag register	DCU3_FLAG	DCU's result identification	0x4005_2804
DCU3 data register 0	DCU3_DATA0	Storage of computing data	0x4005_2808
DCU3 data register 1	DCU3_DATA1	Storage of computing data	0x4005_280C
DCU3 data register 2	DCU3_DATA2	Storage of computing data	0x4005_2810
DCU3 flag reset register	DCU3_FLAGCLR	Clear DCU result markers	0x4005_2814
DCU3 interrupt condition selection register	DCU3_INTEVTSEL	Selecting the conditions for DCU generated interrupts	0x4005_2818

Unit 4

Name	English Abbreviation	Description	Address
DCU4 control register	DCU4_CTL	Configure the action mode of the DCU	0x4005_2C00
DCU4 flag register	DCU4_FLAG	DCU's result identification	0x4005_2C04
DCU4 data register 0	DCU4_DATA0	Storage of computing data	0x4005_2C08
DCU4 data register 1	DCU4_DATA1	Storage of computing data	0x4005_2C0C
DCU4 data register 2	DCU4_DATA2	Storage of computing data	0x4005_2C10
DCU4 flag reset register	DCU4_FLAGCLR	Clear DCU result markers	0x4005_2C14
DCU4 interrupt condition selection register	DCU4_INTEVTSEL	Selecting the conditions for DCU generated interrupts	0x4005_2C18

33.4.1 DCU control register (DCU_CTL)

Register Description: This register is used to configure the action mode of DCU.

Address: 0x4005_2000, 0x4005_2400, 0x4005_2800, 0x4005_2C00

Reset value: 0x8000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
INTE N	Reserved														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								COM PTRG	Reserved			DATASIZE[1:0]	MODE[2:0]		

position	Marker	Place Name	Function	Reading and writing
b31	INTEN	Interrupt Enable	0: Interrupt generation is not allowed 1: Allow generation of interrupts	R/W
b30~b9	Reserved	-	Read 0 when reading, please write 0 when writing	R
b8	COMPTRG	Timing of comparison mode trigger	0: compare after writing to DATA0 1: Write DATA0 or DATA1 or DATA2 and compare	R/W
b7~b5	Reserved	-	Read 0 when reading, please write 0 when writing	R
b4~b3	DATASIZE[1:0]	Data Size	00: 8bit 01: 16bit 10: 32bit	R/W
b2~b0	MODE[2:0]	Action Mode	000: DCU invalid 001: Addition mode, operation is performed after data is written in DATA1 register 010: Subtraction mode, the operation is performed after the data is written in the DATA1 register 011: Hardware-triggered addition mode, triggered by other peripheral circuits to start the addition operation 100: Hardware-triggered subtraction mode, triggered by other peripheral circuits to start the subtraction operation 101: Comparison mode	R/W

33.4.2 DCU flag register (DCU_FLAG)

Register Description: This register generates the result identification of the DCU.

Address: 0x4005_2004, 0x4005_2404, 0x4005_2804, 0x4005_2C04

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															

position	Marker	Place Name	Function	Reading and writing
b31~b7	Reserved	–	Read 0 when reading, please write 0 when writing	R
b6	FLAG_GT1	Greater than flag bit 1	Set when DATA0>DATA1 in compare mode and clear when CLR_GT1 bit of DCU_FLAGCLR is written 1 in DCU flag reset register	R
b5	FLAG_EQ1	Equal to flag bit 1	In compare mode, set when DATA0 = DATA1 and cleared when CLR_EQ1 bit of DCU_FLAGCLR is written 1 in DCU flag reset register	R
b4	FLAG_LS1	Less than flag bit 1	In compare mode, set when DATA0 < DATA1 and clear when CLR_LS1 bit of DCU_FLAGCLR is written 1 in DCU flag reset register	R
b3	FLAG_GT2	Greater than flag bit 2	In comparison mode, set when DATA0 > DATA2 and clear when CLR_GT2 bit of DCU_FLAGCLR is written 1 in DCU flag reset register	R
b2	FLAG_EQ2	Equal to flag bit 2	In compare mode, set when DATA0 = DATA2 and clear when CLR_EQ2 bit of DCU_FLAGCLR is written 1 in DCU flag reset register	R
b1	FLAG_LS2	Less than flag bit 2	In comparison mode, set when DATA0 < DATA2 and clear when CLR_LS2 bit of DCU_FLAGCLR is written 1 in DCU flag reset register	R
b0	FLAG_OP	Operation flag bits	Addition, subtraction and triggered addition, triggered subtraction modes are set when addition generates an overflow or subtraction generates a downflow, and cleared when the CLR_OP bit of the DCU flag reset register DCU_FLAGCLR is written to 1	R



33.4.3 DCU data register (DCU_DATAx) (x=0,1,2)

Register description: Used to store the operation data. The functions of each data register in each mode are as follows:

	DATA0	DATA1	DATA2
Additive mode	Number to be added/result to be stored	Adding Numbers	Storage halving results
Trigger plus mode	Number to be added/result to be stored	Adding Numbers	Storage halving results
Subtractive mode	Subtracted number/storage result	Subtractions	Storage halving results
Trigger minus mode	Subtracted number/storage result	Subtractions	Storage halving results
Comparison Mode	Object being compared	Comparison object 1	Comparator 2
Comparison mode (window comparison)	Object being compared	Window cap	Lower window limit

Address: 0x4005_2008, 0x4005_2408, 0x4005_2808, 0x4005_2C08 0x4005_200C,

0x4005_240C, 0x4005_280C, 0x4005_2C0C

0x4005_2010, 0x4005_2410, 0x4005_2810, 0x4005_2C10

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DAT [31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DAT[15:0]															

position	Marker	Place Name	Function	Reading and writing
b31~b0	DAT[31:0]	Operational data	The actual number of bits used is set according to DCU_CTL.DATASIZE, when DCU_CTL.DATASIZE=00, DATA[7:0] is valid data. DATA[15:0] is valid data when DCU_CTL.DATASIZE=01. DATA[31:0] is valid data when DCU_CTL.DATASIZE=10	R/W

33.4.4 DCU flag reset register (DCU_FLAGCLR)

Register Description: This register is used to clear the result identification of the DCU.

Address: 0x4005_2014, 0x4005_2414, 0x4005_2814, 0x4005_2C14

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
CLR_GT	CLR_EQ	CLR_LS	CLR_GT	CLR_EQ2	CLR_LS	CLR_OP									

position	Marker	Place Name	Function	Reading and writing
b31~b7	Reserved	-	Read 0 when reading, please write 0 when writing	R/W
b6	CLR_GT1	Clear greater than flag bit 1	Clear the FLAG_GT1 bit of DCU_FLAG when writing 1, writing 0 has no effect	R/W
b5	CLR_EQ1	Clear equals flag bit 1	Clear the FLAG_EQ1 bit of DCU_FLAG when writing 1, writing 0 has no effect	R/W
b4	CLR_LS1	Clear less than flag bit 1	Clear the FLAG_LS1 bit of DCU_FLAG when writing 1, writing 0 has no effect	R/W
b3	CLR_GT2	Clear greater than flag bit 2	Clear the FLAG_GT2 bit of DCU_FLAG when writing 1, writing 0 has no effect	R/W
b2	CLR_EQ2	Clear equals flag bit 2	Clear the FLAG_EQ2 bit of DCU_FLAG when writing 1, writing 0 has no effect	R/W
b1	CLR_LS2	Clear less than flag bit 2	Clear the FLAG_LS2 bit of DCU_FLAG when writing 1, writing 0 has no effect	R/W
b0	CLR_OP	Clear arithmetic flag bit	Clear the FLAG_OP bit of DCU_FLAG when writing 1, writing 0 has no effect	R/W

33.4.5 DCU interrupt condition selection register (DCU_INTEVTSEL)

Register Description: This register can select the conditions under which the

DCU generates interrupt and output event signals Address: 0x4005_2018,

0x4005_2418, 0x4005_2818, 0x4005_2C18

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved							SEL_WIN[1:0]	SEL_GT1	SEL_EQ1	SEL_LS1	SEL_GT2	SEL_EQ2	SEL_LS2	SEL_OP	

position	Marker	Place Name	Function	Reading and writing
b31~b9	Reserved	-	Read 0 when reading, please write 0 when writing	R
b8~b7	SEL_WIN[1:0]	Window comparison break condition selection	The interrupt and output event signals are generated when the window comparison conditions set by SEL_WIN are met in the comparison mode, while the interrupt and output event signals are not generated when other comparison conditions are met when the SEL_WIN setting is valid 00: No window comparison interrupt and output signal are generated, other interrupt and event signal generation conditions are selected by b1~b6 of this register under this setting 01: Generate interrupt and output event signal when DATA0 data is in the window, i.e. DATA2≤DATA0≤DATA1 10: Generate interrupt and output event signal when DATA0 data is outside the window, i.e. DATA0> DATA1 or DATA0<DATA2 11: No interrupts and event signals are generated in comparison mode	R/W
b6	SEL_GT1	Greater than interrupt condition selection 1	In comparison mode and SEL_WIN=00, when DATA0>DATA1, the interrupt and output event signal are generated, when SEL_WIN≠00, the bit is invalid.	R/W
b5	SEL_EQ1	Equivalent to interrupt condition selection 1	In comparison mode and SEL_WIN=00, when DATA0=DATA1, the interrupt and output event signal are generated, when SEL_WIN≠00, the bit is invalid.	R/W
b4	SEL_LS1	Less than interrupt condition selection 1	In comparison mode and SEL_WIN=00, interrupt and output event signal are generated when DATA0<DATA1, and the bit is invalid when SEL_WIN≠00.	R/W
b3	SEL_GT2	Greater than interrupt condition option 2	In comparison mode and SEL_WIN=00, when DATA0>DATA2, the interrupt and output event signal are generated, when SEL_WIN≠00, the bit is invalid.	R/W
b2	SEL_EQ2	Equivalent to interrupt	In comparison mode and SEL_WIN=00, when DATA0=DATA2, the interrupt and output event signal are generated, when SEL_WIN≠00, the	R/W

		condition selection 2	bit is invalid.	
b1	SEL_LS2	Less than interrupt condition selection 2	In comparison mode and SEL_WIN=00, interrupt and output event signal is generated when DATA0<DATA2, and the bit is invalid when SEL_WIN≠00.	R/W
b0	SEL_OP	Operation flag bits	Generate interrupts and output event signals when the result overflows or underflows in addition and subtraction modes	R/W

34CRC Operations (CRC)

34.1 Introduction

In many applications, CRC algorithms are needed to verify the integrity and correctness of data. In particular, CRC checks are widely used in data transmission. This module can use both CRC16 and CRC32 algorithms to calculate and verify the data.

34.2 Functional Block Diagram

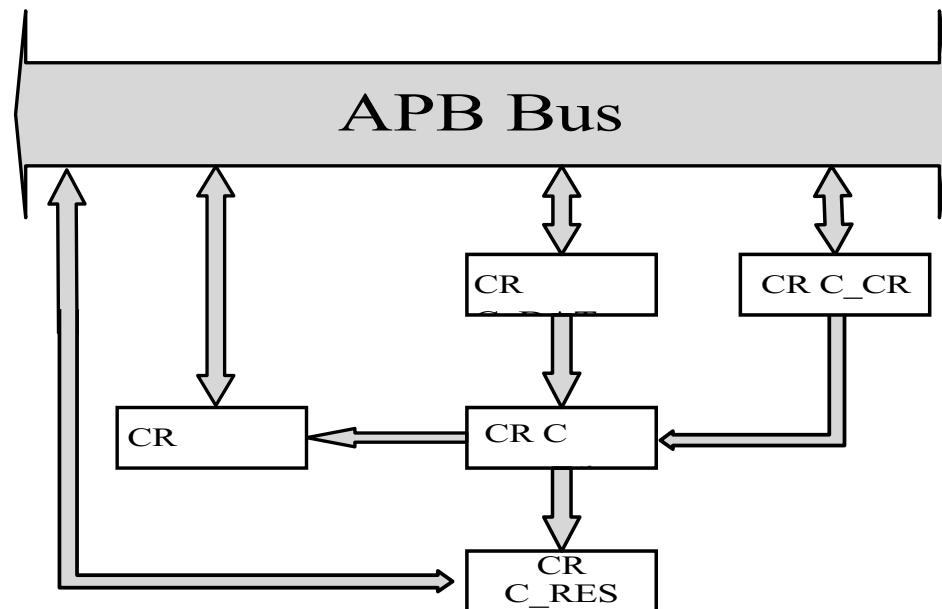


Figure 34-1 CRC Module Block Diagram

34.3 Function Description

The CRC algorithm of this module follows the definition of ISO/IEC13239 and uses 32-bit and 16-bit CRC respectively. the polynomial generated by CRC32 is $X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^8+X^7+X^5+X^4+X^2+X+1$. the polynomial generated by CRC16 is $X^{16}+X^{12}+X^5+1$.

The functions of this module include CRC code generation and CRC code verification.

34.3.1 CRC code generation

CRC encoding is the operation of a string of data to produce a 16/32-bit

CRC encoded result. The operation flow is as follows:

1. Set the CRC_CR register bits XOROUT,REFOUT,REFIN,CRC_SEL as required.
2. **CRC16 (CRC_SEL=0)** write 16-bit initial value to **CRC_REG[15:0]** of **CRC_RESLT** register
Medium.
CRC32 (CRC_SEL=1) Writes the 32-bit initial value to **CRC_REG[31:0]** of **the** **CRC_RESLT** register.
3. The data to be calculated is written to the CRC_DAT register sequentially, and each write operation corresponds to one data input (16 bits, 32 bits). For example, if there are 10 pieces of data, the CRC_DAT register is written 10 times in sequence.
4. After writing all the data to be computed into the CRC_DAT register, read the CRC_RESLT register CRC_REG to obtain the 16 / 32 - bit CRC encoding value.

