



HC32A4A0 Series

32-bit ARM® Cortex® -M4 Microcontroller

Reference Manual

Rev1.0 November 2022

Statement

- ★ Xiaohua Semiconductor Co., Ltd. (hereinafter referred to as "XHSC") reserves the right to change, correct, enhance, modify Xiaohua Semiconductor products and / or this document at any time without prior notice. Users can get the latest information before placing orders. XHSC products are sold in accordance with the terms and conditions of sale set forth in the Basic Contract for Purchase and Sales.
- ★ It is the customer's responsibility to select the appropriate XHSC product for your application and to 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 intellectual property license has been granted by express or implied means.
- ★ The resale of XHSC Products shall be invalidated by any warranty commitment of XHSC to such Products if its terms are different from those set forth herein.
- ★ Any graphics or words marked “®” or “™” are trademarks of XHSC. All other products or services shown on XHSC products are the property of their respective owners.
- ★ Information in this notification replaces and replaces information in previous versions.

©2022 Xiaohua Semiconductor Co., Ltd. All rights reserved

Table of Contents

Statement	2
Table of Contents.....	3
Table Index.....	39
Graph Index	45
Overview.....	57
1 Memory Mapping	58
1.1 Memory mapping	58
1.2 External memory mapping	64
1.3 Bit band region.....	64
1.4 Address remapping	64
1.5 Remap register.....	66
1.5.1 Access Protection Register (MMF_REMPRT).....	66
1.5.2 Remap Register 0 (MMF_REMCR0)	67
1.5.3 Remap Register 1 (MMF_REMCR1).....	68
2 Bus Architecture (BUS).....	69
2.1 Overview	69
2.2 Bus architecture	70
2.3 Bus function	71
3 Reset control (RMU).....	72
3.1 Introduction	72
3.2 Reset source and reset flag bit.....	73
3.3 Reset timing	75
3.3.1 Power-on reset	75
3.3.2 NRST pin reset	75
3.3.3 Brown-out reset	76
3.3.4 Programmable voltage detection 1 reset, programmable voltage detection 2 reset	
76	
3.3.5 Watchdog reset, dedicated watchdog reset.....	78
3.3.6 Power-down wake-up reset	78
3.3.7 Software reset.....	79
3.3.8 MPU error reset.....	79
3.3.9 RAM parity reset	80
3.3.10 RAM ECC reset	80
3.3.11 Clock frequency abnormal reset	80
3.3.12 External High-Speed Oscillator Fault Reset.....	81
3.3.13 Cortex-M4 Lockup Reset	81

3.3.14	Identification of the reset cause	82
3.3.15	Reset conditions for individual modules	83
3.4	Register description.....	84
3.4.1	Reset Control Register (RMU_PRSTCR0).....	85
3.4.2	Reset Flag Register 0 (RMU_RSTF0).....	86
4	Clock controller (CMU)	88
4.1	Introduction.....	88
4.2	System block diagram.....	89
4.2.1	System block diagram	89
4.2.2	Clock Frequency Measurement Block Diagram	90
4.3	clock source specification.....	91
4.4	Working clock specification	92
4.5	Crystal oscillator circuit	95
4.5.1	External high-speed oscillator	95
4.5.2	External high-speed oscillator fault detection	96
4.5.3	External low-speed oscillator	99
4.6	Internal RC clock.....	100
4.6.1	HRC clock.....	100
4.6.2	MRC clock	100
4.6.3	LRC clock	101
4.6.4	SWDTLRC clock.....	101
4.6.5	RTCLRC clock	101
4.7	PLL clock.....	102
4.8	Clock switching step.....	102
4.8.1	Clock switch.....	103
4.8.2	Clock division switching	104
4.9	Clock output function	105
4.10	Clock frequency measurement (FCM).....	106
4.10.1	Clock frequency measurement	106
4.10.2	Digital filter function	107
4.10.3	Interrupt/Reset Function	107
4.11	Register description.....	108
4.11.1	CMU XTAL Configuration Register (CMU_XTALCFGR).....	110
4.11.2	CMU XTAL Configuration Register (CMU_XTALSTBCR)	110
4.11.3	CMU XTAL Control Register (CMU_XTALCR)	111
4.11.4	CMU XTAL Oscillatory Fault Control Register (CMU_XTALSTDRCR)	112
4.11.5	CMU XTAL Oscillatory Fault State Register (CMU_XTALSTDTSR).....	113
4.11.6	CMU XTAL32 Configuration Register (CMU_XTAL32CFGR).....	113

4.11.7	CMU XTAL32 Filter Register (CMU_XTAL32NFR).....	113
4.11.8	CMU XTAL32 Control Register (CMU_XTAL32CR).....	114
4.11.9	CMU HRC Calibration Register (CMU_HRCTRM).....	114
4.11.10	CMU HRC control register (CMU_HRCCR)	115
4.11.11	CMU MRC Calibration Register (CMU_MRCTRM)	115
4.11.12	CMU MRC Control Register (CMU_MRCCR)	116
4.11.13	CMU LRC Calibration Register (CMU_LRCTRM)	116
4.11.14	CMU LRC control register (CMU_LRCCR).....	117
4.11.15	CMU RTCLRC Calibration Register (CMU_RTCLRCTRM)	117
4.11.16	CMU PLLH Configuration Register (CMU_PLLHCFGR)	118
4.11.17	CMU PLLH Control Register (CMU_PLLHCR).....	120
4.11.18	CMU PLLA Configuration Register (CMU_PLLACFGR).....	121
4.11.19	CMU PLLA Control Register (CMU_PLLACR)	123
4.11.20	CMU clock Stabilizer (CMU_OSCSTBSR)	124
4.11.21	Clock Switch Register of System (CMU_CKSWR).....	124
4.11.22	CMU Clock Divider Configuration Register (CMU_SCFGR)	125
4.11.23	CMU USB Clock Configuration Register (CMU_USBCKCFGR)	127
4.11.24	CMU CAN Clock Configuration Register (CMU_CANCKCFGR).....	128
4.11.25	CMU AD/TRNG/DA Clock Configuration Register (CMU_PERICKSEL).....	129
4.11.26	CMU Debug Clock Configuration Register (CMU_TPIUCKCFGR)	129
4.11.27	CMU MCO1 Configuration Register (CMU_MCO1CFGR)	130
4.11.28	CMU MCO2 Configuration Register (CMU_MCO2CFGR)	130
4.11.29	FCM Lower Limit Compare Value Register (FCM_LVR)	131
4.11.30	FCM Upper Limit Compare Value Register (FCM_UVR)	131
4.11.31	FCM Counter Value Register (FCM_CNTR)	132
4.11.32	FCM Start Stop Register (FCM_STR)	132
4.11.33	FCM Measurement Object Control Register (FCM_MCCR)	133
4.11.34	FCM Measurement Reference Control Register (FCM_RCCR).....	134
4.11.35	FCM Interrupt Reset Control Register (FCM_RIER).....	135
4.11.36	FCM Flag Register (FCM_SR)	136
4.11.37	FCM Flag Clear Register (FCM_CLR)	136
5	Power controller (PWC)	137
5.1	Introduction	137
5.2	Distribution of power	138
5.2.1	Battery backup power domain.....	139
5.3	Description of Power Supply Voltage Detection Unit (PVD)	141
5.3.1	Description of the action of power-on/power-down reset.....	141
5.3.2	Description of Brown-out Reset (BOR)	141

5.3.3	Programmable voltage detection 1 (PVD1), programmable voltage detection 2 (PVD2)	142
5.3.4	PVD1, PVD2 Interrupt/Reset Block Diagram.....	144
5.3.5	Input/output pin	144
5.3.6	PVD1 Interrupt and Reset	144
5.3.7	PVD2 Interrupt and Reset	145
5.3.8	Internal voltage sampling and detection function	146
5.4	Wakeup timer	148
5.5	Operation Mode and Low Power Mode.....	148
5.5.1	Operating mode.....	152
5.5.2	Sleep mode.....	153
5.5.3	Stop mode	153
5.5.4	Power-down mode.....	155
5.5.5	Reduce system clock speed.....	158
5.5.6	Turn off unused clock source.....	158
5.5.7	Stop functional clock	159
5.5.8	Turn off unused RAM	159
5.6	Register protection function	160
5.7	Register description.....	161
5.7.1	Power Mode Control Register 0 (PWC_PWRC0)	166
5.7.2	Power Mode Control Register 1 (PWC_PWRC1)	166
5.7.3	Power Mode Control Register 2 (PWC_PWRC2)	167
5.7.4	Power Mode Control Register 3 (PWC_PWRC3)	167
5.7.5	Power Mode Control Register 4 (PWC_PWRC4)	168
5.7.6	Power-down Wake-up Enable Register 0 (PWC_PDWKE0)	168
5.7.7	Power-down Wake-up Enable Register 1 (PWC_PDWKE1)	169
5.7.8	Power-down Wake-up Enable Register 2 (PWC_PDWKE2)	169
5.7.9	Power-Down Wake-Up Event Edge Select Register (PWC_PDWKES).....	170
5.7.10	Power-down Wake-up Flag Register 0 (PWC_PDWKF0).....	170
5.7.11	Power-down Wake-up Flag Register 1 (PWC_PDWKF1).....	171
5.7.12	Function Clock Control 0 (PWC_FCG0)	172
5.7.13	Function Clock Control 1 (PWC_FCG1)	174
5.7.14	Function Clock Control 2 (PWC_FCG2)	176
5.7.15	Function Clock Control 3 (PWC_FCG3)	178
5.7.16	PWC_FCG0 Protection Control (PWC_FCG0PC)	180
5.7.17	Function Protection Control Register (PWC_FPRC).....	180
5.7.18	STOP Mode Control Register (PWC_STPMCR)	181
5.7.19	RAM Power Control Register 0 (PWC_RAMPC0)	182

5.7.20	RAM Operating Conditions Register (PWC_RAMOPM)	182
5.7.21	Peripheral RAM Low Power Control Register (PWC_PRAMLPC)	183
5.7.22	PVD Control Register 0 (PWC_PVDCR0)	184
5.7.23	PVD Control Register 1 (PWC_PVDCR1)	184
5.7.24	PVD Filter Control Register (PWC_PVDFCR)	185
5.7.25	PVD Level Control Register (PWC_PVDLCR)	185
5.7.26	PVD Interrupt Control Register (PWC_PVDICR)	186
5.7.27	PVD Detection Status Register (PWC_PVDDSR)	186
5.7.28	Backup Domain Reset Register (PWC_VBATORSTR)	187
5.7.29	Backup Domain Control Register (PWC_VBATCR)	187
5.7.30	VBAT Backup Registers 0~127 (PWC_BKR0~PWC_BKR127)	187
5.7.31	Wake-up Timer Control Register 0 (PWC_WKTC0)	188
5.7.32	Wake-up Timer Control Register 1 (PWC_WKTC1)	188
5.7.33	Wake-up Timer Control Register 2 (PWC_WKTC2)	188
6	Initial Configuration (ICG)	189
6.1	Introduction	189
6.2	Register description.....	190
6.2.1	Initial configuration register0 (ICG0).....	190
6.2.2	Initial configuration register1 (ICG1).....	192
6.2.3	Initial configuration register3 (ICG3).....	192
7	Embedded FLASH (EFM)	193
7.1	Introduction	193
7.2	Main features.....	193
7.3	Embedded FLASH	193
7.4	Read interface	195
7.4.1	Relationship between CPU clock and FLASH read cycle.....	195
7.5	FLASH Read Acceleration	195
7.6	FLASH program and erase.....	196
7.6.1	Unlock EFM_KEY1 register.....	196
7.6.2	Write protection	197
7.6.3	Single program without readback.....	197
7.6.4	Single program with readback	197
7.6.5	Series program	198
7.6.6	Sector Erase	199
7.6.7	Mass erase.....	199
7.6.8	Data security configuration	200
7.6.9	D-BUS read protection	201
7.6.10	BGO	201

7.6.11	Interruption.....	202
7.7	One Time Program (OTP).....	203
7.8	Boot swap.....	204
7.9	Register description.....	207
7.9.1	Access protection register (EFM_FAPRT)	208
7.9.2	FLASH KEY 1 Register (EFM_KEY1).....	208
7.9.3	FLASH KEY 2 register (EFM_KEY2).....	208
7.9.4	FLASH Stop Register (EFM_FSTP)	209
7.9.5	Read mode register (EFM_FRMC)	209
7.9.6	Erase mode register (EFM_FWMC)	210
7.9.7	Status register (EFM_FSR).....	211
7.9.8	Status purge register (EFM_FSCLR).....	213
7.9.9	Interruption enable register (EFM_FITE).....	214
7.9.10	Boot Swap Status Register (EFM_FSWP)	214
7.9.11	CHIPID register (EFM_CHIPID)	215
7.9.12	Unique ID register (EFM_UQID 0)	215
7.9.13	Unique ID register (EFM_UQID 1)	215
7.9.14	Unique ID register (EFM_UQID 2)	216
7.9.15	FLASH write protection lock register (EFM_WLOCK).....	217
7.9.16	FLASH0 write protection register 0 (EFM_F0NWPRT0)	218
7.9.17	FLASH0 write protection register 1 (EFM_F0NWPRT1)	219
7.9.18	FLASH0 write protection register 2 (EFM_F0NWPRT2)	220
7.9.19	FLASH0 write protection register 3 (EFM_F0NWPRT3)	221
7.9.20	FLASH1 write protection register 0 (EFM_F1NWPRT0)	222
7.9.21	FLASH1 write protection register 1 (EFM_F1NWPRT1)	223
7.9.22	FLASH1 write protection register 2 (EFM_F1NWPRT2)	224
7.9.23	FLASH1 write protection register 3 (EFM_F1NWPRT3)	225
7.10	Attention.....	226
8	Built-in SRAM (SRAM).....	227
8.1	Introduction	227
8.2	Register description.....	229
8.2.1	SRAM Wait Control Register (SRAM_WTCR)	230
8.2.2	SRAM Waiting Protection Register (SRAM_WTPR).....	232
8.2.3	SRAM Check Control Register (SRAM_CKCR)	233
8.2.4	SRAM Check Protection Register (SRAM_CKPR).....	234
8.2.5	SRAM Check Status Register (SRAM_CKSR)	235
9	General IO (GPIO)	236
9.1	Introduction	236

9.2	Port function summary	237
9.3	Functional Description	238
9.3.1	General Purpose Input/Output GPIO function.....	238
9.3.2	Peripheral functions	239
9.3.3	Dual peripheral function	239
9.3.4	Event Port input and output function.....	239
9.3.5	External interrupt EIRQ input function	240
9.3.6	Analogfunction.....	240
9.3.7	General control	240
9.4	Register description.....	242
9.4.1	General input register (PIDRx)	244
9.4.2	General output data register (PODRx)	244
9.4.3	General output enable Register (POERx)	245
9.4.4	General output set register (POSRx).....	245
9.4.5	General output reset register (PORRx)	245
9.4.6	General output toggle register (POTRx).....	246
9.4.7	Special function control register (PSPCR).....	246
9.4.8	Common control register (PCCR)	247
9.4.9	Input always enable register (PINAER).....	248
9.4.10	Write protection register (PWPR)	248
9.4.11	General control register (PCRxy)	249
9.4.12	Function select register (PFSRxy).....	250
9.4.13	Event port trigger source selection register (PEVNTTRGSR12, PEVNTTRGSR34) ...	251
9.4.14	Event port direction selection register (PEVNTDIRRm)	252
9.4.15	Event port input data register (PEVNTIDRm)	252
9.4.16	Event port output data register (PEVNTODRm)	253
9.4.17	Event port output data reset register (PEVNTORRm).....	253
9.4.18	Event port output data set register (PEVNTOSRm)	254
9.4.19	Event port rising edge input enable register (PEVNTRISRm)	254
9.4.20	Event port falling edge input enable register (PEVNTFALRm)	255
9.4.21	Event port input noise filter control register (PEVNTNFCR).....	256
9.4.22	32bit access.....	257
9.5	Precautions.....	258
10	Interrupt controller (INTC).....	259
10.1	Overview	259
10.2	Block Diagram	260
10.2.1	Interrupt system block diagram.....	260
10.3	Vector Table	261

10.3.1	Interrupt Vector Table	261
10.3.2	Event Numbers	266
10.4	Functions	280
10.4.1	Non-maskable interrupt	280
10.4.2	External interrupt.....	280
10.4.3	Interrupt source selection.....	281
10.4.4	Software interrupt.....	281
10.4.5	Interrupt/Event Selection	281
10.4.6	WFE Wake-up Event Management	282
10.4.7	Noise filter	283
10.4.8	Low power mode return	284
10.5	Registers	285
10.5.1	External Interrupt Filter Control Register (INTC_NOCCR).....	286
10.5.2	Non-Maskable Interrupt Enable Register (INTC_NMIENR).....	286
10.5.3	Non-Maskable interrupt flag register (INTC_NMIFR)	287
10.5.4	Non-Maskable Interrupt Flag Clear Register (INTC_NMICFR)	288
10.5.5	External Interrupt Control Register (INTC_EIRQCR _x , x=0~15)	289
10.5.6	External Interrupt Flag Register (INTC_EIFR).....	290
10.5.7	External Interrupt Flag Clear Register (INTC{EIFCR})	290
10.5.8	Interrupt Selection Register (INTC_SEL _x , x=0~127)	291
10.5.9	Vector Shared Interrupt Selection Register (INTC_VSEL _x , x=128~143)	293
10.5.10	Stop Mode Wake-up Enable Register (INTC_WUPEN).....	294
10.5.11	Software Interrupt Register (INTC_SWIER)	295
10.5.12	Event Enable Register (INTC_EVTER)	295
10.5.13	Interrupt Enable Register (INTC_IER).....	295
10.6	Precautions.....	296
11	Automatic Operation System (AOS).....	297
11.1	Introduction	297
11.1.1	Function Overview	297
11.1.2	Module Diagram	298
11.2	Functional Description	299
11.2.1	AOS Source Event List	299
11.2.2	AOS Target List.....	299
11.3	Action Description	300
11.3.1	Dedicated Trigger Source.....	300
11.3.2	Common Trigger Source.....	300
11.4	Register Description	301
11.4.1	Register Overview.....	301

11.4.2	Peripheral Trigger Event Register (INTSFTTRG)	302
11.4.3	DCU Trigger Source Selection Register (DCU_TRGSELx) (x=1~4).....	303
11.4.4	DMA1 Transfer Start Trigger Source Selection Register (DMA1_TRGSELx) (x=0~7) 304	
11.4.5	DMA2 Transfer Start Trigger Source Selection Register (DMA2_TRGSELx) (x=0~7) 305	
11.4.6	DMA Channel Re-configure Trigger Source Selection Register (DMA_TRGSELRC) ..	306
11.4.7	Timer6 Trigger Source Selection Register (TMR6_HTSSRx) (x=0~3).....	307
11.4.8	Event Port Trigger Source Selection Register (PEVNTRGSR12, PEVNTRGSR34)..	308
11.4.9	Timer0 Trigger Source Select Register (TMR0_HTSSR).....	309
11.4.10	Timer2 Trigger Source Select Register (TMR2_HTSSR).....	310
11.4.11	HASH Trigger Source Selection Register A (HASH_ITRGSELA)	311
11.4.12	HASH Trigger Source Selection Register B (HASH_ITRGSELB)	312
11.4.13	TimerA Trigger Source Selection Register (TMRA_HTSSRx) (x=0~3)	313
11.4.14	OTS Trigger Source Selection Register (OTS_TRG).....	314
11.4.15	ADC1 Conversion Start Trigger Source Selection Register ADC1_ITRGSELRx (x=0, 1) 315	
11.4.16	ADC2 Conversion Start Trigger Source Selection Register ADC2_ITRGSELRx (x=0, 1) 316	
11.4.17	ADC3 Conversion Start Trigger Source Selection Register ADC3_ITRGSELRx(x=0,1) 317	
11.4.18	Common Trigger Source Selection Register 1 (AOS_COMTRG1).....	318
11.4.19	Common Trigger Source Selection Register 2 (AOS_COMTRG2).....	318
12	Storage Protection Unit (MPU).....	319
12.1	Introduction	319
12.2	Functional description	320
12.2.1	Locale settings.....	320
12.2.2	Permission settings.....	320
12.2.3	MPU action selection.....	320
12.2.4	Start the MPU.....	320
12.3	Application examples	321
12.3.1	Only allow partial space access	321
12.3.2	Only block access to some spaces.....	321
12.4	Register description.....	322
12.4.1	Area range description register MPU_RGDn (n=0~15)	323
12.4.2	Status flag register MPU_SR.....	323
12.4.3	Flag Clear Register MPU_ECLR	324

12.4.4	Write protection register MPU_WP	324
12.4.5	SMPU1 area enable register MPU_S1RGE.....	325
12.4.6	SMPU1 area write permission register MPU_S1RGWP	326
12.4.7	SMPU1 area read permission register MPU_S1RGWP.....	327
12.4.8	SMPU1 Control Register MPU_S1CR	328
12.4.9	SMPU2 area enable register MPU_S2RGE.....	329
12.4.10	SMPU2 area write permission register MPU_S2RGWP	330
12.4.11	SMPU2 area read permission register MPU_S2RGWP.....	331
12.4.12	SMPU2 Control Register MPU_S2CR	332
12.4.13	FMPU region enable register MPU_FRGE	332
12.4.14	FMPU area write permission register MPU_FRGWP	333
12.4.15	FMPU area read permission register MPU_FRGRP	333
12.4.16	FMPU control register MPU_FCR	334
12.4.17	HMPU area enable register MPU_HRGE	334
12.4.18	HMPU area write permission register MPU_HRGWP	335
12.4.19	HMPU area read permission register MPU_HRGRP	335
12.4.20	HMPU control register MPU_HCR	336
12.4.21	EMPU area enable register MPU_ERGE	336
12.4.22	EMPU area write permission register MPU_ERGWP	337
12.4.23	EMPU area read permission register MPU_ERGRP	337
12.4.24	EMPU control register MPU_ECR	338
12.4.25	IP Access Protection Register MPU_IPPR.....	339
13	Keyboard Scanning Control Module (KEYSCAN)	341
13.1	Introduction	341
13.2	KEYSCAN system block diagram.....	341
13.3	Pin description	342
13.4	Functional description	342
13.4.1	Key recognition function	342
13.4.2	Keyboard scan function	342
13.4.3	Precautions for use	343
13.5	Register description.....	344
13.5.1	KEYSCAN Scan Control Register (KEYSCAN_SCR).....	345
13.5.2	KEYSCAN Scan Enable Register (KEYCAN_SER).....	346
13.5.3	KEYSCAN Scan Status Register (KEYSCAN_SS)	346
14	Internal clock calibrator (CTC)	347
14.1	Introduction	347
14.2	Structural block diagram	347
14.3	Functional description	348

14.3.1	Reference clock	348
14.3.2	Frequency calibration	350
14.3.3	Programming guide	351
14.4	Register Description	352
14.4.1	Clock Calibration Control Register1 (CTC _ CR1).....	353
14.4.2	Clock Calibration Control Register2 (CTC _ CR2).....	354
14.4.3	Clock calibration status register (CTC _ STR)	355
15	DMA controller (DMA)	356
15.1	Introduction.....	356
15.2	Module diagram	357
15.3	Functional description	358
15.3.1	Enable DMA controller	358
15.3.2	Channel selection and channel priority.....	358
15.3.3	Start DMA	358
15.3.4	Data block.....	358
15.3.5	Address control	358
15.3.6	Number of transfers.....	359
15.3.7	Interrupt and event signal output.....	360
15.3.8	Linked list transfer	360
15.3.9	Nonsequence address mode.....	361
15.3.10	Reconfigure transfer	363
15.3.11	Stop DMA transfer.....	365
15.4	Application examples	366
15.4.1	Memory-to-memory transfer.....	366
15.4.2	Memory-to-peripheral transfer.....	367
15.4.3	Memory-to-memory with LLP enabled	368
15.5	Registers	371
15.5.1	Register list.....	371
15.5.2	DMA Enable Register (DMA_EN).....	373
15.5.3	Interrupt State Register 0 (DMA_INTSTAT0)	373
15.5.4	Interrupt State Register 1 (DMA_INTSTAT1)	374
15.5.5	Interrupt Mask Register 0 (DMA_INTMASK0)	374
15.5.6	Interrupt Mask Register 1 (DMA_INTMASK1)	375
15.5.7	Interrupt Clear Register 0 (DMA_INTCLR0)	375
15.5.8	Interrupt Clear Register 1 (DMA_INTCLR1)	376
15.5.9	Channel Enable Register (DMA_CHEN).....	376
15.5.10	Channel Enable Clear Register (DMA_CHENCLR)	377
15.5.11	Reconfigure Control Register (DMA_RCFGCTL).....	378

15.5.12 Transfer Request Status Register (DMA_REQSTAT).....	379
15.5.13 Transfer Status Monitor Register (DMA_CHSTAT)	379
15.5.14 Transfer Request Selection Register (DMA_TRGSELx) (x=0~7)	380
15.5.15 Reconfigure Request Selection Register (DMA_TRGSELRC).....	380
15.5.16 Source Address Register (DMA_SARx) (x=0~7).....	381
15.5.17 Transfer Target Address Register (DMA_DARx) (x=0~7)	381
15.5.18 Data Control Register (DMA_DTCTLx) (x=0~7).....	382
15.5.19 Repeat Control Register (DMA_RPTx) (x=0~7)	383
15.5.20 Repeat Control Register B (DMA_RPTBx) (x=0~7)	384
15.5.21 Source Address Nonsequence Control Cegister (DMA_SNSEQCTLx) (x=0~7).....	385
15.5.22 Source Address Nonsequence Control Register B (DMA_SNSEQCTLBx) (x=0~7) ..	386
15.5.23 Destination Address Nonsequence Control Register (DMA_DNSEQCTLx) (x=0~7)	387
15.5.24 Destination Address Nonsequence control register B (DMA_DNSEQCTLBx) (x=0~7)	388
15.5.25 Linked List Pointer Register (DMA_LLPx) (x=0~7)	389
15.5.26 Channel Control Register (DMA_CHCTLx) (x=0~7)	389
15.5.27 Channel Monitor Registers (DMA_MONSARx, DMA_MONDARx, DMA_MONDTCTLx, DMA_MONRPTx, DMA_MONSNSEQCTLx, DMA_MONDNSEQCTLx) (x=0~7)	390
15.6 Cautions	391
16 Voltage comparator (CMP).....	392
16.1 Introduction	392
16.2 Functional block diagram	393
16.3 Functional description	395
16.3.1 General comparison mode.....	395
16.3.2 Window comparison mode.....	396
16.3.3 Timer window output.....	397
16.3.4 Digital filter.....	397
16.3.5 Comparator interrupt.....	398
16.3.6 Peripheral triggering event	398
16.3.7 External pin output	398
16.4 Precautions.....	399
16.4.1 Module stop function	399
16.4.2 The act of stopping a module.	399
16.4.3 The act of stopping a low-power mode.....	399
16.5 Register description.....	400
16.5.1 Comparator mode setting register (CMPx_MDR, x=1~4)	401
16.5.2 Comparator Filter and Interrupt Control Register (CMPx_FIR, x=1~4)	402
16.5.3 Comparator output control register (CMPx_OCR, x=1~4)	403

16.5.4	Comparator positive and negative input selection register (CMPx_PMSR, x=1~4)	404
16.5.5	Comparator Voltage Input Source Selection Register (CMPx_VISR, x=1, 3)	405
16.5.6	Comparator timer window selection register (CMPx_TWSR, x=1~4)	407
16.5.7	Comparator Timer Window Polarity Register (CMPx_TWPR, x=1~4)	408
17	Analog-to-digital conversion module (ADC)	409
17.1	Introduction	409
17.2	ADC system block diagram	411
17.3	Functional description	413
17.3.1	ADC clock.....	413
17.3.2	Channel selection	413
17.3.3	Trigger source selection	415
17.3.4	Sequence A single scan mode	415
17.3.5	Sequence A continuous scanning mode	416
17.3.6	Double sequence scanning mode	418
17.3.7	Analog watchdog function	420
17.3.8	Sampling time and conversion time of analog input.....	421
17.3.9	Automatic clearing function of A/D data register.....	422
17.3.10	Converted data average calculation function	423
17.3.11	Programmable Gain Amplifier PGA	423
17.3.12	Dedicated sample and hold circuit SH	425
17.3.13	Multi ADC synchronized mode	426
17.3.14	Interrupt and event signal output.....	431
17.4	Register description.....	432
17.4.1	Register overview	432
17.4.2	ADC Start Register (ADC_STR)	434
17.4.3	ADC Control Register 0 (ADC_CR0)	435
17.4.4	ADC Control Register 1 (ADC_CR1)	436
17.4.5	ADC Trigger Select Register (ADC_TRGSR).....	437
17.4.6	ADC Trigger Source Select Register (ADC_ITRGSEL0, ADC_ITRGSEL1)	438
17.4.7	ADC Sequence A Channel Select Register(ADC_CHSELRA)	439
17.4.8	ADC Sequence B Channel Select Register (ADC_CHSELRB).....	440
17.4.9	ADC Average Channel Select Register (ADC_AVCHSEL).....	441
17.4.10	ADC Extend Channel Select Register (ADC_EXCHSEL).....	442
17.4.11	ADC Sampling Time register (ADC_SSTR x)	442
17.4.12	ADC Channel Mapping Register 0 (ADC_CHMUXR)	443
17.4.13	ADC Interrupt Flag Register (ADC_ISR).....	444
17.4.14	ADC Interrupt Enable Register (ADC_ICR)	444
17.4.15	ADC Interrupt Flag Clear Register (ADC_ISCLRR)	445

17.4.16	Multi ADC Synchronous Mode Control Register (ADC_SYNCCR)	446
17.4.17	ADC Channel Convert Data Register (ADC_DR)	447
17.4.18	ADC Analog Watch Dog Control Register (ADC_AWDCR).....	448
17.4.19	ADC Analog Watch Dog Status Register (ADC_AWDUSR)	449
17.4.20	ADC Analog Watch Dog Status Clear Register (ADC_AWDSCLRR).....	449
17.4.21	ADC Analog Watch Dog Compare Data Register (ADC_AWD0DR0, ADC_AWD0DR1, ADC_AWD1DR0, ADC_AWD1DR1)	450
17.4.22	ADC Analog Watch Dog Channel Select Register (ADC_AWD0CHSR, ADC_AWD1CHSR).....	450
17.4.23	ADC Dedicated SH Control Register(ADC_SHCR)	451
17.4.24	ADC PGA Control Register(ADC_PGACRx)	452
17.4.25	ADC PGA VSS Source Select Register (ADC_PGA_VSSENTR).....	453
17.5	Precautions for use.....	454
17.5.1	Considerations when reading data register	454
17.5.2	Scan Finish Interrupt Handling Considerations	454
17.5.3	Module Stop and Low Power Settings Considerations.....	454
17.5.4	Pin Setting for A/D Converter Analog Channel Input.....	454
17.5.5	Noise control.....	454
18	Digital-to-analog converter DAC	455
18.1	Introduction	455
18.2	Functional block diagram	456
18.3	Functional description	457
18.3.1	D/A conversion.....	457
18.3.2	Synchronous conversion.....	458
18.3.3	External data conversion	458
18.3.4	A/D conversion priority mode	458
18.4	Precautions.....	460
18.4.1	Setting of the module stop function	460
18.4.2	DAC operation when the module is stopped	460
18.4.3	Stop DAC action in low power mode	460
18.4.4	DAC Action in Power-Down Low-Power Mode	460
18.4.5	Precautions for Using Output Amplifiers	460
18.5	Register description.....	461
18.5.1	DAC data register (DADRx_y x,y=1,2)	462
18.5.2	DAC Control Register (DACRx, x=1,2).....	463
18.5.3	DAC Analog Output Control Register (DAOCRx, x=1,2)	464
18.5.4	DAC A/D conversion priority control register (DAADPCRx, x=1,2)	465
19	Temperature Sensor (OTS).....	466

19.1	Introduction	466
19.2	Usage notes	467
19.3	Register description.....	469
19.3.1	OTS Control Register (OTS_CTL)	470
19.3.2	OTS Data Register 1 (OTS_DR1)	470
19.3.3	OTS Data Register 2 (OTS_DR2)	470
19.3.4	OTS Error Compensation Register (OTS_ECR)	471
19.3.5	OTS Preset Temperature Data Registers (OTS_PDR1,2,3)	471
19.3.6	OTS trigger source selection register (OTS_TRG).....	471
20	Advanced Control Timer (Timer6)	472
20.1	Introduction	472
20.2	Basic block diagram	472
20.3	Function description	474
20.3.1	Waveform mode.....	474
20.3.2	Clock source selection	474
20.3.3	Counting direction	475
20.3.4	Comparison output	475
20.3.5	Input capture	476
20.3.6	Counter update.....	477
20.3.7	Software synchronization.....	478
20.3.8	Hardware synchronization	479
20.3.9	Pulse width measurement	481
20.3.10	Period measurement.....	482
20.3.11	Buffer function	483
20.3.12	Digital filtering	488
20.3.13	General PWM output.....	490
20.3.14	Periodic interval response.....	497
20.3.15	Quadrature encoding count	498
20.3.16	EMB control.....	503
20.3.17	Typical application example.....	504
20.3.18	Function Summary.....	510
20.4	Interrupt and Event Description	511
20.4.1	Interrupt output	511
20.4.2	Event output	512
20.5	Register description.....	514
20.5.1	Counter Value Register (TMR6_CNTER).....	516
20.5.2	Update Value Register (TMR6_UPDAR).....	516
20.5.3	Period Value Register (TMR6_PERmR) (m=A~C).....	517

20.5.4	General Compare Value Register (TMR6_GCMmR) (m=A~F)	517
20.5.5	Special Compare Value Register (TMR6_SCMmR) (m=A~F)	518
20.5.6	DeadTime Value Register (TMR6_DTmnR) (m=D, U&&n=A, B).....	518
20.5.7	General Control Register (TMR6_GCONR)	519
20.5.8	Interrupt Control Register (TMR6_ICONR)	520
20.5.9	Buffer Control Register (TMR6_BCONR)	521
20.5.10	DeadTime Control Register (TMR6_DCONR).....	523
20.5.11	Port Control A Register (TMR6_PCNAR)	524
20.5.12	Port Control B Register (TMR6_PCNBR)	526
20.5.13	Filter Control General Port Register (TMR6_FCNGR).....	528
20.5.14	Filter Control Trigger Port Register (TMR6_FCNTR).....	529
20.5.15	Valid Period Register (TMR6_VPERR)	530
20.5.16	Status Register (TMR6_STFLR)	531
20.5.17	Hardware Start Event Select Register (TMR6_HSTAR).....	533
20.5.18	Hardware Stop Event Select Register (TMR6_HSTPR)	535
20.5.19	Hardware Clear Event Select Register (TMR6_HCLRR).....	537
20.5.20	Hardware Update Event Select Register (TMR6_HUPDR)	539
20.5.21	Hardware Input Capture A Event Select Register (TMR6_HCPAR)	541
20.5.22	Hardware Input Capture B Event Select Register (TMR6_HCPBR)	543
20.5.23	Hardware Count Up Event Select Register (TMR6_HCUPR)	545
20.5.24	Hardware Count Down Event Select Register (TMR6_HCDOR).....	547
20.5.25	Hardware Trigger Event Select Register (TMR6_HTSSRm) (m=0~3).....	549
20.5.26	Software Sync Start Control Register (TMR6_SSTAR)	550
20.5.27	Software Sync Stop Control Register (TMR6_SSTPR).....	551
20.5.28	Software Sync Clear Control Register (TMR6_SCLRR)	552
20.5.29	Software Sync Update Control Register (TMR6_SUPDR).....	553
20.6	Notice for use	554
21	High Resolution PWM (HRPWM)	556
21.1	Introduction	556
21.2	Basic block diagram	556
21.3	Functional description	557
21.3.1	Calibration function	557
21.3.2	High Resolution PWM adjustment function	557
21.3.3	Note	558
21.4	Register description.....	559
21.4.1	HRPWM Control Register (HRPWM_CRn, n=1...16)	560
21.4.2	HRPWM Calibration Control Register 0 (HRPWM_CALCRn, n=0,1)	561
22	General control timer (Timer4)	562

22.1	Introduction	562
22.2	Basic block diagram	562
22.3	Function description	564
22.3.1	Waveform mode.....	564
22.3.2	Count action	564
22.3.3	Compare output.....	566
22.3.4	Buffer function.....	567
22.3.5	General PWM output.....	573
22.3.6	Period interval response	579
22.3.7	EMB control.....	581
22.3.8	Monitor output	582
22.4	Interrupt and Event Description	583
22.4.1	Count compare match interrupt	583
22.4.2	Count period match interrupt	583
22.4.3	Reload count match interrupt	583
22.4.4	Special compare match event	584
22.5	Register description.....	585
22.5.1	Counter Value Register (TMR4_CNTR).....	587
22.5.2	Period Value Register (TMR4_CPSR).....	587
22.5.3	Control Status Register (TMR4_CCSR).....	588
22.5.4	Valid Period Register (TMR4_CVPR).....	589
22.5.5	General Compare Value Register (TMR4_OCCRm).....	589
22.5.6	General Control Status Register (TMR4_OCSRn).....	590
22.5.7	General Expand Control Register (TMR4_OCERn)	591
22.5.8	General Mode Control Register (TMR4_OCMRm).....	593
22.5.9	Special Compare Value Register (TMR4_SCCRm).....	598
22.5.10	Special Control Status Register (TMR4_SCSRm)	599
22.5.11	Special Expand Control Register (TMR4_SCER).....	600
22.5.12	Special Mode Control Register (TMR4_SCMRm)	600
22.5.13	PWM Output Control Register (TMR4_POCRn)	601
22.5.14	PWM Status Control Register (TMR4_PSCR)	602
22.5.15	PWM Filter Control Register (TMR4_PFSRn)	605
22.5.16	PWM Deadtime Control Register (TMR4_PDARn)	605
22.5.17	Reload Control Status Register (TMR4_RCSR)	606
23	Emergency Brake Module (EMB)	607
23.1	Introduction	607
23.2	Functional description	609
23.2.1	Overview.....	609

23.2.2	Control PWM signal output when external port input level changes.....	611
23.2.3	Stop PWM signal output when the level of PWM output port is in the same phase (same high or same low)	612
23.2.4	Stop PWM signal output according to voltage comparator comparison result.....	614
23.2.5	Stop PWM signal output when external oscillator stops oscillating.....	614
23.2.6	Write register software control PWM signal output	615
23.3	Register description.....	616
23.3.1	EMB Control Registers 1_0~3 (EMB_CTL1_0~3)	617
23.3.2	EMB Control Registers 1_4~6 (EMB_CTL1_4~6)	619
23.3.3	EMB Control Registers 2_0~3 (EMB_CTL2_0~3)	620
23.3.4	EMB Control Registers 2_4~6 (EMB_CTL2_4~6)	621
23.3.5	EMB Software Output Enable Register (EMB_SOEx) (x=0~6)	622
23.3.6	EMB Status Register (EMB_STATx) (x=0~6).....	623
23.3.7	EMB Status Clear Register (EMB_STATCLR x) (x=0~6).....	624
23.3.8	EMB Interrupt Enable Register (EMB_INTENx) (x=0~6)	625
23.3.9	EMB Release Select Register (EMB_RLSSEL x) (x=0~6).....	626
24	General Timer (TimerA)	627
24.1	Introduction	627
24.2	Basic block diagram	627
24.3	Functional description	629
24.3.1	Waveform mode.....	629
24.3.2	Clock source selection	629
24.3.3	Compare output.....	630
24.3.4	Capture input.....	630
24.3.5	Synchronous startup.....	631
24.3.6	Digital filtering	632
24.3.7	Buffer function	633
24.3.8	Cascade count	634
24.3.9	PWM output	634
24.3.10	Quadrature encoding counting	636
24.4	Interrupt and Event Description	640
24.4.1	Compare Matched Interrupt and Events	640
24.4.2	Periodic Matching Interrupt and Events	641
24.5	Register description.....	642
24.5.1	Counter Value Register (TMRA_CNTER).....	644
24.5.2	Period Value Register (TMRA_PERAR).....	644
24.5.3	Compare Value Register (TMRA_CMPAR m) ($m=1~4$)	644
24.5.4	Basic Control and Status Register (TMRA_BCSTR)	645

24.5.5	Interrupt Control Register (TMRA_ICONR)	646
24.5.6	Event Control Register (TMRA_ECONR)	647
24.5.7	Filter Control Register (TMRA_FCONR)	648
24.5.8	Status Flag Register (TMRA_STFLR)	649
24.5.9	Buffer Control Register (TMRA_BCONR _m) ($m=1\sim 2$)	650
24.5.10	Capture Control Register (TMRA_CCONR _m) ($m=1\sim 4$)	651
24.5.11	Port Control Register (TMRA_PCONR _m) ($m=1\sim 4$)	652
24.5.12	Hardware Control Event Select Register (TMRA_HCONR).....	653
24.5.13	Hardware Count Up Event Select Register (TMRA_HCUPR).....	655
24.5.14	Hardware Count Down Event Select Register (TMRA_HCDOR).....	656
24.5.15	Hardware Trigger Source Select Register (TMRA_HTSSR _m) ($m=0\sim 3$).....	657
25	General Timer (Timer2)	658
25.1	Introduction.....	658
25.2	Basic block diagram	658
25.3	Functional description	660
25.3.1	Clock source selection	660
25.3.2	Compare output.....	661
25.3.3	Hardware triggering.....	661
25.3.4	Digital filtering	664
25.4	Interrupt and Event Description	666
25.4.1	Interrupt output	666
25.4.2	Event output	666
25.5	Register description.....	667
25.5.1	Counter Register (TMR2_CNT _m R) ($m=A, B$).....	668
25.5.2	Compare Value Register (TMR2_CMP _m R) ($m=A, B$)	668
25.5.3	Basic Control Register (TMR2_BCONR).....	669
25.5.4	Interrupt Control Register (TMR2_ICONR)	671
25.5.5	Port Control Register (TMR2_PCONR).....	672
25.5.6	Hardware Control Register (TMR2_HCONR)	674
25.5.7	Hardware Source Trigger Select Register (TMR2_HTSSR).....	676
25.5.8	Status Flag Register (TMR2_STFLR)	676
25.6	Precautions for use.....	677
26	General Timer (Timer0)	678
26.1	Introduction	678
26.2	Basic block diagram	678
26.3	Functional description	679
26.4	Clock source selection	679
26.4.2	Basic count	680

26.4.3	Hardware triggering.....	680
26.5	Interrupt and Event Description	681
26.5.1	Interrupt output	681
26.5.2	Event output.....	681
26.6	Register description.....	682
26.6.1	Counter Register (TMR0_CNTmR) (m=A~B)	683
26.6.2	Compare Value Register (TMR0_CMPmR) (m=A~B)	683
26.6.3	Basic Control Register (TMR0_BCONR).....	684
26.6.4	Hardware Trigger Source Selection Register (TMR0_HTSSR).....	686
26.6.5	Status Flag Register (TMR0_STFLR)	687
26.7	Precautions for use.....	688
27	Real Time Clock RTC.....	689
27.1	Introduction.....	689
27.2	Basic block diagram	690
27.3	Functional description	691
27.3.1	Power-on settings	691
27.3.2	RTC count start setting	691
27.3.3	System low power mode switching.....	691
27.3.4	Read count register	692
27.3.5	Write count register	692
27.3.6	Alarm setting	692
27.3.7	Clock Error Compensation	692
27.3.8	1Hz output.....	692
27.3.9	Intrusion detection.....	694
27.3.10	Timestamp function.....	694
27.4	Interrupt Description	695
27.4.1	Alarm interrupt	695
27.4.2	Periodic interrupt	695
27.4.3	Intrusion detection interrupt.....	695
27.5	Register description.....	696
27.5.1	Control Register 0 (RTC_CR0).....	697
27.5.2	Control Register 1 (RTC_CR1).....	698
27.5.3	Control Register 2 (RTC_CR2).....	699
27.5.4	Control Register 3 (RTC_CR3).....	700
27.5.5	Second Count Register (RTC_SEC)	700
27.5.6	Minute Count Register (RTC_MIN)	701
27.5.7	Hour count register (RTC_HOUR).....	701
27.5.8	Day Count Register (RTC_DAY)	703

27.5.9	Week Count Register (RTC_WEEK)	704
27.5.10	Month Count Register (RTC_MON).....	705
27.5.11	Year Count Register (RTC_YEAR)	705
27.5.12	Sub-alarm register (RTC_ALMMIN).....	706
27.5.13	Time alarm register (RTC_ALMHOUR).....	706
27.5.14	Weekly Alarm Register (RTC_ALM WEEK)	707
27.5.15	Clock Error Compensation Registers (RTC_ERRCRH, RTC_ERRCRL)	708
27.5.16	Intrusion Control Register 0 (RTC_TPCR0)	710
27.5.17	Intrusion Control Register 1 (RTC_TPCR1)	711
27.5.18	Intrusion Status Register (RTC_TPSR).....	712
27.5.19	Second Timestamp Register (RTC_SECTP)	712
27.5.20	Minute Timestamp Register (RTC_MINTP)	713
27.5.21	Time stamp register (RTC_HOURTP)	713
27.5.22	Day Timestamp Register (RTC_DAYTP).....	714
27.5.23	Month Timestamp Register (RTC_MONTP).....	714
28	Watchdog Timer (WDT/SWDT).....	715
28.1	Introduction	715
28.2	Function description	716
28.2.1	Start the Watchdog	716
28.2.2	Hardware startup mode	716
28.2.3	Software startup mode	717
28.2.4	Refresh action	718
28.2.5	Flag bit	718
28.2.6	Interrupt reset	718
28.2.7	Count underflow	719
28.2.8	Refresh error	720
28.3	Register description	721
28.3.1	Control Register (SWDT_CR, WDT_CR).....	722
28.3.2	State Register (SWDT_SR, WDT_SR)	723
28.3.3	Refresh Register (SWDT_RR, WDT_RR)	723
28.4	Notice	724
29	Universal Synchronous Asynchronous Receiver/Transmitter (USART)	725
29.1	Introduction	725
29.2	USART system block diagram	727
29.3	Pin description	727
29.4	Functional description	728
29.4.1	UART	728
29.4.2	Multiprocessor communication	738

29.4.3	UART-LIN	742
29.4.4	Smartcard	744
29.4.5	Clock synchronization mode	749
29.4.6	Digital filtering function	754
29.4.7	Interrupt.....	755
29.5	Register description.....	756
29.5.1	USART status register (USART_SR).....	757
29.5.2	USART data register (USART_DR).....	760
29.5.3	USART baud rate register (USART_BRR).....	761
29.5.4	USART control register1 (USART_CR1)	762
29.5.5	USART control register2 (USART_CR2)	765
29.5.6	USART control register3 (USART_CR3)	767
29.5.7	USART prescaler register (USART_PR)	768
29.5.8	USART LIN baudrate measuring counter (USART_LBMC).....	769
29.5.9	USART1 filter control register (USART1_NFC).....	770
29.6	Precautions for use.....	771
29.6.1	UART considerations	771
29.6.2	Clock synchronization mode considerations	771
29.6.3	Other considerations	771
30	Inter-integrated circuit bus (I2C)	772
30.1	Introduction	772
30.2	I2C System Block Diagram	773
30.2.1	System block diagram	773
30.2.2	Structural diagram.....	774
30.3	Action description.....	775
30.3.1	I2C protocol	775
30.3.2	Address match	784
30.3.3	SMBus action	790
30.3.4	Software Reset.....	792
30.3.5	Interrupt and event signal output.....	792
30.3.6	Programmable digital filtering	794
30.4	Application software setting I2C initialization process.....	794
30.5	Register description.....	795
30.5.1	I2C Controlled Register1 (I2C_CR1).....	796
30.5.2	I2C Control Register 1 (I2C_CR2)	797
30.5.3	I2C Control Register 1 (I2C_CR3)	798
30.5.4	I2C Control Register 4 (I2C_CR4)	799
30.5.5	I2C slave address register0 (I2C_SLR0)	800

30.5.6	I2C slave address register1 (I2C_SLR1)	801
30.5.7	I2C SCL Level Timeout Control Register (I2C_SLTR)	802
30.5.8	I2C Status register (I2C_SR).....	803
30.5.9	I2C Status Clear Register (I2C_CLR).....	806
30.5.10	I2C Data Transmitting Register (I2C_DTR).....	807
30.5.11	I2C Data Receiving Register (I2C_DRR).....	807
30.5.12	I2C Data Shift Register (I2C_DSR).....	808
30.5.13	I2C Clock Control Register (I2C_CCR).....	809
30.5.14	I2C Filtering Control Register (I2C_FLTR).....	811
31	Serial Peripheral Interface (SPI)	812
31.1	Introduction.....	812
31.2	SPI System Block Diagram	813
31.3	Pin description.....	813
31.4	SPI System Description.....	814
31.4.1	Pin Status in Master Mode	814
31.4.2	Pin Status in slave mode.....	814
31.4.3	SPI System Connection Examples.....	815
31.5	Data communication description.....	816
31.5.1	Baud rate	816
31.5.2	Data format	817
31.5.3	Transfer format	820
31.5.4	Communication mode.....	821
31.6	Operating instructions.....	823
31.6.1	Operational mode summary	823
31.6.2	Action of master device in SPI Operation Mode	824
31.6.3	Action of Slave device in SPI Operation Mode.....	824
31.6.4	Action of Master device in Clock Synchronous Operation Mode	826
31.6.5	Action of Slave device in Clock Synchronous Operation Mode.....	826
31.6.6	Examples of SPI communication processing flow	827
31.7	Self-diagnosis of parity	828
31.8	Error detection.....	828
31.8.1	Data Underflow error	829
31.8.2	Mode fault error	829
31.8.3	Data overflow error.....	830
31.8.4	Parity error.....	831
31.9	SPI Initialization	832
31.9.1	Clear SPE for initialization.....	832
31.9.2	System reset initialization.....	833

31.10 Interrupt	833
31.11 Available event trigger sources	833
31.12 Register description.....	834
31.12.1 SPI data register (SPI _ DR).....	835
31.12.2 SPI control register (SPI _ CR)	836
31.12.3 SPI Communication Configuration Register1 (SPI _ CFG1)	837
31.12.4 SPI status register (SPI _ SR).....	839
31.12.5 SPI Communication Configuration Register2 (SPI _ CFG2)	840
32 Quad Serial Peripheral Interface (QSPI).....	841
32.1 Introduction	841
32.2 Pin Description	842
32.3 Memory Map.....	843
32.3.1 Internal Bus Space	843
32.3.2 ROM Space and Bus Address Width	844
32.4 QSPI Bus	845
32.4.1 SPI Protocol	845
32.4.2 SPI Mode	847
32.5 QSPI Bus Timing Adjustment	848
32.5.1 QSPI Bus Reference Clock	848
32.5.2 SPI Bus Reference Clock	849
32.5.3 QSSN Signal Minimum High-Level Width	849
32.5.4 QSSN Signal Setup Time	850
32.5.5 QSSN Signal Hold Time	850
32.5.6 Serial Data Reception Delay	851
32.6 Introduction to SPI Instructions for ROM Access	852
32.6.1 Existing QSPI-ROM Instruction Reference	852
32.6.2 Standard Read Instruction	853
32.6.3 Fast Read Instruction	853
32.6.4 Fast Read Dual Output Instruction	855
32.6.5 Fast Read Dual I/O Instruction.....	856
32.6.6 Fast Read Quad Output Instruction	858
32.6.7 Fast Read Quad I/O Instruction	859
32.6.8 Enter 4-Byte Mode Instruction	860
32.6.9 Exit 4-Byte Mode Instruction.....	861
32.6.10 Write Permission Instruction	861
32.7 QSPI Bus Cycle Arrangement.....	862
32.7.1 Single Flash Read with Independent Conversion	862
32.7.2 Flash Read Using Prefetch Function	862

32.7.3	Prefetch Termination.....	863
32.7.4	Prefetch Status Monitoring.....	863
32.7.5	Flash Read Using QSPI Bus Cycle Extension Function	864
32.8	XIP Control.....	864
32.8.1	XIP Mode Settings.....	865
32.8.2	Exit from XIP Mode.....	865
32.9	QSIO2 and QSIO3 Pin States.....	866
32.10	Direct Communication Mode	867
32.10.1	About Direct Communication Mode	867
32.10.2	Direct Communication Mode Settings.....	867
32.10.3	Generation of QSPI Bus Cycles in Direct Communication Mode.....	867
32.11	Interrupt	868
32.12	Precautions for Use.....	868
32.12.1	Setting Sequence of QSPI Registers.....	868
32.12.2	Module Stop Signal Setting	868
32.13	Register Description	869
32.13.1	QSPI Control Register (QSPI_CR).....	870
32.13.2	QSPI Chip Select Control Register (QSPI_CSCR)	872
32.13.3	QSPI Format Control Register (QSPI_FCR)	873
32.13.4	QSPI Status Register (QSPI_SR).....	875
32.13.5	QSPI Direct Communication Register (QSPI_DCOM).....	876
32.13.6	QSPI Command Code Register (QSPI_CCMD)	876
32.13.7	QSPI XIP Mode Code Register (QSPI_XCMD)	877
32.13.8	QSPI Status Flag Clear Register (QSPI_SR2).....	877
32.13.9	QSPI External Address Register (QSPI_EXAR).....	878
33	Integrated circuit built-in audio bus module (I2S)	879
33.1	Introduction	879
33.2	I2S system block diagram	880
33.3	Pin description.....	881
33.4	Functional description	881
33.4.1	I2S General Instructions.....	881
33.4.2	Communication mode.....	881
33.4.3	Supported Audio Protocols.....	882
33.4.4	Clock generator	887
33.4.5	I2S host mode	889
33.4.6	I2S slave mode	890
33.4.7	I2S interrupt.....	892
33.4.8	Precautions for use	893

33.5 Register description.....	896
33.5.1 I2S Control Register (I2S_CTRL)	897
33.5.2 I2S Status Register (I2S_SR)	898
33.5.3 I2S Error Status Register (I2S_ER).....	898
33.5.4 I2S Configuration Register (I2S_CFGR).....	899
33.5.5 I2S transmit buffer FIFO data register (I2S_TXBUF)	900
33.5.6 I2S receive buffer FIFO data register (I2S_RXBUF).....	901
33.5.7 I2S divider register (I2S_PR)	902
33.5.8 CMU I2S Clock Configuration Register (CMU_I2SCKSEL)	903
34 USB2.0 High Speed Module (USBHS)	904
34.1 Introduction to USBHS	904
34.2 Main features of USBHS.....	904
34.2.1 Common features	904
34.2.2 Host Mode Features	905
34.2.3 Device Mode Features.....	905
34.3 USBHS System Block Diagram	906
34.4 USBHS pin description.....	907
34.5 USBHS function description.....	908
34.5.1 USBHS clock and working mode	908
34.5.2 USBHS mode decision.....	908
34.5.3 USBHS host function.....	908
34.5.4 USBHS Device Features	913
34.5.5 USBHS SOF pulse output function.....	917
34.5.6 USBHS power control.....	917
34.5.7 USBHS dynamically updates the USBHS_HFIR register.....	918
34.5.8 USBHS data FIFO	918
34.5.9 USBHS Host FIFO Architecture	919
34.5.10 USBHS Device FIFO Architecture.....	921
34.5.11 USBHS FIFO RAM allocation	922
34.5.12 USBHS system performance	923
34.5.13 USBHS interrupts and events.....	923
34.6 USBHS programming model.....	926
34.6.1 USBHS module initialization.....	926
34.6.2 USBHS host initialization.....	926
34.6.3 USBHS device initialization	927
34.6.4 USBHS DMA mode	927
34.6.5 USBHS Host Programming Model.....	928
34.6.6 USBHS Device Programming Model	929

34.6.7	USBHS operating model.....	932
34.7	Register description.....	949
34.7.1	USB System Control Register.....	953
34.7.2	USBHS Global Register.....	955
34.7.3	USBHS Host Mode Register.....	977
34.7.4	USBHS Device Mode Register	992
34.7.5	USBHS Clock Gating Control Register	1024
35	USB2.0 Full Speed Module (USBFS)	1025
35.1	Introduction to USBFS.....	1025
35.2	Main Features of USBFS.....	1025
35.2.1	Common features	1025
35.2.2	Host Mode Features	1026
35.2.3	Device Mode Features.....	1026
35.3	USBFS System Block Diagram	1027
35.4	USBFS pin description	1028
35.5	USBFS function description	1028
35.5.1	USBFS clock and working mode	1028
35.5.2	USBFS mode decision	1028
35.5.3	USBFS host function.....	1029
35.5.4	USBFS Device Features	1033
35.5.5	USBFS SOF pulse output function	1037
35.5.6	USBFS power control.....	1037
35.5.7	USBFS dynamically updates the USBFS_HFIR register	1038
35.5.8	USBFS data FIFO	1038
35.5.9	USBFS Host FIFO Architecture.....	1039
35.5.10	USBFS Device FIFO Architecture	1041
35.5.11	USBFS FIFO RAM allocation	1042
35.5.12	USBFS system performance.....	1043
35.5.13	USBFS interrupts and events	1044
35.6	USBFS programming model.....	1045
35.6.1	USBFS module initialization	1045
35.6.2	USBFS host initialization	1045
35.6.3	USBFS device initialization.....	1046
35.6.4	USBFS DMA mode	1047
35.6.5	USBFS Host Programming Model	1047
35.6.6	USBFS Device Programming Model.....	1048
35.6.7	USBFS operating model	1050
35.7	Register description.....	1068

35.7.1	USB System Control Register.....	1073
35.7.2	USBFS global registers.....	1075
35.7.3	USBFS Host Mode Register	1096
35.7.4	USBFS Device Mode Register.....	1109
35.7.5	USBFS Clock Gating Control Register.....	1136
36	CAN FD Controller (CAN FD).....	1137
36.1	Introduction.....	1137
36.2	CAN FD system block diagram	1138
36.3	Pin description.....	1138
36.4	Function description	1139
36.4.1	Operation mode.....	1139
36.4.2	Baud rate setting	1139
36.4.3	Transmit buffer (TB)	1142
36.4.4	Receive buffer (RB).....	1142
36.4.5	Acceptance filters	1143
36.4.6	Data transmission.....	1144
36.4.7	Single shot transmission.....	1144
36.4.8	Transmission abort.....	1144
36.4.9	Data reception	1145
36.4.10	Error handling	1145
36.4.11	Bus Off.....	1146
36.4.12	Arbitration lost capture	1146
36.4.13	Loopback mode	1146
36.4.14	Listen only mode (LOM)	1147
36.4.15	Software reset.....	1148
36.4.16	Upward compatible with CAN-FD	1149
36.4.17	Time-triggered CAN(TTCAN).....	1149
36.4.18	TDC and RDC	1152
36.4.19	Interrupt.....	1153
36.5	Register description.....	1154
36.5.1	CAN Receive Buffer Register (CAN_RBUF).....	1155
36.5.2	CAN Transmit Buffer Register (CAN_TBUF).....	1158
36.5.3	CAN Configuration and Status Register (CAN_CFG_STAT)	1160
36.5.4	CAN Command Register (CAN_TCMD).....	1161
36.5.5	CAN Transmit Control Register (CAN_TCTRL)	1163
36.5.6	CAN Receive Control Register (CAN_RCTRL)	1164
36.5.7	CAN Receive and Transmit Interrupt Enable Register (CAN_RTIE)	1165
36.5.8	CAN Receive and Transmit Interrupt Flag Register (CAN_RTIF)	1166

36.5.9	CAN Error Interrupt Enable and Flag Register (CAN_ERRINT).....	1167
36.5.10	CAN Slow Bit Timing Register (CAN_SBT).....	1168
36.5.11	CAN Fast Bit Timing Register (CAN_FBT).....	1169
36.5.12	CAN Transmitter Delay Compensation Register (CAN_TDC).....	1169
36.5.13	CAN Error and Arbitration Lost Capture Register (CAN_EALCAP)	1170
36.5.14	CAN Warning Limits Register (CAN_LIMIT)	1170
36.5.15	CAN Receive Error Counter Register (CAN_RECNT).....	1171
36.5.16	CAN Transmit Error Counter Register (CAN_TECNT).....	1171
36.5.17	CAN Acceptance Filter Control Register (CAN_ACFCTRL)	1172
36.5.18	CAN Acceptance Filter Enable Register (CAN_ACFEN).....	1173
36.5.19	CAN Acceptance CODE and MASK Register(CAN_ACF).....	1174
36.5.20	TTCAN TB Slot Pointer Register(CAN_TBSLOT)	1175
36.5.21	TTCAN Time Trigger Configuration Register (CAN_TTCFG).....	1176
36.5.22	TTCAN Reference Message Register (CAN_REF_MSG)	1177
36.5.23	TTCAN Trigger Configuration Register (CAN_TRG_CFG).....	1178
36.5.24	TTCAN Trigger Time Register (CAN_TT_TRIG)	1179
36.5.25	TTCAN Watch Trigger Time Register (CAN_TT_WTRIG)	1179
36.6	Precautions for use.....	1180
36.6.1	CAN bus anti-interference measures	1180
36.6.2	CAN Controller Noise Constraints.....	1180
37	CAN2.0B Controller (CAN2.0B)	1181
37.1	Introduction	1181
37.2	CAN system block diagram	1182
37.3	Pin description	1182
37.4	Function description	1182
37.4.1	Operation mode.....	1182
37.4.2	Baud rate setting	1183
37.4.3	Transmit buffer (TB)	1184
37.4.4	Receive buffer (RB).....	1184
37.4.5	Acceptance filters	1185
37.4.6	Data transmission.....	1186
37.4.7	Single shot transmission.....	1186
37.4.8	Transmission abort.....	1186
37.4.9	Data reception	1187
37.4.10	Error handling	1187
37.4.11	Bus Off.....	1188
37.4.12	Arbitration lost capture	1188
37.4.13	Loopback mode	1188

37.4.14 Listen only mode (LOM)	1189
37.4.15 Software reset.....	1190
37.4.16 Upward compatible with CAN-FD function	1191
37.4.17 Time-Triggered CAN(TTCAN).....	1191
37.4.18 Interrupt.....	1194
37.5 Register description.....	1195
37.5.1 CAN Receive Buffer Register (CAN_RBUF).....	1196
37.5.2 CAN Transmit Buffer Register (CAN_TBUF).....	1199
37.5.3 CAN Configuration and Status Register (CAN_CFG_STAT)	1201
37.5.4 CAN Command Register (CAN_TCMD).....	1202
37.5.5 CAN Transmit Control Register (CAN_TCTRL)	1204
37.5.6 CAN Receive Control Register (CAN_RCTRL)	1205
37.5.7 CAN Receive and Transmit Interrupt Enable Register (CAN_RTIE).....	1206
37.5.8 CAN Receive and Transmit Interrupt Flag Register (CAN_RTIF)	1207
37.5.9 CAN Error Interrupt Enable and Flag Register (CAN_ERRINT).....	1209
37.5.10 CAN Slow Bit Timing Register (CAN_SBT).....	1210
37.5.11 CAN Error and Arbitration Lost Capture Register (CAN_EALCAP)	1211
37.5.12 CAN Warning Limits Register (CAN_LIMIT)	1211
37.5.13 CAN Receive Error Counter Register (CAN_RECNT).....	1212
37.5.14 CAN Transmit Error Counter Register (CAN_TECNT).....	1212
37.5.15 CAN Acceptance Filter Control Register (CAN_ACFCTRL)	1213
37.5.16 CAN Acceptance Filter Enable Register (CAN_ACFEN).....	1214
37.5.17 CAN Acceptance CODE and MASK Register (CAN_ACF).....	1215
37.5.18 TTCAN TB Slot Pointer Register(CAN_TBSLOT)	1216
37.5.19 TTCAN Time Trigger Configuration Register (CAN_TTCFG).....	1217
37.5.20 TTCAN Reference Message Register (CAN_REF_MSG)	1218
37.5.21 TTCAN Trigger Configuration Register (CAN_TRG_CFG).....	1219
37.5.22 TTCAN Trigger Time Register (CAN_TT_TRIG)	1220
37.5.23 TTCAN Watch Trigger Time Register (CAN_TT_WTRIG)	1220
37.6 Precautions for use.....	1221
37.6.1 CAN bus anti-interference measures	1221
37.6.2 CAN Controller Noise Constraints.....	1221
38 SDIO Controller (SDIOC)	1222
38.1 Introduction	1222
38.2 Functional description	1223
38.2.1 Port assignment.....	1223
38.2.2 Basic access	1224
38.2.3 Data transmission.....	1224

38.2.4	SD clock.....	1224
38.2.5	Interrupts and DMA Start Requests.....	1224
38.2.6	Host and device initialization.....	1226
38.2.7	SD/MMC single block (single block) read and write.....	1227
38.2.8	SD/MMC multi-block read and write.....	1228
38.2.9	Transfer abort, suspend and resume	1230
38.2.10	Read wait.....	1231
38.2.11	Wakeup.....	1231
38.3	Register description.....	1233
38.3.1	Block Length Register (BLKSIZE).....	1234
38.3.2	Block Count Register (BLKCNT).....	1234
38.3.3	Parameter register 0 (ARG0).....	1234
38.3.4	Parameter register 1 (ARG1).....	1235
38.3.5	Transfer Mode Register (TRANSMODE).....	1235
38.3.6	Command Register (CMD)	1236
38.3.7	Response Register 0 (RESP0).....	1236
38.3.8	Response Register 1 (RESP1).....	1237
38.3.9	Response Register 2 (RESP2).....	1237
38.3.10	Response Register 3 (RESP3).....	1237
38.3.11	Response Register 4 (RESP4).....	1238
38.3.12	Response Register 5 (RESP5).....	1238
38.3.13	Response Register 6 (RESP6).....	1238
38.3.14	Response Register 7 (RESP7).....	1239
38.3.15	Data Buffer Register 0 (BUF0).....	1239
38.3.16	Data Buffer Register 1 (BUF1).....	1239
38.3.17	Host Status Register (PSTAT)	1240
38.3.18	Host Control Register (HOSTCON).....	1241
38.3.19	Power Control Register (PWRCON)	1241
38.3.20	Block Gap Control Register (BLKGPCON)	1242
38.3.21	Clock Control Register (CLKCON)	1242
38.3.22	Timeout Control Register (TOUTCON)	1243
38.3.23	Software Reset Register (SFTRST).....	1243
38.3.24	Normal Interrupt Status Register (NORINTST).....	1244
38.3.25	Error Interrupt Status Register (ERRINTST)	1245
38.3.26	Normal Interrupt Status Enable Register (NORINTSTEN)	1246
38.3.27	Error Interrupt Status Enable Register (ERRINTSTEN)	1247
38.3.28	Normal interrupt signal enable register (NORINTSGEN).....	1248
38.3.29	Error Interrupt Signal Enable Register (ERRINTSGEN).....	1249

38.3.30	AutoCommand Error Status Register (ATCERRST).....	1250
38.3.31	Forced Autocommand Error Status Control Register (FEA)	1250
38.3.32	Forced Error Status Control Register (FEE).....	1251
38.3.33	MMC Mode Enable Register (MMCER)	1252
39	Ethernet MAC Controller (ETHMAC)	1253
39.1	Summary.....	1253
39.2	Basic Features	1254
39.2.1	Basic block diagram.....	1254
39.2.2	ETH_MAC features.....	1254
39.2.3	ETH_PTP features.....	1257
39.2.4	ETH_DMA characteristics	1257
39.3	Interface Description	1258
39.3.1	MII interface.....	1258
39.3.2	RMII interface.....	1260
39.3.3	SMI interface.....	1260
39.3.4	Ethernet port configuration	1261
39.4	Functional description	1262
39.4.1	ETH_MAC function.....	1262
39.4.2	ETH_PTP function	1283
39.4.3	ETH_DMA function	1289
39.5	Interrupt Description	1312
39.5.1	DMA interrupt	1313
39.5.2	PMT interrupt	1314
39.5.3	PTP interrupt.....	1314
39.5.4	MMC interrupt.....	1314
39.6	Register description.....	1315
39.6.1	ETH_MAC register	1318
39.6.2	ETH_PTP register.....	1346
39.6.3	ETH_DMA register	1356
39.6.4	ETH_MMC register.....	1370
40	External Memory Controller (EXMC)	1381
40.1	Summary.....	1381
40.2	Basic Features	1381
40.2.1	Function list	1381
40.2.2	Controller Architecture.....	1382
40.2.3	Basic Access Specification	1383
40.2.4	Address mapping	1383
40.2.5	Protocol interface.....	1388

40.3	Functional description	1390
40.3.1	SMC-SRAM/PSRAM/NOR Flash Controller.....	1390
40.3.2	DMC-SDRAM Controller	1407
40.3.3	NFC-NAND Flash Controller.....	1418
40.4	Interrupt Description	1421
40.5	Register description.....	1422
40.5.1	SMC-SRAM/PSRAM/NOR Flash register	1424
40.5.2	DMC-SDRAM register	1433
40.5.3	NFC-NAND Flash register	1445
40.6	Precautions for use.....	1461
41	Digital Video Interface (DVP)	1462
41.1	Introduction	1462
41.2	System block diagram.....	1462
41.3	Functional description	1464
41.3.1	Video data format.....	1464
41.3.2	Parallel port storage format	1465
41.3.3	Pattern selection	1466
41.3.4	Synchronous control	1468
41.3.5	Window cropping	1469
41.3.6	FIFO control	1470
41.3.7	DMA control	1470
41.4	Interrupt and Event Description	1470
41.4.1	Frame transfer status interrupts and events.....	1471
41.4.2	Software Synchronization Error Interrupts and Events.....	1471
41.4.3	FIFO overflow error interrupts and events	1471
41.5	Register description.....	1472
41.5.1	Control Register (DVP_CTR)	1473
41.5.2	Data Register (DVP_DTR)	1474
41.5.3	Status Register (DVP_STR).....	1474
41.5.4	Interrupt Register (DVP_IER)	1475
41.5.5	DMA Data Transfer Register (DVP_DMR)	1475
41.5.6	Software Synchronization Data Register (DVP_SSYNDR).....	1476
41.5.7	Software Synchronization Mask Register (DVP_SSYNMR).....	1476
41.5.8	Window Clipping Offset Register (DVP_CPSFTR)	1477
41.5.9	Window Crop Size Register (DVP_CPSZER)	1477
42	Cryptographic Coprocessing Module (CPM)	1478
42.1	Introduction	1478
42.2	Encryption and Decryption Algorithm Processor (AES).....	1479

42.2.1	Introduction to Algorithms	1479
42.2.2	AES module function description	1481
42.2.3	Encryption Operation Process.....	1481
42.2.4	Decryption operation process.....	1481
42.2.5	Data example	1482
42.2.6	Running time description.....	1485
42.2.7	Operation Precautions	1485
42.2.8	Register description	1486
42.3	Secure Hash Algorithm (HASH).....	1491
42.3.1	Introduction to Algorithms	1491
42.3.2	Operating procedures	1492
42.3.3	Message padding.....	1492
42.3.4	HMAC operation.....	1494
42.3.5	Interrupt Description.....	1496
42.3.6	Hardware trigger event selection	1497
42.3.7	Register description	1498
42.4	True random number generator TRNG	1505
42.4.1	Block Diagram	1505
42.4.2	Operating procedures	1505
42.4.3	Interrupt and event output	1505
42.4.4	Operation Precautions	1505
42.4.5	Register description	1506
43	CRC operation (CRC)	1508
43.1	Introduction	1508
43.2	Functional block diagram	1508
43.3	Functional description	1508
43.3.1	CRC 16 coding mode.....	1508
43.3.2	CRC 16 check mode.....	1509
43.3.3	CRC 32 coding mode.....	1509
43.3.4	CRC 32 check mode.....	1509
43.4	Register description.....	1510
43.4.1	Control register (CRC _ CR)	1510
43.4.2	Results register (CRC _ RESLT)	1511
43.4.3	Data register (CRC _ DAT)	1511
44	Data Computation Unit (DCU)	1512
44.1	Summary	1512
44.2	Functional description	1513
44.2.1	Additive mode.....	1513

44.2.2	Subtraction mode	1513
44.2.3	Hardware Triggered Boot Mode.....	1513
44.2.4	Compare mode	1514
44.2.5	Interrupt and event signal output.....	1515
44.2.6	Triangle wave output mode	1515
44.2.7	Incremental sawtooth output mode.....	1518
44.2.8	Decreasing sawtooth output mode	1520
44.3	Register description.....	1522
44.3.1	DCU Control Register (DCU x _CTL) ($x=1\sim 8$).....	1526
44.3.2	DCU Flag Register (DCU x _FLAG) ($x=1\sim 4$)	1527
44.3.3	DCU Flag Register (DCU x _FLAG) ($x=5\sim 8$)	1528
44.3.4	DCU data register (DCU x _DATA y) ($x=1\sim 8, y=0,1,2$).....	1529
44.3.5	DCU Flag Reset Register (DCU x _FLAGCLR) ($x=1\sim 4$)	1532
44.3.6	DCU Flag Reset Register (DCU x _FLAGCLR) ($x=5\sim 8$)	1533
44.3.7	DCU Interrupt and Event Register (DCU x _INTEVTSEL) ($x=1\sim 4$)	1534
44.3.8	DCU Interrupt and Event Register (DCU x _INTEVTSEL) ($x=5\sim 8$)	1536
44.3.9	DCU trigger source selection register (DCU x _TRGSEL) ($x=1\sim 4$)	1537
45	Mathematical Operations Unit (MAU)	1538
45.1	Introduction	1538
45.2	Functional description	1538
45.2.1	Square root operation.....	1538
45.2.2	Sine operation	1539
45.3	Interrupt and Event Description	1540
45.3.1	Interrupt output	1540
45.3.2	Event output	1540
45.4	Register description.....	1541
45.4.1	Control Status Register (MAU_CSR)	1541
45.4.2	Data Input Register 0 (MAU_DTR0)	1542
45.4.3	Result output register 0 (MAU_RTR0).....	1542
45.4.4	Data Input Register 1 (MAU_DTR1)	1542
45.4.5	Result Output Register 1 (MAU_RTR1).....	1543
46	Filter Math Accelerator (FMAC).....	1544
46.1	Introduction	1544
46.2	Basic block diagram	1544
46.3	Operating procedures.....	1545
46.4	Module enable	1545
46.5	Coefficient normalization.....	1545
46.6	Interrupt and Event Description	1546

46.6.1	Interrupt output	1546
46.6.2	Event output.....	1546
46.7	Register description.....	1547
46.7.1	Module Enable Register (FMAC_ENR).....	1548
46.7.2	Basic Control Register (FMAC_CTR).....	1549
46.7.3	Interrupt Control Register (FMAC_IER)	1550
46.7.4	Data Input Register (FMAC_DTR)	1550
46.7.5	Filter coefficient register (FMAC_COR0~16).....	1550
46.7.6	Result Output Register 0 (FMAC_RTR0).....	1551
46.7.7	Result Output Register 1 (FMAC_RTR1).....	1551
46.7.8	Operation Status Register (FMAC_STR).....	1551
47	Debug controller (DBG)	1552
47.1	Introduction.....	1552
47.2	DBG System Block Diagram	1552
47.3	SWJ-DP debug port (SWD and JTAG)	1553
47.3.1	Switching mechanism of JTAG-DP or SW-DP.....	1554
47.4	Pinout and debug port pins	1555
47.4.1	SWJ debug port pins	1555
47.4.2	Flexible SWJ-DP pinout.....	1555
47.4.3	Internal pull-ups on JTAG pins	1556
47.4.4	Use the serial interface and free unused debug pins for GPIO.....	1556
47.5	Register description.....	1557
47.5.1	DBG State Register (MCUDBGSTAT)	1557
47.5.2	Peripheral debugging pause register (MCUSTPCTL).....	1558
47.5.3	Debug Component Configuration Register (MCUTRACECTL).....	1560
47.5.4	Peripheral Debug Suspend Register 2 (MCUSTPCTL2)	1561
47.6	SW debug port.....	1563
47.6.1	Introduction to SW Agreement.....	1563
47.7	TPIU (Trace Port Interface Unit)	1563
47.7.1	Introduction	1563
47.7.2	TRACE pin assignment.....	1564
47.7.3	MCU internal TRACECLKIN connection	1566
47.7.4	TPIU register	1566
47.7.5	TPIU configuration example.....	1566
Version revision history	1567	

Table Index

Table 1-1	Memory map	58
Table 1-2	QSPI address space allocation.....	64
Table 1-3	Example of target address configuration main flash (MMF_REMCR0.EN0=1 or MMF_REMCR1.EN1=1)	65
Table 1-4	Example of target address configuration high-speed SRAM (MMF_REMCR0.EN0=1 or MMF_REMCR1.EN1=1)	65
Table 1-5	Register list	66
Table 3-1	Reset source and generation conditions	73
Table 3-2	Reset method and reset flag	74
Table 3-3	Reset conditions for each module	83
Table 3-4	List of RMU registers	84
Table 4-1	Main characteristics table of clock source.....	91
Table 4-2	Specification of internal clocks.....	92
Table 5-1	BOR configuration	142
Table 5-2	PVD1/PVD2 characteristics.....	143
Table 5-3	Operation mode	149
Table 5-4	Low power mode	149
Table 5-5	Low-power mode operating conditions and the status of each module in low-power mode	150
Table 5-6	Operation Modes	152
Table 5-7	Power-down mode sub-mode	156
Table 5-8	RAM module and RAM power-down control bits	159
Table 5-9	Protected register list	160
Table 5-10	List of Registers.....	161
Table 6-1	List of Registers.....	189
Table 7-1	CPU clock frequency and FLASH read wait period comparison table	195
Table 7-2	FLASH read cycles.....	196
Table 7-3	OTP address allocation table.....	203
Table 7-4	List of Registers.....	207
Table 8-1	The relationship between the wait period setting of SRAM read/write access and the CPU clock frequency.....	227
Table 8-2	SRAM address map.....	228
Table 9-1	PORT register list 1	242
Table 9-2	PORT register list 2	243
Table 9-3	List of PORT registers during 32bit access	257
Table 11-1	AOS Target List	299

Table 11-2 Register List	301
Table 12-1 Introduction to MPU Module	319
Table 12-2 MPU register list	322
Table 13-1 KEYS defense pin description.....	342
Table 13-2 KEYS defense register list	344
Table 14-1 Measurement error when HRC target frequency is 20MHz.....	349
Table 14-2 Measurement error when HRC target frequency is 16MHz.....	349
Table 14-3 List of CTC Registers.....	352
Table 15-1 Channel reset instructions	363
Table 15-2 List of Registers.....	371
Table 16-1 CMP pin	394
Table 16-2 List of CMP registers.....	400
Table 16-3 Positive input voltage list	406
Table 16-4 Negative input voltage list	406
Table 16-5 Timer Window PWM List	407
Table 17-1 Specifications of each ADC unit	412
Table 17-2 Various competitions of sequences A and B.....	418
Table 17-3 AD conversion time.....	422
Table 17-4 ADC Register List 1/2.....	432
Table 17-5 ADC Register List 2/2.....	433
Table 18-1 DAC pin	456
Table 18-2 DAC1 register list	461
Table 18-3 DAC2 Register List.....	461
Table 18-4 D/A conversion and analog output control	464
Table 19-1 OTS preset temperature data.....	468
Table 19-2 How to use and set E_hrc.....	468
Table 20-1 Basic functions and features of Timer6	472
Table 20-2 Timer 6 port list.....	473
Table 20-3 Comparison table of functions in different modes.....	510
Table 20-4 Timer6 register list	514
Table 20-5 Counter (COUNTER) control priority	554
Table 20-6 PWMA port output control priority	555
Table 20-7 PWMB port output control priority	555
Table 21-1 Register list	559
Table 22-1 Basic functions and features	562
Table 22-2 Timer4 port list.....	563
Table 22-3 Timer4 register list	585
Table 22-4 PWM port output status and register setting value	604

Table 23-1	EMB port assignment	611
Table 23-2	EMB Group Control Timer	613
Table 23-3	List of EMB registers.....	616
Table 24-1	Basic functions and features of TimerA.....	627
Table 24-2	TimerA port list.....	628
Table 24-3	TimerA Register List	642
Table 24-4	Internal trigger event HTSSR selection relationship correspondence table.....	643
Table 25-1	Basic functions and features of Timer2	658
Table 25-2	Timer2 port list.....	659
Table 25-3	Timer2 Register List	667
Table 26-1	Timer0 register list.....	682
Table 27-1	Basic Specifications of RTC.....	689
Table 27-2	Register list	696
Table 28-1	Basic Characteristics of Watchdog Timer	715
Table 28-2	Register list	721
Table 29-1	USART pin description	727
Table 29-2	Tolerance of UART receiver when DIV_Fraction is 0	733
Table 29-3	Tolerance of UART receiver when DIV_Fraction is not 0	733
Table 29-4	UART Interrupt/Event Table	738
Table 29-5	Multiprocessor Mode Interrupt/Event Table	742
Table 29-6	LIN Interrupt/Event Table.....	744
Table 29-7	Smartcard Mode Interrupt/Event Table.....	749
Table 29-8	Clock Synchronized Mode Interrupt/Event Table.....	754
Table 29-9	USART overall interrupt list	755
Table 29-10	USART Register List	756
Table 29-11	Baud rate calculation formula (fractional baud rate is invalid FBME=0)	761
Table 29-12	Baud rate calculation formula (fractional baud rate valid FBME=1).....	761
Table 30-1	Input/output pins.....	774
Table 30-2	Interrupt List	792
Table 30-3	Event signal output list	793
Table 30-4	I2C Registers	795
Table 31-1	Features of SPI	812
Table 31-2	Pin Description	813
Table 31-3	SPI pin status description in master mode	814
Table 31-4	SPI pin status description in slave mode	814
Table 31-5	baud rate calculated according to the set value	816
Table 31-6	The relationship of SPI operation mode and register setting.....	823
Table 31-7	Error detection correspondence table	829

Table 31-8 SPI Interrupt Source Description	833
Table 31-9 List of SPI registers.....	834
Table 32-1 QSPI main specifications	841
Table 32-2 QSPI pins	842
Table 32-3 QSPI bus reference clock selection list	848
Table 32-4 Reference instruction list.....	852
Table 32-5 Pin states of QIO2 and QIO3.....	866
Table 32-6 List of QSPI registers	869
Table 33-1 I2S main features	879
Table 33-2 I2S pin description.....	881
Table 33-3 Audio frequency accuracy (for VCO input frequency = 1MHz)	888
Table 33-4 I2S interrupt request	892
Table 33-5 List of I2S registers.....	896
Table 34-1 USBHS pin description.....	907
Table 34-2 USBHS_GLB interrupt event table	924
Table 34-3 List of USBHS System Control Registers.....	950
Table 34-4 List of USBHS System Control Registers.....	951
Table 35-1 USBFS pin description	1028
Table 35-2 USBFS_GLB interrupt event table	1044
Table 35-3 List of USBFS System Control Registers	1070
Table 35-4 List of USBFS System Control Registers	1070
Table 36-1 CAN pin description.....	1138
Table 36-2 CAN bit time setting rules	1140
Table 36-3 Recommendations for 20MHz can_clk.....	1141
Table 36-4 Recommendations for 40MHz can_clk.....	1141
Table 36-5 Recommendations for 80MHz can_clk.....	1141
Table 36-6 Software Reset	1148
Table 36-7 CAN Interrupt flag	1153
Table 36-8 CAN register list	1154
Table 36-9 Standard format of CAN receive buffer	1155
Table 36-10 Extended format of CAN receive buffer.....	1156
Table 36-11 DLC control bits.....	1157
Table 36-12 Standard format of CAN transmit buffer.....	1158
Table 36-13 Extended format of CAN transmit buffer	1159
Table 37-1 CAN pin description.....	1182
Table 37-2 CAN bit time setting rules	1183
Table 37-3 Software Reset Range Table	1190
Table 37-4 CAN Interrupt flag	1194

Table 37-5 CAN register list	1195
Table 37-6 Standard format CAN receive mailbox format.....	1196
Table 37-7 Extended format CAN receive mailbox format	1197
Table 37-8 DLC control bits.....	1197
Table 37-9 Standard format CAN send mailbox format.....	1199
Table 37-10 Extended format CAN send mailbox format	1200
Table 38-1 Port allocation table	1223
Table 38-2 Register list	1233
Table 39-1 MII interface signal description	1259
Table 39-2 Signal state description when sending data.....	1259
Table 39-3 Signal state description when sending data.....	1260
Table 39-4 SMI interface signal description	1261
Table 39-5 SMI frame format	1261
Table 39-6 ETHMAC port function assignment.....	1261
Table 39-7 Timestamp snapshot target message	1284
Table 39-8 Regular Tx Descriptor.....	1299
Table 39-9 Enhanced Tx Descriptor	1302
Table 39-10 Regular Rx Descriptor	1307
Table 39-11 Enhanced Descriptor.....	1310
Table 39-12 ETHMAC register list.....	1315
Table 40-1 Basic functions of EXMC	1381
Table 40-2 AHB access width and memory bit width corresponding access mode table.....	1383
Table 40-3 SMC protocol interface	1388
Table 40-4 DMC protocol interface.....	1388
Table 40-5 NFC protocol interface.....	1389
Table 40-6 EXMC port function assignment	1389
Table 40-7 One-time read operation basic setting example	1393
Table 40-8 Basic setting example of address data line multiplexing single read operation.....	1396
Table 40-9 One-time write operation basic setting example	1398
Table 40-10 Basic setting example of address data line multiplexing single write action	1400
Table 40-11 Basic setting example of burst read operation.....	1401
Table 40-12 Basic setting example of burst write operation.....	1406
Table 40-13 Command truth table for DMC	1410
Table 40-14 ONFI access command.....	1419
Table 40-15 EXMC register list	1422
Table 40-16 CS0 setting and access address correspondence table.....	1437
Table 40-17 List of command parameters	1448
Table 40-18 Index register value and MEM address correspondence table	1450

Table 40-19 ECCR register list in 4bitECC mode	1458
Table 41-1 DVP basic functions and features.....	1462
Table 41-2 DVP port list	1463
Table 41-3 Single-color format video data storage method	1464
Table 41-4 YCbCr format video data storage method	1464
Table 41-5 RGB565 format video data storage method.....	1464
Table 41-6 8bit data DVP storage mode	1465
Table 41-7 10bit data DVP storage mode	1466
Table 41-8 12bit data DVP storage mode	1466
Table 41-9 14bit data DVP storage mode	1466
Table 41-10 Frame transfer interrupted.....	1471
Table 41-11 Sync code error sequence.....	1471
Table 41-12 DVP register list.....	1472
Table 42-1 128-bit operation register example.....	1482
Table 42-2 192-bit operation register example.....	1483
Table 42-3 256-bit operation register example.....	1484
Table 42-4 AES encryption and decryption running time.....	1485
Table 42-5 Register list	1486
Table 42-6 HASH register list	1498
Table 42-7 TRNG register list	1506
Table 43-1 CRC register list.....	1510
Table 44-1 List of DCU registers.....	1522
Table 45-1 MAU Register List	1541
Table 46-1 FMAC register list	1547
Table 47-1 SWJ debug port pins.....	1555
Table 47-2 Flexible SWJ-DP pinout	1555
Table 47-3 Register list	1557

Graph Index

Figure 2-1 Bus Architecture Diagram.....	70
Figure 3-1 Power-on reset.....	75
Figure 3-2 NRST reset timing.....	75
Figure 3-3 Brown-out reset	76
Figure 3-4 Programmable Voltage Detect 1 Reset	77
Figure 3-5 Programmable Voltage Detect 2 Reset	77
Figure 3-6 Watchdog and Dedicated Watchdog Reset	78
Figure 3-7 Power-down wake-up reset	79
Figure 3-8 Software reset	79
Figure 3-9 MPU error reset.....	79
Figure 3-10 RAM parity reset	80
Figure 3-11 RAM ECC reset	80
Figure 3-12 Clock frequency abnormal reset	81
Figure 3-13 External high-speed oscillator fault reset	81
Figure 3-14 M4 Lockup Reset.....	81
Figure 4-1 Clock System Block Diagram.....	89
Figure 4-2 Clock Frequency Measurement Block Diagram.....	90
Figure 4-3 External high-speed oscillator connection example.....	95
Figure 4-4 Connection example diagram of external clock input.....	96
Figure 4-5 External high-speed oscillator fault detection example.....	97
Figure 4-6 XTAL is selected as the system clock, and XTAL oscillation fault is detected.....	98
Figure 4-7 External low-speed oscillator connection example	99
Figure 4-8 Clock source switching	103
Figure 4-9 Clock division switching.....	104
Figure 4-10 Clock Frequency Measurement Timing Diagram.....	106
Figure 5-1 Power supply diagram	139
Figure 5-2 Battery Backup Domain Power Switching Diagram.....	140
Figure 5-3 Power-on reset, power-down reset waveform	141
Figure 5-4 Brown-out reset waveform.....	142
Figure 5-5 PVD1 Interrupt/Reset Block Diagram	144
Figure 5-6 PVD2 interrupt/Reset Block Diagram	144
Figure 5-7 Power Voltage Detection 1 Interrupt Timing Chart.....	145
Figure 5-8 Power Voltage Detection 1 Reset Timing Chart.....	145
Figure 5-9 Power Voltage Detection 2 Interrupt Operation Timing Chart	146
Figure 5-10 Power Voltage Detction 2 Reset Operation Timing Chart.....	146
Figure 5-11 Schematic diagram of internal voltage sampling	147

Figure 5-12 PTWK _n block diagram.....	157
Figure 7-1 FLASH address distribute.....	194
Figure 7-2 boot swap of FLASH address when OTP is disabled	205
Figure 7-3 boot swap of FLASH address when OTP is enabled.....	205
Figure 7-4 Boot Swap 1.....	206
Figure 7-5 Boot Swap 2.....	206
Figure 9-1 Port basic structure diagram.....	237
Figure 10-1 Interrupt System Block Diagram.....	260
Figure 10-2 Interrupt event selection	282
Figure 10-3 Working diagram of digital filter	283
Figure 11-1 Schematic Diagram of AOS Module	298
Figure 13-1 KEYSAN system block diagram	341
Figure 13-2 Schematic diagram of keyboard scanning function	342
Figure 14-1 Basic block diagram of CTC	347
Figure 14-2 Schematic diagram of CTC calibration	350
Figure 15-1 DMA structure diagram.....	357
Figure 15-2 Linked list diagram	361
Figure 15-3 Example of source address nonsequence mode.....	362
Figure 15-4 An example of the DMA reconfigure transfer.....	364
Figure 15-5 Application Example 1: Memory-to-Memory Transfer	367
Figure 15-6 Application Example 2: Transfer from Memory to Peripheral module	368
Figure 16-1 Functional block diagram.....	393
Figure 16-2 Working diagram of common comparison mode	395
Figure 16-3 Window comparison mode working diagram	397
Figure 17-1 ADC block diagram.....	411
Figure 17-2 Channel Mapping Diagram	414
Figure 17-3 Internal analog channel selection	415
Figure 17-4 Sequence A single scan mode	416
Figure 17-5 Continuous scan	417
Figure 17-6 Double sequence scan mode (Sequence A restarts from interrupted channel)	419
Figure 17-7 Double Sequence Scan Mode (Sequence A restarts from the first channel)	419
Figure 17-8 Analog Watchdog Protection Area (Compare Mode)	420
Figure 17-9 A/D conversion time	421
Figure 17-10 Conversion operation when the averaging function is valid	423
Figure 17-11 Schematic diagram of PGA and SH channels of ADC unit 1 and unit 2	424
Figure 17-12 Dedicated sample-and-hold circuit is valid	425
Figure 17-13 Single-Shot Parallel Trigger Mode (Triple ADCs)	426
Figure 17-14 Single Delay Trigger Mode (Triple ADC).....	428

Figure 17-15	Cyclic Parallel Trigger Mode (Triple ADCs)	429
Figure 17-16	Cyclic Delay Trigger Mode (Two ADCs)	430
Figure 17-17	Cyclic Delay Trigger Mode (Triple ADC)	430
Figure 17-18	ADC Interrupt and Event Output Timing	431
Figure 18-1	D/A conversion channel block diagram.....	456
Figure 18-2	D/A conversion diagram	457
Figure 18-3	A/D conversion priority mode operation diagram	459
Figure 19-1	OTS functional block diagram.....	466
Figure 20-1	Basic block diagram of Timer6.....	473
Figure 20-2	Sawtooth waveform (count up).....	474
Figure 20-3	Triangular waveform	474
Figure 20-4	Comparison output action	476
Figure 20-5	Input capture action	477
Figure 20-6	Hardware update action	477
Figure 20-7	Software synchronization action	479
Figure 20-8	Hardware synchronization action.....	481
Figure 20-9	Pulse width measurement	482
Figure 20-10	Period measurement.....	482
Figure 20-11	Timing of comparison output with Single buffer	484
Figure 20-12	Timing of input capture with double buffer.....	485
Figure 20-13	Count buffer action in sawtooth waveform mode	486
Figure 20-14	Count buffer action 1 in triangular waveform mode	487
Figure 20-15	Count buffer action 2 in triangular waveform mode.....	488
Figure 20-16	Filtering function at input capture port	489
Figure 20-17	Unilaterally aligned independent PWM	490
Figure 20-18	Bilateral symmetrical independent PWM	491
Figure 20-19	Complementary PWM waveform output by software setting of GCMR	492
Figure 20-20	Complementary PWM waveform output by hardware setting of GCMR.....	493
Figure 20-21	Bilateral asymmetrical PWM output.....	494
Figure 20-22	6-phase unilateral aligned independent PWM	495
Figure 20-23	3-phase bilateral symmetrical complementary PWM with dead time	496
Figure 20-24	Action of periodic interval valid request signal	497
Figure 20-25	Position mode-basic counting	499
Figure 20-26	Position counting mode-phase difference counting (1 time counting).....	499
Figure 20-27	Position counting mode-phase difference counting (2 times counting)	499
Figure 20-28	Position counting mode-phase difference counting (4 times counting)	500
Figure 20-29	Position counting mode-direction counting.....	500
Figure 20-30	Revolution counting mode-Z phase counting	501

Figure 20-31	Revolution counting mode-position overflow counting	501
Figure 20-32	Revolution counting mode-mixed counting	502
Figure 20-33	Revolution counting mode-mixed counting Z phase mask operation example 1.....	502
Figure 20-34	Revolution counting mode-mixed counting Z phase mask operation example 2.....	503
Figure 20-35	Example of interrupt & event output in sawtooth waveform mode	513
Figure 21-1	Basic block diagram of HRPWM	556
Figure 21-2	HRPWM adjustment waveform	557
Figure 22-1	Basic block diagram of Timer4.....	563
Figure 22-2	Timer4 sawtooth waveform	564
Figure 22-3	Timer4 triangular waveform	564
Figure 22-4	Timer4 sawtooth wave mode count action	564
Figure 22-5	Timer4 triangle wave mode counting action.....	565
Figure 22-6	Sawtooth wave mode waveform output example.....	566
Figure 22-7	Triangular wave mode waveform output example	566
Figure 22-8	Modify the sawtooth count period when the buffer function is disabled.....	567
Figure 22-9	Modify the sawtooth count period when the buffer function is enabled	568
Figure 22-10	Modify the triangular wave count period when the buffer function is enabled.....	568
Figure 22-11	OCCR buffer data transfer (when the period interval response link is disabled)	569
Figure 22-12	OCCR buffer data transfer (period interval response link is enabled).....	570
Figure 22-13	Output compare buffer data tansfer (OCMR buffer function is enabled)	571
Figure 22-14	SCCR buffer transfer operation (when period interval response link is disabled)	572
Figure 22-15	SCCR buffer transfer operation (when period interval response link is enabled).....	573
Figure 22-16	Sawtooth wave independent PWM output example.....	574
Figure 22-17	Triangular wave independent PWM output example	574
Figure 22-18	Sawtooth wave extended PWM output	575
Figure 22-19	Software realizes complementary PWM output	576
Figure 22-20	Complementary PWM output in dead-time timer mode.....	577
Figure 22-21	Waveform output in dead-time timer mode when the pulse width is abnormal	577
Figure 22-22	Complementary PWM output in dead time timer filter mode	579
Figure 22-23	Period interval response timing diagram	580
Figure 22-24	The period interval response of special event output signal	581
Figure 22-25	Example of counting direction signal output	583
Figure 22-26	Output timing of special event output signal in delay start mode	584
Figure 23-1	EMB structure diagram	608
Figure 23-2	EMB Group Control.....	610
Figure 24-1	TimerA basic block diagram.....	628
Figure 24-2	sawtooth waveform (increasing counting).....	629
Figure 24-3	triangular wave.....	629

Figure 24-4	Compare output action	630
Figure 24-5	Capture input action	631
Figure 24-6	Software synchronization action	632
Figure 24-7	Filtering function of clock input port	632
Figure 24-8	Buffer action in sawtooth mode	633
Figure 24-9	32-bit cascade counting action	634
Figure 24-10	unilaterally aligned PWM output example	635
Figure 24-11	Bilateral Symmetrical PWM Output Example	636
Figure 24-12	Position mode-basic counting	637
Figure 24-13	Position counting mode-phase difference counting (1 times counting)	637
Figure 24-14	Position counting mode-phase difference counting (2 times counting)	638
Figure 24-15	Position counting mode-phase difference counting (4 times counting)	638
Figure 24-16	Position counting mode-direction counting	638
Figure 24-17	Revolution counting mode-Z phase counting	639
Figure 24-18	Revolution counting mode-position overflow counting	639
Figure 24-19	Revolution counting mode-mixed counting	640
Figure 25-1	Timer2 basic block diagram	659
Figure 25-2	Compare output action	661
Figure 25-3	Hardware startup, reset action	662
Figure 25-4	Capture input action	663
Figure 25-5	Pulse width measurement	663
Figure 25-6	Period measurement	664
Figure 25-7	Digital filtering of the TRIG input port	665
Figure 26-1	Timer0 basic block diagram	678
Figure 26-2	Timer0 count timing diagram	680
Figure 27-1	Basic block diagram of RTC	690
Figure 28-1	Hardware startup example	716
Figure 28-2	Software startup example	717
Figure 28-3	Timing examples of various refresh actions (action confirmation, falling edge of refresh request signal, etc.)	718
Figure 28-4	Counter underflow action example	719
Figure 28-5	Example of counter refresh operation	720
Figure 29-1	USART System Block Diagram	727
Figure 29-2	UART data format	729
Figure 29-3	UART transmit data legend 1	731
Figure 29-4	UART transmit data legend 2	731
Figure 29-5	UART internal synchronization and sampling timing	732
Figure 29-6	UART receive data legend 1	734

Figure 29-7	UART receive data legend 2.....	734
Figure 29-8	Multiprocessor Communication Legend	738
Figure 29-9	Multiprocessor Mode Data Format	739
Figure 29-10	Example of transmitting data in multiprocessor mode	740
Figure 29-11	Figure 1 for receiving data in multiprocessor mode.....	741
Figure 29-12	Figure 2 for receiving data in multiprocessor mode.....	741
Figure 29-13	LIN bus data behavior.....	742
Figure 29-14	Smartcard Connection Diagram.....	745
Figure 29-15	Smartcard Mode Synchronization Timing and Sampling Timing Diagram.....	746
Figure 29-16	smartcard mode Data Format.....	746
Figure 29-17	Example of transmitting data in smartcard mode	748
Figure 29-18	Example of receiving data in smartcard mode	748
Figure 29-19	Clock Sync Mode Data Format	750
Figure 29-20	Example 1 of transmitting data in clock synchronization mode	752
Figure 29-21	Example 2 of transmitting data in clock synchronization mode	752
Figure 29-22	Clock synchronization mode receiving data legend 1.....	753
Figure 29-23	Clock synchronization mode receiving data legend 2.....	753
Figure 30-1	I ² C System Block Diagram	773
Figure 30-2	Structure example of I ² C bus	774
Figure 30-3	Timing diagram for the I ² C bus.....	775
Figure 30-4	I ² C bus data format.....	776
Figure 30-5	Sequence diagram of the master sending data in 7-bit address format.....	777
Figure 30-6	Timing diagram of master receiving data in 7-bit address format.....	778
Figure 30-7	Slave Transmit Mode Timing Chart in 7-bit Address Format (Example).....	779
Figure 30-8	7-bit address format slave receiver mode timing diagram	780
Figure 30-9	SCL synchronization timing	781
Figure 30-10	Slave sending timing diagram1	782
Figure 30-11	Slave sending timing diagram2	783
Figure 30-12	Slave Receive Timing Diagram	783
Figure 30-13	Fast ACK/NACK timing diagram.....	784
Figure 30-14	Timing for 7-bit address format	785
Figure 30-15	Timing for 10-bit address format	786
Figure 30-16	General call address receive timing diagram	787
Figure 30-17	SMBus master address receive timing diagram.....	788
Figure 30-18	SMBus alarm response address reception timing diagram	789
Figure 30-19	SMBus default address receive timing diagram.....	790
Figure 30-20	Digital filter circuit block diagram.....	794
Figure 31-1	System Block Diagram.....	813

Figure 31-2 SPI operation mode structure	815
Figure 31-3 Clock synchronous operation mode (3 wire) structure	816
Figure 31-4 Data Format.....	817
Figure 31-5 MSB first, parity check disabled.....	818
Figure 31-6 LSB first, parity check disabled.....	818
Figure 31-7 MSB first, parity check enabled	819
Figure 31-8 LSB first, parity check enabled	819
Figure 31-9 Data transfer format diagram (CPHA=0)	820
Figure 31-10 Data transfer format diagram (CPHA=1)	821
Figure 31-11 Full-duplex synchronous serial communication	821
Figure 31-12 Serial communications with transmit-only.....	822
Figure 31-13 Parity check flow.....	828
Figure 31-14 Data overflow error handling	830
Figure 31-15 Schematic diagram of the operation when the automatic clock stop function is enabled (CPHA=1).....	831
Figure 31-16 Schematic diagram of the operation when the automatic clock stop function is enabled (CPHA=0).....	831
Figure 31-17 Parity error.....	832
Figure 32-1 Block diagram of QSPI	842
Figure 32-2 Default area setting and AHB bus space memory map	843
Figure 32-3 QSPI-ROM space memory map	844
Figure 32-4 Extended SPI protocol operation diagram 1 (Fast Read mode).....	845
Figure 32-5 Extended SPI protocol operation diagram 2 (Fast Read Quad I/O mode)	845
Figure 32-6 Dual SPI protocol operation diagram (Fast Read mode).....	846
Figure 32-7 Quad SPI protocol operation diagram (Fast Read mode)	846
Figure 32-8 Basic timing of serial interface	847
Figure 32-9 Correction diagram of output clock duty cycle when the reference clock selects HCLK divided by three.....	849
Figure 32-10 QSSN setup time configuration diagram.....	850
Figure 32-11 QSSN hold time configuration diagram.....	850
Figure 32-12 Data reception delay time diagram	851
Figure 32-13 Standard Read bus cycle diagram	853
Figure 32-14 Schematic diagram of Fast Fead bus cycle	854
Figure 32-15 Schematic diagram of Fast Read bus cycle with XIP mode selected	854
Figure 32-16 Schematic diagram of Fast Read Dual Output bus cycle.....	855
Figure 32-17 Schematic diagram of Fast Read Dual Output bus cycle with XIP mode selected.....	855
Figure 32-18 Schematic diagram of Fast Read Dual I/O bus cycle	856
Figure 32-19 Schematic diagram of Fast Read Dual I/O with XIP mode selected	857

Figure 32-20 Schematic diagram of Fast Read Quad Output bus cycle	858
Figure 32-21 Schematic diagram of Fast Read Quad Output with XIP mode selected	858
Figure 32-22 Fast Read Quad I/O bus cycle diagram	859
Figure 32-23 Schematic diagram of Fast Read Quad I/O with XIP mode selected.....	860
Figure 32-24 Enter 4-Byte Mode instruction bus cycle diagram	860
Figure 32-25 Exit 4-Byte Mode instruction bus cycle.....	861
Figure 32-26 Schematic diagram of Write Permission instruction bus cycle.....	861
Figure 32-27 Schematic diagram of a single flash data read operation with independent conversion	862
Figure 32-28 Schematic diagram of data read operation using prefetch function	863
Figure 32-29 Schematic diagram of data read operation using the QSPI bus cycle extension function	864
Figure 32-30 XIP mode control diagram	865
Figure 33-1 I2S system block diagram.....	880
Figure 33-2 I2S Philips protocol waveform (16/32 bit full precision).....	882
Figure 33-3 I2S Philips protocol waveform (16-bit data packed in 32-bit frames).....	883
Figure 33-4 I2S Philips protocol waveform (24-bit data packed in 32-bit frame)	883
Figure 33-5 I2S MSB protocol waveform (16/32-bit full precision)	883
Figure 33-6 I2S MSB protocol waveform (16-bit data packed in 32-bit frame)	884
Figure 33-7 I2S MSB protocol waveform (24-bit data packed in 32-bit frame)	884
Figure 33-8 I2S LSB protocol waveform (16/32-bit full precision)	885
Figure 33-9 I2S LSB protocol waveform (16-bit data packed in 32-bit frame)	885
Figure 33-10 I2S LSB protocol waveform (24-bit data packed in 32-bit frame)	885
Figure 33-11 I2S PCM protocol waveform (16/32-bit full precision)	886
Figure 33-12 I2S PCM protocol waveform (16-bit data packed in 32-bit frame).....	886
Figure 33-13 I2S PCM protocol waveform (24-bit data packed in 32-bit frame).....	887
Figure 33-14 Audio sampling frequency definition	887
Figure 33-15 Clock Generator Architecture.....	888
Figure 33-16 The host only receives and temporarily stops receiving.....	893
Figure 33-17 Mode 1 of PCM short frame retransmission after host transmission pauses.....	894
Figure 33-18 Mode 2 of PCM short frame retransmission after host sending pause	895
Figure 34-1 USBHS System Block Diagram.....	906
Figure 34-2 USBHS host mode system construction diagram.....	909
Figure 34-3 USBHS device mode system construction diagram	913
Figure 34-4 Schematic diagram of USBHS dynamically updating USBHS_HFIR register	918
Figure 34-5 Schematic diagram of FIFO architecture in USBHS host mode	919
Figure 34-6 Schematic diagram of FIFO architecture in USBHS device mode.....	921
Figure 34-7 USBHS Control Status Register Memory Map.....	950

Figure 35-1	USBFS system block diagram	1027
Figure 35-2	USBFS host mode system construction diagram	1029
Figure 35-3	USBFS device mode system construction diagram.....	1033
Figure 35-4	Schematic diagram of USBFS dynamically updating USBFS_HFIR register	1038
Figure 35-5	Schematic diagram of FIFO architecture in USBFS host mode.....	1039
Figure 35-6	Schematic diagram of FIFO architecture in USBFS device mode	1041
Figure 35-7	USBFS Control Status Register Memory Map	1069
Figure 36-1	CANFD system block.....	1138
Figure 36-2	CAN Bit Time Definition	1139
Figure 36-3	Schematic of CANFD TBUF register write transmit buffer	1142
Figure 36-4	Schematic of CANFD RBUF register read receive buffer	1143
Figure 36-5	Access to the CANFD Acceptance Filters.....	1143
Figure 36-6	Schematic of CANFD LBMI and LBME.....	1147
Figure 36-7	Transmitter Delay	1152
Figure 37-1	CAN system block.....	1182
Figure 37-2	CAN Bit Time Definition	1183
Figure 37-3	Schematic of CAN TBUF register write transmit buffer	1184
Figure 37-4	Schematic of CAN RBUF register read receive buffer	1185
Figure 37-5	Access to the CAN Acceptance Filters.....	1185
Figure 37-6	Schematic of CAN LBMI and LBME	1189
Figure 39-1	Ethernet MAC Controller Architecture Diagram	1254
Figure 39-2	MII interface connection diagram	1258
Figure 39-3	RMII interface connection diagram	1260
Figure 39-4	SMI interface connection diagram	1261
Figure 39-5	MAC frame structure.....	1263
Figure 39-6	MII/RMII send bit sequence	1270
Figure 39-7	Conflict-free sending diagram	1270
Figure 39-8	Conflict send map.....	1271
Figure 39-9	Transmission diagram in MII/RMII mode.....	1271
Figure 39-10	MII/RMII receive bit sequence	1274
Figure 39-11	Error-free sending of graphs	1275
Figure 39-12	Error sending image	1275
Figure 39-13	Reception diagram under false carrier indication	1275
Figure 39-14	Remote Wakeup Frame Filter Register	1280
Figure 39-15	Clock synchronization diagram.....	1283
Figure 39-16	System time calibration.....	1286
Figure 39-17	Descriptor structure.....	1290
Figure 39-18	TxDMA action flow (non-OSF mode)	1295

Figure 39-19 TxDMA operation flow (OSF mode).....	1297
Figure 39-20 RxDMA action flow.....	1305
Figure 39-21 ETHMAC interrupt scheme.....	1312
Figure 39-22 DMA interrupt composition.....	1313
Figure 40-1 EXMC Architecture Diagram.....	1382
Figure 40-2 External space address allocation	1384
Figure 40-3 SMC's address space division	1385
Figure 40-4 DMC address space division	1386
Figure 40-5 NFC address space division	1387
Figure 40-6 SMC state diagram	1391
Figure 40-7 SMC initial setting process.....	1392
Figure 40-8 Basic timing of single read operation (asynchronous mode (RSYN=0) & 16-bit width (MW=01))	1394
Figure 40-9 Basic sequence of single read operation (asynchronous mode (RSYN=0) & 32-bit width (MW=10))	1394
Figure 40-10 Basic timing sequence of single read operation (synchronous mode (RSYN=1) & 16-bit bit width (MW=01)).....	1395
Figure 40-11 Basic timing sequence of single read operation (synchronous mode (RSYN=1) & 32-bit width (MW=10)).....	1395
Figure 40-12 Address data line multiplexing single read operation basic timing (asynchronous mode (RSYN=0) & 16-bit width (MW=01))	1396
Figure 40-13 Address data line multiplexing single read operation basic timing (asynchronous mode (RSYN=0) & 32-bit width (MW=10))	1397
Figure 40-14 Basic timing of single write operation (asynchronous mode (WSYN=0) & 16-bit width (MW=01) & BLS=0)	1398
Figure 40-15 Basic timing of single write operation (asynchronous mode (WSYN=0) & 32-bit bit width (MW=10) & BLS=1)	1399
Figure 40-16 Basic timing of single write operation (synchronous mode (WSYN=1) & 16-bit bit width (MW=01) & BLS=1)	1399
Figure 40-17 Basic timing of single write operation (synchronous mode (WSYN=1) & 32-bit bit width (MW=10) & BLS=0)	1400
Figure 40-18 Address data line multiplexing single write action basic timing (asynchronous mode (WSYN=0) & 16-bit width (MW=01)).....	1401
Figure 40-19 Basic Timing Of Address Data Multiplexing Burst Read Action (synchronous/asynchronous mode (RSYN=1/0) &16 bit width (MW=01) &RBL=001)	1402
Figure 40-20 Basic Timing Of Address Data Non-Multiplexing Burst Read Action (synchronous/asynchronous mode (RSYN=1/0) &16 bit width (MW=01) &RBL=001)	1403

Figure 40-21 Basic Timing Of Address Data Multiplexing Burst Read Action (synchronous/asynchronous mode (RSYN=1/0) & 32-bit bit width (MW=10) &RBL=001)	1404
Figure 40-22 Basic Timing Of Address Data Non-Multiplexing Burst Read Action (synchronous/asynchronous mode (RSYN=1/0) & 32-bit bit width (MW=10) &RBL=001)	1405
Figure 40-23 Burst write operation basic timing (asynchronous mode (WSYN=0) & 32 bit width (MW=10) & WBL=011)	1406
Figure 40-24 Row activation to read and write operation timing	1408
Figure 40-25 Command mode precharge	1408
Figure 40-26 Automatically precharge after reading and writing.....	1409
Figure 40-27 Entry and exit timing of self-refresh action.....	1409
Figure 40-28 DMC state diagram.....	1412
Figure 40-29 DMC initial setup process	1413
Figure 40-30 Basic timing of single read operation (16-bit width (DMCMW=00)).....	1414
Figure 40-31 Basic timing of single write operation (32-bit width (DMCMW=01))	1415
Figure 40-32 Burst read operation basic timing (32-bit width (DMCMW=10) & BURST=010)	1416
Figure 40-33 Burst write operation basic timing (32-bit width (DMCMW=10) & BURST=011)	1416
Figure 40-34 DMC_CKE port output control timing	1417
Figure 40-35 NFC timing control diagram	1455
Figure 41-1 DVP basic block diagram	1463
Figure 41-2 DVP signal waveform diagram	1465
Figure 41-3 Single frame mode data acquisition action.....	1467
Figure 41-4 Continuous Mode Data Acquisition Action	1467
Figure 41-5 Frame acquisition frequency control.....	1468
Figure 41-6 DVP window setting diagram	1469
Figure 41-7 Window cropping data acquisition plot	1470
Figure 42-1 Schematic diagram of AES encryption and decryption process.....	1479
Figure 42-2 AES encryption flow chart.....	1480
Figure 42-3 HASH algorithm flow chart.....	1491
Figure 42-4 TRNG system block diagram.....	1505
Figure 43-1 CRC Application Diagram.....	1508
Figure 44-1 Hardware Triggered Boot Plus Mode Example Timing Diagram	1514
Figure 44-2 Triangle wave output mode	1516
Figure 44-3 Triangular wave output example timing diagram	1517
Figure 44-4 Incremental sawtooth output mode.....	1519
Figure 44-5 Incremental Ramp Output Example Timing Diagram.....	1519
Figure 44-6 Decreasing sawtooth output mode.....	1521
Figure 44-7 Decreasing Ramp Output Example Timing Diagram.....	1521
Figure 45-1 Schematic diagram of sine operation angle value.....	1539

Figure 46-1 FIR basic block diagram.....	1544
Figure 47-1 Debug control system.....	1552
Figure 47-2 Debug control system.....	1554
Figure 47-3 JTAG-DP to SW-DP switching timing	1554
Figure 47-4 TPIU block diagram.....	1563

Overview

The HC32A4A0 series is a high-performance MCU based on ARM® Cortex®-M4 32-bit RISC CPU with a maximum operating frequency of 240MHz. The Cortex-M4 core integrates a floating-point arithmetic unit (FPU) and a DSP to implement single-precision floating-point arithmetic operations, supports all ARM single-precision data processing instructions and data types, and supports the complete DSP instruction set. The kernel integrates the MPU unit and superimposes the DMAC dedicated MPU unit at the same time to ensure the safety of system operation.

The HC32A4A0 series integrates high-speed on-chip memory, including a maximum of 2MB of Flash and a maximum of 512KB of SRAM. Integrated Flash access acceleration unit to achieve single cycle program execution of the CPU on Flash. The polled bus matrix supports multiple bus hosts to access memory and peripherals simultaneously, improving performance. The bus host includes CPU, DMA, USB dedicated DMA, ETHMAC dedicated DMA. In addition to the bus matrix, data transfer between peripherals is supported, and basic arithmetic operations and events are triggered each other, which can significantly reduce the transaction processing load of the CPU.

The HC32A4A0 series integrates rich peripheral functions. Including 3 independent 12bit 2.5MSPS ADC, 4 gain adjustable PGA, 4 12-bit 15MSPS DAC, 4 high-speed voltage comparator (CMP), 8 multi-function PWM Timer (Timer6), support 16 complementary PWM Output, 16 high-precision PWM (HRPWM) extend the resolution of Timer6 PWM signal, 3 motor PWM Timer (Timer4) support 18 complementary PWM outputs, 12 16bit general Timer (TimerA) support 6 3-phase quadrature Code input and 48 independent PWM outputs for Duty, 22 serial communication interfaces (I2C/UART/SPI), 1 QSPI interface, 2 CAN, 4 I2S support audio PLL, 2 SDIO, 1 ETHMAC, USBFS Controller and USBHS Controller with built-in USBFS PHY, 1 external expansion bus controller, including NFC controller, SMC controller and DMC controller, 1 digital video interface DVP, 1 math operation unit (MAU) and 4 filtering Mathematics Accelerator (FMAC).

HC32A4A0 series supports wide voltage range (1.8-3.6V), wide temperature range (-40-105°C) and various low power consumption modes. Supports fast wake-up in low power consumption mode, the fastest wake-up to STOP mode is 2us, and the fastest wake-up to PowerDown mode is 25us.

Typical application

The HC32A4A0 series provides 100pin and 176pin LQFP packages, which are suitable for automotive electronics, such as T-Box, BMS controller, gateway, car light control and other applications.

About this manual

This manual mainly introduces the function, operation and usage of the chip. For the specifications of the chip, please refer to the corresponding "Datasheet".

1 Memory Mapping

1.1 Memory mapping

The MCU supports 4 GB of linear address space from 0000 0000h to FFFF FFFFh, which contains code and data. Refer to the following table for more memory mapping.

Table 1-1 Memory map

Memory classification		Start address	End address	Space size	Module * 3	Protection * 4	Note
System	Private Peripheral External Bus	0xE0100000	0xFFFFFFFF	511MB	Reserved		Custom space
		0xE00FF000	0xE00FFFFF	4KB	ROMTABLE		Debug Control Register Area
		0xE0042400	0xE00FEFFF	755KB			
		0xE0042000	0xE00423FF	1KB	DBG C		
		0xE0041000	0xE0041FFF	4KB	ETM		
		0xE0040000	0xE0040FFF	4KB	TPIU		
	Private Peripheral Internal Bus	0xE000F000	0xE003FFFF	196KB			System control space NVIC/MPU, etc.
		0xE000E000	0xE000EFFF	4KB	SCS		
		0xE0003000	0xE000DFFF	44KB			
		0xE0002000	0xE0002FFF	4KB	FPB		
		0xE0001000	0xE0001FFF	4KB	DWT		
		0xE0000000	0xE0000FFF	4KB	ITM		
External device	-	0xA0000000	0xFFFFFFFF	1024MB	Reserved		
External memory	AHB5 clock: HCLK	0x98000000	0x9FFFFFFF	128MB	QSPI		
		0x88200000	0x97FFFFFF	254MB	Reserved		
	AHB5 clock: BCLK	0x88100000	0x881FFFFF	1MB	NFC		
		0x88000800	0x880FFFFFF	1022KB	BLANK		
		0x88000400	0x880007FF	1KB	DMCR		
		0x88000000	0x880003FF	1KB	SMCR		
		0x80000000	0x87FFFFFF	128MB	DMC		
		0x60000000	0x7FFFFFFF	512MB	SMC		
Peripheral	AHB4 clock: PCLK1	-	0x44000000	0x5FFFFFFF	448MB	Reserved	
			0x42000000	0x43FFFFFF	32MB	PeriBitBand	
			0x40100000	0x41FFFFFF	31MB	Reserved	
		0x400C0000	0x400FFFFFF	256KB	USBHS		
	AHB2 clock: PCLK1						
		0x40080000	0x400BFFFF	256KB	USBFS		
		0x40078800	0x4007FFFF	30KB	BLANK		
		0x40078400	0x400787FF	1KB	SDIOC_2		
		0x40078000	0x400703FF	1KB	CAN_2		

Memory classification		Start address	End address	Space size	Module * 3	Protection * 4	Note	
Peripheral I	AHB3 clock: PCLK1	0x40070400	0x40077FFF	31KB	BLANK			
		0x40070000	0x400703FF	1KB	SDIOC_1			
		0x40060000	0x4006FFFF	64KB	ETHMAC			
	AHB1 clock: HCLK	0x40059000	0x4005FFFF	28KB	BLANK			
		0x40058C00	0x40058FFF	1KB	FMAC_4			
		0x40058800	0x40058BFF	1KB	FMAC_3			
		0x40058400	0x400587FF	1KB	FMAC_2			
		0x40058000	0x400583FF	1KB	FMAC_1			
		0x40057C00	0x40057FFF	1KB	DCU_8			
		0x40057800	0x40057BFF	1KB	DCU_7			
		0x40057400	0x400577FF	1KB	DCU_6			
		0x40057000	0x400573FF	1KB	DCU_5			
		0x40056C00	0x40056FFF	1KB	DCU_4			
		0x40056800	0x40056BFF	1KB	DCU_3			
		0x40056400	0x400567FF	1KB	DCU_2			
		0x40056000	0x400563FF	1KB	DCU_1			
		0x40055800	0x40055BFF	1KB	BLANK			
		0x40055800	0x40055BFF	1KB	DVP			
Peripheral I	APB4 Clock: PCLK3	0x40055400	0x400557FF	1KB	PERIC		Peripheral Module Control Register	
		0x40055000	0x400553FF	1KB	MAU			
		0x40054400	0x40054FFF	3KB	BLANK		No Access	
		0x40054000	0x400543FF	1KB	CMU	with protection		
		0x40053800	0x40053FFF	2KB	GPIO			
		0x40053400	0x400537FF	1KB	DMA_2			
		0x40053000	0x400533FF	1KB	DMA_1			
		0x40052000	0x40052FFF	4KB	BLANK			
		0x40051000	0x40051FFF	4KB	INTC	with protection		
		0x40050C00	0x40050FFF	1KB	KEYSCAN			
		0x40050800	0x40050BFF	1KB	RAMIF	with protection		
		0x40050400	0x400507FF	1KB	BLANK			
		0x40050000	0x400503FF	1KB	DMPU	with protection		
		0x4004F800	0x4004FFFF	2KB	BLANK			
		0x4004F400	0x4004F7FF	1KB	I2C_6			
Peripheral I		0x4004F000	0x4004F3FF	1KB	I2C_5			
		0x4004EC00	0x4004EFFF	1KB	I2C_4			
		0x4004E800	0x4004EBFF	1KB	I2C_3			
		0x4004E400	0x4004E7FF	1KB	I2C_2			

Memory classification		Start address	End address	Space size	Module * 3	Protection * 4	Note
		0x4004E000	0x4004E3FF	1KB	I2C_1		
		0x4004D000	0x4004DFFF	1KB	BLANK		
		0x4004CC00	0x4004CFFF	1KB	EMU	with protection	
		0x4004C800	0x4004CBFF	1KB	BLANK		
		0x4004C400	0x4004C7FF	1KB	VBAT	with protection	
		0x4004C000	0x4004C3FF	1KB	RTC	with protection	
		0x4004AC00	0x4004BFFF	5KB	BLANK		
		0x4004A800	0x4004ABFF	1KB	OTS		
		0x4004A400	0x4004A7FF	1KB	CMP_3,4		
		0x4004A000	0x4004A3FF	1KB	CMP_1,2		
		0x40049C00	0x40049FFF	1KB	CTC		
		0x40049800	0x40049BFF	1KB	BLANK		
		0x40049400	0x400497FF	1KB	SWDT	with protection	
		0x40049000	0x400493FF	1KB	WDT	with protection	
		0x40048800	0x40048FFF	2KB	BLANK		
	APB3 Clock: PCLK4	0x40048400	0x400487FF	1KB	FCM		
		0x40048000	0x400483FF	1KB	FCG	with protection	
		0x40042400	0x40047FFF	23KB	BLANK		
		0x40042000	0x400423FF	1KB	TRNG	with protection	
		0x40041800	0x40041FFF	2KB	BLANK		
		0x40041400	0x400417FF	1KB	DAC_2		
		0x40041000	0x400413FF	1KB	DAC_1		
		0x40040C00	0x40040FFF	1KB	BLANK		
		0x40040800	0x40040BFF	1KB	ADC_3		
Periphera l	APB5 Clock: PCLK0	0x40040400	0x400407FF	1KB	ADC_2		
		0x40040000	0x400403FF	1KB	ADC_1		
		0x4003C400	0x4003FFFF	15KB	BLANK		
		0x4003C000	0x4003C3FF	1KB	HRPWM		
		0x4003B000	0x4003BFFF	4KB	BLANK		
		0x4003AC00	0x4003AFFF	1KB	TimerA_4		
		0x4003A800	0x4003ABFF	1KB	TimerA_3		
		0x4003A400	0x4003A7FF	1KB	TimerA_2		
		0x4003A000	0x4003A3FF	1KB	TimerA_1		
		0x40038C00	0x40039FFF	5KB	BLANK		
		0x40038800	0x40038BFF	1KB	Timer4_3		
		0x40038400	0x400387FF	1KB	Timer4_2		
		0x40038000	0x400383FF	1KB	Timer4_1		
		0x40028000	0x40037FFF	64KB	BLANK		

Memory classification		Start address	End address	Space size	Module * 3	Protection * 4	Note
Peripheral	APB2 Clock: PCLK1	0x40027C00	0x40027FFF	1KB	TimerA_12		
		0x40027800	0x40027BFF	1KB	TimerA_11		
		0x40027400	0x400277FF	1KB	TimerA_10		
		0x40027000	0x400273FF	1KB	TimerA_9		
		0x40026C00	0x40026FFF	1KB	TimerA_8		
		0x40026800	0x40026BFF	1KB	TimerA_7		
		0x40026400	0x400267FF	1KB	TimerA_6		
		0x40026000	0x400263FF	1KB	TimerA_5		
		0x40025800	0x40025FFF	2KB	BLANK		
		0x40025400	0x400257FF	1KB	Timer2_4		
		0x40025000	0x400253FF	1KB	Timer2_3		
		0x40024C00	0x40024FFF	1KB	Timer2_2		
		0x40024800	0x40024BFF	1KB	Timer2_1		
		0x40024400	0x400247FF	1KB	Timer0_2		
		0x40024000	0x400243FF	1KB	Timer0_1		
		0x40022800	0x40023FFF	6KB	BLANK		
		0x40022400	0x400227FF	1KB	I2S_4		
		0x40022000	0x400223FF	1KB	I2S_3		
Peripheral	APB1 Clock: PCLK1	0x40021C00	0x40021FFF	1KB	USART_10		
		0x40021800	0x40021BFF	1KB	USART_9		
		0x40021400	0x400217FF	1KB	USART_8		
		0x40021000	0x400213FF	1KB	USART_7		
		0x40020C00	0x40020FFF	1KB	USART_6		
		0x40020800	0x40020BFF	1KB	SPI_6		
		0x40020400	0x400207FF	1KB	SPI_5		
		0x40020000	0x400203FF	1KB	SPI_4		
		0x4001E800	0x4001FFFF	6KB	BLANK		
		0x4001E400	0x4001E7FF	1KB	I2S_2		
Peripheral	APB1 Clock: PCLK1	0x4001E000	0x4001E3FF	1KB	I2S_1		
		0x4001DC00	0x4001DFFF	1KB	USART_5		
		0x4001D800	0x4001DBFF	1KB	USART_4		
		0x4001D400	0x4001D7FF	1KB	USART_3		
		0x4001D000	0x4001D3FF	1KB	USART_2		
		0x4001CC00	0x4001CFFF	1KB	USART_1		
		0x4001C800	0x4001CBFF	1KB	SPI_3		
		0x4001C400	0x4001C7FF	1KB	SPI_2		
		0x4001C000	0x4001C3FF	1KB	SPI_1		

Memory classification		Start address	End address	Space size	Module * 3	Protection * 4	Note
SRAM	AHB4 clock: PCLK1	0x40018000	0x4001BFFF	16KB	Timer6		Counting clock: PCLK0
		0x40017C00	0x40017FFF	1KB	EMB		
		0x40010C00	0x40017BFF	28KB	BLANK		
		0x40010800	0x40010BFF	1KB	AOS		Internal trigger event register area
		0x40010400	0x400107FF	1KB	EFM	with protection	
		0x40010000	0x400103FF	1KB	BLANK		
	SRAM Clock: HCLK	0x40009400	0x4000FFFF	27KB	BLANK		
		0x40009000	0x400093FF	1KB	CAN_1		
		0x40008C00	0x40008FFF	1KB	CRC	with protection	
		0x40008800	0x40008BFF	1KB	BLANK		
		0x40008400	0x400087FF	1KB	HASH	with protection	
		0x40008000	0x400083FF	1KB	AES	with protection	
		0x40000000	0x40007FFF	32KB	Reserved		
CODE	OTP, Flash Clock: HCLK	0x24000000	0x3FFFFFFF	448MB	Reserved		
		0x22000000	0x23FFFFFF	32MB	SRAMBitBand		
		0x20100000	0x21FFFFFF	31MB	Reserved		
		0x200F1000	0x200FFFFFF	60KB	BLANK		
		0x200F0000	0x200F0FFF	4KB	SRAMB		ECC RAM
		0x20060000	0x200EFFFF	576KB	BLANK		
		0x20058000	0x2005FFFF	32KB	SRAM4		ECC RAM
		0x20040000	0x20057FFF	96KB	SRAM3		
		0x20020000	0x2003FFFF	128KB	SRAM2		
		0x20000000	0x2001FFFF	128KB	SRAM1		
	SRAM Clock: HCLK	0x1FFE0000	0x1FFFFFFF	128KB	SRAMH		
		0x03004000	0x1FFDFFFF	463.86MB	BLANK		
	OTP, Flash Clock: HCLK	0x0300400C	0x03005FFF	8180B	BLANK		
		0x03004000	0x0300400B	12 B	Data security protection		Used to configure data security protection
		0x03002004	0x03003FFF	8187B	BLANK		
		0x03002000	0x03002003	4 B	Boot exchange		Only CPU can access
		0x03001ADC	0x03001FFF	1316B	BLANK		
		0x03000000	0x03001AD B	6876B	OTP		Only CPU can access
		-	0x02100000	0x02FFFFFF	15M	BLANK	

Memory classification		Start address	End address	Space size	Module * 3	Protection * 4	Note
REMAP clock: HCLK	0x02080000	0x020FFFFF	512KB	REMAP1			address remapping area 1
	0x02000000	0x0207FFFF	512KB	REMAP0			Address remapping area 0
	-	0x00200000	0x01FFFFFF	30M	BLANK		
	Flash Clock: HCLK	0x00100000	0x001FFFFF	1MB	Embedded Flash 1		
		0x00000000	0x000FFFFF	1MB	Embedded Flash 0		

*1 Please refer to the ARM Cortex-M4 Processor Technical Reference Manual " System address map".

*2 Please refer to the bus chapter for bus description.

*3 Reserved: Accessing the bus will cause a bus error; BLANK: Write access is invalid, read access reads 0.

*4 Modules with protection function only support CPU access in privileged mode when the protection function is enabled. For specific registers and descriptions, please refer to [Storage Protection Unit (MPU)] chapter.

1.2 External memory mapping

The QSPI space is divided into 2 segments, including the QSPI I/O register space of 64MB and the external QSPI device space of 64MB. Please refer to the following figure for the distribution relationship.

Table 1-2 QSPI address space allocation

QSPI	0x98000000	0x9FFFFFFF	128MB	QSPI I/O registers	0x9C000000	0x9FFFFFFF	64MB
				External QSPI device	0x98000000	0x9BFFFFFF	64MB

1.3 Bit band region

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

In this MCU, both the peripheral register and the SRAM are mapped to a bit band area, which enables read and write operations of a single bit band. These operations only apply to Cortex™-M4F accesses and have no effect on other bus masters such as DMA.

1.4 Address remapping

This MCU provides a memory address remapping function, MMF_REMCR0 and MMF_REMCR1 can be used to configure the remapping target address. This MCU provides 2 remapping addresses for free configuration, and the target address can be the main flash memory address or the high-speed SRAM address.

Remap address 0:

0x0200_0000H~0x0208_0000H (depending on the set remap size MMF_REMCR0.RM0SIZE [4:0])

Remap address 1:

0x0208_0000H~0x0210_0000H (depending on the set remapping size MMF_REMCR1.RM1SIZE [4:0])

When the remapping function is enabled (MMF_REMCR0.EN0=1 or MMF_REMCR1.EN1=1), the address correspondence table is as follows:

Table 1-3 Example of target address configuration main flash (MMF_REMCR0.EN0=1 or MMF_REMCR1.EN1=1)

	Remap address (CPU address—CPUADDR[31:0])	Remap target address (main flash address)		
		Upper 12 bits	Middle 6	Lower 14 bits
RM0SIZE[4:0]=01110 case (Remap space: 16K)	0x0200_0000h~0x0200_3FFFh	All 0s	RM0TADDR[5:0]	CPUADDR[13:0]
RM0SIZE[4:0]=01111 case (Remap space: 32K)	0x0200_0000h~0x0200_7FFFh	All 0s	RM0TADDR[4:0] CPUADDR[14]	CPUADDR[13:0]
RM1SIZE[3:0]=10000 case (Remap space: 64K)	0x0208_0000h~0x0208_7FFFh	All 0s	RM1TADDR[3:0] CPUADDR[15:14]	CPUADDR[13:0]
RM1SIZE[4:0]=10001 case (Remap space: 128K)	0x0208_0000h~0x0208_FFFFh	All 0s	RM1TADDR[2:0] CPUADDR[16:14]	CPUADDR[13:0]

Table 1-4 Example of target address configuration high-speed SRAM (MMF_REMCR0.EN0=1 or MMF_REMCR1.EN1=1)

	Remap address (CPU address—CPUADDR[31:0])	Remap target address (main flash address)		
		High 3	Middle 17	Lower 12 bits
RM0SIZE[4:0]=01100 case (Remap space: 4K)	0x0200_0000h~0x0200_0FFFh	All 0s	RM0TADDR[16:0]	CPUADDR[11:0]
RM0SIZE[4:0]=01101 case (Remap space: 8K)	0x0200_0000h~0x0200_1FFFh	All 0s	RM0TADDR[15:0] CPUADDR[12]	CPUADDR[11:0]
RM1SIZE[3:0] = 01100 case (Remap space: 4K)	0x0208_0000h~0x0208_0FFFh	All 0s	RM1TADDR[16:0]	CPUADDR[11:0]
RM1SIZE[4:0] = 01101 case (Remap space: 8K)	0x0208_0000h~0x0208_1FFFh	All 0s	RM1TADDR[15:0] CPUADDR[12]	CPUADDR[11:0]

1.5 Remap register

There are three registers in the remapping module. The address space is as follows:

Register base address: 0x40010500

Table 1-5 Register list

Register name	Symbol	Offset address	Bit width	Reset value
Access protection register	MMF_REMPRT	0000h	32	00000000h
Remap Register 0	MMF_REMCR0	0004h	32	00000000h
Remap Register 1	MMF_REMCR1	0008h	32	00000000h

1.5.1 Access Protection Register (MMF_REMPRT)

Reset value: 0x00000000h

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
-															
<hr/>															
Bit	Symbol	Bit name	Description										Read and write		
b15~0	MMF_REMPRT[15:0]	Protection register	Registers MMF_REMCR0 and MMF_REMCR1 are write protected: First write 0123H to MMF_REMPRT[15:0] and then write 3210H to release the protection; When the registers MMF_REMCR0 and MMF_REMCR1 are write-protected, the read register is 0 When the registers MMF_REMCR0 and MMF_REMCR1 release the write protection state, the read register is 1										R/W		

1.5.2 Remap Register 0 (MMF_REMCRO)

Reset value: 0x0000_0000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
EN0	-	RM0TADDR[16:4]														
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0	
RM0TADDR[3:0]				-						RM0SIZE[4:0]						
<hr/>																
Bit	Marking		Place name		Function								Read and write			
b31	EN0		Remap 0 significant bits		0: Remap 0 is invalid 1: remapping 0 is valid								R/W			
b30~29		Reserved		-		Read as "0", write as "0"								R		
b28~12		RM0TADDR[16:0]		Remap target address		The effective number of digits is related to the setting of RM0SIZE[4:0]. For the setting, please refer to table 3.1 Example when the address remapping function is valid								R/W		
b11~b5		Reserved		-		Read as "0", write as "0"								R		
b4~b0		RM0SIZE[4:0]		Remap space size		Sets the size of the remapping 0 area 00000~01011: Reserved, setting prohibited 01100: 4KByte 01101: 8KByte 01110: 16KByte 01111: 32KByte 10000: 64KByte 10001: 128KByte 10010: 256KByte 10011: 512KByte 10100~11111: Reserved, setting prohibited								R/W		

1.5.3 Remap Register 1 (MMF_REMCR1)

Reset value: 0x0000_0000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
EN1	-	RM1TADDR[16:4]														
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0	
RM1TADDR[3:0]				-					RM1SIZE[4:0]							
<hr/>																
Bit	Marking	Place name			Function										Read and write	
b31	EN1	Remap 1 significant bit			0: Remap 1 is invalid 1: Remap 1 is valid										R/W	
b30 ~29	Reserved	-			Read as "0", write as "0"										R	
b28 ~12	RM1TADD R[16:0]	Remap target address			The effective number of digits is related to the setting of RM1SIZE[4:0]. Settings can refer to Table 1-1 Example when the address remapping function is valid										R/W	
b11 ~b5	Reserved	-			Read as "0", write as "0"										R	
b4~ b0	RM1SIZE[4 :0]	Remap space size			Sets the size of the remap 1 region 00000~01011: Reserved, setting prohibited 01100: 4KByte 01101: 8KByte 01110: 16KByte 01111: 32KByte 10000: 64KByte 10001: 128KByte 10010: 256KByte 10011: 512KByte 10100~11111: Reserved, setting prohibited										R/W	

2 Bus Architecture (BUS)

2.1 Overview

The main bus system is composed of 32-bit multilayer AHB bus matrix, which can interconnect the following host bus and slave bus.

- 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
 - USBHS_DMA bus
 - ETHMAC_DMA bus
- Slave bus
 - Flash ICODE bus
 - Flash DCODE bus
 - Flash MCODE bus (bus for other hosts other than CPU to access Flash)
 - High-speed SRAMH (SRAMH 128kB) bus
 - System SRAMA (SRAM1 128KB) bus
 - System SRAMB (SRAM2 128KB) bus
 - System SRAMC (SRAM3 96KB, SRAM4 32KB, SRAMB 4KB)
 - APB1 peripheral bus (EMB/Timers/SPI/USART/I2S/HRPWM/EFM)
 - APB2 peripheral bus (Timers/SPI/USART/I2S)
 - APB3 peripheral bus (ADC/DAC/TRNG)
 - APB4 peripheral bus (FCM/WDT/SWDT/CMP/EMU/CTC/OTS/RTC/VBAT/WKTM/I2C)
 - APB5 peripheral bus (Timers/HRPWM)
 - AHB1 peripheral bus (DMPU/KEYSCAN/INTC/DCU/GPIO/DMA/CMU/DVP/MAU/FMAC)
 - AHB2 peripheral bus (CAN/ SDIOC/USBFS)
 - AHB3 peripheral bus (SDIOC/ETHMAC)
 - AHB4 peripheral bus (AES/HASH/CRC/CAN/USBHS)
 - AHB5 peripheral bus (SMC/DMC/SMCR/DMCR/NFC/QSPI)

With the help of the bus matrix, high-efficiency concurrent access from the host bus to the slave bus can be realized.

2.2 Bus architecture

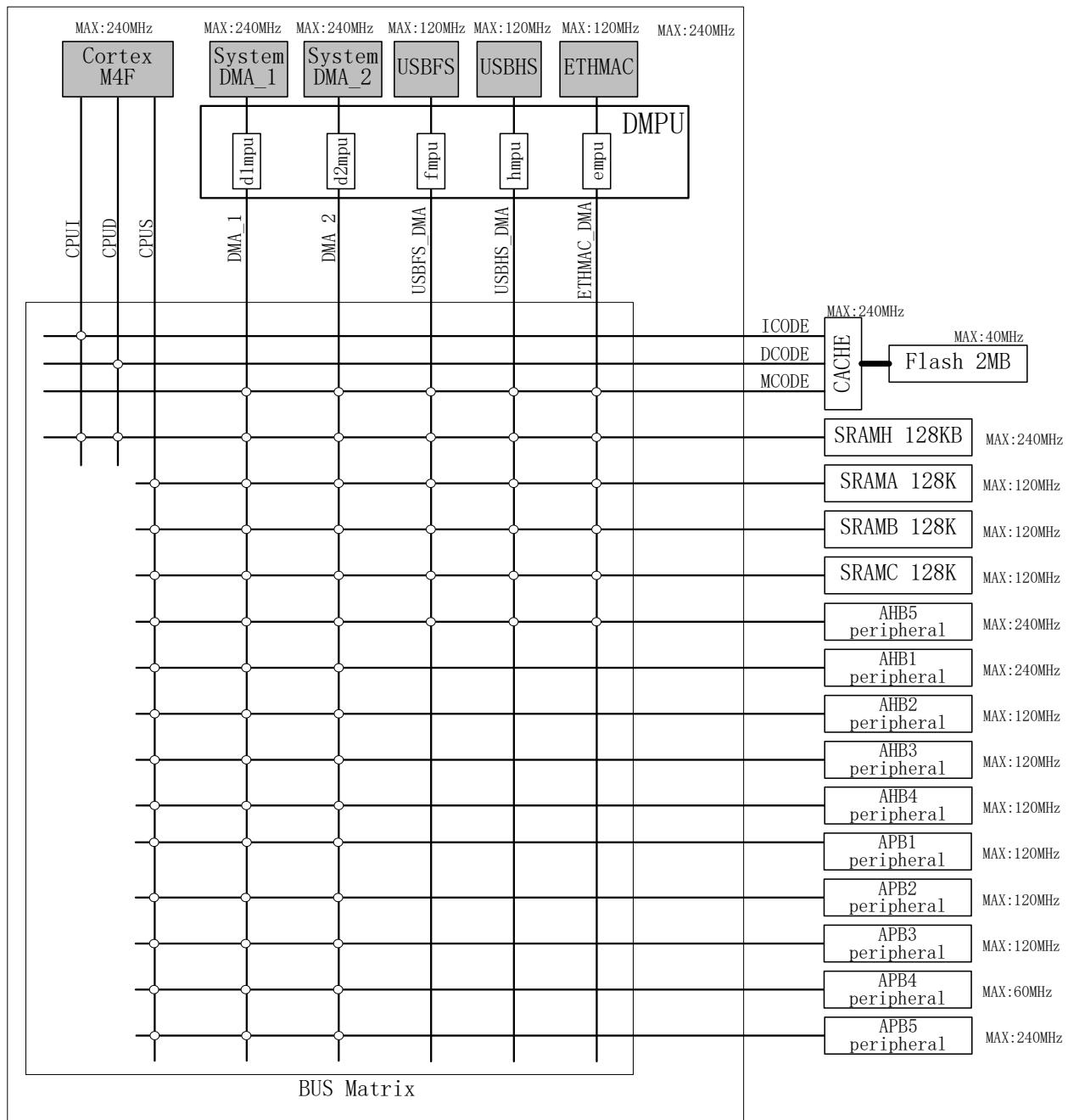


Figure 2-1 Bus Architecture Diagram

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

- CPU-I bus

The instruction bus of the M4F core, through which the CPU fetch instructions. The access objects are Flash and SRAMH that containing code.

- CPU-D bus

The data bus of the M4F core, through which the CPU performs immediate data loading and debug access. The access objects are Flash and SRAMH that containing code or data.

- CPU-S bus

The system bus of the M4F core, through which the CPU accesses peripherals or system SRAM, and can also fetch instructions and immediate data through this bus (the efficiency is lower than through the CPU-I bus and the CPU-D bus). The access objects are SRAM1/SRAM2/SRAM3/SRAM4/SRAMB/all peripherals and external memories.

- DMA_1 bus, DMA_2 bus

System DMA_1/System DMA_2 dedicated bus, DMA_1/DMA_2 access data memory and peripherals through this bus, and the access objects are Flash/SRAMH/SRAM1/SRAM2/SRAM3/SRAM4/SRAMB/all peripherals and external memories.

- USBFS-DMA bus

DMA dedicated bus of USBFS, USBFS accesses all memory space through this bus. The access objects are Flash/SRAMH/SRAM1/SRAM2/SRAM3/SRAM4/SRAMB and external memories.

- USBHS-DMA bus

DMA dedicated bus for USBHS, USBHS accesses all memory space through this bus. The access objects are Flash/SRAMH/SRAM1/SRAM2/SRAM3/SRAM4/SRAMB and external memories.

- ETHMAC-DMA bus

ETHMAC's DMA dedicated bus, through which ETH accesses all memory spaces. The access objects are Flash/SRAMH/SRAM1/SRAM2/SRAM3/SRAM4/SRAMB and external memories.

2.3 Bus function

The bus system is responsible for reading and writing access from the host to the slave. When the operating frequency of the host is higher than the slave (such as CPU-S accessing RTC), the bus will automatically perform sync-down processing. When the operating frequency of the master is lower than the slave (eg USBFS_DMA accesses SRAMH), the bus will automatically perform sync-up processing.

Through the bus matrix, when the access targets of different hosts do not conflict, each access can be carried out simultaneously. For example, CPU-I accesses Flash, CPU-D accesses SRAMH, CPU-S accesses APB peripherals, DMA_1 accesses SRAM1, DMA_2 accesses SRAM2, and USBFS-DMA accesses the external memory of AHB5. These accesses can be performed at the same time.

3 Reset control (RMU)

3.1 Introduction

Support 15 reset source.

- Power-on Reset (POR)
- NRST pin reset (NRST)
- Brown-out Reset (BOR)
- Programmable Voltage Detection 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)
- RAM ECC reset (RAMECCR)
- Clock Frequency Abnormal Reset (CKFER)
- External High-Speed Oscillator Fault Reset (XTALER)
- Cortex-M4 Lockup reset (LKUPR)

3.2 Reset source and reset flag bit

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

Table 3-1 Reset source and generation conditions

Reset source	Generating condition
Power-on reset	VCC power up
NRST pin reset	NRST pin input low level
Brown-out reset	VCC voltage drops below VBOR voltage
Programmable Voltage Detect 1 Reset	VCC voltage drops below PVD1 voltage
Programmable Voltage Detect 2 Reset	VCC voltage drops below PVD2 voltage
Watchdog reset	Refresh error or overflow error occurs in watchdog
Dedicated watchdog reset	Refresh error or overflow error occurred in the dedicated watchdog
Power-down wake-up reset	Power-down wake-up event occurs when device is in power-down mode, the device will wake up from power-down mode after reset
Software reset	Set the reset register bit (ARM register AIRCR.SYSRESETREQ bit)
MPU error reset	Reset generated by MPU access error
RAM parity reset	Parity error occurs in the RAM
RAM ECC error reset	ECC error occurs in the RAM
Clock frequency abnormal reset	Clock frequency abnormal detected by the clock frequency measurement module (FCM)
External High-Speed Oscillator Fault Reset	When external high-speed oscillator oscillation failure is detected
Cortex-M4 Lockup reset (LKUPR)	When the Cortex-M4 encounters a serious exception, it stops its PC pointer at the current address, locks itself, and resets the entire chip after a delay of several clock cycles

When reset occurs, the corresponding reset flag bit is set according to the reset sources. The reset flag bit is shown in Table 3-2. For example, external NRST pin reset occurs, pin reset flag bit RMU_RSTF0.PINRF is set to 1. After RMU_RSTF0.PINRF is set, it can be cleared by writing 1 to RMU_RSTF0.CLRF.

Table 3-2 Reset method and reset flag

Reset flag	Reset mode											
	Cortex-M4 Lockup reset	External High-Speed Oscillator Fault	D o wn a t	Clock frequency abnormal reset	RAM ECC error reset	RAM parity error reset	MPU error reset	Software reset	Power-down wake-up reset	Dedicated watchdog reset	Watchdog reset	Voltage detection 2 reset
Power-on reset flag (RMU_RSTF0.PORF)	√	X	—	—	—	—	—	—	—	—	—	—
Pin reset flag (RMU_RSTF0.PINRF)	X	√	—	—	—	—	—	—	—	—	—	—
Brown-out Reset Flag (RMU_RSTF0.BORF)	X	X	√	—	—	—	—	—	—	—	—	—
Programmable Voltage Detect 1 Reset Flag (RMU_RSTF0.PVD1RF)	X	X	X	√	—	—	—	—	—	—	—	—
Programmable Voltage Detect 2 Reset Flag (RMU_RSTF0.PVD2RF)	X	X	X	—	√	—	—	—	—	—	—	—
Watchdog reset flag (RMU_RSTF0.WDRF)	X	X	X	—	—	√	—	X	—	—	—	—
Dedicated watchdog reset flag (RMU_RSTF0.SWDRF)	X	X	X	—	—	—	√	X	—	—	—	—
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 fault reset (RMU_RSTF0.XTALERF)	X	X	X	—	—	—	—	X	—	—	—	√
Cortex-M4 Lockup reset (RSTF0.CPULKUPRF)	X	X	X	—	—	—	—	X	—	—	—	√

√: Set X: Clear —: Not change

3.3 Reset timing

3.3.1 Power-on reset

The power-on reset is an internal reset caused by the power-on reset circuit. The reset timing is shown in Figure 3-1. A power-on reset occurs when the power is turned on with the NRST pin set high. After the VCC voltage is higher than the power-on reset voltage V_{POR} , internal power supply is turned on after internal Power-on signal release. After a certain period of time (T_{RSTPOR}), the internal reset is released, and the CPU starts to execute the code.

When power-on reset is generated, the reset flag RMU_RSTF0.PORF is set. For details on power-on reset, please refer to [Description of the action of power-on/power-down reset].

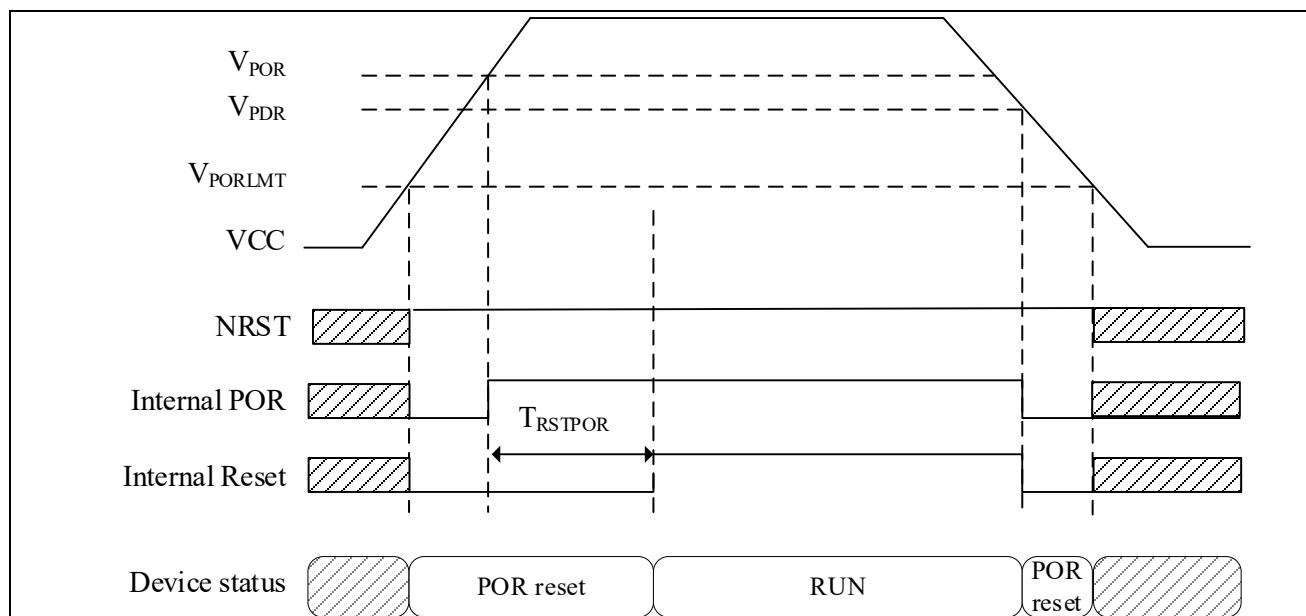


Figure 3-1 Power-on reset

3.3.2 NRST pin reset

The pin reset is a reset caused by the NRST pin being driven low. The reset timing is in Figure 3-2. To generate the pin reset, NRST pin must maintains a low level for a time longer than T_{NRST} , after a certain internal reset time (T_{INRST}), the internal reset is released.

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

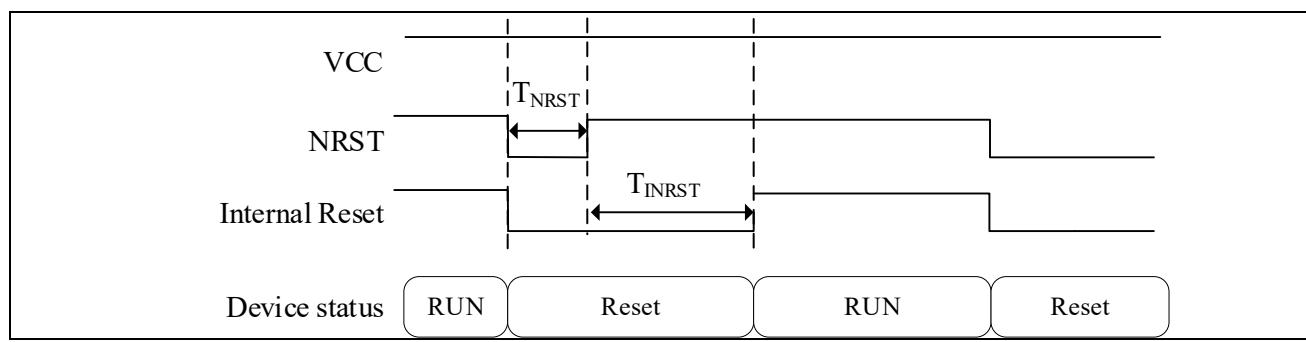
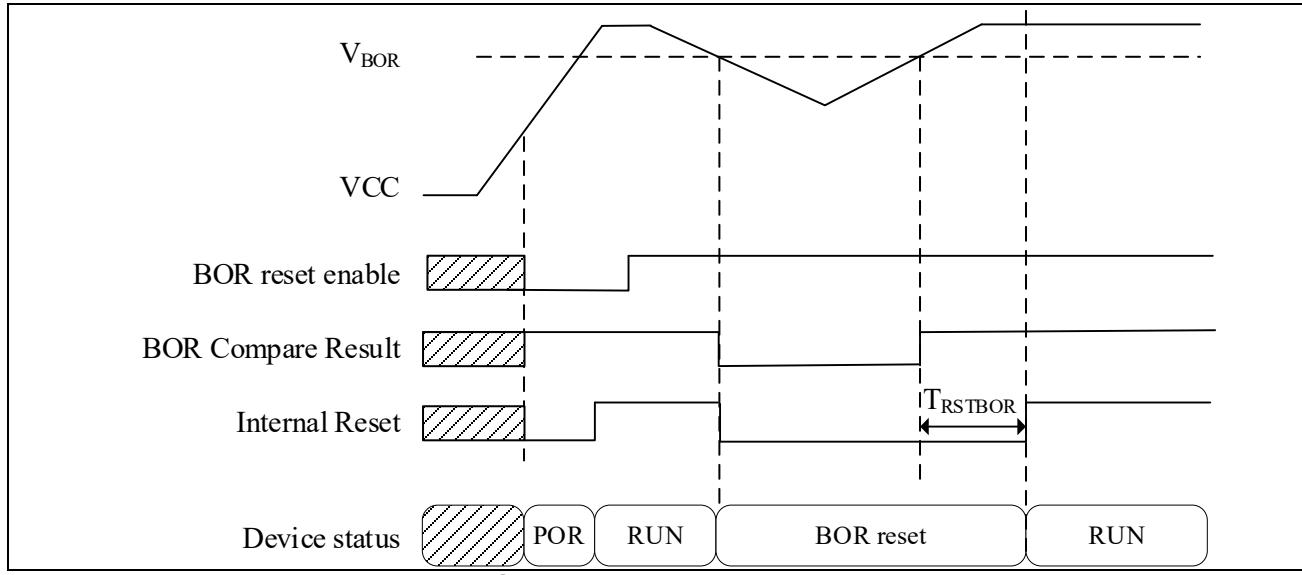


Figure 3-2 NRST reset timing

3.3.3 Brown-out reset

Brown-out reset is an internal reset caused by the voltage monitoring circuit. The reset timing is shown in Figure 3-3. The brown-out reset function can be configured to be enabled by default through the ICG register. RMU_RSTF0.VBORF is set if the VCC voltage is lower than the monitor voltage V_{BOR} . When the VCC voltage is higher than the monitoring voltage V_{BOR} and after a certain internal reset time (T_{RSTBOR}), the reset is released.

For the setting of brown-out reset, please refer to [Description of Brown-out Reset (BOR)].



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

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

After the programmable voltage detection 1 is enabled and configured as reset mode, if VCC is lower than the specified voltage threshold of the programmable voltage detection 1, a programmable voltage detection 1 reset is generated, and RMU_RSTF0.PVD1F is set. When the VCC voltage is higher than the threshold, and after a certain internal reset time (T_{IPVD1}), the reset is released.

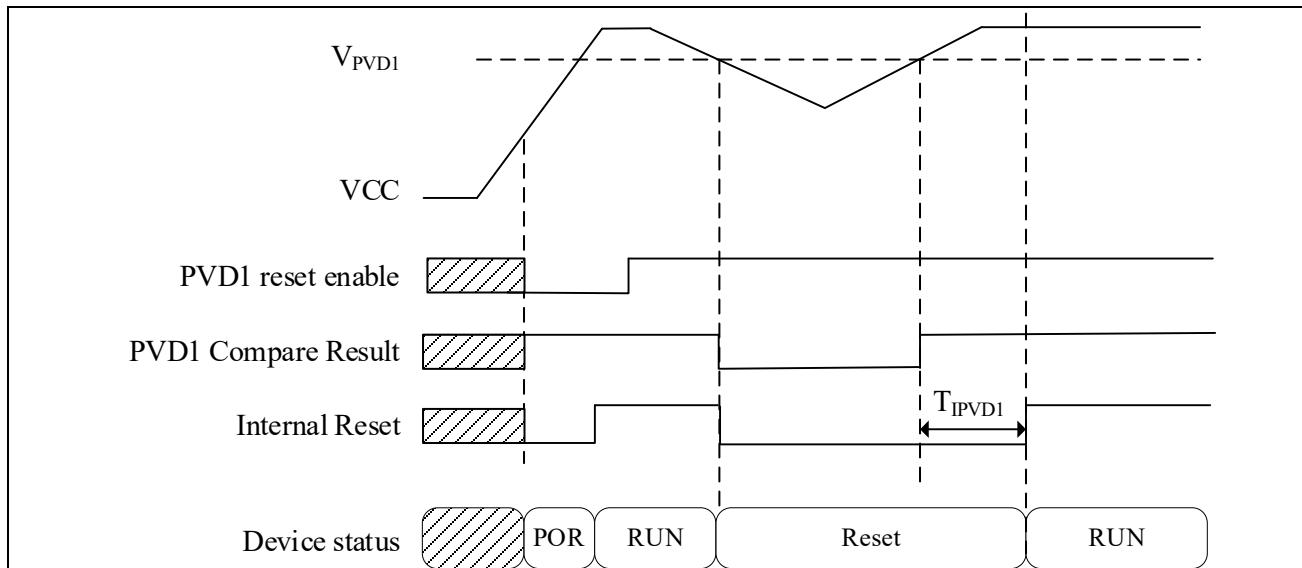


Figure 3-4 Programmable Voltage Detect 1 Reset

After the programmable voltage detection 2 is enabled and configured as reset mode, if VCC is lower than the specified voltage threshold of the programmable voltage detection 2, a programmable voltage detection 2 reset is generated, and RMU_RSTF0.PVD2F is set. When the VCC voltage is higher than the threshold, and after a certain internal reset time (T_{IPVD2}), the reset is released.

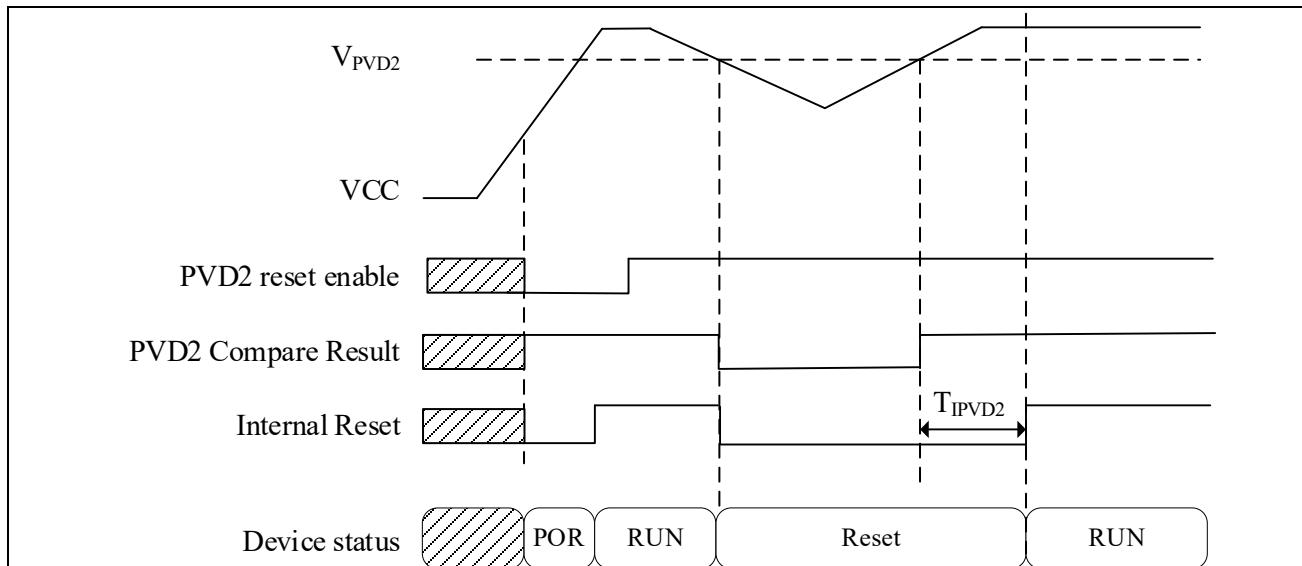


Figure 3-5 Programmable Voltage Detect 2 Reset

For the settings of programmable voltage detection 1 and programmable voltage detection 2 reset, please refer to [Description of Power Supply Voltage Detection Unit (PVD)].

3.3.5 Watchdog reset, dedicated watchdog reset

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

After the watchdog reset is enabled, when the watchdog timer underflows or when a refresh operation is executed outside the allowed refresh period, the watchdog reset is generated and the reset flag RMU_RSTF0.WDRF is set. After a certain internal reset time (T_{RIPT}), the reset is released.

After the dedicated watchdog reset is enabled, when the dedicated watchdog timer underflows or a refresh operation is executed outside the allowed refresh period, the dedicated watchdog reset is generated and the reset flag RMU_RSTF0.SWDRF is set. After a certain internal reset time (T_{RIPT}), the reset is released.

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

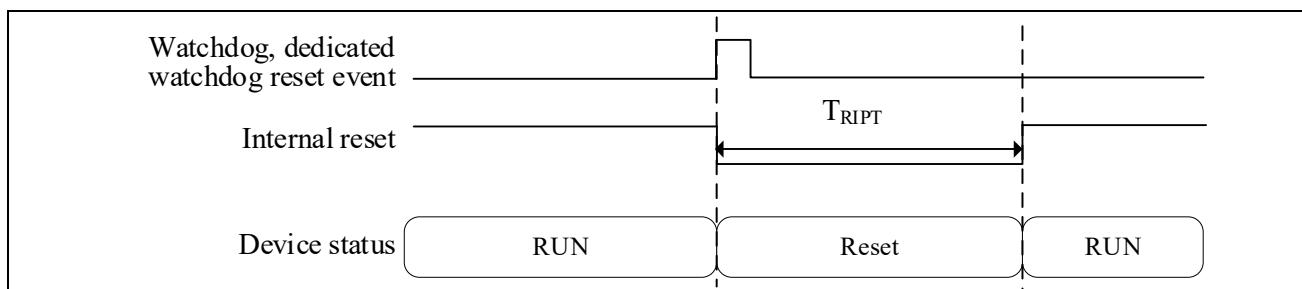


Figure 3-6 Watchdog and Dedicated Watchdog Reset

3.3.6 Power-down wake-up reset

The power-down wake-up reset is the internal reset generated when the power-down wake-up event occurs when the device is in power-down mode. The device can enter power-down mode by executing the WFI command with PWC_PWRCON.PWDN set to 1. The reset timing is shown in Figure 3-7. After the power-down wake-up reset asserts and the return time (T_{IPDX} , $x=1, 2, 3, 4$) has elapsed, the power-down wake-up reset is released. The return time varies according to the specific power-down mode set. It is the smallest in power-down mode 1 and the largest in power-down mode 3.

For details of power-off wake-up reset, please refer to [Power-down mode].