34.3.2 CRC Checksum

CRC check is to judge a string of data and 16/32-bit CRC code, and check whether it is correct or not.

1. Set CRC_CR as required, including XOROUT,REFOUT,REFIN,CRC_SEL.
2. **CRC(CRC_SEL=0)** Writes the 16-bit initial value to **CRC_REG[15:0]** of **CRC_RESLT** register.
CRC32(CRC_SEL=1): Write the 32-bit initial value to **CRC_REG[31:0]** of the **CRC_RESLT** register.
3. Write the data to be calculated to the CRC_DAT register.
4. Writes the 16/32-bit CRC encoding value to the CRC_DAT register.
5. Read CRC_FLG register, a 1 means successful checksum, a 0 means failed checksum (for CRC16, you can also read CRC_RESLT[16],CRCFLAG_16 bits to determine whether the checksum is successful)

34.3.3 CRC checks for XOROUT,REFOUT,REFIN not all 1

When the options XOROUT,REFOUT,REFIN are not all 1s, the CRC code value following the data input needs to be transformed as follows and then written to CRC_DAT for checksum .

1. If REFOUT=0, invert all BITS of the CRC encoded value; otherwise, do not invert.
2. (b) When XOROUT=0, the result of (1) is inverted; otherwise, it is not inverted;
3. When REFIN=0, the BIT of each BYTE in the result of (2) is reversed completely; otherwise, it is not reversed;
4. Write the result of (3) to the CRC_DAT register for checksumming.

34.4 Register Description

Table 34-1 shows the register list of CRC

module. CRC_BASE_ADDR: 0x40008C00

Table 34-1 CRC Register List

Register Name	Sym bols	Offset Address	Bit width	Reset value
CRC control register	CRC_CR	0x00	32	0x0000_001C
CRC result register	CRC_RESLT	0x04	32	0x0000_0000
CRC flag register	CRC_FLG	0x0C	32	0x0000_0000
CRC Data Register	CRC_DAT	0x80~0xFC	32	0xFFFF_FFFF

34.4.1 Control register (CRC_CR)

Reset value: 0x001C

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	XOROUT	REFOUT	REFIN	CR	-

position	Marker	Place Name	Function	Reading and writing
b30~b5	Reserved	-	Read "1", write "0" when writing	R
b4	XOROUT	The result is inverted and output	0: The result is not inverted output 1: The results are inverted and output	R/W
b3	REFOUT	The result is output with all digits inverted	0: All digits of the result are output without reversal 1: Output all digits of the result upside down	R/W
b2	REFIN	Bit reversal within the input data byte	0: Input data bytes are not inverted in bits within the input 1: Bit reversal within the input data byte	R/W
b1	CR	Operation bit selection	0: CRC16 1: CRC32	R/W
b0	Reserved	-	Reserved Bits	R/W

34.4.2 Result register (CRC_RESLT)

Reset value: 0x0000

Select CRC16: (CRC_SEL=0)

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CRC_REG[15: 0]															

position	Marker	Place Name	Function	Reading and writing
b31~b17	Reserved	–	Read "0", write "0" when writing	R
b16	CRCFLAG_16	CRC16 checksum results	0: CRC16 operation check error 1 : CRC16 operation checks correctly	R
b15~b0	CRC_REG[15:0]	Result position	This 16-bit register is used to update and save the result of each CRC16 calculation; after the operation, reading this register will get the 16-bit CRC encoding result	R/W

Select CRC32: (CRC_SEL=1)

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CRC_REG[31:16]															
C_REG[15:0] 1															

position	Marker	Place Name	Function	Reading and writing
b31~b0	CRC_REG[31:0]	Result position	This 32-bit register is used to update and save the result of each CRC32 calculation; after the operation, reading this register will get the 32-bit CRC encoding result	R/W

34.4.3 Flag register (CRC_FLG)

Reset value: 0x0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CRC_FLAG

position	Marker	Place Name	Function	Reading and writing
b31~b1	Reserved	-	Read W , write "0" when writing	R
b0	CRC_FLAG	CRC32 checksum result	CRC checksum result flag bit; 0: current checksum error; 1: current checksum correct	R

34.4.4 Data register (CRC_DAT)

Reset value: 0x0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CRC_DAT[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
(C_DAT[15:C])															

position	Marker	Place Name	Function	Reading and writing
b31~b0	CRC_DAT[31:0]	Data Register	This register is used to input the data to be operated; the address of this register is a range (0x80~0xFC), which corresponds to 32 addresses (each address corresponds to 32 bits of data) An operation on any of the 32 addresses is considered to be an operation on this register.	W

35SDIO Controller (SDIOC)

35.1 Introduction

SDIOC provides an SD host interface and an MMC host interface for communication with SD cards supporting SD2.0 protocol, SDIO devices and MMC devices supporting eMMC4_2 protocol.

SDIOC features are as follows:

- Support SDSC, SDHC, SDXC format SD cards and SDIO devices
- Supports one-wire (1bit) and four-wire (4bit) SD buses
- Supports one-wire (1bit), four-wire (4bit) and eight-wire (**8bit**) MMC buses
- SD clock up to 50MHz
- With card recognition and hardware write protection

This product has 2 SDIO controllers, which can communicate with 2 SD/MMC/SDIO devices simultaneously.

35.2 Function Description

35.2.1 Port assignment

Port Name	IO	Function
SDIOx_CK(x=1~2)	0	SD clock output signal
SDIOx_CMD(x=1~2)	I/O	SD command and answer signals
SDIOx_D0(x=1~2)	I/O	SD data signal
SDIOx_D1(x=1~2)	I/O	SD data signal
SDIOx_D2(x=1~2)	I/O	SD data signal
SDIOx_D3(x=1~2)	I/O	SD data signal
SDIOx_D4(x=1~2)	I/O	SD data signal
SDIOx_D5(x=1~2)	I/O	SD data signal
SDIOx_D6(x=1~2)	I/O	SD data signal
SDIOx_D7(x=1~2)	I/O	SD data signal
SDIOx_CD(x=1~2)	I	SD card recognition status signal
SDIOx_WP(x=1~2)	I	SD write protect status signal

According to SD2.0 protocol and eMMC4.51 protocol, SD clock output signal (SDIOx_CK(x=1~2)) is used to output SD clock when SDIOC and SD/MMC device communicate; SD command and answer signal (SDIOx_CMD(x=1~2)) is used to output SD/MMC command to the device and receive answer information from the device; SD data signal (SDIOx_Dy(x=1~2) (y=0~7)) is used to send and receive data when SDIOC and device communicate. The SD data signal (SDIOx_Dy(x=1~2) (y=0~7)) is used to send and receive data between SDIOC and the device during the communication process. SDIOx_Dy(x=1~2) (y=0~3) is valid for communication, SDIOx_Dy(x=1~2) (y=4~7) is held high, SDIOx_Dy(x=1~2) (y=0~7) is valid for eight-wire (8bit) communication, and eight-wire (8bit) communication is limited to MMC device communication.

35.2.2 Basic access method

The user initiates the SDIOC to communicate with off-chip SD/MMC devices by reading and writing the SDIOC registers. Since writing the command register will trigger the transmit command action of the SDIOC, writing the command register must be done last. Before that, the user needs to set the transmission mode through the TRANSMODE register, set the command parameters through the parameter registers (ARG0, ARG1), and set the correct command number (command index), type, response type, etc. while setting the command register (CMD). Once the write command register is executed, the SDIOC will send the command and the user can read the

interrupt status register (NORINTST,ERRINTST) to check if the answer message is received and if there are any error messages. When a command is executed, the user can get the answer of the command by reading the answer registers (RESP0~7).

35.2.3 Data Transfer

SDIOC's data buffer registers are used to exchange data between host devices such as CPU/DMA and SD devices. SDIOC has built-in FIFO to speed up the data exchange. When the command contains data to be sent, user writes data to the data buffer registers (BUF0, BUF1) sequentially, and these data will be cached in the FIFO of the SDIOC first, and when the number of data written reaches the setting of the data length register (BLKSIZE) or 512 bytes, the SDIOC will send the data via SDIOx_Dy ($x=1\sim 2$) ($y=0\sim 7$) to send data. During this time the SDIOC is able to continue writing data to the data buffer register via the FIFO. Similarly, when the command contains data reception, the SDIOC sends data via SDIOx_Dy($x=1\sim 2$) ($y=0\sim 7$) and caches the data in the buffer (BUFFER) of the **SDIOC**, and the user gets the data by reading the data buffer register (BUF0, BUF1). Please refer to SD and MMC protocols for the format of data.

35.2.4 SD Clock

The SD clock (SDIOx_CK($x=1\sim 2$)) is generated by the host and output to the device for communication between them. The dividing factor of the divider is set by the clock control register (CLKCON). According to the requirements of SD2.0 protocol, the fastest frequency should be 50MHz in data transfer mode, so the user needs to set a reasonable dividing factor according to the frequency of EXCLK.

35.2.5 Interrupts and DMA start requests

35.2.5.1 SD Interrupt

SDIOC provides two types of interrupts, normal interrupt and error interrupt. Normal interrupt is an interrupt that occurs when various events are generated during the communication between SDIOC and SD/MMC card. The error interrupt is an interrupt generated when various errors occur during the communication of SDIOC actions. The normal interrupt and error interrupt have their own interrupt status register, interrupt status enable register, and interrupt signal enable register, respectively. The interrupt status register is used to indicate the cause of interrupt generation, the interrupt status enable register is used to enable each status bit of the interrupt status register, each status bit of the interrupt status register is valid when the enable bit of the interrupt status enable register is enabled, and the interrupt signal enable register is used to allow each interrupt cause to apply for interrupt to the CPU.

35.2.5.2 SDIO interrupt

The SDIO device can send a card interrupt request to the host when the transmission is idle. The card interrupt (CINTEN) in the interrupt status enable register (NORINTSTEN) is required to use SDIO interrupt, and the card interrupt (CINTEN) in the interrupt

signal enable register (NORINTSGEN) is also required to enable if an interrupt is requested from the CPU. The SDIO device requests an interrupt from the host by pulling the SDIO_x_D1(x=1~2) data line low. SDIOC will set the card interrupt (CINT) in the interrupt status register (NORINTST) **when** it detects that SDIO_x_D1(x=1~2) is pulled low and request an interrupt from the CPU according to the setting. SDIO_x_D1(x=1~2).

35.2.5.3 DMA Request

The SDIOC is able to read and write data in communication via DMA. When using DMA transmission to write data to the device, the start source of one channel of DMA is set to the write request of SDIOC, and the destination address of DMA transmission is set to the data buffer register (BUF0, BUF1) and is a fixed address. After SDIOC sends the write command, if the data FIFO is empty, it sends the write request signal to the DMA to start the DMA to write data to the data buffer registers (BUF0, BUF1), and the data will enter the data FIFO in turn. When the data in FIFO reaches the set value of Data Length Register (BLKSIZE) or 512 bytes, SDIOC will send the data via SDIOx_Dy ($x=1\sim 2$) ($y=0\sim 7$). If it is a multi-block write command, the SDIOC will continue to send a start request to write data to the DMA at the same time. When using DMA to transmit read data from the device, set the start source of the other channel of DMA to the read request of SDIOC, and then set the transfer source address of DMA to the data buffer register (BUF0,BUF1) and fixed address. After SDIOC sends the read data command, the device will send data through SDIOx_Dy ($x=1\sim 2$) ($y=0\sim 7$), and the data FIFO of SDIOC will start to receive data. When the data in FIFO reaches the setting value of data length register (BLKSIZE) or 512 bytes, SDIOC sends the read request signal to DMA to start DMA from data buffer register (BUF0, BUF1). buffer register (BUF0, BUF1). If it is a multi-block read command, SDIOC will continue to read the next data block from the device at the same time.

35.2.5.4 Card insertion (insert) and removal (remove)

The insertion (**insert**) and removal (**remove**) of SD/MMC/SDIO devices is recognized by the SDIOx_CD ($x=1\sim 2$) signal line. When there is no device in the card slot, the card slot will pull the SDIOx_CD($x=1\sim 2$) signal high through a resistor. When there is a device inserted, the SDIOx_CD($x=1\sim 2$) signal will be pulled low and will change to high again when the device is removed. SDIOC determines whether there is a device by the level of SDIOx_CD($x=1\sim 2$). The user can determine if a device is inserted by reading the CDPL bit in the host status register (**PSTAT**). sdio is capable of generating corresponding interrupts during device insertion and removal, which are enabled by the interrupt status enable register (**NORINTSEN**) and interrupt signal enable register (**NORINTSGEN**).

35.2.6 Host and device initialization

35.2.6.1 Host initialization

The SDIOC initialization steps are as follows:

1. Read the host status register (**PSTAT**), query the clock status and device insertion status
2. Configure the power control register (**PWRCON**) to enable SDIOC
3. Configure the clock control register (**CLKCON**) to enable SDIOC to output SD clock and

configure SD clock division according to the frequency of EXCLK to ensure that the SD clock frequency does not exceed 400KHz in card recognition mode

4. Configure the host control register (HOSTCON), select the 1-bit mode and disable the high speed mode
5. Configure the timeout control register (TOUTCON) according to the device characteristics to make the host end communication and report errors when the communication timeout occurs
6. Configure the normal and error interrupt status enable registers, the interrupt signal enable register to enable the SDIOC to generate an interrupt when needed and

Position the logo from

35.2.6.2 SD card initialization

After completing the SDIOC initialization configuration, if an SD card is connected. It needs to be initialized according to the SD protocol, and the initialization sequence is as follows:

1. Card reset, send reset command CMD0 to the device, CMD0 no answer message (response), then the card will enter the card identification mode
2. Confirm the operation condition of the card, send a CMD8 to the device and wait for an answer (response)
3. Send the initialization command ACMD41 (CMD55 first, then CMD41) to determine if the device has completed initialization based on the answer information, otherwise keep sending ACMD41 until the initialization is complete.
4. Send CMD2 to get the card's CID information
5. Send CMD3 to get the card's

RCA information The card will enter data transfer mode and initialization is complete.