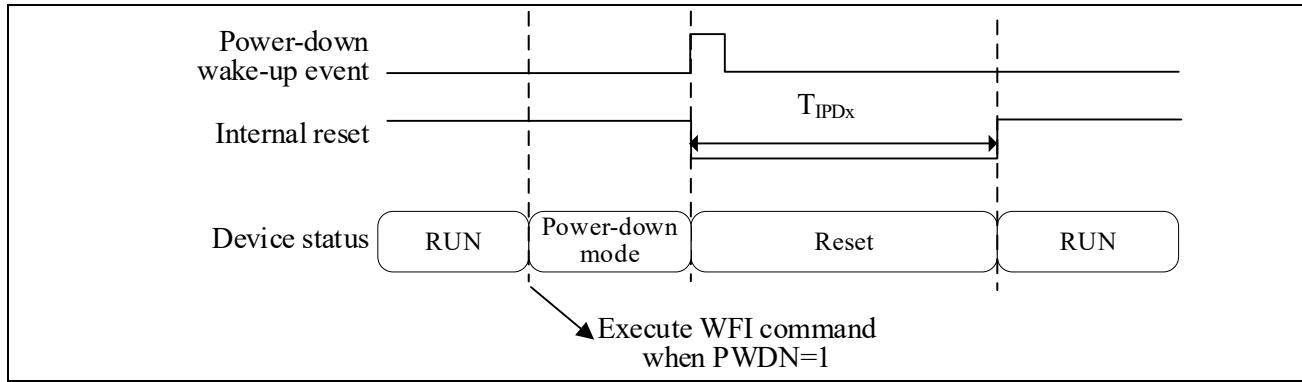


Figure 3-7 Power-down wake-up reset

3.3.7 Software reset

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

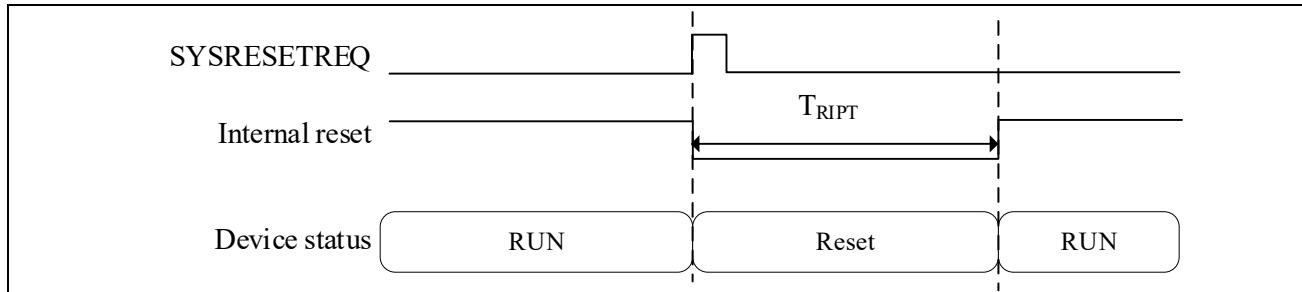


Figure 3-8 Software reset

3.3.8 MPU error reset

When an MPU error occurs, the MPU error reset is generated, the reset flag RMU_RSTF0.MPUERF is set. The reset timing is shown in Figure 3-9. After the internal reset time (T_{RIPT}), the reset is released.

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

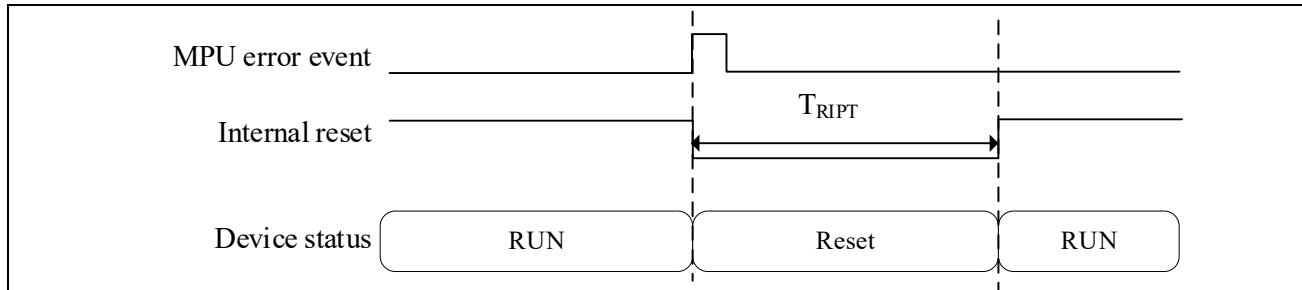


Figure 3-9 MPU error reset

3.3.9 RAM parity reset

When an error occurs in the RAM parity check, the RAM parity reset is generated, and the reset flag RMU_RSTF0.RAPERF is set. The reset timing is shown in Figure 3-10. After the internal reset time (T_{RIPT}), the reset is released.

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

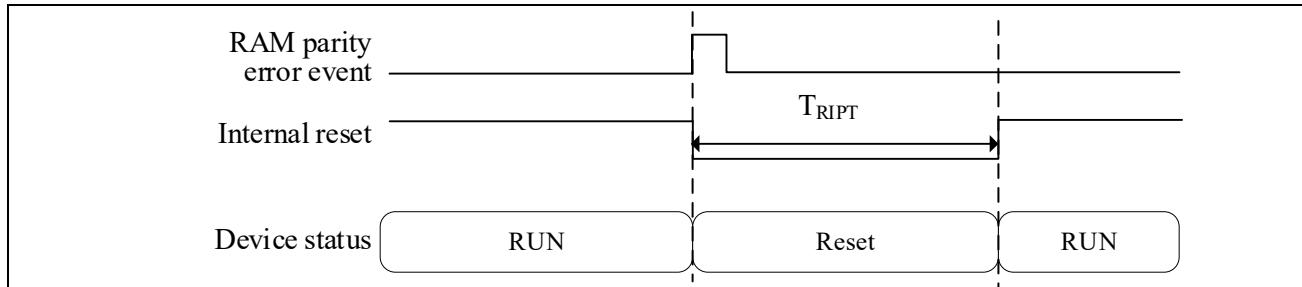


Figure 3-10 RAM parity reset

3.3.10 RAM ECC reset

When an error occurs in the RAM ECC check, the RAM ECC reset is generated, and the reset flag RMU_RSTF0.RAECRF is set. The reset timing is shown in Figure 3-11.. After the internal reset time (T_{RIPT}), the reset is released.

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

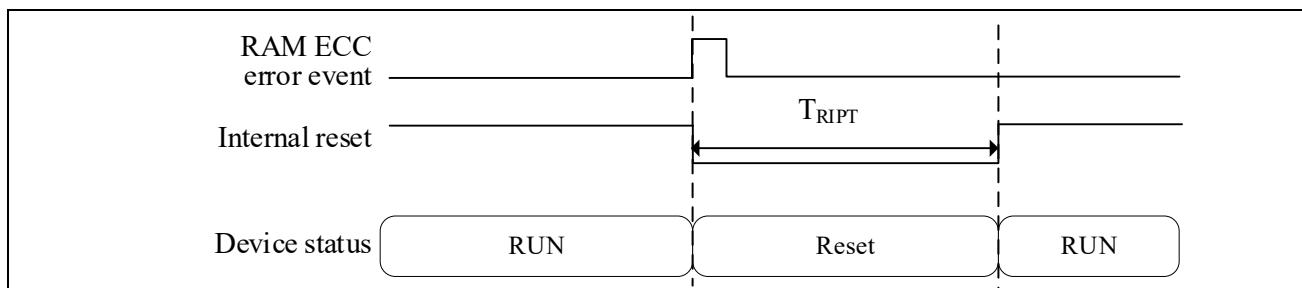


Figure 3-11 RAM ECC reset

3.3.11 Clock frequency abnormal reset

When the built-in FCM module detects that the clock frequency is abnormal, and it is configured as reset source, then a clock frequency abnormal reset is generated. The reset timing is shown in Figure 3-12. When clock frequency abnormal reset asserts, reset flag RMU_RSTF0.CKFERF is set. After the internal reset time (T_{RIPT}), the internal reset is released.

For the setting of clock frequency abnormal reset, please refer to [Clock frequency measurement].

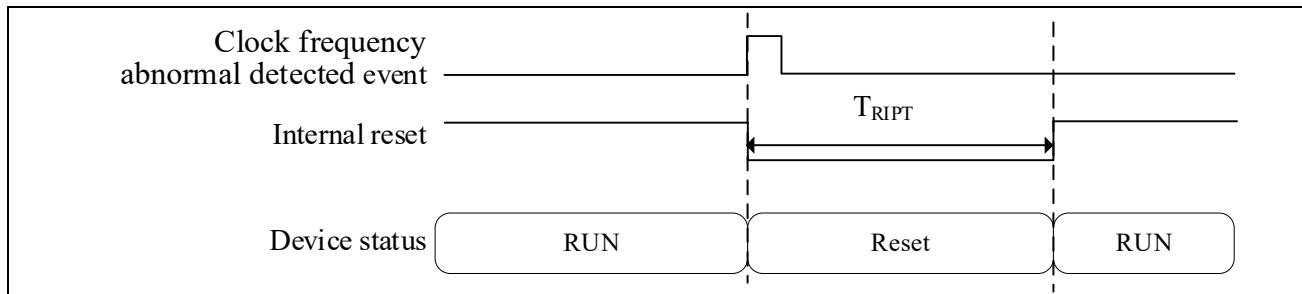


Figure 3-12 Clock frequency abnormal reset

3.3.12 External High-Speed Oscillator Fault Reset

When the external high-speed oscillator fault detection function is enabled and configured as reset mode, if an external high-speed oscillator oscillation failure is detected, the external high-speed oscillator fault reset is generated, and the reset flag RMU_RSFTF0.XTALERF is set. After the internal reset time (T_{RIPT}), the internal reset is released.

For the setting of the external high-speed oscillator fault reset, please refer to [External high-speed oscillator fault detection].

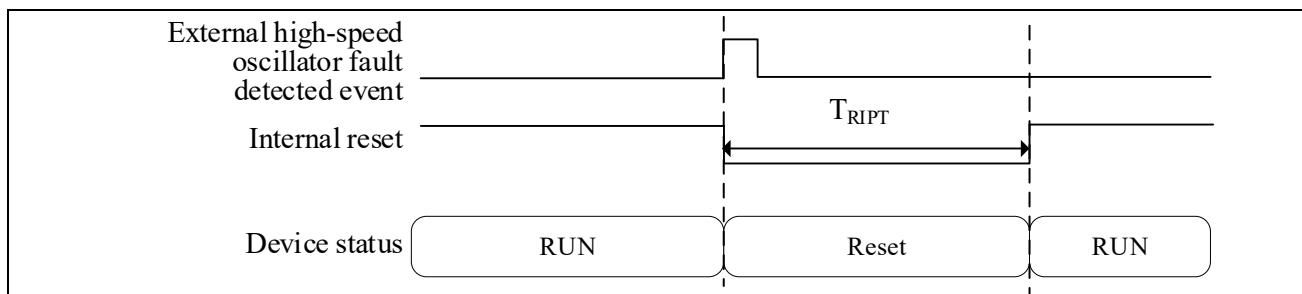


Figure 3-13 External high-speed oscillator fault reset

3.3.13 Cortex-M4 Lockup Reset

When Cortex-M4 encounters a serious exception, it will stop its PC pointer at the current address, lock itself, and reset the entire chip after a delay of several clock cycles, the reset timing is shown in Figure 3-14. When Cortex-M4 Lockup reset is generated, the reset flag RMU_RSTF0.LKUPRF is set. After the internal reset time (T_{RIPT}), the internal reset is released.

The Cortex-M4 Lockup reset is available only when RMU_PRSTCR0.LKUPREN is set to 1.

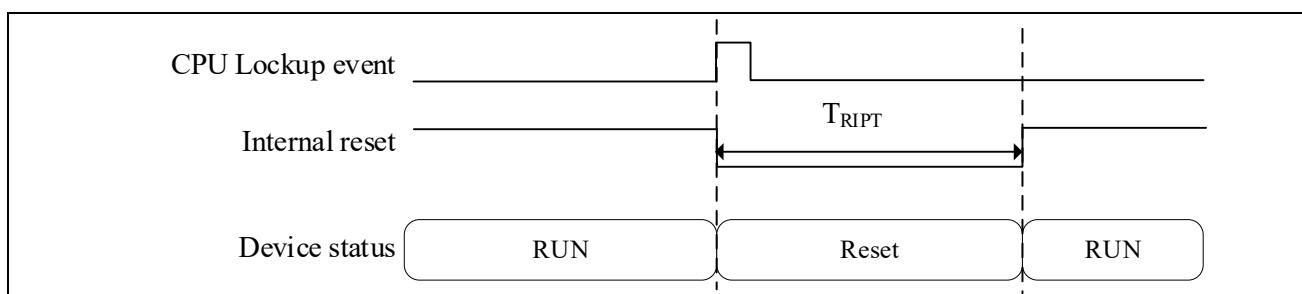


Figure 3-14 M4 Lockup Reset

3.3.14 Identification of the reset cause

The cause of reset can be identified by checking reset flag register RMU_RSTF0. Multiple reset flags may set when two or more reset sources occur simultaneously. When the bit RMU_RSTF0.MULTIRF is 1, it indicates that multiple resets have occurred. All reset flags can be cleared by writing 1 to RMU_RSTF0.CLRF bit after reading RMU_RSTF0. After writing 1 to RMU_RSTF0.CLRF, wait at least 6 CPU clock cycles before reading the RMU_RSTF0 register again.

3.3.15 Reset conditions for individual modules

"All reset sources" in the table refer to the 15 reset sources described in this chapter.

Table 3-3 Reset conditions for each module

Modules	Register	Reset source
Debug controller (DBG)	MCUSTPCTL MCUSTPCTL2 MCUTRACECTL MCUDBGSTAT	1. Power-on reset 2. Power-down wake-up reset
Real Time Clock(RTC)	RTC internal registers	Module software reset control bits: RTC_CRO.RESET
Built-in SRAM (SRAM)	SRAM_CKSR	1. Power-on reset 2. Power-down mode 3 wake-up reset
	Registers other than SRAM_CKSR	All reset sources
Power control (PWC) Clock control (CMU)	PWC_PWRC0 PWC_PWRC1 PWC_PDWKE0 PWC_PDWKE1 PWC_PDWKE2 PWC_PDWKES PWC_PWRC1.DTS CMU_XTALCFGR	All reset sources except the following resets: 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 except power-down wake-up reset
	PWC_PVDLCR PWC_PVDCLR1 PWC_PVDFCR PWC_PVDCR0 Bits other than PWC_PWRC2.DTS PWR_PWRC4	1. Power-on reset 2. NRST pin reset 3. Brown-out reset 4. Watchdog reset 5. Dedicated watchdog reset 6. Power-down mode 3 wake-up reset
	PWC_PVDICR[2:0] PWC_PVDDSR.PVD1DETFLG PWC_PVDICR[6:4] PWC_PVDDSR.PVD2DETFLG	1. Power-on reset 2. NRST pin reset 3. Brown-out reset 4. Watchdog reset 5. Dedicated watchdog reset 6. Power-down wake-up reset
	PWC_VBATCR PWC_BKR00-PWC_BKR127 PWC_WKTC0 PWC_WKTC1 PWC_WKTC2	VBAT domain reset, method: write 0xA5 to PWC.VBTRSTR[7:0]
	Other than the above	All reset sources
	Registers other than the above modules	All reset sources

3.4 Register description

Register list

BASE ADDR: 0x4004CCF0

Table 3-4 List of RMU registers

Register name	Symbol	Offset address	Bit width	Reset value
Reset control register	RMU_PRSTCR0	0x08	8	0x40
Reset flag register 0	RMU_RSTF0	0x0C	32	Different Reset Values According to Different Reset causes

3.4.1 Reset Control Register (RMU_PRSTCR0)

Reset value: 0x40h

B7	b6	b5	b4	b3	b2	b1	b0
-	-	LKUPREN	-	-	-	-	-

Bit	Symbol	Bit name	Description	Read and write
B7	Reserved	-	Read as "0", write as "0"	R/W
b6	Reserved	-	Read as 1, write 1.	R/W
b5	LKUPREN	Lockup reset enable	0: Lockup reset is invalid 1: Lockup reset enable	R/W
b4-b0	Reserved	-	Read as "0", write as "0"	R/W

3.4.2 Reset Flag Register 0 (RMU_RSTF0)

Reset value: 0XXXXh (Depending on the reset cause, the reset value is different)

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CLR F	MULTIR F	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
-	LKUPRF	XTALER F	CKFER F	RAECR F	RAPER F	MPUER F	SWR F	PDR F	SWDR F	WDR F	PVD2R F	PVD1R F	BOR F	PINR F	POR F

Bit	Symbol	Bit name	Description	Read and write
b31	CLRF	Reset flag clear bit	Software write 1 in this bit to clear the reset flag bits. Read as "0". The write action should be performed after reading RMU_RSTF0. 0: No operation 1: Clear all reset flags	R/W
b30	MULTIRF	More than 2 resets Occurrence flag bit	This flag is set when two or more resets occur. Cleared by writing 1 to CLRF. 0: No or only one reset occurred 1: Two or more resets occurred	R/W
b29-b15	Reserved	-	Read as "0", write as "0"	R/W
b14	LKUPRF	Cortex-M4 Lockup Reset	This flag is set when Cortex-M4 Lockup reset occurs. Cleared by writing 1 to CLRF. 0: Cortex-M4 Lockup reset has not occurred 1: Cortex-M4 Lockup reset occurred	R/W
b13	XTALERF	External high-speed oscillator fault reset flag	This flag is set when external high-speed oscillator fault reset occurs. Cleared by writing 1 to CLRF. 0: No external high-speed oscillator fault reset occurred 1: External high-speed oscillator fault reset occurred	R/W
b12	CKFERF	Clock frequency abnormal Reset flag	This flag is set when clock frequency abnormal reset occurs. Cleared by writing 1 to CLRF. 0: No clock frequency abnormal reset occurred 1: Clock frequency abnormal reset occurred	R/W
b11	RAECRF	RAM ECC reset flag	This flag is set when a RAM ECC reset occurs. Cleared by writing 1 to CLRF. 0: RAM ECC reset has not occurred 1: RAM ECC reset occurred	R/W
b10	RAPERF	RAM parity error Reset flag	This flag is set when RAM parity error reset occurs. Cleared by writing 1 to CLRF. 0: No RAM parity error reset occurred 1: RAM parity error reset occurred	R/W
b9	MPUERF	MPU error reset flag	This flag is set when MPU error reset occurs. Cleared by writing 1 to CLRF. 0: MPU error reset has not occurred 1: MPU error reset occurred	R/W
b8	SWRF	Software reset flag	This flag is set when software reset occurs. Cleared by writing 1 to CLRF. 0: No software reset occurred 1: Software reset occurred	R/W
B7	PDRF	Power-down wake-up reset flag	This flag is set when power-down wake-up reset occurs. Cleared by writing 1 to CLRF. 0: No power-down wake-up reset occurred 1: Power-down wake-up reset occurred	R/W
b6	SWDRF	Dedicated watchdog reset flag	This flag is set when dedicated watchdog reset occurs. Cleared by writing 1 to CLRF. 0: No dedicated watchdog reset occurred 1: Dedicated watchdog reset occurred	R/W
b5	WDRF	Watchdog reset flag	This flag is set when watchdog reset occurs. Cleared by writing 1 to CLRF. 0: No watchdog reset occurred 1: Watchdog reset occurred	R/W
b4	PVD2RF	Programmable Voltage Detection 2 reset flag	This flag is set when Programmable Voltage Detect 2 reset occurs. Cleared by writing 1 to CLRF. 0: Programmable voltage detection 2 reset has not occurred 1: Programmable voltage detection 2 reset occurred	R/W
b3	PVD1RF	Programmable Voltage Detection 1 reset flag	This flag is set when a Programmable Voltage Detect 1 reset occurs. Cleared by writing 1 to CLRF.	R/W

			0: Programmable voltage detection 1 reset has not occurred 1: Programmable voltage detection 1 reset occurred	
b2	BORF	Brown-out reset flag	This flag is set when a Brown-out reset occurs. Cleared by writing 1 to CLRF. 0: Brown-out reset has not occurred 1: Brown-out reset occurred	R/W
b1	PINRF	NRST pin reset flag	This flag is set when pin reset occurs. Cleared by writing 1 to CLRF. 0: NRST reset has not occurred 1: NRST reset occurred	R/W
b0	PORF	Power-on reset flag	This flag is set when power-on reset occurs. Cleared by writing 1 to CLRF. 0: Power-on reset has no occurred 1: Power-on reset occurred	R/W

4 Clock controller (CMU)

4.1 Introduction

The clock control unit provides clock functions for a range of frequencies, including: One external high-speed oscillator, one external low-speed oscillator, two PLL clock, one internal high-speed oscillator, one Internal medium speed oscillator, one internal low-speed oscillator, one internal low-speed oscillator for RTC, an SWDT dedicated internal low-speed oscillator, Clock prescaler, clock multiplexer and clock gating circuit.

The clock control unit also provides a clock frequency measurement function. The clock frequency measurement circuit (FCM) monitors and measures the measurement target clock using the measurement reference clock. An interrupt or reset occurs when the setting range is exceeded.

The AHB, APB and Cortex-M4 clocks are all derived from the system clock, and the source of the system clock can be selected from 6 clock sources:

- External high-speed oscillator (XTAL)
- External low speed oscillator (XTAL32)
- PLLH clock circuit (PLLH)
- Internal high-speed oscillator (HRC)
- Internal Medium Speed Oscillator (MRC)
- Internal low-speed oscillator (LRC)

The maximum clock frequency of the system clock can reach 240MHz. SWDT has a dedicated internal low-speed oscillator (SWDTLRC). The real-time clock (RTC) uses the external low-speed oscillator or the internal low-speed oscillator as the clock source. The system clock, PLLH output clock, PLLA output clock could be selected as the source of USB-FS clock with frequency 48MHz.

Each clock could be turn on and off separately when not in use to reduce power consumption.

4.2 System block diagram

4.2.1 System block diagram

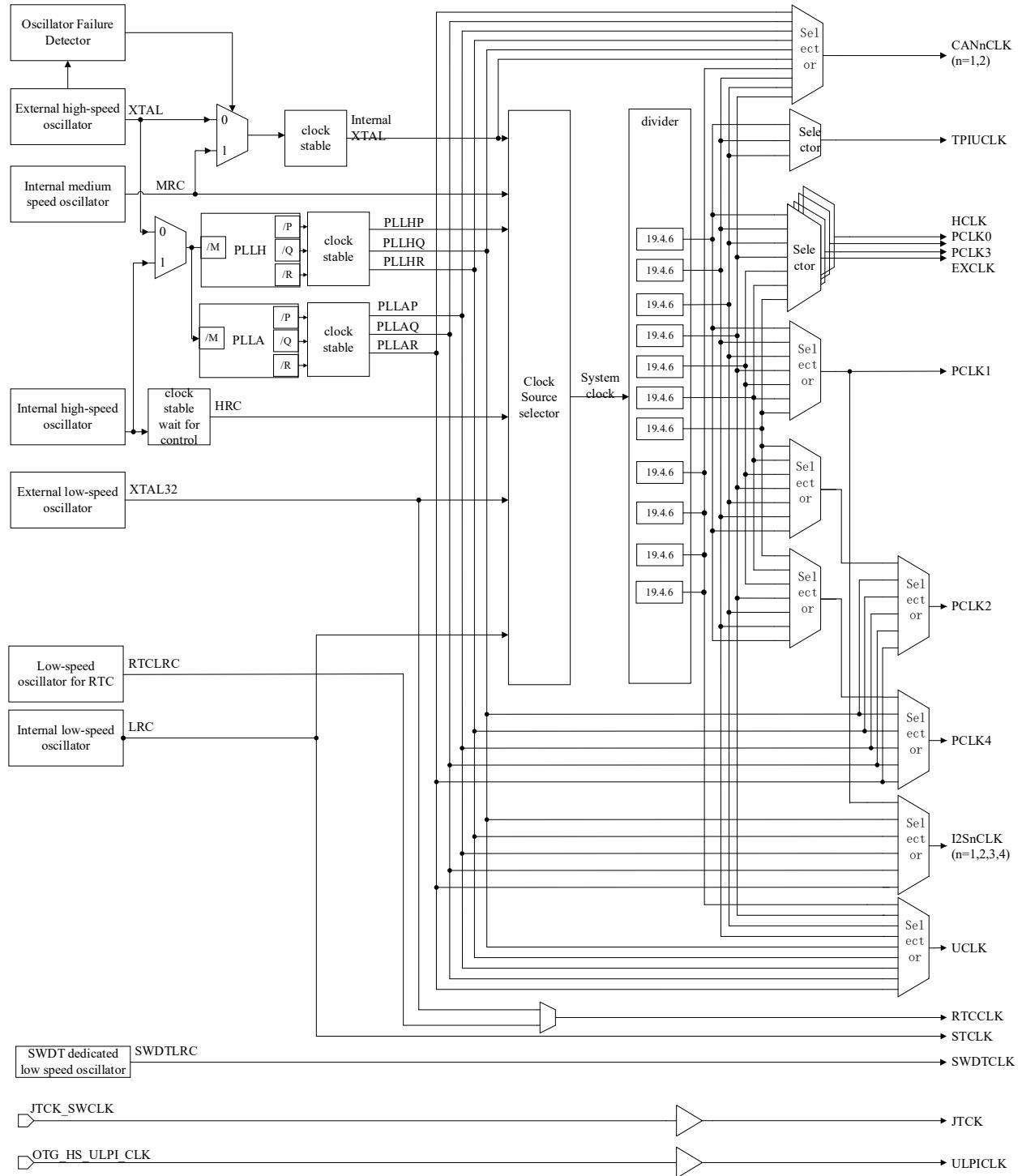


Figure 4-1 Clock System Block Diagram

4.2.2 Clock Frequency Measurement Block Diagram

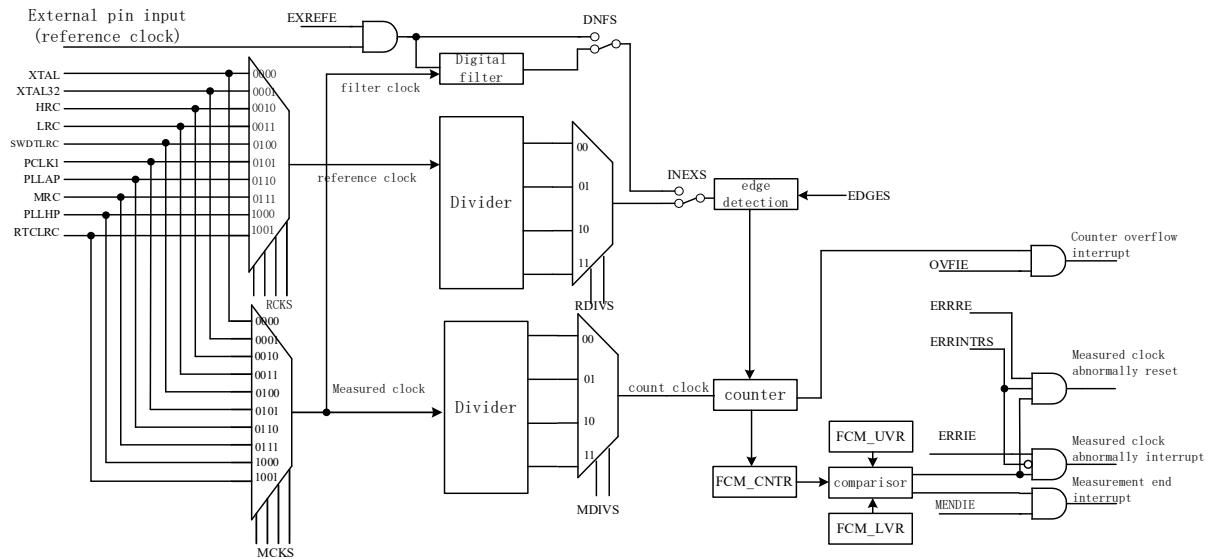


Figure 4-2 Clock Frequency Measurement Block Diagram

4.3 clock source specification

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

Table 4-1 Main characteristics table of clock source

clock source	Specifications
External high-speed oscillator (XTAL)	Crystal frequency range: 4~25MHz External clock input: up to 25MHz Oscillator Fault Detection Function
External low speed oscillator (XTAL32)	Crystal frequency: 32.768kHz
PLLH clock (PLLH)	Input clock: External high-speed oscillator or internal high-speed oscillator PLLH input clock division: 1~4 optional frequency division PFD input clock frequency = input clock/PLLH input clock frequency division, frequency range 8MHz~25MHz PLLH frequency multiplication factor: 25~150 times VCO Oscillation Frequency: 600MHz~1200MHz PLLHQ output divider ratio: 2~16 arbitrary frequency division PLLHP output divider ratio: 2~16 arbitrary frequency division PLLHR output divider ratio: 2~16 arbitrary frequency division PLLHP output frequency = (input clock/PLLH input clock frequency division)*PLLH frequency multiplication factor/PLLHP output frequency division ratio PLLHQ output frequency = (input clock/PLLH input clock frequency division)*PLLH frequency multiplication factor/PLLHQ output frequency division ratio PLLHR output frequency = (input clock/PLLH input clock frequency division)*PLLH frequency multiplication factor/PLLHR output frequency division ratio
PLLA clock (PLLA)	Input clock: External high-speed oscillator or internal high-speed oscillator PLLA input clock division: 1~25 optional frequency division PFD input clock frequency = input clock/PLLA input clock frequency division, frequency range 1MHz~25MHz PLLA frequency multiplication factor: 20~480 times VCO Oscillation Frequency: 240MHz~480MHz PLLAP output divider ratio: 2~16 arbitrary frequency division PLLAQ output divider ratio: 2~16 arbitrary frequency division PLLAR output divider ratio: 2~16 arbitrary frequency division PLLAP output frequency = (input clock/PLLA input clock frequency division)*PLLA frequency multiplication factor/PLLAP output frequency division ratio PLLAQ output frequency = (input clock/PLLA input clock frequency division)*PLLA frequency multiplication factor/PLLAQ output frequency division ratio PLLAR output frequency = (input clock/PLLA input clock frequency division)*PLLA frequency multiplication factor/PLLAR output frequency division ratio
Internal high-speed oscillator (HRC)	Frequency: 16MHz or 20MHz User-Writable Registers for Frequency Trimming
Internal Medium Speed Oscillator (MRC)	Frequency: 8 MHz User-Writable Registers for Frequency Trimming
Internal low-speed oscillator (LRC)	Frequency: 32.768 KHz User-Writable Registers for Frequency Trimming
Internal low-speed oscillator for RTC (RTCLRC)	Frequency: 32.768 KHz User-Writable Registers for Frequency Trimming
SWDT dedicated internal low-speed oscillator (SWDTLRC)	Frequency: 10 KHz

4.4 Working clock specification

Table 4-2 Specification of internal clocks

Clock	Scope	Specification
HCLK	CPU, DMA(n=1, 2), EFM, SRAM(n=1~4), SRAMHS, SRAMB, MPU, GPIO, DCUn(n=1~8), INTC, QSPI, DVP, KEYSAN, FIRn(n=1~4), CORDIC	The maximum frequency is 240MHz. Configured by the HCLKS bits in the CMU_SCFGR register. Optional clock division: 1, 2, 4, 8, 16, 32, 64
PCLK0	Timer6 counter clock, Timer4n(n=1~3), HRPWM, TimerAn(n=1~4)	Maximum frequency 240MHz Configured by the PCLK0S bit in the CMU_SCFGR register. Optional clock division: 1, 2, 4, 8, 16, 32, 64
PCLK1	USBHS/USBFS control logic, USARTn(n=1~10), SPI(n=1~6), Timer0n(n=1, 2), Timer2n(n=1~4), TimerAn(n=5~12), Timer6 (control logic), EMB, CRC, HASH, AES, I2Sn (n=1~4) control logic, SDIOC, ETHMAC, CANn (n=1, 2) control logic	Maximum frequency 120MHz Configured by the PCLK1S bit in the CMU_SCFGR register. Optional clock division: 1, 2, 4, 8, 16, 32, 64
PCLK2	ADC conversion clock	Maximum frequency 60MHz Configured by the PCLK2S bit in the CMU_SCFGR register and the PERICKSEL bit in the CMU_PERICKSEL register. Optional clock division: 1, 2, 4, 8, 16, 32, 64 Optional independent clock source: PLLAP, PLLAQ, PLLAR, PLLHQ, PLLHR
PCLK3	RTC (Control Logic), I2C n(n=1~6), CMPn(n=1, 2), WDT, SWDT (control logic), WKTM, OTS, FCM, VBAT backup register, CTC	Maximum frequency 60MHz Configured by the PCLK3S bit in the CMU_SCFGR register. Optional clock division: 1, 2, 4, 8, 16, 32, 64
PCLK4	ADCn(n=1~3) (control logic), DACn(n=1~2) (control logic), TRNG	Maximum frequency 120MHz Configured by the PCLK4S bit in the CMU_SCFGR register and the PERICKSEL bit in the CMU_PERICKSEL register. Optional clock division: 1, 2, 4, 8, 16, 32, 64 Optional independent clock source: PLLAP, PLLAQ, PLLAR, PLLHQ, PLLHR
EXCLK	NFC, DMC, SMC	Maximum frequency 120MHz Configured by the EXCKS bits in the CMU_SCFGR register. Optional clock division: 1, 2, 4, 8, 16, 32, 64
UCLK	Clock for USBFS/USBHS FullSpeed Communication	Frequency 48MHz Configuration by the USBCKS bits in the CMU_USBCKCFG register. Optional clock division: 2, 3, 4, 5, 6, 7, 8. Optional independent clock source: PLLAP, PLLAQ, PLLAR, PLLHQ, PLLHR
CANnCLK	CAN1/CAN2 communication clock	Maximum frequency 80MHz Configured by the CAN2CKS/CAN1CKS bits in the CMU_CANCKCFG register. Optional clock divider by 2, 3, 4, 5, 6, 7, 8. Optional independent clock source: PLLAP, PLLAQ, PLLAR, PLLHQ, PLLHR, XTAL
STCLK	SYSTICK timer external reference clock	Derived by internal low-speed oscillator clock. SYSTICK timer clock is configured by the CLKSOURCE bit in the CPU's SYSTICK control and status register.
SWDCLK	SWDT Mcounter clock	Frequency 10kHz
TCK	JTAG clock	Maximum frequency 25MHz
TPIUCLK	Clock for Cortex-M4 debug tracer	Maximum frequency 60MHz

Clock	Scope	Specification
		Configured by the TPIUCKS bit in the CMU_TPIUCKCFG register. optional clock frequency division: 1, 2, 4
ULPCLK	Clock for USB-HS HighSpeed Communication	Frequency 60MHz
I2SnCLK (n=1~4)	I2Sn(n=1~4)	Maximum frequency 240MHz Configured by the CMU_I2SCKSEL register. Optional independent clock source: PLLAP, PLLAQ, PLLAR, PLLHQ, PLLHR, PCLK1

Configuration of clock should follow these rules :

- HCLK frequency \geq PCLK2 frequency, PCLK3 frequency and PCLK4 frequency
- In ETHMAC MII mode:
 - PCLK1 frequency $> 2 * (\text{MII_TX_CLK}/\text{MII_RX_CLK} \text{ frequency})$
 - HCLK frequency $>$ PCLK1 frequency
- In ETHMAC RMII mode:
 - PCLK1 frequency $>$ RMII_REF_CLK frequency
 - HCLK frequency $>$ PCLK1 frequency
 - When ETHMAC is not in use: HCLK frequency \geq PCLK1 frequency
- when using SDIOC : HCLK frequency $>$ PCLK1 frequency
 - When SDIOC is not in use: HCLK frequency \geq PCLK1 frequency
- When using NFC: HCLK frequency $>$ PCLK1 frequency
 - When NFC is not in use: HCLK frequency \geq PCLK1 frequency
- PCLK0 frequency \geq PCLK1 frequency, PCLK0 frequency \geq PCLK3 frequency
- The ratio of HCLK frequency / PCLK0 frequency = N/1 or 1/N
- The ratio of PCLK2 frequency / PCK4 frequency=1/8,1/4,1/2,1/1 or 2/1,4/1, 8/1

4.5 Crystal oscillator circuit

4.5.1 External high-speed oscillator

4.5.1.1 Oscillator mode

External high-speed oscillator provides a accurate clock source for the system clock. Frequency range 4~25MHz.

XTAL opens and closes by set the XTALSTP bit of CMU_XTALCR.

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

The electrical characteristics of the crystal oscillator varies with the parasitic capacitance of the crystal oscillator and the installed circuit, so it must be selected carefully with the crystal oscillator manufacturer. Oscillator features are closely associated with the user's circuit board design, the crystal oscillator and load capacitance must be as close to the oscillator pin as possible to minimize output distortion and vibration stabilization time. The load capacitance value must be adjusted appropriately depending on the oscillator selected. It is not permitted to route the signal line near the oscillating circuit, otherwise it may oscillate anomaly due to inductance.

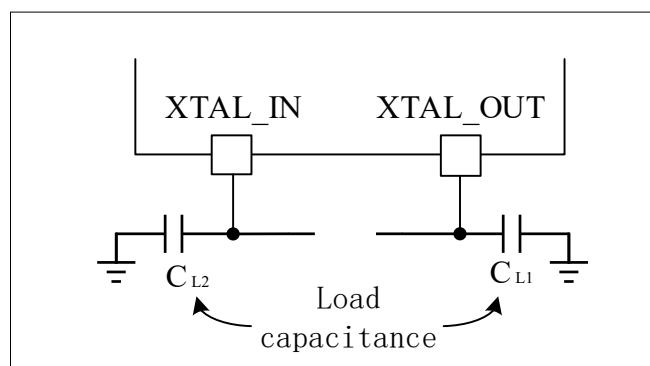


Figure 4-3 External high-speed oscillator connection example

4.5.1.2 Clock input mode

In the external input mode clock must be provided external. This mode is selected by set the XTALMS bit “1” in CMU_XTALCFGR and the XTALSTP bit “0” in CMU_XTALCR. The XTAL_IN pin must be driven with an external clock signal with a duty cycle of approximately 50%. At this time, the XTAL_OUT pin can be configured as a GPIO according to the register setting.

The external clock input is connected as shown in the following figure.

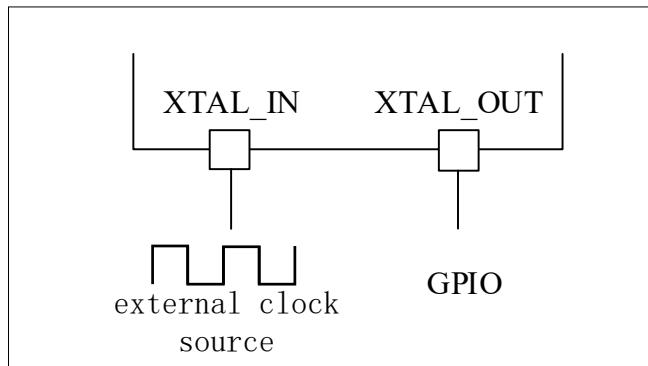


Figure 4-4 Connection example diagram of external clock input

4.5.2 External high-speed oscillator fault detection

The oscillator fault detection is to detect whether the external high-speed oscillator (XTAL) oscillates normally.

Open or close by set XTALSTDE bit of CMU_XTALSTDRCR.

After the reset is removed, the external high-speed oscillator stops oscillating and the external high-speed oscillator fault detection function is invalid. To enable the external high-speed oscillator failure detection function, the external high-speed oscillator must be oscillated, and when the external high-speed oscillator is stable, that is, CMU_OSCSTBSR.XTALSTBF is “1”, it is turned on by set XTALSTDE bit “1” of the register CMU_XTALSTDRCR.

If the XTAL is selected as input clock source of PLLH or PLLA, the XTAL oscillation fault can only be selected to generate a reset but not interrupt.

Since the oscillator fault detection is to detect the oscillator abnormal oscillation caused by external factors, in order to prevent to be triggered by internal switch action, the oscillator fault detection function should be disabled before the XTAL is set to be stopped or the system enter into stop or power-off mode by software.

If the external high-speed oscillator oscillates failure, the waveform is shown in the following figure.
[External high-speed oscillator fault detection example].

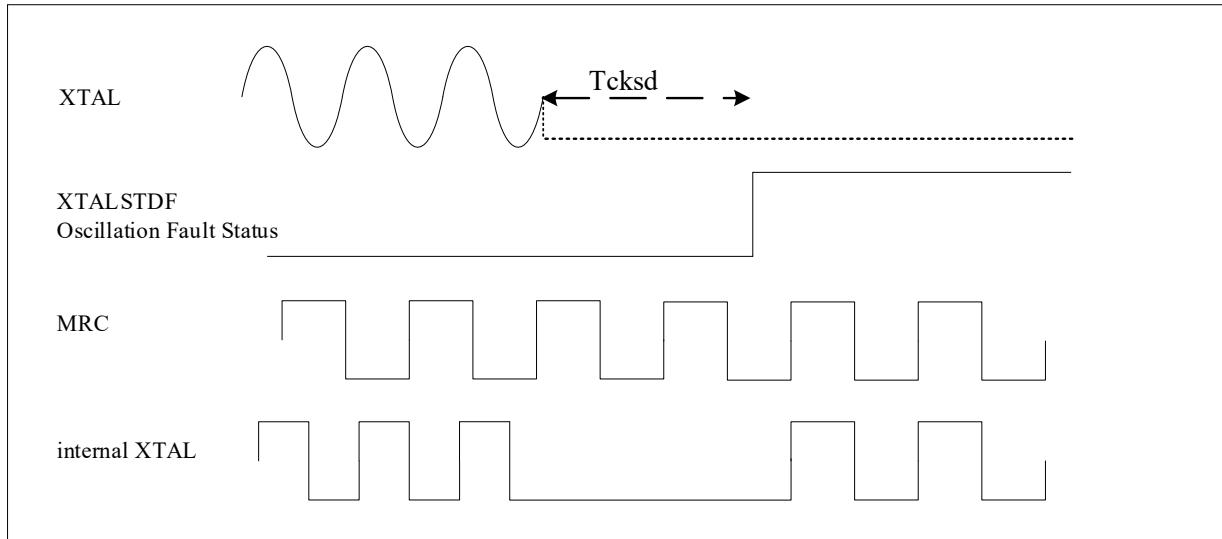


Figure 4-5 External high-speed oscillator fault detection example

4.5.2.1 External high-speed oscillator fault detected

When the external high-speed oscillator oscillation failure is detected, if the system clock selects the external high-speed oscillator as the system clock, the system clock will automatically switch to MRC.

When an external high-speed oscillator oscillation failure is detected, it can trigger EMB and set the PWM output of Timer6/Timer4 to Hiz output. Please refer to [Emergency Brake Module (EMB)] chapter.

If the XTAL is selected as system clock. When an XTAL fault is detected, the action is shown in the following flowchart.

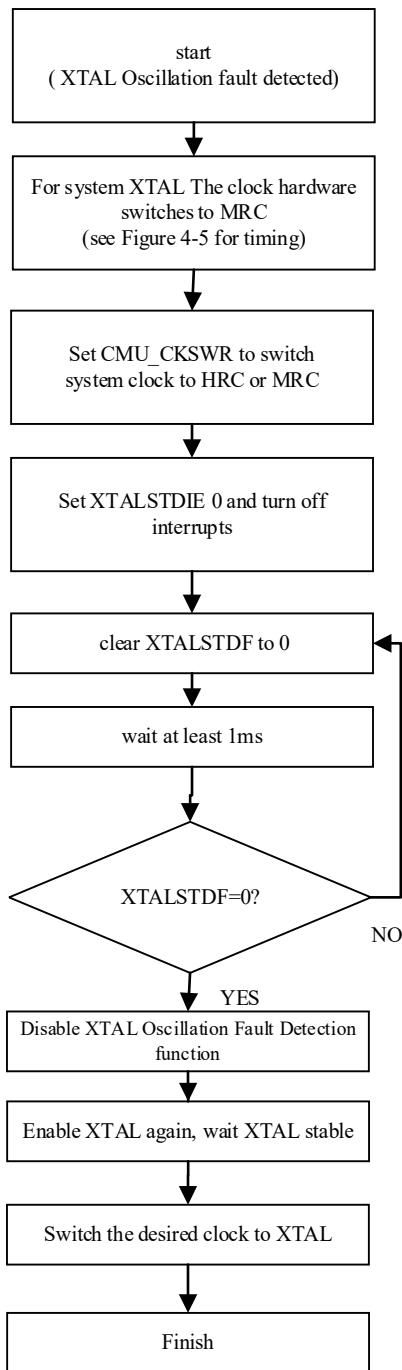


Figure 4-6 XTAL is selected as the system clock, and XTAL oscillation fault is detected

4.5.2.2 XTAL oscillation fault interrupt and reset

XTAL Oscillation Fault Interrupt can be configured to be masked or non-masked interrupt, refer to Chapter. [Interrupt controller (INTC)].

If the XTAL oscillation fault is configured as reset, when the XTAL oscillation fault is detected, the chip is reset. Refer to [Reset control (RMU)].

4.5.3 External low-speed oscillator

The 32.768kHz external low-speed oscillator can provide a precise clock source for the system clock and real-time clock circuit (RTC). It has the advantages of low power consumption and high precision.

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

The electrical characteristics of the crystal oscillator varies with the parasitic capacitance of the crystal oscillator and the installed circuit, so it must be selected carefully with the crystal oscillator manufacturer. Oscillator features are closely associated with the user's circuit board design, and crystal oscillator and load capacitance must be as close to the oscillator pin as possible to minimize output distortion and vibration stabilization time. The load capacitance value must be properly adjusted according to the selected drive capability. It is not permitted to route the signal line near the oscillating circuit, otherwise it may oscillate anomaly due to inductance.

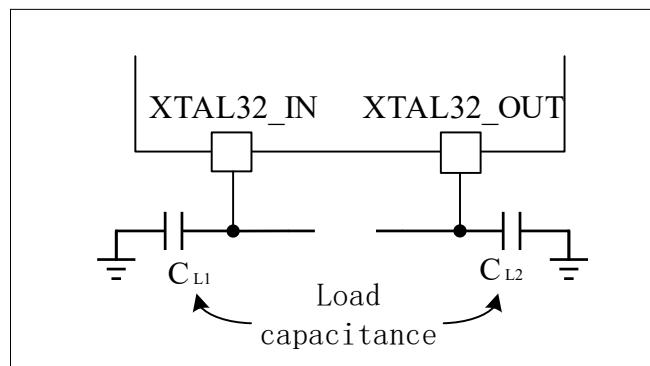


Figure 4-7 External low-speed oscillator connection example

After power-on the initialization process of XTAL32 for the first time is shown as:

1. Write 1 to CMU_XTAL32CR.XTAL32STP bit to stop XTAL32
2. Initialize and set VBATRSTR register
3. Set the applicable XTAL32 drive capability through CMU_XTAL32FGR
4. Set the filter function through CMU_XTAL32NFR
5. Write 0 to CMU_XTAL32CR.XTAL32STP bit and enable XTAL32 to oscillate
6. The software waits for the XTAL32 to stabilize. Please refer to the chapter on electrical characteristics for the stabilization time.

If the external low-speed oscillator is not used, set the XTAL32STP bit of CMU_XTAL32CR to 1 to disable 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 the system clock, or as a PLLH/PLLA input reference clock. The frequency of HRC can be configured to 16MHz or 20MHz by set ICG1.HRCFREQSEL.

HRC oscillator has the advantage of lowcost (no external components needed). In addition, the start-up speed is higher than that of the XTAL crystal, but the accuracy is lower than that of the external crystal even after calibration.

Frequency calibration

Because the production process is different, the RC oscillator frequency of different chips is also different, so each device will be factory calibrated, to ensure the accuracy, please refer to the **internal high-speed (HRC) oscillator chapter in the electrical characteristics of the datasheet**.

If the application is subject to temperature changes, this may also affect the frequency of the RC oscillator. The user can fine-tune the HRC frequency through registers.

The HRCSTBF flag in CMU_OSCSTBSR indicates the stability of HRC. The HRC is not available until the hardware set the HRCSTBF flag “1” during startup.

HRC can be open or closed by set HRCSTP bits in register CMU_HRCCR.

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 the faster start-up.

Frequency calibration

Because the production process is different, the RC oscillator frequency of different chips is also different, so each device will be factory calibrated, to ensure the accuracy, please refer to the **internal medium speed (MRC) oscillator chapter in the electrical characteristics of the datasheet**.

If the application is subject to temperature changes, this may also affect the frequency of the RC oscillator. The user can fine-tune the MRC frequency through registers.

MRC can be open or closed by set the MRCSTP bit in CMU_MRCCR register.

The MRC clock can also be used as a backup clock source in case the XTAL crystal failure. See [External high-speed oscillator fault detection].

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 the system clock. The LRC can be kept running in power-down and stop modes as a low-power clock source for Timer0/KEYSCAN.

The start-up of the LRC oscillator is fast.

Frequency calibration

Because the production process is different, the RC oscillator frequency of different chips is also different, so each device will be factory calibrated, to ensure the accuracy, refer to the **internal low-speed (LRC) oscillator chapter in the electrical characteristics of the data sheet**.

If the application is affected by a voltage or temperature change, this may also affect the frequency of RC oscillator. The user can fine-tune the LRC frequency through registers.

The LRC can be open or closed by set the LRCSTP bits in register CMU_LRCCR.

4.6.4 SWDTLRC clock

The SWDTLRC clock signal is generated by the internal 10kHz low-speed oscillator, the SWDT dedicated clock. If SWDT has been started by ICG settings, then the dedicated internal low-speed oscillator for SWDT will be forced to open and cannot be disabled.

Because the production process is different, the RC oscillator frequency of different chips is also different, so each device will be factory calibrated to ensure the accuracy, refer to the **SWDT dedicated internal low-speed (SWDTLRC) oscillator chapter in the electrical characteristics of the datasheet**.

4.6.5 RTCLRC clock

The RTCLRC clock signal is generated by the internal 32.768kHz low speed oscillator. RTCLRC can be kept running in power-down and stop modes as a low-power clock source for RTC/WKTM.

The RTCLRC oscillator starts fast.

Frequency calibration

Because the production process is different, the RC oscillator frequency of different chips is also different, so each device will be factory calibrated to ensure the accuracy, refer to the **RTC dedicated internal low-speed (RTCLRC) oscillator chapter in the electrical characteristics of the datasheet**.

If the application is affected by a voltage or temperature change, this may also affect the frequency of RC oscillator. The user can fine-tune the RTCLRC frequency through registers.

RTCLRC can be open and closed by set the LRCEN bit in the RTC_CR3 control register.

4.7 PLL clock

HC32A4Axx devices have two PLLs:

- The PLLH with its input clock could be XTAL or HRC oscillator has three different output clocks:
 - The P divider output is used to generate the system clock (up to 240 MHz)
 - All three outputs can be used to generate USB, TRNG, ADC, I2S, CAN clocks.
- The three outputs of the PLLA can also be used to generate USB, TRNG, ADC, I2S, CAN clocks.
PLLA uses the same input clock source as PLLH, and can select HRC or XTAL oscillator as the clock source, which is configured by the CMU_PLLHCFGR.PLLSRC bit. Configures the PLL after the HRC or XTAL oscillator has stabilized.

The frequency division coefficients M, N, P, Q, and R of PLLH/PLLA can be configured independently. Since the PLL configuration cannot be changed after the PLL is enabled, this requires to configure the PLL before enabling it.

Both of two PLLs will be disabled by hardware when entering power-down or stop modes.

4.8 Clock switching step

After system reset, the default system clock is MRC. refers to the switching procedure to switch clock source by setting register CMU_CKSW. Only after the target clock is stable, the source of system clock could switch from the current clock to the target clock.

The waiting period of Flash/SRAM should be configured correctly to prevent the system clock frequency from being larger than the maximum operating frequency of Flash/SRAM. Refer to [Relationship between CPU clock and FLASH read], [Built-in SRAM (SRAM)] section.

4.8.1 Clock switch

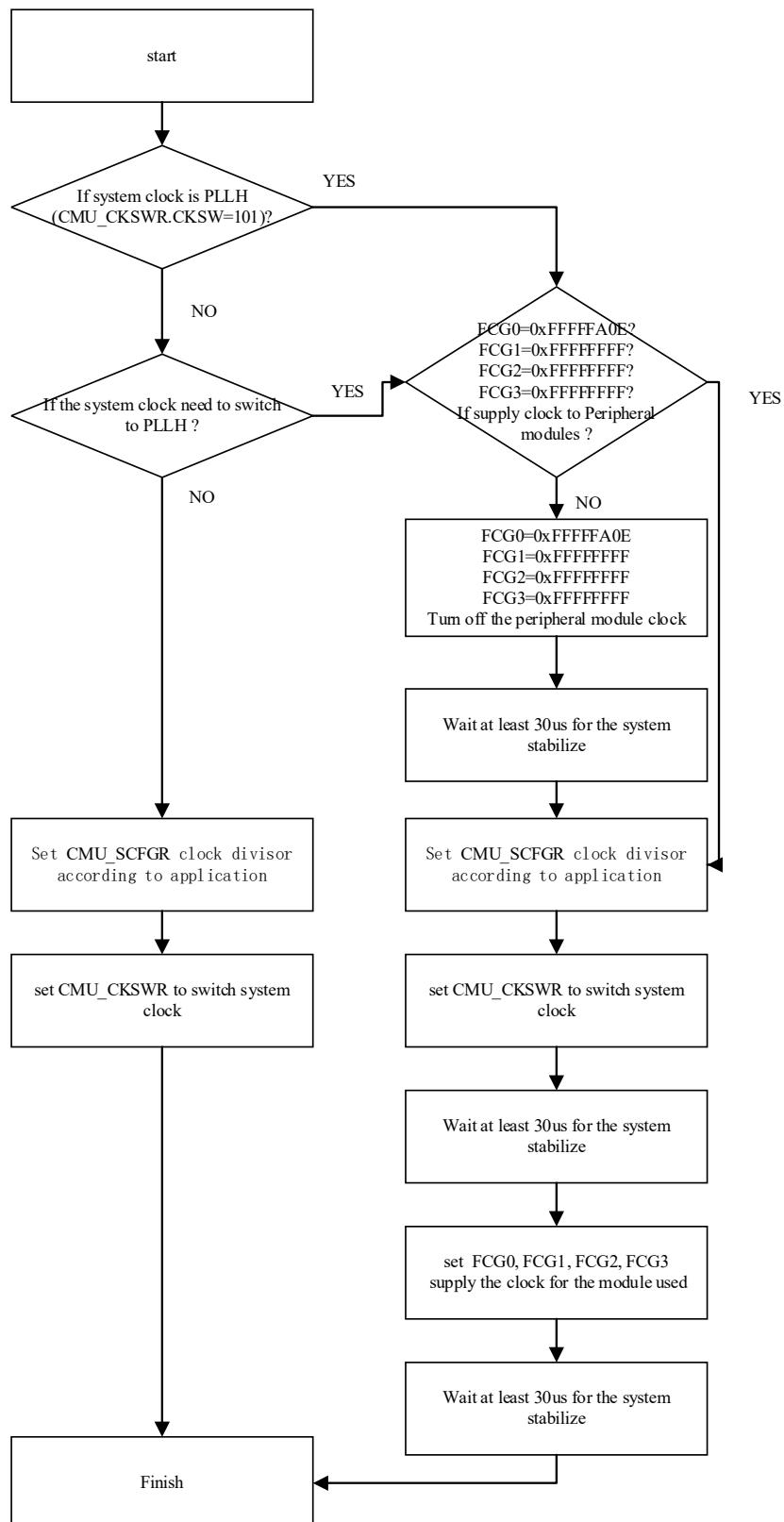


Figure 4-8 Clock source switching

4.8.2 Clock division switching

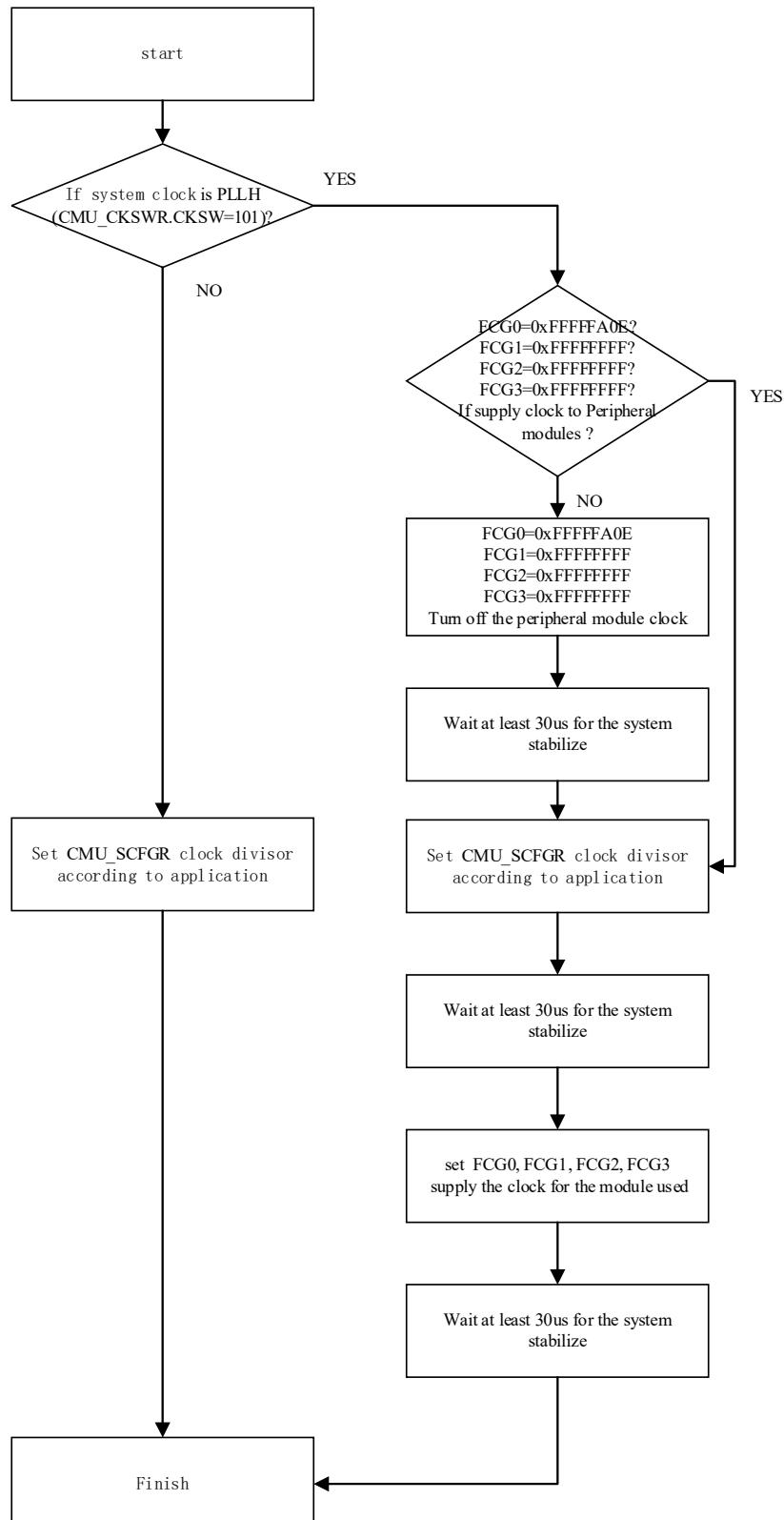


Figure 4-9 Clock division switching

4.9 Clock output function

There are two clock outputs:

■ MCO_1

User can output different clock sources to the MCO_1 pin through a configurable pre-scaler (from 1 to 128):

- HRC clock
- MRC clock
- LRC clock
- XTAL clock
- XTAL32 clock
- PLLHP/PLLHQ clock
- PLLAP/PLLAQ/PLLAR clock
- System clock

The desired clock source is selected by the CMU_MCO1CFG.R.MCO1SEL bits.

■ MCO_2

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

- HRC clock
- MRC clock
- LRC clock
- XTAL clock
- XTAL32 clock
- PLLHP/PLLHQ clock
- PLLAP/PLLAQ/PLLAR clock
- System clock

The desired clock source is selected by the CMU_MCO2CFG.R.MCO2SEL bits.

The MCO_1/MCO_2 output clock must not exceed 100 MHz (maximum I/O speed).

4.10 Clock frequency measurement (FCM)

4.10.1 Clock frequency measurement

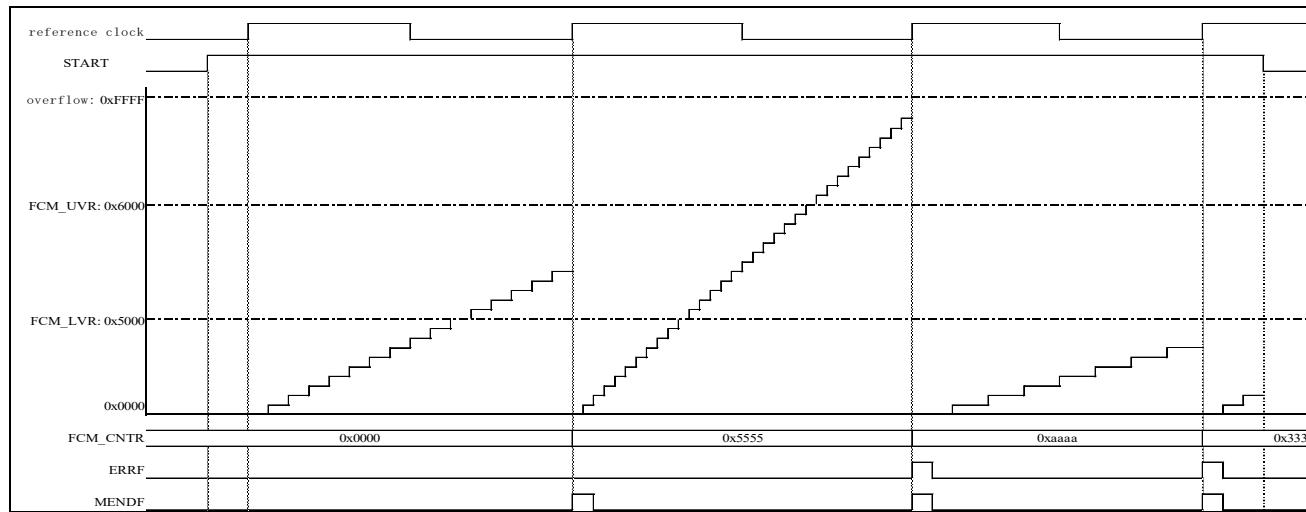


Figure 4-10 Clock Frequency Measurement Timing Diagram

1. Set the registers FCM_MCCR/FCM_RCCR to select the reference clock and measured clock, the frequency division of the clock, and the valid edge of the reference clock.
2. After the START bit of FCM_STR is set to 1, if the valid edge which selected by the EDGES bit is detected, the counter starts to count up.
3. When the next valid edge selected by the EDGES bit of the reference clock is detected, the value of the counter is saved to the FCM_CNTR register and compared with the setting of FCM_LVR/FCM_UVR. When $FCM_LVR \leq FCM_CNTR \leq FCM_UVR$, the frequency of measured clock is normal. When $FCM_LVR > FCM_CNTR$ or $FCM_CNTR > FCM_UVR$, the frequency of measured clock is abnormal, and interrupt or reset occurs according to ERRINTRS/ERRRE/ERRIE settings.
4. After writing 0 to the START bit of FCM_STR, the counter stops counting and count value is cleared.

4.10.2 Digital filter function

The external pin input reference clock FCMREF has digital filter function. The digital filter function performs three samplings according to the sampling clock selected by the DNFS bit. When the signal level of the three samplings is the same, this level is sent to the internal circuit.

The digital filter function can be open and closed by setting register and the sampling clock is configurable.

4.10.3 Interrupt/Reset Function

The clock frequency measurement circuit has three interrupts. They are:

- Frequency abnormal interrupt
- Frequency measure end interrupt
- Counter overflow interrupt

The clock frequency measurement circuit has a reset request:

- Frequency abnormal reset

4.11 Register description

Base address 1: 0x40048400

Register name	Symbol	Offset address	Bit width	Reset value
FCM Lower Limit Compare Value Register	FCM_LVR	0x00	32	0x00000000
FCM Upper Limit Compare Value Register	FCM_UVR	0x04	32	0x00000000
FCM Counter Value Register	FCM_CNTR	0x08	32	0x00000000
FCM Start and Stop Register	FCM_STR	0x0C	32	0x00000000
FCM Measure Object Control Register	FCM_MCCR	0x10	32	0x00000000
FCM Measure Reference Control Register	FCM_RCCR	0x14	32	0x00000000
FCM Interrupt and Reset Control Register	FCM_RIER	0x18	32	0x00000000
FCM flag register	FCM_SR	0x1C	32	0x00000000
FCM flag clear register	FCM_CLR	0x20	32	0x00000000

Base address 2: 0x4004C400

CMU_XTAL32 Control Register	CMU_XTAL32CR	0x000	8	0x00
CMU_XTAL32 Configuration Register	CMU_XTALC32CFGR	0x004	8	0XX
CMU_XTAL32 filter register	CMU_XTAL32NFR	0x014	8	0XX
CMU_LRC control register	CMU_LRCCR	0x01C	8	0x00
CMU_LRCTR Calibration Register	CMU_LRCTR	0x024	8	0x00
CMU_RTCLRCTR Calibration Register	CMU_RTCLRCTR	0x02C	8	0x00

Base address 3: 0x4004CC00

CMU_XTAL configuration register	CMU_XTALCFGR	0x78	8	0x80
---------------------------------	--------------	------	---	------

Base address 4: 0x40054000

Register name	Symbol	Offset address	Bit width	Reset value
CMU_XTAL Stable Configuration Register	CMU_XTALSTBCR	0x0A2	8	0x05
CMU_XTAL control register	CMU_XTALCR	0x032	8	0x01
CMU_XTAL Oscillatory Fault Control Register	CMU_XTALSTDRCR	0x040	8	0x00
CMU_XTAL Oscillatory Fault States	CMU_XTALSTDTSR	0x041	8	0x00
CMU_HRC control register	CMU_HRCCR	0x036	8	Depend on ICG1.HRCSTP value decision
CMU_HRC Calibration Register	CMU_HRCTRM	0x062	8	0x00
CMU_MRC Control Register	CMU_MRCCR	0x038	8	0x80
CMU_MRC Calibration Register	CMU_MRCTRM	0x061	8	0x00
CMU_PLLH Configuration Register	CMU_PLLHCFGR	0x100	32	0x11101300
CMU_PLLH Control Register	CMU_PLLHCR	0x02A	8	0x01
CMU_PLLA Configuration Register	CMU_PLLACFGR	0x104	32	0x11101300
CMU_PLLA Control Register	CMU_PLLACR	0x02E	8	0x01
CMU_Clock Source Stable Status Register	CMU_OSCSTBSR	0x03C	8	0x00
CMU_System clock switch register	CMU_CKSWR	0x026	8	0x01
CMU_Clock Divide Configuration Register	CMU_SCFGR	0x020	32	0x00000000
CMU_USB Clock Configuration Register	CMU_USBCKCFGR	0x024	8	0x40
CMU_CAN Clock Configuration Register	CMU_CANCKCFGR	0x018	8	0xdd
CMU_AD/TRNG Clock Configuration Register	CMU_PERICKSEL	0x010	16	0x0000
CMU_DEBUG CLOCK CONFIGURATION REGISTER	CMU_TPIUCKCFGR	0x03F	8	0x01
CMU_I2S clock configuration register	CMU_I2SCKSEL	0x012	16	0xbbbb
CMU_MCO1 clock output configuration register	CMU_MCO1CFGR	0x03D	8	0x00
CMU_MCO2 Clock Output Configuration Register	CMU_MCO2CFGR	0x03E	8	0x00

4.11.1 CMU XTAL Configuration Register (CMU_XTALCFGR)

Reset value: 0x80

B7	b6	b5	b4	b3	b2	b1	b0
-	XTALMS	XTALDRV[1:0]		-	-	-	-
<hr/>							
Bit	Marking	Place name		Function			Read and write
B7	Reserved	-		Write as "1"			W
b6	XTALMS	XTAL mode selection bit		0: Oscillator mode 1: External clock input mode			R/W
b5~b4	XTALDRV[1:0]	XTAL Driving Ability Selection		00: High drive capability (20~25MHz crystal oscillator is recommended) 01: Medium drive capability (16~20MHz crystal oscillator is recommended) 10: Small drive capability (8~16MHz crystal oscillator is recommended) 11: Ultra-small drive capability (4~8MHz crystal oscillator is recommended)			R/W
b3~b0	Reserved	-		Read as "0", write as "0"			R/W

4.11.2 CMU XTAL Configuration Register (CMU_XTALSTBCR)

Reset value: 0x05

B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	XTALSTB[3:0]			
<hr/>							
Bit	Marking	Place name		Function			Read and write
b7~b4	Reserved	-		Read as "0", write as "0"			R/W
b3~b0	XTALSTB[3:0]	XTAL Stability Time Selection		0001: Stable counter for 35 cycles 0010: Stable counter 67 cycles 0011: Stable counter 131 cycles 0100: Stable counter 259 cycles 0101: Stable counter 547 cycles 0110: Stable counter 1059 cycles 0111: Stable counter 2147 cycles 1000: Stable counter 4291 cycles 1001: Stable counter 8163 cycles One count cycle of the stable counter = LRC cycle / 8 This register is configured with CMU_XTALCR.XTALSTP bit is 1 and CMU_OSCSTBSR.XTALSTBF bit is 0.			R/W

4.11.3 CMU XTAL Control Register (CMU_XTALCR)

Reset value: 0x01

B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	XTALSTP

Bit	Marking	Place name	Function	Read and write
b7~b1	Reserved	-	Read as "0", write as "0"	R/W
b0	XTALSTP	XTAL oscillator open stop bit	0: XTAL oscillator starts 1: XTAL oscillator stops	R/W

Note:

- When XTAL is selected as the system clock or PLLH/PLLA clock source, prohibit writing "1" to XTALSTP to stop the XTAL oscillator.
- Once the software sets the XTAL oscillator to oscillate, only after confirming that the XTAL oscillator is stable through the XTALSTBF bit, it can enter the stop mode, power-down mode, or software set the XTAL oscillator to stop.
- Once the software sets the XTAL oscillator to stop, only after confirming that the XTAL oscillator is stopped by the XTALSTBF bit, it can enter the stop mode, power-down mode or start the XTAL oscillator again.

4.11.4 CMU XTAL Oscillatory Fault Control Register (CMU_XTALSTDCR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
XTALSTDE	-	-	-	-	XTALSTDRI	XTALSTDRE	XTALSTDIE
Bit	Marking	Place name	Function				Read and write
B7	XTALSTDE	XTAL Oscillatory Fault Detection Function enable	0: Disable XTAL Oscillation Fault Detection 1: Enable XTAL Oscillation Fault Detection Note: Oscillator fault detection is to detect the oscillator abnormal caused by external factors. Please disable the oscillator fault detection function before entering stop mode or power-off mode.				R/W
b6~b3	Reserved	-	Read as "0", write as "0"				R/W
b2	XTALSTDRI	XTAL Oscillatory Fault Reset and Interrupt Selection	0: XTAL Oscillation Fault generates interrupt 1: XTAL Oscillation Fault generates reset Note: When PLLH and PLLA select the XTAL clock as the input source, the XTAL oscillation fault can only be selected to generate a reset.				R/W
b1	XTALSTDRE	XTAL Oscillatory Fault Reset enable	0: Disable XTAL Oscillatory Fault Reset 1: Enable XTAL Oscillatory Fault Reset				R/W
b0	XTALSTDIE	XTAL Oscillation Fault Interrupt enable	0: Disable XTAL Oscillatory Fault interrupt 1: Enable XTAL Oscillatory Fault interrupt If set the PWM output of Timer6/Timer4 to Hiz output through EMB, and the XTALSTDIE bit needs to be set to 1 before.				R/W

Note:

- When XTAL is selected as system clock or PLLH/PLLA clock source, writing XTALSTP to "1" to stop XTAL oscillator is forbidden.

4.11.5 CMU XTAL Oscillatory Fault State Register (CMU_XTALSTDSR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	XTALSTDF

Bit	Marking	Place name	Function	Read and write
b7~b1	Reserved	-	Read as "0", write as "0"	R/W
b0	XTALSTDF	XTAL Oscillatory Fault Status flag	0: No XTAL oscillation fault detected 1: XTAL Oscillation Fault is Detected Setting conditions: XTAL Oscillation Faults under XTALSTDE is set 1 Clear condition: When the system clock selects a clock other than XTAL, read 1 and write 0.	R/W

4.11.6 CMU XTAL32 Configuration Register (CMU_XTAL32CFGGR)

Reset value: 0xXX

B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	XTAL32IE			XTAL32DRV[2:0]

Bit	Marking	Place name	Function	Read and write
b7~b4	Reserved	-	Read as "0", write as "0"	R/W
b3	XTAL32IE	XTAL32 clock input enable	0: Disable input of XTAL32 clock through PC15 pin 1: Enable input of XTAL32 clock through PC15 pin	R/W
b2~b0	XTAL32DRV[2:0]	XTAL32 drive capability selection	000: Medium drive capability 001: Big drive capability other: Disable setting Note: For the usage, refer to the chapter on electrical characteristics [low-speed external clock generated by crystal oscillator/ceramic resonator]	R/W

4.11.7 CMU XTAL32 Filter Register (CMU_XTAL32NFR)

Reset value: 0xXX

B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-		XTAL32NF[1:0]

Bit	Marking	Place name	Function	Read and write
b7~b2	Reserved	-	Read as "0", write as "0"	R/W
b1~b0	XTAL32NF[1:0]	XTAL32 oscillator filter selection	00: RUN Mode/Stop Mode/Power Down Mode, input filter 3us of XTAL32 is valid 01: The input filter 3us of XTAL32 in RUN mode is valid, but invalid in stop mode or power-down mode 10: Prohibitions 11: RUN mode/stop mode/power-down mode, the input filter 3us of XTAL32 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

Bit	Marking	Place name	Function	Read and write
b7~b1	Reserved	-	Read as "0", write as "0"	R/W
b0	XTAL32STP	XTAL32 oscillator start and stop	0: XTAL32 oscillator oscillates 1: XTAL32 oscillator stopped	R/W

Note:

- When XTAL32 is selected as the system clock source, it is forbidden to write "1" to XTAL32STP to stop the XTAL32 oscillator.
- Once the software sets the XTAL32 to start, waits for 5 XTAL32 cycles before stopping XTAL32 again.
- Once the software sets XTAL32 to stop, 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
HRCTRM[7:0]							

Bit	Marking	Place name	Function	Read and write
b7~b0	HRCTRM[7:0]	HRC frequency calibration	Frequency calibration needs to be within the guaranteed frequency range of HRC. 10000000: -128 10000001: -127 11111111: -1 00000000: Center Code 00000001: +1 01111110: +126 01111111: +127	R/W

Note:

- Frequency calibration needs to be within the guaranteed frequency range of HRC.

4.11.10 CMU HRC control register (CMU_HRCCR)

Reset value: Determined by the value of ICG1.HRCSTP

B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	HRCSTP

Bit	Marking	Place name	Function	Read and write
b31~b1	Reserved	-	Read as "0", write as "0"	R/W
b0	HRCSTP	HRC start and stop	0: HRC oscillator Oscillates 1: HRC oscillator stops After reset, the HRC oscillator starts or stops according to the configuration of ICG1.HRCSTOP.	R/W

Note:

- When HRC is selected as the system clock source or the PLLH/PLLA clock source, prohibit writing "1" to CMU_HRCCR.HRCSTP to stop the HRC clock.
- Once the software starts the HRC oscillation. only after confirming that the HRC is stable through the HRCSTBF bit, it can enter the stop mode, power-down mode or stop the HRC again.
- Once the software stops the HRC oscillation. only after confirming that the HRC is stopped by the HRCSTBF bit, it can enter the 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]							

Bit	Marking	Place name	Function	Read and write
b7~b0	MRCTRIM[7:0]	MRC frequency calibration	10000000: -128 10000001: -127 11111111: -1 00000000: Center Code 00000001: +1 01111110: +126 01111111: +127	R/W

Note:

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

4.11.12 CMU MRC Control Register (CMU_MRCCR)

Reset value: 0x80

B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	MRCSTP
Bit	Marking	Place name	Function				Read and write
B7	Reserved	-	Read as "1", write as "1"				R/W
b6~b1	Reserved	-	Read as "0", write as "0"				R/W
b0	MRCSTP	MRC oscillator start and stop	0: MRC oscillator oscillates 1: MRC oscillator stopped Note: 1) Once the XTAL oscillation faults is detected, this bit is cleared at the same time, and the MRC oscillates. 2) when the PWC_STPMCR.CKSMRC bit is 1, stop mode wake-up action is active during the MRC oscillator oscillates.				R/W

Note:

- When MRC is selected as the system clock source, it is forbidden to write "1" to MRCSTP to stop the MRC clock.
- Once the software starts the MRC oscillator, waits for 5 MRC cycles before entering the stop mode, power-down mode or stopping the MRC again.
- Once the software stops the MRC oscillator, waits for 5 MRC cycles before entering the stop mode, power-down mode or restarting the MRC again.
- When the MRC is used as an RTC calibration clock, if the RTC is not initialized, the MRC may oscillate even if the MRCSTP is written "1". If the RTC calibration function is valid, MRC oscillates ignored whether MRCSTP bit is set.

4.11.13 CMU LRC Calibration Register (CMU_LRCTRIM)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
			LRCTRIM[7:0]				
Bit	Marking	Place name	Function				Read and write
b7~b0	LRCTRIM[7:0]	LRC frequency calibration	10000000: -128 10000001: -127 11111111: -1 00000000: Center Code 00000001: +1 01111110: +126 01111111: +127				R/W

Note:

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

4.11.14 CMU LRC control register (CMU_LRCCR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	LCRSTP

Bit	Marking	Place name	Function	Read and write
b31~b1	Reserved	-	Read as "0", write as "0"	R/W
b0	LCRSTP	LRC oscillator start and stop	0: LRC oscillator Oscillates 1: LRC oscillator stops	R/W

Note:

- When LRC is selected as system clock, writing "1" to LRCSTP to stop LRC clock is forbidden.
- Once the software starts the LRC oscillator, waits for 5 LRC cycles before entering the stop mode, power-down mode or stopping the LRC again.
- Once the software stops the LRC oscillator, waits for 5 LRC cycles before entering the stop mode, power-down mode or starting the LRC again.
- When waiting for the XTAL oscillator, HRC, PLLH, and PLLA clocks to stabilize, the LRCSTP bit setting is ignored, and the LRC is forced to oscillate.

4.11.15 CMU RTCLRC Calibration Register (CMU_RTCLRCTRIM)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
RTCLRCTRIM[7:0]							

Bit	Marking	Place name	Function	Read and write
b7~b0	RTCLRCTRIM[7:0]	RTCLR frequency calibration	10000000: -128 10000001: -127 11111111: -1 00000000: Center Code 00000001: +1 01111110: +126 01111111: +127	R/W

Note:

- The frequency calibration needs to be within the RTCLR frequency guarantee range.

4.11.16 CMU PLLH Configuration Register (CMU_PLLHCFGR)

Reset value: 0x11101300

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PLLHP[3:0]			PLLHQ[3:0]			PLLHR[3:0]			-	-	-	-	-	-	-

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
PLLHN[7:0]			PLLHS RC			-	-	-	-	-	-	-	PLLHM[1:0]		

Bit	Marking	Place name	Function	Read and write
b31-b28	PLLHP[3:0]	PLLHP frequency division factor	Used for setting PLLHP clock frequency, write to PLLHP when PLLH in stop condition. PLLHP output clock frequency = PLLH VCO frequency / PLLHP 0000: Disable setting 0001: 2 frequency division 0010: 3 frequency division 0011: 4 frequency division 1101: 14 frequency division 1110: 15 frequency division 1111: 16 frequency division	R/W
b27-b24	PLLHQ[3:0]	PLLHQ frequency division factor	Used for setting PLLHQ clock frequency, write to PLLHQ when PLLH in stop condition. PLLHQ output clock frequency = PLLH VCO frequency / PLLHQ 0000: Disable setting 0001: 2 frequency division 0010: 3 frequency division 0011: 4 frequency division 1101: 14 frequency division 1110: 15 frequency division 1111: 16 frequency division	R/W
b23-b20	PLLHR[3:0]	PLLHR frequency division factor	Used for setting PLLHR clock frequency, write PLLHR in PLLH stop condition. PLLHR output clock frequency = PLLH VCO frequency / PLLHR 0000: Disable setting 0001: 2 frequency division 0010: 3 frequency division 0011: 4 frequency division 1101: 14 frequency division 1110: 15 frequency division 1111: 16 frequency division	R/W
b19-b16	-	-	Read as "0", write as "0"	R/W
b15-b8	PLLHN[7:0]	PLLHN frequency multiplication factor	Used for setting loop divider of PLLH VCO clock frequency, it's multiplication factor of PFD input clock and get VCO clock frequency. write to PLLHN when PLLH in stop condition. Make sure that the VCO frequency of the PLLH is between 600MHz and 1200MHz. VCO frequency of PLLH = PFD input clock frequency of PLLH * PLLHN 00011000: 25	R/W

			00011001: 26 00011010: 27 00011011: 28 10010100: 149 10010101: 150	
B7	PLLSRC	PLLH/PLLA input clock source selection	0: Select the external high-speed oscillator as the input clock of PLLH/PLLA 1: Select the internal high-speed oscillator as the input clock for PLLH/PLLA	R/W
b6-b2	-	-	Read as "0", write as "0"	R/W
b1-b0	PLLHM[1:0]	PLLH input clock frequency division factor	Used to divide the PLLH input clock before the PLLH's VCO and get the PFD input clock. Write to PLLHM when PLLH in stop condition. PFD input clock frequency of PLLH= PLLH input clock frequency / PLLHM Make sure that the PFD input clock frequency to PLLH is between 8MHz and 25MHz 00: 1 frequency division 01: 2 frequency division 10: 3 frequency division 11: 4 frequency division	R/W

4.11.17 CMU PLLH Control Register (CMU_PLLHCR)

Reset value: 0x01

B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	PLLHOFF

Bit	Marking	Place name	Function	Read and write
b7~b1	Reserved	-	Read as "0", write as "0"	R/W
b0	PLLHOFF	PLLH enable	Used to start and stop PLLH. Do not set this bit to 1 if the PLLH clock is used as the system clock. 0: PLLH starts working 1: PLLH is stopped	R/W

Note:

- When PLLH is selected as the system clock source, write "1" to PLLHOFF to stop the PLLH clock is forbiden.
- Once the software sets the PLLH to start working, only after confirming that the PLLH is stable through the PLLHSTBF bit, it can enter the stop mode, power-down mode or software setting to stop the PLLH again.
- Once the software sets the PLLH to stop, only after confirming that the PLLH is stopped by the PLLHSTBF bit, it can enter the stop mode, power-down mode or start the PLLH again.
- When PLLH selects the XTAL oscillator as the clock source, the PLLH only can be set to start after confirming that the XTAL oscillator is stable through the XTALSTBF bit. When PLLH selects HRC as the clock source, the PLLH only can be set to start after confirming that the HRC is stable through the HRCSTBF bit.

4.11.18 CMU PLLA Configuration Register (CMU_PLLACFGR)

Reset value: 0x11101300

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PLLAP[3:0]				PLLAQ[3:0]				PLLAR[3:0]				-	-	-	PLLA N[8]
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
PLLAN[7:0]								-	-	-	PLLAM[4:0]				
Bit	Marking	Place name	Function										Read and write		
b31-b28	PLLAP[3:0]	PLLA frequency division factor	Used for setting PLLAP clock frequency, write to PLLAP when PLLA in stop condition PLLAP output clock frequency = PLLA's VCO frequency / PLLAP 0000: Disable setting 0001: 2 frequency division 0010: 3 frequency division 0011: 4 frequency division 1101: 14 frequency division 1110: 15 frequency division 1111: 16 frequency division										R/W		
b27-b24	PLLAQ[3:0]	PLLA frequency division factor	Used for setting PLLAQ clock frequency, write to PLLAQ when PLLA in stop condition PLLAQ output clock frequency = PLLA VCO frequency / PLLAQ 0000: Disable setting 0001: 2 frequency division 0010: 3 frequency division 0011: 4 frequency division 1101: 14 frequency division 1110: 15 frequency division 1111: 16 frequency division										R/W		
b23-b20	PLLAR[3:0]	PLLA frequency division factor	Used for setting PLLAR clock frequency, write to PLLAR when PLLA in stop condition PLLAR output clock frequency = PLLA's VCO frequency / PLLAR 0000: Disable setting 0001: 2 frequency division 0010: 3 frequency division 0011: 4 frequency division 1101: 14 frequency division 1110: 15 frequency division 1111: 16 frequency division										R/W		
b19-b17	-	-	Read as "0", write as "0"										R/W		
b16-b8	PLLAN[8:0]	PLLA frequency multiplication factor	Used for setting loop divider of PLLA VCO clock frequency, it's multiplication factor of PLLA's PFD input clock and get VCO clock frequency, write to PLLAN when PLLA in stop condition. Make sure that the VCO frequency of the PLLA is between 240MHz and 480MHz. PLLA's VCO frequency = PLLA's PFD input clock frequency * PLLAN 000010011: 20 000010100: 21										R/W		

			000010101: 22 000010110: 23 111011101: 478 111011110: 479 111011111: 480 Other prohibitions	
b7-b5	-	-	Read as "0", write as "0"	R/W
b4-b0	PLLAM[4:0]	PLL input clock frequency division factor	<p>Used to divide the PLLA input clock before the PLLA's VCO and get the PFD input clock. Write to PLLAM when PLLA in stop condition. Make sure that the PFD input clock frequency of the PLLA is between 1MHz and 25MHz.</p> <p>PFD input clock frequency of PLLA= PLLA input clock frequency / PLLAM</p> <p>00000: Disable setting</p> <p>00001: 2 frequency division</p> <p>00010: 3 frequency division</p> <p>.....</p> <p>11000: 25 frequency division</p>	R/W

4.11.19 CMU PLLA Control Register (CMU_PLLACR)

Reset value: 0x01

B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	PLLAOFF

Bit	Marking	Place name	Function	Read and write
b7~b1	Reserved	-	Read as "0", write as "0"	R/W
b0	PLLAOFF	PLLA enable	Used to start and stop PLLA. 0: PLLA starts working 1: PLLA stopped	R/W

Note:

- When PLLA is selected as the clock source of I2S/TRNG/ADC/USB/CAN, write "1" to PLLAOFF to stop PLLA clock is forbidden.
- Once the software sets the PLLA to start working, only after confirming that the PLLA is stable through the PLLASTBF bit, it can enter the stop mode, power-down mode or software setting to stop the PLLA.
- Once the software sets the PLLA to stop, only after confirming that the PLLA is stopped by the PLLASTBF bit, it can enter the stop mode, power-down mode or start the PLLA again.
- When the PLLA selects the XTAL oscillator as the clock source, the PLLA only can be set to start after confirming that the XTAL oscillator is stable through the XTALSTBF bit. When the PLLA selects HRC as the clock source, the PLLA only can be set to start only after confirming that the HRC is stable through the HRCSTBF bit.

4.11.20 CMU clock Stabilizer (CMU_OSCSTBSR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
-	PLLASTBF	PLLHSTBF	-	XTALSTBF	-	-	HRCSTBF

Bit	Marking	Place name	Function	Read and write
B7	Reserved	-	Read as "0", write as "0"	R
b6	PLLASTBF	PLL stable flag	0: PLLA stopped or not stable 1: PLLA is stable	R
b5	PLLHSTBF	PLLH stable flag	0: PLLH stopped or not stable 1: PLLH is stable	R
b4	Reserved	-	Read as "0", write as "0"	R
b3	XTALSTBF	XTAL stability sign bit	0: XTAL stopped or unstabilized 1: XTAL Stabilization	R
b2~b1	Reserved	-	Read as "0", write as "0"	R
b0	HRCSTBF	HRC stable sign bit	0: HRC stopped or unstabilized 1: HRC stability	R

4.11.21 Clock Switch Register of System (CMU_CKSWR)

Reset value: 0x01

B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-		CKSW[2:0]

Bit	Marking	Place name	Function	Read and write
b31~b3	Reserved	-	Read as "0", write as "0"	R/W
b2-b0	CKSW[2:0]	System clock switch	0 0 0: select HRC clock as system clock 0 0 1: Select MRC clock as system clock 0 1 0: Select LRC clock as system clock 0 1 1: Select XTAL clock as system clock 1 0 0: select XTAL32 clock as system clock 1 0 1: Select PLLH as system clock 1 1 0: Disable setting 1 1 1: Disable setting Note: 1. Ensure the target clock is stable before switch to it. 2. Switch flow refers to [Clock switch] chapter 3. When the PWC_STPMCR.CKSMRC bit is 1, after the stop mode wakes up, this register is initialized and the MRC is selected as system clock source.	R/W

4.11.22 CMU Clock Divider Configuration Register (CMU_SCFGREG)

Reset value: 0x00000000

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
<hr/>															
b31~b27	Reserved	-													Read and write
<hr/>															
b26~24	HCLKS[2:0]	HCLK clock divider selection	000: 1 frequency division of system clock 001: 2 frequency division of system Clock 010: 4 frequency division of system clock 011: 8 frequency division of system clock 100: 16 frequency division of system clock 101: 32 frequency division of system clock 110: 64 frequency division of system clock 111: Disable setting Note: When the PWC_STPMCR.CKSMRC bit is 1, after the stop mode wakes up, this register is initialized and HCLK is divided by 1 of system clock.												R/W
															R/W
b23	Reserved	-													R/W
<hr/>															
b22~20	EXCKS[2:0]	EXCLK clock divider selection	000: 1 frequency division of system clock 001: 2 frequency division of system Clock 010: 4 frequency division of system clock 011: 8 frequency division of system clock 100: 16 frequency division of system clock 101: 32 frequency division of system clock 110: 64 frequency division of system clock 111: Disable setting Note: When the PWC_STPMCR.CKSMRC bit is 1, after the stop mode wakes up, this register is initialized and EXCLK is divided by 1 of system clock.												R/W
															R/W
b19	Reserved	-													R/W
<hr/>															
b18~16	PCLK4S[2:0]	PCLK4 clock divider selection	000: 1 frequency division of system clock 001: 2 frequency division of system Clock 010: 4 frequency division of system clock 011: 8 frequency division of system clock 100: 16 frequency division of system clock 101: 32 frequency division of system clock 110: 64 frequency division of system clock 111: Disable setting Note: When the PWC_STPMCR.CKSMRC bit is 1, after the stop mode wakes up, this register is initialized, and PCLK4 is divided by 1 of system clock.												R/W
															R/W
b15	Reserved	-													R/W
<hr/>															
b14~12	PCLK3S[2:0]	PCLK3 clock divider selection	000: 1 frequency division of system clock 001: 2 frequency division of system Clock 010: 4 frequency division of system clock 011: 8 frequency division of system clock 100: 16 frequency division of system clock 101: 32 frequency division of system clock 110: 64 frequency division of system clock 111: Disable setting Note: When the PWC_STPMCR.CKSMRC bit is 1, after the stop mode wakes up, this register is initialized, and PCLK3 is divided by 1 of system clock.												R/W
															R/W
b11	Reserved	-													R/W
<hr/>															
b10~8	PCLK2S[2:0]	PCLK2 clock divider selection	000: 1 frequency division of system clock 001: 2 frequency division of system Clock 010: 4 frequency division of system clock 011: 8 frequency division of system clock 100: 16 frequency division of system clock 101: 32 frequency division of system clock												R/W
															R/W

<p style="text-align: center;">110: 64 frequency division of system clock 111: Disable setting Note: When the PWC_STPMCR.CKSMRC bit is 1, after the stop mode wakes up, this register is initialized, and PCLK2 is divided by 1 of system clock.</p>			
B7	Reserved	-	Read as "0", write as "0" R/W
b6~4	PCLK1S[2:0]	PCLK1 clock divider selection	<p style="text-align: center;">000: 1 frequency division of system clock 001: 2 frequency division of system Clock 010: 4 frequency division of system clock 011: 8 frequency division of system clock 100: 16 frequency division of system clock 101: 32 frequency division of system clock 110: 64 frequency division of system clock 111: setting is prohibited</p> <p>Note: When the PWC_STPMCR.CKSMRC bit is 1, after wake-up from stop mode, this register is initialized, and PCLK1 is divided by 1 of system clock.</p>
b3	Reserved	-	Read as "0", write as "0" R/W
b2~0	PCLK0S[2:0]	PCLK0 clock divider selection	<p style="text-align: center;">000: 1 frequency division of system clock 001: 2 frequency division of system Clock 010: 4 frequency division of system clock 011: 8 frequency division of system clock 100: 16 frequency division of system clock 101: 32 frequency division of system clock 110: 64 frequency division of system clock 111: setting is prohibited</p> <p>Note: When the PWC_STPMCR.CKSMRC bit is 1, after the stop mode wakes up, this register is initialized, and PCLK0 is divided by 1 of system clock.</p>

4.11.23 CMU USB Clock Configuration Register (CMU_USBCKCFGR)

Reset value: 0x40

B7	b6	b5	b4	b3	b2	b1	b0
			USBCKS[3:0]	-	-	-	-
Bit	Marking	Place name	Function				Read and write
b7~b4	USBCKS[3:0]	48MHz clock source selection for USB-FS/USB-HS	0001: system clock divided by 2 0010: System clock divided by 3 0011: system clock divided by 4 0100: system clock divided by 5 0101: System clock divided by 6 0110: System clock divided by 7 0111: System clock divided by 8 1000: PLLH/Q 1001: PLLH/R 1010: PLLA/P 1011: PLLA/Q 1100: PLLA/R Other setting is prohibited	R/W			
b3~b0	Reserved	-	Read as "0", write as "0"	R/W			

4.11.24 CMU CAN Clock Configuration Register (CMU_CANCKCFGR)

Reset value: 0xdd

B7	b6	b5	b4	b3	b2	b1	b0	
			CAN2CKS[3:0]	CAN1CKS[3:0]				
Bit	Marking	Place name	Function					Read and write
b7~b4	CAN2CKS[3:0]	CAN2 communication clock selection	0001: system clock divided by 2 0010: system clock divided by 3 0011: system clock divided by 4 0100: system clock divided by 5 0101: System clock divided by 6 0110: System clock divided by 7 0111: System clock divided by 8 1000: PLLH/Q 1001: PLLH/R 1010: PLLA/P 1011: PLLA/Q 1100: PLLA/R 1101: XTAL Other setting is prohibited					R/W
			Note: 1. Ensure that the PLLH/PLLA clock is stable before switch the target clock source to it. 2. When PLLH is selected as the system clock, USB, CAN, QSPI, SPI, universal timer, FCM, ADC, DAC should be set as the module stop state, and then CMU_SCFGR register should be written to switch clock frequency division. After writing the CMU_CANCKCFGR register, the software waits for the system to stabilize for at least 30us. 3. When the PWC_STPMCR.CKSMRC bit is 1, after wake-up from stop mode this register is initialized, and CANCLK is the clock source of XTAL.					
b3~b0	CAN1CKS[3:0]	CAN communication clock selection	0001: system clock divided by 2 0010: system clock divided by 3 0011: system clock divided by 4 0100: system clock divided by 5 0101: System clock divided by 6 0110: System clock divided by 7 0111: System clock divided by 8 1000: PLLH/Q 1001: PLLH/R 1010: PLLA/P 1011: PLLA/Q 1100: PLLA/R 1101: XTAL Other setting is prohibited					R/W
			Note: 1. Ensure that the PLLH/PLLA clock is stable before switch the target clock source to it. 2. When PLLH is selected as the system clock, USB, CAN, QSPI, SPI, universal timer, FCM, ADC, DAC should be set as the module stop state, and then CMU_SCFGR register should be written to switch clock frequency division. After writing the CMU_CANCKCFGR register, the software waits for the system to stabilize for 30μs. 3. When the PWC_STPMCR.CKSMRC bit is 1, after wake-up from stop mode this register is initialized, and CANCLK is the clock source of XTAL.					

4.11.25 CMU AD/TRNG/DA Clock Configuration Register (CMU_PERICKSEL)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PERICKSEL[3:0]

Bit	Marking	Place name	Function	Read and write
b15~b4	Reserved	-	Read as "0", write as "0"	R/W
b3~b0	PERICKSEL[3:0]	AD/TRNG clock source selection	0000: PCLK2/PCLK4 set by CMU_SCFGR 1000: PCLK2/PCLK4 configured as PLLHQ 1001: PCLK2/PCLK4 configured as PLLHR 1010: PCLK2/PCLK4 configured as PLLAP 1011: PCLK2/PCLK4 configured as PLLAQ 1100: PCLK2/PCLK4 configured as PLLAR Other setting is prohibited.	R/W

Note:

- Ensure that the PLLH/PLLA clock is stable before switch the target clock source to it..

4.11.26 CMU Debug Clock Configuration Register (CMU_TPIUCKCFGREG)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
TPIUCKOE	-	-	-	-	-	-	TPIUCKS[1:0]

Bit	Marking	Place name	Function	Read and write
B7	TPIUCKOE	TPIU clock supply enable	0: Disable 1: Enable	R/W
b6~b2	-	-	Read as "0", write as "0"	R/W
b1~0	TPIUCKS[1:0]	TPIU clock divider select bits	00: 1 frequency division 01: 2 frequency division 10: 4 frequency division Other setting is prohibited.	R/W

4.11.27 CMU MCO1 Configuration Register (CMU_MCO1CFGR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
MCO1EN	MCO1DIV[2:0]				MCO1SEL[3:0]		
<hr/>							
Bit	Marking	Place name	Function		Read and write		
B7	MCO1EN	MCO_1 Output enable	0: Disable MCO_1 Output 1: Enable MCO_1 Output		R/W		
b6~b4		MCO_1 Frequency Division Selection	000: 1 frequency division 001: 2 frequency division 010: 4 frequency division 011: 8 frequency division 100: 16 frequency division 101: 32 frequency division 110: 64 frequency division 111: 128 frequency division		R/W		
b3~b0		MCO_1 output clock source selection	0000: HRC clock 0001: MRC clock 0010: LRC clock 0011: XTAL Clock 0100: XTAL32 clock 0110: PLLHP 0111: PLLAP 1000: PLLHQ 1001: PLLAQ 1010: PLLAR 1011: System clock Other setting is prohibited.		R/W		

4.11.28 CMU MCO2 Configuration Register (CMU_MCO2CFGR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
MCO2EN	MCO2DIV[2:0]				MCO2SEL[3:0]		
<hr/>							
Bit	Marking	Place name	Function		Read and write		
B7	MCO2EN	MCO_2 output enable	0: Disable MCO_2 output 1: Enable MCO_2 output		R/W		
b6~b4		MCO_2 frequency division selection	000: 1 frequency division 001: 2 frequency division 010: 4 frequency division 011: 8 frequency division 100: 16 frequency division 101: 32 frequency division 110: 64 frequency division 111: 128 frequency division		R/W		
b3~b0		MCO_2 output clock source selection	0000: HRC clock 0001: MRC clock 0010: LRC clock 0011: XTAL Clock 0100: XTAL32 clock 0110: PLLHP 0111: PLLAP 1000: PLLHQ 1001: PLLAQ 1010: PLLAR 1011: System clock Other setting is prohibited.		R/W		

4.11.29 FCM Lower Limit Compare Value Register (FCM_LVR)

Reset value: 0x00000000

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]															

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15~b0	LVR[15:0]	Lower limit comparison value	This register is configured when the START bit is 0.	R/W

4.11.30 FCM Upper Limit Compare Value Register (FCM_UVR)

Reset value: 0x00000000

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]															

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15~b0	UVR[15:0]	upper limit comparison value	This register is configured when the START bit is 0.	R/W

4.11.31 FCM Counter Value Register (FCM_CNTR)

Reset value: 0x00000000

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]															
Bit	Marking	Place name	Function										Read and write		
b31~b16	Reserved	-	Read as "0", write as "0"										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 the first valid edge after START=1)										R		

4.11.32 FCM Start Stop Register (FCM_STR)

Reset value: 0x00000000

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
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	STAR T
Bit	Marking	Place name	Function										Read and write		
b31~b1	Reserved	-	Read as "0", write as "0"										R/W		
b0	START	Frequency measurement start	0: Frequency measurement stopped 1: Frequency measurement starts										R/W		

4.11.33 FCM Measurement Object Control Register (FCM_MCCR)

Reset value: 0x00000000

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]		

Bit	Marking	Place name	Function	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
b7~b4	MCKS[3:0]	Measured clock selection	0000: XTAL 0001: XTAL32 0010: HRC 0011: LRC 0100: SWDTLRC 0101: PCLK1 0110: PLLAP 0111: MRC 1000: PLLHP 1001: RTCLRC Other setting is prohibited	R/W
b3~b2	Reserved	-	Read as "0", write as "0"	R/W
b1~b0	MDIVS[1:0]	Measured clock frequency division selection	00: no frequency division 01: 4 frequency division 10: 8 frequency division 11: 32 frequency division	

4.11.34 FCM Measurement Reference Control Register (FCM_RCCR)

Reset value: 0x00000000

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
EXREFE	-	EDGES[1:0]	-	-	-	DNFS[1:0]	INEXS		RCKS[3:0]	-		RDIVS[1:0]			

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15	EXREFE	External input reference clock 'FCMREF' enable	0: Disable external input reference clock FCMREF 1: Enable external to input reference clock FCMREF	R/W
b14	Reserved	-	Read as "0", write as "0"	R/W
b13~b12	EDGES[1:0]	Measurement reference edge selection	1: Rising edge 0: Falling edge 10: Rising and falling edges 11: Prohibit setting	R/W
b11~b10	Reserved	-	Read as "0", write as "0"	R/W
b9~b8	DNFS[1:0]	Digital filter function selection	00: No filter function 01: The clock selected by the MCKS bit is used as the filter clock 10: 4 division of the clock selected by the MCKS bit is used as the filter clock 11: 16 division of the clock selected by the MCKS bit is used as the filter clock	R/W
B7	INEXS	Measurement reference clock input type selection	0: External input is reference clock 'FCMREF' 1: Clock selected by RCKS is reference clock 'FCMREF'	R/W
b6~b3	RCKS[3:0]	Measurement reference clock selection	0000: XTAL 0001: XTAL32 0010: HRC 0011: LRC 0100: SWDTLRC 0101: PCLK1 0110: PLLAP 0111: MRC 1000: PLLHP 1001: RTCLRC other: Prohibit setting.	R/W
b2	Reserved	-	Read as "0", write as "0"	R/W
b1~b0	RDIVS[1:0]	Measurement reference frequency division selection	00: 32 frequency division 01: 128 frequency division 10: 1024 frequency division 11: 8192 frequency division	

4.11.35 FCM Interrupt Reset Control Register (FCM_RIER)

Reset value: 0x00000000

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
-	-	-	-	-	-	-	-	ERRE	-	-	ERRINTRS	-	OVFIE	MENDIE	ERRIE

Bit	Marking	Place name	Function	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
B7	ERRE	Frequency Abnormal Reset Enable	0: Disable 1: Enable	
b6~b5	Reserved	-	Read as "0", write as "0"	R/W
b4	ERRINTRS	Frequency Abnormal Interrupt Reset Selection	0: Interrupt when frequency abnormal occurs 1: Reset when frequency abnormal occurs	
b3	Reserved	-	Read as "0", write as "0"	R/W
b2	OVFIE	Counter overflow interrupt enable	0: Disable counter overflow interrupt 1: Enable counter overflow interrupt	R/W
b1	MENDIE	Measurement end interrupt enable	0: Interrupt at the end of measurement is disabled 1: Interrupt at the end of the measurement is enabled	R/W
b0	ERRIE	Frequency Abnormal Interrupt Enable	0: Disable frequency abnormal interrupt 1: Enable frequency abnormal interrupt	R/W

4.11.36 FCM Flag Register (FCM_SR)

Reset value: 0x00000000

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	ERRF

Bit	Marking	Place name	Function	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
b2	OVF	Counter overflow flag	0: The counter is not overflow 1: Counter is overflow	R
b1	MENDF	Measurement end flag	0: Measurement is going on 1: Measurement ends	R
b0	ERRF	Frequency Abnormal Flag	0: No frequency abnormal occurs 1: Abnormal frequency occurs	R

4.11.37 FCM Flag Clear Register (FCM_CLR)

Reset value: 0x00000000

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 DFCLR	ERRF CLR

Bit	Marking	Place name	Function	Read and write
b31~b3	Reserved	-	Read as "0", write as "0"	R/W
b2	OVFCLR	Counter overflow flag clear	Write "1" to clear the counter overflow flag	W
b1	MENDFCLR	Measurement end flag clear	Write "1" to clear the measurement end flag	W
b0	ERRFCLR	Frequency Abnormal Flag Clear	Write "1" to clear the frequency abnormal flag	W

5 Power controller (PWC)

5.1 Introduction

Power controller is used to control the supply, switching and detection of multiple power domains in multiple operation modes and low power modes. The power controller is composed of power consumption control unit (PWCL), power supply voltage detection unit (PVD), and battery backup control unit (BATBKUP) which automatically switches between VCC and battery power supply.

The chip's operating voltage (VCC) is 1.8V to 3.6V. The voltage regulator (LDO) powers the VDD domain and the VDDR domain, and the VDDR voltage regulator (RLDO) powers the VDDR domain in power-down mode or VCC power-down conditions. The chip provides two operating modes such as high-speed and ultra-low speed through the power control unit (PWCL), and three low-power modes such as sleep, stop and power-down.

Power supply voltage detection unit (PVD) provides power-on reset (POR), power-down reset (PDR), brown-out reset (BOR), programmable voltage detection 1 (PVD1), programmable voltage detection 2 (PVD2), reference voltage measurement, VBAT voltage detection, VBAT voltage measurement and other functions, among which POR, PDR, BOR control the reset action of the chip by detecting the VCC voltage. PVD1 generates a reset or interrupt according to the register setting by detecting the VCC voltage. PVD2 generates a reset or interrupt according to register selection by detecting VCC voltage or external input detection voltage. The reference voltage measurement is the function of using the ADC to measure the reference voltage. VBAT voltage detection is the function of reading the register to get the VBAT voltage higher or lower than the VBAT detection voltage. The VBAT detection reference voltage can be selected as 1.8V or 2.1V by register. The VBAT voltage measurement function is the function of using the ADC to measure the 1/2 voltage of VBAT to obtain the VBAT voltage value.

The battery backup domain maintains the power supply through VBAT under the condition of VCC power down, which ensures that the real-time clock module (RTC) and the wake-up timer (WKT) can continue to operate, and provides power for the RLDO. The VDDR power domain can maintain the power supply through RLDO while the chip enters the power-down mode or the VCC is powered down, and maintains the data of the 4KB backup SRAM (Backup-SRAM). The analog modules are equipped with dedicated power supply pins to improve the performance.

5.2 Distribution of power

Figure 5-1 is the power distribution diagram of the chip. The chip consists of VCC power domain, VDD power domain, AVCC power domain, BATBKUP power domain and VDDR power domain.

The VCC power domain is powered by the VCC/VSS pin, and consists of low-power control unit(PWCL), power supply voltage detection unit (PVD), IO state hold control, voltage regulator (LDO), external high-speed oscillator (XTAL), internal Low-speed oscillator (LRC) and other circuits.

The VDD power domain is composed of digital logic such as CPU, digital peripherals, RAM, FLASH, etc., and is powered by VDD generated by LDO. The RAM in the VDD power domain is divided into 5 groups and consists of 11 independent modules, and the power-down of each module can be independently controlled through registers.

The BATBKUP power domain is a battery backup domain, and the power supply in this domain is automatically swithched between VCC and VBAT through the power switch. When VCC is powered down, the power supply of the backup domain is automatically switched to VBAT. The battery backup domain consists of real-time clock (RTC), wake-up timer (WKTM), external low-speed oscillator (XTAL32), and internal low-speed oscillator (RTCLRC) for RTC.

The VDDR domain consists of 4KB backup RAM (Backup-SRAM). Backup-SRAM is powered by RLDO in power-down mode or when VCC is powered down, and powered by LDO in other power modes. In power-down mode, Backup-SRAM can retain data. When backup RAM is not required in power-down mode or VCC is powered down, PWC_BATCR.VBTRSD can be set to cut off the VDDR domain power to further reduce power consumption.

The AVCC power domain is mainly composed of analog-to-digital converter (ADC), digital-to-analog converter (DAC), comparator (CMP), programmable gain amplifier (PGA), and analog input and output pins, which is powered by AVCC/AVSS pins. To provide high-accuracy analog performance, the analog domain is equipped with independent power supply. To ensure that the ADC has higher accuracy, the reference voltage of the ADC uses independet VREFH/VREFL pins.

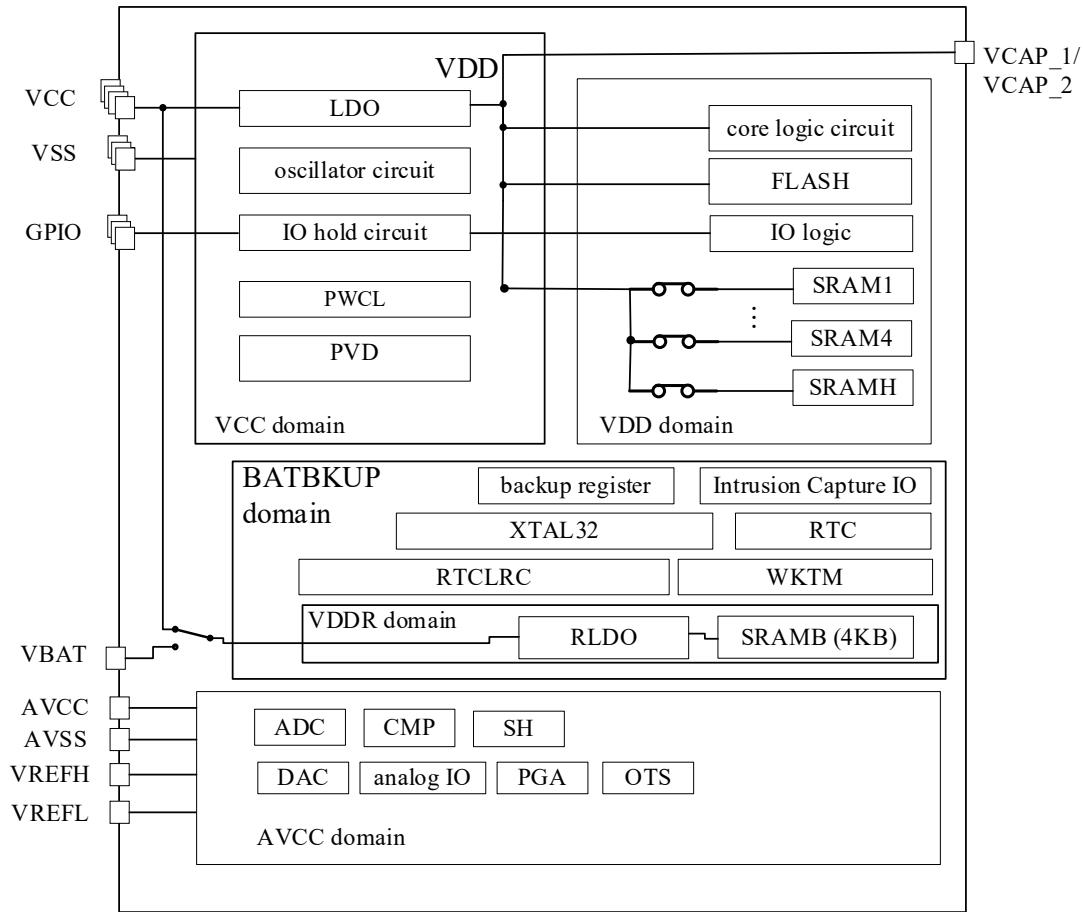


Figure 5-1 Power supply diagram

5.2.1 Battery backup power domain

The battery backup power domain is powered by VCC or VBAT which is selected by an internal power switch. The backup domain includes RTC (real time clock), XTAL32 (external low-speed oscillator), 128 bytes of backup registers, 4K bytes of backup SRAM, internal low-speed oscillator and two input pins of RTCIC0-1. When the voltage on the VCC pin decreases, the real-time clock (RTC) and the internal low-speed oscillator for RTC are powered by battery backup power pin (VBAT). For applications without an external battery, it is recommended to connect the VBAT pin to the VCC pin with an external 100nF ceramic decoupling capacitor.

The switching of the battery backup function operates as Figure 5-2 shown. When the VCC voltage drops below V_{PDR} , the battery backup domain is switched to be powered by VBAT; when the VCC voltage rises above V_{POR} , the battery backup domain is switched to be powered by VCC. If the chip enters power-down mode 3 or power-down mode 4, the detection voltage of V_{PDR} cannot be determined. When the VCC voltage drops, it cannot be guaranteed that the power is correctly switched to the VBAT power supply. Therefore, the chip cannot use power-down mode 3 and battery backup function at the same time, also cannot use power-down mode 4 and battery backup function at the same time.

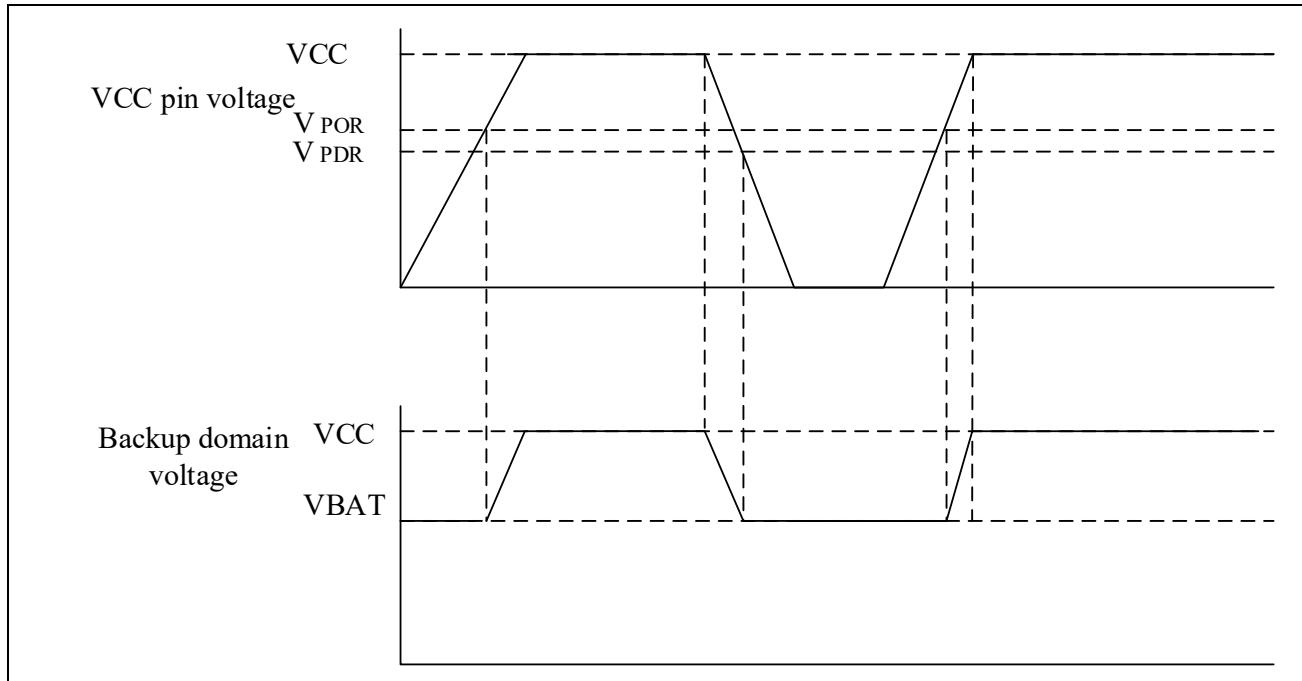


Figure 5-2 Battery Backup Domain Power Switching Diagram

The clock source of the RTC can be the internal low-speed oscillator for the RTC (RTCLRC) or the external low-speed oscillator (XTAL32). Before entering low power mode through WFI/WFE instruction, Cortex™-M4 needs to set expected wakeup time and enable wakeup function through RTC register to configure RTC timer wakeup event.

The backup SRAM can still retain data in power-down mode or even after VCC is powered down, so if VBAT is continuously powered, the backup SRAM can be regarded as an internal EEPROM. When an intrusion event occurs, according to software settings, you can choose whether the backup domain SRAM is powered down or not, and whether the backup register needs to be initialized. Backup SRAM can be powered off by setting the VBTRSD bit in register BRAMC0. After the backup SRAM is powered off, circuits such as a voltage regulator that supplies power to the backup SRAM are also turned off, which can further reduce power consumption. After the VBTRSD bit is reset, the backup SRAM read and write operations cannot be performed until the RAMVALID flag becomes 1.

When using the battery backup domain, you need to write the PWC_VBATTRSTR register to initialize the VBAT domain. After writing 0xA5 to PWC_VBATTRSTR, the external low-speed oscillator XTAL32 and the RLDO for backup RAM are started, and the wake-up timer WKTM is initialized.

5.3 Description of Power Supply Voltage Detection Unit (PVD)

Power supply voltage detection unit (PVD) includes power-on reset (POR), power-down reset (PDR), brown-out reset (BOR), programmable voltage detection 1 (PVD1), programmable voltage detection 2 (PVD2), reference voltage measurement, VBAT voltage detection, VBAT voltage measurement and other functions.

5.3.1 Description of the action of power-on/power-down reset

The chip is integrated with power-on reset circuit and power-down reset circuit. The power-on reset and power-down reset timmings are shown in Figure 5-3, when VCC is higher than the specified threshold V_{POR} , after the T_{RSTPOR} time has passed, the chip is released from the power-on reset state, and the CPU starts to execute instruction. When VCC is below V_{PDR} , the chip remains in reset state. When using power-on reset, the reset pin NRST must be 1. If the reset pin is pulled down, the chip will reset and start by means of pin reset.

For details on parameters such as V_{POR} , V_{PDR} , and T_{RSTPOR} , please refer **to the electrical characteristics in the data sheet**.

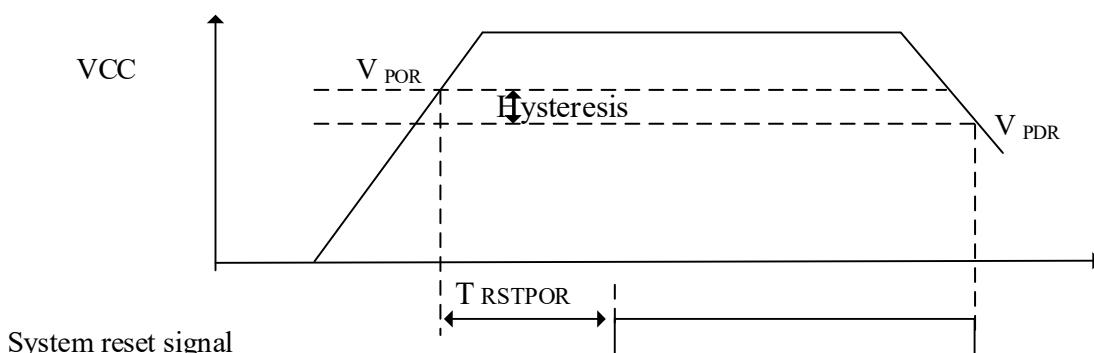


Figure 5-3 Power-on reset, power-down reset waveform

5.3.2 Description of Brown-out Reset (BOR)

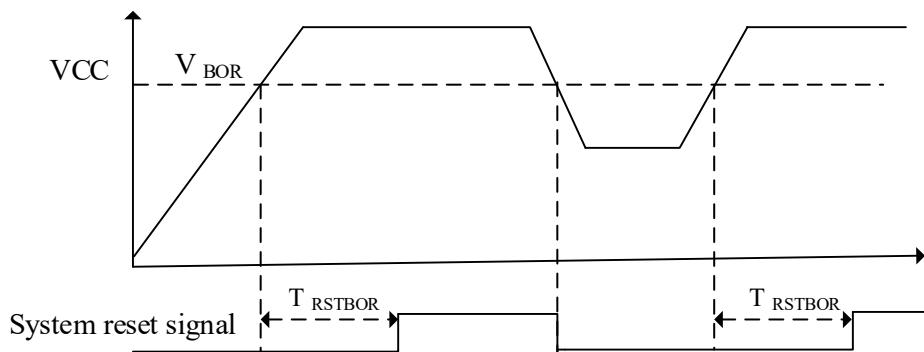
During power-on, a Brown-Out Reset (BOR) will place the chip in reset state until VCC is above V_{BOR} .

The V_{BOR} threshold is configured by BORLEV and BORDIS bits of the initialization configuration registers (ICG). When BORDIS is configured as 0, the BOR detection voltage can be selected from 4 thresholds. When BORDIS is configured as 1, the chip is reset by power-on reset and power-down reset.

Table 5-1 BOR configuration

BORDIS	BOR_lev	Note
1	xx	BOR is invalid
0	00	BOR is valid, select BOR Threshold 0 (V_{BOR0})
0	01	BOR is valid, select BOR Threshold 1 (V_{BOR1})
0	10	BOR is valid, select BOR Threshold 2 (V_{BOR2})
0	11	BOR is valid, select BOR Threshold 3 (V_{BOR3})

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

**Figure 5-4 Brown-out reset waveform**

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

Programmable voltage detection 1 and programmable voltage detection 2 trigger corresponding reset or interrupt actions by detecting whether the VCC supply voltage has passed the detection threshold. The detection circuits are individually programmable.

The event can be programmed to reset/interrupt (maskable/non-maskable)/AOS trigger functions when the supply voltage passes the threshold voltage points of each detection circuit.

Programmable voltage detection key features such as Table 5-2 shown.

Table 5-2 PVD1/PVD2 characteristics

Item	PVD1	PVD2
Detection voltage	During VCC fallings/riseing, whether the voltage passes through the threshold voltage point (V_{PVD1})	During VCC falling/riseing, whether the voltage passes through the threshold voltage point (V_{PVD2}), Whether the fall and rise of the external input voltage passes through the threshold voltage point ($V_{PVD2}, PWC_PVDLCR.PVD2LVL[2:0]=111$)
Detection voltage point	Configured by PVD1LVL[2:0]	Configured by PVD2LVL[2:0]
Reset	Reset: $VCC < V_{PVD1}$; Reset release: When $VCC > V_{PVD1}$, after a certain reset processing time.	Reset: $VCC < V_{PVD2}$; Reset release: When $VCC > V_{PVD2}$, after a certain reset processing time.
Interrupt	Configured voltage detection 1 as interrupt or non-maskable interrupt VCC falls past the threshold voltage point (V_{PVD1}) or VCC rises past the threshold voltage point (V_{PVD1}) Or VCC rises/falls past the threshold voltage point (V_{PVD1})	Configured voltage detection 2 as interrupt or non-maskable interrupt VCC falls past the threshold voltage point (V_{PVD2}) or VCC rises past the threshold voltage point (V_{PVD2}) Or VCC rises/falls past the threshold voltage point (V_{PVD2})
Filtering function	Digital filtering	Digital filtering
AOS trigger function	VCC falls past the threshold voltage point (V_{PVD1})	VCC falls past the threshold voltage point (V_{PVD2})

5.3.4 PVD1, PVD2 Interrupt/Reset Block Diagram

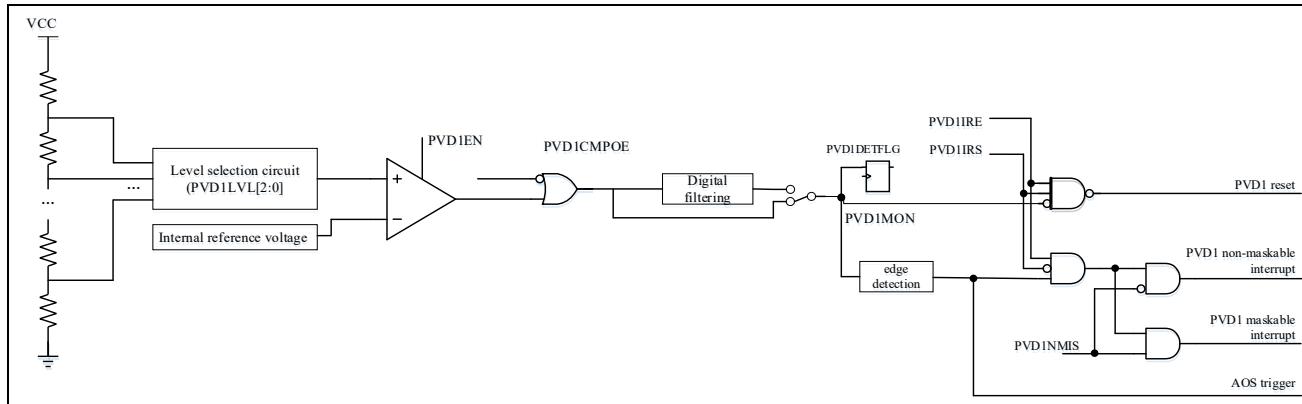


Figure 5-5 PVD1 Interrupt/Reset Block Diagram

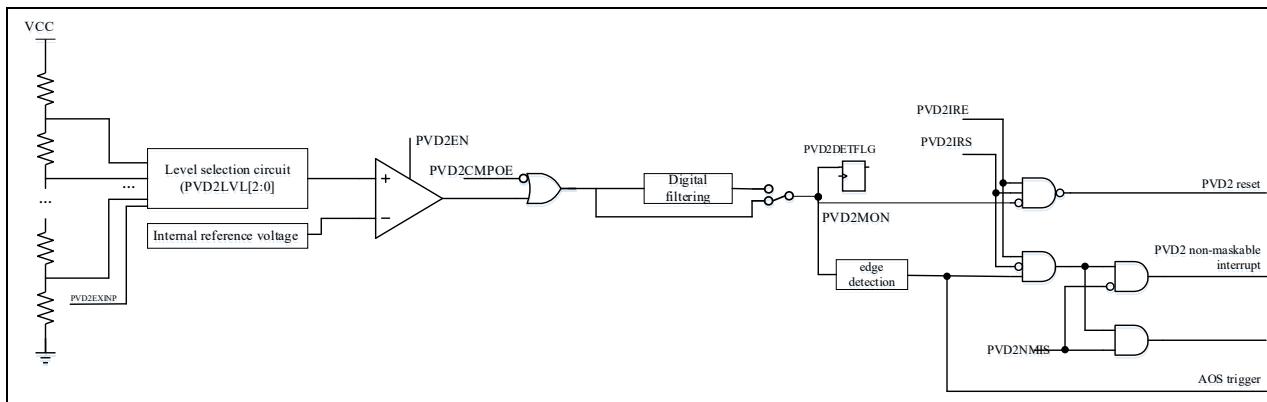


Figure 5-6 PVD2 interrupt/Reset Block Diagram

5.3.5 Input/output pin

Pin name	Input/Output	Function
PVD2EXINP	Input	External input PVD2comparison voltage

5.3.6 PVD1 Interrupt and Reset

Observe the following precautions when using the PVD1 detection in stop mode or power-down mode.

1. Stop mode
 - 1) The digital filter must be disabled.
2. Power-down mode
 - 1) The digital filter must be disabled.
 - 2) Configure PVD1IRS as 0, and PVD1 is selected to generate an interrupt; when the reset function is selected, the power-down mode cannot be entered.

The following figure is the timing sequence diagram of the voltage detection 1 interrupt. PVD1DETFLG needs to be cleared before the interrupt can occur again.

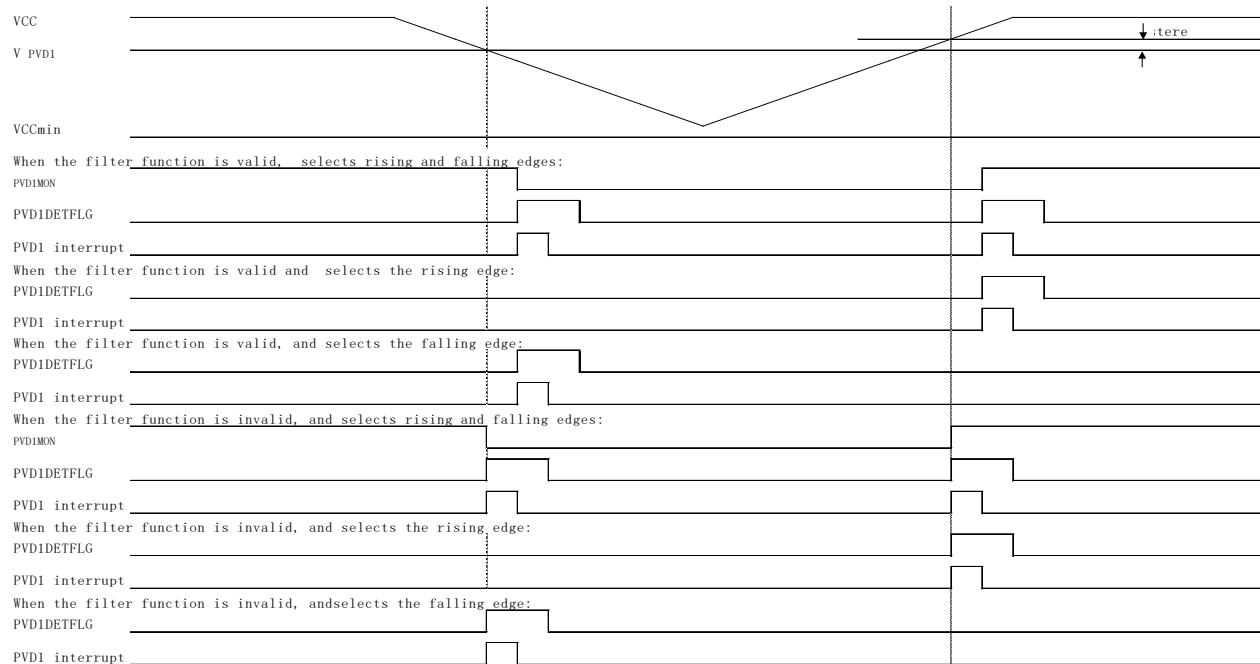


Figure 5-7 Power Voltage Detection 1 Interrupt Timing Chart

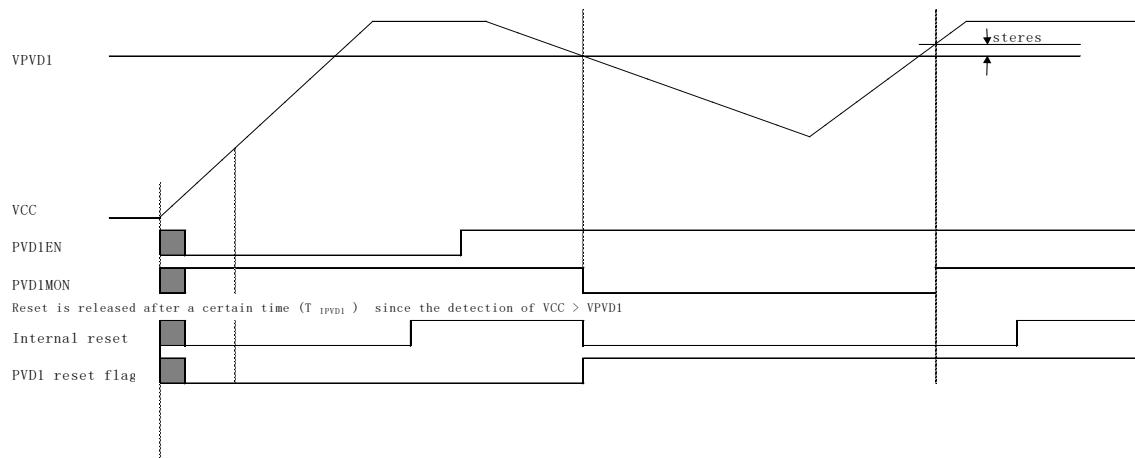


Figure 5-8 Power Voltage Detection 1 Reset Timing Chart

5.3.7 PVD2 Interrupt and Reset

When using the PVD2 detection in stop mode or power down mode, please observe the following precautions:

1. Stop mode
 - 1) The digital filter must be disabled.
2. Power-down mode
 - 1) The digital filter must be disabled.
 - 2) Configure PVD2INTRS as 0, and PVD2 is selected to generate an interrupt; when the reset function is selected, the power-down mode cannot be entered.

The following figure is the timing sequence diagram of the voltage detection 2 interrupt, PVD2DETFLG needs to be cleared before the interrupt can occur again.

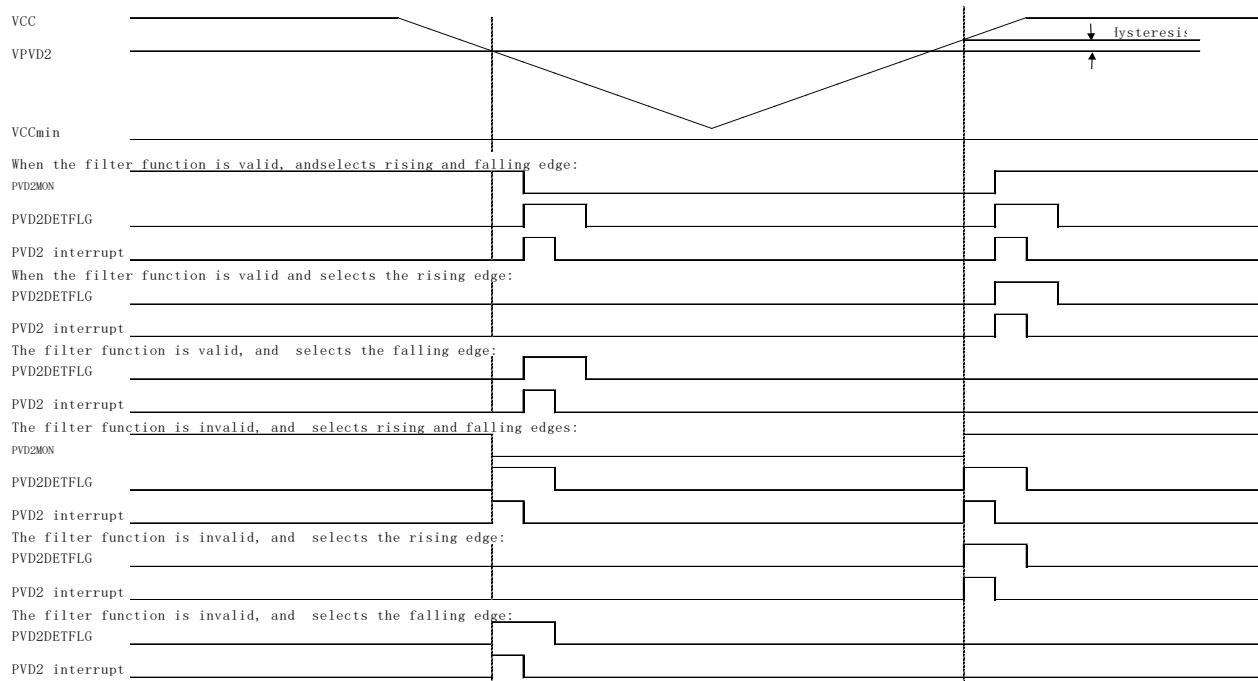


Figure 5-9 Power Voltage Detection 2 Interrupt Operation Timing Chart

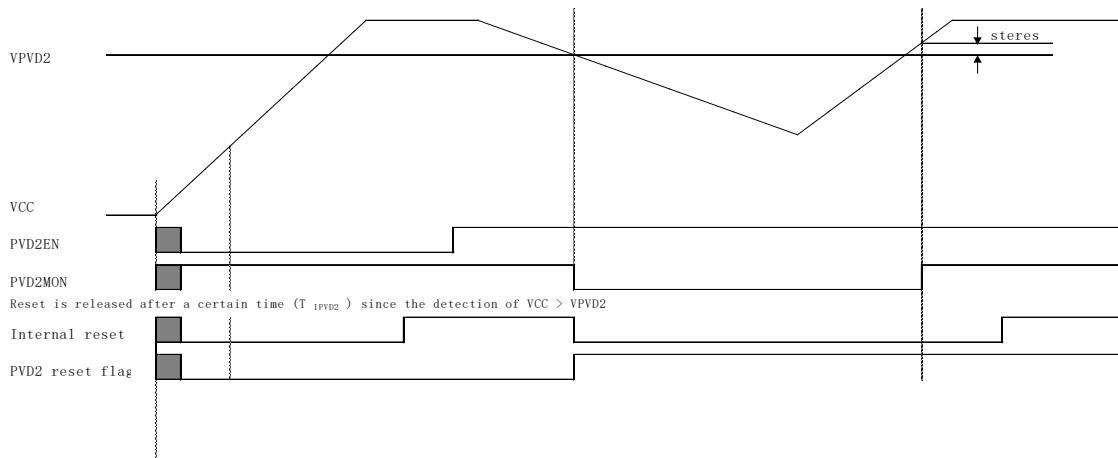


Figure 5-10 Power Voltage Detection 2 Reset Operation Timing Chart

5.3.8 Internal voltage sampling and detection function

The internal voltage sampling and detection functions of the chip include three functions: reference voltage measurement, VBAT voltage detection, and VBAT voltage measurement. The reference voltage measurement is the function of using the ADC to measure the reference voltage. The internal reference voltage is about 1.15V. VBAT voltage detection is the function of reading the register to obtain the VBAT voltage higher or lower than the VBAT detection reference voltage. The VBAT detection reference voltage can be selected by register to 1.8V or 2.1V. The VBAT voltage

measurement function refers to the function of using the ADC to measure the 1/2 voltage of VBAT to obtain the VBAT voltage value.

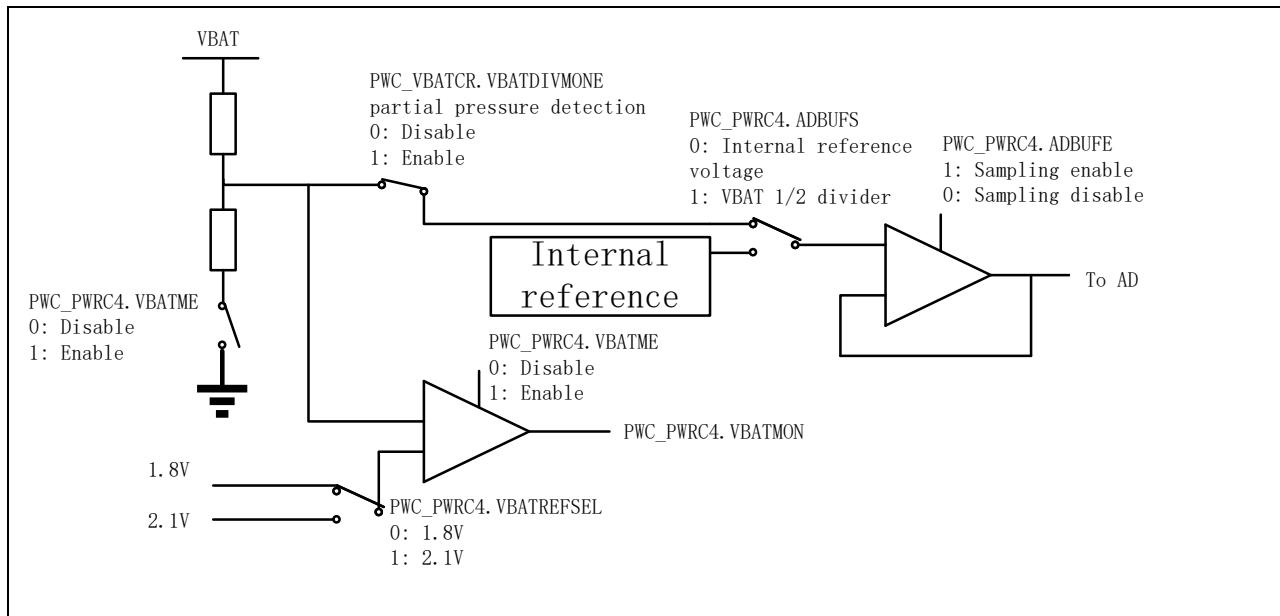


Figure 5-11 Schematic diagram of internal voltage sampling

■ Reference voltage measurement

When using the reference voltage measurement function, you need to set the registers according to the following steps.

- 1) PWC_PWRC4.ADBUFS=0, select the internal reference voltage
- 2) PWC_PWRC4.ADBUFE=1, enable the internal voltage measurement function
- 3) ADC_EXCHSELR. EXCHSEL=1, enable the ADC to select the internal reference voltage channel
- 4) Use ADC to measure internal reference voltage after waiting 50uS

■ VBAT voltage detection

When using the VBAT voltage detection function, you need to set the registers according to the following steps.

- 1) PWC_PWRC4.VBATREFSEL=1, select VBATREF=2.1V
PWC_PWRC4.VBATREFSEL=0, select VBATREF=1.8V
- 2) PWC_PWRC4.VBATME=1, enable VBAT voltage divider circuit
- 3) Wait 50uS
- 4) Read the PWC_PWRC4.VBATMON bit, when it is read as 0, VBAT>VBATREF. When read as 1, VBAT<VBATREF

■ VBAT voltage measurement

When using the VBAT voltage measurement function, you need to set the registers according to the following steps.

- 1) PWC_VBATCR.VBATDIVMONE=1, enable VBAT voltage measurement

- 2) PWC_PWRC4.VBATME=1, enable the VBAT voltage divider circuit
- 3) PWC_PWRC4.ADBUFS=1, select 1/2 voltage of VBAT
- 4) PWC_PWRC4.ADBUFE=1, enable the internal voltage measurement function
- 5) ADC_EXCHSEL.R. EXCHSEL=1, enable the ADC to select 1/2 voltage of VBAT
- 6) Use ADC to measure VBAT voltage after waiting 50uS
- 7) PWC_VBATCR.VBATDIVMONE=0, disable VBAT voltage measurement

5.4 Wakeup timer

The chip has integrated a wake-up timer WKTM. This counter can choose the internal low-speed oscillator for the RTC (RTCLRC) and the external low-speed oscillator (XTAL32) as the clock source. When the RTC uses the external low-speed oscillator as the clock source, the 64Hz internal clock signal can also be selected as the clock source of WKTM. This counter is an up-counting counter. After PWC_WKTC2.WKTCE is set, the counter starts to count. When the count value is equal to the WKTCMP[11:0] setting value of PWC_WKTC1 and PWC_WKTC0, the counter's value is cleared and re-counted, and a compare match event occurs at the same time. The compare match event, which can be used as a normal interrupt, can be used to wake-up from stop mode, and can be used to wake-up from power-down mode.

The WKTM register value is uncertain after power-on reset. After power-on reset, the VBAT domain reset register must be set first, and then PWC_WKTC0/PWC_WKTC1/PWC_WKTC2 must be set. Before starting the WKTM, make sure the count clock is on.

5.5 Operation Mode and Low Power Mode

After system reset or power-on reset, all power domains of the chip are in the power supply state, and the chip enters the high-speed run mode. In run mode, HCLK provides the CPU clock to execute instructions. The chip provides two operation modes: high-speed run mode and ultra-low-speed run mode. The chip can be configured in operating modes such as Table 5-3 shown.

In order to save the power consumption when the CPU does not need to keep running, the system provides three low-power modes such as sleep mode, stop mode, and power-down mode. The chip can be configured with low power modes such asTable 5-4 shown. In the sleep mode, the Cortex™ -M4F core of the chip stops working, and the peripherals keep running; in the stop mode, the Cortex™ -M4F core and peripherals of the chip stop working; in the power-down mode, the power supply of the VDD domain is powered off, and all modules of the VDD domain are powered down. The real-time clock and wake-up timer located in the battery backup domain can operate in low-power mode, and the backup SRAM can retain data.

It is up to the user to select the mode that gives the best compromise between low power consumption, short start-up time, available wake-up 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 below.

Table 5-3 Operation mode

Operating mode	Note
High Speed Run Mode	Main frequency below 240MHz
Ultra Low Speed Run Mode	Main frequency below 8MHz

Table 5-4 Low power mode

Pattern	Note
Sleep Mode	Cortex™ -M4F core clock stopped, peripherals kept running
Stop Mode	Cortex™ -M4F core and peripherals clocks stopped
Power-down mode	Power-down mode 1 (PD MD1)
	Power-down mode 2 (PD MD2)
	Power-down mode 3 (PD MD3)
	Power-down mode 4 (PD MD4)

Table 5-5 Low-power mode operating conditions and the status of each module in low-power mode

Item	Sleep mode	Stop mode	Power-down mode
Entry	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
Exit	Exit by any interrupt or reset.	Exit by interrupt or reset that can be used in stop mode	Exit by wake-up events or resets that can be used in power-down mode
External high-speed oscillator	Configurable	Stop	Stop
External low-speed oscillator	Configurable	Configurable	Configurable
Internal high-speed oscillator	Configurable	Stop	Power down
Internal medium speed oscillator	Configurable	Stop	Power down
Internal low-speed oscillator	Configurable	Configurable	Configurable
WDT dedicated clock oscillator	Configurable	Configurable	Power down
PLLA	Configurable	Stop	Power down
PLLH	Configurable	Stop	Power down
CPU	Stop (Retained).	Stop (Retained).	Power down
RAM	Configurable (Operating or power down)	Stop (Retained). Power down or sleep according to the settings before entering standby	Power down
Backup RAM	Configurable (Operating or power down or sleep)	Stop (Retained) Power down or sleep	Configurable (Power down or sleep)
Flash	Configurable	Stop (Retained)	Power down(Retained)
DMA	Configurable	Stop (Retained)	Power down
Voltage regulator	Operating (Drive adjustable)	Operating (Drive adjustable)	Stop
Power-on reset circuit	Operating	Operating	Operating Power-down mode 1, power-down mode 2 reset circuit accuracy can be guaranteed, power-down mode 3 and power-down mode 4 power-on reset voltage is not guaranteed
Brown-out reset BOR	Configurable	Configurable	Configurable (Power-down mode 1) Stop (Power Down Mode 2/3/4)
Voltage detection module PVD	Configurable	Configurable	Configurable (Power-down mode 1) Stop (Power Down Mode 2/3/4)
WDT	Configurable	Stop (Retained)	Power down
SWDT	Configurable	Configurable	Power down
RTC	Configurable	Configurable	Configurable
USB-FS	Configurable	Stop (Retained)	Power down
Timer0	Configurable	Configurable	Power down
Timer2	Configurable	Configurable	Power down

Item	Sleep mode	Stop mode	Power-down mode
WKTM	Configurable	Configurable	Configurable
Other peripheral modules	Configurable	Stop (Retained)	Power down
AD	Configurable	Stop	Power down
DA	Configurable	Configurable	Power down
PGA	Configurable	Configurable	Power down
CMP	Configurable	Configurable	Power down
PA0-PA10, PB0-PB2, PB5- PB13 , PC0-PC13, PD0-PD15, PE0-PE15, PH2 - PH15,PI0-PI12,PI14-15	Configurable	Retained	Retained or high impedance
PC14-PC15	Configurable	When used as the pin of the external low-speed oscillator, keep the oscillator action; when set as GPIO or other peripheral functions, please set and keep the two pins at the same level	When set to GPIO or other peripheral functions, the status of PC14 and PC15 can be set to hold or high impedance, please set to keep the two pins at the same level
PH0-PH1	Configurable	When used as an external high-speed oscillator, oscillator stops and the pin state keeps the state before entering STOP mode. When GPIO or other peripheral functions are set, keep the status before STOP	When used as an external high-speed oscillator, the oscillator stops, and the state of the pins remains in the state before entering the power-down mode; when set to GPIO or other peripheral functions, the state before the power-down mode is maintained
NRST reset pin	The chip is pulled up to VCC through a resistor	The chip is pulled up to VCC through a resistor	The chip is pulled up to VCC through a resistor
PA11-PA12, PB14- PB15	Configurable	Retained Since the level of this pin is pulled high, a current will be generated , when the USB STOP mode wake-up function is not used , disable pull-up when entering STOP mode , .	Retained or high resistance; Since redundant current will be generated when the level of this pin is pulled high, the pull-up is disabled when entering power-down mode
PI13/MD	Configurable	Retained	Retained The outside of the chip is connected to ground through a resistor
PA13-PA15, PB3, PB4	Work may be set up; When used as JTAG function, the built-in pull-up circuit is valid	Retained When used as JTAG function, the built-in pull-up circuit is valid	Retained When used as JTAG function, the built-in pull-up circuit is valid

5.5.1 Operating mode

The chip has two operating modes: high-speed and ultra-low-speed. According to the speed requirements of the system, set the best operating mode, so that the core voltage and drive capability are adapted to the system clock frequency, so as to achieve the purpose of reducing power consumption. Such as Table 5-6 shown, according to the DVS and DDAS bits of PWC_PWRC2, the chip can be operated in the corresponding operating mode. When the chip selection action is in the ultra-low speed mode, the FLASH and RAM also need to be set to work under the low core voltage, so it is necessary to set the register bits FRMC.LVM=1, PWC_RAMOPM=0x9062.

Table 5-6 Operation Modes

Operating mode	Frequency Range	Register setting		
		PWC_PWRC2.DVS	PWC_PWRC2.DDAS	PWC_PWRC3.DDAS
High-speed runmode	Below 240MHz	11	1111	11111111
Ultra-low speed run mode*	Below 8MHz	10	0000	00000000

* : overtake Low speed mode only supports T a=-40°C~ 85 °C

The switching between the high-speed run mode and the ultra-low-speed run mode needs to follow the following process 1 and process 2.

1. Switch from high-speed mode to ultra-low-speed mode

- 1) Set the timer used in ultra-low-speed mode to ensure that the timer meets the frequency requirements in ultra-low-speed mode
- 2) Turn off the timer and modules that are not required in ultra-low-speed mode, and confirm that Flash is not programmed or erased
- 3) Set FRMC.LVM=1 of FLASH, and set the RAM operation mode register PWC_RAMOPM to 0x9062
- 4) Confirm FRMC.LVM=1, PWC_RAMOPM=0x9062
- 5) Set PWC_PWRC2.DDAS[3:0] to 0000; set PWC_PWRC3.DDAS[7:0] to 0x00; PWC_PWRC2.DVS[1:0] is 10
- 6) Wait for TSWMD1 (30uS)
- 7) The chip operates in ultra-low speed mode

2. Switch from ultra low speed mode to high speed mode

- 1) Set PWC_PWRC2. DDAS[3:0] is 1111
PWC_PWRC3.DDAS[7:0] is 0xff;
PWC_PWRC2. DVS[1:0] is set to 11 according to the system frequency requirement
- 2) Waiting for TSWMD 2 (30uS)
- 3) Set FRMC.LVM=0 of FLASH, set RAM operation mode register PWC_RAMOPM to 0x8043
- 4) Confirm FRMC.LVM=0, PWC_RAMOPM=0x8043
- 5) The chip operates in high-speed mode

5.5.2 Sleep mode

In sleep mode, the Cortex™-M4F stops running and its internal register retains in sleep mode. The operating status of peripherals and other system modules other than the watchdog and dedicated watchdog will not change.

When the WDT is set to start automatically by ICG, if the WDTSLPOFF bit of the ICG is 1, the watchdog stops counting in sleep mode; if the WDTSLPOFF bit is 0, the watchdog continues to count in sleep mode. If the WDT is not set to start automatically and the watchdog is started by software, then if the WDT_CR.SLPOFF bit is 1, the watchdog stops counting in sleep mode; if the WDT_CR.SLPOFF bit is 0, the watchdog continues to count in sleep mode.

When the SWDT is set to start automatically by ICG, if the SWDTSLPOFF bit of the ICG is 1, the dedicated watchdog stops counting in sleep mode; if the SWDTSLPOFF bit is 0, the dedicated watchdog continues to count in sleep mode.

- Entering sleep mode

The sleep mode can be entered by executing the WFI instruction when PWC_STPMCR.STOP=0.

- Exiting sleep mode

Any interrupt and reset can wake the chip from sleep mode. When waking up by interrupt, the chip enters interrupt handling subroutine. The chip enters the reset state when exits the sleep mode by reset.

5.5.3 Stop mode

In stop mode, the Cortex™-M4F, most peripherals, and clock sources are stopped. The chip maintains Cortex™-M4F internal register and SRAM data, peripheral state and pin state. In the stop mode, because most of the clock sources stop working, the voltage regulator also reduces the drive capability, so the chip power consumption will be significantly reduced.

When the SWDT is set to start automatically by ICG, if the SWDTSLPOFF bit of the ICG is 1, the dedicated watchdog stops counting in stop mode; if the SWDTSLPOFF bit is 0, the dedicated watchdog continues to count in stop mode.

Before executing the WFI instruction to enter the stop mode, it is necessary to ensure that the FLASH is not in the programming or erasing state, and the oscillation stop detection function is invalid, otherwise the chip will enter the sleep mode instead of the stop mode.

In stop mode, the ADC and DAC also consume power unless disabled before entering stop mode. To disable DAC, you need to clear DACR.DAE, DAOE0, DAOE1 to "0". To disable the ADC, it is necessary to clear the ADC_STR.START bit to "0", set PWC_FCG3.ADC3, PWC_FCG3.ADC2, and PWC_FCG3.ADC1 as '1', and execute the WFI instruction to enter the stop mode.

When waking up from stop mode, select the clock after waking up and whether to wait for the Flash to stabilize through bits CKSMRC and FLNWT of the PWC_STPMCR register. CKSMRC is used to control the clock source after wake-up,

When CKSMRC = 1, the system clock source after wake-up is selected as MRC; when CKSMRC = 0, the system clock after wake-up maintains the same clock source before entering STOP. FLNWT is used to control whether to wait for the Flash to be stable after wake-up. When FLNWT=0, it needs to wait for the Flash to be stable when it wakes up; when FLNWT=1, it does not need to wait for the Flash to be stable when it wakes up. FLNWT must be set when the program is running on RAM, otherwise the behavior of the chip after waking up from STOP cannot be guaranteed. When the program runs on RAM and enters STOP mode, select CKSMRC = 1, FLNWT = 1 will wake up the system in the shortest time.

Before executing the WFI instruction to enter the stop mode, it is necessary to ensure that the DMA is in the stop state, otherwise the chip may perform unguaranteed actions.

The leakage current in STOP mode varies with voltage and temperatures, and the drive capability must meet the leakage demand 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.

The chip needs to set PWC_PWRC1.STPDAS=11 and then enter STOP mode in ultra-low speed mode; if PWC_PWRC1.STPDAS=00 and enter STOP mode, the chip will consume more power in STOP mode.

When the stop mode is set to exit through a non-maskable interrupt, the corresponding bit of INT_NMIER needs to be set; when the stop mode is set to exit through a maskable interrupt, the corresponding bit of the INT_WUPENR register needs to be set. Before executing the WFI or WFE instruction, make sure that all interrupts not used for wake-up from stop mode have been disabled.

- Entering stop mode

When PWC_STPMCR.STOP=1, PWC_PWRC0.PWDN=0, execute the WFI instruction to enter the stop mode. Table 5-5 shows the state of the chip's peripherals and clock sources in stop mode.

- Exiting stop mode

The stop mode can be exited by reset and interrupt. The reset that can be used to exit stop mode includes pin reset, power-on reset, brown-out reset (BOR), programmable voltage detection 1/2 reset, and dedicated watchdog reset. The interrupt events that can be used to exit the stop mode are as follows:

Pin interrupt EIRQ0-15, Voltage detection 1 interrupt,
Voltage Detect 2 interrupt, Dedicated watchdog underflow interrupt, Periodic interrupt of RTC , Alarm interrupt of RTC,
Wake-up timer interrupt, Comparator interrupt, USART1 RX interrupt, Timer0 compare match interrupt, Timer2 Compare match interrupt

When the chip exits the stop mode by interrupts, it starts the clock source which same as before entering the stop mode first. Then, after all of the clock sources stabilize, the chip

exits from the stop mode.

5.5.4 Power-down mode

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

When the SWDT is set to start automatically by ICG, if the SWDTSLOFF bit of the ICG is 1, the dedicated watchdog will be the same as other modules in the VDD domain, the power supply will be cut off and will no longer count. If the SWDTSLOFF bit is 0, the chip will enter stop mode instead of power-down mode, and the dedicated watchdog oscillator and dedicated watchdog will continue to run if auto-start is set in the ICG.

When the reset of voltage detection 1(PVD1) and voltage detection 2(PVD2) is enabled, the chip will enter stop mode instead of power-down mode.

Before executing the WFI instruction to enter the power-down mode, it is necessary to ensure that the FLASH is not in the programming or erasing state, and the oscillation stop detection function is invalid, otherwise the chip will enter the sleep mode instead of the power-down mode.

The capacitors used by 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 a 0.047uF or 0.1uF capacitor (the total capacity is 0.094uF or 0.2uF). 2) For chips with only VCAP_1 pin, 0.1uF or 0.22uF capacitors can be used. When waking up from power-down mode, VCAP_1/VCAP_2 needs to be charged during core voltage establishment. On the one hand, the smaller VCAP_1/VCAP_2 total capacity can shorten the charging time and bring fast response to the application; on the other hand, the larger VCAP_1/VCAP_2 total capacity will prolong the charging time, but also provide stronger electromagnetic compatibility (EMC). Users can choose smaller or larger capacitors according to the requirements of electromagnetic compatibility and system response speed. The total VCAP_1/VCAP_2 capacity of the chip must match the assignment of the PWC_PWRC3.PDTS bit. When the total capacity of VCAP_1/VCAP_2 is 0.2uF or 0.22uF, it is necessary to ensure that the PWC_PWRC3.PDTS bit is cleared before entering the power-down mode. When the total capacity of VCAP_1/VCAP_2 is 0.094uF or 0.1uF, it is necessary to ensure that the PWC_PWRC3.PDTS bit is set before entering the power-down mode.

The power consumption in power-down mode can be further reduced by setting PWC_PWRC0.PDMD[1:0]. Sub-modes of power-down mode are shown in the following Table 5-7. The voltage detection circuit can be used in power-down mode 1, and the power-on reset detection circuit is always active. So it does not need to wait for the stabilization of the VCC domain reference voltage, voltage detection circuit and power-on reset detection circuit while waking up. So it can not only achieve the purpose of low power consumption, but also wake up quickly. In power-down mode 2, the VCC domain reference voltage circuit and the voltage detection circuit stop working, but the power-on reset detection circuit is in an active state. So it is necessary to wait for the stabilization

time of the VCC domain reference voltage circuit and the voltage detection circuit while waking up. In power-down mode 3, the VCC domain reference voltage circuit, voltage detection circuit, and power-on reset detection circuit all stop working, and need to wait for these circuits to stabilize while waking up. Therefore, while achieving the lowest power consumption, the wake-up time is longer than that in power-down mode 2 and power-down mode 1. Power-down mode 4 stops the same circuits as power-down mode 3, so power-down mode 4 has the same power consumption as power-down mode 3. For specific power consumption and wake-up time, please refer to the electrical characteristics.

Table 5-7 Power-down mode sub-mode

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

The relationship between power consumption and wake-up time: $IPD1 > IPD2 > IPD3 = IPD4$, $TPD1 < TPD2 < TPD4 < TPD3$

- Entering power-down mode

When PWC_STPMCR.STOP=1, PWC_PWRC0.PWDN=1, execute the WFI instruction to enter the power-down mode.

- Exiting power-down mode

Power-down mode can be exited by a power-down mode wake-up event or reset. The resets that can be used to wake up from power-down mode are pin reset, power-on reset, and voltage detect 0 reset. Events that can be used for wake-up from Power-down include:

WKUPn_0/1/2/3 (n=0/1/2/3) wake-up events, real-time clock alarms and timing events, voltage detection 1 Wake event, voltage detection 2 wakeup event, wakeup timer wakeup event

After waking up from power-down mode 1 or power-down mode 2, the chip resets and executes the program again. The wake-up event can be queried by the power-down wake-up flag registers, and the reset flag can be queried by RMU_RSTF0.PDRF.

In power-down mode 3, the POR, PDR, BOR, PVD1, and PVD2 circuits are all inactive. After waking up from power-down mode 3, all registers are reset like power on mode except PWC_PDWF0/PWC_PDWF1/RSTF0; reset flag can be queried through RMU_RSTF0.PDRF.

In power-down mode 4, the POR, PDR, BOR, PVD1, and PVD2 circuits are all inactive, and the program is re-executed after the chip is reset; the reset flag can be queried through RMU_RSTF0.PDRF.

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 selection register (PWC_PDWKES). When a power-down wake-up event occurs, the corresponding power-down

wake-up flag (in registers PWC_PDWKF0-PWC_PDWKF1) will be set. After power-down wake-up, if the power-down wake-up flag is not cleared, the chip cannot enter power-down mode again. The edge of the power-down wake-up event can be selected by PWC_PDWKES.

When the power-down mode wakes up, the VDD domain will be powered again, the system will perform a power-down wake-up reset, and the internal medium-speed oscillator will work as a start up clock. The registers that are not reset by wake-up reset are as follows.

Power-down mode	Registers that are not reset
Power-down mode 1	PWC_PWRC0 PWC_PWRC1 PWC_PWRC3 PWC_PDWK_E0
Power-down mode 2	PWC_PDWKE1 PWC_PDWKE2
Power-down mode 4	PWC_PDWKES PWC_PDWKF0 PWC_PDWKF1 PWC_PVDCR0 PWC_PVDCR1 PWC_PVDFCR PWC_PVDLCR PWC_PVDICR PWC_PVDDSR
Power-down mode 3	PWC_PDWKF0 PWC_PDWKF1 RMU_RSTF0

■ Pin status after exiting power-down mode

In the power-down mode, according to the register settings, the chip pins will keep the state before entering the power-down mode or will be the high-impedance state. If PWC_PWRC0.IORTN[1:0]=10 or 11, the pin state is a high-impedance state in the power-down mode, and the pin is initialized as high-impedance after exiting the power-down mode. If PWC_PWRC0.IORTN[1:0]=00, the pin state in the power-down mode remains the state before entering the power-down mode, and the pin is initialized as high-impedance after wake-up. If PWC_PWRC0.IORTN[1:0]=01, the pins will keep the state before entering the power-down mode, and the state of the pins will not change even if the peripherals or the registers of the pins are set after waking up. After the PWC_PWRC0.IORTN is cleared by software, the pin state is controlled by the peripheral or the setting of the pin's register.

■ PTWK power-down mode wake-up event

PTWK is used to wake up the chip from power-down mode. Through software setting, each PTWK event can be selected from 4 pins, and the rising or falling trigger edge can be configured, and it has an independent flag. The structure diagram of PTWKn is as follows Figure 5-12 shown.

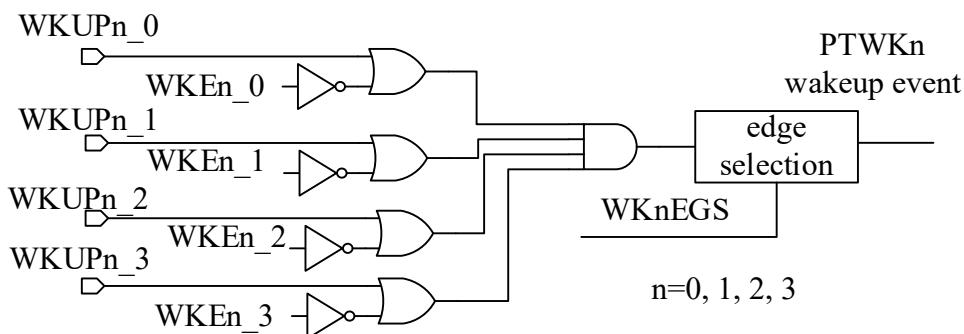


Figure 5-12 PTWKn block diagram

■ Action description of battery backup domain in power-down mode

In the battery backup domain, after the chip enters the power-down mode, the backup RAM continues to supply power through RLDO, and the RTC/WKTM/backup registers continue to supply power through VCC, so the RTC/WKTM/Backup-SRAM can continue to operate or retain data. The VDDR domain is powered by LDO after power-down wake-up. Before entering power-down mode, the VBAT voltage measurement function needs to be turned off.

■ Methods of reducing power consumption

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

- 1) Set the optimum operating mode
- 2) Reduce system clock speed
- 3) Turn off unused clock source
- 4) Set the function clock control registers PWC_FCGn (n=0/1/2/3) to close the functions that do not need to be used
- 5) Power down the unused RAM

5.5.5 Reduce system clock speed

In run mode, the system clock (HCLK), external bus clock (EXCLK), peripheral clocks PCLK0/PCLK1/PCLK2/PCLK3/PCLK4 can be slowed down by setting the prescaler registers. These pre-dividers can also be used to reduce the speed of peripherals before entering sleep mode. For details, please refer to [Clock Controller CMU].

5.5.6 Turn off unused clock source

The system of the chip has six clock sources :

- External high-speed oscillator (XTAL)
- External low-speed oscillator (XTAL32)
- PLLH clock (PLLH)
- Internal high-speed oscillator (HRC)
- Internal Medium Speed Oscillator (MRC)
- Internal low-speed oscillator (LRC)

SWDT has an independent dedicated internal low-speed oscillator (SWDTLRC); RTC can choose external low-speed oscillator or internal low-speed oscillator for RTC (RTCLRC) as the clock source. Both HRC and PLL are equipped with independent power supply circuits. After HRC is turned off, the power of HRC can be cut off by setting the PWC_PWRC1.VHRCSD bit to further reduce power consumption; After both PLLA and PLLH are turned off, the power can be cut off by setting the register PWC_PWRC1.VPLLSD=11.