35.2.6.3 MMC card initialization

After completing the SDIOC initialization configuration, if an MMC card is connected. It needs to be initialized according to the MMC protocol, and the initialization sequence is as follows:

1. Card reset, send reset command CMD0 to the device, CMD0 no answer message (response), then the card will enter the card identification mode
2. Confirm the operation condition of the card, send CMD1 to the device and wait for the reply message (response), determine whether the device has completed initialization based on the reply message, otherwise continue to send CMD1 until the initialization is complete.
3. Send CMD2 to get the card's CID information
4. Send CMD3 to get the card's

RCA information The card will enter data transfer mode and initialization is complete.

35.2.6.4 SDIO initialization

After completing the SDIOC initialization configuration, if an SDIO device is connected. It needs to be initialized according to SDIO protocol, and the initialization sequence is as follows:

1. Device reset, send reset command CMD0 to the device, CMD0 has no answer message (response)
2. Confirm the operation condition of the SDIO device, send CMD5 to the device and

wait to receive the reply message (response)

3. Send CMD3 to get the RCA information of the device, initialization is complete

35.2.7 SD/MMC single block read/write

After the SD/MMC card enters the data transfer mode, the SDIOC is able to access the data of the SD/MMC card through read/write commands. The sequence of single data block (block) read/write is as follows:

1. Configure the clock control register (CLKCON) to enable SDIOC to output SD clock and configure the SD clock divider according to the frequency of EXCLK so that the SD clock frequency does not exceed the maximum clock speed in **default** speed mode ($f_{pp} \leq 25\text{MHz}$).
2. Send CMD7 and the SD/MMC card will enter the data transfer state.
3. If necessary, you can configure the Host Control Register (HOSTCON) to set the host bus width and send ACMD6 to set the SD bus width (**1bit or 4bit**) for SD cards, or configure the Host Control Register (HOSTCON) to set the host bus width and rewrite the Ext_CSD register of the MMC card with the CMD6(SWITCH) command to set the bus width (1bit, 4bit or 8bit) for MMC cards. CSD register to set the bus width (**1bit, 4bit or 8bit**).
4. If needed, for SD card, the data block length can be set by configuring the data length register (BLKSIZE), and the data block size (number of bytes) can be set by CMD16 command in the range of 1~512 bytes. For SDHC/SDXC and MMC cards, the data block size is fixed 512 bytes.
5. If necessary, for SD card, you can configure the host control register (HOSTCON) to set the host to high speed mode and send CMD6 to switch the SD card to high speed mode ($f_{pp} \leq 50\text{MHz}$), after successful switching, you can set SDIO x _CK ($x=1\sim 2$) to Maximum 50MHz
6. When writing data, first configure the transfer mode register (TRANSMODE) and set the transfer mode to write, single block transfer. If you send single block write command CMD24, if you use CPU to write data, write data to data buffer register (BUF0, BUF1) after confirming BWR bit of interrupt status register (NORINTST) is 1. If you use DMA to write data, wait for DMA transfer to finish, SDIOC will send data through data signal SDIO x _Dy ($x=1\sim 2$) Send data. After the transmission is completed, the TC of the interrupt status register (NORINTST) will be set to 1 and an interrupt request will be generated. If an error occurs during the transmission, the corresponding error flag bit will be set and an interrupt request will be generated.
7. When reading data, first configure the transfer mode register (TRANSMODE) and set the transfer mode to read, single block transfer. Send the single block read command CMD17, and SDIOC will receive data by data signal SDIO x _Dy ($x=1\sim 2$). If the CPU is

used to read data, the data will be read from the data buffer register (BUF0, BUF1) after confirming the BRR bit of the interrupt status register (NORINTST) is 1. If an error occurs during the transfer, the corresponding error flag bit will be set and an interrupt request will be generated.

35.2.8 SD/MMC multi block read/write

The order of reading and writing multiple data blocks (block) versus single data blocks (block) is as follows:

1. Configure the clock control register (CLKCON) to enable SDIOC to output SD clock, and configure the SD clock division according to the frequency of EXCLK so that the SD clock frequency does not exceed the maximum clock speed in the **default** speed mode (**default** speed mode, **default** speed mode).

fpp<=25MHz).

2. Send CMD7 and the SD/MMC card will enter the data transfer state.
3. If necessary, you can configure the Host Control Register (HOSTCON) to set the host bus width and send ACMD6 to set the SD bus width (**1bit or 4bit**) for SD cards, or configure the Host Control Register (HOSTCON) to set the host bus width and rewrite the Ext_CSD register of the MMC card with the CMD6(SWITCH) command to set the bus width (1bit, 4bit or 8bit) for MMC cards. CSD register to set the bus width (**1bit, 4bit or 8bit**).
4. If needed, for SD card, you can configure the data length register (BLKSIZE) to set the data block length, and set the data block size (number of bytes) by CMD16 command, the configuration range is 1~512 bytes. For SDHC/SDXC and MMC cards, the data block size is fixed 512 bytes.
5. Configure the transfer mode register (**TRANSMODE**), set the transfer mode (read/write), and multi-block transfer. If Allow data block counting enable is set, the number of data blocks to be transferred needs to be set in the data block counting register (**BLKCNT**), and multi-block transfer cannot be started without setting the data block counting register (**BLKCNT**). If the block count disable enable is set, the block count register (**BLKCNT**) does not need to be set. At this time, an unlimited number of multi-block transfers can be performed, and when the host decides to end the transfer, it needs to send a CMD12 to the device after the last block transfer is completed to inform the device that the data transfer is finished.
6. If necessary, for SD card, you can configure the host control register (HOSTCON) to set the host to high speed mode and send CMD6 to switch the SD card to high speed mode (fpp<=50MHz), after successful switching, you can set SDIOx_CK ($x=1\sim 2$) to Maximum 50MHz
7. When writing data, the multi-block write command CMD25 is sent. If the CPU is used to write data, the data is written to the data buffer register (BUF0,BUF1) **after** confirming that the BWR bit in the interrupt status register (NORINTST) is 1, and the BWR will remain 0 during the sending process and reset to 1 after the sending is finished. If DMA is used to write data, the SDIOC will wait for the DMA transfer to be completed and send the data through the data signal SDIOx_Dy ($x=1\sim 2$). The TC of the interrupt status register (NORINTST) will be set to 1 and an interrupt request will be generated. If an error occurs during the transfer, the corresponding error flag bit will be set and an interrupt request will be generated. After all data is sent, the SDIOC will automatically send a CMD12 to end the transmission if it is set to send CMD12 automatically, otherwise it needs to send CMD12 to the device manually to inform the device that the data transmission

is finished.

8. When reading data, send the multi-block read command CMD18, and SDIOC will receive data through the data signal SDIOx_Dy ($x=1\sim 2$). If the CPU reads the data, the data will be read from the data buffer register (BUF0, BUF1) after confirming that the BRR bit in the interrupt status register (NORINTST) is 1. The BRR will remain 0 during the reception process and reset to 1 after the transmission is finished, and the user can read the data of the second block at this time. If DMA is used to read data, it will wait for the DMA transfer to be completed, and when the reading is finished, the TC of the interrupt status register (NORINTST) will be set to 1 and an interrupt request will be generated, and if there is an error in the transfer process, the corresponding error flag bit will be set and an interrupt request will be generated. After all the data is received, the SDIOC will automatically send a CMD12 to end the transmission if it is set to send CMD12 automatically, otherwise it needs to send CMD12 to the device manually to inform the device that the data transmission is finished.

35.2.9 Abort, suspend and resume

When a multi-block transfer is performed, it can be terminated (**abort**) or suspended (**suspend**) through software control. The **abort** operation can be performed regardless of whether the number of transferred blocks is set. There are asynchronous and synchronous abort operations. The asynchronous abort operation requires sending CMD12 through the Write Command Register (CMD) to abort the transfer while it is in progress, and the transfer will be terminated immediately. For write operation, the SD/MMC card will discard the current block of data and enter the burn mode (**program**) to write the previously received block of data to FLASH. For a read operation, the SD/MMC will stop transferring data. Synchronous termination means that the transfer is stopped at the block interval by setting the block interval register (**BLKGAP**) while the transfer is in progress, after the setting is completed, the transfer will be stopped at the end of the current block transfer. At this point, CMD12 needs to be sent to end the transmission.

To perform a **suspend** operation, first stop the transmission at the block interval by setting the SABGR bit in the block interval register (**BLKGAP**) and enable the **read wait** function. After the setting is completed, the transmission will be stopped at the end of the current data block transmission. At this point, send CMD52 through the Write Command Register (CMD) to suspend the transfer. If BS=0, it means the transmission is hung and the SD bus is idle. After the transfer is hung, the host can perform operations on other functions of the SDIO. However, if you want to **resume the operation later**, you need to backup the SDIOC registers (offset address 00h~0Dh) after the suspend, and then restore these register settings after the other operation. When performing the **resume operation**, first clear the block interval register (**BLKGAP**) to stop the transmission at the block interval, and then send CMD52 through the write command register (CMD) to resume the transmission. After sending the resume command, it is necessary to continue to read the DF bit (**data flag**) of the CCCR register of SDIO through CMD52. If DF=1 means there is data to continue transmission after executing the resume, if DF=0 means there is no data to be transmitted, then the software reset register (**SFTRST**) should be written to reset the data line.

Caution:

- Suspend and resume operations require SDIO devices and combo card devices to support such operations as well as **read wait** operations. Only the **read wait** function can be used.

35.2.10 **read wait**

The **read wait** allows the host to insert the CMD52 during a continuous data transfer to access other functions of the SDIO device. **Read wait is** performed by first stopping the

transfer at the block interval by setting the SABGR bit of the Block Interval Register (BLKGAP) and enabling the read **wait** function. After the setting is completed, the host will pull the SDIOx_D2 (x=1~2) data line low after the current data block transmission is finished. At this point, the transfer will be suspended and the host can insert CMD52 to access other functions that do not require data transfer. To end the read **wait** function, set the CR of the data block interval register (BLKGAP) and clear SABGR to resume transmission.

Caution:

- Read wait** Read wait requires SDIO device and combo card device support, and requires a four-wire bus transfer.

35.2.11 Wakeup

When not working for a long time, the system can be shifted to a low-power state to reduce power consumption. In the low-power state, the system can be woken up by **insert/removal** and **card interrupt of SD/MMC/SDIO devices** to continue operation. To use the wake-up function, the corresponding enable bits in the interrupt status enable register (NORINTSTEN) and the interrupt signal enable register (NORINTSGEN) need to be enabled for **insert/removal** or **card interrupt**. The corresponding wake-up function of SDIOx_CD(x=1~2)/SDIOx_D1(x=1~2) port is also enabled. After the configuration is completed, the system can be switched to low power mode. The wake-up function of SDIOx_CD(x=1~2)/SDIOx_D1(x=1~2) port will wake up the system when **insert/removal** and **card interrupt occur**.

Use **insert** to wake up the stop mode process:

1. Configure the SDIOC port and select SDIO1_CD as PA10
2. Configure the PCR register of PA10 and select IRQ10 active
3. Configure the PWRC3 register of the power control module to put the CPU into stop mode after executing the WFI command
4. Configure the interrupt control module WUPEN register to enable the wake-up function of IRQ10
5. Configure the EIRQCR10 register of the interrupt control module to select falling edge trigger
6. Configure PWRCON to enable SDIOC
7. Configure the interrupt status enable register (NORINTSTEN) and the interrupt signal enable register (NORINTSGEN) **by inserting (insert) the corresponding enable bits to enable**
8. Configure the interrupt source selection register to select the SDIOC interrupt as the interrupt source and enable the interrupt
9. Execute the **WFI** command to put the system into stop mode
10. When the device is inserted, PA10 will be pulled low, waking up the CPU via IRQ10 and entering the interrupt subroutine according to the SD interrupt request
11. Clear the insert status in the interrupt status register (NORINTST), exit the interrupt subroutine, and perform subsequent operations

Wake-up power down mode process using insert:

1. Configure the SDIOC port and select SDIO1_CD as PA10
2. Configure the PCR register of PA10 and select IRQ10 active
3. Configure the PWRC3 and PWRC0 registers of the power control module to enable the CPU to enter power-down mode after executing the WFI command
4. Configure the PDWKE1 register of the power control module to enable the power-down wake-up function of IRQ10
5. Configure the interrupt control module WUPEN register to enable the wake-up function of IRQ10
6. Configure the EIRQCR10 register of the interrupt control module to select falling edge trigger
7. Configure PWRCON to enable SDIOC
8. Configure the interrupt status enable register (NORINTSTEN) and the interrupt signal enable register (NORINTSGEN) by inserting (insert) the corresponding enable bits to enable
9. Configure the interrupt source selection register to select the SDIOC interrupt as the interrupt source and enable the interrupt
10. Execute the **WFI** command to put the system into power-down mode
11. When the device is plugged in, PA10 will be pulled low to wake up the system for re-powering via IRQ10

35.3 Register Description

The SDIOC module is designed according to the SD **Host Controller Standard Specification**, so the registers are also the same as described in the standard, with unused addresses and bits replaced by reserved bits.

The following table shows the register list of the SDIOC module.