For details, please refer to [Clock Controller CMU].

5.5.7 Stop functional clock

The peripheral module of the chip is provided with the function clock stop function. By setting the corresponding register bit, the clock of the corresponding module can be stopped to reduce the power consumption. In the module stop state, the register inside the module will remain in the state before it stops.

5.5.8 Turn off unused RAM

Each RAM module in the chip is configured with a function clock stop bit and a power-down control bit. The RAM clock will be stopped by setting the module stop bit, thereby reducing power consumption. By setting the power-down control bit of the module, the corresponding RAM module can be powered down, thereby reducing power consumption. Table 5-8 is the corresponding relationship between the RAM module and the power-down control bit. By setting the RAMPDCn (n=0-10) bit of the PWC_RAMPC0 register or the VBTRSD bit of the PWC_VBATCR register can power down the corresponding RAM.

Table 5-8 RAM module and RAM power-down control bits

RAM module	Note	Power-down control bits
SRAM1	0x20000000~ 0x20000FFFF RAM for system	PWC_RAMPC0.RAMPDC0
	0x20010000~ 0x20001FFFF RAM for system	PWC_RAMPC0.RAMPDC1
SRAM2	0x20020000~0x2002FFFF RAM for system	PWC_RAMPC0.RAMPDC2
	0x20030000 ~0x2003FFFF RAM for system	PWC_RAMPC0.RAMPDC3
SRAM3	0x20040000~0x200 4 FFFF RAM for system	PWC_RAMPC0.RAMPDC4
	0x20050000~0x200 57 FFF RAM for system	PWC_RAMPC0.RAMPDC5
SRAM4	0x20058000~0x200 5F FFF RAM for system	PWC_RAMPC0.RAMPDC6
SRAMH	0x1FFE0000~0x1FFE7FFF RAM for system	PWC_RAMPC0.RAMPDC7
	0x1FFE8000~0x1FFEFFFF RAM for system	PWC_RAMPC0.RAMPDC8
	0x1FFF0000~0x1FFF7FFF RAM for system	PWC_RAMPC0.RAMPDC9
	0x1FFF8000~0x1FFFFFFF RAM for system	PWC_RAMPC0.RAMPDC10
Backup -SRAM	0x200F0000~0x200F0FFF RAM for system	PWC_VBATCR. VBTRSD
CAN_1 RAM	RAM for CAN unit 1	PWC_PRAMLPC.PRAMDC0
CAN_2 RAM	RAM for CAN unit 2	PWC_PRAMLPC.PRAMDC1
CACHERAM	RAM for Cache	PWC_PRAMLPC.PRAMDC2
USBFS RAM	RAM for USBFS	PWC_PRAMLPC.PRAMDC3
USBHS RAM	RAM for USBHS	PWC_PRAMLPC.PRAMDC4
ETHMAC TX RAM	RAM for ETHMAC TX	PWC_PRAMLPC.PRAMDC5
ETHMAC RX RAM	RAM for ETHMAC RX	PWC_PRAMLPC.PRAMDC6

RAM module	Note	Power-down control bits
SDIOC1 RAM	RAM for SDIOC1	PWC_PRAMLPC.PRAMDC7
SDIOC2 RAM	RAM for SDIOC2	PWC_PRAMLPC.PRAMDC8
NFCRAM	RAM for NFC	PWC_PRAMLPC.PRAMDC9

5.6 Register protection function

Register protection is used to prevent the write operation of the register from accidental overwriting. Table 5-9 is a list of relationship between register protection bits and protected registers.

Table 5-9 Protected register list

Register protection bit	Protected register
PWC_FPRC.FPRCB0	CMU_XTALCFGR, CMU_XTALSTBCR, CMU_XTALCR, CMU_XTALSTDRCR, CMU_XTALSTDTR, CMU_HRCTRM, CMU_HRCCR, CMU_MRCTR, CMU_MRCCR, CMU_PLLHCFGR, CMU_PLLHCR, CMU_PLLACFGR, CMU_PLLACR, CMU_OSCSTBSR, CMU_CKSWR, CMU_SCFGR, CMU_USBCKCFG, CMU_CANCKCFG, CMU_TPIUCKCFG, CMU_M_C01CFG, CMU_MCO2CFG, CMU_XTAL32CR, CMU_XTALC32CFG, CMU_XTAL32NFR, CMU_LRCCR, CMU_LRCTR, CMU_RTCLRCTR
PWC_FPRC.FPRCB1	PWC_PWRC0, PWC_PWRC1, PWC_PWRC2, PWC_PWRC3, PWC_PWRC4, PWC_PDWKE0, PWC_PDWKE1, PWC_PDWKE2, PWC_PDWKES, PWC_PDWKF0, PWC_PDWKF1, CMU_PERICKSEL, CMU_I2SCKSEL, PWC_STPMCR, PWC_RAMPC0, PWC_RAMOPM, PWC_PRAMLPC, RMU_RSTF0, PWC_VBATRSTR, PWC_BATCR0, PWC_WKTC0, PWC_WKTC1, PWC_WKTC2, RMU_PRSTCR0
PWC_FPRC.FPRCB3	PWC_PVDCR0, PWC_PVDCR1, PWC_PVDFCR, PWC_PVDLCR, PWC_PVDICR, PWC_PVDDSR

Register protection bit	Protected register
PWC_FCG0PC.B0	PWC_FCG0

5.7 Register description

The list of registers is as follows Table 5-10 shown.

Table 5-10 List of Registers

BASE ADDR: 0x4004CC00

Register name	Symbol	Offset address	Bit width	Reset value
Power Mode Control Register 0	PWC_PWRC0	0x00	8	0x00
Power Mode Control Register 1	PWC_PWRC1	0x04	8	0x00
Power Mode Control Register 2	PWC_PWRC2	0x08	8	0xFF
Power Mode Control Register 3	PWC_PWRC3	0x0C	8	0xFF
Power Mode Control Register 4	PWC_PWRC4	0x10	8	0x00
PVD Control Register 0	PWC_PVDCR0	0x14	8	0x00
PVD Control Register 1	PWC_PVDCR1	0x18	8	0x00
PVD Filter Control Register	PWC_PVDFCR	0x1C	8	0x11
PVD Level Control Register	PWC_PVDLCR	0x20	8	0x00
Power-down Wake-up Enable Register 0	PWC_PDWKE0	0x28	8	0x00
Power-down Wake-up Enable Register 1	PWC_PDWKE1	0x2C	8	0x00
Power-down Wake-up Enable Register 2	PWC_PDWKE2	0x30	8	0x00
Power-down Wake-up Event Edge Select Register	PWC_PDWKES	0x34	8	0x00
Power-down Wake-up Flag Register 0	PWC_PDWKF0	0x38	8	0x00
Power-down Wake-up Flag Register 1	PWC_PDWKF1	0x3C	8	0x00
RAM Power Control Register	PWC_RAMPC0	0xE0	32	0x00000000
RAM Operating Condition Register	PWC_RAMOPM	0xE4	16	0x8043
Peripheral RAM Low Power Control Register	PWC_PRAMLPC	0xE8	32	0x00000000
PVD Interrupt Control Register	PWC_PVDICR	0xF0	8	0x00
PVD Detection Status Register	PWC_PVDDSR	0xF4	8	0x11

BASE ADDR: 0x40054000

Register name	Symbol	Offset address	Bit width	Reset value
STOP Mode Control Register	PWC_STPMCR	0x0C	16	0x0000
Function Protection Control Register	PWC_FPRC	0x3FE	8	0x00

BASE ADDR: 0x4004C400

Register name	Symbol	Offset address	Bit width	Reset value
Backup Domain Reset Register	PWC_VBATRSTR	0x30	8	0x00
Backup Domain Control Register	PWC_VBATCR	0x40	8	0x00
Wake-up Timer Control Register 0	PWC_WKTC0	0x50	8	0x00
Wake-up Timer Control Register 1	PWC_WKTC1	0x54	8	0x00
Wake-up Timer Control Register 2	PWC_WKTC2	0x58	8	0x00

Register name	Symbol	Offset address	Bit width	Reset value
VBAT Backup Registers 0~127	PWC_BKR000	0x200	8	0x00
	PWC_BKR001	0x204	8	0x00
	PWC_BKR002	0x208	8	0x00
	PWC_BKR003	0x20C	8	0x00
	PWC_BKR004	0x210	8	0x00
	PWC_BKR005	0x214	8	0x00
	PWC_BKR006	0x218	8	0x00
	PWC_BKR007	0x21C	8	0x00
	PWC_BKR008	0x220	8	0x00
	PWC_BKR009	0x224	8	0x00
	PWC_BKR010	0x228	8	0x00
	PWC_BKR011	0x22C	8	0x00
	PWC_BKR012	0x230	8	0x00
	PWC_BKR013	0x234	8	0x00
	PWC_BKR014	0x238	8	0x00
	PWC_BKR015	0x23C	8	0x00
	PWC_BKR016	0x240	8	0x00
	PWC_BKR017	0x244	8	0x00
	PWC_BKR018	0x248	8	0x00
	PWC_BKR019	0x24C	8	0x00
	PWC_BKR020	0x250	8	0x00
	PWC_BKR021	0x254	8	0x00
	PWC_BKR022	0x258	8	0x00
	PWC_BKR023	0x25C	8	0x00
	PWC_BKR024	0x260	8	0x00
	PWC_BKR025	0x264	8	0x00
	PWC_BKR026	0x268	8	0x00
	PWC_BKR027	0x26C	8	0x00
	PWC_BKR028	0x270	8	0x00
	PWC_BKR029	0x274	8	0x00
	PWC_BKR030	0x278	8	0x00
	PWC_BKR031	0x27C	8	0x00
	PWC_BKR032	0x280	8	0x00
	PWC_BKR033	0x284	8	0x00
	PWC_BKR034	0x288	8	0x00
	PWC_BKR035	0x28C	8	0x00
	PWC_BKR036	0x290	8	0x00
	PWC_BKR037	0x294	8	0x00

Register name	Symbol	Offset address	Bit width	Reset value
	PWC_BKR038	0x298	8	0x00
	PWC_BKR039	0x29C	8	0x00
	PWC_BKR040	0x2A0	8	0x00
	PWC_BKR041	0x2A4	8	0x00
	PWC_BKR042	0x2A8	8	0x00
	PWC_BKR043	0x2AC	8	0x00
	PWC_BKR044	0x2B0	8	0x00
	PWC_BKR045	0x2B4	8	0x00
	PWC_BKR046	0x2B8	8	0x00
	PWC_BKR047	0x2BC	8	0x00
	PWC_BKR048	0x2C0	8	0x00
	PWC_BKR049	0x2C4	8	0x00
	PWC_BKR050	0x2C8	8	0x00
	PWC_BKR051	0x2CC	8	0x00
	PWC_BKR052	0x2D0	8	0x00
	PWC_BKR053	0x2D4	8	0x00
	PWC_BKR054	0x2D8	8	0x00
	PWC_BKR055	0x2DC	8	0x00
	PWC_BKR056	0x2E0	8	0x00
	PWC_BKR057	0x2E4	8	0x00
	PWC_BKR058	0x2E8	8	0x00
	PWC_BKR059	0x2EC	8	0x00
	PWC_BKR060	0x2F0	8	0x00
	PWC_BKR061	0x2F4	8	0x00
	PWC_BKR062	0x2F8	8	0x00
	PWC_BKR063	0x2FC	8	0x00
	PWC_BKR064	0x300	8	0x00
	PWC_BKR065	0x304	8	0x00
	PWC_BKR066	0x308	8	0x00
	PWC_BKR067	0x30C	8	0x00
	PWC_BKR068	0x310	8	0x00
	PWC_BKR069	0x314	8	0x00
	PWC_BKR070	0x318	8	0x00
	PWC_BKR071	0x31C	8	0x00
	PWC_BKR072	0x320	8	0x00
	PWC_BKR073	0x324	8	0x00
	PWC_BKR074	0x328	8	0x00
	PWC_BKR075	0x32C	8	0x00

Register name	Symbol	Offset address	Bit width	Reset value
	PWC_BKR076	0x330	8	0x00
	PWC_BKR077	0x334	8	0x00
	PWC_BKR078	0x338	8	0x00
	PWC_BKR079	0x33C	8	0x00
	PWC_BKR080	0x340	8	0x00
	PWC_BKR081	0x344	8	0x00
	PWC_BKR082	0x348	8	0x00
	PWC_BKR083	0x34C	8	0x00
	PWC_BKR084	0x350	8	0x00
	PWC_BKR085	0x354	8	0x00
	PWC_BKR086	0x358	8	0x00
	PWC_BKR087	0x35C	8	0x00
	PWC_BKR088	0x360	8	0x00
	PWC_BKR089	0x364	8	0x00
	PWC_BKR090	0x368	8	0x00
	PWC_BKR091	0x36C	8	0x00
	PWC_BKR092	0x370	8	0x00
	PWC_BKR093	0x374	8	0x00
	PWC_BKR094	0x378	8	0x00
	PWC_BKR095	0x37C	8	0x00
	PWC_BKR096	0x380	8	0x00
	PWC_BKR097	0x384	8	0x00
	PWC_BKR098	0x388	8	0x00
	PWC_BKR099	0x38C	8	0x00
	PWC_BKR100	0x390	8	0x00
	PWC_BKR101	0x394	8	0x00
	PWC_BKR102	0x398	8	0x00
	PWC_BKR103	0x39C	8	0x00
	PWC_BKR104	0x3A0	8	0x00
	PWC_BKR105	0x3A4	8	0x00
	PWC_BKR106	0x3A8	8	0x00
	PWC_BKR107	0x3AC	8	0x00
	PWC_BKR108	0x3B0	8	0x00
	PWC_BKR109	0x3B4	8	0x00
	PWC_BKR110	0x3B8	8	0x00
	PWC_BKR111	0x3BC	8	0x00
	PWC_BKR112	0x3C0	8	0x00
	PWC_BKR114	0x3C4	8	0x00

Register name	Symbol	Offset address	Bit width	Reset value
	PWC_BKR115	0x3C8	8	0x00
	PWC_BKR116	0x3CC	8	0x00
	PWC_BKR117	0x3D0	8	0x00
	PWC_BKR118	0x3D4	8	0x00
	PWC_BKR119	0x3D8	8	0x00
	PWC_BKR120	0x3DC	8	0x00
	PWC_BKR121	0x3E0	8	0x00
	PWC_BKR122	0x3E4	8	0x00
	PWC_BKR123	0x3E8	8	0x00
	PWC_BKR124	0x3EC	8	0x00
	PWC_BKR125	0x3F0	8	0x00
	PWC_BKR126	0x3F4	8	0x00
	PWC_BKR127	0x3F8	8	0x00

BASE ADDR: 0x40048000

Register name	Symbol	Offset address	Bit width	Reset value
Function Clock Control 0	PWC_FCG0	0x00	32	0xFFFFFA0E
Function Clock Control 1	PWC_FCG1	0x04	32	0xFFFFFFFF
Function Clock Control 2	PWC_FCG2	0x08	32	0xFFFFFFFF
Function Clock Control 3	PWC_FCG3	0x0C	32	0xFFFFFFFF
PWC_FCG0 Protection Control	PWC_FCG0PC	0x10	32	0x00000000

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]		-	-	PDMDS[1:0]	
Bit	Symbol	Bit name	Description			Read and write	
B7	PWDN	Power-down mode control bits	0: Power-down mode disable 1: Power-down mode enable			R/W	
b6	Reserved	-	Read as "0", write as "0"			R/W	
b5-b4	IORTN[1:0]	IO state hold control in power-down mode	00: In power-down mode, IO holds state. Hardware will release IO hold state after power-down wake-up 01: In power-down mode, IO remains in the state. After power-down wake-up, set IORTN [1:0] as 00b will release the IO hold state 1x: In power-down mode and power-down wake-up, the IO is high impedance			R/W	
b3-b2	Reserved	-	Read as "0", write as "0"			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]		-	PDTS	VHRCSD	VPLLSD[1:0]	
Bit	Symbol	Bit name	Description			Read and write	
b7-b6	STPDAS[1:0]	STOP mode LDO driver selection	00: The drive capacity set when entering STOP mode from high-speed mode 11: The drive capacity set when entering STOP from the ultra-low speed mode 10/01: Settings are prohibited.			R/W	
b5-b4	Reserved	-	Read as "0", write as "0"			R/W	
b3	PDTS	Power-down wake-up time control bits	0: When the total capacitance of VCAP_1/VCAP_2 is two 0.1uF or one 0.22uF, the PDTS bit must be cleared before entering the power-down mode. 1: When the total capacitance of VCAP_1/VCAP_2 is two 0.047uF or one 0.1uF, the PDTS bit must be set before entering power-down mode.			R/W	
b2	VHRCSD	HRC power control	0: HRC power on 1: HRC power down When the HRC is not in use, set VHRCSD to cut off the power for the HRC to further reduce power consumption. After VHRCSD is cleared, wait 25us before starting the HRC module			R/W	
b1-b0	VPLLSD[1:0]	PLL power control	00: PLL power on 11: PLL power down 01: Setting is prohibited 10: Setting is prohibited After PLLA and PLLH are turned off, wait for 50us before setting VPLLSD=11 to cut off the power for PLL. Power consumption will be further reduced. After setting VPLLSD=00, you need to wait 25uS before starting the PLLA or PLLH 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[11:8]		

Bit	Symbol	Bit name	Description	Read and write
b7-b6	Reserved	-	Read as "1", write as "1"	R/W
b5-b4	DVS[1:0]	Voltage selection in operation mode	10: Select ultra-low speed operating voltage 11: Select high-speed operation mode voltage 00/01: Settings are prohibited.	R/W
b3-b0	DDAS[11:8]	Power driver selection	Combined with DDAS[7:0] in PWC_PWRC3 to adjust the drive capability DDAS[11:0]= 0xffff: High-speed action mode 0x000: Low-speed drive mode others: Settings are prohibited.	R/W

5.7.4 Power Mode Control Register 3 (PWC_PWRC3)

Reset value: 0xFF

B7	b6	b5	b4	b3	b2	b1	b0
DDAS[7:0]							

Bit	Symbol	Bit name	Description	Read and write
b7-b0	DDAS[7:0]	Power driver selection	Combined with DDAS[11:8] in PWC_PWRC2 to adjust the drive capability DDAS[11:0]= 0xffff: High-speed action mode 0x000: Ultra-low speed drive mode others: Settings are prohibited	R/W

5.7.5 Power Mode Control Register 4 (PWC_PWRC4)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
ADBUFE	ADBUFS	-	VBATMON	-	-	VBATME	VBATREFSEL

Bit	Symbol	Bit name	Description	Read and write
B7	ADBUFE	Internal sampling voltage enable	When using ADC to sample the internal voltage of the chip, you need to set this bit to 1 0: disable 1: enable	R/W
b6	ADBUFS	Internal sampling voltage selection	0: Internal reference voltage 1: 1/2 of VBAT voltage	R/W
b5	Reserved	-	Read as "0", write as "0"	R/W
b4	VBATMON	VBAT voltage detection status	0: VBAT > VBATREF 1: VBAT < VBATREF	R
b3-b2	Reserved	-	"0" when reading, "0" when writing, prohibit writing "1"	R/W
b1	VBATME	VBAT detection enabled	0: Disable detection 1: Enable detection	R/W
b0	VBATREFSEL	VBAT detection reference voltage (VBATREF) selection	0: 1.8V 1: 2.1V	R/W

5.7.6 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

Bit	Symbol	Bit name	Description	Read and write
B7	WKE13	WKUP1_3 wake-up event enable	0: WKUP1_3 wake-up event disable 1: WKUP1_3 wake-up event enable	R/W
b6	WKE12	WKUP1_2 wake-up event enable	0: WKUP1_2 wake-up event disable 1: WKUP1_2 wake-up event enable	R/W
b5	WKE11	WKUP1_1 wake-up event enable	0: WKUP1_1 wake-up event disable 1: WKUP1_1 wake-up event enable	R/W
b4	WKE10	WKUP1_0 wake-up event enable	0: WKUP1_0 wake-up event disable 1: WKUP1_0 wake-up event enable	R/W
b3	WKE03	WKUP0_3 wake-up event enable	0: WKUP0_3 wake-up event disable 1: WKUP0_3 wake-up event enable	R/W
b2	WKE02	WKUP0_2 wake-up event enable	0: WKUP0_2 wake-up event disable 1: WKUP0_2 wake-up event enable	R/W
b1	WKE01	WKUP0_1 wake-up event enable	0: WKUP0_1 wake-up event disable 1: WKUP0_1 wake-up event enable	R/W
b0	WKE00	WKUP0_0 wake-up event enable	0: WKUP0_0 wake-up event disable 1: WKUP0_0 wake-up event enable	R/W

5.7.7 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

Bit	Symbol	Bit name	Description	Read and write
B7	WKE33	WKUP3_3 wake-up event enable	0: WKUP3_3 wake-up event disable 1: WKUP3_3 wake-up event enable	R/W
b6	WKE32	WKUP3_2 wake-up event enable	0: WKUP3_2 wake-up event disable 1: WKUP3_2 wake-up event enable	R/W
b5	WKE31	WKUP3_1 wake-up event enable	0: WKUP3_1 wake-up event disable 1: WKUP3_1 wake-up event enable	R/W
b4	WKE30	WKUP3_0 wake-up event enable	0: WKUP3_0 wake-up event disable 1: WKUP3_0 wake-up event enable	R/W
b3	WKE23	WKUP2_3 wake-up event enable	0: WKUP2_3 wake-up event disable 1: WKUP2_3 wake-up event enable	R/W
b2	WKE22	WKUP2_2 wake-up event enable	0: WKUP2_2 wake-up event disable 1: WKUP2_2 wake-up event enable	R/W
b1	WKE21	WKUP2_1 wake-up event enable	0: WKUP2_1 wake-up event disable 1: WKUP2_1 wake-up event enable	R/W
b0	WKE20	WKUP2_0 wake-up event enable	0: WKUP2_0 wake-up event disable 1: WKUP2_0 wake-up event enable	R/W

5.7.8 Power-down Wake-up Enable Register 2 (PWC_PDWKE2)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
WKTMWKE	-	RTCALMWKE	RTCPRDWKE	-	-	VD2WKE	VD1WKE

Bit	Symbol	Bit name	Description	Read and write
B7	WKTMWKE	WKT M wake-up event enable	0: WKT M wake-up event disable 1: WKT M wake-up event enable	R/W
b6	Reserved	-	Read as "0", write as "0"	R/W
b5	RTCALMWKE	RTC alarm wake-up event enable	0: RTC alarm wake-up event disable 1: RTC alarm wake-up event enable	R/W
b4	RTCPRDWKE	RTC periodic wake-up event enable	0: RTC periodic wake-up event disable 1: RTC periodic wake-up event enable	R/W
b3-b2	Reserved	-	Read as "0", write as "0"	R/W
b1	VD2WKE	PVD2 wake-up event enable	0: PVD2 wake-up event disable 1: PVD2 wake-up event enable	R/W
b0	VD1WKE	PVD1 wake-up event enable	0: PVD1 wake-up event disable 1: PVD1 wake-up event enable	R/W

5.7.9 Power-Down Wake-Up Event Edge Select Register (PWC_PDWKES)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
-	-	VD2EGS	VD1EGS	WK3EGS	WK2EGS	WK1EGS	WKOEGS

Bit	Symbol	Bit name	Description	Read and write
b7-b6	Reserved	-	Read as "0", write as "0"	R/W
b5	VD2EGS	VD2 edge select	0: VCC < V _{PVD2} 1: VCC > V _{PVD2}	R/W
b4	VD1EGS	VD1 edge select	0: VCC < V _{PVD1} 1: VCC > V _{PVD1}	R/W
b3	WK3EGS	PTWK3 edge select	0: Falling edge 1: Rising edge	R/W
b2	WK2EGS	PTWK2 edge select	0: Falling edge 1: Rising edge	R/W
b1	WK1EGS	PTWK1 edge select	0: Falling edge 1: Rising edge	R/W
b0	WKOEGS	PTWK0 edge select	0: Falling edge 1: Rising edge	R/W

5.7.10 Power-down Wake-up Flag Register 0 (PWC_PDWKF0)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
-	-	VD2WKF	VD1WKF	PTWK3F	PTWK2F	PTWK1F	PTWK0F

Bit	Symbol	Bit name	Description	Read and write
b7-b6	Reserved	-	Read as "0", write as "0"	R/W
b5	VD2WKF	PVD2 wake-up flag	0: No PVD2 wake-up event occurred 1: PVD2 wake-up event occurred After power-down wake-up, you need to write zero to clear this bit.	R/W
b4	VD1WKF	PVD1 wake-up flag	0: No PVD1 wake-up event occurred 1: PVD1 wake-up event occurred After power-down wake-up, you need to write zero to clear this bit.	R/W
b3	PTWK3F	PTWK3 wake-up flag	0: No PTWK3 wake-up event occurred 1: PTWK3 wake-up event occurred After power-down wake-up, you need to write zero to clear this bit.	R/W
b2	PTWK2F	PTWK2 wake-up flag	0: No PTWK2 wake-up event occurred 1: PTWK2 wake-up event occurred After power-down wake-up, you need to write zero to clear this bit.	R/W
b1	PTWK1F	PTWK1 wake-up flag	0: No PTWK1 wake-up event occurred 1: PTWK1 wake-up event occurred After power-down wake-up, you need to write zero to clear this bit.	R/W
b0	PTWK0F	PTWK0 wake-up flag	0: No PTWK0 wake-up event occurred 1: PTWK0 wake-up event occurred After power-down wake-up, you need to write zero to clear this bit.	R/W

5.7.11 Power-down Wake-up Flag Register 1 (PWC_PDWKF1)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
WKTMWKF	-	RTCALMWKF	RTCPRDWKF	-	-	-	

Bit	Symbol	Bit name	Description	Read and write
B7	WKTMWKF	WKTM wake-up flag	0: No WKT M wake-up event occurred 1: WKT M wake-up event occurred After power-down wake-up, you need to write zero to clear this bit.	R/W
b6	Reserved	-	Read as "0", write as "0"	R/W
b5	RTCALMWKF	RTC alarm wake-up flag	0: No RTC alarm wake-up event occurred 1: RTC alarm wake-up event occurred After power-down wake-up, you need to write zero to clear this bit.	R/W
b4	RTCPRDWKF	RTC periodic wake-up flag	0: No RTC periodic wake-up event occurred 1: RTC periodic wake-up event occurred After power-down wake-up, you need to write zero to clear this bit.	R/W
b3~0	Reserved	-	Read as "0", write as "0"	R/W

5.7.12 Function Clock Control 0 (PWC_FCG0)

Reset value: 0xFFFFFA0E

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DCU8	DCU7	DCU6	DCU5	DCU4	DCU3	DCU2	DCU1	CRC	TRNG	HASH	AES	MAU	CTC	AOS	FCM
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
DMA2	DMA1	KEY	-	-	SRAMB	-	-	SRAM4	SRAM3	SRAM2	SRAM1	-	-	-	SRAMH

Bit	Symbol	Bit name	Description	Read and write
b31	DCU8	DCU8 function clock control	0: Digital computing unit DCU8 function clock enable 1: Digital computing unit DCU8 function clock disable	R/W
b30	DCU7	DCU7 function clock control	0: Digital computing unit DCU7 function clock enable 1: Digital computing unit DCU7 function clock disable	R/W
b29	DCU6	DCU6 function clock control	0: Digital computing unit DCU6 function clock enable 1: Digital computing unit DCU6 function clock disable	R/W
b28	DCU5	DCU5 function clock control	0: Digital computing unit DCU5 function clock enable 1: Digital computing unit DCU5 function clock disable	R/W
b27	DCU4	DCU4 function clock control	0: Digital computing unit DCU4 function clock enable 1: Digital computing unit DCU4 function clock disable	R/W
b26	DCU3	DCU3 function clock control	0: Digital computing unit DCU3 function clock enable 1: Digital computing unit DCU3 function clock disable	R/W
b25	DCU2	DCU2 function clock control	0: Digital computing unit DCU2 function clock enable 1: Digital computing unit DCU2 function clock disable	R/W
b24	DCU1	DCU1 function clock control	0: Digital computing unit DCU1 function clock enable 1: Digital computing unit DCU1 function clock disable	R/W
b23	CRC	CRC function clock control	0: CRC function clock enable 1: CRC function clock disalbe	R/W
b22	TRNG	TRNG function clock control	0: The true random generator TRNG function clock in the encryption coprocessing module CPM is enabled 1: The true random generator TRNG function clock in the encryption coprocessing module CPM is disabled	R/W
b21	HASH	HASH function clock control	0: The HASH function clock in the encryption coprocessing module CPM is enabled 1: The HASH function clock in the encryption coprocessing module CPM is disabled	R/W
b20	AES	AES function clock control	0: The encryption and decryption algorithm processor AES function clock in the encryption co-processing module CPM is enabled 1: The encryption and decryption algorithm processor AES function clock in the encryption coprocessing module CPM is disabled	R/W
b19	MAU	MAU function clock control	0: Math operation unit MAU function clock enable 1: Math operation unit MAU function clock disable	R/W
b18	CTC	CTC functional clock control	0: Internal clock calibrator CTC function clock enable 1: Internal clock calibrator CTC function clock disable	R/W
b17	AOS	AOS functional clock control	0: The AOS function clock enable 1: The AOS function clock disable	R/W
b16	FCM	FCM function clock control	0: The clock frequency measurement module FCM function clock in the clock controller CMU is enabled 1: The clock frequency measurement module FCM function clock in the clock controller CMU is disabled	R/W
b15	DMA2	DMA2 function clock control	0: DMA controller DMA2 function clock enable 1: DMA controller DMA2 function clock disable	R/W
b14	DMA1	DMA1 function clock control	0: DMA controller DMA1 function clock enable 1: DMA controller DMA1 function clock disable	R/W
b13	KEY	KEYSCAN function clock control	0: The KEYSAN function clock enable 1: The KEYSAN function clock disable	R/W
b12-b11	Reserved	-	Read as "1", write as "1"	R/W
b10	SRAMB	BACKUP-SRAM clock control	0: BACKUP-SRAM clock enable 1: BACKUP-SRAM clock disable	R/W
b9-b8	Reserved	-	Read as "10", write as "10"	R/W
B7	SRAM4	SRAM4 clock control	0: SRAM4 (0x20058000~0x2005FFFF) clock enable 1: SRAM4 (0x20058000~0x2005FFFF) clock disable	R/W
b6	SRAM3	SRAM3 clock control	0: SRAM3 (0x20040000~0x20057FFF) clock enable 1: SRAM3 (0x20040000~0x20057FFF) clock disable	R/W
b5	SRAM2	SRAM2 clock control	0: SRAM2 (0x20020000~0x2003FFFF) clock enable 1: SRAM2 (0x20020000~0x2003FFFF) clock disable	R/W

b4	SRAM1	SRAM1 clock control	0: SRAM1 (0x20000000~0x2001FFFF) clock enable 1: SRAM1 (0x20000000~0x2001FFFF) clock disable	R/W
b3-b1	Reserved	-	Read as "1", write as "1"	R/W
b0	SRAMH	SRAMH clock control	0: SRAMH (0x1FFE0000~0x1FFFFFFF) clock enable 1: SRAMH (0x1FFE0000~0x1FFFFFFF) clock disable	R/W

5.7.13 Function Clock Control 1 (PWC_FCG1)

Reset value: 0xFFFFFFFF

b31	b30	b29	b28	b27	b26	b25	b24
-	-	-	-	FMAC4	FMAC3	FMAC2	FMAC1
b23	b22	b21	b20	b19	b18	b17	b16
USBHS	USBFS	SPI6	SPI5	SPI4	SPI3	SPI2	SPI1
b15	b14	b13	b12	b11	b10	b9	b8
I2S4	I2S3	I2S2	I2S1	SDIOC2	SDIOC1	I2C6	I2C5
B7	b6	b5	b4	b3	b2	b1	b0
I2C4	I2C3	I2C2	I2C1	QSPI	ETHMAC	CAN2	CAN1

Bit	Symbol	Bit name	Description	Read and write
b31-b28	Reserved	-	Read as "1", write as "1"	R/W
b27	FMAC4	FMAC4 function clock control	0: Filter math accelerator FMAC4 function clock enable 1: Filter math accelerator FMAC4 function clock disable	R/W
b26	FMAC3	FMAC3 function clock control	0: Filter math accelerator FMAC3 function clock enable 1: Filter math accelerator FMAC3 function clock disable	R/W
b25	FMAC2	FMAC2 function clock control	0: Filter math accelerator FMAC2 function clock enable 1: Filter math accelerator FMAC2 function clock disable	R/W
b24	FMAC1	FMAC1 function clock control	0: Filter math accelerator FMAC1 function clock enable 1: Filter math accelerator FMAC1 function clock disable	R/W
b23	USBHS	USBHS function clock control	0: USB2.0 high-speed module USBHS function clock enable 1: USB2.0 high-speed module USBHS function clock disable	R/W
b22	USBFS	USBFS function clock control	0: USB2.0 full-speed module USBFS function clock enable 1: USB2.0 full-speed module USBFS function clock disable	R/W
b21	SPI6	SPI6 function clock control	0: Serial peripheral interface SPI6 function clock enable 1: Serial peripheral interface SPI6 function clock disable	R/W
b20	SPI5	SPI5 function clock control	0: Serial peripheral interface SPI5 function clock enable 1: Serial peripheral interface SPI5 function clock disable	R/W
b19	SPI4	SPI4 function clock control	0: Serial peripheral interface SPI4 function clock enable 1: Serial peripheral interface SPI4 function clock disable	R/W
b18	SPI3	SPI3 function clock control	0: Serial peripheral interface SPI3 function clock enable 1: Serial peripheral interface SPI3 function clock disable	R/W
b17	SPI2	SPI2 function clock control	0: Serial peripheral interface SPI2 function clock enable 1: Serial peripheral interface SPI2 function clock disable	R/W
b16	SPI1	SPI1 function clock control	0: Serial peripheral interface SPI1 function clock enable 1: Serial peripheral interface SPI1 function clock disable	R/W
b15	I2S4	I2S4 function clock control	0: The integrated circuit built-in audio bus module I2S4 function clock enable 1: The integrated circuit built-in audio bus	R/W

			module I2S4 function clock disable	
b14	I2S3	I2S3 function clock control	0: The integrated circuit built-in audio bus module I2S3 function clock enable 1: The integrated circuit built-in audio bus module I2S3 function clock disable	R/W
b13	I2S2	I2S2 function clock control	0: The integrated circuit built-in audio bus module I2S2 function clock enable 1: The integrated circuit built-in audio bus module I2S2 function clock disable	R/W
b12	I2S1	I2S1 function clock control	0: The integrated circuit built-in audio bus module I2S1 function clock enable 1: The integrated circuit built-in audio bus module I2S1 function clock disable	R/W
b11	SDIOC2	SDIOC2 function clock control	0: SDIO controller SDIOC2 function clock enable 1: SDIO controller SDIOC2 function clock disable	R/W
b10	SDIOC1	SDIOC1 function clock control	0: SDIO controller SDIOC1 function clock enable 1: SDIO controller SDIOC1 function clock disable	R/W
b9	I2C6	I2C6 function clock control	0: Integrated circuit bus I2C6 function clock enable 1: Integrated circuit bus I2C6 function clock disable	R/W
b8	I2C5	I2C5 function clock control	0: Integrated circuit bus I2C5 function clock enable 1: Integrated circuit bus I2C5 function clock disable	R/W
B7	I2C4	I2C4 function clock control	0: Integrated circuit bus I2C4 function clock enable 1: Integrated circuit bus I2C4 function clock disable	R/W
b6	I2C3	I2C3 function clock control	0: Integrated circuit bus I2C3 function clock enable 1: Integrated circuit bus I2C3 function clock disable	R/W
b5	I2C2	I2C2 function clock control	0: Integrated circuit bus I2C2 function clock enable 1: Integrated circuit bus I2C2 function clock disable	R/W
b4	I2C1	I2C1 function clock control	0: Integrated circuit bus I2C1 function clock enable 1: Integrated circuit bus I2C1 function clock disable	R/W
b3	QSPI	QSPI function clock control	0: Quad serial peripheral interface QSPI function clock enable 1: Quad serial peripheral interface QSPI function clock disable	R/W
b2	ETHMAC	ETHMAC function clock control	0: Ethernet MAC controller ETHMAC function clock enable 1: Ethernet MAC controller ETHMAC function clock disable	R/W
b1	CAN2	CAN2 function clock control	0: CAN2 function clock enable 1: CAN2 function clock disable	R/W
b0	CAN1	CAN1 function clock control	0: CAN1 function clock enable 1: CAN1 function clock disable	R/W

5.7.14 Function Clock Control 2 (PWC_FCG2)

Reset value: 0xFFFFFFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TIMER A12	TIMER A11	TIMER A10	TIMER A9	TIMER A8	TIMER A7	TIMER A6	TIMER A5	TIMER A4	TIMER A3	TIMER A2	TIMER A1	TIMER 2_4	TIMER 2_3	TIMER 2_2	TIMER 2_1
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
EMB	-	TIMER 0_2	TIMER 0_1	HRPWM	TIMER 4_3	TIMER 4_2	TIMER 4_1	TIMER 6_8	TIMER 6_7	TIMER 6_6	TIMER 6_5	TIMER 6_4	TIMER 6_3	TIMER 6_2	TIMER 6_1

Bit	Marking	Place name	Function	Read and write
b31	TIMERA12	TIMERA12 function clock control	0: Timer A unit 12 function clock enable 1: Timer A unit 12 function clock disable	R/W
b30	TIMERA11	TIMERA11 function clock control	0: Timer A unit 11 function clock enable 1: Timer A unit 11 function clock disable	R/W
b29	TIMERA10	TIMERA10 function clock control	0: Timer A unit 10 function clock enable 1: Timer A unit 10 function clock disable	R/W
b28	TIMERA9	TIMERA9 function clock control	0: Timer A unit 9 function clock enable 1: Timer A unit 9 function clock disable	R/W
b27	TIMERA8	TIMERA8 function clock control	0: Timer A unit 8 function clock enable 1: Timer A unit 8 function clock disable	R/W
b26	TIMERA7	TIMERA7 function clock control	0: Timer A unit 7 function clock enable 1: Timer A unit 7 function clock disable	R/W
b25	TIMERA6	TIMERA6 function clock control	0: Timer A unit 6 function clock enable 1: Timer A unit 6 function clock disable	R/W
b24	TIMERA5	TIMERA5 function clock control	0: Timer A unit 5 function clock enable 1: Timer A unit 5 function clock disable	R/W
b23	TIMERA4	TIMERA4 function clock control	0: Timer A unit 4 function clock enable 1: Timer A unit 4 function clock disable	R/W
b22	TIMERA3	TIMERA3 function clock control	0: Timer A unit 3 function clock enable 1: Timer A unit 3 function clock disable	R/W
b21	TIMERA2	TIMERA2 function clock control	0: Timer A unit 2 function clock enable 1: Timer A unit 2 function clock disable	R/W
b20	TIMERA1	TIMERA1 function clock control	0: Timer A unit 1 function clock enable 1: Timer A unit 1 function clock disable	R/W
b19	TIMER2_4	TIMER2_4 function clock control	0: Timer 2 unit 4 function clock enable 1: Timer 2 unit 4 function clock disable	R/W
b18	TIMER2_3	TIMER2_3 function clock control	0: Timer 2 unit 3 function clock enable 1: Timer 2 unit 3 function clock disable	R/W
b17	TIMER2_2	TIMER2_2 function clock control	0: Timer 2 unit 2 function clock enable 1: Timer 2 unit 2 function clock disable	R/W
b16	TIMER2_1	TIMER2_1 function clock control	0: Timer 2 unit 1 function clock enable 1: Timer 2 unit 1 function clock disable	R/W
b15	EMB	EMB functional clock control	0: Emergency brake EMB function clock enable 1: Emergency brake EMB function clock disable	R/W
b14	Reserved	-	Read as "1", write as "1"	R/W
b13	TIMER0_2	TIMER0_2 function clock control	0: Timer 0 unit 2 function clock enable 1: Timer 0 unit 2 function clock disable	R/W
b12	TIMER0_1	TIMER0_1 function clock control	0: Timer 0 unit 1 function clock enable 1: Timer 0 unit 1 function clock disable	R/W
b11	HRPWM	HRPWM function clock control	0: HRPWM function clock enable 1: HRPWM function clock disable	R/W
b10	TIMER4_3	TIMER4_3 function clock control	0: Timer 4 unit 3 function clock enable 1: Timer 4 unit 3 function clock disable	R/W
b9	TIMER4_2	TIMER4_2 function clock control	0: Timer 4 unit 2 function clock enable 1: Timer 4 unit 2 function clock disable	R/W
b8	TIMER4_1	TIMER4_1 function clock control	0: Timer 4 unit 1 function clock enable 1: Timer 4 unit 1 function clock disable	R/W
B7	TIMER6_8	TIMER6_8 function clock control	0: Timer 6 unit 8 function clock enable 1: Timer 6 unit 8 function clock disable	R/W
b6	TIMER6_7	TIMER6_7 function clock control	0: Timer 6 unit 7 function clock enable 1: Timer 6 unit 7 function clock disable	R/W
b5	TIMER6_6	TIMER6_6 function clock control	0: Timer 6 unit 6 function clock enable 1: Timer 6 unit 6 function clock disable	R/W
b4	TIMER6_5	TIMER6_5 function clock control	0: Timer 6 unit 5 function clock enable 1: Timer 6 unit 5 function clock disable	R/W

b3	TIMER6_4	TIMER6_4 function clock control	0: Timer 6 unit 4 function clock enable 1: Timer 6 unit 4 function clock disable	R/W
b2	TIMER6_3	TIMER6_3 function clock control	0: Timer 6 unit 3 function clock enable 1: Timer 6 unit 3 function clock disable	R/W
b1	TIMER6_2	TIMER6_2 function clock control	0: Timer 6 unit 2 function clock enable 1: Timer 6 unit 2 function clock disable	R/W
b0	TIMER6_1	TIMER6_1 function clock control	0: Timer 6 unit 1 function clock enable 1: Timer 6 unit 1 function clock disable	R/W

5.7.15 Function Clock Control 3 (PWC_FCG3)

Reset value: 0xFFFFFFFF

b31	b30	b29	b28	b27	b26	b25	b24
-	-	USART10	USART9	USART8	USART7	USART6	USART5
b23	b22	b21	b20	b19	b18	b17	b16
USART4	USART3	USART2	USART1	-	EXMC_NFC	EXMC_DMC	EXMC_SMC
b15	b14	b13	b12	b11	b10	b9	b8
DVP	-	-	OTS	-	-	CMP2	CMP1
B7	b6	b5	b4	b3	b2	b1	b0
-	-	DAC2	DAC1	CMBIAS	ADC3	ADC2	ADC1

Bit	Symbol	Bit name	Description	Read and write
b31-b30	Reserved	-	Read as "1", write as "1"	R/W
b29	USART10	USART10 function clock control	0: Universal synchronous asynchronous transceiver USART10 function clock enable. 1: Universal synchronous asynchronous transceiver USART10 function clock disable	R/W
b28	USART9	USART9 function clock control	0: Universal synchronous asynchronous transceiver USART9 function clock enable. 1: Universal synchronous asynchronous transceiver USART9 function clock disable	R/W
b27	USART8	USART8 function clock control	0: Universal synchronous asynchronous transceiver USART8 function clock enable. 1: Universal synchronous asynchronous transceiver USART8 function clock disable	R/W
b26	USART7	USART7 function clock control	0: Universal synchronous asynchronous transceiver USART7 function clock enable. 1: Universal synchronous asynchronous transceiver USART7 function clock disable	R/W
b25	USART6	USART6 function clock control	0: Universal synchronous asynchronous transceiver USART6 function clock enable. 1: Universal synchronous asynchronous transceiver USART6 function clock disable	R/W
b24	USART5	USART5 function clock control	0: Universal synchronous asynchronous transceiver USART5 function clock enable. 1: Universal synchronous asynchronous transceiver USART5 function clock disable	R/W
b23	USART4	USART4 function clock control	0: Universal synchronous asynchronous transceiver USART4 function clock enable. 1: Universal synchronous asynchronous transceiver USART4 function clock disable	R/W
b22	USART3	USART3 function clock control	0: Universal synchronous asynchronous transceiver USART3 function clock enable. 1: Universal synchronous asynchronous transceiver USART3 function clock disable	R/W
b21	USART2	USART2 function clock control	0: Universal synchronous asynchronous transceiver USART2 function clock enable. 1: Universal synchronous asynchronous transceiver USART2 function clock disable	R/W
b20	USART1	USART1 function clock control	0: Universal synchronous asynchronous transceiver USART1 function clock enable. 1: Universal synchronous asynchronous transceiver USART1 function clock disable	R/W
b19	Reserved	-	Read as "1", write as "1"	R/W
b18	EXMC_NFC	EXMC_NFC function clock control	0: NAND FLASH controller of EXMC clock enable 1: NAND FLASH controller of EXMC clock disable	R/W
b17	EXMC_DMC	EXMC_DMC function clock control	0: DMC controller of EXMC clock enable 1: DMC controller of EXMC clock disable	R/W
b16	EXMC_SMC	EXMC_SMC function clock control	0: SMC controller of EXMC clock enable 1: SMC controller of EXMC clock disable	R/W
b15	DVP	DVP function clock control	0: Digital video interface DVP function clock enable 1: Digital video interface DVP function clock disable	R/W

b14-b13	Reserved	-	Read as "1", write as "1"	R/W
b12	OTS	OTS function clock control	0: Temperature sensor OTS function clock enable 1: Temperature sensor OTS function clock disable	
b11-b10	Reserved	-	Read as "1", write as "1"	R/W
b9	CMP2	CMP2 function clock control	0: Voltage comparator CMP2 function clock enable 1: Voltage comparator CMP2 function clock disable	R/W
b8	CMP1	CMP1 function clock control	0: Voltage comparator CMP1 function clock enable 1: Voltage comparator CMP1 function clock disable	R/W
b7-b6	Reserved	-	Read as "1", write as "1"	R/W
b5	DAC2	DAC2 function clock control	0: Digital-to-analog converter DAC2 function clock enable 1: Digital-to-analog converter DAC12 function clock disable	R/W
b4	DAC1	DAC1 function clock control	0: Digital-to-analog converter DAC1 function clock enable 1: Digital-to-analog converter DAC1 function clock disable	R/W
b3	CMBIAS	CMP/PGA/SH reference current source control	0: CMP/PGA/SH reference current source enable 1: CMP/PGA/SH reference current source dsiable	R/W
b2	ADC3	ADC3 function clock control	0: Analog-to-digital conversion ADC3 function clock enable 1: Analog-to-digital conversion ADC3 function clock disable	R/W
b1	ADC2	ADC2 function clock control	0: Analog-to-digital conversion ADC2 function clock enable 1: Analog-to-digital conversion ADC2 function clock disable	R/W
b0	ADC1	ADC1 function clock control	0: Analog-to-digital conversion ADC1 function clock enable 1: Analog-to-digital conversion ADC1 function clock disable	R/W

5.7.16 PWC_FCG0 Protection Control (PWC_FCG0PC)

Reset value: 0x00000000

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

Bit	Symbol	Bit name	Description	Read and write
b31~b16	FCG0PCWE[15:0]	PWC_FCG0PC write enable	Change the value of the PRT0 bit while writing 0xA5A5	R/W
b15-b1	Reserved	-	Read as "0", write as "0"	R/W
b0	PRT0	guard bit	PWC_FCG0 write enable control bit 0: PWC_FCG0 write protection 1: PWC_FCG0 write enable	R/W

5.7.17 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]								FPRCB7	FPRCB6	FPRCB5	FPRCB4	FPRCB3	FPRCB2	FPRCB1	FPRCB0

Bit	Symbol	Bit name	Description	Read and write
b15~b8	PWC_FPRCWE[7:0]	PWC_FPRC register write enable	PWC_FPRC can be updated while writing 0xA5h value, Otherwise, it has no effect on the lower 8-bit write value. Read at 0x00.	R/W
B7	FPRCB7	FPRC bit 7	Reserved, "0" when reading, "0" when writing	R/W
b6	FPRCB6	FPRC bit 6	Reserved, "0" when reading, "0" when writing	R/W
b5	FPRCB5	FPRC bit 5	Reserved, "0" when reading, "0" when writing	R/W
b4	FPRCB4	FPRC bit 4	Reserved, "0" when reading, "0" when writing	R/W
b3	FPRCB3	FPRC bit 3	Protect register Bit, protection object refer to table 7-9 0: write protection 1: write enable	R/W
b2	FPRCB2	FPRC bit 2	Reserved, "0" when reading, "0" when writing	R/W
b1	FPRCB1	FPRC bit 1	Protect register Bit, protection object refer to table 7-9 0: write protection 1: write enable	R/W
b0	FPRCB0	FPRC bit 0	Protect register Bit, protection object refer to table 7-9 0: write protection 1: write enable	R/W

5.7.18 STOP Mode Control Register (PWC_STPMCR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
STOP	EXBUSOE	-	-	-	-	-	-	-	-	-	-	-	-	CKSMRC	FLNWT

Bit	Symbol	Bit name	Description	Read and write
b15	STOP	STOP mode control bit	0: STOP mode is invalid 1: STOP mode is valid	R/W
b14	EXBUSOE	State control for EXMC address bus and control signal in power-down mode or stop mode 0: High impedance 1: Maintain the state before power-down mode or stop mode	State control for EXMC address bus and control signal in power-down mode or stop mode 0: High impedance 1: Maintain the state before power-down mode or stop mode	R/W
b13-b2	Reserved	-	Read as "0", write as "0"	R/W
b1	CKSMRC	Clock switch to MRC control bit	0: After the STOP mode wakes up, the clock and frequency division maintain the settings before entering 1: After the STOP mode wakes up, the system clock switches to MRC, SCKCFGR registers are initialized	R/W
b0	FLNWT	FLASH stability waiting control	0: After waking up from stop mode, wait for FLASH to stabilize 1: After waking up from stop mode, do not wait for FLASH to stabilize	R/W

5.7.19 RAM Power Control Register 0 (PWC_RAMPC0)

Reset value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	RAMDC10	RAMDC9	RAMPDC8
7	6	5	4	3	2	1	0
RAMPDC7	RAMPDC6	RAMPDC5	RAMPDC4	RAMPDC3	RAMPDC2	RAMPDC1	RAMPDC0

Bit	Symbol	Bit name	Description	Read and write
b32-b11	Reserved	-	Read as "0", write as "0"	R/W
b10	RAMPDC10	RAM power-down control bit 10	0: RAM 0x1FFF8000~0x1FFFFFF power on 1: RAM 0x1FFF8000~0x1FFFFFF power down	R/W
b9	RAMPDC9	RAM power-down control bit 9	0: RAM 0x1FFF0000~0x1FFF7FFF power on 1: RAM 0x1FFF0000~0x1FFF7FFF power down	R/W
b8	RAMPDC8	RAM power-down control bit 8	0: RAM 0x1FFE8000~0x1FFFFFF power on 1: RAM 0x1FFE8000~0x1FFFFFF power down	R/W
B7	RAMPDC7	RAM power-down control bit 7	0: RAM 0x1FFE0000~0x1FFE7FFF power on 1: RAM 0x1FFE0000~0x1FFE7FFF power down	R/W
b6	RAMPDC6	RAM power-down control bit 6	0: RAM 0x20058000~0x2005FFFF power on 1: RAM 0x20058000~0x2005FFFF power down	R/W
b5	RAMPDC5	RAM power-down control bit 5	0: RAM 0x20050000~0x20057FFF power on 1: RAM 0x20050000~0x20057FFF power down	R/W
b4	RAMPDC4	RAM power-down control bit 4	0: RAM 0x20040000~0x2004FFFF power on 1: RAM 0x20040000~0x2004FFFF power down	R/W
b3	RAMPDC3	RAM power-down control bit 3	0: RAM 0x20030000~0x2003FFFF power on 1: RAM 0x20030000~0x2003FFFF power down	R/W
b2	RAMPDC2	RAM power-down control bit 2	0: RAM 0x20020000~0x2002FFFF power on 1: RAM 0x20020000~0x2002FFFF power down	R/W
b1	RAMPDC1	RAM power-down control bit 1	0: RAM 0x20010000~0x2001FFFF power on 1: RAM 0x20010000~0x2001FFFF power down	R/W
b0	RAMPDC0	RAM power-down control bit 0	0: RAM 0x20000000~0x2000FFFF power on 1: RAM 0x20000000~0x2000FFFF power down	R/W

5.7.20 RAM Operating Conditions 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]							

Bit	Marking	Place name	Function	Read and write
B15-b0	PWC_RAMOPM[15:0]	RAM operation mode selection bit	When the chip works in high-speed run mode, PWC_RAMOPM is set as 0x8043 When the chip works in ultra-low speed run mode, PWC_RAMOPM is set as 0x9062	R/W

5.7.21 Peripheral RAM Low Power Control Register (PWC_PRAMLPC)

Reset value: 0x00000000

-	-	-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16		
-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8		
-	-	-	-	-	-	-	-	PRAMPDC9	PRAMPDC8
7	6	5	4	3	2	1	0		
PRAMPDC7	PRAMPDC6	PRAMPDC5	PRAMPDC4	PRAMPDC3	PRAMPDC2	PRAMPDC1	PRAMPDC0		

Bit	Symbol	Bit name	Description	Read and write
b31~b10	Reserved	-	Read as "0", write as "0"	R/W
b9	PRAMPDC9	Peripheral RAM power-down control bit 9	NFC_RAM power-down control 0: power on 1: Power down	R/W
b8	PRAMPDC8	Peripheral RAM power-down control bit 8	SDIO2_RAM power-down control 0: power on 1: Power down	R/W
B7	PRAMPDC7	Peripheral RAM power-down control bit 7	SDIO1_RAM power-down control 0: power on 1: Power down	R/W
b6	PRAMPDC6	Peripheral RAM power-down control bit 6	ETHER_RX RAM power-down control 0: power on 1: Power down	R/W
b5	PRAMPDC5	Peripheral RAM power-down control bit 5	ETHER_TX RAM power-down control 0: power on 1: Power down	R/W
b4	PRAMPDC4	Peripheral RAM power-down control bit 4	USBHS_RAM Power Down Control 0: power on 1: Power down	R/W
b3	PRAMPDC3	Peripheral RAM power-down control bit 3	USBFS_RAM Power Down Control 0: power on 1: Power down	R/W
b2	PRAMPDC2	Peripheral RAM power-down control bit 2	CACHE_RAM Power Down Control 0: power on 1: Power down	R/W
b1	PRAMPDC1	Peripheral RAM power-down control bit 1	CAN_2_RAM power-down control 0: power on 1: Power down	R/W
b0	PRAMPDC0	Peripheral RAM power-down control bit 0	CAN_1_RAM power-down control 0: power on 1: Power down	R/W

5.7.22 PVD Control Register 0 (PWC_PVDCR0)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
-	PVD2EN	PVD1EN	-	-	-	-	EXVCCINEN

Bit	Symbol	Bit name	Description	Read and write
B7	Reserved	-	Read as "0", write as "0"	R/W
b6	PVD2EN	Voltage detection 2 enable	0: Voltage detection 2 circuit disable 1: Voltage detection 2 circuit enable	R/W
b5	PVD1EN	Voltage detection 1 enable	0: Voltage detection 1 circuit disable 1: Voltage detection 1 circuit enable	R/W
b4-b1	Reserved	-	Read as "0", write as "0"	R/W
b0	EXVCCINEN	External VCC voltage input enable	0: External VCC Voltage input disable 1: External VCC voltage input enable	R/W

5.7.23 PVD Control Register 1 (PWC_PVDCR1)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
-	PVD2CMPOE	PVD2IRS	PVD2IRE	-	PVD1CMPOE	PVD1IRS	PVD1IRE

Bit	Symbol	Bit name	Description	Read and write
B7	Reserved	-	Read as "0", write as "0"	R/W
b6	PVD2CMPOE	PVD2 compare result output enable	0: PVD2 compare result output disable 1: PVD2 compare result output enable	R/W
b5	PVD2IRS	PVD2 interrupt reset selection	0: Interrupt is generated when the VCC change meets the PVD2 detection condition 1: Reset is generated when VCC drops through V_{PVD2} . Note: When the PVD1IRS bit is "1" or the PVD2IRS bit is "1", the power-down mode cannot be entered. To enter power-down mode, the PVD1IRS and PVD2IRS must be set to "0"	R/W
b4	PVD2IRE	PVD2 interrupt reset enable	0: PVD2 interrupt or reset disable 1: PVD2 interrupt or reset enable Note: Please set the PVD2IRE bit to "1" when both the PVD2IRE and PVD2CMPOE are "1"	R/W
b3	Reserved	-	Read as "0", write as "0"	R/W
b2	PVD1CMPOE	PVD1 compare result output enable	0: PVD1 compare result output disable 1: PVD1 compare result output enable	R/W
b1	PVD1IRS	PVD1 interrupt reset selection	0: Interrupt is generated when the VCC change meets the PVD1 detection condition 1: Reset is generated when VCC drops through V_{PVD1} . Note: When the PVD1IRS bit is "1" or the PVD2IRS bit is "1", the power-down mode cannot be entered. To enter power-down mode, the PVD1IRS and PVD2IRS must be set to "0"	R/W
b0	PVD1IRE	PVD1 interrupt reset enable	0: PVD1 interrupt or reset disable 1: PVD1 interrupt or reset enable Note: Please set the PVD1IRE bit to "1" when both the PVD1IRE and PVD1CMPOE are "1"	R/W

5.7.24 PVD Filter Control Register (PWC_PVDFCR)

Reset value: 0x11

B7	b6	b5	b4	b3	b2	b1	b0
	PVD2NFCKS[1:0]		PVD2NFDIS		PVD1NFCKS[1:0]		PVD1NFDIS
<hr/>							
Bit	Symbol	Bit name			Description		
B7	Reserved	-			Read as "0", write as "0"		
b6~b5	PVD2NFCKS	PVD2 digital filter sampling clock selection			00: 2 filter clock cycles 01: 4 filter clock cycles 10: 8 filter clock cycles 11: 16 filter clock cycles 1 Filter clock cycle = LRC cycle / 8 Note: This bit can only be rewritten when the PVD2NFDIS bit is "1"		
b4	PVD2NFDIS	PVD2 digital filter control			0: Digital filter enable 1: Digital filter disable		
b3	Reserved	-			Read as "0", write as "0"		
b2~b1	PVD1NFCKS	PVD1 digital filter sampling clock selection			00: 2 filter clock cycles 01: 4 filter clock cycles 10: 8 filter clock cycles 11: 16 filter clock cycles 1 Filter clock cycle = LRC cycle / 8 Note: This bit can only be rewritten when the PVD1NFDIS bit is "1"		
b0	PVD1NFDIS	PVD1 digital filter control			0: Digital filter enable 1: Digital filter disable		

5.7.25 PVD Level Control Register (PWC_PVDLCR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
		PVD2LVL[2:0]		-		PVD1LVL[2:0]	
<hr/>							
Bit	Symbol	Bit name			Description		
B7	Reserved	-			Read as "0", write as "0"		
b6~b4	PVD2LVL[2:0]	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 (It can only be set when PWC_PVDCR0.EXVCCINEN=1, and it is forbidden to set in other cases.)		
b3	Reserved	-			Read as "0", write as "0"		
b2~b0	PVD1LVL[2:0]	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		

5.7.26 PVD Interrupt Control Register (PWC_PVDICR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
-	PVD2EDGS[1:0]		PVD2NMIS	-	PVD1EDGS[1:0]		PVD1NMIS

Bit	Symbol	Bit name	Description	Read and write
B7	Reserved	-	Read as "0", write as "0"	R/W
b6~b5	PVD2EDGS[1:0]	PVD2 detection condition selection	00: When $V_{CC} < V_{PVD2}$ (falling) is detected 01: When $V_{CC} \geq V_{PVD2}$ (rising) is detected 10: When $V_{CC} < V_{PVD2}$ (falling) is detected or when $V_{CC} \geq V_{PVD2}$ (rising) is detected 11: Setting is prohibited.	R/W
b4	PVD2NMIS	PVD2 interrupt type selection	0: PVD2 detection generates non-maskable interrupt 1: PVD2 detection generates maskable interrupt	R/W
b3	Reserved	-	Read as "0", write as "0"	R/W
b2~b1	PVD1EDGS[1:0]	PVD1 detection condition selection	00: When $V_{CC} < V_{PVD1}$ (falling) is detected 01: When $V_{CC} \geq V_{PVD1}$ (rising) is detected 10: When $V_{CC} < V_{PVD1}$ (falling) is detected or when $V_{CC} \geq V_{PVD1}$ (rising) is detected 11: Setting is prohibited.	R/W
b0	PVD1NMIS	PVD1 interrupt type selection	0: PVD1 detection generates non-maskable interrupt 1: PVD1 detection generates maskable interrupt	R/W

5.7.27 PVD Detection Status Register (PWC_PVDDSR)

Reset value: 0x11

B7	b6	b5	b4	b3	b2	b1	b0
-	-	PVD2DETFGL	PVD2MON	-	-	PVD1DETFGL	PVD1MON

Bit	Symbol	Bit name	Description	Read and write
b7~b6	Reserved	-	Read as "0", write as "0"	R/W
b5	PVD2DETFGL	PVD2 detection flag	0: PVD2 does not detect V_{CC} passing through V_{PVD2} 1: PVD2 detects that V_{CC} passes through V_{PVD2} Write 0 to PVD1DETFGL to clear this bit after reading. Note: This flag is valid when the PVD2EN bit is "1" and the PVD2CMPOE bit is "1"	R/W
b4	PVD2MON	PVD2 monitor bit	0: $V_{CC} \leq V_{PVD2}$ or external input comparison voltage \leq PVD2 internal reference voltage 1: When PVD2 is invalid or $V_{CC} > V_{PVD2}$ or external input comparison voltage $>$ PVD2 internal reference voltage	R
b3~b2	Reserved	-	Read as "0", write as "0"	R/W
b1	PVD1DETFGL	PVD1 detection flag	0: PVD1 does not detect V_{CC} passing through V_{PVD1} 1: PVD1 detects that V_{CC} passes through V_{PVD1} Write 0 to clear this bit after reading. Note: This flag is valid when the PVD1EN bit is "1" and the PVD1CMPOE bit is "1"	R/W
b0	PVD1MON	PVD1 monitor bit	0: $V_{CC} < V_{PVD1}$ 1: When PVD1 is invalid or $V_{CC} > V_{PVD1}$	R

5.7.28 Backup Domain Reset Register (PWC_VBATRSTR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
RST[7:0]							
Bit	Symbol	Bit name	Description				Read and write
b7-0	RST[7:0]	Backup Domain Reset Register	Write 0xA5 to reset the backup field				W

5.7.29 Backup Domain Control Register (PWC_VBATCR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
CSDIS	-	-	-	VBATDIVMONE	RAMPDF	RAMVALID	VBTRSD
Bit	Symbol	Bit name	Description				Read and write
B7	CSDIS	VBAT domain current source control	0: VBAT domain current source is enabled 1: VBAT domain current source is disabled, this bit can be set to 1 only when backup RAM, XTAL32, XTAL32NF are not used				R/W
b6-b4	Reserved	-	Read as "0", write as "0"				R/W
b3	VBATDIVMONE	VBAT voltage divider circuit control bit	0: The voltage divider circuit disable 1: The voltage divider circuit enable				R/W
b2	RAMPDF	RAM power-down flag	0: Backup-RAM is powered on 1: Backup-RAM is powered down				R
b1	RAMVALID	RAM valid flag	0: Backup-RAM cannot be read or written 1: Backup-RAM can read and write				R
b0	VBTRSD	VBAT RLDO close control bit	0: VBAT RLDO on 1: VBAT RLDO off				R/W

5.7.30 VBAT Backup Registers 0~127 (PWC_BKR0~PWC_BKR127)

Reset value: 0XX

B7	b6	b5	b4	b3	b2	b1	b0
bkrNb[7:0]							
N=0~127							
Bit	Symbol	Bit name	Description				Read and write
b7-b0	bkrNb[7:0]	backup register	backup register				R/W

5.7.31 Wake-up Timer Control Register 0 (PWC_WKTC0)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
WKTMCMP[7:0]							
Bit	Symbol	Bit name	Description				Read and write
b7-0	WKTMCMP[7:0]	Compare value of WKTMCMP counter	Compare value of WKTMCMP counter				R/W

5.7.32 Wake-up Timer Control Register 1 (PWC_WKTC1)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	WKTMCMP[11:8]			
Bit	Symbol	Bit name	Description				Read and write
b7-b4	Reserved	-	Read as "0", write as "0"				R/W
b3-0	WKTMCMP[11:8]	Compare value of WKTMCMP counter	Compare value of WKTMCMP counter				R/W

5.7.33 Wake-up Timer Control Register 2 (PWC_WKTC2)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
WKTCE	WKCKS[1:0]		WKOVF	-	-	-	-
Bit	Symbol	Bit name	Description				Read and write
B7	WKTCE	WKTMCMP control bit	0: WKTMCMP stop 1: WKTMCMP count Clearing the WKTCE bit when the WKTMCMP is counting will clear the counter and stop the counter.				R/W
b6-b5	WKCKS[1:0]	WKTMCMP Clock Selection	00: 64Hz clock 01: XTAL32 10: RTCLRC 11: Setting is prohibited.				R/W
b4	WKOVF	WKTMCMP comparison result flag	0: The counter is inconsistent with the WKTMCMP value 1: The counter is consistent with the WKTMCMP value Write 0 to clear this flag.				R/W
b3-b0	Reserved	-	Read as "0", write as "0"				R/W

6 Initial Configuration (ICG)

6.1 Introduction

After chip reset is released, the hardware circuit will load the data of FLASH address 0x00000400~0x0000045F to circuit. Addresses 0x0000_0408~0x0000_040B,

0x0000_0410~0x0000_041F, 0x0000_0438~0x0000_045F are reserved, they should be all 1 to ensure the normal function of the chip. When boot exchange of FLASH is valid and OTP is not enabled, the ICG is loaded from FLASH1, otherwise, it is from FLASH0. Customer can modify the initial configuration register by programming or erasing FLASH. The address 0x0000_0420~0x0000_0437 is the data security protection configuration area. For details, please refer to the specification [Data security configuration]. The register reset value is determined by the FLASH data.

The initial configuration register address is listed as follows:

ICG_BASE_ADDR: 0x00000400

Table 6-1 List of Registers

Register name	Symbol	Offset address	Bit width	Reset value
Initial configuration register0	ICG0	0x000	32	reset value
Initial configuration register1	ICG1	0x004	32	reset value
Initial configuration register3	ICG3	0x00C	32	reset value

6.2 Register description

6.2.1 Initial configuration register0 (ICG0)

Reset value: reset value

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	WDT SLPO FF		WDTWDPT[3:0]			WDTCKS[3:0]		WDTPERI[1: 0]		WDTIT S		WDTA UTS	
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	SWD TSLP OFF		SWDTWDPT[3:0]			SWDTCKS[3:0]		SWDTPERI[1 :0]		SWDTIT S		SWDT AUTS	

Bit	Symbol	Bit name	Description	Read and write
b31~b29	Reserved	-	Functional reservation bit	R
b28	WDTSLPOFF	WDT stops counting in sleep mode	0: WDT does not stop counting in sleep mode 1: WDT stops counting in sleep mode	R
b27~b24	WDTWDPT[3:0]	count percentage of refresh permit region	refresh permit region of WDT count 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 count 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: Reservation	R
b19~b18	WDTPERI[1:0]	WDT count overflow period	00: 256 cycles 01: 4,096 cycles 10: 16384 cycle 11: 65536 cycle	R
b17	WDTITS	WDT interrupt selection	0: Interrupt request 1: Reset request	R
b16	WDTAUTS	WDT starts automatically	0: WDT starts after reset (hardware start) 1: WDT stops after reset	R
b15~b13	Reserved	-	Functional reservation bit	R
b12	SWDTSLPOFF	SWDT count stops in Sleep and Stop mode	0: SWDT doesn't stop counting in sleep and stop mode 1: SWDT stops counting in sleep and stop mode	R
b11~b8	SWDTWDPT[3:0]	count percentage of refresh permit region	refresh permit region of SWDT count 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%	R

			1010: 25%~50%, 75%~100% 1011: 0%~50%, 75%~100% 1100: 50%~100% 1101: 0%~25%, 50%~100% 1110: 25%~100% 1111: 0%~100%	
b7~b4	SWDTCKS[3:0]	SWDT counter clock	0000: SWDTCLK 0100: SWDTCLK/16 0101: SWDTCLK/32 0110: SWDTCLK/64 0111: SWDTCLK/128 1000: SWDTCLK/256 1011: SWDTCLK/2048 Other values: Reservation	R
b3~b2	SWDTPERI[1:0]	SWDT count overflow cycle	00: 256 cycles 01: 4,096 cycles 10: 16384 cycle 11: 65536 cycle	R
b1	SWDTITS	SWDTinterrupt selection	0: Interrupt request 1: Reset request	R
b0	SWDTAUTS	SWDT auto start	0: SWDT starts after reset (hardware starts) 1: SWDT stops after reset	R

6.2.2 Initial configuration register1 (ICG1)

Reset value: reset value

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	BOR DIS	BOR_lev[1:0]	
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0

-	-	-	-	-	-	-	HRC STOP	-	-	-	-	-	-	-	HRCF REQSEL
---	---	---	---	---	---	---	----------	---	---	---	---	---	---	---	-------------

Bit	Symbol	Bit name	Description	Read and write
b31~b19	Reserved	-	Functional reservation bit	R
b18	BORDIS	BOR action selection	0: BOR action is allowed after reset 1: Disable BOR action 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	-	Functional reservation bit	R
b8	HRCSTOP	HRC oscillation stop	0: HRC oscillates 1: HRC stops	R
b7~b1	Reserved	-	Read as "1", write as "1"	R
b0	HRCFREQSEL	HRC frequency selection	0: 20MHz 1: 16MHz	R

6.2.3 Initial configuration register3 (ICG3)

Reset value: reset value

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
DBUSPRT[15:0]															

Bit	Symbol	Bit name	Description	Read and write
b31~b16	Reserved	-	Functional reservation bit	R
b15~b0	DBUSPRT[15:0]	D-BUS read protection	When DBUSPRT[15:0]=0x4450, Enable the D-BUS read protection for the 0x0000_0000~0x0001_FFFF area; otherwise, the D-BUS read protection is invalid.	R

7 Embedded FLASH (EFM)

7.1 Introduction

FLASH The interface accesses FLASH through FLASH ICODE, DCODE and MCODE bus. This interface can perform programming, sector erasing and mass erasing operations on FLASH; code execution is accelerated through instruction prefetching and caching mechanisms.

7.2 Main features

- Two independent FLASH that can realize BGO (BackGroud Operation) function
- 134Kbytes of OTP area
- prefetch on ICODE bus
- 256 cache lines of 128 bits on ICODE
- 64 cache lines of 128 bits on DCODE
- Supports boot swap function
- Support data security protection

7.3 Embedded FLASH

FLASH has the following detail features:

- The capacity is 2Mbytes, composed of two 1Mbytes FLASH. It is 256 sectors in all, and each sector is 8Kbytes. Sector 0 to sector 15 in FLASH0 can be configured as an OTP area.
- The OTP (One Time Program) area has a total of 134KBytes, of which 128Kbytes are configured in the FLASH0 address 0x0000_0000~0x0001_FFFF, and 6Kbytes are configured in the address 0x0300_0000~0x0300_17FF. Address 0x0300_1800~0x0300_1AD7 is the OTP lock area.
- 128-bit width read buffer that can speed up code execution.
- The program unit is 4bytes, and the sector erase unit is 8Kbytes.

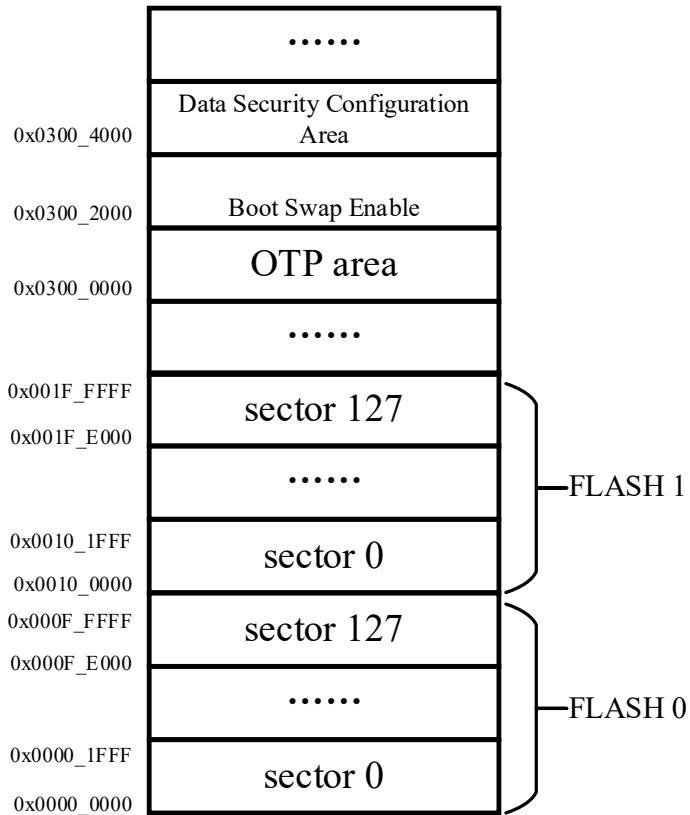


Figure 7-1 FLASH address distribute

OTP area is 0x0300_0000~0x0300_1ADB, boot swap eanble address is 0x0300_2000~0x0300_2003 and data security configuration area is 0x0300_4000~0x0300_400B. They possess three separate sectors who belong to FLASH0. Mass erase to these three sectors is prohibited, and the data in these three sectors will not be erased even FLASH0 has been mass erased.

7.4 Read interface

7.4.1 Relationship between CPU clock and FLASH read cycle

Customer needs to set wait cycles (EFM_FRMC.FLWT[3:0]) correctly according to the CPU clock frequency. After the system is reset, the CPU clock source is MRC (8MHz), and FLASH wait cycle is 0. It is recommended that the customer should set CPU frequency and FLASH wait cycles by following steps. Regarding the number of waiting cycles, please refer to Table 7-1.

Increase CPU clock Frequency Step:

1. Set new wait cycles (FLWT[3:0]) to register EFM_FRMC.
2. Read register EFM_FRMC and check whether the new setting is successful.
3. Increase CPU clock frequency 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 the register CMU_CKSWR or CMU_SCFGR to check whether the new setting is successful.

Decrease CPU Frequency Step:

1. Reduce the CPU clock frequency by setting the system clock source switch register CMU_CKSWR(CKSW[2:0]) or the system clock configuration register CMU_SCFGR(HCLKS[2:0]).
2. Read the register CMU_CKSWR or CMU_SCFGR to check whether the new setting is successful.
3. Set new wait cycles(FLWT[3:0]) to register EFM_FRMC.
4. Read register EFM_FRMC and check whether the new setting is successful.

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

CPU clock frequency (hclk)	Flash wait cycles setting
200MHz < Fhclk ≤ 240MHz	FLWT[3:0]=4'b0101
160MHz < Fhclk ≤ 200MHz	FLWT[3:0]=4'b0100
120MHz < Fhclk ≤ 160MHz	FLWT[3:0]=4'b0011
80MHz < Fhclk ≤ 120MHz	FLWT[3:0]=4'b0010
40MHz < Fhclk ≤ 80MHz	FLWT[3:0]=4'b0001
Fhclk ≤ 40MHz	FLWT[3:0]=4'b0000

7.5 FLASH Read Acceleration

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

FLASH controller has a read acceleration called cache, which can optimize flash read time. In order to enhance processor performance, cache will store the data of ICODE and DCODE to cache RAM, thereby improving the program execution speed.

This product provides 5Kbytes SRAM as cache memory, which can effectively reduce the time loss caused by instruction jumps. Cache is enabled by setting the enable (ICACHE/DCACHE) bit in the EFM_FRMC register. Whenever an instruction or data miss occurs (the requested instruction or data isn't existed in read buffer and cache memory), cache will copy new read data(128 bits) to cache memory. If CPU requests an instruction or data that is already in cache, it can be acquired immediately without reading actual FLASH. If cache memory is full, the LRU (least recently used) policy is used to determine which data should to be replaced.

When CPU reads instructions or data, FLASH address and 128bit-width data are stored in the read buffer. Also, the FLASH read cycles will change if cache hit. For details, please refer to Table 7-2 .

Table 7-2 FLASH read cycles

EFM_FRMC.FLWT[3:0] setting	Cache is not enabled		Cache enable	
	Read buffer hit	Read buffer miss	Read buffer or cache hit	Read buffer and cache miss
0	1	1	1	1
1	1	2	1	2
2	1	3	1	3
N(N>2)	1	N+1	1	N+1

7.6 FLASH program and erase

FLASH supports program, sector erase, and mass erase operations.

FLASH address must be aligned with 4 if program, sector erase and mass erase operation is sent. Program unit is 4bytes, sector erase unit is 8Kbytes, and mass erase can be a single FLASH or double FLASH according to the mass erase mode. This product has three kinds of program mode: single program without readback, single program with readback, and series program.

Bus will hang on by setting EFM_FWMC.BUSHLDCTL to 0 while programming and erasing. If EFM_FWMC.BUSHLDCTL is set to 1, bus willn't hang on, CPU will go ahead.

And please disable cache and prefetch enable before program and erase. The following steps are for program and erase operation.

7.6.1 Unlock EFM_KEY1 register

After the reset is released, FLASH write mode register (EFM_FWMC) is in the write-disabled state. And First we should release protection from access protection register (EFM_FAPRT), then release protection from EFM_KEY1. The following steps are used to disable the above protection.

- 1) Release access protection (write 0x0123 first, then 0x3210 to EFM_FAPRT)
- 2) Unlock EFM_KEY1 (write 0x01234567 first, then 0xFEDCBA98 to EFM_KEY1)

If wrong sequence is written to EFM_KEY1, a bus error will occur, and the register will be self-locked until reset occurs. It is possible to write 1 to EFM_FWMC.KEY1LOCK in order to get EFM_KEY1 locked state back.

7.6.2 Write protection

Each sector of FLASH has one write protection bit, that is existed in the write protection register EFM_F0/1NWPRTx (x=0~3). Each write protection register has a write protection lock bit (WLCK[7:0]). Once the write protection lock bit is set to 1, the write protection register bit can only be set from write permission to write prohibition. When the FLASH sector is set to write-disabled, program and sector erase to this sector address will fail, and an error flag (EFM_FSR.PRTWERR0/1=1) will be set.

In mass erase mode, if one or more sectors in the FLASH are set to write-disable, then mass erase will fail and an error flag (EFM_FSR.PRTWERR0/1=1) will be set.

7.6.3 Single program without readback

The steps for setting single program without readback are as follows:

- 1) release protection from access protection register. (write 0x0123 first, then 0x3210 to EFM_FAPRT)
- 2) unlock EFM_KEY1. (write 0x01234567 first, then 0xFEDCBA98 to EFM_KEY1)
- 3) set single program without readback mode. (EFM_FWMC.PEMODE[2:0]=001)
- 4) release write protection. (write 1 to corresponding bit in EFM_F0/1NWPRTx (x=0~3))
- 5) write 32-bit data to the program address
- 6) wait for FLASH to be idle. (EFM_FSR.RDY0/1=1)
- 7) read the programmed address and compare whether it is the same as write data
If same, it indicates program is successful, else it means that the FLASH address has been destroyed and permanently abandoned.
- 8) clear program end flag. (EFM_FSR.OPTEND0/1)
Program data to an OTP area that has been locked will fail, and flag bit EFM_FSR.OTPWERRO will be set.

7.6.4 Single program with readback

Single program with readback mode means that hardware will automatically reads the programmed address after programmed and compares. The flag EFM_FSR.MISMTCH0/1 indicates compare result.

The steps for setting single program with readback are as follows:

- 1) release protection from access protection register. (write 0x0123 first, then 0x3210 to EFM_FAPRT)
- 2) unlock EFM_KEY1. (write 0x01234567 first, then 0xFEDCBA98 to EFM_KEY1)

- 3) set single program with readback mode. (EFM_FWMC.PEMODE[2:0]=010)
- 4) release write protection. (write 1 to corresponding bit in EFM_F0/1NWPRTx (x=0~3))
- 5) write 32-bit data to the program address
- 6) wait for FLASH to be idle. (EFM_FSR.RDY0/1=1)
- 7) check data mismatch flag bit. (EFM_FSR.MISMTCH0/1)
If it is 0, it means that the program is successful; else it means that the FLASH address has been destroyed and is permanently abandoned.
- 8) clears program end flag. (EFM_FSR.OPTEND0/1)
Program data to an OTP area that has been locked will fail, and flag bit EFM_FSR.OTPWERRO will be set.

7.6.5 Series program

Series program mode is recommended when bulk data need to be programmed. Series program can save more than 50% time than single program. In series program mode, interval latency between two write data cannot exceed 16us.

The steps for setting series program are as follows:

- 1) release protection from access protection register. (write 0x0123 first, then 0x3210 to EFM_FAPRT)
- 2) unlock EFM_KEY1. (write 0x01234567 first, then 0xFEDCBA98 to EFM_KEY1)
- 3) set series program mode. (EFM_FWMC.PEMOD[2:0]=011)
- 4) release write protection. (write 1 to corresponding bit in EFM_F0/1NWPRTx (x=0~3))
- 5) writes 32-bit data to program address. (The program address needs to be a different FLASH from where CPU is executing.)
- 6) wait for operation end flag (EFM_FSR.OPTEND0/1) to be set.
- 7) clear operation end flag until read out of EFM_FSR.OPTEND0/1 is 0.
- 8) repeat 5), 6), 7) until all data has been written.
- 9) change series program mode to read only mode. (EFM_FWMC.PEMOD[2:0]=000)
- 10) wait for FLASH to be idle. (EFM_FSR.RDY0/1=1)
- 11) read the programmed address and compare whether it is the same as write data.
- 12) If same, it indicates program is successful, else it means that the FLASH address has been destroyed and permanently abandoned.

Program data to an OTP area that has been locked will fail, and flag bit EFM_FSR.OTPWERRO will be set.

7.6.6 Sector Erase

After the sector erase is performed, the data in the sector (8KBytes) is erased to all 1.

The steps for setting sector erase are as follows:

- 1) release protection from access protection register. (write 0x0123 first, then 0x3210 to EFM_FAPRT)
- 2) unlock EFM_KEY1. (write 0x01234567 first, then 0xFEDCBA98 to EFM_KEY1)
- 3) set sector erase mode (EFM_FWMC.PEMOD[2: 0]=100).
- 4) release write protection. (write 1 to corresponding bit in EFM_F0/1NWPRTx (x=0~3))
- 5) write an arbitrary 32-bit data to any address in erase sector (address should be aligned with 4).
- 6) wait for FLASH to be idle. (EFM_FSR.RDY0/1=1)
- 7) clear operation end flag. (EFM_FSR.OPTEND0/1)

Erase to an OTP area that has been locked will fail, and flag bit EFM_FSR.OTPWERRO will be set.

7.6.7 Mass erase

The production provides two mass erase modes: single FLASH mass erase mode and double FLASH mass erase mode. The steps for setting mass erase are as follows:

- 1) release protection from access protection register. (write 0x0123 first, then 0x3210 to EFM_FAPRT)
- 2) unlock EFM_KEY1. (write 0x01234567 first, then 0xFEDCBA98 to EFM_KEY1)
- 3) set EFM_FWMC.PEMOD[2:0]=101 if single FLASH mass erase mode
- 4) set EFM_FWMC.PEMOD[2:0]=110 if double FLASH mass erase mode
- 5) release write protection. (write all 1 to bits in EFM_F0/1NWPRTx (x=0~3))
- 6) single FLASH mass erase: write an arbitrary 32-bit data to any address in mass erase FLASH (address should be aligned with 4).
- 7) double FLASH mass erase: write an arbitrary 32-bit data to either FLASH0 or FLASH1 (the address needs to be aligned with 4).
- 8) wait for FLASH to be idle. (EFM_FSR.RDY0/1=1)
- 9) clear operation end flag. (EFM_FSR.OPTEND0/1)

After OTP is enabled, single mass erase to FLASH0 will fail, the flag bit EFM_FSR.OTPWERRO will be set. If double mass erase operations are sent, FLASH0 data will be retained, FLASH1 data is mass erased, and the flag bit EFM_FSR.OTPWERRO will be set.

7.6.8 Data security configuration

This product provides 4 levels of protection for FLASH data to prevent untrusted customers from reading and tampering with FLASH through the debug interface (JTAG and SWD), ISP interface (In System Program) and test interface.

Protection level 0: no protection

Debug interface, ISP interface and test interface can read and write all address including FLASH data.

Protection level 1:

Protection level 1 is enabled after FLASH address 0x0000_0430-0x0000_0433 has been programmed with 0xAF180402 and FLASH0 address 0x0300_4000~0x0300_400B has been programmed with 96-bit password.

If protection level 1 is enabled,

- debug interface, ISP interface and test interface cannot read and write all address except DBGC special register.
- sector 0 cannot be programmed and sector erased even by the customer program.
- the data in address 0x0300_4000~0x0300_400B cannot be read.

After activating protection level 1, it can be back to protection level 0 through password authentication and mass erase. But, mass erase will fail if OTP is enabled. Please consult the sales window for password authentication and mass erae method.

Protection level 2:

Protection level 2 is enabled after FLASH address 0x0000_0434-0x0000_0437 is programmed with 0xA85173AE.

If protection level 2 is enabled,

- debug interface, ISP interface and test interface cannot access MCU resources.
- sector 0 cannot be programmed and sector erased by the customer program.

After activation of protection level 2, it can be back to protection level 0 by mass erase. But, mass erase will fail if OTP is enabled.

Protection level 3:

Protection level 3 is enabled after FLASH addresses 0x0000_0420-0x0000_0423, 0x0000_0424-0x0000_0427, 0x0000_0428-0x0000_042B are programmed with 0x42545048,

If protection level 3 is enabled

- debug interface, ISP interface and test interface cannot write all address except DBGC special register.

- sector 0 cannot be programmed and sector erased even by the customer program.

After activating protection level 3, it can be back to protection level 0 by mass erase. But, mass erase will fail if OTP is enabled.

Protection level 1, protection level 2 and protection level 3 can be enabled individually or simultaneously. When enabled at the same time, the protection are superimposed.

7.6.9 D-BUS read protection

D-BUS read protection is provided for the address 0x0000_0000h~0x0001_FFFFh (128KBytes) space. It is enabled by setting 0x4450 to ICG3.DBUSPRT3[15:0]. When CPU-PC is within this 128KB space (that is CPU executes program in this area), CPU and DMAC can access the data in beginning 128KB area correctly; when CPU-PC is out of this 128K space (that is CPU executes program out of beginning 128KB), CPU and DMAC cann't read FLASH in beginning 128KB through D-BUS, and the read operation will return bus error. D-BUS is data access bus from CPU, including operand access, stack data access, general data access, etc. When D-BUS read protection is enabled, customers need to pay attention to the following matters.

- 1) single independent program cannot cross the boundary of 0x0002_0000h.
- 2) When CPU executes programs in area rather than 128KB space, it cannot read the interrupt vector table configured in the 128KB area.
- 3) When CPU executes programs in area rather than 128KB space, it can only be back to the 128K area by jumping instructions and subroutine calls.

7.6.10 BGO

If the target address of FLASH program and erase isn't within the same FLASH where C-program executes, bus hold control bit (EFM_FWMC.BUSHLDCTL) can be set to 1. Then, C-program is executing while FLASH is programming or erasing. The following are the steps for setting the BGO function.

- 1) release protection from access protection register. (write 0x0123 first, then 0x3210 to EFM_FAPR)
- 2) unlock EFM_KEY1. (write 0x01234567 first, then 0xFEDCBA98 to EFM_KEY1)
- 3) set program, erase mode. (EFM_FWMC.PEMODE[2:0]=001, 010, 011, 100, 101,110)
- 4) release write protection
- 5) set 1 to bus hold control (EFM_FWMC.BUSHLDCTL=1)
- 6) write 32-bit data to the target address
- 7) wait for the operation end flag to be set. (EFM_FSR.OPTEND0/1=1)

7.6.11 Interruption

The EFM module has a total of 3 interrupts, namely PEERR (programm/erase error) interruption, bus conflict interruption and operation end interrupt.

1. program erase error interruption EFM _ PEERR:

set condition:

- send a program or erase operation to a locked OTP address. (OTPWERRO=1)
- sector erase or mass erase to address 0x0300_0000~0x0300_01AB is performed after the OTP is enabled.(OTPWERRO=1).
- program or erase to write protected sectors (PRTWERRO/1=1).
- operaiton address is not aligned with 4 or the write data is not 32 bits (PGSZERRO/1=1).
- The data read out by hardware doesn't match the write data in single program with readback mode. (MISMTCH0/1=1).

clear condition:

write 1 to corresponding bit in register EFM_FSCLR

2. read and write conflict interruption EFM_COLER:

set condition:

- read or write to FLASH that is in busy status (RDY0=0 or RDY1=0). (except for writing in series program mode)

clear condition:

write 1 to corresponding bit in register EFM_FSCLR

3. end of operation interrupt EFM_OPTEND:

set condition:

- program : end of program (OPTEND0/1=1)..
- erase: end of sector erase or mass erase (OPTEND0/1=1)..

clear condition:

write 1 to corresponding bit in register EFM_FSCLR

7.7 One Time Program (OTP)

This product provides a maximum OTP area of 134KBytes, composed of sixteen 8KBytes, two 2KBytes, four 256Bytes, thirty-two 16Bytes, and one hundred twenty-eight 4Bytes. The address allocation is shown as follows.

Table 7-3 OTP address allocation table

Sector	OTP data address	Capacity	OTP lock address
0	0x0000_0000~0x0000_1FFF	8KB	0x0300_1800~0x0300_1803
1	0x0000_2000~0x0000_3FFF	8KB	0x0300_1804~0x0300_1807
.	.	.	.
14	0x0001_C000~0x0001_DFFF	8KB	0x0300_1838~0x0300_183B
15	0x0001_E000~0x0001_FFFF	8KB	0x0300_183C~0x0300_183F
16	0x0300_0000~0x0300_07FF	2KB	0x0300_1840~0x0300_1843
17	0x0300_0800~0x0300_0FFF	2KB	0x0300_1844~0x0300_1847
18	0x0300_1000~0x0300_10FF	256B	0x0300_1848~0x0300_184B
19	0x0300_1100~0x0300_11FF	256B	0x0300_184C~0x0300_184F
20	0x0300_1200~0x0300_12FF	256B	0x0300_1850~0x0300_1853
21	0x0300_1300~0x0300_13FF	256B	0x0300_1854~0x0300_1857
22	0x0300_1400~0x0300_140F	16B	0x0300_1858~0x0300_185B
.	.	.	.
53	0x0300_15F0~0x0300_15FF	16B	0x0300_18D4~0x0300_18D7
54	0x0300_1600~0x0300_1603	4B	0x0300_18D8~0x0300_18DB
.	.	.	.
181	0x0300_17FC~0x0300_17FF	4B	0x0300_1AD4~0x0300_1AD7

It is necessary to program 32 bits data that must include one bit 0 at least but recommended all 0 to address 0x0300_1AD8~0x0300_1ADB in order to enable OTP. When data of OTP lock address contains 0, the OTP data address cannot be programmed, sector erased or mass erased once more. Sector0~15 in OTP and spcae(0x0000_0000~0x0001_FFFF) in FLASH0 share the same 128KB physical FLASH. If the data of these lock address is all 1, this sector is normal FLASH area, which can be programmed and erased many times. Else, the sector is read only sector, and FLASH0 cannot be mass erased.

It is necessary to unlock protection from EFM_KEY2 before program OTP lock address. And unlocking EFM_KEY2 protection needs to write 0x10325476 first, then 0xEFCDAB89. If a wrong sequence is written to EFM_KEY2, a bus error will occur and the EFM_KEY2 register will be self-locked until reset occurs.

To set OTP area, please follow the steps below:

- 1) release protection from access protection register. (write 0x0123 first, then 0x3210 to EFM_FAPRT)
- 2) unlock EFM_KEY1. (write 0x01234567 first, then 0xFEDCBA98 to EFM_KEY1)
- 3) set programmode. (EFM_FWMC.PEMODE[2:0]=001, 010, 011).
- 4) write data to OTP data address
- 5) unlock EFM_KEY2. (write 0x10325476 first, then 0xEFCDAB89 to EFM_KEY2)
- 6) write an arbitrary 32-bit data that includes one bit 0 at least to OTP enable address 0x0300_1AD8~0x0300_1ADB. (recommend to write 32bit 0)
- 7) write an arbitrary 32-bit data that includes one bit 0 at least to OTP lock address (recommend to write 32bit 0)

After reset is released, hardware will automatically load data from OTP lock address to the circuit. Therefore, if the address 0x0300_1AD8~0x0300_1ADB has been programmed before, step 6 should omit.

Program, sector erase or mass erase to OTP data address whose lock address has been programmed will occur the OTPWERR0 error.

After having programmed OTP lock address, it is possible to write 1 to EFM_FWMC.KEY2LOCK in order to get EFM_KEY2 locked state back.

7.8 Boot swap

Boot swap can be realized by Programming 0x5A5A5A to FLASH address (0x0300_2000). If OTP is disabled (data in 0x0300_1AD8~0x0300_1ADB is all 1), the whole address of FLASH0 and FLASH1 will be swapped, in that case, the OTP enable address cannot be programmed. And status bit EFM_FSWP.FSWP will set to 1.

If OTP is enabled (data in 0x0300_1AD8~0x0300_1ADB isn't all 1), the addresses of FLASH0 and FLASH1 are partially swapped. And, EFM_FSWP.FSWP is 0 in this case. Detail please see below Figure 7-2 , Figure 7-3 .

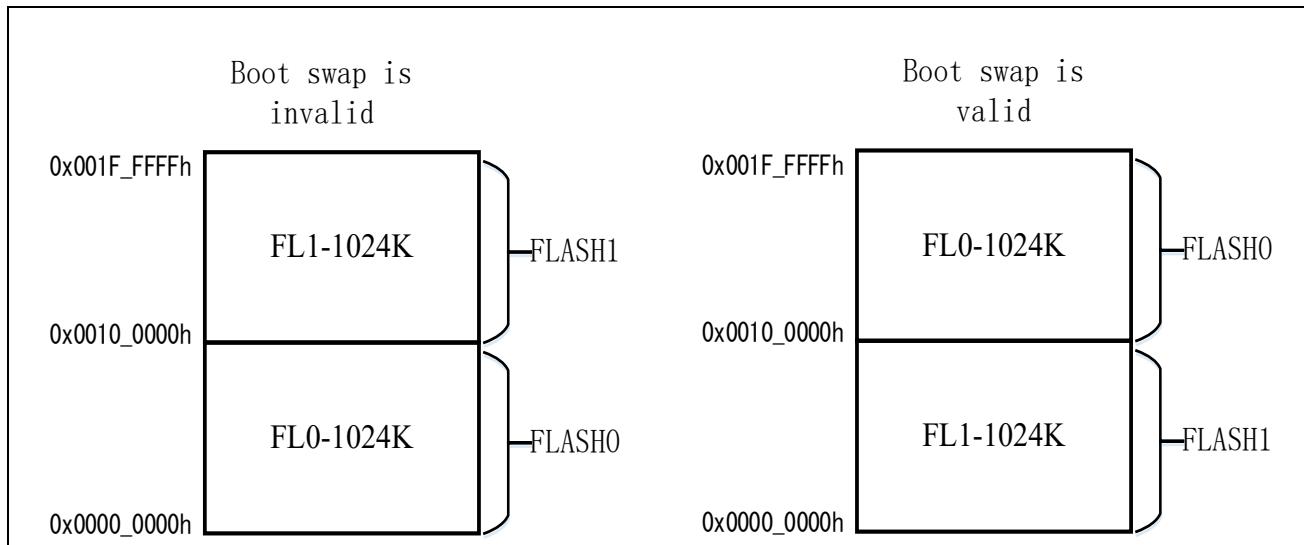


Figure 7-2 boot swap of FLASH address when OTP is disabled

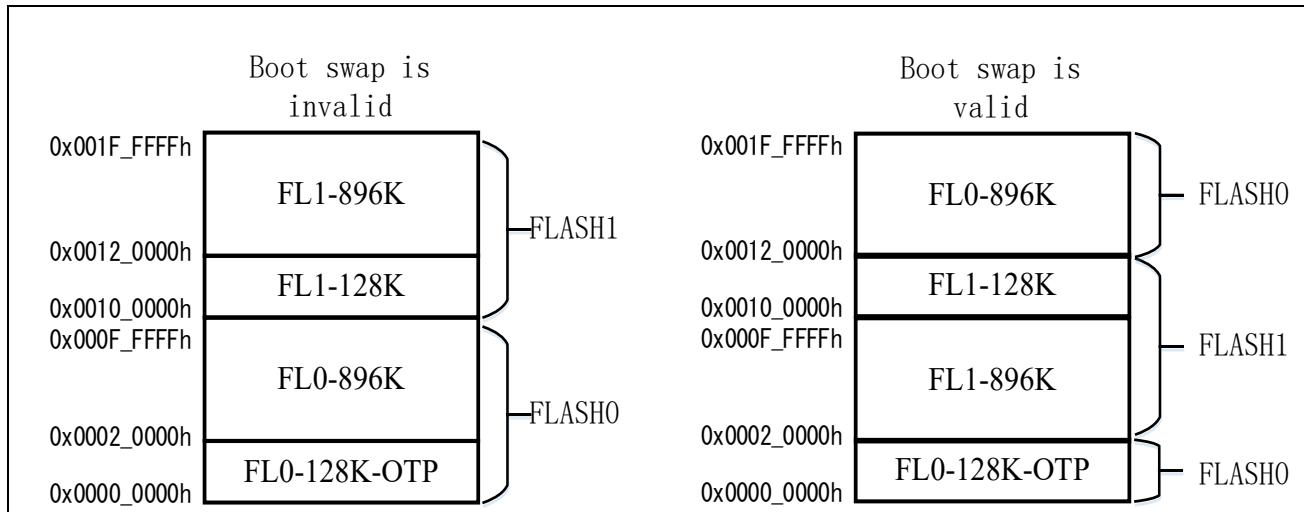


Figure 7-3 boot swap of FLASH address when OTP is enabled

The following takes the boot exchange without OTP enabled as an example to introduce the usage of this function Figure 7-4 .

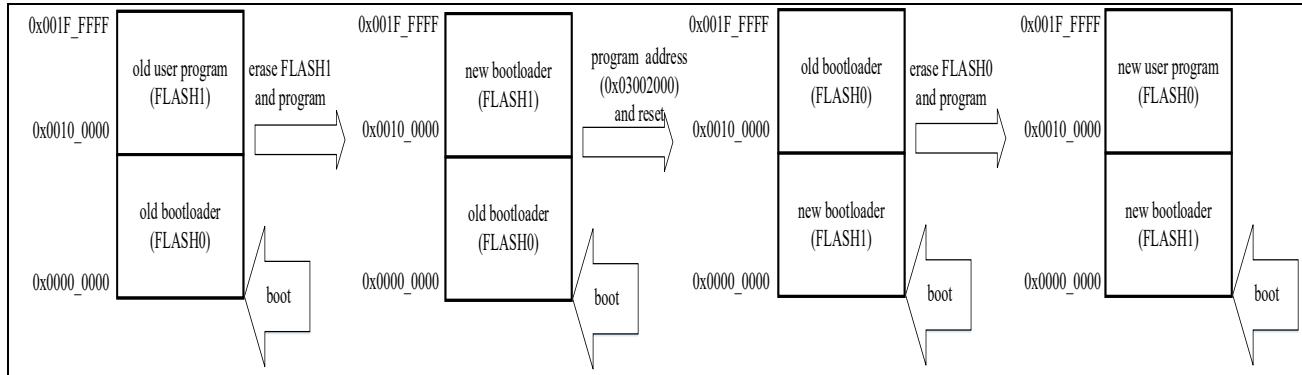


Figure 7-4 Boot Swap 1

When customer needs to update bootloader program again, since the address 0x0300_2000 has been programmed, it needs to be erased for updating. The procedure is shown as Figure 7-5 .

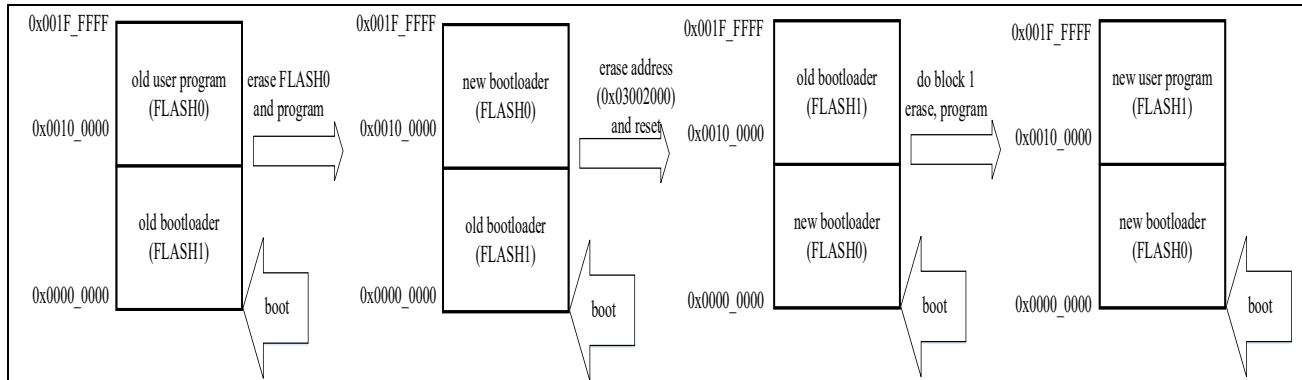


Figure 7-5 Boot Swap 2

7.9 Register description

EFM_BASE_ADDR: 0x40010400

Table 7-4 List of Registers

Register description	Register name	Offset	Bitwidth	Reset value
FLASH access protection register	EFM_FAPRT	0x0000h	32	0x0000_0000
FLASH KEY1 register	EFM_KEY1	0x0004h	32	0x0000_0000
FLASH KEY2 register	EFM_KEY2	0x0008h	32	0x0000_0000
FLASH stop register	EFM_FSTP	0x0014h	32	0x0000_0000
FLASH read mode register	EFM_FRMC	0x0018h	32	0x0000_0000
FLASH write mode register	EFM_FWMC	0x001Ch	32	0x0003_0000
FLASH status register	EFM_FSR	0x0020h	32	0x0100_0100
FLASH state clear register	EFM_FSCLR	0x0024h	32	0x0000_0000
FLASH interruption enable register	EFM_FITE	0x0028h	32	0x0000_0000
FLASH boot swap status Register	EFM_FSWP	0x002Ch	32	reset value
CHIPID register	EFM_CHIPID	0x0040h	32	0x4844_04A0
uniqueID register 0	FEM_UQID0	0x0050h	32	reset value
uniqueID register 1	EFM_UQID1	0x0054h	32	reset value
uniqueID register 2	EFM_UQID2	0x0058h	32	reset value
FLASH write protection lock register	EFM_WLOCK	0x0180h	32	0x0000_0000
FLASH0 write protection register 0	EFM_F0NWPRT0	0x0190h	32	0x0000_0000
FLASH0 write protection register 1	EFM_F0NWPRT1	0x0194h	32	0x0000_0000
FLASH0 write protection register 2	EFM_F0NWPRT2	0x0198h	32	0x0000_0000
FLASH0 write protection register 3	EFM_F0NWPRT3	0x019Ch	32	0x0000_0000
FLASH1 write protection register 0	EFM_F1NWPRT0	0x01A0h	32	0x0000_0000
FLASH1 write protection register 1	EFM_F1NWPRT1	0x01A4h	32	0x0000_0000
FLASH1 write protection register 2	EFM_F1NWPRT2	0x01A8h	32	0x0000_0000
FLASH1 write protection register 3	EFM_F1NWPRT3	0x01ACh	32	0x0000_0000

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]															

Bit	Symbol	Bit name	Description	Read and write
b31-b16	Reserved	-	Read as "0", write as "0"	R/W
b15-b0	FAPRT[15:0]	EFM register access protection	EFM register access protection register. All the registers in EFM is write disabled before releasing access protection. release: write "0x0123" first and then "0x3210". When register access protection is valid, the read value of this register is 0x00000000. When register access protection is invalid, the read value of this register is 0x00000001.	R/W

7.9.2 FLASH KEY 1 Register (EFM_KEY1)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
KEY1[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
KEY1[15:0]															

Bit	Symbol	Bit name	Description	Read and write
b31-b0	KEY1[31:0]	key 1 register	write protection for EFM_FWMC register. write 0x01234567 first, and then 0xFEDCBA98 to release the write protection. The read value of this register is 0x00000000.	R/W

7.9.3 FLASH KEY 2 register (EFM_KEY2)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
KEY2[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
KEY2[15:0]															

Bit	Symbol	Bit name	Description	Read and write
b31-b0	KEY2[31:0]	key 2 register	write protection register for OTP lock addresses. write 0x10325476 first, and then 0xEFCDAB89 to release the write protection. The read value of this register is 0x00000000.	R/W

7.9.4 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
-	-	-	-	-	-	-	-	-	-	-	-	-	-	F1STP	F0STP

Bit	Symbol	Bit name	Description	Read and write
b31~b2	Reserved	-	Read as "0", write as "0"	R/W
b1	F1STP	FLASH1 stop	0: FLASH1 active state 1: FLASH1 is in stop mode When this bit changes from 1 to 0, please make sure that the FSR.RDY1 bit is 1 before accessing FLASH1.	R/W
b0	F0STP	FLASH0 stop	0: FLASH0 active state 1: FLASH0 is in stop mode When this bit changes from 1 to 0, please make sure that the FSR.RDY0 bit is 1 before accessing FLASH0.	R/W

7.9.5 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
-	-	-	-	-	-	-	-	-	-	-	-	CRST	PREFETE	DCA CHE	ICAC HE
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	LVM	-	-	-	-	-	-	-	FLWT[3:0]

Bit	Symbol	Bit name	Description	Read and write
b31~b20	Reserved	-	Read as "0", write as "0"	R/W
b19	CRST	Cache reset	0: Cached data isn't reset 1: Reset cache data (ICODE and DCODE)	R/W
b18	PREFETE	ICODE prefetch enable	0: Disable ICODE prefetch 1: Enable ICODE prefetch	R/W
b17	DCACHE	DCODE cache enable	0: Disable DCODE cache 1: Enable DCODE cache	R/W
b16	ICACHE	ICODE cache enable	0: Disable ICODE cache 1: Enable ICODE cache	R/W
b15~b9	Reserved	-	Read as "0", write as "0"	R/W
b8	LVM	Low voltage read mode	0: Normal voltage read mode 1: Low voltage read mode it needs to be set to 1 in ultra low speed mode	R/W
b7-b4	Reserved	-	Read as "0", write as "0"	R/W
b3~b0	FLWT[3:0]	Waiting cycles of FLASH read	4'b0000: No wait cycles 4'b0001: Insert 1 wait cycle 4'b0010: Insert 2 wait cycles 4'b1110: Insert 14 wait cycles 4'b1111: Insert 15 wait cycles	R/W

7.9.6 Erase mode register (EFM_FWMC)

Reset value: 0x0003_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	KEY2 LOCK	KEY1LOCK
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	BUS HLDC TL	-	-	-	-	-	-	PEMOD[2:0]	

Bit	Symbol	Bit name	Description	Read and write
b31~b18	Reserved	-	Read as "0", write as "0"	R/W
b17	KEY2LOCK	KEY2 protection bit	0: Protection from EFM_KEY2 is invalid. 1: Protection from EFM_KEY2 is valid. This bit can only be written 1. When EFM_KEY2 is written with an error sequence, this bit remains 1 until reset.	R/W
b16	KEY1LOCK	KEY1 protection bit	0: Protection from EFM_KEY1 is invalid. 1: Protection from EFM_KEY1 is invalid. This bit can only be written 1. When EFM_KEY1 is written with an error sequence, this bit remains 1 until reset.	R/W
b15~b9	Reserved	-	Read as "0", write as "0"	R/W
b8	BUSHLDCTL	bus hang on control while programming or erasing FLASH	0: The bus is hanged on during programming and erasing 1: The bus isn't hanged on during programming and erasing.	R/W
b7~b3	Reserved	-	Read as "0", write as "0"	R/W
b2~b0	PEMOD[2:0]	FLASH program and erase mode	3'b000: read-only 3'b001: single program without readback 3'b010: single program with readback 3'b011: series program 3'b100: sector erase 3'b101: single FLASH mass erase 3'b110: double FLASH mass erase 3'b111: read only	R/W

7.9.7 Status register (EFM_FSR)

Reset value: 0x0100_0100

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	RDY1	-	-	COLE RR1	OPTE ND1	MISM TCH1	PGS ZERR 1	PRT WER R1	-
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	RDY0	-	-	COLE RRO	OPTE NDO	MISM TCHO	PGS ZERR 0	PRT WER R0	OTP WER R0

Bit	Symbol	Bit name	Description	Read and write
b31~b25	Reserved	-	Read as "0", write as "0"	R/W
b24	RDY1	FLASH1 ready state	0: FLASH1 is in busy 1: FLASH1 is in ready	R
b23~b22	Reserved	-	Read as "0", write as "0"	R/W
b21	COLERR1	read and write conflict error of FLASH1	set conditions: read or write to FLASH1 when RDY1=0 clear condition: write 1 to the clear bit.	R
b20	OPTEND1	operation end of FLASH1	set conditions: end of program and erase Clear condition: writing 1 to the clear bit.	R
b19	MISMTCH1	mismatch in single program with readback mode of FLASH1	set conditions: read out data is different from the write data. clear condition: write 1 to the clear bit	R
b18	PGSZERR1	program size error of FLASH1	set conditions: The programmed address isn't aligned with 4, or a byte/half-word write has occurred. clear condition: write 1 to the clear bit	R
b17	PRTWERR1	write protection of FLASH1	set conditions: send program and erase operation to write protected sectors. clear condition: write 1 to the clear bit	R
b16~b9	Reserved	-	Read as "0", write as "0"	R/W
b8	RDY0	FLASH0 ready state	0: FLASH0 is in busy 1: FLASH0 is in ready	R
b7~b6	Reserved	-	Read as "0", write as "0"	R/W
b5	COLERR0	read and write conflict error of FLASH0	set conditions: read or write to FLASH1 when RDY1=0 clear condition: write 1 to the clear bit.	R
b4	OPTEND0	operation end of FLASH0	set conditions: end of program and erase Clear condition: writing 1 to the clear bit.	R
b3	MISMTCH0	mismatch in single program with readback mode of FLASH0	set conditions: read out data is different from the write data. clear condition: write 1 to the clear bit	R
b2	PGSZERRO	program size error of FLASH0	set conditions: The programmed address isn't aligned with 4, or a byte/half-word write has occurred. clear condition: write 1 to the clear bit	R
b1	PRTWERR0	write protection of FLASH0	set conditions: send program and erase operation	R

			to write protected sectors. clear condition: write 1 to the clear bit	
b0	OTPWERRO	OTP error of FLASH0	set conditions: send program and erase operations to an OTP locked address Clear condition: write 1 to the clear bit	R

7.9.8 Status purge register (EFM_FSCLR)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	COLE RRCL R1	OPTE NDCL R1	MISM TCHC LR1	PGS ZERR CLR1	PRT WER RCLR 1	-
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	-	-	-	COLE RRCL R0	OPTE NDCL R0	MISM TCHC LR0	PGS ZERR CLR0	PRT WER RCLR 0	OTP WER RCLR 0

Bit	Symbol	Bit name	Description	Read and write
b31~b22	Reserved	-	Read as "0", write as "0"	R/W
b21	COLERRCLR1	clear COLERR1	0: no action 1: clear the FSR.COLERR1 it is always 0 when reading.	R/W
b20	OPTENDCLR1	clear OPTEND1	0: no action 1: clear the FSR.OPTEND1 it is always 0 when reading.	R/W
b19	MISMTCHCLR1	clear MISMTCH1	0: no action 1: clear the FSR.MISMTCH1 it is always 0 when reading	R/W
b18	PGSZERRCLR1	clear PGSZERR1	0: no action 1: clear the FSR.PGSZERR1 it is always 0 when reading	R/W
b17	PRTERRCLR1	clear PRTERR1	0: no action 1: clear the FSR.PRTWERR1 it is always 0 when reading	R/W
b16~b6	Reserved	-	Read as "0", write as "0"	R/W
b5	COLERRCLR0	clear COLERR0	0: no action 1: clear the FSR.COLERR0 it is always 0 when reading.	R/W
b4	OPTENDCLR0	clear OPTEND0	0: no action 1: clear the FSR.OPTEND0 it is always 0 when reading.	R/W
b3	MISMTCHCLR0	clear MISMTCH0	0: no action 1: clear the FSR.MISMTCH0 it is always 0 when reading.	R/W
b2	PGSZERRCLR0	clear PGSZERR0	0: no action 1: clear the FSR.PGSZERR0 it is always 0 when reading.	R/W
b1	PRTERRCLR0	clear PRTERR0	0: no action 1: clear the FSR.PRTWERR0 it is always 0 when reading.	R/W
b0	OTPWERRCLR0	clear OTPWERR0	0: no action 1: clear the FSR.OTPWERR0 it is always 0 when reading.	R/W

7.9.9 Interruption enable 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
-	-	-	-	-	-	-	-	-	-	-	-	-	COLE RRITE	OPT ENDIT E	PEER RITE

Bit	Symbol	Bit name	Description	Read and write
b31~b3	Reserved	-	Read as "0", write as "0"	R/W
b2	COLERRITE	read and write conflict error interruption enable	0: Read and write conflict error interruption is disabled. 1: Read and write conflict error interruption is enabled.	R/W
b1	OPTENDITE	End of operation interruption enable	0: operation end interruption is disabled. 1: operation end interruption is enable.	R/W
b0	PEERRITE	program and erase error interruption enable	0: program and erase interruption is disabled. 1: program and erase interruption is enable.	R/W

7.9.10 Boot Swap Status Register (EFM_FSWP)

Reset value: reset value

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
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	FSW P

Bit	Symbol	Bit name	Description	Read and write
b31-b1	Reserved	-	Read as "0", write as "0"	R/W
b0	FSWP	FLASH0 and FLASH1 swap status	0: ICG is loaded from FLASH0. 1: ICG is loaded from FLASH1 The reset value of the register is determined by the value of FLASH address 0x03002000~0x03002003 and whether OTP is enabled.	R

7.9.11 CHIPID register (EFM_CHIPID)

Reset value: 0x4844_04A0

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CHIPID[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
CHIPID[15:0]															

Bit	Symbol	Bit name	Description	Read and write
b31-b0	CHIPID[31:0]	chip identity	CHIPID[31:16]: 0x4844 CHIPID[15:0]: 04A0, product type name This register is read only.	R

7.9.12 Unique ID register (EFM_UQID 0)

Reset value: reset value

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
UQID0[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
UQID0[15:0]															

Bit	Symbol	Bit name	Description	Read and write
b31-b0	UQID0[31:0]	Unique code	Chip unique code	R

7.9.13 Unique ID register (EFM_UQID 1)

Reset value: reset value

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]															

Bit	Symbol	Bit name	Description	Read and write
b31-b0	UQID1[31:0]	Unique code	Chip unique code	R

7.9.14 Unique ID register (EFM_UQID 2)

Reset value: reset value

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]															
<hr/>															
Bit	Symbol		Bit name												Read and write
b31-b0	UQID2[31:0]														R

7.9.15 FLASH write protection lock register (EFM_WLOCK)

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
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	WLOCK[7:0]

Bit	Symbol	Bit name	Description	Read and write
b31-b8	Reserved	-	Read as "0", write as "0"	R/W
B7	WLOCK[7]	F1NWPRT3 register lock bit	0: The sector write protection controlled by F1NWPRT3 is not locked. 1: The sector write protection controlled by F1NWPRT3 is locked. Once this bit is set to 1, it can only be restored to 0 by reset.	R/W
b6	WLOCK[6]	F1NWPRT2 register lock bit	0: The sector write protection controlled by F1NWPRT2 is not locked. 1: The sector write protection controlled by F1NWPRT2 is locked. Once this bit is set to 1, it can only be restored to 0 by reset.	R/W
b5	WLOCK[5]	F1NWPRT1 register lock bit	0: The sector write protection controlled by F1NWPRT1 is not locked. 1: The sector write protection controlled by F1NWPRT1 is locked. Once this bit is set to 1, it can only be restored to 0 by reset.	R/W
b4	WLOCK[4]	F1NWPRT0 register lock bit	0: The sector write protection controlled by F1NWPRT0 is not locked. 1: The sector write protection controlled by F1NWPRT0 is locked. Once this bit is set to 1, it can only be restored to 0 by reset.	R/W
b3	WLOCK[3]	F0NWPRT3 register lock bit	0: The sector write protection controlled by F0NWPRT3 is not locked. 1: The sector write protection controlled by F0NWPRT3 is locked. Once this bit is set to 1, it can only be restored to 0 by reset.	R/W
b2	WLOCK[2]	F0NWPRT2 register lock bit	0: The sector write protection controlled by F0NWPRT2 is not locked. 1: The sector write protection controlled by F0NWPRT2 is locked. Once this bit is set to 1, it can only be restored to 0 by reset.	R/W
b1	WLOCK[1]	F0NWPRT1 register lock bit	0: The sector write protection controlled by F0NWPRT1 is not locked. 1: The sector write protection controlled by F0NWPRT1 is locked. Once this bit is set to 1, it can only be restored to 0 by reset.	R/W
b0	WLOCK[0]	F0NWPRT0 register lock bit	0: The sector write protection controlled by F0NWPRT0 is not locked. 1: The sector write protection controlled by F0NWPRT0 is locked. Once this bit is set to 1, it can only be restored to 0 by reset.	R/W

7.9.16 FLASH0 write protection register 0 (EFM_FONWPRT0)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
FON WPRT 31	FON WPRT 30	FON WPRT 29	FOW PRT2	FON WPRT 8	FON WPRT 27	FON WPRT 26	FON WPRT 25	FON WPRT 24	FON WPRT 23	FON WPRT 22	FON WPRT 21	FON WPRT 20	FON WPRT 19	FON WPRT 18	FON WPRT 17	FON WPRT 16
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0	
FON WPRT 15	FON WPRT 14	FON WPRT 13	FON WPRT 12	FON WPRT 11	FON WPRT 10	FON WPRT 9	FON WPRT 8	FON WPRT 7	FON WPRT 6	FON WPRT 5	FON WPRT 4	FON WPRT 3	FON WPRT 2	FON WPRT 1	FON WPRT 0	

Bit	Symbol	Bit name	Description	Read and write
b31	FONWPRT31	write protection of sector 31 in FLASH0	0: write protection is valid of sector31 in FLASH0 1: write protection is invalid of sector31 in FLASH0 When WLOCK[0]=0, 0 and 1 can be written. When WLOCK[0]=1, only 0 can be written.	R/W
b30	FONWPRT30	write protection of sector 30 in FLASH0	0: write protection is valid of sector30 in FLASH0 1: write protection is invalid of sector30 in FLASH0 When WLOCK[0]=0, 0 and 1 can be written. When WLOCK[0]=1, only 0 can be written.	R/W
b29~b2	R/W
b1	FONWPRT1	write protection of sector1 in FLASH0	0: write protection is valid of sector1 in FLASH0 1: write protection is invalid of sector1 in FLASH0 When WLOCK[0]=0, 0 and 1 can be written. When WLOCK[0]=1, only 0 can be written.	R/W
b0	FONWPRT0	write protection of sector0 in FLASH0	0: write protection is valid of sector0 in FLASH0 1: write protection is invalid of sector0 in FLASH0 When WLOCK[0]=0, 0 and 1 can be written. When WLOCK[0]=1, only 0 can be written.	R/W

7.9.17 FLASH0 write protection register 1 (EFM_FONWPRT1)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
FON WPRT 63	FON WPRT 62	FON WPRT 61	FON WPRT 60	FON WPRT 59	FON WPRT 58	FON WPRT 57	FON WPRT 56	FON WPRT 55	FON WPRT 54	FON WPRT 53	FON WPRT 52	FON WPRT 51	FON WPRT 50	FON WPRT 49	FON WPRT 48
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
FON WPRT 47	FON WPRT 46	FON WPRT 45	FON WPRT 44	FON WPRT 43	FON WPRT 42	FON WPRT 41	FON WPRT 40	FON WPRT 39	FON WPRT 38	FON WPRT 37	FON WPRT 36	FON WPRT 35	FON WPRT 34	FON WPRT 33	FON WPRT 32

Bit	Symbol	Bit name	Description	Read and write
b31	FONWPRT63	write protection of sector 63 in FLASH0	0: write protection is valid of sector63 in FLASH0 1: write protection is invalid of sector63 in FLASH0 When WLOCK[1]=0, 0 and 1 can be written. When WLOCK[1]=1, only 0 can be written.	R/W
b30	FONWPRT62	write protection of sector 62 in FLASH0	0: write protection is valid of sector62 in FLASH0 1: write protection is invalid of sector62 in FLASH0 When WLOCK[1]=0, 0 and 1 can be written. When WLOCK[1]=1, only 0 can be written.	R/W
b29~b2	R/W
b1	FONWPRT33	write protection of sector 33in FLASH0	0: write protection is valid of sector33in FLASH0 1: write protection is invalid of sector33 in FLASH0 When WLOCK[1]=0, 0 and 1 can be written. When WLOCK[1]=1, only 0 can be written.	R/W
b0	FONWPRT32	write protection of sector 32 in FLASH0	0: write protection is valid of sector32 in FLASH0 1: write protection is invalid of sector32 in FLASH0 When WLOCK[1]=0, 0 and 1 can be written. When WLOCK[1]=1, only 0 can be written.	R/W

7.9.18 FLASH0 write protection register 2 (EFM_FONWPRT2)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
F0N WPRT 95	F0N WPRT 94	F0N WPRT 93	F0N WPRT 92	F0N WPRT 91	F0N WPRT 90	F0N WPRT 89	F0N WPRT 88	F0N WPRT 87	F0N WPRT 86	F0N WPRT 85	F0N WPRT 84	F0N WPRT 83	F0N WPRT 82	F0N WPRT 81	F0N WPRT 80
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
F0N WPRT 79	F0N WPRT 78	F0N WPRT 77	F0N WPRT 76	F0N WPRT 75	F0N WPRT 74	F0N WPRT 73	F0N WPRT 72	F0N WPRT 71	F0N WPRT 70	F0N WPRT 69	F0N WPRT 68	F0N WPRT 67	F0N WPRT 66	F0N WPRT 65	F0N WPRT 64

Bit	Symbol	Bit name	Description	Read and write
b31	F0NWPRT95	write protection of sector 95 in FLASH0	0: write protection is valid of sector95 in FLASH0 1: write protection is invalid of sector95 in FLASH0 When WLOCK[2]=0, 0 and 1 can be written. When WLOCK[2]=1, only 0 can be written.	R/W
b30	F0NWPRT94	write protection of sector 94 in FLASH0	0: write protection is valid of sector94 in FLASH0 1: write protection is invalid of sector94 in FLASH0 When WLOCK[2]=0, 0 and 1 can be written. When WLOCK[2]=1, only 0 can be written.	R/W
b29~b2	R/W
b1	F0NWPRT65	write protection of sector 65 in FLASH0	0: write protection is valid of sector65 in FLASH0 1: write protection is invalid of sector65 in FLASH0 When WLOCK[2]=0, 0 and 1 can be written. When WLOCK[2]=1, only 0 can be written.	R/W
b0	F0NWPRT64	write protection of sector 64 in FLASH0	0: write protection is valid of sector64 in FLASH0 1: write protection is invalid of sector64 in FLASH0 When WLOCK[2]=0, 0 and 1 can be written. When WLOCK[2]=1, only 0 can be written.	R/W

7.9.19 FLASH0 write protection register 3 (EFM_FONWPRT3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
FON WPRT 127	FON WPRT 126	FON WPRT 125	FON WPRT 124	FON WPRT 123	FON WPRT 122	FON WPRT 121	FON WPRT 120	FON WPRT 119	FON WPRT 118	FON WPRT 117	FON WPRT 116	FON WPRT 115	FON WPRT 114	FON WPRT 113	FON WPRT 112
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
FON WPRT 111	FON WPRT 110	FON WPRT 109	FON WPRT 108	FON WPRT 107	FON WPRT 106	FON WPRT 105	FON WPRT 104	FON WPRT 103	FON WPRT 102	FON WPRT 101	FON WPRT 100	FON WPRT 99	FON WPRT 98	FON WPRT 97	FON WPRT 96

Bit	Symbol	Bit name	Description	Read and write
b31	FONWPRT127	write protection of sector 127 in FLASH0	0: write protection is valid of sector127 in FLASH0 1: write protection is invalid of sector127 in FLASH0 When WLOCK[3]=0, 0 and 1 can be written. When WLOCK[3]=1, only 0 can be written.	R/W
b30	FONWPRT126	write protection of sector 126 in FLASH0	0: write protection is valid of sector126 in FLASH0 1: write protection is invalid of sector126 in FLASH0 When WLOCK[3]=0, 0 and 1 can be written. When WLOCK[3]=1, only 0 can be written.	R/W
b29~b2	R/W
b1	FONWPRT97	write protection of sector 97 in FLASH0	0: write protection is valid of sector97 in FLASH0 1: write protection is invalid of sector97 in FLASH0 When WLOCK[3]=0, 0 and 1 can be written. When WLOCK[3]=1, only 0 can be written.	R/W
b0	FONWPRT96	write protection of sector 96 in FLASH0	0: write protection is valid of sector96 in FLASH0 1: write protection is invalid of sector96 in FLASH0 When WLOCK[3]=0, 0 and 1 can be written. When WLOCK[3]=1, only 0 can be written.	R/W

7.9.20 FLASH1 write protection register 0 (EFM_F1NWPRT0)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
F1N WPRT 31	F1N WPRT 30	F1N WPRT 29	F1N WPRT 28	F1N WPRT 27	F1N WPRT 26	F1N WPRT 25	F1N WPRT 24	F1N WPRT 23	F1N WPRT 22	F1N WPRT 21	F1N WPRT 20	F1N WPRT 19	F1N WPRT 18	F1N WPRT 17	F1N WPRT 16
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
F1N WPRT 15	F1N WPRT 14	F1N WPRT 13	F1N WPRT 12	F1N WPRT 11	F1N WPRT 10	F1N WPRT 9	F1N WPRT 8	F1N WPRT 7	F1N WPRT 6	F1N WPRT 5	F1N WPRT 4	F1N WPRT 3	F1N WPRT 2	F1N WPRT 1	F1N WPRT 0

Bit	Symbol	Bit name	Description	Read and write
b31	F1NWPRT31	write protection of sector 31 in FLASH1	0: write protection is valid of sector31 in FLASH1 1: write protection is invalid of sector31 in FLASH1 When WLOCK[4]=0, 0 and 1 can be written. When WLOCK[4]=1, only 0 can be written.	R/W
b30	F1NWPRT30	write protection of sector 30 in FLASH1	0: write protection is valid of sector30 in FLASH1 1: write protection is invalid of sector30 in FLASH1 When WLOCK[4]=0, 0 and 1 can be written. When WLOCK[4]=1, only 0 can be written.	R/W
b29~b2	R/W
b1	F1NWPRT1	write protection of sector1 in FLASH1	0: write protection is valid of sector1 in FLASH1 1: write protection is invalid of sector1 in FLASH1 When WLOCK[4]=0, 0 and 1 can be written. When WLOCK[4]=1, only 0 can be written.	R/W
b0	F1NWPRT0	write protection of sector0 in FLASH1	0: write protection is valid of sector0 in FLASH1 1: write protection is invalid of sector0 in FLASH1 When WLOCK[4]=0, 0 and 1 can be written. When WLOCK[4]=1, only 0 can be written.	R/W

7.9.21 FLASH1 write protection register 1 (EFM_F1NWPRT1)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
F1N WPRT 63	F1N WPRT 62	F1N WPRT 61	F1N WPRT 60	F1N WPRT 59	F1N WPRT 58	F1N WPRT 57	F1N WPRT 56	F1N WPRT 55	F1N WPRT 54	F1N WPRT 53	F1N WPRT 52	F1N WPRT 51	F1N WPRT 50	F1N WPRT 49	F1N WPRT 48
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
F1N WPRT 47	F1N WPRT 46	F1N WPRT 45	F1N WPRT 44	F1N WPRT 43	F1N WPRT 42	F1N WPRT 41	F1N WPRT 40	F1N WPRT 39	F1N WPRT 38	F1N WPRT 37	F1N WPRT 36	F1N WPRT 35	F1N WPRT 34	F1N WPRT 33	F1N WPRT 32

Bit	Symbol	Bit name	Description	Read and write
b31	F1NWPRT63	write protection of sector 63 in FLASH1	0: write protection is valid of sector63 in FLASH1 1: write protection is invalid of sector63 in FLASH1 When WLOCK[5]=0, 0 and 1 can be written. When WLOCK[5]=1, only 0 can be written.	R/W
b30	F1NWPRT62	write protection of sector 62 in FLASH1	0: write protection is valid of sector62 in FLASH1 1: write protection is invalid of sector62 in FLASH1 When WLOCK[5]=0, 0 and 1 can be written. When WLOCK[5]=1, only 0 can be written.	R/W
b29~b2	R/W
b1	F1NWPRT33	write protection of sector 33 in FLASH1	0: write protection is valid of sector33 in FLASH1 1: write protection is invalid of sector33 in FLASH1 When WLOCK[5]=0, 0 and 1 can be written. When WLOCK[5]=1, only 0 can be written.	R/W
b0	F1NWPRT32	write protection of sector 32 in FLASH1	0: write protection is valid of sector32 in FLASH1 1: write protection is invalid of sector32 in FLASH1 When WLOCK[5]=0, 0 and 1 can be written. When WLOCK[5]=1, only 0 can be written.	R/W

7.9.22 FLASH1 write protection register 2 (EFM_F1NWPRT2)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
F1N WPRT 95	F1N WPRT 94	F1N WPRT 93	F1N WPRT 92	F1N WPRT 91	F1N WPRT 90	F1N WPRT 89	F1N WPRT 88	F1N WPRT 87	F1N WPRT 86	F1N WPRT 85	F1N WPRT 84	F1N WPRT 83	F1N WPRT 82	F1N WPRT 81	F1N WPRT 80
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
F1N WPRT 79	F1N WPRT 78	F1N WPRT 77	F1N WPRT 76	F1N WPRT 75	F1N WPRT 74	F1N WPRT 73	F1N WPRT 72	F1N WPRT 71	F1N WPRT 70	F1N WPRT 69	F1N WPRT 68	F1N WPRT 67	F1N WPRT 66	F1N WPRT 65	F1N WPRT 64

Bit	Symbol	Bit name	Description	Read and write
b31	F1NWPRT95	write protection of sector 95 in FLASH1	0: write protection is valid of sector95 in FLASH1 1: write protection is invalid of sector95 in FLASH1 When WLOCK[6]=0, 0 and 1 can be written. When WLOCK[6]=1, only 0 can be written.	R/W
b30	F1NWPRT94	write protection of sector 94 in FLASH1	0: write protection is valid of sector94 in FLASH1 1: write protection is invalid of sector94 in FLASH1 When WLOCK[6]=0, 0 and 1 can be written. When WLOCK[6]=1, only 0 can be written.	R/W
b29~b2	R/W
b1	F1NWPRT65	write protection of sector 65 in FLASH1	0: write protection is valid of sector65 in FLASH1 1: write protection is invalid of sector65 in FLASH1 When WLOCK[6]=0, 0 and 1 can be written. When WLOCK[6]=1, only 0 can be written.	R/W
b0	F1NWPRT64	write protection of sector 64 in FLASH1	0: write protection is valid of sector64 in FLASH1 1: write protection is invalid of sector64 in FLASH1 When WLOCK[6]=0, 0 and 1 can be written. When WLOCK[6]=1, only 0 can be written.	R/W

7.9.23 FLASH1 write protection register 3 (EFM_F1NWPRT3)

Reset value: 0x0000_0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
F1N WPRT 127	F1N WPRT 126	F1N WPRT 125	F1N WPRT 124	F1N WPRT 123	F1N WPRT 122	F1N WPRT 121	F1N WPRT 120	F1N WPRT 119	F1N WPRT 118	F1N WPRT 117	F1N WPRT 116	F1N WPRT 115	F1N WPRT 114	F1N WPRT 113	F1N WPRT 112
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
F1N WPRT 111	F1N WPRT 110	F1N WPRT 109	F1N WPRT 108	F1N WPRT 107	F1N WPRT 106	F1N WPRT 105	F1N WPRT 104	F1N WPRT 103	F1N WPRT 102	F1N WPRT 101	F1N WPRT 100	F1N WPRT 99	F1N WPRT 98	F1N WPRT 97	F1N WPRT 96

Bit	Symbol	Bit name	Description	Read and write
b31	F1NWPRT127	write protection of sector 127 in FLASH1	0: write protection is valid of sector127 in FLASH1 1: write protection is invalid of sector127 in FLASH1 When WLOCK[7]=0, 0 and 1 can be written. When WLOCK[7]=1, only 0 can be written.	R/W
b30	F1NWPRT126	write protection of sector 126 in FLASH1	0: write protection is valid of sector126 in FLASH1 1: write protection is invalid of sector126 in FLASH1 When WLOCK[7]=0, 0 and 1 can be written. When WLOCK[7]=1, only 0 can be written.	R/W
b29~b2	R/W
b1	F1NWPRT97	write protection of sector 97 in FLASH1	0: write protection is valid of sector97 in FLASH1 1: write protection is invalid of sector97 in FLASH1 When WLOCK[7]=0, 0 and 1 can be written. When WLOCK[7]=1, only 0 can be written.	R/W
b0	F1NWPRT96	write protection of sector 96 in FLASH1	0: write protection is valid of sector96 in FLASH1 1: write protection is invalid of sector96 in FLASH1 When WLOCK[7]=0, 0 and 1 can be written. When WLOCK[7]=1, only 0 can be written.	R/W

7.10 Attention

1. Programming and erasing of FLASH will be forced to stop when reset happens, and FLASH data will not be guaranteed. The customer needs to erase the address after reset release.
2. The hardware circuit will automatically clean up the cache memory after program and erase operation has been ended.
3. Over program to the same address doesn't ensure the value of the programmed data.
4. In series program mode, FLASH-IP is under high voltage state, long time high voltage state will reduce FLASH characteristic. It is not allowed to enter low power mode (sleep mode, stop mode, power-off mode) when FLASH is in series program mode.
5. Make sure that the cache RAM is not powered down (PWC_PRAMLPC.PRAMPDC2=0) if cache is enabled.

8 Built-in SRAM (SRAM)

8.1 Introduction

This product has 4KB backup SRAM (SRAMB) and 512KB system SRAM (SRAMH/SRAM1/SRAM2/SRAM3/SRAM4).

Each SRAM can be accessed as bytes, half-words (16 bits) or full-words (32 bits). High-speed SRAM (SRAMH) read/write operations are performed at the fastest CPU speed (240MHz) with 0 wait (ie 1 cycle), or wait cycles can be inserted. The relationship between the wait period setting of read/write access and the CPU clock frequency is shown in Table 8-1. The wait period for read/write access of each SRAM is set by the SRAM wait control register (SRAM_WTCR).

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

Wait cycle (CPU access cycle)	Access to high-speed SRAM(SRAMB) allows CPU clock range	Access to power-down mode SRAM (SRAMB) allows CPU clock range	Access to other SRAM (SRAM1,2,3,4) allows CPU clock range
0wait (1 CPU cycle access)	0~240MHz	0~120MHz	0~200MHz
1wait (2 CPU cycle access)	0~240MHz	0~240MHz	0~240MHz
2wait (3 CPU cycle access)	0~240MHz	0~240MHz	0~240MHz
3wait (4 CPU cycle access)	0~240MHz	0~240MHz	0~240MHz
4wait (5 CPU cycle access)	0~240MHz	0~240MHz	0~240MHz
5wait (6 CPU cycle access)	0~240MHz	0~240MHz	0~240MHz
6wait (7 CPU cycle access)	0~240MHz	0~240MHz	0~240MHz
7wait (8 CPU cycle access)	0~240MHz	0~240MHz	0~240MHz

SRAMB can provide 4KB data retention space in Power down mode.

SRAM4 and SRAMB have ECC check (Error Checking and Correcting). The ECC check is single error correction double error detection,, which can correct one-bit error and check two-bit error. SRAMH/SRAM1/SRAM2/SRAM3 have parity check, and each byte of data has a parity bit. The detailed definition of SRAM is shown in Table 8-2 .

Table 8-2 SRAM address map

Name	Capacity	Address range	Check mode
SRAM1	128KB	0x20000000~0x2001FFFF	Parity check
SRAM2	128KB	0x20020000~0x2003FFFF	Parity check
SRAM3	96KB	0x20040000~0x20057FFF	Parity check
SRAM4	32KB	0x20058000~0x2005FFFF	ECC check
SRAMB	4KB	0x200F0000~0x200F0FFF	ECC check
SRAMH	128KB	0x1FFE0000~0x1FFFFFFF	Parity check

Note:

- In the case that the NMI interrupt and reset are allowed to be generated by the RAM parity check error, before read/write access operation, the RAM space used must be initialized in word units; when executing instructions from RAMH space, the RAM space used must be +3 word and the area is initialized in word units.
- In the case that the NMI interrupt and reset are allowed to be generated by the RAM ECC check error, before read/write access operation, the RAM space used must be initialized in word units.
- The VBAT domain needs to be initialized before the backup SRAM(SRAMB) is used. For the initialization method, please refer to [Battery backup power domain]

8.2 Register description

Table 8-1 List of Registers

Register name	Starting address	Reset value
SRAM Wait Control Register (SRAM_WTCR)	0x40050800	0x00000000
SRAM Wait Control Protection Register (SRAM_WTPR)	0x40050804	0x00000000
SRAM Check Control Register (SRAM_CKCR)	0x40050808	0x00000000
SRAM Check Control Protection Register (SRAM_CKPR)	0x4005080C	0x00000000
SRAM Check Status Register (SRAM_CKSR)	0x40050810	0x00000000

8.2.1 SRAM Wait Control Register (SRAM_WTCR)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Rev	SRAMBWWT[2:0]	Rev	SRAMBRWT[2:0]	Rev	SRAMHWWT[2:0]	Rev	SRAMHRWT[2:0]	Rev	SRAM4WWT[2:0]	Rev	SRAM4RWT[2:0]	Rev	SRAM123WWT[2:0]	Rev	SRAM123RWT[2:0]
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
Rev	SRAM4WWT[2:0]	Rev	SRAM4RWT[2:0]	Rev	SRAM123WWT[2:0]	Rev	SRAM123RWT[2:0]	Rev	SRAM4WWT[2:0]	Rev	SRAM4RWT[2:0]	Rev	SRAM123WWT[2:0]	Rev	SRAM123RWT[2:0]

Bit	Symbol	Bit name	Description	Read and write
b31	Reserved	-	Read as the reset value. Write the reset value.	R/W
b30~b28	SRAMBKWWT[2:0]	SRAMB write cycle selection	000b: RAM write takes 1 cycle. 001b: RAM write takes 2 cycle. 010b: RAM write takes 3 cycle. 011b: RAM write takes 4 cycle. 100b: RAM write takes 5 cycle. 101b: RAM write takes 6 cycle. 110b: RAM write takes 7 cycle. 111b: RAM write takes 8 cycle.	R/W
b27	Reserved	-	Read as the reset value. Write the reset value.	R/W
b26~b24	SRAMBKRWT[2:0]	SRAMB read cycle selection	000b: RAM read takes 1 cycle. 001b: RAM read takes 2 cycle. 010b: RAM read takes 3 cycle. 011b: RAM read takes 4 cycle. 100b: RAM read takes 5 cycle. 101b: RAM read takes 6 cycle. 110b: RAM read takes 7 cycle. 111b: RAM read takes 8 cycle.	R/W
b23	Reserved	-	Read as the reset value. Write the reset value	R/W
b22~b20	SRAMHWWT[2:0]	SRAMH write cycle selection	000b: RAM write takes 1 cycle. 001b: RAM write takes 2 cycle. 010b: RAM write takes 3 cycle. 011b: RAM write takes 4 cycle. 100b: RAM write takes 5 cycle. 101b: RAM write takes 6 cycle. 110b: RAM write takes 7 cycle. 111b: RAM write takes 8 cycle.	R/W
b19	Reserved	-	Read as the reset value. Write the reset value.	R/W
b18~b16	SRAMHRWT[1:0]	SRAMH read cycle selection	000b: RAM read takes 1 cycle. 001b: RAM read takes 2 cycle. 010b: RAM read takes 3 cycle. 011b: RAM read takes 4 cycle. 100b: RAM read takes 5 cycle. 101b: RAM read takes 6 cycle. 110b: RAM read takes 7 cycle. 111b: RAM read takes 8 cycle.	R/W
b15	Reserved	-	Read as the reset value. Write the reset value.	R/W
b14~b12	SRAM4WWT[2:0]	SRAM4 write cycle selection	000b: RAM write takes 1 cycle. 001b: RAM write takes 2 cycle. 010b: RAM write takes 3 cycle. 011b: RAM write takes 4 cycle. 100b: RAM write takes 5 cycle. 101b: RAM write takes 6 cycle. 110b: RAM write takes 7 cycle. 111b: RAM write takes 8 cycle.	R/W
b11	Reserved	-	Read as the reset value. Write the reset value.	R/W
b10~b8	SRAM4RWT[2:0]	SRAM4 read cycle selection	000b: RAM read takes 1 cycle. 001b: RAM read takes 2 cycle. 010b: RAM read takes 3 cycle. 011b: RAM read takes 4 cycle. 100b: RAM read takes 5 cycle. 101b: RAM read takes 6 cycle. 110b: RAM read takes 7 cycle. 111b: RAM read takes 8 cycle.	R/W

B7	Reserved	-	Read as the reset value. Write the reset value.	R/W
b6~b4	SRAM123WWT[2:0]	SRAM1, SRAM2 and SRAM3 write cycle selection	000b: RAM write takes 1 cycle. 001b: RAM write takes 2 cycle. 010b: RAM write takes 3 cycle. 011b: RAM write takes 4 cycle. 100b: RAM write takes 5 cycle. 101b: RAM write takes 6 cycle. 110b: RAM write takes 7 cycle. 111b: RAM write takes 8 cycle.	R/W
b3	Reserved	-	Read as the reset value. Write the reset value.	R/W
b2~b0	SRAM123RWT[2:0]	SRAM1, SRAM2 and SRAM3 read cycle selection	000b: RAM read takes 1 cycle. 001b: RAM read takes 2 cycle. 010b: RAM read takes 3 cycle. 011b: RAM read takes 4 cycle. 100b: RAM read takes 5 cycle. 101b: RAM read takes 6 cycle. 110b: RAM read takes 7 cycle. 111b: RAM read takes 8 cycle.	R/W

8.2.2 SRAM Waiting Protection Register (SRAM_WTPR)

Reset value: 0x00000000

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]						WTP RC	

Bit	Symbol	Bit name	Description	Read and write
b31~b8	Reserved	-	Read as the reset value. Write the reset value.	R/W
b7~b1	WTPRKW[6:0]	protection key	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 waitcontrol register write enable	R/W

WTPRC: Controls the write operation of the SRAMWTCR register. When WTPRC is set to 1, the SRAMWTCR register can be written, and if it is set to 0, the SRAMWTCR register cannot be written. When writing this bit, 3bh must be written to WTPRKW[6:0] at the same time.

8.2.3 SRAM Check Control Register (SRAM_CKCR)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16		
Rev				BECCMOD[1:0]		ECCMOD[1:0]		Rev				BECCOAD		ECCOAD			
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0		
Rev														PYOAD			
Bit	Symbol		Bit name		Description				Read and write								
b31~b28	Reserved		-		Read as the reset value. Write the reset value.				R/W								
b27~b26	BECCMOD[1:0]		ECC check enable bit of SRAMB		00: Disable ECC check function 01: If 1 bit error, ECC error correction, No 1-bit error flag, no interrupt/reset; If 2 bits are wrong, ECC error detection, Generate 2-bit error flag, generate interrupt/reset. 10: If 1 bit error, ECC error correction, Generate 1-bit error flag, no interrupt/reset; If 2 bits are wrong, ECC error detection, Generate 2-bit error flag, generate interrupt/reset. 11: If 1 bit error, ECC error correction, Generate 1-bit error flag, generate interrupt/reset; If 2 bits are wrong, ECC error detection, Generate 2-bit error flag, generate interrupt/reset.				R/W								
b25~b24	ECCMOD[1:0]		ECC check enable bit of SRAM4		00: Disable ECC check function 01: ECC perform 1 bit error correction. No 1-bit error flag, no interrupt/reset; ECC perform 2 bit error detection. Generate 2-bit error flag, generate interrupt/reset. 10: ECC perform 1 bit error correction Generate 1-bit error flag, no interrupt/reset; ECC perform 2 bit error detection. Generate 2-bit error flag, generate interrupt/reset. 11: ECC perform 1 bit error correction Generate 1-bit error flag, generate interrupt/reset; ECC perform 2 bit error detection. Generate 2-bit error flag, generate interrupt/reset.				R/W								
b23~b18	Reserved		-		Read as the reset value. Write the reset value.				R/W								
b17	BECCOAD		Operation after ECC check error of SRAMB		0: Non-maskable interrupt 1: Reset				R/W								
b16	ECCOAD		Operation after ECC check error of SRAM4		0: Non-maskable interrupt 1: Reset				R/W								
b15~b1	Reserved		-		Read as the reset value. Write the reset value.				R/W								
b0	PYOAD		Operation after Parity check error		0: Non-maskable interrupt 1: Reset				R/W								

8.2.4 SRAM Check Protection Register (SRAM_CKPR)

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]							
Bit	Symbol	Bit name	Description											Read and write	
b31~b8	Reserved	-	Read as the reset value. Write the reset value.											R/W	
b7~b1	CKPRKW[6:0]	Write protection key	When writing to the current register, write "3b" to these bits to enable the current register											R/W	
b0	CKPRC	SRAM check control register write enable	0: SRAM verification control register write disabled 1: SRAM verification control register write enable											R/W	

CKPRC: Controls the write operation of the SRAMCKCR register. When CKPRC is set to 1, the SRAMCKCR register can be written, and if it is set to 0, the SRAMCKCR register cannot be written. When writing this bit, 3bh must be written to CKPRKW[6:0] at the same time.

8.2.5 SRAM Check Status Register (SRAM_CKSR)

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						CAH CERA M_ PYER R	SRA MB_ 2ERR	SRA MB_ 1ERR	SRA M4_ 2ERR	SRA M4_ 1ERR	SRA MH_ PYER R	SRA M3_ PYER R	SRA M2_ PYER R	SRA M1_ PYER R	
Bit	Symbol	Bit name	Description								Read and write				
b31~b9	Reserved	-	Read as the reset value. Write the reset value.								R/W				
b8	CACHERAM_PYERR	CACHE RAM parity check error flag	0: No parity error occurred 1: A parity error has occurred								R/W (Note 1)				
B7	SRAMB_2ERR	SRAMB ECC 2-bit error flag	0: No 2-bit ECC error occurred 1: 2-bit ECC error occurred								R/W (Note 1)				
b6	SRAMB_1ERR	SRAMB ECC 1-bit error flag	0: No 1-bit ECC error occurred 1: 1-bit ECC error occurred								R/W (Note 1)				
b5	SRAM4_2ERR	SRAM4 ECC 2-bit error flag	0: No 2-bit ECC error occurred 1: 2-bit ECC error occurred								R/W (Note 1)				
b4	SRAM4_1ERR	SRAM4 ECC 1-bit error flag	0: No 1-bit ECC error occurred 1: 1-bit ECC error occurred								R/W (Note 1)				
b3	SRAMH_PYERR	SRAMH parity check error flag	0: No parity error occurred 1: A parity error has occurred								R/W (Note 1)				
b2	SRAM3_PYERR	SRAM3 parity check error flag	0: No parity error occurred 1: A parity error has occurred								R/W (Note 1)				
b1	SRAM2_PYERR	SRAM2 parity check error flag	0: No parity error occurred 1: A parity error has occurred								R/W (Note 1)				
b0	SRAM1_PYERR	SRAM1 parity check error flag	0: No parity error occurred 1: A parity error has occurred								R/W (Note 1)				

Note 1: Write 1 to clear 0.

9 General IO (GPIO)

Some of the abbreviations used in this chapter are:

- Px (x=A~I) indicates a group of ports, eg: PA indicates the 16 I/O ports of the group PA0~PA15
- Pxy (x= A~I, y=0~15, the same below) indicates a single I/O port, for example, the PB10 port indicates the tenth I/O of the PB group
- General Purpose Input Output (GPIO)
- NOD/POD (Noss/Pmos Open Drain) NMOS/PMOS Open-drain Output Mode

9.1 Introduction

Main Features:

- Each port group has 16 I/O Pins, which may be less than 16 depending on actual configuration
- Support pull-up
- Support push-pull, open-drain output mode
- Support high, medium and low drive modes
- Support free switching between CMOS/Schmitt input modes
- Support for external interrupt input
- Support I/O pin peripheral function multiplexing, each I/O pin can have up to 64 selectable multiplexing functions
- Each I/O pins can be programmed independently
- Each I/O pin can select two functions to be active at the same time (two output functions are not supported to be active at the same time)

9.2 Port function summary

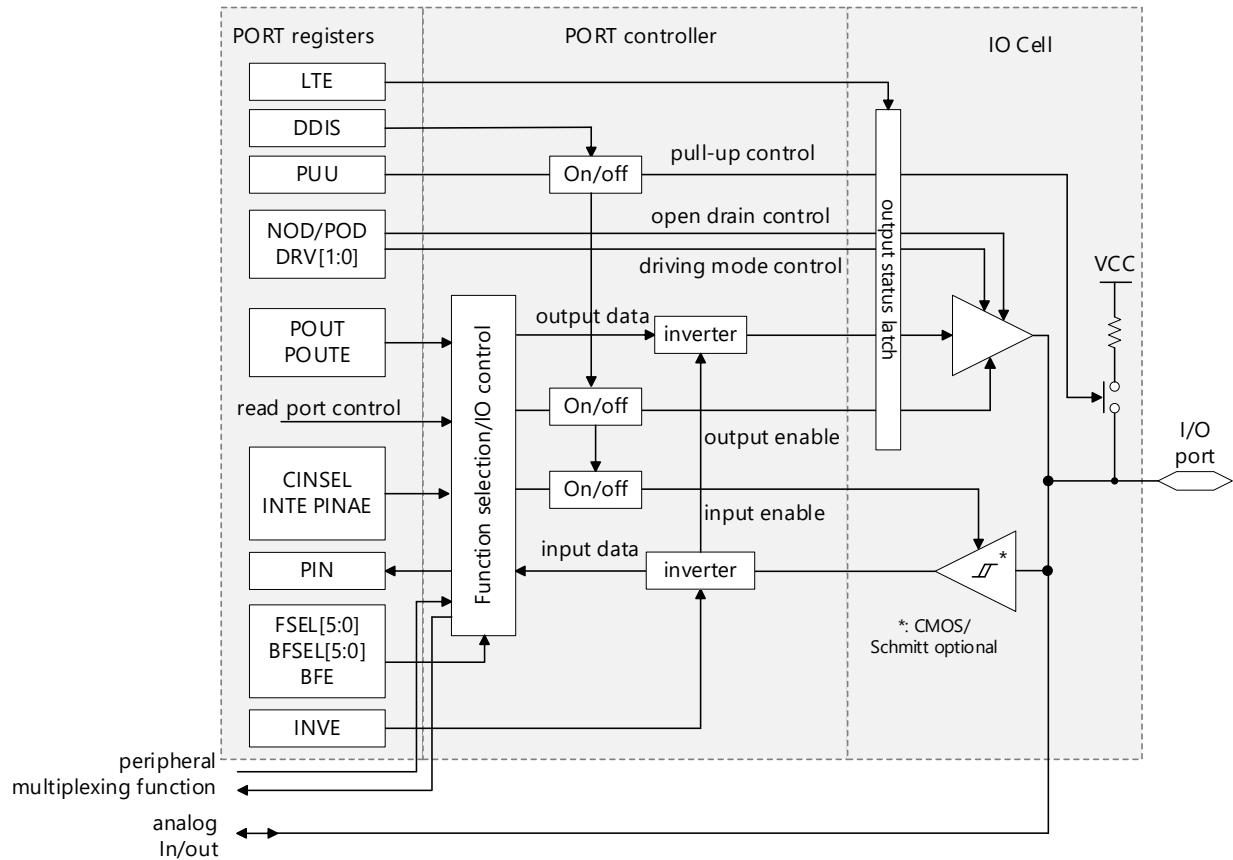


Figure 9-1 Port basic structure diagram

For a detailed description of the number of GPIO ports, 5V durability, and drive capability configuration, please refer to the **Pin Configuration and Function** chapter of the data sheet.

9.3 Functional Description

9.3.1 General Purpose Input/Output GPIO function

General Input function GPI:

Each I/O has a general-purpose input GPI function, and when the digital function disable bit PCRxy.DDIS is 0, the GPI function is always valid, regardless of the function selection register setting value of PFSRxy.FSEL[5:0]. The status of the current port can be obtained by accessing port input data register PIDRx. User can also query the corresponding single I/O port status through the bit of the port control register PCRxy.PIN. The PIDRx.PIN[y] register bit is equivalent to the PCRxy.PIN bit.

By default, I/O input MOS is turned off to reduce power consumption. Only when PIDRx and PCRxy are read, I/O input MOS is turned on. As required, the input MOS of the I/O can also be kept on by setting the register PINAER.PINAE[x] to 1.

When the system is running on a high-speed clock and PINAER is set to 0, the input status value may not be read correctly in a single cycle due to the delay in the I/O input. At this point, the register PCCR.RDWT[2:0] needs to be set, and several wait cycles are inserted. For details, refer to the description of the register PCCR.

General Output Function GPO:

Most of the I/O ports have general-purpose output GPO function except the input dedicated port do not have. 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 generic output permission register POERRx to enable or disable the output of I/O and the generic output value is controlled by register PODRx. The output value of the I/O can also be controlled using the following 3 registers: Output data clear register PORRx, output data set register POSRx, output data toggle register POTRx. Writing 1 to the corresponding bits in theose register can make the corresponding I/O output 0, 1 or invert. I/O output status does not change when write 0 to PORRx, POSRx or POTRx.

The registers above are operated by 16 PORT groups. For easy control of a single I/O, I/O output can also be operated individually by setting PCRxy.POUTE to enable or disable a single I/O, moreover, the PCRxy.POUTE register bit is equivalent to POERRx.POUTE [y]. The output value of I/O can be controlled by setting PCRxy.POUT, where the PCRxy.POUT register bit is equivalent to PODRx.POUT [y]. In a word, PCRxy is suitable for controlling a single PORT and POERRx/PODRx is suitable for controlling a 16-bit set of PORTs.

After system reset, except for JTAG multiplexed ports PA13, PA14, PA15, PB3, PB4, and secondary oscillator multiplexed ports PC14 and PC15, the initial functions of other ports are GPO (FSEL[5:0]=0x0), and In high impedance state (output disabled POUTE=0).

Note:

- Port PI13 is multiplexed with MD. When the reset port NRST is released, PI13/MD must be fixed to a low level so that the system can enter the user mode. After the mode is established, PI13 can be used as general purpose IO.

9.3.2 Peripheral functions

Each port can be configured with up to 64 functions through PFSRxy.FSEL[5:0] including the general-purpose output GPO function when PFSRxy.FSEL[5:0] is 0x0. For the specific configuration functions of each port, please refer to the **Pin Configuration and Function** chapter of the data sheet.

JTAG/SWD debug function, selected using register PSPCR. PFSRxy.FSEL[5:0] register bits of the corresponding port will be invalid when PSPCR.SPFE[z] is equal to 1 where z takes the value 0 to 4, that is, the priority of SPFE is higher than that of FSEL. The initial value of the PSPCR register is 0x1f, the JTAG/SWD function valid. If you want to set these ports to function other than JTAG/SWD, you need to write 0 to the corresponding SPFE[z] bit first.

9.3.3 Dual peripheral function

In some applications, one port needs to be set to two functions at the same time. In this case, one function can be selected by PFSRxy.FSEL[5:0] first, and then the second function can be selected by setting the common control register PCCR.BFSEL[5:0], and PFSRxy.BFE is set to 1. E.g: Set PFSRxy.FSEL[5:0]=0x2, PCCR.BFSEL[5:0]=0x5, PFSRxy.BFE=0x1, then function 2 and function 5 on Pxy will be valid at the same time. It is forbidden to have 2 output functions active at the same time on the same port.

9.3.4 Event Port input and output function

Supports 4 groups of Event Ports, each with 16 ports. Event Port1 includes EVNTP100~EVNTP115, Event Port2 includes EVNTP200~EVNTP215, and so on. EVNTPmn ($m=1\sim4$, $n=0\sim15$) the port can be used as a trigger source to generate events according to the port input to trigger other peripheral devices (such as TIMER, ADC, DMA, etc.) to execute specific actions. It can also act as a triggered object, accept events, and automatically input or output.

As a trigger source, set PEVNTRISRm, PEVNTFALRm, PEVNTNFCR to select rising edge or falling edge detection, and digital filtering function, and set the 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.

As the triggered object, set PEVNTRGSRm to select the trigger event source, and set PEVNTDIRRm to select the output or input function. When the output function is selected, EVNTPmn outputs the specified level or inversion according to the set values of PEVNTODRm, PEVNTORRm, and

PEVNTOSRm when the selected event occurs. When the input function is used, the EVNTPmn input state is saved into the register PEVNTIDRx when the selected event occurs.

When using the Event Port function, you need to Function Clock Control 0 Register (PWC_FCG0) The auto-run system AOS function enable bit is set to valid first.

9.3.5 External interrupt EIRQ input function

Except for the PI13/MD pin, each I/O port has an external interrupt input function. When the PCRxy.INTE bit is set to 1, this I/O will be enabled as an external interrupt source EIRQy (eg: PA0 corresponds to EIRQ0, and PA2 corresponds to EIRQ2). Each EIRQy can be configured with more than one I/O. When using, each EIRQy should not allow multiple I/O inputs at the same time. The EIRQy input function and the peripheral functions (including GPIO) selected by PFSRxy.FSEL can be active at the same time.

When I/O port is used as external interrupt EIRQ, I/O port needs to combine with interrupt controller INTC, set filter, interrupt trigger edge, interrupt number and so on. Please refer to Interrupt controller (INTC) chapter for details.

9.3.6 Analogfunction

Some I/O ports have analog I/O functions (including main and sub oscillators). When used as an analog function, please write 1 to the register PCRxy.DDIS to disable the digital function of the current port.

9.3.7 General control

1. Pull-up/Pull-down resistors

Each I/O port has an internal pull-up resistor. Set the register PCRxy.PUU bit to 1 to enable this feature, which is in a weak 1 state when I/O ports do not have inputs. Pull-up is automatically invalid when the I/O port is in the output state.

When the I/O port selects the I2Cx_SCL/I2Cx_SDA function, the setting of the register PCRxy.PUU will be ignored, and the internal pull-up function will be forcibly disabled.

The pins PA11/USBFS_DM, PA12/USBFS_DP, PB14/USBHS_DM and PB15/USBHS_DP are multiplexed with the USB port. All four pins have built-in pull-down resistors of about 400KΩ, which are always valid.

2. Drive capability control

Each I/O port has high, medium and low drive capability adjustable. The corresponding drive capability can be selected by setting the register of PCRxy.DRV [1:0] according to the requirements.

This feature is only valid if the port is in the output state.

3. Open-drain output mode

Set the PCRxy.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, and the port will be in high resistance state when output 1.

When the I/O port selects the I₂C_x_SCL/I₂C_x_SDA function, the setting of the register NOD will be ignored, and the open-drain output mode will be forced to be valid.

4. CMOS/Schmitt input mode

Each I/O port supports CMOS and Schmitt input modes. Register PCRxy.CINSEL is set to 0 for Schmitt input and set 1 for CMOS input. Defaults to Schmitt input.

When I/O is used as input function, please set the input type according to actual needs.

The general control functions mentioned above, unless otherwise specified, are not related to the specific function that the port has been selected through the setting of FSEL[5:0].

9.4 Register description

BASE_ADDR: 0x40053800

Table 9-1 PORT register list 1

Register name	Symbol	Offset address	Bit width	Reset value
General input data register	PIDRx	0x00+0x10*n *1	16/32	0xFFFF
General output data register	PODRx	0x04+0x10*n	16/32	0x0000
General output enable register	POERx	0x06+0x10*n	16/32	0x0000
General output set register	POSRx	0x08+0x10*n	16/32	0x0000
General output reset register	PORRx	0x0A+0x10*n	16/32	0x0000
General output toggle register	POTRx	0x0C+0x10*n	16/32	0x0000
Special function control register	PSPCR	0x3F4	16/32	0x001F
Common control register	PCCR	0x3F8	16/32	0x1000
Input Always enable 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: In the address calculation formula, x=A~I corresponding to n=0~8

*2: The reset value of 32K sub-oscillator multiplexed port PCRC14 and PCRC15 is 0x8100.

BASE_ADDR: 0x40010800

Table 9-2 PORT register list 2

Register name	Symbol	Offset address	Bit width	Reset value
Event Port1,2 trigger source selection register	PEVNTTRGSR12	0x68	32	0x000001FF
Event Port3,4 trigger source selection register	PEVNTTRGSR34	0x6C	32	0x000001FF
Event port direction selection register	PEVNTDIRm	0x100+0x1C*(m-1)	32	0x00000000
Event port input data register	PEVNTIDRm	0x104+0x1C*(m-1)	32	0x00000000
Event port output data register	PEVNTODRm	0x108+0x1C*(m-1)	32	0x00000000
Event port output data reset register	PEVNTORRm	0x10C+0x1C*(m-1)	32	0x00000000
Event port output data set Register	PEVNTOSRm	0x110+0x1C*(m-1)	32	0x00000000
Event port rising edge input enable register	PEVNTRISRm	0x114+0x1C*(m-1)	32	0x00000000
Event port falling edge input enable register	PEVNTFALRm	0x118+0x1C*(m-1)	32	0x00000000
Event port input noise filter control register	PEVNTNFCR	0x170	32	0x00000000

Note: m=1~4

9.4.1 General input register (PIDRx)

Reset value: 0XXXXX

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
PIN[15:0]															

Bit	Marking	Bit name	Function	Read and write
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 read-only and the write is invalid. When the digital function is not disabled (i.e. DDIS=0) , the input state of the port can be obtained by reading this register, regardless of the setting value of PFSRxy.FSEL[5:0] of the function selection register. There is no variable read value for the corresponding bit of the port. When the digital function of the port is disabled (i.e. DDIS=1) ,because the I/O input MOS is in the off state, so the corresponding PIN bit read value is a fixed value of 0x1.

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]															

Bit	Marking	Bit name	Function	Read and write
b15~b0	POUT[15:0]	Output data	0: Output low level 1: Output high level	R/W

Modifying this register will change the output status of the corresponding port when the I/O port is set to GPO.

9.4.3 General output enable Register (POERx)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
POUTE[15:0]															

Bit	Marking	Bit name	Function	Read and write
b15~b0	POUTE[15:0]	Export enable	0: Output disable 1: Output enable	R/W

When the I/O port is set to GPO and this register is set to 1, the PODRx setting is output to the corresponding I/O port. When this register is set to 0, the output is closed and the port is in high resistance state. Do not write 1 to the bit that do not correspond to any port.

9.4.4 General output set register (POSRx)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
POS[15:0]															

Bit	Marking	Bit name	Function	Read and write
b15~b0	POS[15:0]	Output “1”	0: No change corresponding to PODRx.POUT 1: Set to 1 corresponding to PODRx.POUT	R/W

The read value of this register is always 0x0000. During 32bit access, when POR[y] and POS[y] of the same I/O write 1 at the same time, POR[y] has a higher priority, that is, the corresponding POUT[y] is cleared .

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]															

Bit	Marking	Bit name	Function	Read and write
b15~b0	POR[15:0]	Output “0”	0: No change corresponding to PODRx.POUT 1: Corresponding to PODRx.POUT is cleared	R/W

The read value of this register is always 0x0000.

9.4.6 General output toggle register (POTRx)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
POT[15:0]															

Bit	Marking	Bit name	Function	Read and write
b15~b0	POT[15:0]	Output inversion	0: No change to PODRx.POUT 1: Reverse PODRx.POUT	R/W

The read value of this register is always 0x0000.