BASE ADDR_ 0x4006FC00 (SDIOC1) 0x40070000 (SDIOC2)

Register Name	Sym bols	Offset Address	Bit width	Reset value
Data block length	BLKSIZE	0x04	16	0x0000h
Data block counting	BLKCNT	0x06	16	0x0000h
Parameter 0	ARG0	0x08	16	0x0000h
Parameter 1	ARG1	0x0A	16	0x0000h
Transfer Mode	TRANSMODE	0x0C	16	0x0000h
Command	CMD	0x0E	16	0x0000h
Respond 0	RESP0	0x10	16	0x0000h
Answer 1	RESP1	0x12	16	0x0000h
Answer 2	RESP2	0x14	16	0x0000h
Response 3	RESP3	0x16	16	0x0000h
Answer 4	RESP4	0x18	16	0x0000h
Answer 5	RESP5	0x1A	16	0x0000h
Answer 6	RESP6	0x1C	16	0x0000h
Answer 7	RESP7	0x1E	16	0x0000h
Data buffer 0	BUF0	0x20	16	0x0000h
Data buffer 1	BUF1	0x22	16	0x0000h
Host Status	PSTAT	0x24	32	0x00000000h
Host Control	HOSTCON	0x28	8	0x00h
Power Control	PWRCON	0x29	8	0x00h
Data block interval control	BLKGPCON	0x2A	8	0x00h
Clock Control	CLKCON	0x2C	16	0x0002h
Timeout control	TOUTCON	0x2E	8	0x00h
Software Reset	SFTRST	0x2F	8	0x00h
General interrupt status	NORINTST	0x30	16	0x0000h
Error interrupt status	ERRINTST	0x32	16	0x0000h
General interrupt state enable	NORINTSTEN	0x34	16	0x0000h
Error interrupt status enable	ERRINTSTEN	0x36	16	0x0000h
General interrupt signal enable	NORINTSGEN	0x38	16	0x0000h
Error interrupt signal enable	ERRINTSGEN	0x3A	16	0x0000h
Auto Command Error Status	ATCERRST	0x3C	16	0x0000h
Forced automatic command error status control	FEA	0x50	16	0x0000h
Forced error state control	FEE	0x52	16	0x0000h

In addition, SDIOC1 and SDIOC2 share an MMC mode enable register, which is used by the controller to switch between SD and MMC modes

Register Name	Symbols	Address	Bit width	Reset value
---------------	---------	---------	-----------	-------------

MMC mode enable register	MMCER	0x40055404	32	0x00000000h
--------------------------	-------	------------	----	-------------

35.3.1 Data block length register (BLKSIZE)

Offset address:

0x04 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0														
															Reserved	TBS [11:0]													

position	Marker	Place Name	Function	Reading and writing
b15~b12	Reserved	-	Read 0 when reading, please write 0 when writing	R
b11~0	TBS	Data block length	Set the length of the transfer block (Transfer Block Size), the length of the transfer block is in bytes, the setting range is 1~512	R/W

35.3.2 Data Block Counting Memory (BLKCNT)

Offset address:

0x6 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0														
															BC[15:0]														

position	Marker	Place Name	Function	Reading and writing
b15~b0	BC	Data block counting	Set the number of data blocks to be transferred (Block count), set the register to be performed when the transfer is stopped, and require the block count enable bit (BCE) of the transfer mode register to be active	R/W

35.3.3 Parameter register 0(ARG0)

Offset address:

0x08 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0														
															ARG0[15:0]														

position	Marker	Place Name	Function	Reading and writing

b15~b0	ARG[15:0]	Command parameters	Set the parameter contained in the current send command, this register is the low 16 bits of the parameter	R/W
--------	-----------	--------------------	--	-----

35.3.4 Parameter register 1(ARG1)

Offset address:

0x0A Reset

value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ARG1[15:0]															

position	Marker	Place Name	Function	Reading and writing
b15~b0	ARG[15:0]	Command parameters	Set the parameter contained in the current send command, this register is the high 16 bits of the parameter	R/W

35.3.5 Transmission mode register (TRANSMODE)

Offset address:

0x0C Reset

value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved										MULB	DDIR	ATCEN[1:0]	BCE	Rsvd	

position	Marker	Place Name	Function	Reading and writing
b15~b6	Reserved	–	Read 0 when reading, please write 0 when writing	R
b5	MULB	Multiple data blocks	0: The current transfer is a single block transfer (single block) 1: The current transfer is a multi block transfer (multi block)	R/W
b4	DDIR	Data transmission direction	0: Write operation (host sends data) 1: Read operation (host receives data)	R/W
b3~b2	ATCEN	Auto Command Enable	00: Do not send automatic commands 01: CMD12 is sent automatically at the end of multi-block transfer 10: Prohibit setting 11: Prohibit setting	R/W
b1	BCE	Block Count Enable	0: Disable data block counting enable 1: Allow data block counting enable	R/W
b0	Reserved	–	Read 0 when reading, please write 0 when writing	R

35.3.6 Command Register (CMD)

Offset address:

0x0E Reset

value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved						IDX[5:0]		TYP[1:0]	DAT	ICE	CCE	Rsvd		RESTYP[1:0]	

position	Marker	Place Name	Function	Reading and writing
b15~b14	Reserved	-	Read 0 when reading, please write 0 when writing	R
b13~b8	IDX	Order number	Number of the sending command	R/W
b7~b6	TYP	Command Type	00 : General command 01 : Suspend command 10 : Resume command 11 : Abort command	R/W
b5	DAT	Command with data	0: The current command only uses the SDIOx_CMD (x=1~2) command line 1 : The current command needs to use SDIOx_Dy(x=1~2) data line	R/W
b4	ICE	Number Check	0 : Do not check the command number in the response 1 : Check the command number in the response	R/W
b3	CCE	CRC Check	0: Do not check the CRC checksum in the reply 1: Check the CRC checksum in the reply	R/W
b2	Reserved	-	Read 0 when reading, please write 0 when writing	R
b1~b0	RESTYP	Response Type	00: No answer for this command 01: Command has a length of 136bit answer 10: Command has a length of 48bit answer 11: The command has a length of 48 bits and a busy status of the answer	R/W

35.3.7 Response register 0 (RESP0)

Offset address:

0x10 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP0[15:0]															

position	Marker	Place Name	Function	Reading and

b15~b0	RESP0[15:0]	Response Message	Bits 15~0 of the answer message	R
--------	-------------	---------------------	---------------------------------	---

35.3.8 Response register 1 (RESP1)

Offset address:

0x12 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP1[15:0]															

position	Marker	Place Name	Function	Reading and writing
b15~b0	RESP1[15:0]	Answer message	31~16 bits of response message	R

35.3.9 Response register 2 (RESP2)

Offset address:

0x14 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP2[15:0]															

position	Marker	Place Name	Function	Reading and writing
b15~b0	RESP2[15:0]	Response Message	47~32 bits of response message	R

35.3.10 Response register 3 (RESP3)

Offset address:

0x16 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP3[15:0]															

position	Marker	Place Name	Function	Reading and writing
b15~b0	RESP3[15:0]	Response	63~48 bits of the response message	R

35.3.11 Response register 4 (RESP4)

Offset address:

0x18 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP4															

position	Marker	Place Name	Function	Reading and writing
b15~b0	RESP4 [15:0]	Response Message	79~64 bits of response message	R

35.3.12 Response register 5 (RESP5)

Offset address:

0x1A Reset

value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP5[15:0]															

Bit tag	bit name function read and write
b15~b0	RESP5[15:0] answer message Bits 95~80 of the answer message R

35.3.13 Response register 6 (RESP6)

Offset address:

0x1C Reset

value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP6															

position	Marker	Place Name	Function	Reading and writing
b15~b0	RESP6 [15:0]	Response Message	Bit 111~96 of the response message	R

35.3.14 Response register 7 (RESP7)

Offset address:

0x1E Reset

value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESP7															

position	Marker	Place Name	Function	Reading and writing
b15~b0	RESP7 [15:0]	Response Message	Bits 127~112 of the response message	R

35.3.15 Data buffer register 0(BUF0)

Offset address:

0x20 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
BUF0[15:0]															

position	Marker	Place Name	Function	Reading and writing
b15~b0	BUF0	Data buffering	Write send data and read receive data, this register is the low 16 bits of data	R/W

35.3.16 Data buffer register 1(BUF1)

Offset address:

0x22 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
BUF1[15:0]															

position	Marker	Place Name	Function	Reading and writing
b15~b0	BUF1	Data buffering	Write send data and read receive data, this register is the high 16 bits of data	R/W

35.3.17 Host Status Register (PSTAT)

Offset address: 0x24

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved				CMDL	DATL[3:0]				WPL	CDL	CSS	CIN			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved		BRE	BWE	RTA	WTA	Reserved				DA	CID	CIC			

position	Marker	Place Name	Function	Reading and writing
b31~b25	Reserved	–	Read 0 when reading, please write 0 when writing	R
B24	CMDL	Command line status	Status of command line (SDIOx_CMD(x=1~2))	R
B23~b20	DATL	Data Cable Status	Status of data line (SDIOx_Dy(x=1~2) (y=0~3))	R
B19	WPL	Write protect line status	Status of write protect line (SDIOx_WP(x=1~2))	R
B18	CDL	Card Identification Line Status	Status of card identification line (SDIOx_CD(x=1~2))	R
B17	CSS	Equipment stable state	0: Card identification line status is unstable 1: Card identification line status is stable, the device has been inserted or not inserted	R
B16	CIN	Device insertion status	0: No device inserted 1: With equipment inserted	R
B15~b12	Reserved	–	Read 0 when reading, please write 0 when writing	R
B11	BRE	Data buffer full	0: The data buffer does not have enough data for reading 1: The data buffer has enough data for reading	R
B10	BWE	Data buffer empty	0: The data buffer can write data 1: The data buffer cannot write data	R
B9	RTA	Read operation status	0: No read operation in progress 1: There is an ongoing read operation	R
B8	WTA	Write operation status	0: No write operation in progress 1: There is an ongoing write operation	R
B7~b3	Reserved	–	Read 0 when reading, please write 0 when writing	R
B2	DA	Data cable transmission status	0: Data line idle 1: The data line is transferring data	R
B1	CID	With data command suppression	0: Allow sending commands with data 1: Prohibit sending commands with data	R
B0	CIC	Command Suppression	0: Allow sending commands 1 : Disable sending command	R

35.3.18 Host control register (HOSTCON)

Offset address:

0x28 Reset

value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
CDSS	CDTL	EXDW		Reserved	HSEN	DW	Rsvd

position	Marker	Place Name	Function	Reading and writing
b7	CDSS	Card identification line selection	0: Select the real SDIO _x _CD (x=1~2) line to reflect the card identification status 1: Select the card identification test signal to reflect the card identification status	R/W
b6	CDTL	Card identification test signal status	0: Card identification test signal is low (with device inserted) 1: Card identification test signal is high (no device inserted)	R/W
b5	EXDW	Extended data bit width	0: Data line bit width using the DW bit setting 1: Data line bit width is 8 bits (8bit)	R/W
b4~b3	Reserved	–	Read 0 when reading, please write 0 when writing	R
b2	HSEN	High-speed mode enable	0: Disable high speed mode 1: Enable high speed mode	R/W
b1	DW	Data bit width selection	0: Data line bit width is 1 bit (1bit) 1: Data line bit width is 4 bits (4bit)	R/W
b0	Reserved	–	Read 0 when reading, please write 0 when writing	R

35.3.19 Power Control Register (PWRCON)

Offset address:

0x29 Reset

value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
Reserved							PWON

position	Marker	Place Name	Function	Reading and writing
b7~b1	Reserved	–	Read 0 when reading, please write 0 when writing	R
b0	PWON	SDIOC enable	0: Disable SDIOC 1: Enable SDIOC	R/W

35.3.20 Data block gap control register (BLKGPCON)

Offset address:

0x2A Reset

value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
Reserved				IABG	RWC	CR	SABGR

position	Marker	Place Name	Function	Reading and writing
b7~b4	Reserved	-	Read 0 when reading, please write 0 when writing	R
b3	IABG	Data block gap interrupt control	0: Receive SDIO device interrupt during data block gap off (card interrupt) 1: Receive SDIO device interrupt during data block gap on (card interrupt)	R/W
b2	RWC	Read Wait Control	0: Disable the read wait function (read wait) 1: Enable the read wait function (read wait)	R/W
b1	CR	Continue transmission	0: No effect 1: Release the transmission stopped by the SABGR position bit	R/W
b0	SABGR	Data block gap stop transmission	0: Stop transmission when not in a data block gap 1: Stop transmission at data block gap	R/W

35.3.21 Clock control register (CLKCON)

Offset address:

0x2C Reset

value: 0x0002

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FS[7:0]								Reserved				CE	Rsvd	ICE	

position	Marker	Place Name	Function	Reading and writing
B15~b8	FS	Crossover Selection	SDIO _x _CK(x=1~2) clock division selection, the reference clock is EXCLK 0x80: 256 divisions of EXCLK 0x40: 128 divisions of EXCLK 0x20: 64 divisions of EXCLK 0x10: 32 divisions of EXCLK 0x08: 16 divisions of EXCLK 0x04: EXCLK's 8-way frequency 0x02: 4-way frequency of EXCLK 0x01: 2-way frequency of EXCLK 0x00: EXCLK Other: Prohibit setting	R/W

b2	CE	SDIOx_CK(x=1~2) lose Out of control	0: SDIOx_CK (x=1~2) stop output 1: SDIOx_CK(x=1~2) output	R/W
b0	ICE	Clock Enable	0: SDIOC action clock on 1: SDIOC action clock off	R/W

35.3.22 Timeout control register (TOUTCON)

Offset address:

0x2E Reset

value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
Reserved				DTO[3:0]			

position	Marker	Place Name	Function	Reading and writing
b7~b4	Reserved	-	Read 0 when reading, please write 0 when writing	R
b3~b0	DTO	Data timeout time	Set the data line SDIOx_Dy (x=1~2) (y=0~7) timeout determination time in Clock period of EXCLK 0000: EXCLK×213 0001: EXCLK×214 ----- 1110: EXCLK×227 1111: Prohibit setting	R/W

35.3.23 Software Reset Register (SFTRST)

Offset address:

0x2F Reset

value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
Reserved				RSTD RSTC RSTA			

position	Marker	Place Name	Function	Reading and writing
b7~b3	Reserved	-	Read 0 when reading, please write 0 when writing	R
b2	RSTD	Data Reset	Resets all data-related registers, containing the following register bits: BUFO, BUF1 BRE, PSTAT.BWE, PSTAT.RTA, PSTAT.WTA. PSTAT.DLA, PSTAT.CID BLKGPCON.CR, BLKGPCON.SABGR brr, norintst.bwr, norintst.bge. NORINTST.TC 0: Normal operation 1: Perform a reset	R/W
b1	RSTC	Command	Resets all command-related registers and contains the following	R/W

	Reset	register bits: PSTAT. NORINTST. 0: Normal operation 1: Perform a reset	
b0	RSTA	Reset all Reset all SDIOC registers except for card recognition function 0: Normal operation 1: Perform a reset	R/W

35.3.24 General interrupt status register (NORINTST)

Offset address:

0x30 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
El	Reserved					CINT	CRM	CIST	BRR	BWR	Rsvd	BGE	TC	CC	

position	Marker	Place Name	Function	Reading and writing
b15	El	Error interruption	Set when any error occurs in the error interrupt status register (ERRINTST)	R
b14~b9	Reserved	-	Read 0 when reading, please write 0 when writing	R
b8	CINT	Card Interruption	Set when the SDIO device issues a card interrupt request, reset when the SDIO device withdraws the request	R
b7	CRM	Card Removal	Set on device removal (remove), reset on write 1	R/W
b6	CIST	Card insertion	Set when device is inserted (insert), reset by writing 1	R/W
b5	BRR	Buffer readable	Set when data in buffer can be read (PSTAT.BRE=1), reset when write 1	R/W
b4	BWR	Buffer writable	Set when the buffer is writable (PSTAT.BWE=1), reset when write 1	R/W
b3	Reserved	-	Read 0 when reading, please write 0 when writing	R
b2	BGE	Data block gap stop transmission	Set when transfer stops at data block gap, write 1 reset	R/W
b1	TC	Transmission completed	Set on completion of read/write transfer, reset on write 1	R/W
b0	CC	Command completion	Set after the completion of sending the command without answer command and receiving the answer with answer command, and reset by writing 1	R/W

35.3.25 Error interrupt status register (ERRINTST)

Offset address:

0x32 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved						ACE	Rsvd	DEBE	DCE	DTOE	CIE	CEBE	CCE	CTOE	

position	Marker	Place Name	Function	Reading and writing
b15~b9	Reserved	-	Read 0 when reading, please write 0 when writing	R
b8	ACE	Auto send command error	Set when an error occurs in auto CMD, the type of error can be checked in the auto command error register (ATCERRST) and reset by writing 1	R/W
b7	Reserved	-	Read 0 when reading, please write 0 when writing	R
b6	DEBE	Data stop bit error	Set when the data line detects a low level in the stop bit, write 1 reset	R/W

b5	DCE	Data CRC checksum error	Set when CRC check error occurs on data line, write 1 reset	R/W
b4	DTOE	Data timeout error	Set when data timeout occurs, write 1 reset, data timeout time is set by the timeout control register (TOUTCON)	R/W
b3	CIE	Command number error	Set if the command number contained in the received answer is wrong, write 1 to reset	R/W
b2	CEBE	Command stop bit error	Set when the command line detects a low level in the stop bit and write 1 to reset	R/W
b1	CCE	Command CRC checksum error	Set when CRC check error occurs on command line, write 1 to reset	R/W
b0	CTOE	Command timeout error	Set when no answer is received for more than 64 SDIOx_CK (x=1~2) cycles after the command is sent, and write 1 to reset	R/W

35.3.26 General interrupt status enable register (NORINTSTEN)

Offset address:

0x34 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
							Reserved	CINTEN	CRMEN	CISTEN	BRREN	BWREN	Rsvd	BGEEN	TCEN	CCEN

position	Marker	Place Name	Function	Reading and writing
b15~b9	Reserved	–	Read 0 when reading, please write 0 when writing	R
b8	CINTEN	Card Interrupt Status Enable	0: Disable NORINTST.CINT from being set 1: Allow NORINTST.CINT to be set	R
b7	CRMEN	Card removal status enable	0: Disable NORINTST.CRM from being set 1: Allow NORINTST.CRM to be set	R/W
b6	CISTEN	Card insertion status enable	0: Disable NORINTST.CIST from being set 1: Allow NORINTST.CIST to be set	R/W
b5	BRREN	Buffer readable status enable	0: Disable NORINTST.BRR from being set 1: Allow NORINTST.BRR to be set	R/W
b4	BWREN	Buffer writable state enable	0: Disable NORINTST.BWR from being set 1: Allow NORINTST.BWR to be set	R/W
b3	Reserved	–	Read 0 when reading, please write 0 when writing	R
b2	BGEEN	Data Block Gap Stop Transmission Status Enable	0: disable NORINTST.BGE setting 1: Allow NORINTST.BGE to be set	R/W
b1	TCEN	Transmission completion status enable	0: disable NORINTST.TC setting 1: Allow NORINTST.TC to be set	R/W
b0	CCEN	Command completion status enable	0: Disable NORINTST.CC from being set 1: Allow NORINTST.CC to be set	R/W

35.3.27 Error interrupt status enable register (ERRINTSTEN)

Offset address:

0x36 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved														CTOEEN	

position	Marker	Place Name	Function	Reading and writing
b15~b9	Reserved	–	Read 0 when reading, please write 0 when writing	R
b8	ACEEN	Auto send command error status enable	0: Disable ERRINTST.ACE from being set 1: Allow ERRINTST.ACE to be set	R/W
b7	Reserved	–	Read 0 when reading, please write 0 when writing	R
b6	DEBEEN	Data stop bit error status enable	0: Disable ERRINTST.DEBE from being set 1: Allow ERRINTST.DEBE to be set	R/W
b5	DCEEN	Data CRC check error status enable	0: Disable ERRINTST.DCE from being set 1: Allow ERRINTST.DCE to be set	R/W
b4	DTOEEN	Data Timeout Error Status Enable	0: Disable ERRINTST.DTOE setting 1: Allow ERRINTST.DTOE to be set	R/W
b3	CIEEN	Command number error status enable	0: Disable ERRINTST.CIE from being set 1: Allow ERRINTST.CIE to be set	R/W
b2	CEBEEEN	Command Stop Bit Error Status Enable	0: Mask ERRINTST.CEBE set 1: Allow ERRINTST.CEBE to be set	R/W
b1	CCEEN	Command CRC checksum error status enable	0: Disable ERRINTST.CCE from being set 1: Allow ERRINTST.CCE to be set	R/W
b0	CTOEEN	Command Timeout Error Status Enable	0: disable ERRINTST.CTOE setting 1: Allow ERRINTST.CTOE to be set	R/W

35.3.28 General interrupt signal enable register (NORINTSGEN)

Offset address:

0x38 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
							Reserved	CINTSEN	CRMSEN	CISTSEN	BRRSEN	BWRSEN	Rsvd	BGESEN	TCESN	CCSEN

position	Marker	Place Name	Function	Reading and writing
b15~b9	Reserved	–	Read 0 when reading, please write 0 when writing	R
b8	CINTSEN	Card interrupt signal enable	0: Disable NORINTST.CINT application interrupt 1: Allow NORINTST.CINT to request an interrupt	R
b7	CRMSEN	Card removal signal enable	0: Disable NORINTST.CRM application interrupt 1: Allow NORINTST.CRM to request an interrupt	R/W
b6	CISTSEN	Card insertion signal enable	0: Disable NORINTST.CIST application interrupt 1: Allow NORINTST.CIST to request an interrupt	R/W
b5	BRRSEN	Buffer readable signal enable	0: Disable NORINTST.BRR application interrupt 1: Allow NORINTST.BRR to request an interrupt	R/W
b4	BWRSEN	Buffer writable signal enable	0: Disable NORINTST.BWR application interrupt 1: Allow NORINTST.BWR to request an interrupt	R/W
b3	Reserved	–	Read 0 when reading, please write 0 when writing	R
b2	BGESEN	Data block gap stop transmission signal enable	0: disable NORINTST.BGE application interrupt 1: Allow NORINTST.BGE to request an interrupt	R/W
b1	TCSEN	Transmission completion signal enable	0: disable NORINTST.TC to apply interrupt 1: Allow NORINTST.TC to request an interrupt	R/W
b0	CCSEN	Command completion signal enable	0: Disable NORINTST.CC application interrupt 1: Allow NORINTST.CC to request an interrupt	R/W

35.3.29 Error interrupt signal enable register (ERRINTSGEN)

Offset address:

0x3A Reset

value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
							ACEESN	Rsvd	DEBESEN	DCESEN	DTOESEN	CIESEN	CEBESEN	CCESEN	CTOESEN

position	Marker	Place Name	Function	Reading and writing
b15~b9	Reserved	–	Read 0 when reading, please write 0 when writing	R
b8	ACESEN	Auto send command error signal enable	0: ERRINTST.ACE application interrupt is prohibited 1: Allow ERRINTST.ACE to request an interrupt	R/W
b7	Reserved	–	Read 0 when reading, please write 0 when writing	R
b6	DEBESEN	Data stop bit error signal enable	0: ERRINTST.DEBE application interrupt is prohibited 1: Allow ERRINTST.DEBE to apply for interrupt	R/W
b5	DCESEN	Data CRC check error signal enable	0: ERRINTST.DCE application interrupt is prohibited 1: Allow ERRINTST.DCE to request an interrupt	R/W
b4	DTOESEN	Data timeout error signal enable	0: ERRINTST.DTOE application interrupt is prohibited 1: Allow ERRINTST.DTOE to apply for interrupt	R/W
b3	CIESEN	Command number error signal enable	0: Prohibit ERRINTST.CIE to apply interrupt 1: Allow ERRINTST.CIE to apply for interrupt	R/W
b2	CEBESEN	Command stop bit error signal enable	0: Mask ERRINTST.CEBE application interrupt 1: Allow ERRINTST.CEBE to request an interrupt	R/W
b1	CCESEN	Command CRC checksum error signal enable	0: ERRINTST.CCE application interrupt is prohibited 1: Allow ERRINTST.CCE to request an interrupt	R/W
b0	CTOESEN	Command timeout error signal enable	0: ERRINTST.CTOE application interrupt is prohibited 1: Allow ERRINTST.CTOE to request an interrupt	R/W

35.3.30 Automatic Command Error Status Register (ATCERRST)

Offset address:

0x3C Reset

value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
							Reserved	CMDE	Reserved		IE	EBC	CE	TOE	NE

position	Marker	Place Name	Function	Reading and writing
b15~b8	Reserved	–	Read 0 when reading, please write 0 when writing	R
b7	CMDE	Undelivered error	Set when other commands are not sent due to errors in bits b4~b0 of this register	R
b6~b5	Reserved	–	Read 0 when reading, please write 0 when writing	R
b4	IE	Command number error	Set if the number of the auto command contained in the received answer is wrong	R
b3	EBC	Stop bit error	Set when the command line detects a low level in the stop bit	R
b2	CE	Data timeout error	Set when a CRC check error occurs on the command line	R
b1 64 SDIOx_CK (x=1~2) cycles	TOE	Command timeout error is set when no answer is received after the automatic command is sent.	for more than position	R
b0	NE	not executed error Set when auto command is not sent for other reasons		R

35.3.31 Forced automatic command error status control register (FEA)

Offset address:

0x50 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved		FCMDE		Reserved		FIE		FEBE		FCE		FTOE		FNE	

position	Marker	Place Name	Function	Reading and writing
b15~b8	Reserved	–	Read 0 when reading, please write 0 when writing	R
b7	FCMDE	Forced unsent error	0: No effect 1: Force the occurrence of ATCERRST.CMDE error	W
b6~b5	Reserved	–	Read 0 when reading, please write 0 when writing	W
b4	FIE	Mandatory command numbering error	0: No effect 1: Force the occurrence of ATCERRST.IE error	W
b3	FEBE	Forced stop bit error	0: No effect 1: Force the occurrence of ATCERRST.EBE error	W
b2	FCE	Forced data timeout error	0: No effect 1: Force the occurrence of ATCERRST.CE error	W
b1	FTOE	Force command timeout error	0: No effect 1: Force the occurrence of ATCERRST.TOE error	W
b0	FNE	Forced unexecuted errors	0: No effect 1: Force the occurrence of ATCERRST.NE error	W

35.3.32 Forced Error Status Control Register (FEE)

Offset address:

0x52 Reset value:

0x0000

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
							Reserved	FACE	Rsvd	FDEBE	FDCE	FDTOE	FCIE	FCEBE	FCCE	FCTOE

position	Marker	Place Name	Function	Reading and writing
b15~b9	Reserved	–	Read 0 when reading, please write 0 when writing	R
b8	FACE	Force auto send command error	0: No effect 1: force the occurrence of ERRINTST.ACE error	W
b7	Reserved	–	Read 0 when reading, please write 0 when writing	R
b6	FDEBE	Forced data stop bit error	0: No effect 1: Force the occurrence of ERRINTST.DBE error	W
b5	FDCE	Forced data CRC checksum error	0: No effect 1: Force the occurrence of ERRINTST.DCE error	W
b4	FDTOE	Forced data timeout error	0: No effect 1: Force the occurrence of ERRINTST.DTOE error	W
b3	FCIE	Mandatory command numbering error	0: No effect 1: Force the occurrence of ERRINTST.CIE error	W
b2	FCEBE	Force command stop bit error	0: No effect 1: Force the occurrence of ERRINTST.CEBE error	W
b1	FCCE	Force command CRC checksum error	0: No effect 1: Force the occurrence of ERRINTST.CCE error	W
b0	FCTOE	Force command timeout error	0: No effect 1: Force the occurrence of ERRINTST.CTOE error	W

35.3.33 MMC Mode Enable Register (MMCER)

Address: 0x40055404

Reset value:

0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16																																																																																																
Reserved																																																																																																															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																																																																																																
Reserved																																																																																																															
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left; padding-bottom: 2px;">position</th><th style="text-align: left; padding-bottom: 2px;">Marker</th><th style="text-align: left; padding-bottom: 2px;">Place Name</th><th colspan="4" style="text-align: left; padding-bottom: 2px;">Function</th><th colspan="8" style="text-align: right; padding-bottom: 2px;">Reading and writing</th></tr> </thead> <tbody> <tr> <td>b31~b4</td><td>Reserved</td><td>–</td><td colspan="4">Read 1, write "0" when writing</td><td colspan="8" style="text-align: right;">R</td></tr> <tr> <td>b3</td><td>SELMMC2</td><td>SDIOC2</td><td>MMC mode</td><td>0: SDIOC2 selects SD mode</td><td>1: SDIOC2 selects MMC mode</td><td colspan="4"></td><td colspan="8" style="text-align: right;">R/W</td></tr> <tr> <td>b2</td><td>Reserved</td><td>–</td><td colspan="4">Read 1, write "0" when writing</td><td colspan="8" style="text-align: right;">R</td></tr> <tr> <td>b1</td><td>SELMMC1</td><td>SDIOC1</td><td>MMC mode</td><td>0: SDIOC1 selects SD mode</td><td>1: SDIOC1 selects MMC mode</td><td colspan="4"></td><td colspan="8" style="text-align: right;">R/W</td></tr> <tr> <td>b0</td><td>Reserved</td><td>–</td><td colspan="4">Read 1, write "0" when writing</td><td colspan="8" style="text-align: right;">R</td></tr> </tbody> </table>																position	Marker	Place Name	Function				Reading and writing								b31~b4	Reserved	–	Read 1, write "0" when writing				R								b3	SELMMC2	SDIOC2	MMC mode	0: SDIOC2 selects SD mode	1: SDIOC2 selects MMC mode					R/W								b2	Reserved	–	Read 1, write "0" when writing				R								b1	SELMMC1	SDIOC1	MMC mode	0: SDIOC1 selects SD mode	1: SDIOC1 selects MMC mode					R/W								b0	Reserved	–	Read 1, write "0" when writing				R							
position	Marker	Place Name	Function				Reading and writing																																																																																																								
b31~b4	Reserved	–	Read 1, write "0" when writing				R																																																																																																								
b3	SELMMC2	SDIOC2	MMC mode	0: SDIOC2 selects SD mode	1: SDIOC2 selects MMC mode					R/W																																																																																																					
b2	Reserved	–	Read 1, write "0" when writing				R																																																																																																								
b1	SELMMC1	SDIOC1	MMC mode	0: SDIOC1 selects SD mode	1: SDIOC1 selects MMC mode					R/W																																																																																																					
b0	Reserved	–	Read 1, write "0" when writing				R																																																																																																								

36 Debug Controller (DBG)

This product is referenced in the following ARM technical documents:

- Cortex™-M4F r0p1 Technical Reference Manual (TRM)
- ARM Debug Interface V5
- ARM **CoreSight** Design Suite Version r0p1 Technical Reference Manual

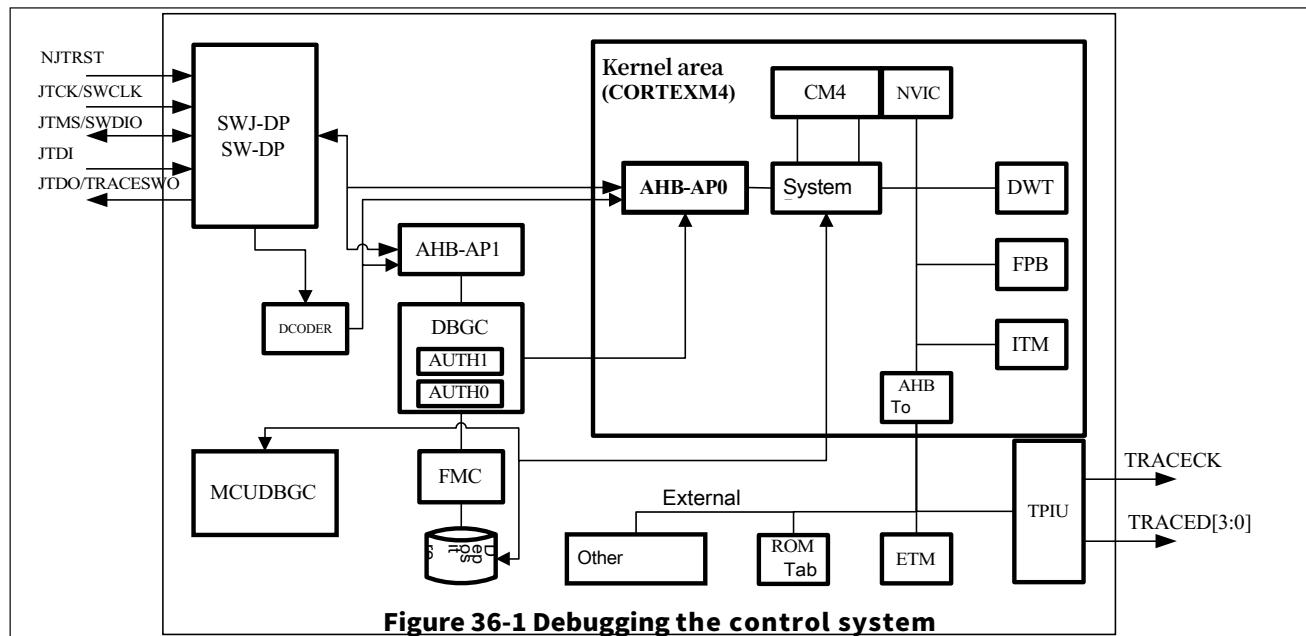
36.1 Introduction

The core of this MCU is the Cortex™-M4F, which contains hardware for advanced debug functions. With these debug functions, the kernel can be stopped when fetching fingers (instruction breakpoints) or fetching access data (data breakpoints). When the kernel is stopped, the internal state of the kernel and the external state of the system can be queried. When the query is completed, the kernel and system will be restored and program execution will resume. This MCU is not equipped with an ETM debugging device.

Two debugging interfaces are provided:

- Serial Debug Trace Interface SWD
- Parallel debug trace interface JTAG

36.2 DBGC System Block Diagram



The ARM Cortex™-M4F core provides integrated on-chip debug support. It includes:

- SWJ-DP: SWD/JTAG debugging port
- AHB-AP: AHB Access Port
- ITM: Instruction Tracking Unit
- FPB: Flash instruction breakpoint
- DWT: Data Breakpoint Trigger
- TPIU: Trace Port Unit Interface (available on large packages, where the corresponding pins will be mapped)
- Flexible debugging pin

assignment notes:

- For more information on the debugging features supported by the ARM Cortex™-M4F core support, see the **Cortex™-M4F- r0p1 Technical Reference Manual** and the **CoreSight Design Suite r0p1 Technical Reference M**.

36.3 SWJ-DP debug port (SWD and JTAG)

The MCU core incorporates the SWD/JTAG Debug Port (SWJ-DP). This port is the ARM standard CoreSight debug port with a JTAG-DP (5-pin) interface and a SW-DP (2-pin) interface.

- The JTAG Debug Port (JTAG-DP) provides a 5-pin standard JTAG interface for connection to the AHP-AP port.
- The Serial Wire Debug Port (SW-DP) provides a 2-pin (clock + data) interface for connection to the AHP-AP port.

In SWJ-DP, **the** two JTAG pins of SW-DP are multiplexed with some of the five JTAG pins of JTAG-DP. This means that asynchronous tracking is only possible on **the** SW-DP, not on **the** JTAG-DP.

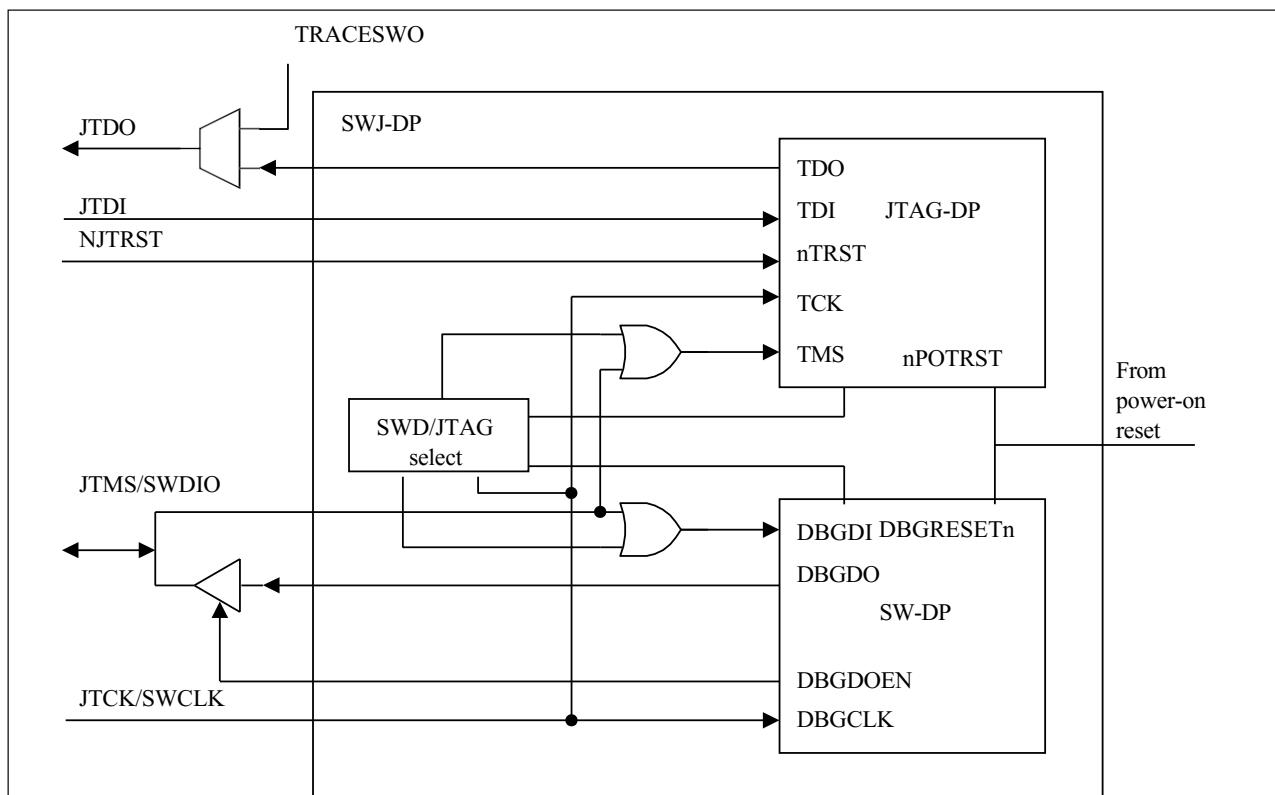


Figure 36-2 Debugging the control system

36.3.1 Switching mechanism for JTAG-DP or SW-DP

The default debug interface is the JTAG-DP interface.

If the debug tool wants to switch to SW-DP, it must provide a dedicated JTAG sequence on JTMS(SWDIO)/JTCK(SWCLK) to disable JTAG-DP and enable SW-DP, so that only the SWCLK and SWDIO pins can be used to access SW-DP.

The sequence is:

1. Outputs JTMS (SWDIO) =1 signal for more than 50 JTCK cycles
2. Output 16 JTMS (SWDIO) signals 0111_1001_1110_0111 (MSB)
3. Outputs JTMS (SWDIO) =1 signal for more than 50 JTCK cycles

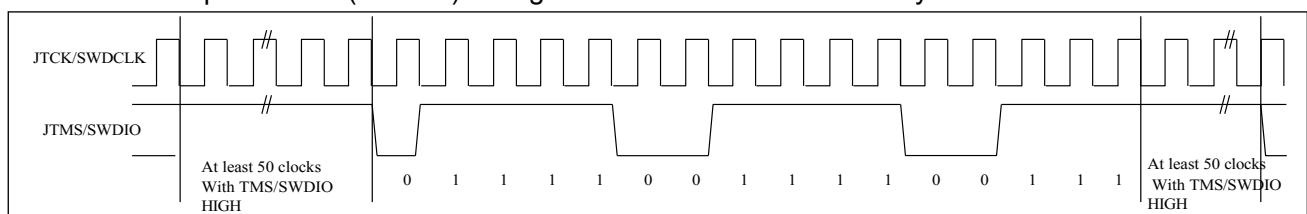


Figure 36-3 JTAG-DP to SW-DP Switching Timing

36.4 Pinouts and debug port pins

Depending on the package of the MCU, there are different numbers of active pins. Therefore, certain pin-related functions may vary with the package.

36.4.1 SWJ Debug Port Pins

The 5 common ports of the MCU can be used as SWJ-DP interface pins.

Table 36-1 SWJ Debug Port Pinout

SWJ-DP Pin Name	JTAG debug port		SW debug port	
	Type	Description	Type	Debugging distribution
JTMS/SWDIO	I	JTAG mode selection	I/O	Serial line data input/output
JTCK/SWCLK	I	JTAG clock	I	Serial Clock
JTDI	I	JTAG data input	-	-
JTDO/TRACESWO	O	JTAG data output	-	TRACESWO (if asynchronous tracking is enabled)
NJTRST	I	JTAG reset	-	-

36.4.2 Flexible SWJ-DP pin assignment

After a reset (power-up or pin reset), all five pins used for SWJ-DP are designated as dedicated pins and are available for immediate use by the debug tool (note that trace outputs are not assigned unless explicitly programmed) However, the MCU can disable some or all of the SWJ-DP ports, thereby releasing the associated pins for use as GPIOs. for more detailed information on how to disable the SWJ-DP port pins, see: General IO Special Control Register PSPCR.