9.4.7 Special function 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]			

Bit	Marking	Bit name	Function	Read and write
b15~b5	Reserved	-	Read as 0, write as 0	R/W
b4	SPFE[4]	Special function selection	0: NJTRST function is invalid 1: NJTRST function is valid	R/W
b3	SPFE[3]	Special function selection	0: JTDI can be disabled 1: JTDI function is valid	R/W
b2	SPFE[2]	Special function selection	0: JTDO_TRACESWO function is invalid 1: JTDO_TRACESWO function is valid	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

Note:

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

9.4.8 Common control register (PCCR)

Reset value: 0x1000

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0									
-	RDWT[2:0]		-	-	-	-	-	-	-	BFSEL[5:0]														
Bit	Marking	Bit name		Function										Read and write										
b15	Reserved			Read as 0, write as 0										R/W										
b14-b12	RDWT[2:0]	Read Port Waiting			Set the number of wait cycles inserted when reading registers PIDRx, PCRxy																			
					Set value	Waiting period		Recommended frequency of work																
					000	No wait		~50MHz																
					001 (default)	1 cycle		50~100MHz																
					010	2 cycles		100~150MHz																
					011	3 cycles		150~200MHz																
					100	4 cycles		200~250MHz																
					101	5 cycles		Above 250MHz																
b11~b6		Reserved				Read as 0, write as 0																		
b5~b0		BFSEL[5:0]		Secondary function selection		For the function configuration of each port, please refer to the Pin Configuration and Function chapter of the data sheet.																		

9.4.9 Input always enable register (PINAER)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0						
-	-	-	-	-	-	-	-	PINAЕ[8:0]													
PINAЕ[5:0]														-							

Bit	Marking	Bit name	Function	Read and write
b15~b9	Reserved	-	Read as 0, write as 0	R/W
b8~b0	PINAЕ[8:0]	Input normally open	0: Input MOS normally open invalid 1: Input MOS normally open valid PINAЕ[0] controls PA0~PA15, PINAЕ[1] controls PB0~PB15, PINAЕ[2] controls PC0~PC15, PINAЕ[3] controls PD0~PD15, PINAЕ[4] controls PE0~PE15, PINAЕ[5] controls PF0~PF15, PINAЕ[6] controls PG0~PG15, PINAЕ[7] controls PH0~PH15, PINAЕ[8] controls PI0~PI13	R/W

9.4.10 Write protection 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
Bit	Marking	Bit name	Function				Read and write								
b15~b8	WP[7:0]	Write protection code	Read as 0x00 When b15 ~ b8 write value is 0xA5, b0 write WE valid When writing a value other than 0xA5, WE automatically clears				W								
b7~b1	Reserved	-	Read asas 0, write asas 0				R/W								
b0	WE	Write enable	0: PSPCR, PCCR, PINAER, PCRxy, PFSRxy register 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	-	CINSEL	INVE	PIN	-	PUU	DRV[1:0]	-	NOD	POUTE	POUT	

Bit	Marking	Bit name	Function	Read and write
b15	DDIS	Digital function disabled	0: Digital function enable 1: Digital function disabled	R/W
b14	LTE	Output state latch	0: Output latch invalid 1: Output latch valid	R/W
b13	Reserved	-	Read as 0, write as 0	R/W
b12	INTE	External interrupt enable	0: External interrupt input disable 1: External interrupt input enable	R/W
b11	Reserved	-	Read as 0, write as 0	R/W
b10	CINSEL	Input mode selection	0: Schmitt input 1: CMOS input	R/W
b9	INVE	Inverse enable	0: Input/output data not inverted 1: Input/output data inverted	R/W
b8	PIN	Input status	0: I/O port input status is low 1: I/O port input status is high Consistent with the PIN [y] feature in registerPDIRx	R
B7	Reserved	-	Read as 0, write as 0	R/W
b6	PUU	Pull-up enable	0: Pull-up internal resistor invalid 1: Pullup internal resistance is valid	R/W
b5~b4	DRV[1:0]	Drive mode selection	b00: Low driving mode b01: Medium driving mode b10, b11: High driving mode	R/W
b3	Reserved	-	Read as 0, write as 0	R/W
b2	NOD	NMOS open-drain output	0: Normal CMOS Output Mode 1: NMOS open-drain output	R/W
b1	POUTE	Out enable	0: Output disable 1: Output enable Consistent with the POUTE [y] function in registerPOERx	R/W
b0	POUT	Output data	0: Output low level 1: Output high level Consistent with the POUT [y] function in registerPODRx	R/W

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

When LTE is set to 1, the output latch is valid, which means that the current output state of the port is maintained until LTE is written to 0. This function is mainly used when the port function is switched. To avoid system malfunction caused by unexpected burrs in port output during function switching, set LTE to 1 before the function switching, the output status of the port is latched, and then the modify the function select register. Finally, set LTE to 0 and the port status is updated to a new function.

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

CINSEL is used to set the I/O input mode, acting on all digital input functions such as GPI, EIRQ and peripheral input.

9.4.12 Function select 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]			

Bit	Marking	Bit name	Function	Read and write
b15~b9	Reserved	-	Read as 0, write as 0	R/W
b8	BFE	Secondary function enabled	Controls whether the secondary function selected by PCCR.BFSEL[5:0] is enable 0: Sub-function disabled 1: Sub-function is enabled	R/W
b7~b6	Reserved	-	Read as 0, write as 0	R/W
b5~b0	FSEL[5:0]	Functional selection	For the function configuration of each port, please refer to the Pin Configuration and Function chapter in the data sheet	R/W

Each I/O port can select one of several functions configured on that port through FSEL[5:0]. Refer to the **Pin Configuration and Function** chapter in the data sheet, i.e.: set FSEL[5:0] to b000000 to select Func0, set to b000001 to select Func1, and so on, set b001111 to select Func15. The general output function GPO corresponding to Func0.

Note:

- After the PA13, PA14, PA15, PB3, PB4 ports are reset, the initial state is that the JTAG/SWD function is valid. When selecting the function of above port by configuring the FSEL[5:0], firstly, need to write 0 to the corresponding bit of the register PSPCR to invalidate the JTAG/SWD function. After the PC14 and PC15 ports are reset, the initial state is the digital function disabled state. When selecting the digital function, it is necessary to write 0 to the DDIS bit of the corresponding register PCRxy for the effective digital function.

9.4.13 Event port trigger source selection register (PEVNTRGSR12, PEVNTRGSR34)

Reset value: 0x0000001ff

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]		-								-					
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-								TRGSEL[8:0]							

Bit	Marking	Bit name	Function	Read and write
b31	COMEN[1]	common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG2 to trigger this AOS target For common trigger events, please refer to the AOS chapter	R/W
b30	COMEN[0]	common trigger enable	0: Disable the common trigger event of AOS_COMTRG1 from triggering this AOS target 1: Enable the common trigger event of AOS_COMTRG1 to trigger this AOS target For common trigger events, please refer to the AOS chapter	R/W
b29~b9	Reserved	-	Read as 0, write as 0	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 input state of the I/O port. PEVNTRGSR12 sets the trigger source of Event Port1 and 2, and PEVNTRGSR34 sets the trigger source of Event Port3 and 4. For specific numbers, please refer to [table 10-3 interrupt enet request number and selection].	R/W

9.4.14 Event port direction selection register (PEVNTDIRm)

Reset value: 0x00000000

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
PDIR[15:0]															
<hr/>															
Bit	Marking	Bit name	Function	Read and write											
b31~b16	Reserved	-	Read as 0, write as 0	R/W											
b15~b0	PDIR15:0]	direction selection	0: Event Port is input function 1: Event Port is output function	R/W											

9.4.15 Event port input data register (PEVNTIDRm)

Reset value: 0x00000000

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
PIN[15:0]															
<hr/>															
Bit	Marking	Bit name	Function	Read and write											
b31~b16	Reserved	-	Read as 0, write as 0	R/w											
b15~b0	PIN[15:0]	port input status	0: Event Port input state is low level when the event is triggered 1: Event Port input state is high level when the event is triggered	R											

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

9.4.16 Event port output data register (PEVNTODRm)

Reset value: 0x00000000

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
POUT[15:0]															

Bit	Marking	Bit name	Function	Read and write
b31~b16	Reserved	-	Read as 0, write as 0	R/W
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 the Event Port is set to the output, write the initial output value to this register, before the selected event is triggered. When the selected event is triggered, according to the set value of PEVNTORRm and PEVNTOSRm, the corresponding bit of PEVNTODRm.POUT is cleared to 0, set to 1, or inverted, and output to the EVNTPmn port at the same time.

9.4.17 Event port output data reset register (PEVNTORRm)

Reset value: 0x00000000

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
POR[15:0]															

Bit	Marking	Bit name	Function	Read and write
b31~b16	Reserved	-	Read as 0, write as 0	R/W
b15~b0	POR[15:0]	output value reset	0: The corresponding PEVNTODRm.POUT does not change when the event is triggered 1: corresponding to PEVNTODRm.POUT reset when the event is triggered	R/W

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

9.4.18 Event port output data set register (PEVNTOSRm)

Reset value: 0x00000000

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
POS[15:0]															

Bit	Marking	Bit name	Function	Read and write
b31~b16	Reserved	-	Read as 0, write as 0	R/W
b15~b0	POS[15:0]	output value set	0: The corresponding PEVNTODRm.POUT does not change when the event is triggered 1: The corresponding PEVNTODRm.POUT is set when the event is triggered	R/W

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

9.4.19 Event port rising edge input enable register (PEVNTRISRm)

Reset value: 0x00000000

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
RIS[15:0]															

Bit	Marking	Bit name	Function	Read and write
b31~b16	Reserved	-	Read as 0, write as 0	R/W
b15~b0	RIS[15:0]	Rising edge detection permission	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. When the RIS bit is set to 1, corresponding to the rising edge of the EVNTPmn input, the event will be output to trigger other peripheral modules. The edge events of EVNTPm0~15 are combined into one event EVENT_PORTm output, and any port will output the event EVENT_PORTm after detecting the edge.

9.4.20 Event port falling edge input enable register (PEVNTFALRm)

Reset value: 0x00000000

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
FAL[15:0]															
<hr/>															
Bit	Marking	Bit name	Function												Read and write
b31~b16	Reserved	-	Read as 0, write as 0												R/W
b15~b0	FAL[15:0]	Falling edge detection permission	0: EVNTPmn falling edge event detection 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. When the FAL bit is set to 1, corresponding to the falling edge of the EVNTP input, an event is output to trigger other peripheral modules. The edge events of EVNTPm0~15 are combined into one event EVENT_PORTm output, and any port will output the event EVENT_PORTm after detecting the edge.

9.4.21 Event port input noise filter control register (PEVNTNFCR)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-		DIVS4[1:0]		NFEN4		-		-		DIVS3[1:0]		NFEN3			
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-		DIVS2[1:0]		NFEN2		-		-		DIVS1[1:0]		NFEN1			

Bit	Marking	Bit name	Function	Read and write
b31~b27	Reserved	-	Read as 0, write as 0	R/W
b26-b25	DIVS4[1:0]	Digital Filter Sampling Clock Selection	Event Port4 digital filter sampling clock selection 00: PCLK1 01: PCLK1/8 10: PCLK1/32 11: PCLK1/64	R/W
b24	NFEN4	Digital Filter Enabled	0: Event Port4 digital filtering is invalid 1: Event Port4 digital filter is valid	R/W
b23~b19	Reserved	-	Read as 0, write as 0	R/W
b18-b17	DIVS3[1:0]	Digital Filter Sampling Clock Selection	Event Port3 digital filter sampling clock selection 00: PCLK1 01: PCLK1/8 10: PCLK1/32 11: PCLK1/64	R/W
b16	NFEN3	Digital Filter Enabled	0: Event Port3 digital filtering is invalid 1: Event Port3 digital filter is valid	R/W
b15~b11	Reserved	-	Read as 0, write as 0	R/W
b10-b9	DIVS2[1:0]	Digital Filter 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 Filter enabled	0: Event Port2 digital filtering is invalid 1: Event Port2 digital filter is valid	R/W
b7~b3	Reserved	-	Read as 0, write as 0	R/W
b2-b1	DIVS1[1:0]	Digital Filter 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 Filter Enabled	0: Event Port1 digital filtering is invalid 1: Event Port1 digital filter is valid	R/W

9.4.22 32bit access

Among the registers mentioned above, except for Event Port-related registers that only support 32-bit access, other registers support 16-bit and 32-bit access, but do not support 8-bit access. These registers are combined as follows when accessing 32bit:

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

Address	b31 ~ b16	b15 ~ b0
0x40053800+0x10*n *1	Reserved	PIDRx
0x40053804+0x10*n	POERx	PODRx
0x40053808+0x10*n	PORRx	POSRx
0x4005380C+0x10*n	Reserved	POTRx
0x40053BF4	Reserved	PSPCR
0x40053BF8	PINAER	PCCR
0x40053BFC	Reserved	PWPR
0x40053C00+0x40*n+0x04*y	PFSRxy	PCRxy

Note*1: In the address calculation formula, $x=A\sim I$ corresponds to $n=0\sim 8$

9.5 Precautions

Do not set a feature to multiple ports.

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

Please perform port feature switching when the output latch is active (LTE = 1) to avoid unexpected burrs on the port during the switching.

10 Interrupt controller (INTC)

10.1 Overview

The Interrupt Controller (INTC) controls which event signals are linked to the NVIC IRQ input and RXEV input of CPU. When the system enters low power mode (sleep mode and stop mode), the INTC also controls its wake-up events. Additionally, the INTC controls external interrupts and software interrupts generation.

INTC main specifications:

1) NVIC interrupt vector

For the actual number of interrupt vectors used, refer to Table 10-2. Each interrupt vector (Excluding the 16 interrupt lines of Cortex™-M4F) can select the corresponding interrupt event according to the interrupt selection register (INTC_SEL). For more on exceptions and NVIC programming, refer to “Chapter 5 Exceptions” and “Chapter 8 Nested Vectored Interrupt Controller” of “ARM Cortex™-M4F Technical Reference Manual”.

2) Programmable priority

16 programmable priority levels (using 4-bit interrupt priority register).

3) Non-maskable interrupts

A variety of system interrupt events can be selected as non-maskable interrupts and have independent enable, flag, and flag clear registers.

4) 16 external interrupts (EIRQ).

5) Up to 512 peripheral interrupt events, refer to Table 10-3.

6) 32 software interrupts.

7) System sleep mode and stop mode wake-up support.

Table 10-1 INTC I/O pins

Pin name	I/O	Description
EIRQ0 to EIRQ15	Input	External interrupt request pins

10.2 Block Diagram

10.2.1 Interrupt system block diagram

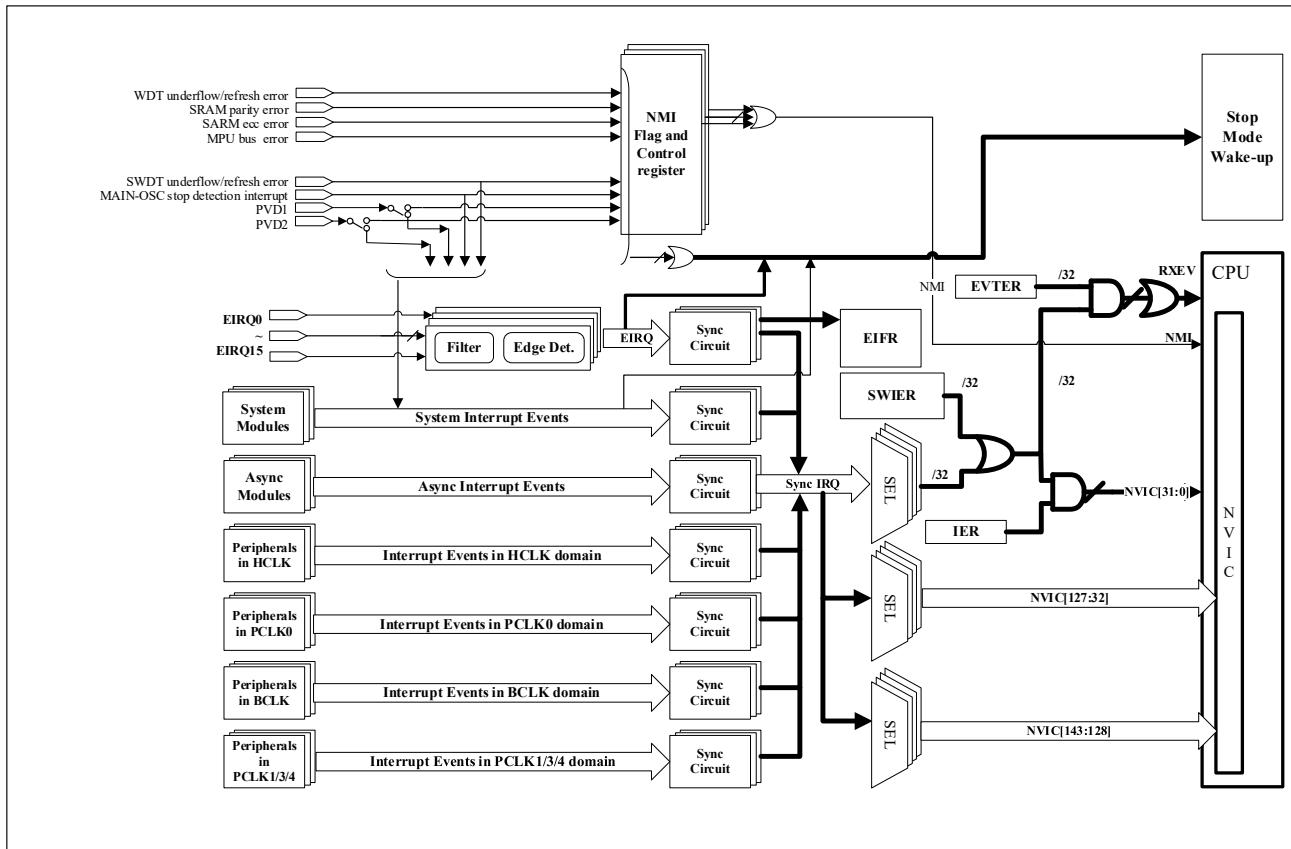


Figure 10-1 Interrupt System Block Diagram

10.3 Vector Table

10.3.1 Interrupt Vector Table

Table 10-2 Interrupt vector table

Interrupt Address	Vector number	IRQ number	Interrupt Source	Description
ARM Core Interrupt Processing Vector				
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 Vector				
0x0000_0040	16	0	INTC_SEL0/ INTC_SWIER	Event selected in the INTC_SEL0 register/ Software event
0x0000_0044	17	1	INTC_SEL1/ INTC_SWIER	Event selected in the INTC_SEL1 register/ Software event
0x0000_0048	18	2	INTC_SEL2/ INTC_SWIER	Event selected in the INTC_SEL2 register/ Software event
0x0000_004C	19	3	INTC_SEL3/ INTC_SWIER	Event selected in the INTC_SEL3 register/ Software event
0x0000_0050	20	4	INTC_SEL4/ INTC_SWIER	Event selected in the INTC_SEL4 register/ Software event
0x0000_0054	21	5	INTC_SEL5/ INTC_SWIER	Event selected in the INTC_SEL5 register/ Software event
0x0000_0058	22	6	INTC_SEL6/ INTC_SWIER	Event selected in the INTC_SEL6 register/ Software event
0x0000_005C	23	7	INTC_SEL7/ INTC_SWIER	Event selected in the INTC_SEL7 register/ Software event
0x0000_0060	24	8	INTC_SEL8/ INTC_SWIER	Event selected in the INTC_SEL8 register/ Software event
0x0000_0064	25	9	INTC_SEL9/ INTC_SWIER	Event selected in the INTC_SEL9 register/ Software event
0x0000_0068	26	10	INTC_SEL10/ INTC_SWIER	Event selected in the INTC_SEL10 register/ Software event
0x0000_006C	27	11	INTC_SEL11/ INTC_SWIER	Event selected in the INTC_SEL11 register/ Software event
0x0000_0070	28	12	INTC_SEL12/ INTC_SWIER	Event selected in the INTC_SEL12 register/ Software event

Interrupt Address	Vector number	IRQ number	Interrupt Source	Description
0x0000_0074	29	13	INTC_SEL13/ INTC_SWIER	Event selected in the INTC_SEL13 register/ Software event
0x0000_0078	30	14	INTC_SEL14/ INTC_SWIER	Event selected in the INTC_SEL14 register/ Software event
0x0000_007C	31	15	INTC_SEL15/ INTC_SWIER	Event selected in the INTC_SEL15 register/ Software event
0x0000_0080	32	16	INTC_SEL16/ INTC_SWIER	Event selected in the INTC_SEL16 register/ Software event
0x0000_0084	33	17	INTC_SEL17/ INTC_SWIER	Event selected in the INTC_SEL17 register/ Software event
0x0000_0088	34	18	INTC_SEL18/ INTC_SWIER	Event selected in the INTC_SEL18 register/ Software event
0x0000_008C	35	19	INTC_SEL19/ INTC_SWIER	Event selected in the INTC_SEL19 register/ Software event
0x0000_0090	36	20	INTC_SEL20/ INTC_SWIER	Event selected in the INTC_SEL20 register/ Software event
0x0000_0094	37	21	INTC_SEL21/ INTC_SWIER	Event selected in the INTC_SEL21 register/ Software event
0x0000_0098	38	22	INTC_SEL22/ INTC_SWIER	Event selected in the INTC_SEL22 register/ Software event
0x0000_009C	39	23	INTC_SEL23/ INTC_SWIER	Event selected in the INTC_SEL23 register/ Software event
0x0000_00A0	40	24	INTC_SEL24/ INTC_SWIER	Event selected in the INTC_SEL24 register/ Software event
0x0000_00A4	41	25	INTC_SEL25/ INTC_SWIER	Event selected in the INTC_SEL25 register/ Software event
0x0000_00A8	42	26	INTC_SEL26/ INTC_SWIER	Event selected in the INTC_SEL26 register/ Software event
0x0000_00AC	43	27	INTC_SEL27/ INTC_SWIER	Event selected in the INTC_SEL27 register/ Software event
0x0000_00B0	44	28	INTC_SEL28/ INTC_SWIER	Event selected in the INTC_SEL28 register/ Software event
0x0000_00B4	45	29	INTC_SEL29/ INTC_SWIER	Event selected in the INTC_SEL29 register/ Software event
0x0000_00B8	46	30	INTC_SEL30/ INTC_SWIER	Event selected in the INTC_SEL30 register/ Software event
0x0000_00BC	47	31	INTC_SEL31/ INTC_SWIER	Event selected in the INTC_SEL31 register/ Software event
0x0000_00C0	48	32	INTC_SEL32	Event selected in the INTC_SEL32 register
0x0000_00C4	49	33	INTC_SEL33	Event selected in the INTC_SEL33 register
0x0000_00C8	50	34	INTC_SEL34	Event selected in the INTC_SEL34 register
0x0000_00CC	51	35	INTC_SEL35	Event selected in the INTC_SEL35 register
0x0000_00D0	52	36	INTC_SEL36	Event selected in the INTC_SEL36 register
0x0000_00D4	53	37	INTC_SEL37	Event selected in the INTC_SEL37 register
0x0000_00D8	54	38	INTC_SEL38	Event selected in the INTC_SEL38 register
0x0000_00DC	55	39	INTC_SEL39	Event selected in the INTC_SEL39 register
0x0000_00E0	56	40	INTC_SEL40	Event selected in the INTC_SEL40 register
0x0000_00E4	57	41	INTC_SEL41	Event selected in the INTC_SEL41 register
0x0000_00E8	58	42	INTC_SEL42	Event selected in the INTC_SEL42 register
0x0000_00EC	59	43	INTC_SEL43	Event selected in the INTC_SEL43 register
0x0000_00F0	60	44	INTC_SEL44	Event selected in the INTC_SEL44 register
0x0000_00F4	61	45	INTC_SEL45	Event selected in the INTC_SEL45 register
0x0000_00F8	62	46	INTC_SEL46	Event selected in the INTC_SEL46 register

Interrupt Address	Vector number	IRQ number	Interrupt Source	Description
0x0000_00FC	63	47	INTC_SEL47	Event selected in the INTC_SEL47 register
0x0000_0100	64	48	INTC_SEL48	Event selected in the INTC_SEL48 register
0x0000_0104	65	49	INTC_SEL49	Event selected in the INTC_SEL49 register
0x0000_0108	66	50	INTC_SEL50	Event selected in the INTC_SEL50 register
0x0000_010C	67	51	INTC_SEL51	Event selected in the INTC_SEL51 register
0x0000_0110	68	52	INTC_SEL52	Event selected in the INTC_SEL52 register
0x0000_0114	69	53	INTC_SEL53	Event selected in the INTC_SEL53 register
0x0000_0118	70	54	INTC_SEL54	Event selected in the INTC_SEL54 register
0x0000_011C	71	55	INTC_SEL55	Event selected in the INTC_SEL55 register
0x0000_0120	72	56	INTC_SEL56	Event selected in the INTC_SEL56 register
0x0000_0124	73	57	INTC_SEL57	Event selected in the INTC_SEL57 register
0x0000_0128	74	58	INTC_SEL58	Event selected in the INTC_SEL58 register
0x0000_012C	75	59	INTC_SEL59	Event selected in the INTC_SEL59 register
0x0000_0130	76	60	INTC_SEL60	Event selected in the INTC_SEL60 register
0x0000_0134	77	61	INTC_SEL61	Event selected in the INTC_SEL61 register
0x0000_0138	78	62	INTC_SEL62	Event selected in the INTC_SEL62 register
0x0000_013C	79	63	INTC_SEL63	Event selected in the INTC_SEL63 register
0x0000_0140	80	64	INTC_SEL64	Event selected in the INTC_SEL64 register
0x0000_0144	81	65	INTC_SEL65	Event selected in the INTC_SEL65 register
0x0000_0148	82	66	INTC_SEL66	Event selected in the INTC_SEL66 register
0x0000_014C	83	67	INTC_SEL67	Event selected in the INTC_SEL67 register
0x0000_0150	84	68	INTC_SEL68	Event selected in the INTC_SEL68 register
0x0000_0154	85	69	INTC_SEL69	Event selected in the INTC_SEL69 register
0x0000_0158	86	70	INTC_SEL70	Event selected in the INTC_SEL70 register
0x0000_015C	87	71	INTC_SEL71	Event selected in the INTC_SEL71 register
0x0000_0160	88	72	INTC_SEL72	Event selected in the INTC_SEL72 register
0x0000_0164	89	73	INTC_SEL73	Event selected in the INTC_SEL73 register
0x0000_0168	90	74	INTC_SEL74	Event selected in the INTC_SEL74 register
0x0000_016C	91	75	INTC_SEL75	Event selected in the INTC_SEL75 register
0x0000_0170	92	76	INTC_SEL76	Event selected in the INTC_SEL76 register
0x0000_0174	93	77	INTC_SEL77	Event selected in the INTC_SEL77 register
0x0000_0178	94	78	INTC_SEL78	Event selected in the INTC_SEL78 register
0x0000_017C	95	79	INTC_SEL79	Event selected in the INTC_SEL79 register
0x0000_0180	96	80	INTC_SEL80	Event selected in the INTC_SEL80 register
0x0000_0184	97	81	INTC_SEL81	Event selected in the INTC_SEL81 register
0x0000_0188	98	82	INTC_SEL82	Event selected in the INTC_SEL82 register
0x0000_018C	99	83	INTC_SEL83	Event selected in the INTC_SEL83 register
0x0000_0190	100	84	INTC_SEL84	Event selected in the INTC_SEL84 register

Interrupt Address	Vector number	IRQ number	Interrupt Source	Description
0x0000_0194	101	85	INTC_SEL85	Event selected in the INTC_SEL85 register
0x0000_0198	102	86	INTC_SEL86	Event selected in the INTC_SEL86 register
0x0000_019C	103	87	INTC_SEL87	Event selected in the INTC_SEL87 register
0x0000_01A0	104	88	INTC_SEL88	Event selected in the INTC_SEL88 register
0x0000_01A4	105	89	INTC_SEL89	Event selected in the INTC_SEL89 register
0x0000_01A8	106	90	INTC_SEL90	Event selected in the INTC_SEL90 register
0x0000_01AC	107	91	INTC_SEL91	Event selected in the INTC_SEL91 register
0x0000_01B0	108	92	INTC_SEL92	Event selected in the INTC_SEL92 register
0x0000_01B4	109	93	INTC_SEL93	Event selected in the INTC_SEL93 register
0x0000_01B8	110	94	INTC_SEL94	Event selected in the INTC_SEL94 register
0x0000_01BC	111	95	INTC_SEL95	Event selected in the INTC_SEL95 register
0x0000_01C0	112	96	INTC_SEL96	Event selected in the INTC_SEL96 register
0x0000_01C4	113	97	INTC_SEL97	Event selected in the INTC_SEL97 register
0x0000_01C8	114	98	INTC_SEL98	Event selected in the INTC_SEL98 register
0x0000_01CC	115	99	INTC_SEL99	Event selected in the INTC_SEL99 register
0x0000_01D0	116	100	INTC_SEL100	Event selected in the INTC_SEL100 register
0x0000_01D4	117	101	INTC_SEL101	Event selected in the INTC_SEL101 register
0x0000_01D8	118	102	INTC_SEL102	Event selected in the INTC_SEL102 register
0x0000_01DC	119	103	INTC_SEL103	Event selected in the INTC_SEL103 register
0x0000_01E0	120	104	INTC_SEL104	Event selected in the INTC_SEL104 register
0x0000_01E4	121	105	INTC_SEL105	Event selected in the INTC_SEL105 register
0x0000_01E8	122	106	INTC_SEL106	Event selected in the INTC_SEL106 register
0x0000_01EC	123	107	INTC_SEL107	Event selected in the INTC_SEL107 register
0x0000_01F0	124	108	INTC_SEL108	Event selected in the INTC_SEL108 register
0x0000_01F4	125	109	INTC_SEL109	Event selected in the INTC_SEL109 register
0x0000_01F8	126	110	INTC_SEL110	Event selected in the INTC_SEL110 register
0x0000_01FC	127	111	INTC_SEL111	Event selected in the INTC_SEL111 register
0x0000_0200	128	112	INTC_SEL112	Event selected in the INTC_SEL112 register
0x0000_0204	129	113	INTC_SEL113	Event selected in the INTC_SEL113 register
0x0000_0208	130	114	INTC_SEL114	Event selected in the INTC_SEL114 register
0x0000_020C	131	115	INTC_SEL115	Event selected in the INTC_SEL115 register
0x0000_0210	132	116	INTC_SEL116	Event selected in the INTC_SEL116 register
0x0000_0214	133	117	INTC_SEL117	Event selected in the INTC_SEL117 register
0x0000_0218	134	118	INTC_SEL118	Event selected in the INTC_SEL118 register
0x0000_021C	135	119	INTC_SEL119	Event selected in the INTC_SEL119 register
0x0000_0220	136	120	INTC_SEL120	Event selected in the INTC_SEL120 register
0x0000_0224	137	121	INTC_SEL121	Event selected in the INTC_SEL121 register
0x0000_0228	138	122	INTC_SEL122	Event selected in the INTC_SEL122 register

Interrupt Address	Vector number	IRQ number	Interrupt Source	Description
0x0000_022C	139	123	INTC_SEL123	Event selected in the INTC_SEL123 register
0x0000_0230	140	124	INTC_SEL124	Event selected in the INTC_SEL124 register
0x0000_0234	141	125	INTC_SEL125	Event selected in the INTC_SEL125 register
0x0000_0238	142	126	INTC_SEL126	Event selected in the INTC_SEL126 register
0x0000_023C	143	127	INTC_SEL127	Event selected in the INTC_SEL127 register
0x0000_0240	144	128	INTC_VSSEL128	Events selected in the INTC_VSSEL128 register (all selected events share the vector).
0x0000_0244	145	129	INTC_VSSEL129	Events selected in the INTC_VSSEL129 register (all selected events share the vector).
0x0000_0248	146	130	INT_VSSEL130	Events selected in the INTC_VSSEL130 register (all selected events share the vector).
0x0000_024C	147	131	INT_VSSEL131	Events selected in the INTC_VSSEL131 register (all selected events share the vector).
0x0000_0250	148	132	INT_VSSEL132	Events selected in the INTC_VSSEL132 register (all selected events share the vector).
0x0000_0254	149	133	INT_VSSEL133	Events selected in the INTC_VSSEL133 register (all selected events share the vector).
0x0000_0258	150	134	INT_VSSEL134	Events selected in the INTC_VSSEL134 register (all selected events share the vector).
0x0000_025C	151	135	INT_VSSEL135	Events selected in the INTC_VSSEL135 register (all selected events share the vector).
0x0000_0260	152	136	INT_VSSEL136	Events selected in the INTC_VSSEL136 register (all selected events share the vector).
0x0000_0264	153	137	INT_VSSEL137	Events selected in the INTC_VSSEL137 register (all selected events share the vector).
0x0000_0268	154	138	INT_VSSEL138	Events selected in the INTC_VSSEL138 register (all selected events share the vector).
0x0000_026C	155	139	INT_VSSEL139	Events selected in the INTC_VSSEL139 register (all selected events share the vector).
0x0000_0270	156	140	INT_VSSEL140	Events selected in the INTC_VSSEL140 register (all selected events share the vector).
0x0000_0274	157	141	INT_VSSEL141	Events selected in the INTC_VSSEL141 register (all selected events share the vector).
0x0000_0278	158	142	INT_VSSEL142	Events selected in the INTC_VSSEL142 register (all selected events share the vector).
0x0000_027C	159	143	INT_VSSEL143	Events selected in the INTC_VSSEL143 register (all selected events share the vector).

Note:

- For the event numbers, refer to Table 10-3.

10.3.2 Event Numbers

The interrupt events are generated by the peripherals. When an interrupt event is linked to the IRQ input of the NVIC (CPU), it is called “interrupt source”. When an interrupt event is linked to the RXEV input of the CPU, it is called “event source”.

Table 10-3 Event table

Vector number	Interrupt source	Name	Connect to NVIC	Connect to AOS	Corresponding INTC_SEL ^{*1}	Corresponding INTC_VSSEL
0000h	PORT	PORT_EIRQ0	✓	✓	INTC_SEL32~37	INTC_VSSEL128[0]
001h		PORT_EIRQ1	✓	✓	INTC_SEL32~37	INTC_VSSEL128[1]
002h		PORT_EIRQ2	✓	✓	INTC_SEL32~37	INTC_VSSEL128[2]
003h		PORT_EIRQ3	✓	✓	INTC_SEL32~37	INTC_VSSEL128[3]
004h		PORT_EIRQ4	✓	✓	INTC_SEL32~37	INTC_VSSEL128[4]
005h		PORT_EIRQ5	✓	✓	INTC_SEL32~37	INTC_VSSEL128[5]
006h		PORT_EIRQ6	✓	✓	INTC_SEL32~37	INTC_VSSEL128[6]
007h		PORT_EIRQ7	✓	✓	INTC_SEL32~37	INTC_VSSEL128[7]
008h		PORT_EIRQ8	✓	✓	INTC_SEL32~37	INTC_VSSEL128[8]
009h		PORT_EIRQ9	✓	✓	INTC_SEL32~37	INTC_VSSEL128[9]
00Ah		PORT_EIRQ10	✓	✓	INTC_SEL32~37	INTC_VSSEL128[10]
00Bh		PORT_EIRQ11	✓	✓	INTC_SEL32~37	INTC_VSSEL128[11]
00Ch		PORT_EIRQ12	✓	✓	INTC_SEL32~37	INTC_VSSEL128[12]
00Dh		PORT_EIRQ13	✓	✓	INTC_SEL32~37	INTC_VSSEL128[13]
00Eh		PORT_EIRQ14	✓	✓	INTC_SEL32~37	INTC_VSSEL128[14]
00Fh		PORT_EIRQ15	✓	✓	INTC_SEL32~37	INTC_VSSEL128[15]
010h	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[16]
011h	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[17]
012h	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[18]
013h	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[19]
014h	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[20]
015h	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[21]
016h	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[22]
017h	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[23]
018h	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[24]
019h	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[25]
01Ah	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[26]
01Bh	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[27]
01Ch	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[28]
01Dh	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[29]
01Eh	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[30]
01Fh	-	-	-	-	INTC_SEL32~37	INTC_VSSEL128[31]

Vector number	Interrupt source	Name	Connect to NVIC	Connect to AOS	Corresponding INTC_SEL* ¹	Corresponding INTC_VSEL
020h	DMA_1	DMA_1_TC0	✓	✓	INTC_SEL38~43	INTC_VSEL129[0]
021h		DMA_1_TC1	✓	✓	INTC_SEL38~43	INTC_VSEL129[1]
022h		DMA_1_TC2	✓	✓	INTC_SEL38~43	INTC_VSEL129[2]
023h		DMA_1_TC3	✓	✓	INTC_SEL38~43	INTC_VSEL129[3]
024h		DMA_1_TC4	✓	✓	INTC_SEL38~43	INTC_VSEL129[4]
025h		DMA_1_TC5	✓	✓	INTC_SEL38~43	INTC_VSEL129[5]
026h		DMA_1_TC6	✓	✓	INTC_SEL38~43	INTC_VSEL129[6]
027h		DMA_1_TC7	✓	✓	INTC_SEL38~43	INTC_VSEL129[7]
028h		DMA_1_BT0	✓	✓	INTC_SEL38~43	INTC_VSEL129[8]
029h		DMA_1_BT1	✓	✓	INTC_SEL38~43	INTC_VSEL129[9]
02Ah		DMA_1_BT2	✓	✓	INTC_SEL38~43	INTC_VSEL129[10]
02Bh		DMA_1_BT3	✓	✓	INTC_SEL38~43	INTC_VSEL129[11]
02Ch		DMA_1_BT4	✓	✓	INTC_SEL38~43	INTC_VSEL129[12]
02Dh		DMA_1_BT5	✓	✓	INTC_SEL38~43	INTC_VSEL129[13]
02Eh		DMA_1_BT6	✓	✓	INTC_SEL38~43	INTC_VSEL129[14]
02Eh		DMA_1_BT7	✓	✓	INTC_SEL38~43	INTC_VSEL129[15]
030h		DMA_1_ERR	✓	-	INTC_SEL38~43	INTC_VSEL129[16]
031h	EFM	EFM_PEERR	✓	-	INTC_SEL38~43	INTC_VSEL129[17]
032h		EFM_RDCOL	✓	-	INTC_SEL38~43	INTC_VSEL129[18]
033h		EFM_OPTEND	✓	✓	INTC_SEL38~43	INTC_VSEL129[19]
034h	USBFS	USBFS_SOF	-	✓	INTC_SEL38~43	INTC_VSEL129[20]
035h	USBHS	USBHS_SOF	-	✓	INTC_SEL38~43	INTC_VSEL129[21]
036h	QSPI	QSPI_INTR	✓	-	INTC_SEL38~43	INTC_VSEL129[22]
037h	DCU	DCU1	✓	✓	INTC_SEL38~43	INTC_VSEL129[23]
038h		DCU2	✓	✓	INTC_SEL38~43	INTC_VSEL129[24]
039h		DCU3	✓	✓	INTC_SEL38~43	INTC_VSEL129[25]
03Ah		DCU4	✓	✓	INTC_SEL38~43	INTC_VSEL129[26]
03Bh		DCU5	✓	✓	INTC_SEL38~43	INTC_VSEL129[27]
03Ch		DCU6	✓	✓	INTC_SEL38~43	INTC_VSEL129[28]
03Dh		DCU7	✓	✓	INTC_SEL38~43	INTC_VSEL129[29]
03Eh		DCU8	✓	✓	INTC_SEL38~43	INTC_VSEL129[30]
03Fh	-	-	-	-	INTC_SEL38~43	INTC_VSEL129[31]
040h	DMA_2	DMA_2_TC0	✓	✓	INTC_SEL44~49	INTC_VSEL130[0]
041h		DMA_2_TC1	✓	✓	INTC_SEL44~49	INTC_VSEL130[1]
042h		DMA_2_TC2	✓	✓	INTC_SEL44~49	INTC_VSEL130[2]
043h		DMA_2_TC3	✓	✓	INTC_SEL44~49	INTC_VSEL130[3]
044h		DMA_2_TC4	✓	✓	INTC_SEL44~49	INTC_VSEL130[4]
045h		DMA_2_TC5	✓	✓	INTC_SEL44~49	INTC_VSEL130[5]

Vector number	Interrupt source	Name	Connect to NVIC	Connect to AOS	Corresponding INTC_SEL* ¹	Corresponding INTC_VSEL
046h	DMA	DMA_2_TC6	✓	✓	INTC_SEL44~49	INTC_VSEL130[6]
047h		DMA_2_TC7	✓	✓	INTC_SEL44~49	INTC_VSEL130[7]
048h		DMA_2_BTCo	✓	✓	INTC_SEL44~49	INTC_VSEL130[8]
049h		DMA_2_BTCl	✓	✓	INTC_SEL44~49	INTC_VSEL130[9]
04Ah		DMA_2_BTCl	✓	✓	INTC_SEL44~49	INTC_VSEL130[10]
04Bh		DMA_2_BTCl	✓	✓	INTC_SEL44~49	INTC_VSEL130[11]
04Ch		DMA_2_BTCl	✓	✓	INTC_SEL44~49	INTC_VSEL130[12]
04Dh		DMA_2_BTCl	✓	✓	INTC_SEL44~49	INTC_VSEL130[13]
04Eh		DMA_2_BTCl	✓	✓	INTC_SEL44~49	INTC_VSEL130[14]
04Fh		DMA_2_BTCl	✓	✓	INTC_SEL44~49	INTC_VSEL130[15]
050h		DMA_2_ERR	✓	-	INTC_SEL44~49	INTC_VSEL130[16]
051h	-	-	-	-	INTC_SEL44~49	INTC_VSEL130[17]
052h	-	-	-	-	INTC_SEL44~49	INTC_VSEL130[18]
053h	MAU	MAU_SQRT	✓	✓	INTC_SEL44~49	INTC_VSEL130[19]
054h	DVP	DVP_FRAMSTA	✓	✓	INTC_SEL44~49	INTC_VSEL130[20]
055h		DVP_LINESTA	✓	✓	INTC_SEL44~49	INTC_VSEL130[21]
056h		DVP_LINEEND	✓	✓	INTC_SEL44~49	INTC_VSEL130[22]
057h		DVP_FRAMEND	✓	✓	INTC_SEL44~49	INTC_VSEL130[23]
058h		DVP_SQUERR	✓	✓	INTC_SEL44~49	INTC_VSEL130[24]
059h		DVP_FIFOERR	✓	✓	INTC_SEL44~49	INTC_VSEL130[25]
05Ah		DVP_DMAREQ	-	✓	INTC_SEL44~49	INTC_VSEL130[26]
05Bh	FMAC	FMAC_1_FIR	✓	✓	INTC_SEL44~49	INTC_VSEL130[27]
05Ch		FMAC_2_FIR	✓	✓	INTC_SEL44~49	INTC_VSEL130[28]
05Dh		FMAC_3_FIR	✓	✓	INTC_SEL44~49	INTC_VSEL130[29]
05Eh		FMAC_4_FIR	✓	✓	INTC_SEL44~49	INTC_VSEL130[30]
05Fh	-	-	-	-	INTC_SEL44~49	INTC_VSEL130[31]
060h	Timer0_1	TMR0_1_CMPA	✓	✓	INTC_SEL50~55	INTC_VSEL131[0]
061h		TMR0_1_CMPB	✓	✓	INTC_SEL50~55	INTC_VSEL131[1]
062h	Timer0_2	TMR0_2_CMPA	✓	✓	INTC_SEL50~55	INTC_VSEL131[2]
063h		TMR0_2_CMPB	✓	✓	INTC_SEL50~55	INTC_VSEL131[3]
064h	Timer2_1	TMR2_1_CMPA	✓	✓	INTC_SEL50~55	INTC_VSEL131[4]
065h		TMR2_1_CMPB	✓	✓	INTC_SEL50~55	INTC_VSEL131[5]
066h		TMR2_1_OVFA	✓	-	INTC_SEL50~55	INTC_VSEL131[6]
067h		TMR2_1_OVFB	✓	-	INTC_SEL50~55	INTC_VSEL131[7]
068h	Timer2_2	TMR2_2_CMPA	✓	✓	INTC_SEL50~55	INTC_VSEL131[8]
069h		TMR2_2_CMPB	✓	✓	INTC_SEL50~55	INTC_VSEL131[9]
06Ah		TMR2_2_OVFA	✓	-	INTC_SEL50~55	INTC_VSEL131[10]
06Bh		TMR2_2_OVFB	✓	-	INTC_SEL50~55	INTC_VSEL131[11]

Vector number	Interrupt source	Name	Connect to NVIC	Connect to AOS	Corresponding INTC_SEL* ¹	Corresponding INTC_VSEL
06Ch	Timer2_3	TMR2_3_CMPA	✓	✓	INTC_SEL50~55	INTC_VSEL131[12]
06Dh		TMR2_3_CMPB	✓	✓	INTC_SEL50~55	INTC_VSEL131[13]
06Eh		TMR2_3_OVFA	✓	-	INTC_SEL50~55	INTC_VSEL131[14]
06Fh		TMR2_3_OVFB	✓	-	INTC_SEL50~55	INTC_VSEL131[15]
070h	Timer2_4	TMR2_4_CMPA	✓	✓	INTC_SEL50~55	INTC_VSEL131[16]
071h		TMR2_4_CMPB	✓	✓	INTC_SEL50~55	INTC_VSEL131[17]
072h		TMR2_4_OVFA	✓	-	INTC_SEL50~55	INTC_VSEL131[18]
073h		TMR2_4_OVFB	✓	-	INTC_SEL50~55	INTC_VSEL131[19]
074h	-	-	-	-	INTC_SEL50~55	INTC_VSEL131[20]
075h	-	-	-	-	INTC_SEL50~55	INTC_VSEL131[21]
076h	-	-	-	-	INTC_SEL50~55	INTC_VSEL131[22]
077h	-	-	-	-	INTC_SEL50~55	INTC_VSEL131[23]
078h	RTC	RTC_TP	✓	-	INTC_SEL50~55	INTC_VSEL131[24]
079h		RTC_ALM	✓	✓	INTC_SEL50~55	INTC_VSEL131[25]
07Ah		RTC_PRD	✓	✓	INTC_SEL50~55	INTC_VSEL131[26]
07Bh	-	-	-	-	INTC_SEL50~55	INTC_VSEL131[27]
07Ch	-	-	-	-	INTC_SEL50~55	INTC_VSEL131[28]
07Dh	XTAL	XTAL_STOP	✓	-	INTC_SEL50~55	INTC_VSEL131[29]
07Eh	WKTM	WKTM_PRD	✓	-	INTC_SEL50~55	INTC_VSEL131[30]
07Fh	SWDT	SWDT_REFUDF	✓	-	INTC_SEL50~55	INTC_VSEL131[31]
080h	Timer6_1	TMR6_1_GCMA	✓	✓	INTC_SEL56~61	INTC_VSEL132[0]
081h		TMR6_1_GCMB	✓	✓	INTC_SEL56~61	INTC_VSEL132[1]
082h		TMR6_1_GCMC	✓	✓	INTC_SEL56~61	INTC_VSEL132[2]
083h		TMR6_1_GCMD	✓	✓	INTC_SEL56~61	INTC_VSEL132[3]
084h		TMR6_1_GCME	✓	✓	INTC_SEL56~61	INTC_VSEL132[4]
085h		TMR6_1_GCMF	✓	✓	INTC_SEL56~61	INTC_VSEL132[5]
086h		TMR6_1_GOVF	✓	✓	INTC_SEL56~61	INTC_VSEL132[6]
087h		TMR6_1_GUDF	✓	✓	INTC_SEL56~61	INTC_VSEL132[7]
088h	Timer4_1	TMR4_1_G/SCMUH	✓	✓	INTC_SEL56~61	INTC_VSEL132[8]
089h		TMR4_1_G/SCMUL	✓	✓	INTC_SEL56~61	INTC_VSEL132[9]
08Ah		TMR4_1_G/SCMVH	✓	✓	INTC_SEL56~61	INTC_VSEL132[10]
08Bh		TMR4_1_G/SCMVL	✓	✓	INTC_SEL56~61	INTC_VSEL132[11]
08Ch		TMR4_1_G/SCMWH	✓	✓	INTC_SEL56~61	INTC_VSEL132[12]
08Dh		TMR4_1_G/SCMWL	✓	✓	INTC_SEL56~61	INTC_VSEL132[13]
08Eh		TMR4_1_GOVF	✓	-	INTC_SEL56~61	INTC_VSEL132[14]
08Fh		TMR4_1_GUDF	✓	-	INTC_SEL56~61	INTC_VSEL132[15]
090h	Timer6_2	TMR6_2_GCMA	✓	✓	INTC_SEL56~61	INTC_VSEL132[16]
091h		TMR6_2_GCMB	✓	✓	INTC_SEL56~61	INTC_VSEL132[17]

Vector number	Interrupt source	Name	Connect to NVIC	Connect to AOS	Corresponding INTC_SEL* ¹	Corresponding INTC_VSEL
092h	Timer4_2	TMR6_2_GCMC	✓	✓	INTC_SEL56~61	INTC_VSEL132[18]
093h		TMR6_2_GCMD	✓	✓	INTC_SEL56~61	INTC_VSEL132[19]
094h		TMR6_2_GCME	✓	✓	INTC_SEL56~61	INTC_VSEL132[20]
095h		TMR6_2_GCMF	✓	✓	INTC_SEL56~61	INTC_VSEL132[21]
096h		TMR6_2_GOVF	✓	✓	INTC_SEL56~61	INTC_VSEL132[22]
097h		TMR6_2_GUDF	✓	✓	INTC_SEL56~61	INTC_VSEL132[23]
098h	Timer4_2	TMR4_2_GSCMUH	✓	✓	INTC_SEL56~61	INTC_VSEL132[24]
099h		TMR4_2_G/SCMUL	✓	✓	INTC_SEL56~61	INTC_VSEL132[25]
09Ah		TMR4_2_G/SCMVH	✓	✓	INTC_SEL56~61	INTC_VSEL132[26]
09Bh		TMR4_2_G/SCMVL	✓	✓	INTC_SEL56~61	INTC_VSEL132[27]
09Ch		TMR4_2_G/SCMWH	✓	✓	INTC_SEL56~61	INTC_VSEL132[28]
09Dh		TMR4_2_G/SCMWL	✓	✓	INTC_SEL56~61	INTC_VSEL132[29]
09Eh		TMR4_2_GOVF	✓	-	INTC_SEL56~61	INTC_VSEL132[30]
09Fh		TMR4_2_GUDF	✓	-	INTC_SEL56~61	INTC_VSEL132[31]
0A0h	Timer6_3	TMR6_3_GCMA	✓	✓	INTC_SEL62~67	INTC_VSEL133[0]
0A1h		TMR6_3_GCMB	✓	✓	INTC_SEL62~67	INTC_VSEL133[1]
0A2h		TMR6_3_GCMC	✓	✓	INTC_SEL62~67	INTC_VSEL133[2]
0A3h		TMR6_3_GCMD	✓	✓	INTC_SEL62~67	INTC_VSEL133[3]
0A4h		TMR6_3_GCME	✓	✓	INTC_SEL62~67	INTC_VSEL133[4]
0A5h		TMR6_3_GCMF	✓	✓	INTC_SEL62~67	INTC_VSEL133[5]
0A6h		TMR6_3_GOVF	✓	✓	INTC_SEL62~67	INTC_VSEL133[6]
0A7h		TMR6_3_GUDF	✓	✓	INTC_SEL62~67	INTC_VSEL133[7]
0A8h	Timer4_3	TMR4_3_G/SCMUH	✓	✓	INTC_SEL62~67	INTC_VSEL133[8]
0A9h		TMR4_3_G/SCMUL	✓	✓	INTC_SEL62~67	INTC_VSEL133[9]
0AAh		TMR4_3_G/SCMVH	✓	✓	INTC_SEL62~67	INTC_VSEL133[10]
0ABh		TMR4_3_G/SCMVL	✓	✓	INTC_SEL62~67	INTC_VSEL133[11]
0ACh		TMR4_3_G/SCMWH	✓	✓	INTC_SEL62~67	INTC_VSEL133[12]
0ADh		TMR4_3_G/SCMWL	✓	✓	INTC_SEL62~67	INTC_VSEL133[13]
0AEh		TMR4_3_GOVF	✓	-	INTC_SEL62~67	INTC_VSEL133[14]
0AFh		TMR4_3_GUDF	✓	-	INTC_SEL62~67	INTC_VSEL133[15]
0B0h	Timer6_1	TMR6_1_GDTE	✓	-	INTC_SEL62~67	INTC_VSEL133[16]
0B1h		-	-	-	INTC_SEL62~67	INTC_VSEL133[17]
0B2h		-	-	-	INTC_SEL62~67	INTC_VSEL133[18]
0B3h		TMR6_1_SCMA	✓	✓	INTC_SEL62~67	INTC_VSEL133[19]
0B4h		TMR6_1_SCMB	✓	✓	INTC_SEL62~67	INTC_VSEL133[20]
0B5h	Timer4_1	TMR4_1_RLOU	✓	-	INTC_SEL62~67	INTC_VSEL133[21]
0B6h		TMR4_1_RLOV	✓	-	INTC_SEL62~67	INTC_VSEL133[22]
0B7h		TMR4_1_RLOW	✓	-	INTC_SEL62~67	INTC_VSEL133[23]

Vector number	Interrupt source	Name	Connect to NVIC	Connect to AOS	Corresponding INTC_SEL*¹	Corresponding INTC_VSEL
0B8h	Timer6_2	TMR6_2_GDTE	✓	-	INTC_SEL62~67	INTC_VSEL133[24]
0B9h	-	-	-	-	INTC_SEL62~67	INTC_VSEL133[25]
0BAh	-	-	-	-	INTC_SEL62~67	INTC_VSEL133[26]
0BBh	Timer6_2	TMR6_2_SCMA	✓	✓	INTC_SEL62~67	INTC_VSEL133[27]
0BCh		TMR6_2_SCMB	✓	✓	INTC_SEL62~67	INTC_VSEL133[28]
0BDh	Timer4_2	TMR4_2_RLOU	✓	-	INTC_SEL62~67	INTC_VSEL133[29]
0BEh		TMR4_2_RLOV	✓	-	INTC_SEL62~67	INTC_VSEL133[30]
0BFh		TMR4_2_RLOW	✓	-	INTC_SEL62~67	INTC_VSEL133[31]
0C0h	Timer6_3	TMR6_3_GDTE	✓	-	INTC_SEL68~73	INTC_VSEL134[0]
0C1h	-	-	-	-	INTC_SEL68~73	INTC_VSEL134[1]
0C2h	-	-	-	-	INTC_SEL68~73	INTC_VSEL134[2]
0C3h	Timer6_3	TMR6_3_SCMA	✓	✓	INTC_SEL68~73	INTC_VSEL134[3]
0C4h		TMR6_3_SCMB	✓	✓	INTC_SEL68~73	INTC_VSEL134[4]
0C5h	Timer4_3	TMR4_3_RLOU	✓	-	INTC_SEL68~73	INTC_VSEL134[5]
0C6h		TMR4_3_RLOV	✓	-	INTC_SEL68~73	INTC_VSEL134[6]
0C7h		TMR4_3_RLOW	✓	-	INTC_SEL68~73	INTC_VSEL134[7]
0C8h	-	-	-	-	INTC_SEL68~73	INTC_VSEL134[8]
0C9h	-	-	-	-	INTC_SEL68~73	INTC_VSEL134[9]
0CAh	-	-	-	-	INTC_SEL68~73	INTC_VSEL134[10]
0CBh	-	-	-	-	INTC_SEL68~73	INTC_VSEL134[11]
0CCh	-	-	-	-	INTC_SEL68~73	INTC_VSEL134[12]
0CDh	-	-	-	-	INTC_SEL68~73	INTC_VSEL134[13]
0CEh	-	-	-	-	INTC_SEL68~73	INTC_VSEL134[14]
0CFh	-	-	-	-	INTC_SEL68~73	INTC_VSEL134[15]
0D0h	Timer6_4	TMR6_4_GCMA	✓	✓	INTC_SEL68~73	INTC_VSEL134[16]
0D1h		TMR6_4_GCMB	✓	✓	INTC_SEL68~73	INTC_VSEL134[17]
0D2h		TMR6_4_GCMC	✓	✓	INTC_SEL68~73	INTC_VSEL134[18]
0D3h		TMR6_4_GCMD	✓	✓	INTC_SEL68~73	INTC_VSEL134[19]
0D4h		TMR6_4_GCME	✓	✓	INTC_SEL68~73	INTC_VSEL134[20]
0D5h		TMR6_4_GCMF	✓	✓	INTC_SEL68~73	INTC_VSEL134[21]
0D6h		TMR6_4_GOVF	✓	✓	INTC_SEL68~73	INTC_VSEL134[22]
0D7h		TMR6_4_GUDF	✓	✓	INTC_SEL68~73	INTC_VSEL134[23]
0D8h		TMR6_4_GDTE	✓	-	INTC_SEL68~73	INTC_VSEL134[24]
0D9h	-	-	-	-	INTC_SEL68~73	INTC_VSEL134[25]
0DAh	-	-	-	-	INTC_SEL68~73	INTC_VSEL134[26]
0DBh	Timer6_4	TMR6_4_SCMA	✓	✓	INTC_SEL68~73	INTC_VSEL134[27]
0DCh		TMR6_4_SCMB	✓	✓	INTC_SEL68~73	INTC_VSEL134[28]
0DDh	-	-	-	-	INTC_SEL68~73	INTC_VSEL134[29]

Vector number	Interrupt source	Name	Connect to NVIC	Connect to AOS	Corresponding INTC_SEL*¹	Corresponding INTC_VSEL
0DEh	-	-	-	-	INTC_SEL68~73	INTC_VSEL134[30]
0DFh	-	-	-	-	INTC_SEL68~73	INTC_VSEL134[31]
0E0h	Timer6_5	TMR6_5_GCMA	✓	✓	INTC_SEL74~79	INTC_VSEL135[0]
0E1h		TMR6_5_GCMB	✓	✓	INTC_SEL74~79	INTC_VSEL135[1]
0E2h		TMR6_5_GCMC	✓	✓	INTC_SEL74~79	INTC_VSEL135[2]
0E3h		TMR6_5_GCMD	✓	✓	INTC_SEL74~79	INTC_VSEL135[3]
0E4h		TMR6_5_GCME	✓	✓	INTC_SEL74~79	INTC_VSEL135[4]
0E5h		TMR6_5_GCMF	✓	✓	INTC_SEL74~79	INTC_VSEL135[5]
0E6h		TMR6_5_GOVF	✓	✓	INTC_SEL74~79	INTC_VSEL135[6]
0E7h		TMR6_5_GUDF	✓	✓	INTC_SEL74~79	INTC_VSEL135[7]
0E8h		TMR6_5_GDTE	✓	-	INTC_SEL74~79	INTC_VSEL135[8]
0E9h	-	-	-	-	INTC_SEL74~79	INTC_VSEL135[9]
0EAh	-	-	-	-	INTC_SEL74~79	INTC_VSEL135[10]
0EBh	Timer6_5	TMR6_5_SCMA	✓	✓	INTC_SEL74~79	INTC_VSEL135[11]
0ECh		TMR6_5_SCMB	✓	✓	INTC_SEL74~79	INTC_VSEL135[12]
0EDh	TimerA_1	TMRA_1_OVF	✓	✓	INTC_SEL74~79	INTC_VSEL135[13]
0EEh		TMRA_1_UDF	✓	✓	INTC_SEL74~79	INTC_VSEL135[14]
0EFh		TMRA_1_CMP	✓	✓	INTC_SEL74~79	INTC_VSEL135[15]
0F0h	Timer6_6	TMR6_6_GCMA	✓	✓	INTC_SEL74~79	INTC_VSEL135[16]
0F1h		TMR6_6_GCMB	✓	✓	INTC_SEL74~79	INTC_VSEL135[17]
0F2h		TMR6_6_GCMC	✓	✓	INTC_SEL74~79	INTC_VSEL135[18]
0F3h		TMR6_6_GCMD	✓	✓	INTC_SEL74~79	INTC_VSEL135[19]
0F4h		TMR6_6_GCME	✓	✓	INTC_SEL74~79	INTC_VSEL135[20]
0F5h		TMR6_6_GCMF	✓	✓	INTC_SEL74~79	INTC_VSEL135[21]
0F6h		TMR6_6_GOVF	✓	✓	INTC_SEL74~79	INTC_VSEL135[22]
0F7h		TMR6_6_GUDF	✓	✓	INTC_SEL74~79	INTC_VSEL135[23]
0F8h		TMR6_6_GDTE	✓	-	INTC_SEL74~79	INTC_VSEL135[24]
0F9h	-	-	-	-	INTC_SEL74~79	INTC_VSEL135[25]
0FAh	-	-	-	-	INTC_SEL74~79	INTC_VSEL135[26]
0FBh	Timer6_6	TMR6_6_SCMA	✓	✓	INTC_SEL74~79	INTC_VSEL135[27]
0FCh		TMR6_6_SCMB	✓	✓	INTC_SEL74~79	INTC_VSEL135[28]
0FDh	TimerA_2	TMRA_2_OVF	✓	✓	INTC_SEL74~79	INTC_VSEL135[29]
0FEh		TMRA_2_UDF	✓	✓	INTC_SEL74~79	INTC_VSEL135[30]
OFFh		TMRA_2_CMP	✓	✓	INTC_SEL74~79	INTC_VSEL135[31]
100h	Timer6_7	TMR6_7_GCMA	✓	✓	INTC_SEL80~85	INTC_VSEL136[0]
101h		TMR6_7_GCMB	✓	✓	INTC_SEL80~85	INTC_VSEL136[1]
102h		TMR6_7_GCMC	✓	✓	INTC_SEL80~85	INTC_VSEL136[2]
103h		TMR6_7_GCMD	✓	✓	INTC_SEL80~85	INTC_VSEL136[3]

Vector number	Interrupt source	Name	Connect to NVIC	Connect to AOS	Corresponding INTC_SEL* ¹	Corresponding INTC_VSEL
104h	Timer6_7	TMR6_7_GCME	✓	✓	INTC_SEL80~85	INTC_VSEL136[4]
105h		TMR6_7_GCMF	✓	✓	INTC_SEL80~85	INTC_VSEL136[5]
106h		TMR6_7_GOVF	✓	✓	INTC_SEL80~85	INTC_VSEL136[6]
107h		TMR6_7_GUDF	✓	✓	INTC_SEL80~85	INTC_VSEL136[7]
108h		TMR6_7_GDTE	✓	-	INTC_SEL80~85	INTC_VSEL136[8]
109h	-	-	-	-	INTC_SEL80~85	INTC_VSEL136[9]
10Ah	-	-	-	-	INTC_SEL80~85	INTC_VSEL136[10]
10Bh	Timer6_7	TMR6_7_SCMA	✓	✓	INTC_SEL80~85	INTC_VSEL136[11]
10Ch		TMR6_7_SCMB	✓	✓	INTC_SEL80~85	INTC_VSEL136[12]
10Dh	TimerA_3	TMRA_3_OVF	✓	✓	INTC_SEL80~85	INTC_VSEL136[13]
10Eh		TMRA_3_UDF	✓	✓	INTC_SEL80~85	INTC_VSEL136[14]
10Fh		TMRA_3_CMP	✓	✓	INTC_SEL80~85	INTC_VSEL136[15]
110h	Timer6_8	TMR6_8_GCMA	✓	✓	INTC_SEL80~85	INTC_VSEL136[16]
111h		TMR6_8_GCMB	✓	✓	INTC_SEL80~85	INTC_VSEL136[17]
112h		TMR6_8_GCMC	✓	✓	INTC_SEL80~85	INTC_VSEL136[18]
113h		TMR6_8_GCMD	✓	✓	INTC_SEL80~85	INTC_VSEL136[19]
114h		TMR6_8_GCME	✓	✓	INTC_SEL80~85	INTC_VSEL136[20]
115h		TMR6_8_GCMF	✓	✓	INTC_SEL80~85	INTC_VSEL136[21]
116h		TMR6_8_GOVF	✓	✓	INTC_SEL80~85	INTC_VSEL136[22]
117h		TMR6_8_GUDF	✓	✓	INTC_SEL80~85	INTC_VSEL136[23]
118h		TMR6_8_GDTE	✓	-	INTC_SEL80~85	INTC_VSEL136[24]
119h	-	-	-	-	INTC_SEL80~85	INTC_VSEL136[25]
11Ah	-	-	-	-	INTC_SEL80~85	INTC_VSEL136[26]
11Bh	Timer6_8	TMR6_8_SCMA	✓	✓	INTC_SEL80~85	INTC_VSEL136[27]
11Ch		TMR6_8_SCMB	✓	✓	INTC_SEL80~85	INTC_VSEL136[28]
11Dh	TimerA_4	TMRA_4_OVF	✓	✓	INTC_SEL80~85	INTC_VSEL136[29]
11Eh		TMRA_4_UDF	✓	✓	INTC_SEL80~85	INTC_VSEL136[30]
11Fh		TMRA_4_CMP	✓	✓	INTC_SEL80~85	INTC_VSEL136[31]
120h	EMB	EMB_GR0	✓	-	INTC_SEL86~91	INTC_VSEL137[0]
121h		EMB_GR1	✓	-	INTC_SEL86~91	INTC_VSEL137[1]
122h		EMB_GR2	✓	-	INTC_SEL86~91	INTC_VSEL137[2]
123h		EMB_GR3	✓	-	INTC_SEL86~91	INTC_VSEL137[3]
124h		EMB_GR4	✓	-	INTC_SEL86~91	INTC_VSEL137[4]
125h		EMB_GR5	✓	-	INTC_SEL86~91	INTC_VSEL137[5]
126h		EMB_GR6	✓	-	INTC_SEL86~91	INTC_VSEL137[6]
127h	USBHS	USBHS_EP1_OUT	✓	-	INTC_SEL86~91	INTC_VSEL137[7]
128h		USBHS_EP1_IN	✓	-	INTC_SEL86~91	INTC_VSEL137[8]
129h		USBHS_GLB	✓	-	INTC_SEL86~91	INTC_VSEL137[9]

Vector number	Interrupt source	Name	Connect to NVIC	Connect to AOS	Corresponding INTC_SEL*¹	Corresponding INTC_VSEL
12Ah		USBHS_WKUP	✓	-	INTC_SEL86~91	INTC_VSEL137[10]
12Bh	AOS_STRG	AOS_STRG* ²	-	✓	INTC_SEL86~91	INTC_VSEL137[11]
12Ch	USART_1	USART_1_EI	✓	✓	INTC_SEL86~91	INTC_VSEL137[12]
12Dh		USART_1_RI	✓	✓	INTC_SEL86~91	INTC_VSEL137[13]
12Eh		USART_1_TI	✓	✓	INTC_SEL86~91	INTC_VSEL137[14]
12Fh		USART_1_TCI	✓	✓	INTC_SEL86~91	INTC_VSEL137[15]
130h		USART_1_RTO	✓	✓	INTC_SEL86~91	INTC_VSEL137[16]
131h	USART_2	USART_2_EI	✓	✓	INTC_SEL86~91	INTC_VSEL137[17]
132h		USART_2_RI	✓	✓	INTC_SEL86~91	INTC_VSEL137[18]
133h		USART_2_TI	✓	✓	INTC_SEL86~91	INTC_VSEL137[19]
134h		USART_2_TCI	✓	✓	INTC_SEL86~91	INTC_VSEL137[20]
135h		USART_2_RTO	✓	✓	INTC_SEL86~91	INTC_VSEL137[21]
136h	SPI_1	SPI_1_SPRI	✓	✓	INTC_SEL86~91	INTC_VSEL137[22]
137h		SPI_1_SPTI	✓	✓	INTC_SEL86~91	INTC_VSEL137[23]
138h		SPI_1_SPII	✓	✓	INTC_SEL86~91	INTC_VSEL137[24]
139h		SPI_1_SPEI	✓	✓	INTC_SEL86~91	INTC_VSEL137[25]
13Ah		SPI_1_SPEND	-	✓	INTC_SEL86~91	INTC_VSEL137[26]
13Bh	SPI_2	SPI_2_SPRI	✓	✓	INTC_SEL86~91	INTC_VSEL137[27]
13Ch		SPI_2_SPTI	✓	✓	INTC_SEL86~91	INTC_VSEL137[28]
13Dh		SPI_2_SPII	✓	✓	INTC_SEL86~91	INTC_VSEL137[29]
13Eh		SPI_2_SPEI	✓	✓	INTC_SEL86~91	INTC_VSEL137[30]
13Fh		SPI_2_STEND	-	✓	INTC_SEL86~91	INTC_VSEL137[31]
140h	TimerA_5	TMRA_5_OVF	✓	✓	INTC_SEL92~97	INTC_VSEL138[0]
141h		TMRA_5_UDF	✓	✓	INTC_SEL92~97	INTC_VSEL138[1]
142h		TMRA_5_CMP	✓	✓	INTC_SEL92~97	INTC_VSEL138[2]
143h	TimerA_6	TMRA_6_OVF	✓	✓	INTC_SEL92~97	INTC_VSEL138[3]
144h		TMRA_6_UDF	✓	✓	INTC_SEL92~97	INTC_VSEL138[4]
145h		TMRA_6_CMP	✓	✓	INTC_SEL92~97	INTC_VSEL138[5]
146h	TimerA_7	TMRA_7_OVF	✓	✓	INTC_SEL92~97	INTC_VSEL138[6]
147h		TMRA_7_UDF	✓	✓	INTC_SEL92~97	INTC_VSEL138[7]
148h		TMRA_7_CMP	✓	✓	INTC_SEL92~97	INTC_VSEL138[8]
149h	TimerA_8	TMRA_8_OVF	✓	✓	INTC_SEL92~97	INTC_VSEL138[9]
14Ah		TMRA_8_UDF	✓	✓	INTC_SEL92~97	INTC_VSEL138[10]
14Bh		TMRA_8_CMP	✓	✓	INTC_SEL92~97	INTC_VSEL138[11]
14Ch	USART_3	USART_3_EI	✓	✓	INTC_SEL92~97	INTC_VSEL138[12]
14Dh		USART_3_RI	✓	✓	INTC_SEL92~97	INTC_VSEL138[13]
14Eh		USART_3_TI	✓	✓	INTC_SEL92~97	INTC_VSEL138[14]
14Fh		USART_3_TCI	✓	✓	INTC_SEL92~97	INTC_VSEL138[15]

Vector number	Interrupt source	Name	Connect to NVIC	Connect to AOS	Corresponding INTC_SEL* ¹	Corresponding INTC_VSEL
150h	USART_4	USART_4_EI	✓	✓	INTC_SEL92~97	INTC_VSEL138[16]
151h		USART_4_RI	✓	✓	INTC_SEL92~97	INTC_VSEL138[17]
152h		USART_4_TI	✓	✓	INTC_SEL92~97	INTC_VSEL138[18]
153h		USART_4_TCI	✓	✓	INTC_SEL92~97	INTC_VSEL138[19]
154h	CAN_1	CAN_1_HOST	✓	-	INTC_SEL92~97	INTC_VSEL138[20]
155h	CAN_2	CAN_2_HOST	✓	-	INTC_SEL92~97	INTC_VSEL138[21]
156h	SPI_3	SPI_3_SPRI	✓	✓	INTC_SEL92~97	INTC_VSEL138[22]
157h		SPI_3_SPTI	✓	✓	INTC_SEL92~97	INTC_VSEL138[23]
158h		SPI_3_SPII	✓	✓	INTC_SEL92~97	INTC_VSEL138[24]
159h		SPI_3_SPEI	✓	✓	INTC_SEL92~97	INTC_VSEL138[25]
15Ah		SPI_3_SPEND	-	✓	INTC_SEL92~97	INTC_VSEL138[26]
15Bh	SPI_4	SPI_4_SPRI	✓	✓	INTC_SEL92~97	INTC_VSEL138[27]
15Ch		SPI_4_SPTI	✓	✓	INTC_SEL92~97	INTC_VSEL138[28]
15Dh		SPI_4_SPII	✓	✓	INTC_SEL92~97	INTC_VSEL138[29]
15Eh		SPI_4_SPEI	✓	✓	INTC_SEL92~97	INTC_VSEL138[30]
15Fh		SPI_4_SPEND	-	✓	INTC_SEL92~97	INTC_VSEL138[31]
160h	TimerA_9	TMRA_9_OVF	✓	✓	INTC_SEL98~103	INTC_VSEL139[0]
161h		TMRA_9_UDF	✓	✓	INTC_SEL98~103	INTC_VSEL139[1]
162h		TMRA_9_CMP	✓	✓	INTC_SEL98~103	INTC_VSEL139[2]
163h	TimerA_10	TMRA_10_OVF	✓	✓	INTC_SEL98~103	INTC_VSEL139[3]
164h		TMRA_10_UDF	✓	✓	INTC_SEL98~103	INTC_VSEL139[4]
165h		TMRA_10_CMP	✓	✓	INTC_SEL98~103	INTC_VSEL139[5]
166h	TimerA_11	TMRA_11_OVF	✓	✓	INTC_SEL98~103	INTC_VSEL139[6]
167h		TMRA_11_UDF	✓	✓	INTC_SEL98~103	INTC_VSEL139[7]
168h		TMRA_11_CMP	✓	✓	INTC_SEL98~103	INTC_VSEL139[8]
169h	TimerA_12	TMRA_12_OVF	✓	✓	INTC_SEL98~103	INTC_VSEL139[9]
16Ah		TMRA_12_UDF	✓	✓	INTC_SEL98~103	INTC_VSEL139[10]
16Bh		TMRA_12_CMP	✓	✓	INTC_SEL98~103	INTC_VSEL139[11]
16Ch	USART_5	USART_5_BRWKPI	✓	✓	INTC_SEL98~103	INTC_VSEL139[12]
16Dh		USART_5_EI	✓	✓	INTC_SEL98~103	INTC_VSEL139[13]
16Eh		USART_5_RI	✓	✓	INTC_SEL98~103	INTC_VSEL139[14]
16Fh		USART_5_TI	✓	✓	INTC_SEL98~103	INTC_VSEL139[15]
170h		USART_5_TCI	✓	✓	INTC_SEL98~103	INTC_VSEL139[16]
171h	USART_6	USART_6_EI	✓	✓	INTC_SEL98~103	INTC_VSEL139[17]
172h		USART_6_RI	✓	✓	INTC_SEL98~103	INTC_VSEL139[18]
173h		USART_6_TI	✓	✓	INTC_SEL98~103	INTC_VSEL139[19]
174h		USART_6_TCI	✓	✓	INTC_SEL98~103	INTC_VSEL139[20]
175h		USART_6_RTO	✓	✓	INTC_SEL98~103	INTC_VSEL139[21]

Vector number	Interrupt source	Name	Connect to NVIC	Connect to AOS	Corresponding INTC_SEL* ¹	Corresponding INTC_VSEL
176h	SPI_5	SPI_5_SPRI	✓	✓	INTC_SEL98~103	INTC_VSEL139[22]
177h		SPI_5_SPTI	✓	✓	INTC_SEL98~103	INTC_VSEL139[23]
178h		SPI_5_SPII	✓	✓	INTC_SEL98~103	INTC_VSEL139[24]
179h		SPI_5_SPEI	✓	✓	INTC_SEL98~103	INTC_VSEL139[25]
17Ah		SPI_5_SPEND	-	✓	INTC_SEL98~103	INTC_VSEL139[26]
17Bh	SPI_6	SPI_6_SPRI	✓	✓	INTC_SEL98~103	INTC_VSEL139[27]
17Ch		SPI_6_SPTI	✓	✓	INTC_SEL98~103	INTC_VSEL139[28]
17Dh		SPI_6_SPII	✓	✓	INTC_SEL98~103	INTC_VSEL139[29]
17Eh		SPI_6_SPEI	✓	✓	INTC_SEL98~103	INTC_VSEL139[30]
17Fh		SPI_6_SPEND	-	✓	INTC_SEL98~103	INTC_VSEL139[31]
180h	I2S_1	I2S_1_TXIRQOUT	✓	✓	INTC_SEL104~109	INTC_VSEL140[0]
181h		I2S_1_RXIRQOUT	✓	✓	INTC_SEL104~109	INTC_VSEL140[1]
182h		I2S_1_ERRIRQOUT	✓	-	INTC_SEL104~109	INTC_VSEL140[2]
183h	I2S_2	I2S_2_TXIRQOUT	✓	✓	INTC_SEL104~109	INTC_VSEL140[3]
184h		I2S_2_RXIRQOUT	✓	✓	INTC_SEL104~109	INTC_VSEL140[4]
185h		I2S_2_ERRIRQOUT	✓	-	INTC_SEL104~109	INTC_VSEL140[5]
186h	USART_7	USART_7_EI	✓	✓	INTC_SEL104~109	INTC_VSEL140[6]
187h		USART_7_RI	✓	✓	INTC_SEL104~109	INTC_VSEL140[7]
188h		USART_7_TI	✓	✓	INTC_SEL104~109	INTC_VSEL140[8]
189h		USART_7_TCI	✓	✓	INTC_SEL104~109	INTC_VSEL140[9]
18Ah		USART_7_RTO	✓	✓	INTC_SEL104~109	INTC_VSEL140[10]
18Bh	USART_8	USART_8_EI	✓	✓	INTC_SEL104~109	INTC_VSEL140[11]
18Ch		USART_8_RI	✓	✓	INTC_SEL104~109	INTC_VSEL140[12]
18Dh		USART_8_TI	✓	✓	INTC_SEL104~109	INTC_VSEL140[13]
18Eh		USART_8_TCI	✓	✓	INTC_SEL104~109	INTC_VSEL140[14]
18Fh	USBFS	USBFS_GLB	✓	-	INTC_SEL104~109	INTC_VSEL140[15]
190h		USBFS_WKUP	✓	-	INTC_SEL104~109	INTC_VSEL140[16]
191h	HASH	HASH_INT	✓	✓	INTC_SEL104~109	INTC_VSEL140[17]
192h	SDIOC_1	SDIOC_1_DMAR	-	✓	INTC_SEL104~109	INTC_VSEL140[18]
193h		SDIOC_1_DMAW	-	✓	INTC_SEL104~109	INTC_VSEL140[19]
194h		SDIOC_1_SD	✓	-	INTC_SEL104~109	INTC_VSEL140[20]
195h	SDIOC_2	SDIOC_2_DMAR	-	✓	INTC_SEL104~109	INTC_VSEL140[21]
196h		SDIOC_2_DMAW	-	✓	INTC_SEL104~109	INTC_VSEL140[22]
197h		SDIOC_2_SD	✓	-	INTC_SEL104~109	INTC_VSEL140[23]
198h	EVENT port	EVENT_PORT1	✓	✓	INTC_SEL104~109	INTC_VSEL140[24]
199h		EVENT_PORT2	✓	✓	INTC_SEL104~109	INTC_VSEL140[25]
19Ah		EVENT_PORT3	✓	✓	INTC_SEL104~109	INTC_VSEL140[26]
19Bh		EVENT_PORT4	✓	✓	INTC_SEL104~109	INTC_VSEL140[27]

Vector number	Interrupt source	Name	Connect to NVIC	Connect to AOS	Corresponding INTC_SEL* ¹	Corresponding INTC_VSEL
19Ch	ETHER	ETH_GLB_INT	✓	-	INTC_SEL104~109	INTC_VSEL140[28]
19Dh		ETH_WKP_INT	✓	-	INTC_SEL104~109	INTC_VSEL140[29]
19Eh		ETH_PPS_OUT_0	-	✓	INTC_SEL104~109	INTC_VSEL140[30]
19Fh		ETH_PPS_OUT_1	-	✓	INTC_SEL104~109	INTC_VSEL140[31]
1A0h	I2S_3	I2S_3_TXIRQOUT	✓	✓	INTC_SEL110~115	INTC_VSEL141[0]
1A1h		I2S_3_RXIRQOUT	✓	✓	INTC_SEL110~115	INTC_VSEL141[1]
1A2h		I2S_3_ERRIRQOUT	✓	-	INTC_SEL110~115	INTC_VSEL141[2]
1A3h	I2S_4	I2S_4_TXIRQOUT	✓	✓	INTC_SEL110~115	INTC_VSEL141[3]
1A4h		I2S_4_RXIRQOUT	✓	✓	INTC_SEL110~115	INTC_VSEL141[4]
1A5h		I2S_4_ERRIRQOUT	✓	-	INTC_SEL110~115	INTC_VSEL141[5]
1A6h	USART_9	USART_9_EI	✓	✓	INTC_SEL110~115	INTC_VSEL141[6]
1A7h		USART_9_RI	✓	✓	INTC_SEL110~115	INTC_VSEL141[7]
1A8h		USART_9_TI	✓	✓	INTC_SEL110~115	INTC_VSEL141[8]
1A9h		USART_9_TCI	✓	✓	INTC_SEL110~115	INTC_VSEL141[9]
1AAh	USART_10	USART_10_BRWKPI	✓	✓	INTC_SEL110~115	INTC_VSEL141[10]
1ABh		USART_10_EI	✓	✓	INTC_SEL110~115	INTC_VSEL141[11]
1ACh		USART_10_RI	✓	✓	INTC_SEL110~115	INTC_VSEL141[12]
1ADh		USART_10_TI	✓	✓	INTC_SEL110~115	INTC_VSEL141[13]
1AEh		USART_10_TCI	✓	✓	INTC_SEL110~115	INTC_VSEL141[14]
1AFh	-	-	-	-	INTC_SEL110~115	INTC_VSEL141[15]
1B0h	I2C_1	I2C_1_RXI	✓	✓	INTC_SEL110~115	INTC_VSEL141[16]
1B1h		I2C_1_TXI	✓	✓	INTC_SEL110~115	INTC_VSEL141[17]
1B2h		I2C_1_TEI	✓	✓	INTC_SEL110~115	INTC_VSEL141[18]
1B3h		I2C_1_EEI	✓	✓	INTC_SEL110~115	INTC_VSEL141[19]
1B4h	I2C_2	I2C_2_RXI	✓	✓	INTC_SEL110~115	INTC_VSEL141[20]
1B5h		I2C_2_TXI	✓	✓	INTC_SEL110~115	INTC_VSEL141[21]
1B6h		I2C_2_TEI	✓	✓	INTC_SEL110~115	INTC_VSEL141[22]
1B7h		I2C_2_EEI	✓	✓	INTC_SEL110~115	INTC_VSEL141[23]
1B8h	I2C_3	I2C_3_RXI	✓	✓	INTC_SEL110~115	INTC_VSEL141[24]
1B9h		I2C_3_TXI	✓	✓	INTC_SEL110~115	INTC_VSEL141[25]
1BAh		I2C_3_TEI	✓	✓	INTC_SEL110~115	INTC_VSEL141[26]
1BBh		I2C_3_EEI	✓	✓	INTC_SEL110~115	INTC_VSEL141[27]
1BCh	ACMP	CMP1	✓	✓	INTC_SEL110~115	INTC_VSEL141[28]
1BDh		CMP2	✓	✓	INTC_SEL110~115	INTC_VSEL141[29]
1BEh		CMP3	✓	✓	INTC_SEL110~115	INTC_VSEL141[30]
1BFh		CMP4	✓	✓	INTC_SEL110~115	INTC_VSEL141[31]
1C0h	I2C_4	I2C_4_RXI	✓	✓	INTC_SEL116~121	INTC_VSEL142[0]
1C1h		I2C_4_TXI	✓	✓	INTC_SEL116~121	INTC_VSEL142[1]

Vector number	Interrupt source	Name	Connect to NVIC	Connect to AOS	Corresponding INTC_SEL*¹	Corresponding INTC_VSEL
1C2h		I2C_4_TEI	✓	✓	INTC_SEL116~121	INTC_VSEL142[2]
1C3h		I2C_4_EEI	✓	✓	INTC_SEL116~121	INTC_VSEL142[3]
1C4h	I2C_5	I2C_5_RXI	✓	✓	INTC_SEL116~121	INTC_VSEL142[4]
1C5h		I2C_5_TXI	✓	✓	INTC_SEL116~121	INTC_VSEL142[5]
1C6h		I2C_5_TEI	✓	✓	INTC_SEL116~121	INTC_VSEL142[6]
1C7h		I2C_5_EEI	✓	✓	INTC_SEL116~121	INTC_VSEL142[7]
1C8h	I2C_6	I2C_6_RXI	✓	✓	INTC_SEL116~121	INTC_VSEL142[8]
1C9h		I2C_6_TXI	✓	✓	INTC_SEL116~121	INTC_VSEL142[9]
1CAh		I2C_6_TEI	✓	✓	INTC_SEL116~121	INTC_VSEL142[10]
1CBh		I2C_6_EEI	✓	✓	INTC_SEL116~121	INTC_VSEL142[11]
1CCh	USART_1	USART_1_WUPI	✓	-	INTC_SEL116~121	INTC_VSEL142[12]
1CDh	PVD	PVD_PVD1	✓	✓	INTC_SEL116~121	INTC_VSEL142[13]
1CEh		PVD_PVD2	✓	✓	INTC_SEL116~121	INTC_VSEL142[14]
1CFh	OTS	OTS	✓	✓	INTC_SEL116~121	INTC_VSEL142[15]
1D0h	FCM	FCMFERRI	✓	-	INTC_SEL116~121	INTC_VSEL142[16]
1D1h		FCMMENDI	✓	-	INTC_SEL116~121	INTC_VSEL142[17]
1D2h		FCMCOVFI	✓	-	INTC_SEL116~121	INTC_VSEL142[18]
1D3h	WDT	WDT_REFUDF	✓	✓	INTC_SEL116~121	INTC_VSEL142[19]
1D4h	CTC	CTC_ERR	✓	-	INTC_SEL116~121	INTC_VSEL142[20]
1D5h	-	-	-	-	INTC_SEL116~121	INTC_VSEL142[21]
1D6h	-	-	-	-	INTC_SEL116~121	INTC_VSEL142[22]
1D7h	-	-	-	-	INTC_SEL116~121	INTC_VSEL142[23]
1D8h	-	-	-	-	INTC_SEL116~121	INTC_VSEL142[24]
1D9h	-	-	-	-	INTC_SEL116~121	INTC_VSEL142[25]
1DAh	-	-	-	-	INTC_SEL116~121	INTC_VSEL142[26]
1DBh	-	-	-	-	INTC_SEL116~121	INTC_VSEL142[27]
1DCh	-	-	-	-	INTC_SEL116~121	INTC_VSEL142[28]
1DDh	-	-	-	-	INTC_SEL116~121	INTC_VSEL142[29]
1DEh	-	-	-	-	INTC_SEL116~121	INTC_VSEL142[30]
1DFh	-	-	-	-	INTC_SEL116~121	INTC_VSEL142[31]
1E0h	ADC_1	ADC_1_EOCA	✓	✓	INTC_SEL122~127	INTC_VSEL143[0]
1E1h		ADC_1_EOCB	✓	✓	INTC_SEL122~127	INTC_VSEL143[1]
1E2h		ADC_1_CMP0	✓	✓	INTC_SEL122~127	INTC_VSEL143[2]
1E3h		ADC_1_CMP1	✓	✓	INTC_SEL122~127	INTC_VSEL143[3]
1E4h	ADC_2	ADC_2_EOCA	✓	✓	INTC_SEL122~127	INTC_VSEL143[4]
1E5h		ADC_2_EOCB	✓	✓	INTC_SEL122~127	INTC_VSEL143[5]
1E6h		ADC_2_CMP0	✓	✓	INTC_SEL122~127	INTC_VSEL143[6]
1E7h		ADC_2_CMP1	✓	✓	INTC_SEL122~127	INTC_VSEL143[7]

Vector number	Interrupt source	Name	Connect to NVIC	Connect to AOS	Corresponding INTC_SEL*1	Corresponding INTC_VSEL
1E8h	ADC_3	ADC_3_EOCA	✓	✓	INTC_SEL122~127	INTC_VSEL143[8]
1E9h		ADC_3_EOCB	✓	✓	INTC_SEL122~127	INTC_VSEL143[9]
1EAh		ADC_3_CMP0	✓	✓	INTC_SEL122~127	INTC_VSEL143[10]
1EBh		ADC_3_CMP1	✓	✓	INTC_SEL122~127	INTC_VSEL143[11]
1EC _h	TRNG	TRNG_END	✓	✓	INTC_SEL122~127	INTC_VSEL143[12]
1ED _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[13]
1EE _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[14]
1EF _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[15]
1F0 _h	NFC	NFC_INT	✓	-	INTC_SEL122~127	INTC_VSEL143[16]
1F1 _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[17]
1F2 _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[18]
1F3 _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[19]
1F4 _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[20]
1F5 _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[21]
1F6 _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[22]
1F7 _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[23]
1F8 _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[24]
1F9 _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[25]
1FA _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[26]
1FB _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[27]
1FC _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[28]
1FD _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[29]
1FE _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[30]
1FF _h	-	-	-	-	INTC_SEL122~127	INTC_VSEL143[31]

Note:

If the event number written to the interrupt selection register does not correspond to any interrupt event, the setting of this register or this bit is invalid.