Table 36-2 Flexible SWJ-DP Pin Assignments

Available debug ports	Assigned SWJ IO pins				
	JTMS/ SWDIO	JTCK/ SWCLK	JTDI	JTDO	NJTRST
All SWJ (JTAG-DP+SW-DP) - reset state	✓	✓	✓	✓	✓
Disable JTAG-DP and enable SW-DP	✓	✓	Releaseable	Releaseable	Releaseable
Disable JTGA-DP and Disable SW-DP	Releaseable	Releaseable	Releaseable	Releaseable	Releaseable

36.4.3 Internal pull-up on JTAG pins

According to the JTAG IEEE standard, it is important to ensure that the JTAG input pins are not left dangling, as these pins are connected directly to the MCU internals to control the debug functions. Special attention must also be paid to the JTCK/SWCLK pins, which are used directly for debug control clock functions. To avoid floating IO levels, the MCU has built-in internal pull-up resistors on the JTAG pins:

- NJTRST: Internal pull-up
- **JTDI**: Internal pull-up
- JTMS/SWDIO: Internal pull-up
- JTCK/SWCLK: Internal pull-up
- JTDO: High resistance state

In the unconnected debugger state, the user software can release the JTAG IO as a normal I/O port by setting the GPIO special control register. Since the internal pull-up of the chip is a weak pull-up of <100K ohms, it is recommended to use an external pull-up of 10K ohms.

36.4.4 Using the serial interface and releasing unused debug pins for GPIO

Some GPIOs can be released when using SWD, and the user software must change the GPIO configuration in the GPIO control register so that the corresponding pins can be released for use as GPIOs.

During debugging, the host computer performs the following operations:

- Assign all SWJ pins (JTAG-DP+SW-DP) **in** the system reset state.
- In the system reset state, the debug host sends a JTAG sequence to switch from JTAG-DP to SW-DP.
- Still in the system reset state, the debug host sets a breakpoint at the reset address.
- The reset signal is released and the kernel stops at the reset address.
- Switch from this debug port to SW-DP. then reassign the other JTAG pins to GPIO via user software. note

For user software design, these pins are still in the input pull-up (NJTRST, JTMS, JTDI, JTCK) during the period from reset until the user software releases them. For user software designs that require release of debug pins, these pins remain in input pull-ups (NJTRST, JTMS, **JTDI**, JTCK and JTDO) after reset until the pins are released by the user software

36.5 Register

The registers are described as follows:

Base address: 0xE004_2000

Register Name	Sym bols	Offset Address	Bit width	Initial Value	Access to hosts
DBG Status Register	MCUDBGSTAT	0x001C	32	0x00000000	CPU/Debug IDE*
Peripheral debug pause register	MCUSTPCTL	0x0020	32	0x00000003	CPU/Debug IDE*
TRACE Port Control Register	MCUTRACECTL	0x0024	32	0x00000000	CPU/Debug IDE*

Caution:

-The The registers are located in the PPB area and can only be accessed by the CPU in privileged mode.

36.5.1 DBG Status Register (MCUDBGSTAT)

DBG Debug power-up status

acknowledgement register. Reset

value: 0x0000_0001

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	CDB G PWR UPA CK	CDB G PWR UPR EQ

position	Marker	Place Name	Function	Reading and writing
b31~2	Reserved	-	Read '1', write "0" when writing	R/W
b1	CDBGPWRUPACK	Debugger power-up feedback	0: No feedback 1: Debugging power-up feedback	R/W
b0	CDBGPWRUPREQ	Debugger power-up request	0: No power-up request 1: Power-up request	R/W

36.5.2 Peripheral debug pause register (MCUSTPCTL)

When the CPU is in the debug state, the peripheral module suspends control. Reset value: 0x0000_0003

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TMR	-	-	-	-											
A6	A5	A4	A3	A2	A1	63	62	61	43	42	41				
STP															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TMR	TMR	-	-	-	-	-	-	-	PVD	PVD	PVD	RTC	WDT	SWD	
02	01								2	1	0	STP	STP	TST	
STP	STP								STP	STP	STP				P

position	Marker	Place Name	Function	Reading and writing
b31	TMRA6STP	TimerA-6 count pause signal	0: Counter still counts even if the kernel is stopped 1: The counter is suspended when the kernel is stopped	R/W
b30	TMRA5STP	TimerA-5 count pause signal	0: Counter still counts even if the kernel is stopped 1: The counter is suspended when the kernel is stopped	R/W
b29	TMRA4STP	TimerA-4 count pause signal	0: Counter still counts even if the kernel is stopped 1: The counter is suspended when the kernel is stopped	R/W
b28	TMRA3STP	TimerA-3 count pause signal	0: Counter still counts even if the kernel is stopped 1: The counter is suspended when the kernel is stopped	R/W
b27	TMRA2STP	TimerA-2 count pause signal	0: Counter still counts even if the kernel is stopped 1: The counter is suspended when the kernel is stopped	R/W
b26	TMRA1STP	TimerA-1 count pause signal	0: Counter still counts even if the kernel is stopped 1: The counter is suspended when the kernel is stopped	R/W
b25	TMR63STP	Timer6-3 count pause signal	0: Counter still counts even if the kernel is stopped 1: The counter is suspended when the kernel is stopped	R/W
b24	TMR62STP	Timer6-2 count pause signal	0: Counter still counts even if the kernel is stopped 1: The counter is suspended when the kernel is stopped	R/W
b23	TMR61STP	Timer6-1 count pause signal	0: Counter still counts even if the kernel is stopped	R/W

			1: The counter is suspended when the kernel is stopped	
b22	TMR43STP	Timer4-3 count pause signal	0: Counter still counts even if the kernel is stopped 1: The counter is suspended when the kernel is stopped	R/W
b21	TMR42STP	Timer4-2 count pause signal	0: Counter still counts even if the kernel is stopped 1: The counter is suspended when the kernel is stopped	R/W
b20	TMR41STP	Timer4-1 count pause signal	0: Counter still counts even if the kernel is stopped 1: The counter is suspended when the kernel is stopped	R/W
b19	Reserved	-	Read '0', write "0" when writing	R/W
b18	Reserved	-	Read '0', write "0" when writing	R/W
b17	Reserved	-	Read '0', write "0" when writing	R/W
b16	Reserved	-	Read '0', write "0" when writing	R/W
b15	TMR02STP	Timer0-2 count pause signal	0: Counter still counts even if the kernel is stopped 1: The counter is suspended when the kernel is stopped	R/W
b14	TMR01STP	Timer0-1 count pause signal	0: Counter still counts even if the kernel is stopped 1: The counter is suspended when the kernel is stopped	R/W

b13~6	Reserved	-	Read "1", write "0" when writing	R/W
b5	PVD2STP	PVD2 interrupt/reset mask	0: Generate PVD2 interrupt request or reset even if the kernel is stopped 1: Mask PVD2 interrupt request or reset when kernel stops	R/W
b4	PVD1STP	PVD1 interrupt/reset mask	0: Generate PVD1 interrupt request or reset even if the kernel is stopped 1: Mask PVD1 interrupt application or reset when kernel stops	R/W
b3	PVD0STP	PVD0 interrupt/reset mask	0: Generate PVD0 interrupt request or reset even if the kernel is stopped 1: Mask PVD0 interrupt application or reset when the kernel stops	R/W
b2	RTCSTP	RTC count pause signal	0: RTC counter still counts even if the kernel is stopped 1: When the kernel stops, the RTC counter pauses counting	R/W
b1	WDTSTP	WDT count pause signal	0: WDT counter still counts even if the kernel is stopped 1: When the kernel stops, the WDT counter suspends counting	R/W
b0	SWDTSTP	SWDT count pause signal	0: SWDT counter still counts even if the kernel is stopped 1: When the kernel stops, the SWDT counter is suspended	R/W

36.5.3 Debug Component Configuration Register (MCUTRACECTL)

Configure the TRACE output pins with

this register. Reset value:

0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	TRA CEI OEN	TRACEMODE	

position	Marker	Place Name	Function	Reading and writing
b31~b3	Reserved	-	Read "0", write "0" when writing	R/W
b2	TRACEIOEN	TRACE pin output control	0: Synchronous tracking pin output disable 1: Synchronous tracking pin output license	R/W
b1~b0	TRACEMODE	TRACEDATA output pin control	00: Asynchronous tracking 01: Synchronous tracking 1 bit TRACEDATA[0] 10: Synchronous tracking of 2-bit TRACEDATA[1:0] 11: Synchronous tracking of 4-bit TRACEDATA[3:0]	R/W

36.6 SW Debug Port

36.6.1 SW Protocol Introduction

The synchronous serial protocol uses two pins:

- SWCLK: Clocking from host to slave
- SWDIO: Bidirectional

When transferring data, the LSB comes first.

For SWCLK and SWDIO, the line needs to be pulled up on the board (10K ohms recommended)

36.7 TPIU (Tracking Port Interface Unit)

36.7.1 Introduction

TPIU is the bridge between ITM and on-chip trace data.

The output stream is encapsulated into a trace source ID and then captured by the Trace Port Analyzer (TPA).

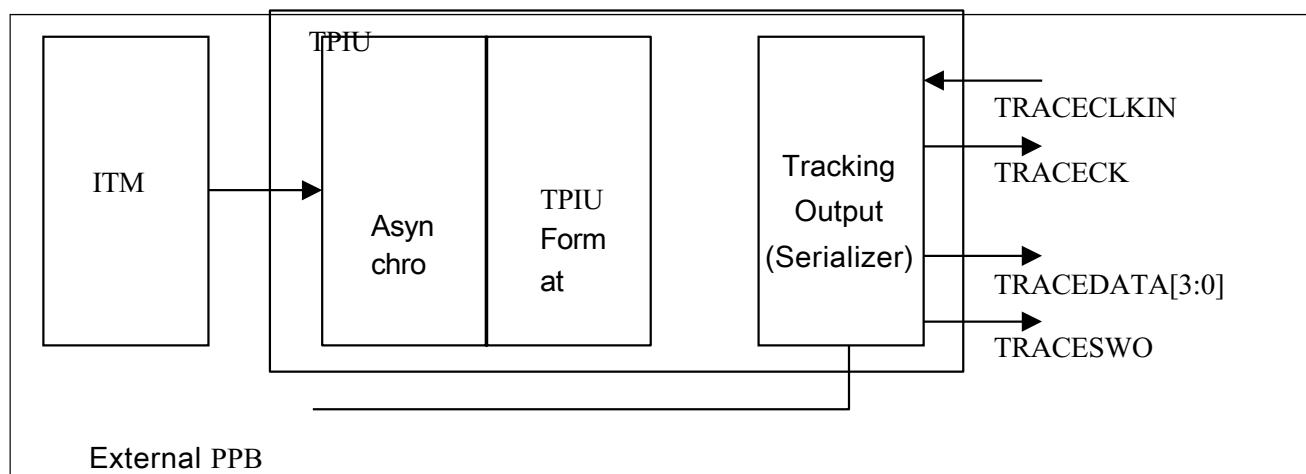


Figure 36-4 TPIU Block Diagram

36.7.2 TRACE Pin Assignment

- Asynchronous mode

Asynchronous mode requires 1 extra pin and is available for all packages. Asynchronous mode is only available when using serial line mode (not available in JTAG mode)

TPIU Pin Name	Tracking Asynchronous Mode	
	Type	Description
TRACESWO	0	TRACE asynchronous data

- Synchronous mode

Synchronous mode requires 2 to 5 additional pins, depending on the length of the data being traced, and is only available in larger packages. In addition, synchronous mode is available in

TPIU Pin Name	Tracking synchronization mode	
	Type	Description
TRACECK	0	TRACE Clock
TRACED[3:0]	0	TRACE synchronized data output, either 1, 2 or

TPIU TRACE Pin Assignment

By default, these pins are not assigned. These pins can be configured by placing the TRACE_IOEN and TRACE_MODE bits in the MCU Debug Component Configuration Register (MCUTRACECTL). This configuration must be done by the debug host or CPU.

In addition, the number of pins to be assigned depends on the trace configuration (asynchronous trace or synchronous trace)

- Asynchronous mode: 1 additional pin required
- Synchronous mode: 5 additional pins required
 - TRACECK
 - TRACED[0] (if port data length is configured as 1, 2 or 4)
 - TRACED[1] (if port data length is configured as 2 or 4)
 - TRACED[2] (if port data length is configured 4)
 - TRACED [3] (if port data length is configured 4)

To assign TRACE pins, the debug host must program bits TRACE_IOEN and TRACE_MODE[1:0] **of** the MCU Debug Configuration Register (MCUTRACECTL). The TRACE pins are not assigned by default.

This register is mapped to the external PPB bus and is reset by power-up (not pin reset). This register can be written via the debugger in the pin reset state.

TPIU Pin Usage	Assigned TRACE IO pins					
	JTDO/ TRACESWO	TRACECK	TRACED[0]	TRACED[1]	TRACED [2]	TRACED [3]
No tracking (default state) TRACE_IOEN =0 TRACE_MODE=XX	Release*	Release	Release	Release	Release	Release
Asynchronous tracking TRACE_IOEN =1 TRACE_MODE=00	TRACESWO	Release	Release	Release	Release	Release
Synchronous tracking 1 bit TRACE_IOEN =1 TRACE_MODE=01	Release*	TRACECK	TRACED[0]	Release	Release	Release
Simultaneous tracking of 2 bits TRACE_IOEN =1 TRACE_MODE=10	Release*	TRACECK	TRACED[0]	TRACED[1]	Release	Release
Simultaneous tracking of 4 bits TRACE_IOEN =1 TRACE_MODE=11	Release*	TRACECK	TRACED[0]	TRACED[1]	TRACED [2]	TRACED [3]

Caution:

- When using serial mode, this pin is released. However, when using JTAG, this pin is assigned to TDO.

36.7.3 MCU internal TRACECLKIN connection

In this MCU, the clock TRACECLKIN of TPIU is connected to the internal clock. the default clock of MCU is the internal MRC oscillator. The frequency in the reset state is different from the frequency after reset release. The reason for this is that since the default MRC calibration is used in the system reset state, this MRC calibration is updated each time the system is reset and released. Therefore, the Tracking Port Analyzer (TPA) should not enable tracking (using the TRACE_IOEN bit) in the system reset state because the bit width of the synchronization frame packet in the reset state is different from the packet after the reset.

36.7.4 TPIU Register

The TPIU APB registers can be read or written to only when bit TRCENA of the Debug Exception and Monitoring Control Register (DEMCR) is set to 1. Otherwise, these registers will be read to zero (the output of this bit enables the TPIU clock)

36.7.5 TPIU Configuration Example

- Set bit TRCENA in the Debug Exception and Monitoring Control Register (DEMCR) to 1
- Write the desired value to the TPIU Current Port Size Register (default value is 0x1 for 1-bit port size)
- Write 0x102 to TPIU formatter and refresh control registers (default value)

- Write TPIU to select the pin protocol to choose synchronous or asynchronous mode. Example:
0x2 indicates asynchronous NRZ mode (similar to USART)
- Write 0x20 to the MCUTRACECTL control register (bit IO_TRACEN) to allocate TRACE for asynchronous mode.
- Send TPIU sync packet (FF_FF_FF_7F) at this time
- Configure the ITM and write to the ITM excitation register to output the value

Version revision record

Version number	Revision Date	Revised content
Rev1.0	2019/11/12	Initial Release
Rev1.1	2020/01/10	<ol style="list-style-type: none">1) Add 256KB of product descriptions to the full text;2) Addition of VFBGA package descriptions throughout the text;3) Initialization configuration (ICG) description pen error modification;4) Control register (WDT_CR) pen error modification;5) Table 34-1 CRC_RESTLT modified to CRC_RESLT;6) The current max value for 105°C in power-down mode is modified in the electrical characteristics.
Rev1.2	2020/08/26	<ol style="list-style-type: none">1) Add description of ultra-high speed operation mode, update CoreMark/DMIPS, add switching flow between ultra-high speed simulation, high speed mode and ultra-low speed mode, add BOR/PVD characteristics and current characteristics in ultra-high speed mode. Update the wait cycle of SRAM and Flash at 200Mhz, and the wait cycle of read port. Updating the frequency values in the bus architecture, updating the functional block diagram;2) Pin configuration diagram to add 256KB models;3) Adding an AOS chapter to increase the common trigger source enable bit of each AOS target register;4) Figure 6.1 TCK_SWCLK modified to JTCK_SWCLK and TCK modified to JTCK; 6.10.1 Correction of typos; 6.11.7 Mnemonic modification of bits 1-0 of CMU_XTAL32NFR; 6.11.13 "Frequency calibration within the LRC frequency guarantee" was corrected by a typo; 6.11.15 Mnemonic modification of bits 27-24 of CMU_PLLCFGR;5) 7.4.3 Add a description of the AD function enable bit corresponding to the STOP mode predecessor bit PWR_FCG3 to reduce power consumption;6) 7.7 Update PWR_PWRC3/PWR_XTAL32CS/PWR_STPMCR bit value; deletion of the underline in the WKE_n_m bit in PWR_PDWKE0/PWR_PDWKE1; deletion of the XTAL32

		<p>stop wake-up related bit in PWR_PDWKF1/PWR_PDWKE2; more detailed functional description of PWR_FCG0/PWR_FCG1/PWR_FCG2;</p> <p>7.7.13 Unified PWR_FCG register, 10. 2SRAM register in Ret-</p> <p>The SRAM/SRAMHS/SRAMECC label;</p> <p>The way the reserved bits are described in the Unified Power Control (PWC);</p> <p>Update the PVD2DETFLG and PVD1DETFLG bits in the PWR_PVDDSR register</p> <p>The clearance instructions;</p> <p>7) 9.6.3 Modification of step 10;</p>
--	--	---

Version number	Revision Date	Revised content
		<p>9.9.2 b1 FSLP modified to FSTP;</p> <p>9.9.3 b24 CRST0 modified to CRST;</p> <p>9 Summary erase modified to sector erase;</p> <p>9.9.7RDCOLERRITE modified to COLERRITE and read conflict modified to read-write conflict;</p> <p>8) 12.3 Modify USARTx_EI /USARTx_RTO in the interrupt vector table to USARTx_REI/USARTx_RT OI, respectively;</p> <p>12.4.9 Add "Using internal trigger events requires clearing the PWR_FCG0.AOS bit to enable the peripheral circuit trigger function";</p> <p>12.5.2 INT_NMIENR.PREENR modified to INT_NMIENR.REPENR and INT_NMIENR.RDEDENR modified to INT_NMIENR.RECCENR;</p> <p>12.5.3 INT_NMIFR.RPEFR modified to INT_NMIFR.REPFR and INT_NMIFR.RDEDFFR modified to INT_NMIFR.RECCFFR;</p> <p>12.5.4 INT_NMICFR.RPECFR modified to INT_NMICFR.REPCFR and INT_NMICFR.RDEDCFR modified to INT_NMICFR.RECCCFFR;</p> <p>12.5.6 INT_EIFR Removal of the INT_ prefix from the register bit description;</p> <p>12.5.7 Removal of the INT_ prefix from the INT_EICFR register bit description;</p> <p>12.5.10 Addition of the prefix V to the INT_VSSEL register bit description;</p> <p>12.5.14 Removal of the INT_ prefix from the INT_IER register bit description;</p> <p>9) 15.3 Examples of updated applications;</p> <p>15.4.7 Updating register descriptions;</p> <p>15.3.1/15.3.2 Removal of illegal access action descriptions;</p> <p>15.4.1 Register tag error modification;</p> <p>15.4.3 b17/b16/b9/b8/b1/b0 tagging and functional error</p>

		<p>modification;</p> <p>10) 16.4 Examples of update applications;</p> <p>11) 18.5.3 Addition of the description of the corresponding AD function enable bit of the pre-bit PWR_FCG3 for entering STOP mode in order to reduce power consumption;</p> <p>12) In chapter 20, optimize the Timer6 section legend; add chapters [pulse width measurement], [cycle measurement], [typical application example]; add cycle by cycle description for EMB control;</p> <p>13) 21.5.3 CCSR.DCLK modified to CCSR.CKDIV;</p>
--	--	--

Version number	Revision Date	Revised content
		<p>14) Optimization of the Timer0 and TimerA part legends in chapters 23 and 24;</p> <p>Modify the description of TimerA CCONR registers bit9~8 and PCONR registers bit1~0 in Chapter 23;</p> <p>15) 25.5.3 Omission of ALMF in the RTC_CR2 register bit description;</p> <p>16) 27.5.3 The initial value modification bit 0x0000 FFFF of USART_BRR;</p> <p>27.5.6 BCN bit-width match in the USART_CR3 register table;</p> <p>17) 32.4.3 RAIE modified to RAFIE;</p> <p>32.5.7 Bit5 mnemonic modification of CAN_RTIE;</p> <p>32.5.8 Bit5 mnemonic modification of CAN_RTIF;</p> <p>32.5.9 CAN_ERRINT form b5 WPIE modified to EPIE;</p> <p>32.5.12 CAN_AFWL modified to CAN_LIMIT;</p> <p>32 Full Chapter TRG_TRIG modified to TT_TRIG and TRG_WTRIG modified to TT_WTRIG;</p> <p>32.5.12 CAN_AFWL modified to CAN_LIMITt;</p> <p>18) Chapter 37 adds the MMC mode enable register description;</p> <p>19) 33.6.2 HFI modified to HFIR;</p> <p>Add 33.7.2.1 USBFS VBUS control register;</p> <p>33.7.2.5 b31 WKUPINT to WKUINT and b7 GOUTNAKEFF to GONAKEFF;</p> <p>20) 33.7.2.6 b31 WKUPINTM modified to WKUIMb30 VBUSVM modified to VBUSWIM,b29 DISCM modified to DISCIM,b28 CIDSCHG modified to CIDSCHGM, b7 GOUTNAKEFFM modified to GONAKEFFM;</p> <p>33.7.3.4 PTXQTOP is modified to PTXQSAV;</p> <p>33.5.4.4 after 15ms modified to within 15ms;</p> <p>33.7.4.8 INEPxkTXFEM modified to INEPTXFEM;</p> <p>33.7.4.2 Deleting TCTL bits from a form;</p> <p>33.7.4.11 TO modified to TOC ITTXFE modified to TTXFE;</p> <p>21) 38.4.1 Update the JTAG/SWJ debug port pins.</p>

Rev1.3	2021/12/10	<ul style="list-style-type: none">1) Deletion of product characteristics, pin configuration, package information, etc. (please refer to the latest datasheet for relevant information) and modification of statements;2) Updating some of the names and optimizing the descriptions, and updating the penmanship;3) Additional description of the accuracy of the UART communication;4) Description of the initialization method for timer6 to add the interval cycle validity function;
--------	------------	---

Version number	Revision Date	Revised content
		<p>5) Addition of notes on CAN usage and sampling points;</p> <p>6) 18.3.4 The digital filtering function of this group of ports is implemented between unit 1 -> the digital filtering function of this group of ports is set by the FCONR of unit 1, and the FCONR of other units is not valid for the digital filtering function setting of this group of ports, so any unit using this function needs to set the TIMER6_1 position bit in the function controller (PWR_FCG2);</p> <p>7) 19.5.7 TMR4_OCERn b13,b12 Description modification;</p>
Rev1.4	2021/12/30	Add HC32F45x series, modify Figure 5-6,Figure5-8,Figure18-13 and Table 18-3.
Rev1.41	2022/03/09	Company logo updated.

Rev1.42	2022/03/29	<p>1) Add the description of the data security protection address in Table 1-1, replace "ROM" with "FLASH";</p> <p>2) 4.11.1 b7 bit name pen error modification; CMU XTAL configuration registers b5~b4 Functional description modification;</p> <p>3) 6.1 "Address 0x0000_0408~0x0000_041F is reserved function" to "Address Address 0x0000_0408~0x0000_040F and 0x0000_0418~0x0000_041F are reserved functions"</p> <p>6.2.3 Initialization of the configuration of the parasite split description, new</p> <p>6.2.4 / 6.2.5 / 6.2.6 Subsections;</p> <p>4) 7.2 Modify the description of support for data security protection in the main features; 7.3 Add data security protection address and related description in FLASH map (Figure 7-1, Figure 7-2). ;</p> <p>7.6.5 A new description subsection on data security protection;</p> <p>5) 18.3.11.3 Add the description "When using this function, it can only be implemented by a combination of units 1 and 2, with unit 1 as the position counting unit and unit 2 as the revolution counting unit"</p> <p>6) 20.3.1 EMB_CTL0 register bits PWMSEN[2:0], CMPEN[2:0] Description of the modification; 20.3.2 EMB_CTL1~3 register bits CMPEN[2:0] description modification; 20.3.3 EMB_PWMLV0 register bit PWMLV description modification;</p> <p>7) 21.3.1.3 Figure 21-5 Correction of a pen error;</p> <p>8) 25.5.1 b1 "CLR.CFE" is modified to "CR1.CFE"; 25.5.4 b20 "CRTOF bit write 1 clear CRTOF" is changed to "CRTOF bit Write 1 to clear the RTOF";</p> <p>9) 26.1 Correction of written errors; 26.2.1 Figure 26-1 Arrow marking error modification;</p> <p>10) 30.4.1 Adding a description of the CAN communication clock constraints;</p>
---------	------------	--

Version number	Revision Date	Revised content
		<p>30.4.5 Adding a description of data sending constraints;</p> <p>30.5.1 The offset address "0x50" is changed to "0x00"; and "mailbox" is changed to "message";</p> <p>30.5.2 The offset address "0x00" is modified to "0x50";</p> <p>30.5.8 b6 bit functional description modification;</p> <p>30.5.9 bit b7 R/W modification; bit b6,b0 functional description modification;</p> <p>30.5.11 Bits b4~b0 R/W modification;</p> <p>30.5.13 Bits b7~b0 R/W modification;</p> <p>30.5.14 Bits b7~b0 R/W modification;</p> <p>30.5.15 b3~b0 Functional description modification;</p> <p>30.5.19 b5 bit function description modification.</p>
Rev1.5	2022/09/14	<ol style="list-style-type: none">1) Addition of HC32A460 series product models;2) The introduction section "HC32F460" is amended to read "HC32F460_F45x_A460"; Add a description of "suitable for automotive electronics";3) 18.3.4 Modification of the digital filtering function description;4) 19.3.1.3 Figure 19-6 Original sawtooth wave with flat peaks modified to spikes; 19.3.3.1 Figure 19-16 Original sawtooth wave with flat peaks modified to spikes; 19.3.3.2 Figure 19-18 Original sawtooth wave with flat peaks modified to spikes;19.5.3 Modification to the description of "Note 2" in the functional description register TMR4_CCSR bit b15;5) 23.3.1 RTC register reset description modification; 23.5.14 A typographical error "hourly alarm register" was changed to "weekly alarm register";6) 25.5.1 Register USART_SR Bit b1 Pen error modification in the functional description;7) 27.12.3 Pen error modification in the functional description of register SPI_CFG1 bits b30~b28;28.3.1 Correction of pencil errors in Figure 28-4 and Figure 28-7;28.4.5 Correction of written errors;

		<p>28.5.6 Correction of pen errors in Figure 28-20;</p> <p>8) 30.4.17 Receive BUF full interrupt flag pen error modification;</p> <p>30.5.18 Register CAN_TBSLOT bit b6 to mark pen error modification;</p> <p>30.5.23 Register CAN_TT_WTRIG bits b15~b0 Marking pen error modification;</p> <p>9) 33.4.4 Register DCU_FLAGCLR read/write modified to "R/W".</p>
--	--	--

feel free to contact us.

E-mail: support@xhsc.com.cn 电话:

021-68667000-7355

Address: 10/F, Block A, 1867 Zhongke Road, Pudong New Area, Shanghai



XIAOHUA SEMICONDUCTOR CO., LTD