*1: As INTC_SEL0~31 can select all interrupt events, here only INTC_SEL32~127 are marked.

*2: AOS_STRG is generated by setting the STRG bit of INTSFTTRG register in AOS.

10.4 Functions

10.4.1 Non-maskable interrupt

The non-maskable interrupt (NMI) sources are as follows:

- XTAL oscillation stop detection interrupt
- WDT underflow/refresh error interrupt
- SWDT underflow/refresh error interrupt
- Low Voltage Detection 1(PVD1) interrupt
- Low Voltage Detection 2(PVD2) interrupt
- SRAM parity error interrupt
- SRAM ecc error interrupt
- MPU bus error interrupt

Non-maskable interrupt has the highest priority. Since the no-maskable interrupt can select multiple interrupt events, the status of each interrupt event can be determined by querying NMI flag register (INTC_NMIFR). Make sure that all flags are “0” before exiting from non-maskable interrupt processing.

Follow the procedure to use NMI:

1. Configure the corresponding function.
2. Write “1” to all bits of INTC_NMICFR register to clear NMI flags.
3. Write “1” to INTC_NMIENR selection register to enable NMI.

Note:

- Once the bit of INTC_NMIENR is set to “1”, it cannot be changed unless with RESET.

10.4.2 External interrupt

Follow the procedure to use EIRQ:

1. Write “0” to the EFEN bit of INTC_EIRQCRm ($m=0\sim15$) register to disable the digital filter.
2. Set the EIRQTRG[1:0] bits of INTC_EIRQCRm register to select trigger edge or level; Set the EISMPCLK[1:0] bits to select digital filter sampling clock.
3. Write “1” to the EFEN bit to enable the digital filter.

10.4.3 Interrupt source selection

The INTC uses a total of 144 interrupt vectors of NVIC, provides three interrupt source selection methods, For information of interrupt vector and event numbers, refer to Table 10-2 Interrupt vector table and Table 10-3 Event table.

Method 1:

Method 1 applies to interrupt vector 0~31 of NVIC. Each interrupt vector has an interrupt selection register (INTC_SEL0~31) to select one out of all interrupt events as the interrupt source. Interrupt vector 0~31 need to be enabled by INTC_IER register.

Method 2:

Method 2 applies to interrupt vector 32~127 of NVIC. Each interrupt vector has an interrupt selection register (INTC_SEL32~127) to select one out of specified 32 interrupt events as the interrupt source. The 96 interrupt vectors are divided into 16 groups with 6 vectors as 1 group, each vector group has the same 32 interrupt events.

Method 3:

Method 3 applies to interrupt vector 128~143 of NVIC. Each interrupt vector has a vector shared interrupt selection register (INTC_VSSEL128~143) to select up to 32 interrupt events as the interrupt source. Every 32 peripheral interrupt event requests can share an interrupt vector. By selecting interrupt event with a peripheral flag, each peripheral can apply for an interrupt vector.

10.4.4 Software interrupt

The software interrupt is generated by writing the software interrupt control register (INTC_SWIER). A total of 32 software interrupt event requests are provided, and use the interrupt vector 0~31, which need to be enabled by INTC_IER register. For details, refer to Table 10-2 Interrupt vector table.

10.4.5 Interrupt/Event Selection

The interrupt events selected by INTC_SEL0~31 registers share the interrupt vector 0~31 of NVIC with the software interrupts. Furthermore, these interrupt event requests can also be used as events to wake up the core (WFE). Different with interrupts, events are enabled by the event enable register (INTC_EVTER).

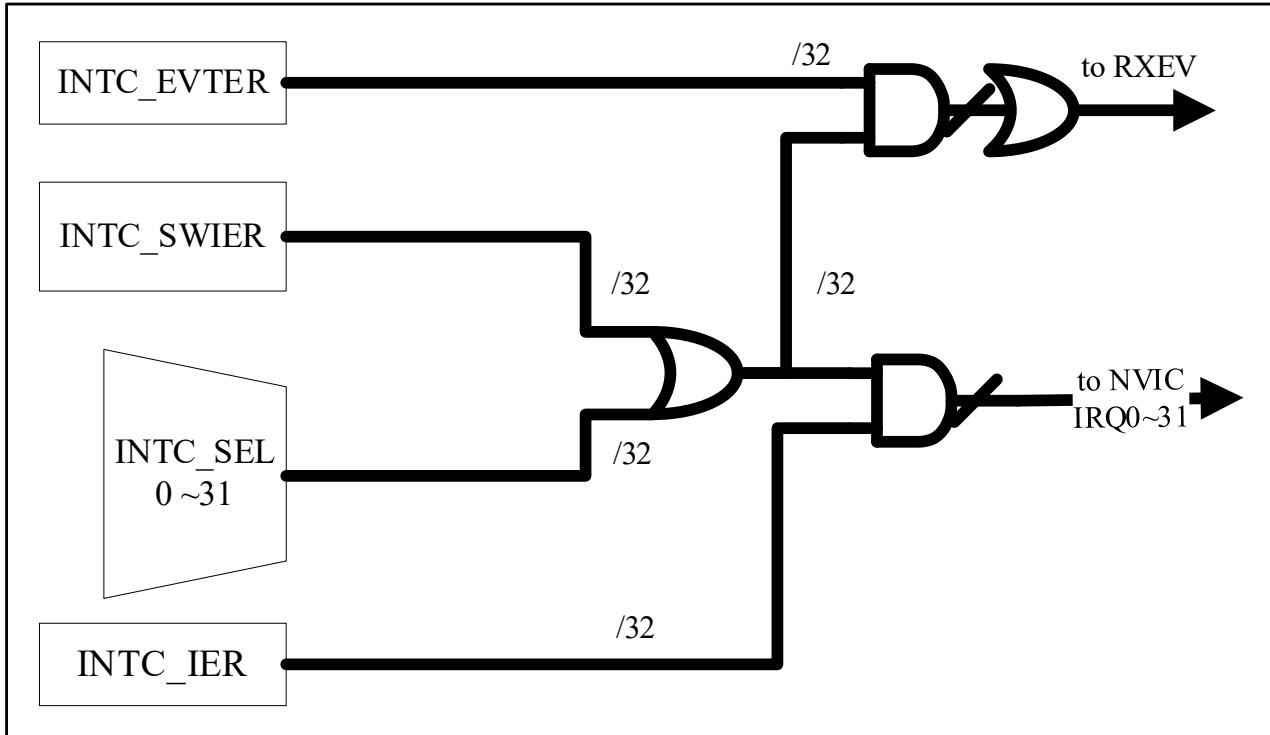


Figure 10-2 Interrupt event selection

10.4.6 WFE Wake-up Event Management

When the core enters sleep mode by executing WFE instruction, it can be woken up by both interrupts and events.

- Woken up by interrupt. Enable a peripheral interrupt and set INTC_SEL to select the interrupt. Enable INT_IER according to the selected interrupt vector, but do not enable it in the NVIC. Disable INTC_EVTER, enable the SEVONPEND bit of Cortex™-M4F system control register(SCR). Then executing WFE instruction to enter sleep mode. When the selected interrupt event occurs, the core is woken up from sleep mode but does not enter the interrupt handler. After return, the interrupt flag in peripheral and NVIC should be cleared.
- Woken up by event. Enable a peripheral interrupt and set INTC_SEL to select the interrupt and enable INTC_EVTER according to the selected interrupt vector. Disable INTC_IER and NVIC, enable the SEVONPEND bit of Cortex™-M4F system control register(SCR). Then executing WFE instruction to enter sleep mode. When the selected interrupt event occurs, the core is woken up from sleep mode.

10.4.7 Noise filter

In order to eliminate the input noise, the external interrupt EIRQ is configured with digital and analog filters.

The digital filter is enabled by the EFEN bit of INTC_EIRQCR register. PCLK3 is used as the sampling clock to sample EIRQ input signals, and signals which width is less than 3 sampling cycles will be filtered out. The sampling period is set by the EISMPCLK[1:0] bits of INTC_EIRQCR register. The analog filter is enabled by the NOCEN bit of INTC_EIRQCR register. If enabled, the signal is filtered with the filtering width selected by the NOCSEL bit of INTC_EIRQCR register. The two filters can be enabled or disabled independently. When enabled at the same time, the signal goes through the analog filter before the digital filter.

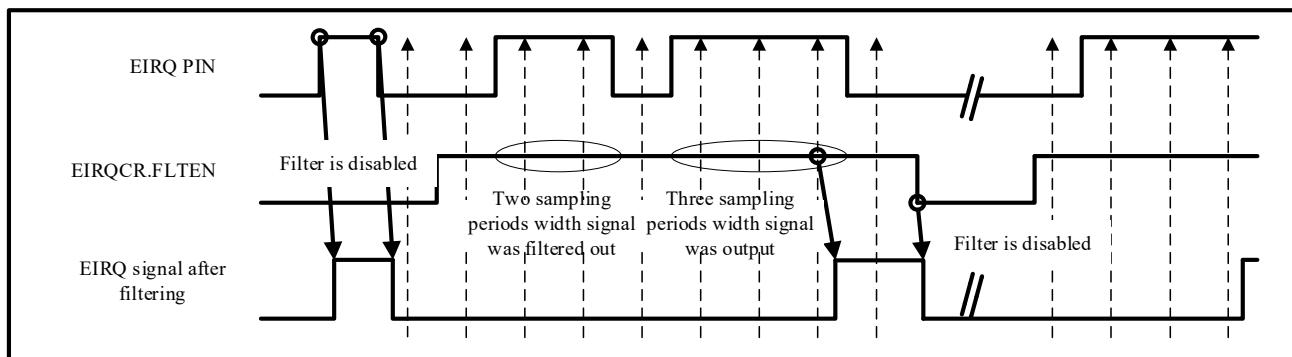


Figure 10-3 Working diagram of digital filter

Before entering stop mode, the digital filter must be disabled, and it can be enabled again after returning from stop mode. The analog filter can work in stop mode.

10.4.8 Low power mode return

10.4.8.1 Sleep mode return

To return from sleep mode with interrupt, the following settings are required:

- Select an interrupt event
 - Enable INTC_IER register if interrupt vector 0~31 is used.
- Enable the interrupt in NVIC
- Enable INTC_NMIER register when using non-maskable interrupts

10.4.8.2 Stop mode return

To return from stop mode with interrupt, the following settings are required:

- Select an interrupt event (must be stop wake-up event)
 - Enable INTC_IER register if interrupt vector 0~31 is used.
- Select the interrupt event for stop mode return
 - As non-maskable interrupts, by setting INTC_NMIENR register, SWDT, PVD1, PVD2 can wake up stop mode.
 - For maskable interrupts, set INTC_WUPEN register.
- Enable the interrupt in NVIC

10.4.8.3 Power down mode return

The system can return from power-down mode under conditions described in the Power Control (PWC) chapter, including RES# pin reset, power-on reset, and voltage detect 0 reset. After returning, the CPU enters the reset interrupt processing. Refer to the Power Control (PWC) chapter for details.

10.4.8.4 Non-maskable interrupts and WFI instruction

Before executing the WFI instruction, make sure that all bits of the non-maskable interrupt flag register (INTC_NMIFR) are "0".

10.5 Registers

The following table is an INTC register list.

INTC base address: 0x40051000

Table 10-4 INTC Register List

Register name	Symbol	Offset address	Bit width	Reset value
External Interrupt Filter Control Register	INTC_NOCCR	0x0000	32	0x00000000
Non-Maskable Interrupt Enable Register	INTC_NMENR	0x0004	32	0x00000000
Non-Maskable Interrupt Flag Register	INTC_NMIFR	0x0008	32	0x00000000
Non-Maskable Interrupt Flag Clear Register	INTC_NMICFR	0x000C	32	0x00000000
External Interrupt Control Register x (x=0~15)	INTC_EIRQCRx (x=0~15)	0x0010+0x4*x	32	0x00000000
Stop Mode Wake-Up Enable Register	INTC_WUPEN	0x0050	32	0x00000000
External Interrupt Flag Register	INTC_EIFR	0x0054	32	0x00000000
External Interrupt Flag Clear Register	INTC{EIFCR}	0x0058	32	0x00000000
Interrupt Selection Register x (x=0~127)	INTC_SELx (x=0~127)	0x005C+0x4*x	32	0x000001FF
Vector Shared Interrupt Selection Register x (x=128~143)	INTC_VSSELx (x=128~143)	0x025C+0x4*x	32	0x00000000
Software Interrupt Register	INTC_SWIER	0x029C	32	0x00000000
Event Enable Register	INTC_EVTER	0x02A0	32	0x00000000
Interrupt Enable Register	INTC_IER	0x02A4	32	0xFFFFFFFF

10.5.1 External Interrupt Filter Control Register (INTC_NOCCR)

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															
<hr/>															
Bit	Symbol	Bit name			Description								Read and Write		
b31~b14	Reserved	-			Read as "0", write as "0"								R		
b13~b12	NOCSEL[1:0]	Analog filter width selection			00: Filter width option 1 01: Filter width option 2 10: Filter width option 3 11: Filter width option 4 For the value of each option, refer to the EIRQ filter characteristics chapter in the electrical characteristics of the data sheet. This filter width selection is valid for all EIRQ interrupts.								R/W		
b11~b0	Reserved	-			Read as "0", write as "0"								R		

10.5.2 Non-Maskable Interrupt Enable Register (INTC_NMIENR)

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															
<hr/>															
Bit	Symbol	Bit name			Description								Read and Write		
b31~b12	Reserved	-			Read as "0", write as "0"								R		
b11	WDTENR	WDT underflow/refresh error interrupt selection			0: Deselect the interrupt as non-maskable interrupt source 1: Select the interrupt as non-maskable interrupt source								R/W		
b10	BUSMENR	MPU bus error interrupt selection			0: Deselect the interrupt as non-maskable interrupt source 1: Select the interrupt as non-maskable interrupt source								R/W		
b9	RECCENR	SRAM ECC error interrupt selection			0: Deselect the interrupt as non-maskable interrupt source 1: Select the interrupt as non-maskable interrupt source								R/W		
b8	REPENR	SRAM parity error interrupt selection			0: Deselect the interrupt as non-maskable interrupt source 1: Select the interrupt as non-maskable interrupt source								R/W		
b7~b6	Reserved	-			Read as "0", write as "0"								R		
b5	XTALSTPENR	XTAL oscillation stop detection interrupt selection			0: Deselect the interrupt as non-maskable interrupt source 1: Select the interrupt as non-maskable interrupt source								R/W		
b4	Reserved	-			Read as "0", write as "0"								R		
b3	PVD2ENR	Low Voltage Detection PVD2 Interrupt Selection			0: Deselect the interrupt as non-maskable interrupt source 1: Select the interrupt as non-maskable interrupt source								R/W		
b2	PVD1ENR	Low Voltage Detection PVD1 Interrupt Selection			0: Deselect the interrupt as non-maskable interrupt source 1: Select the interrupt as non-maskable interrupt source								R/W		
b1	SWDTENR	SWDT underflow/refresh error interrupt selection			0: Deselect the interrupt as non-maskable interrupt source 1: Select the interrupt as non-maskable interrupt source								R/W		
b0	Reserved	-			Read as "0", write as "0"								R		

10.5.3 Non-Maskable interrupt flag register (INTC_NMIFR)

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		WDT FR	BUS MFR	REC CFR	REP FR	Reserved		XTAL STPF R	Rese rved	PVD 2FR	PVD 1FR	SWD TFR	Rese rved						
Bit	Symbol	Bit name		Description								Read and Write							
b31~b12	Reserved	-		Read as "0", write as "0"								R							
b11	WDTFR	WDT underflow/refresh error interrupt flag		0: No WDT underflow/refresh error interrupt occurred 1: WDT underflow/refresh error interrupt occurred								R							
b10	BUSMFR	MPU bus error interrupt flag		0: No MPU bus error interrupt occurred 1: MPU bus error interrupt occurred								R							
b9	RECCFR	SRAM ECC error interrupt flag		0: No SRAM ECC error interrupt occurred 1: SRAM ECC error occurred								R							
b8	REPFR	SRAM parity error interrupt flag		0: No SRAM parity error interrupt occurred 1: SRAM parity error interrupt occurred								R							
b7~b6	Reserved	-		Read as "0", write as "0"								R							
b5	XTALSTPFR	XTAL oscillation stop detection interrupt flag		0: No XTAL oscillation stop detection interrupt occurred 1: XTAL oscillation stop detection interrupt occurred								R							
b4	Reserved	-		Read as "0", write as "0"								R							
b3	PVD2FR	Low voltage detection PVD2 interrupt flag		0: No low voltage detection PVD2 interrupt occurred 1: Low voltage detection PVD2 interrupt occurred								R							
b2	PVD1FR	Low voltage detection PVD1 interrupt flag		0: No low voltage detection PVD1 interrupt occurred 1: Low voltage detection PVD1 occurred								R							
b1	SWDTFR	SWDT underflow/refresh error interrupt flag		0: No SWDT underflow/refresh error interrupt occurred 1: SWDT underflow/refresh error interrupt occurred								R							
b0	Reserved	-		Read as "0", write as "0"								R							

10.5.4 Non-Maskable Interrupt Flag Clear Register (INTC_NMICFR)

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		WDT CFR	BUS MCFR	REC CCFR	REPC FR	Reserved		XTAL STPC FR	Res	PVD 2CFR	PVD 1CFR	SWD TCFR	Res						
<hr/>																			
Bit	Symbol	Bit name		Description								Read and Write							
b31~b12	Reserved	-		Read as "0", write as "0"								R							
b11	WDTCFR	WDT underflow/refresh error interrupt flag clear		0: Invalid 1: Clear the WDT underflow/refresh error interrupt flag								W							
b10	BUSMCFR	MPU bus error interrupt flag clear		0: Invalid 1: Clear the MPU bus error interrupt flag								W							
b9	RECCCFR	SRAM ECC error interrupt flag clear		0: Invalid 1: Clear the SRAM ECC error interrupt flag								W							
b8	REPCFR	SRAM Parity Error Interrupt Flag Clear		0: Invalid 1: Clear the SRAM parity error interrupt flag								W							
b7~b6	Reserved	-		Read as "0", write as "0"								R							
b5	XTALSTPCFR	XTAL oscillation stop detection interrupt flag clear		0: Invalid 1: Clear the XTAL oscillation stop detection interrupt flag								W							
b4	Reserved	-		Read as "0", write as "0"								R							
b3	PVD2CFR	Low voltage detection PVD2 interrupt flag Clear		0: Invalid 1: Clear the low voltage detection PVD2 interrupt flag								W							
b2	PVD1CFR	Low voltage detection PVD1 interrupt flag Clear		0: Invalid 1: Clear the low voltage detection PVD1 interrupt flag								W							
b1	SWDTCFR	SWDT underflow/refresh error interrupt flag clear		0: Invalid 1: Clear the SWDT underflow/refresh error interrupt flag								W							
b0	Reserved	-		Read as "0", write as "0"								R							

10.5.5 External Interrupt Control Register (INTC_EIRQCRx, x=0~15)

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
NOC EN	Reserved					EFEN	Res	EISMPCLK [1:0]	Reserved			EIRQTRG[1:0]			

Bit	Symbol	Bit name	Description	Read and Write
b31~b16	Reserved	-	Read as "0", write as "0"	R
b15	NOCEN	Analog filter enable bit	0: Disable analog filter 1: Enable analog filter	R/W
b14~b8	Reserved	-	Read as "0", write as "0"	R
B7	EFEN	Digital filter enable bit	0: Disable digital filter 1: Enable digital filter	R/W
b6	Reserved	-	Read as "0", write as "0"	R
b5~b4	EISMPCLK[1:0]	Digital filter sampling clock selection	00: PCLK3 01: PCLK3/8 10: PCLK3/32 11: PCLK3/64	R/W
b3~b2	Reserved	-	Read as "0", write as "0"	R
b1~b0	EIRQTRG[1:0]	Trigger selection	00: Falling edge 01: Rising edge 10: Falling edge and rising edge 11: Low level	R/W

10.5.6 External Interrupt Flag Register (INTC_EIFR)

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
EIFR 15	EIFR 14	EIFR 13	EIFR 12	EIFR 11	EIFR 10	EIFR 9	EIFR 8	EIFR 7	EIFR 6	EIFR 5	EIFR 4	EIFR 3	EIFR 2	EIFR 1	EIFR 0
Bit	Symbol	Bit name		Description								Read and Write			
b31~b16	Reserved	-		Read as "0", write as "0"								R			
b15~b0	EIFR	EIFR flag		0: No EIRQ occurred, or was cleared. 1: EIRQ occurred								R			

10.5.7 External Interrupt Flag Clear Register (INTC_EIFCR)

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
EIFCR[15:0]															
Bit	Symbol	Bit name		Description								Read and Write			
b31~b16	Reserved	-		Read as "0", write as "0"								R			
b15~b0	EIFCR	EIFR clear bit		0: Writing "0" has no effect 1: Write "1" to clear the EIFR register								W			

10.5.8 Interrupt Selection Register (INTC_SELx, x=0~127)

INTC_SEL0~31

Reset value: 0x000001FF

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										INTSEL[8:0]					
<hr/>															
Bit	Symbol	Bit name		Description								Read and Write			
b31~b9	Reserved	-		Read as "0", write as "0"								R			
b8~b0	INTSEL[8:0]	Interrupt Event Selection		9'h000~9'h1FE: Select the interrupt event listed in section 10.3.2 .								R/W			

INTC_SEL32~127

Reset value: 0x000001FF

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										INTSEL[8:0]					
<hr/>															
Bit	Symbol	Bit name		Description								Read and Write			
b31~b9	Reserved	-		Read as "0", write as "0"								R			
b8~b0	INTSEL[8:0]	Interrupt Event Selection		INTC_SEL32~INTC_SEL37: Select the interrupt event from 9'h000 to 9'h01F listed in section 10.3.2 , other selections are invalid. INTC_SEL38~INTC_SEL43: Select the interrupt event from 9'h020 to 9'h03F listed in section 10.3.2 , other options are invalid. INTC_SEL44~INTC_SEL49: Select the interrupt event from 9'h040 to 9'h05F listed in section 10.3.2 , other options are invalid. INTC_SEL50~INTC_SEL55: Select the interrupt event from 9'h060 to 9'h07F listed in section 10.3.2 , other options are invalid. INTC_SEL56~INTC_SEL61: Select the interrupt event from 9'h080 to 9'h09F listed in section 10.3.2 , other options are invalid. INTC_SEL62~INTC_SEL67: Select the interrupt event from 9'h0A0 to 9'h0BF listed in section 10.3.2 , other selections are invalid. INTC_SEL68~INTC_SEL73: Select the interrupt event from 9'h0C0 to 9'h0DF listed in section 10.3.2 , other options are invalid. INTC_SEL74~INTC_SEL79: Select the interrupt event from 9'h0E0 to 9'h0FF listed in section 10.3.2 , other selections are invalid. INTC_SEL80~INTC_SEL85: Select the interrupt event from 9'h100 to 9'h11F listed in section 10.3.2 , other selections are invalid. INTC_SEL86~INTC_SEL91: Select the interrupt event from 9'h120 to 9'h13F listed in section 10.3.2 , other options are invalid. INTC_SEL92~INTC_SEL97: Select the interrupt event from 9'h140 to 9'h15F listed in section 10.3.2 , other options are invalid. INTC_SEL98~INTC_SEL103:								R/W			

Bit	Symbol	Bit name	Description	Read and Write
			Select the interrupt event from 9'h160 to 9'h17F listed in section 10.3.2 , other selections are invalid. INTC_SEL104~INTC_SEL109: Select the interrupt event from 9'h180 to 9'h19F listed in section 10.3.2 , other selections are invalid.	
			INTC_SEL110~INTC_SEL115: Select the interrupt event from 9'h1A0 to 9'h1BF listed in section 10.3.2 , other selections are invalid.	
			INTC_SEL116~INTC_SEL121: Select the interrupt event from 9'h1C0 to 9'h1DF listed in section 10.3.2 , other selections are invalid.	
			INTC_SEL122~INTC_SEL127: Select the interrupt event from 9'h1E0 to 9'h1FF listed in section 10.3.2 , other selections are invalid.	

10.5.9 Vector Shared Interrupt Selection Register (INTC_VSELx, x=128~143)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
VSEL 31	VSEL 30	VSEL 29	VSEL 28	VSEL 27	VSEL 26	VSEL 25	VSEL 24	VSEL 23	VSEL 22	VSEL 21	VSEL 20	VSEL 19	VSEL 18	VSEL 17	VSEL 16
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
VSEL 15	VSEL 14	VSEL 13	VSEL 12	VSEL 11	VSEL 10	VSEL 9	VSEL 8	VSEL 7	VSEL 6	VSEL 5	VSEL 4	VSEL 3	VSEL 2	VSEL 1	VSEL 0

Bit	Symbol	Bit name	Description	Read and Write
b31~b0	VSEL[31:0]	Interrupt event enable	INTC_VSEL128: Each bit enables the interrupt event from 9'h000 to 9'h01F listed in section 10.3.2 . INTC_VSEL129: Each bit enables the interrupt event from 9'h020 to 9'h03F listed in section 10.3.2 . INTC_VSEL130: Each bit enables the interrupt event from 9'h040 to 9'h05F listed in section 10.3.2 . INTC_VSEL131: Each bit enables the interrupt event from 9'h060 to 9'h07F listed in section 10.3.2 . INTC_VSEL132: Each bit enables the interrupt event from 9'h080 to 9'h09F listed in section 10.3.2 . INTC_VSEL133: Each bit enables the interrupt event from 9'h0A0 to 9'h0BF listed in section 10.3.2 . INTC_VSEL134: Each bit enables the interrupt event from 9'h0C0 to 9'h0DF listed in section 10.3.2 . INTC_VSEL135: Each bit enables the interrupt event from 9'h0E0 to 9'h0FF listed in section 10.3.2 . INTC_VSEL136: Each bit enables the interrupt event from 9'h100 to 9'h11F listed in section 10.3.2 . INTC_VSEL137: Each bit enables the interrupt event from 9'h120 to 9'h13F listed in section 10.3.2 . INTC_VSEL138: Each bit enables the interrupt event from 9'h140 to 9'h15F listed in section 10.3.2 . INTC_VSEL139: Each bit enables the interrupt event from 9'h160 to 9'h17F listed in section 10.3.2 . INTC_VSEL140: Each bit enables the interrupt event from 9'h180 to 9'h19F listed in section 10.3.2 . INTC_VSEL141: Each bit enables the interrupt event from 9'h1A0 to 9'h1BF listed in section 10.3.2 . INTC_VSEL142: Each bit enables the interrupt event from 9'h1C0 to 9'h1DF listed in section 10.3.2 . INTC_VSEL143: Each bit enables the interrupt event from 9'h1E0 to 9'h1FF listed in section 10.3.2 .	R/W

10.5.10 Stop Mode Wake-up Enable Register (INTC_WUPEN)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved	ETH_WUE_N	USF_WUE_N	USH_WUE_N	RX_WUE_N	TMR20VF_WUE_N	TMR2GC_MWUEN	TMR0GC_MWUEN	RTC_PRD_WUE_N	RTC_ALM_WUE_N	WKT_M_WUE_N	CMP_WUE_N	PVD2WUEN	PVD1WUEN	SWDT_WUE_N	
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
EIRQWUEN[15:0]															

Bit	Symbol	Bit name	Description	Read and Write
b31	Reserved	-	Read as "0", write as "0"	R
b30	Reserved	-	Read as "0", write as "0"	R
b29	ETHWUEN	ETH_PMT stop mode wake-up enable	0: Wake-up disable 1: Wake-up enable	R/W
b28	USFWUEN	USBFS_WKUP stop mode wake-up enable	0: Wake-up disable 1: Wake-up enable	R/W
b27	USHWUEN	USBHS_WKUP stop mode wake-up enable	0: Wake-up disable 1: Wake-up enable	R/W
b26	RXWUEN	USART1_WUPI stop mode wake-up enable	0: Wake-up disable 1: Wake-up enable	R/W
b25	TMR20VFNUEN	TMR21_OVFA stop mode wake-up enable	0: Wake-up disable 1: Wake-up enable	R/W
b24	TMR2GCMWUEN	TMR21_GCMA stop mode wake-up enable	0: Wake-up disable 1: Wake-up enable	R/W
b23	TMR0GCMWUEN	TMR01_GCMA stop mode wake-up enable	0: Wake-up disable 1: Wake-up enable	R/W
b22	RTCPRDWUEN	RTC_PRD stop mode wake-up enable	0: Wake-up disable 1: Wake-up enable	R/W
b21	RTCALMWUEN	RTC_ALM stop mode wake-up enable	0: Wake-up disable 1: Wake-up enable	R/W
b20	WKTMWUEN	WKT_M_WUE_N	0: Wake-up disable 1: Wake-up enable	R/W
b19	CMPWUEN	CMP stop mode wake-up enable	0: Wake-up disable 1: Wake-up enable	R/W
b18	PVD2WUEN	PVD_PVD2 stop mode wake-up enable	0: Wake-up disable 1: Wake-up enable	R/W
b17	PVD1WUEN	PVD_PVD1 stop mode wake-up enable	0: Wake-up disable 1: Wake-up enable	R/W
b16	SWDTWUEN	SWDT_REFUDF stop mode wake-up enable	0: Wake-up disable 1: Wake-up enable	R/W
b15~b0	EIRQWUEN[15:0]	EIRQ stop mode wake-up enable	0: Wake-up disable 1: Wake-up enable	R/W

10.5.11 Software Interrupt Register (INTC_SWIER)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
SWIE 31	SWIE 30	SWIE 29	SWIE 28	SWIE 27	SWIE 26	SWIE 25	SWIE 24	SWIE 23	SWIE 22	SWIE 21	SWIE 20	SWIE 19	SWIE 18	SWIE 17	SWIE 16
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
SWIE 15	SWIE 14	SWIE 13	SWIE 12	SWIE 11	SWIE 10	SWIE 9	SWIE 8	SWIE 7	SWIE 6	SWIE 5	SWIE 4	SWIE 3	SWIE 2	SWIE 1	SWIE 0

Bit	Symbol	Bit name	Description	Read and Write
b31~b0	SWIE	Software interrupt	0: Invalid 1: A software interrupt occurs Note: A software interrupt occurs after writing a '1'. If necessary to occur again, write a "0" to clear it first.	R/W

10.5.12 Event Enable Register (INTC_EVTER)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EVTE 31	EVTE 30	EVTE 29	EVTE 28	EVTE 27	EVTE 26	EVTE 25	EVTE 24	EVTE 23	EVTE 22	EVTE 21	EVTE 20	EVTE 19	EVTE 18	EVTE 17	EVTE 16
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
EVTE 15	EVTE 14	EVTE 13	EVTE 12	EVTE 11	EVTE 10	EVTE 9	EVTE 8	EVTE 7	EVTE 6	EVTE 5	EVTE 4	EVTE 3	EVTE 2	EVTE 1	EVTE 0

Bit	Symbol	Bit name	Description	Read and Write
b31~b0	EVTE	Event enable bit	0: Event disable 1: Event enable	R/W

10.5.13 Interrupt Enable Register (INTC_IER)

Reset value: 0xFFFFFFFF

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

Bit	Symbol	Bit name	Description	Read and Write
b31~b0	IER	Interrupt Enable Register Bits	Register bits 0~31 correspond to NVIC interrupt vectors 0~31 respectively. When disabled, the interrupt event selected by the selection register INTC_INTSEL0~31 will not be linked to NVIC. 0: Interrupt disable 1: Interrupt enable	R/W

10.6 Precautions

For the description of ARM core interrupts, please refer to the ARM manual ARM Processor Cortex®-M4 Technical Reference Manual (ARM DDI 0439D).

11 Automatic Operation System (AOS)

11.1 Introduction

The Automatic Operation System is used to realize the linkage between the peripheral hardware circuits without the aid of the CPU. Use the events generated by the peripheral circuit as the AOS source, such as the comparison match and count overflow of the timer, the periodic signal of the RTC, various states of the communication module sending and receiving data (idle, receiving data full, sending data over, sending data empty), ADC conversion ends, etc, to trigger other peripheral circuit actions. The triggered peripheral circuit action is called AOS target.

11.1.1 Function Overview

- There are a total of 368 AOS sources. Except for special restrictions, each AOS target can choose one of them as the trigger source. In addition, two additional trigger sources can be selected through the common trigger source selection register 1 and 2. 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 peripheral circuit hardware or by software (by writing registers).
- Peripheral circuits that can be AOS targets are as follows:
 - 4 DCU trigger targets, of which DCU1 and DCU5 share an AOS target, DCU2 and DCU6 share an AOS target, DCU3 and DCU7 share an AOS target, and DCU4 and DCU8 share an AOS target
 - 17 DMA trigger targets for two 8-channel DMA data transmission and one DMA event trigger channel re-configuration
 - 4 advanced control timers (Timer6) trigger target
 - 1 general-purpose timer 0 (Timer0) trigger target
 - 1 general-purpose timer 2 (Timer0) trigger target
 - 2 Event Port trigger targets, of which Event Port group1 and Event Port group2 share an aos target, Event Port group3 and Event Port group4 share an aos target
 - 2 HASH trigger targets
 - 4 general purpose timer A trigger target
 - 1 on-chip temperature sensor (OTS) trigger target
 - 3 groups of 2 AD trigger targets for AD1~AD3 sequence triggering

11.1.2 Module Diagram

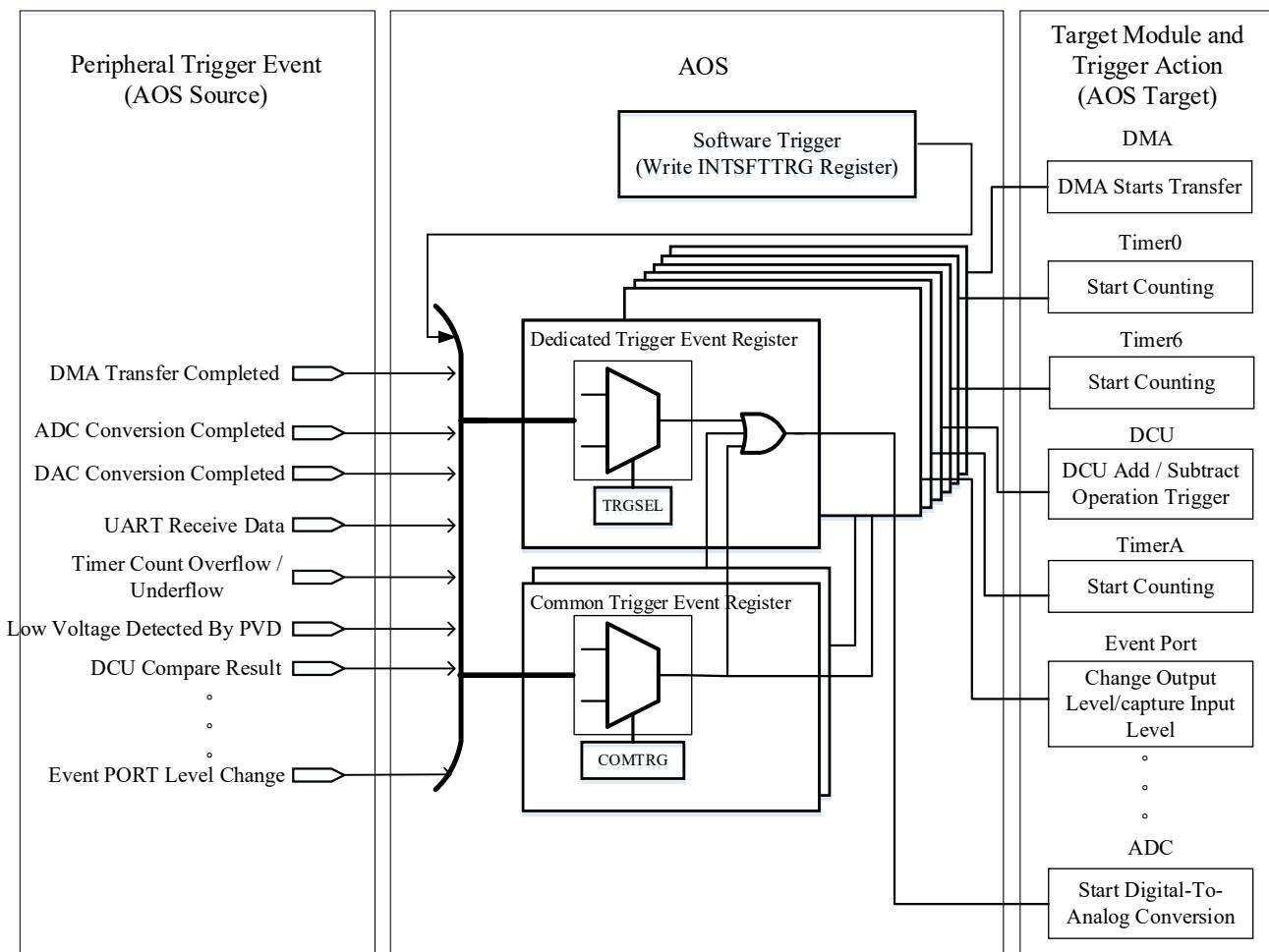


Figure 11-1 Schematic Diagram of AOS Module

11.2 Functional Description

11.2.1 AOS Source Event List

For the number of AOS source events, see the Interrupt Event Request Number Table 10-3 in the chapter [Interrupt controller (INTC)]. The events marked with "√" in the column "Can be selected as events" in the table can be used as AOS sources.

11.2.2 AOS Target List

Table 11-1 AOS Target List

Modules	Action
DCU1	Trigger addition/subtraction operation, trigger data increment or decrement in triangle wave/sawtooth wave mode
DCU2	Trigger addition/subtraction operation, trigger data increment or decrement in triangle wave/sawtooth wave mode
DCU3	Trigger addition/subtraction operation, trigger data increment or decrement in triangle wave/sawtooth wave mode
DCU4	Trigger addition/subtraction operation, trigger data increment or decrement in triangle wave/sawtooth wave mode
DCU5	Trigger addition/subtraction operations
DCU6	Trigger addition/subtraction operations
DCU7	Trigger addition/subtraction operations
DCU8	Trigger addition/subtraction operations
DMA1	Channel 0 starts transmission Channel 1 starts transmission Channel 2 starts transmission Channel 3 starts transmission Channel 4 starts transmission Channel 5 starts transmission Channel 6 starts transmission Channel 7 starts transmission
DMA2	Channel 0 starts transmission Channel 1 starts transmission Channel 2 starts transmission Channel 3 starts transmission Channel 4 starts transmission Channel 5 starts transmission Channel 6 starts transmission Channel 7 starts transmission
DMA1&2	Event-triggered channel re-configure
Timer6	Start counting
Timer0	Start counting
Timer2	Start counting
HASH	Start HASH operation
Event Port	Event Port1&2 trigger action Event Port3&4 trigger action
TimerA	Start counting
OTS	Start measuring temperature
ADC1	Start analog-to-digital conversion
ADC2	Start analog-to-digital conversion

Modules	Action
ADC3	Start analog-to-digital conversion

11.3 Action Description

11.3.1 Dedicated Trigger Source

The peripheral circuit module with AOS target is equipped with a dedicated peripheral trigger source selection register for each AOS target. When this register is written to the event number corresponding to the AOS source, the AOS target selects this AOS source as the trigger source. When the event of the AOS source occurs, the event will be transmitted to the AOS target through the AOS, and the peripheral circuit as the AOS target starts to act according to its own settings.

11.3.2 Common Trigger Source

In addition to the dedicated peripheral trigger source selection registers for each AOS target, AOS also configures two common trigger source selection registers (AOS_COMTRG1, AOS_COMTRG2) to implement multiple AOS sources triggering the same AOS target. When using, firstly set the enable bit of the common trigger source to be valid 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 transmitted to the AOS target through the common trigger source of the AOS, and the peripheral circuit as the AOS target starts to act according to its own settings. When a dedicated trigger source and a common trigger source are set at the same time, a maximum of 3 AOS sources can trigger the same AOS target at the same time, and when a trigger event occurs in any one of the 3 AOS sources, the AOS target will be triggered.

All AOS targets share these two common trigger sources. Therefore, when other AOS targets do not use the event selected by the common trigger source selection register, it is necessary to clear the common trigger source enable bit in its dedicated peripheral trigger source selection register to prevent trigger actions.

11.4 Register Description

11.4.1 Register Overview

Register base address: 0x40010800

Table 11-2 Register List

Abbreviation	Name	Offset address	The chapter
INTSFTTRG	Peripheral Trigger Event Register	0x00	11.4.2
DCU_TRGSELx(x=1~4)	DCU Trigger Source Selection Register	0x04, 0x08, 0x0C, 0x10	44.3.9
DMA1_TRGSELx(x=0~7)	DMA1 Transfer Start Trigger Source Selection Register	0x14, 0x18, 0x1C, 0x20 0x24, 0x28, 0x2C, 0x30	15.5.14
DMA2_TRGSELx(x=0~7)	DMA2 Transfer Start Trigger Source Select Register	0x34, 0x38, 0x3C, 0x40 0x44, 0x48, 0x4C, 0x50	15.5.14
DMA_TRGSELRC	DMA Channel Re-configure Trigger Source Selection Register	0x54	15.5.15
TMR6_HTSSRx(x=0~3)	Timer6 Trigger Source Selection Register	0x58, 0x5C, 0x60, 0x64	20.5.25
PEVNTTRGSR12	Event Port1/2 Trigger Source Selection Register	0x68	9.4.13
PEVNTTRGSR34	Event Port3/4 Trigger Source Selection Register	0x6C	9.4.13
TMR0_HTSSR	Timer0 Trigger Source Selection Register	0x70	26.6.4
TMR2_HTSSR	Timer2 Trigger Source Selection Register	0x74	0000h
HASH_ITRGSELA	HASH Trigger Source Selection Register A	0x7C	42.3.7.4
HASH_ITRGSELB	HASH Trigger Source Selection Register B	0x78	42.3.7.5
TMRA_HTSSRx(x=0~3)	TimerA Trigger Source Selection Register	0x80, 0x84, 0x88, 0x8C	24.5.15
OTS_TRG	OTS Trigger Source Selection Register	0x90	19.3.6
ADC1_ITRGSELRx(x=0,1)	ADC1 Conversion Start Trigger Source Selection Register	0x94, 0x98	17.4.6
ADC2_ITRGSELRx(x=0,1)	ADC2 Conversion Start Trigger Source Selection Register	0x9C, 0xA0	17.4.6
ADC3_ITRGSELRx(x=0,1)	ADC3 Conversion Start Trigger Source Selection Register	0xA4, 0xA8	17.4.6
AOS_COMTRG1	Common Trigger Source Selection Register 1	0xAC	11.4.18
AOS_COMTRG2	Common Trigger Source Selection Register 2	0xB0	11.4.19

11.4.2 Peripheral Trigger Event Register (INTSFTTRG)

Register description: Writing to this register will generate a trigger event.

Offset address: 0x00

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															
SFTG															
Bit	Symbol	Bit name	Description										Read and write		
b31~b1	Reserved	-	Read as "0", write as "0"										R/W		
b0	SFTG	Software triggering	0: No software trigger event is generated 1: Generate a software trigger event Set this bit to 1 to generate a peripheral trigger event, software writing 0 is invalid										W		

11.4.3 DCU Trigger Source Selection Register (DCU_TRGSELx) (x=1~4)

Register description: After the DCU selects the hardware-triggered start mode, the number of the event to be triggered is written into this register. When the peripheral circuit event corresponding to the number occurs, the DCU will be triggered by the event to start and perform operations. DCU1 and DCU5 share the register DCU_TRGSEL1, that is, when the event number is written to the register DCU_TRGSEL1 and the event corresponding to the number occurs, both DCU1 and DCU5 will be triggered at the same time; DCU2 and DCU6 share the register DCU_TRGSEL2, that is, when the event number is written to the register DCU_TRGSEL2 and the event corresponding to the number occurs, DCU2 and DCU6 will be triggered at the same time; DCU3 and DCU7 share the register DCU_TRGSEL3, that is, when the event number is written to the register DCU_TRGSEL3 and the event corresponding to the number occurs, DCU3 and DCU7 will be triggered at the same time; DCU4 and DCU8 share the register DCU_TRGSEL4, that is, when the event number is written to the register DCU_TRGSEL4 and the event corresponding to the number occurs, DCU4 and DCU8 will be triggered at the same time.

Offset address: 0x04, 0x08, 0x0C, 0x10

Reset value: 0x0000001FF

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]							
<hr/>															
Bit	Symbol	Bit name	Description										Read and write		
b31	COMEN[1]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG2 to trigger this AOS target										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable the public trigger event of AOS_COMTRG1 to trigger this AOS target 1: Enable the public trigger event of AOS_COMTRG1 to trigger this AOS target										R/W		
b29~b9	Reserved	-	Read as "0", write as "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected										R/W		

11.4.4 DMA1 Transfer Start Trigger Source Selection Register (DMA1_TRGSELx) (x=0~7)

Register description: After DMA1 selects the hardware-triggered start mode, the number of the event to be triggered is written into this register, and when the peripheral circuit event corresponding to the number occurs, DMA1 will be triggered by the event to start and transmit.

Offset address: 0x14, 0x18, 0x1C, x20, 0x24, 0x28, 0x2C, 0x30

Reset value: 0x000000FF

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]							

Bit	Symbol	Bit name	Description	Read and write
b31	COMEN[1]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG2 to trigger this AOS target	R/W
b30	COMEN[0]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG1 to trigger this AOS target	R/W
b29~b9	Reserved	-	Read as "0", write as "0"	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected	R/W

11.4.5 DMA2 Transfer Start Trigger Source Selection Register (DMA2_TRGSELx) (x=0~7)

Register description: After DMA2 selects the hardware-triggered start mode, the number of the event to be triggered is written into this register, and when the peripheral circuit event corresponding to the number occurs, DMA2 will be triggered by the event to start and transmit.

Offset address: 0x34, 0x38, 0x3C, x40, 0x44, 0x48, 0x4C, 0x50

Reset value: 0x000000FF

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]							

Bit	Symbol	Bit name	Description	Read and write
b31	COMEN[1]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG2 to trigger this AOS target	R/W
b30	COMEN[0]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG1 to trigger this AOS target	R/W
b29~b9	Reserved	-	Read as "0", write as "0"	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected	R/W

11.4.6 DMA Channel Re-configure Trigger Source Selection Register (DMA_TRGSELRC)

Register description: After the DMA selects the hardware-triggered start mode, the number of the event to be triggered is written into this register. When the peripheral circuit event corresponding to the number occurs, the DMA will be triggered by the event to re-configure the channel. DMA1 and DMA2 share this register

Offset address: 0x54

Reset value: 0x0000001FF

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]							

Bit	Symbol	Bit name	Description	Read and write
b31	COMEN[1]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG2 to trigger this AOS target	R/W
b30	COMEN[0]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG1 to trigger this AOS target	R/W
b29~b9	Reserved	-	Read as "0", write as "0"	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected	R/W

11.4.7 Timer6 Trigger Source Selection Register (TMR6_HTSSRx) ($x=0\sim3$)

Register description: After Timer6 selects the hardware-triggered start mode, the number of the event to be triggered is written into this register. When the peripheral circuit event corresponding to the number occurs, Timer6 will be triggered and started by the event.

Offset address: 0x58, 0x5C, 0x60, 0x64

Reset value: 0x0000001FF

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]							
<hr/>															
Bit	Symbol	Bit name	Description										Read and write		
b31	COMEN[1]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG2 to trigger this AOS target										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG1 to trigger this AOS target										R/W		
b29~b9	Reserved	-	Read as "0", write as "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected										R/W		

11.4.8 Event Port Trigger Source Selection Register (PEVNTTRGSR12, PEVNTTRGSR34)

Register description: Set the corresponding event number to trigger the event port to output the specified level, or latch the input state of the I/O port. PEVNTTRGSR12 sets the trigger source of Event Port1 and Event Port2, and PEVNTTRGSR34 sets the trigger source of Event Port3 and Event Port4.

Offset address: 0x68, 0x6C

Reset value: 0x000001FF

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]							
Bit	Symbol	Bit name	Description										Read and write		
b31	COMEN[1]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the public trigger event of AOS_COMTRG2 to trigger this AOS target										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable the public trigger event of AOS_COMTRG1 to trigger this AOS target 1: Enable the public trigger event of AOS_COMTRG1 to trigger this AOS target										R/W		
b29~b9	Reserved	-	Read as "0", write as "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected										R/W		

11.4.9 Timer0 Trigger Source Select Register (TMR0_HTSSR)

Register description: After Timer0 selects the hardware-triggered start mode, the number of the event to be triggered is written into this register, and when the peripheral circuit event corresponding to the number occurs, Timer0 will be triggered and started by the event.

Offset address: 0x70

Reset value: 0x0000001FF

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]							
<hr/>															
Bit	Symbol	Bit name	Description										Read and write		
b31	COMEN[1]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG2 to trigger this AOS target										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG1 to trigger this AOS target										R/W		
b29~b9	Reserved	-	Read as "0", write as "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected										R/W		

11.4.10 Timer2 Trigger Source Select Register (TMR2_HTSSR)

Register description: After Timer2 selects the hardware-triggered start mode, the number of the event to be triggered is written into this register, and when the peripheral circuit event corresponding to the number occurs, Timer2 will be triggered and started by the event.

Offset address: 0x74

Reset value: 0x0000001FF

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]							

Bit	Symbol	Bit name	Description	Read and write
b31	COMEN[1]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG2 to trigger this AOS target	R/W
b30	COMEN[0]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to this AOS target 1: Enable the public trigger event of AOS_COMTRG1 to trigger this AOS target	R/W
b29~b9	Reserved	-	Read as "0", write as "0"	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected	R/W

11.4.11 HASH Trigger Source Selection Register A (HASH_ITRGSEL A)

Register description: After the data is written into HASH_DR, the hardware event trigger source is selected through this register to start the HASH operation. Note, do not select other trigger signals other than DMA_BT Cx(x=0~7).

Offset address: 0x7C

Reset value: 0x0000001FF

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]							
<hr/>															
Bit	Symbol	Bit name	Description										Read and write		
b31	COMEN[1]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG2 to trigger this AOS target										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG1 to trigger this AOS target										R/W		
b29~b9	Reserved	-	Read as "0", write as "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected										R/W		

11.4.12 HASH Trigger Source Selection Register B (HASH_ITRGSELB)

Register description: After the last group of data is written into HASH_DR, the hardware event trigger source is selected through this register to notify HASH to perform the last operation. Note, do not select other trigger signals than DMA_TCx(x=0~7).

Offset address: 0x78

Reset value: 0x0000001FF

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]							
<hr/>															
Bit	Symbol	Bit name	Description										Read and write		
b31	COMEN[1]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG2 to trigger this AOS target										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG1 to trigger this AOS target										R/W		
b29~b9	Reserved	-	Read as "0", write as "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected										R/W		

11.4.13 TimerA Trigger Source Selection Register (TMRA_HTSSRx) (x=0~3)

Register description: After TimerA selects the hardware-triggered start mode, the number of the event to be triggered is written into this register, and when the peripheral circuit event corresponding to the number occurs, TimerA will be triggered and started by the event.

Offset address: 0x80, 0x84, 0x88, 0x8C

Reset value: 0x0000001FF

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]							
<hr/>															
Bit	Symbol	Bit name	Description										Read and write		
b31	COMEN[1]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG2 to trigger this AOS target										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG1 to trigger this AOS target										R/W		
b29~b9	Reserved	-	Read as "0", write as "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected										R/W		

11.4.14 OTS Trigger Source Selection Register (OTS_TRG)

Register description: After the OTS selects the hardware-triggered start mode, the number of the event to be triggered is written into this register, and when the peripheral circuit event corresponding to the number occurs, the OTS will be triggered and started by the event.

Offset address: 0x90

Reset value: 0x0000001FF

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]							
<hr/>															
Bit	Symbol	Bit name	Description										Read and write		
b31	COMEN[1]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG2 to trigger this AOS target										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG1 to trigger this AOS target										R/W		
b29~b9	Reserved	-	Read as "0", write as "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected										R/W		

11.4.15 ADC1 Conversion Start Trigger Source Selection Register ADC1_ITRGSELRx (x=0, 1)

Register description: After ADC1 selects the hardware trigger start mode, write the number of the event to be triggered into this register. When the peripheral circuit event corresponding to the number occurs, ADC1 will be triggered and started by the event.

Offset address: 0x94, 0x98

Reset value: 0x000000FF

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]							
<hr/>															
Bit	Symbol	Bit name	Description										Read and write		
b31	COMEN[1]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG2 to trigger this AOS target										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG1 to trigger this AOS target										R/W		
b29~b9	Reserved	-	Read as "0", write as "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected										R/W		

11.4.16 ADC2 Conversion Start Trigger Source Selection Register ADC2_ITRGSELRx (x=0, 1)

Register description: After ADC2 selects the hardware trigger start mode, write the number of the event to be triggered into this register. When the peripheral circuit event corresponding to the number occurs, ADC2 will be triggered and started by the event.

Offset address: 0x9C, 0xA0

Reset value: 0x000000FF

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]							
<hr/>															
Bit	Symbol	Bit name	Description										Read and write		
b31	COMEN[1]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG2 to trigger this AOS target										R/W		
b30	COMEN[0]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG1 to trigger this AOS target										R/W		
b29~b9	Reserved	-	Read as "0", write as "0"										R/W		
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected										R/W		

11.4.17 ADC3 Conversion Start Trigger Source Selection Register ADC3_ITRGSELRx(x=0,1)

Register description: After ADC3 selects the hardware-triggered startup mode, write the number of the event to be triggered into this register, and when the peripheral circuit event corresponding to the number occurs, ADC3 will be triggered and started by the event.

Offset address: 0xA4, 0xA8

Reset value: 0x000000FF

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]							

Bit	Symbol	Bit name	Description	Read and write
b31	COMEN[1]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG2 to trigger this AOS target	R/W
b30	COMEN[0]	Common trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to trigger this AOS target 1: Enable the common trigger event of AOS_COMTRG1 to trigger this AOS target	R/W
b29~b9	Reserved	-	Read as "0", write as "0"	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected	R/W

11.4.18 Common Trigger Source Selection Register 1 (AOS_COMTRG1)

Register description: Write the number of the event that will generate a trigger in AOS_COMTRG1. If the COMEN[0] bit value of the dedicated trigger source selection register of one or more AOS targets is 1, the peripheral circuit event will trigger one or more AOS targets to start when the peripheral circuit event corresponding to the number occurs.

Offset address: 0xAC

Reset value: 0x0000001FF

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]					
Bit	Symbol	Bit name	Description										Read and write		
b31~b9	Reserved	-	Read as "0", write as "0"										R/W		
b8~b0	COMTRG[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected										R/W		

11.4.19 Common Trigger Source Selection Register 2 (AOS_COMTRG2)

Register description: Write the number of the event that will generate a trigger in AOS_COMTRG2. If the COMEN[1] bit value of the dedicated trigger source selection register of one or more AOS targets is 1, the peripheral circuit event will trigger one or more AOS targets to start when the peripheral circuit event corresponding to the number occurs.

Offset address: 0xB0

Reset value: 0x0000001FF

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]					
Bit	Symbol	Bit name	Description										Read and write		
b31~b9	Reserved	-	Read as "0", write as "0"										R/W		
b8~b0	COMTRG[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected										R/W		

12 Storage Protection Unit (MPU)

12.1 Introduction

The MPU can provide memory protection and improve system security by preventing unauthorized access.

This chip has built-in six MPU units for the host and one MPU unit for the IP.

Table 12-1 Introduction to MPU Module

Modules	Content
ARM MPUs	CPU memory protection unit 8 area, see ARM MPU description for details
System DMA_1 MPU: SMPU1	Storage Protection Unit of System DMA_1 16 areas, 8 areas are dedicated to system DMA, and 8 areas are shared by all DMAs
System DMA_2 MPU: SMPU2	Storage protection unit of system DMA_2 16 areas, 8 areas are dedicated to system DMA, and 8 areas are shared by all DMAs
USBFS-DMA MPU: FMPU	Storage Protection Unit of USBFS-DMA 8 areas, common to all DMAs
USBHS-DMA MPU: HMPU	Storage Protection Unit of USBHS-DMA 8 areas, common to all DMAs
ETH-DMA MPU: EMPU	Storage Protection Unit for ETH-DMA 8 areas, common to all DMAs
IPMPU	Access protection unit for system IP and security-related IP

Among them, the ARM MPU provides the access control of the CPU to the entire 4G address space, and the introduction is omitted.

SMPU1/SMPU2/FMPU/HMPU/EMPU respectively provide system DMA_1/system DMA_2/USBFS-DMA/USBHS-DMA/ETH-DMA read and write access control to all 4G address space. When accessing the forbidden space, the MPU action can be set to ignore/bus error/non-maskable interrupt/reset.

IPMPU provides access control to system IP and security-related IP in unprivileged mode.

12.2 Functional description

12.2.1 Locale settings

The MPU performs permission management on storage space by region. Each area can independently set the base address and area size. The range that can be set is 32Byte~4GByte, the size must be 2^n Byte ($n=5\sim32$), and the lower n bits of the corresponding base address are 0.

The address space that is not covered by any area is called the background area.

12.2.2 Permission settings

Each area including the background area can be independently set to enable read/disable read and enable write/disable write for each DMA. If addresses overlap between different areas, the set prohibition takes precedence.

12.2.3 MPU action selection

When a prohibited access occurs, the access is ignored (read access reads 0, write access is ignored), and the corresponding action can be set, which can be set as:

- ignore
- bus error
- Nomaskable interrupt
- Reduction

12.2.4 Start the MPU

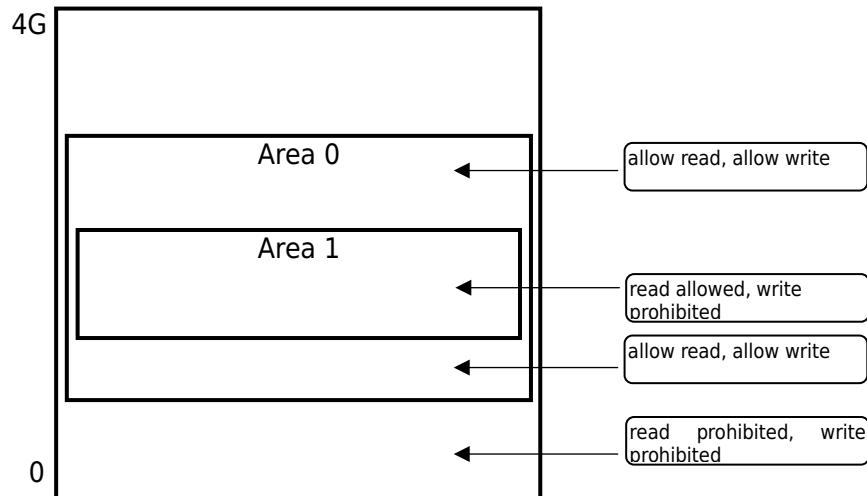
SMPU1/SMPU2/FMPU/HMPU/EMPU can be enabled independently.

It is recommended to enable the MPU after setting the area scope/authority setting/action selection.

12.3 Application examples

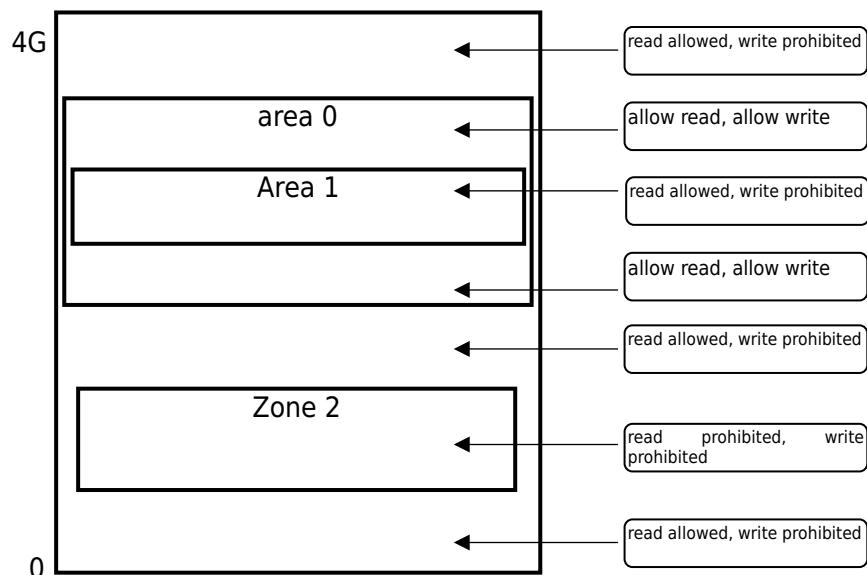
12.3.1 Only allow partial space access

Example: Set the background area permission to prohibit read/disable write, area 0 to allow read/allow write, area 1 to allow read and prohibit write, and area 0 to cover area 1.



12.3.2 Only block access to some spaces

Example: Set the background area permission to allow read/disable write, area 0 to allow read/allow write, area 1 to allow read/disable write, area 0 to cover area 1, and area 2 to prohibit read/disable write.



12.4 Register description

The registers of this module can only be set by the CPU.

MPU base address: 0x40050000

Table 12-2 MPU register list

Offset address	Register name	Initial value	Name	Write protection
+00~+3C	MPU_RGDO~15	0x00000000	Area 0~15 Range Description Register	MPUWE
+40	MPU_SR	0x00000000	MPU Status Register	No
+44	MPU_ECLR	0x00000000	MPU Error Flag Clear Register	No
+48	MPU_WP	0x00000000	MPU write protection register	WKEY
+4C	MPU_IPPR	0x00000000	IP Access Protection Register	MPUWE
+50	MPU_S1RGE	0x00000000	SMPU1 area enable register	MPUWE
+54	MPU_S1RGWP	0x00000000	SMPU1 area write permission register	MPUWE
+58	MPU_S1RGRP	0x00000000	SMPU1 area read permission register	MPUWE
+5C	MPU_S1CR	0x00000000	SMPU1 Control Register	MPUWE
+60	MPU_S2RGE	0x00000000	SMPU2 area enable register	MPUWE
+64	MPU_S2RGWP	0x00000000	SMPU2 area write permission register	MPUWE
+68	MPU_S2RGRP	0x00000000	SMPU2 area read permission register	MPUWE
+6C	MPU_S2CR	0x00000000	SMPU2 Control Register	MPUWE
+70	MPU_FRGE	0x00000000	FMPU Region Enable Register	MPUWE
+74	MPU_FRGWP	0x00000000	FMPU area write permission register	MPUWE
+78	MPU_FRGRP	0x00000000	FMPU area read permission register	MPUWE
+7C	MPU_FCR	0x00000000	FMPU Control Register	MPUWE
+80	MPU_HRGE	0x00000000	HMPU area enable register	MPUWE
+84	MPU_HRGWP	0x00000000	HMPU area write permission register	MPUWE
+88	MPU_HRGRP	0x00000000	HMPU area read permission register	MPUWE
+8C	MPU_HCR	0x00000000	HMPU Control Register	MPUWE
+90	MPU_ERGE	0x00000000	EMPU Region Enable Register	MPUWE
+94	MPU_ERGWP	0x00000000	EMPU area write permission register	MPUWE
+98	MPU_ERGRP	0x00000000	EMPU area read permission register	MPUWE
+9C	MPU_ECR	0x00000000	EMPU Control Register	MPUWE

12.4.1 Area range description register MPU_RGDr (n=0~15)

Reset value: 0x00000000

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]					
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b5	MPURGnADDR[31:5]	area address	base Set the base address of the area n, the effective number of bits is related to 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 the area n 0000~00011: reserved, setting prohibited 00100: 32Byte 00101: 64Byte ... 11110: 2GByte 11111: 4GByte	R/W											

12.4.2 Status flag register MPU_SR

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										EMPUEAF	HMPUEAF	FMPUEAF	SMPU2EAF	SMPU1EAF	
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b5	Reserved	-	Reserved bit, read as 0, write 0 when writing	R/W											
b4	EMPUEAF	EMPU error flag	0: ETH-DMA has no error access 1: ETH-DMA has an incorrect access	R											
b3	HMPUEAF	HMPU error flags	0: USBHS-DMA no error access occurred 1: USBHS-DMA error access occurred	R											
b2	FMPUEAF	FMPU error flag	0: No erroneous access occurred for USBFS-DMA 1: An error has occurred in the USBFS-DMA access	R											
b1	SMPU2EAF	SMPU2 error flag	0: System DMA_2 does not have a bad access 1: Bad access occurred in system DMA_2	R											
b0	SMPU1EAF	SMPU1 error flag	0: System DMA_1 did not have a bad access 1: Bad access occurred in system DMA_1	R											

Writes to this register will be ignored, use MPUECLR to clear the error flag.

12.4.3 Flag Clear Register MPU_ECLR

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															

Bit	Marking	Place name	Function	Read and write
b31~b5	Reserved	-	Reserved bit, read as 0, write 0 when writing	R/W
b4	EMPUECLR	EMPU error flag cleared	Write 1 to clear EMPUEAF to 0	R/W
b3	HMPUECLR	HMPU error flag cleared	Write 1 to clear HMPUEAF to 0	R/W
b2	FMPUECLR	FMPU error flag cleared	Write 1 to clear FMPUEAF to 0	R/W
b1	SMPU2ECLR	SMPU2 error flag cleared	Write 1 to clear SMPU2EAF to 0	R/W
b0	SMPU1ECLR	SMPU1 error flag cleared	Write 1 to clear SMPU1EAF to 0	R/W

The read value of this register is fixed to 0x00000000.

12.4.4 Write protection register MPU_WP

Reset value: 0x00000000

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															

Bit	Marking	Place name	Function	Read and write
b31~b6	Reserved	-	Reserved bit, read as 0, write 0 when writing	R/W
b15~b1	WKEY[15:1]	write code	When writing to MPUWE, you must write to WKEY at the same time 'b1001_0110_1010_0101, read as 0	R/W
b0	MPUWE	MPU register write enable	0: MPU address register/control register not allowed to be written 1: MPU address register/control register write enabled	RW

Writing 0x96a5 to this register can set MPUWE to 1, writing 0x96a4 can clear MPUWE to 0, and writing other values cannot change MPUWE.

12.4.5 SMPU1 area enable register MPU_S1RGE

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
S1RG15E	S1RG14E	S1RG13E	S1RG12E	S1RG11E	S1RG10E	S1RG9E	S1RG8E	S1RG7E	S1RG6E	S1RG5E	S1RG4E	S1RG3E	S1RG2E	S1RG1E	S1RG0E
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Reserved bit, read as 0, write 0 when writing	R/W											
b15	S1RG15E	SMPU1 area 15 enable	0: Area 15 of SMPU1 is invalid 1: Area 15 of SMPU1 is valid	R/W											
b14	S1RG14E	SMPU1 area 14 enable	0: Area 14 of SMPU1 is invalid 1: Area 14 of SMPU1 is valid	R/W											
b13	S1RG13E	SMPU1 area 13 enable	0: Area 13 of SMPU1 is invalid 1: Area 13 of SMPU1 is valid	R/W											
b12	S1RG12E	SMPU1 area 12 enable	0: Area 12 of SMPU1 is invalid 1: Area 12 of SMPU1 is valid	R/W											
b11	S1RG11E	SMPU1 area 11 enable	0: Area 11 of SMPU1 is invalid 1: Area 11 of SMPU1 is valid	R/W											
b10	S1RG10E	SMPU1 area 10 enable	0: Area 10 of SMPU1 is invalid 1: Area 10 of SMPU1 is valid	R/W											
b9	S1RG9E	SMPU1 Region 9 Enable	0: Area 9 of SMPU1 is invalid 1: Area 9 of SMPU1 is valid	R/W											
b8	S1RG8	SMPU1 Region 8 Enable	0: Area 8 of SMPU1 is invalid 1: Area 8 of SMPU1 is valid	R/W											
B7	S1RG7E	SMPU1 Region 7 Enable	0: Area 7 of SMPU1 is invalid 1: Area 7 of SMPU1 is valid	R/W											
b6	S1RG6E	SMPU1 Region 6 Enable	0: Area 6 of SMPU1 is invalid 1: Area 6 of SMPU1 is valid	R/W											
b5	S1RG5E	SMPU1 area 5 enable	0: Area 5 of SMPU1 is invalid 1: Area 5 of SMPU1 is valid	R/W											
b4	S1RG4E	SMPU1 Region 4 Enable	0: Area 4 of SMPU1 is invalid 1: Area 4 of SMPU1 is valid	R/W											
b3	S1RG3E	SMPU1 area 3 enable	0: Area 3 of SMPU1 is invalid 1: Area 3 of SMPU1 is valid	R/W											
b2	S1RG2E	SMPU1 area 2 enable	0: Area 2 of SMPU1 is invalid 1: Area 2 of SMPU1 is valid	R/W											
b1	S1RG1E	SMPU1 area 1 enable	0: Area 1 of SMPU1 is invalid 1: Area 1 of SMPU1 is valid	R/W											
b0	S1RG0E	SMPU1 area 0 enable	0: Area 0 of SMPU1 is invalid 1: Area 0 of SMPU1 is valid	R/W											

12.4.6 SMPU1 area write permission register MPU_S1RGWP

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
S1RG15 WP	S1RG14 WP	S1RG13 WP	S1RG12 WP	S1RG11 WP	S1RG10 WP	S1RG9 WP	S1RG8 WP	S1RG7 WP	S1RG6 WP	S1RG5 WP	S1RG4 WP	S1RG3 WP	S1RG2 WP	S1RG1 WP	S1RG0 WP
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Reserved bit, read as 0, write 0 when writing	R/W											
b15	S1RG15WP	SMPU1 area 15 write permission	0: Region 15 allows system DMA_1 writes 1: Region 15 disables system DMA_1 writes	R/W											
b14	S1RG14WP	SMPU1 area 14 write permission	0: Region 14 allows system DMA_1 writes 1: Region 14 disables system DMA_1 writes	R/W											
b13	S1RG13WP	SMPU1 area 13 write permission	0: Region 13 allows system DMA_1 writes 1: Region 13 disables system DMA_1 writes	R/W											
b12	S1RG12WP	SMPU1 area 12 write permission	0: Region 12 allows system DMA_1 writes 1: Region 12 disables system DMA_1 writes	R/W											
b11	S1RG11WP	SMPU1 area 11 write permission	0: Region 11 allows system DMA_1 writes 1: Region 11 disables system DMA_1 writes	R/W											
b10	S1RG10WP	SMPU1 area 10 write permission	0: Region 10 allows system DMA_1 writes 1: Region 10 disables system DMA_1 writes	R/W											
b9	S1RG9WP	SMPU1 area 9 write permission	0: Region 9 allows system DMA_1 writes 1: Region 9 disables system DMA_1 writes	R/W											
b8	S1RG8WP	SMPU1 area 8 write permission	0: Region 8 allows system DMA_1 writes 1: Region 8 disables system DMA_1 writes	R/W											
B7	S1RG7WP	SMPU1 area 7 write permission	0: Region 7 allows system DMA_1 writes 1: Region 7 disables system DMA_1 writes	R/W											
b6	S1RG6WP	SMPU1 area 6 write permission	0: Region 6 allows system DMA_1 writes 1: Region 6 disables system DMA_1 writes	R/W											
b5	S1RG5WP	SMPU1 area 5 write permission	0: Region 5 allows system DMA_1 to write 1: Region 5 disables system DMA_1 writes	R/W											
b4	S1RG4WP	SMPU1 area 4 write permission	0: Region 4 allows system DMA_1 writes 1: Region 4 disables system DMA_1 writes	R/W											
b3	S1RG3WP	SMPU1 area 3 write permission	0: Region 3 allows system DMA_1 to write 1: Region 3 disables system DMA_1 writes	R/W											
b2	S1RG2WP	SMPU1 area 2 write permission	0: Region 2 allows system DMA_1 to write 1: Region 2 disables system DMA_1 writes	R/W											
b1	S1RG1WP	SMPU1 area 1 write permission	0: Region 1 allows system DMA_1 writes 1: Region 1 disables system DMA_1 writes	R/W											
b0	S1RG0WP	SMPU1 area 0 write permission	0: Region 0 allows system DMA_1 writes 1: Region 0 disables system DMA_1 writes	R/W											

12.4.7 SMPU1 area read permission register MPU_S1RGRP

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
S1RG15RP	S1RG14RP	S1RG13RP	S1RG12RP	S1RG11RP	S1RG10RP	S1RG9RP	S1RG8RP	S1RG7RP	S1RG6RP	S1RG5RP	S1RG4RP	S1RG3RP	S1RG2RP	S1RG1RP	S1RG0RP
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Reserved bit, read as 0, write 0 when writing	R/W											
b15	S1RG15RP	SMPU1 area 15 read permission	0: Region 15 allows system DMA_1 reads 1: Region 15 disables system DMA_1 reads	R/W											
b14	S1RG14RP	SMPU1 area 14 read permission	0: Region 14 allows system DMA_1 reads 1: Region 14 disables system DMA_1 reads	R/W											
b13	S1RG13RP	SMPU1 area 13 read permission	0: Region 13 allows system DMA_1 reads 1: Region 13 disables system DMA_1 read	R/W											
b12	S1RG12RP	SMPU1 area 12 read permission	0: Region 12 allows system DMA_1 reads 1: Region 12 disables system DMA_1 reads	R/W											
b11	S1RG11RP	SMPU1 area 11 read permission	0: Region 11 allows system DMA_1 reads 1: Region 11 disables system DMA_1 reads	R/W											
b10	S1RG10RP	SMPU1 area 10 read permission	0: Region 10 allows system DMA_1 reads 1: Region 10 disables system DMA_1 reads	R/W											
b9	S1RG9RP	SMPU1 area 9 read permission	0: Region 9 allows system DMA_1 reads 1: Region 9 disables system DMA_1 reads	R/W											
b8	S1RG8RP	SMPU1 area 8 read permission	0: Region 8 allows system DMA_1 reads 1: Region 8 disables system DMA_1 reads	R/W											
B7	S1RG7RP	SMPU1 area 7 read permission	0: Region 7 allows system DMA_1 reads 1: Region 7 disables system DMA_1 reads	R/W											
b6	S1RG6RP	SMPU1 area 6 read permission	0: Region 6 allows system DMA_1 reads 1: Region 6 disables system DMA_1 reads	R/W											
b5	S1RG5RP	SMPU1 area 5 read permission	0: Region 5 allows system DMA_1 reads 1: Region 5 disables system DMA_1 reads	R/W											
b4	S1RG4RP	SMPU1 area 4 read permission	0: Region 4 allows system DMA_1 reads 1: Region 4 disables system DMA_1 reads	R/W											
b3	S1RG3RP	SMPU1 area 3 read permission	0: Region 3 allows system DMA_1 reads 1: Region 3 disables system DMA_1 read	R/W											
b2	S1RG2RP	SMPU1 area 2 read permission	0: Region 2 allows system DMA_1 reads 1: Region 2 disables system DMA_1 read	R/W											
b1	S1RG1RP	SMPU1 area 1 read permission	0: Region 1 allows system DMA_1 reads 1: Region 1 disables system DMA_1 read	R/W											
b0	S1RG0RP	SMPU1 area 0 read permission	0: Region 0 allows system DMA_1 reads 1: Region 0 disables system DMA_1 reads	R/W											

12.4.8 SMPU1 Control Register MPU_S1CR

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								SMPU1E	Reserved		SMPU1ACT[1:0]	SMPU1BWP	SMPU1BRP			
Bit	Marking		Place name			Function								Read and write		
b31~b8	Reserved		-			Reserved bit, read as 0, write 0 when writing								R/W		
B7	SMPU1E		SMPU1 enable			0: SMPU1 is invalid 1: SMPU1 is valid								R/W		
b6~b4	Reserved		-			Reserved bit, read as 0, write 0 when writing								R/W		
b3~b2	SMPU1ACT[1:0]		SMPU1 action selection			Set the action when system DMA_1 access is prohibited 00: ignore (read access reads 0, write access is ignored) 01: Ignore + bus error 10: Ignore + Unmaskable Interrupt 11: Reset								R/W		
b1	SMPU1BWP		SMPU1 write setting	background	permission	0: SMPU1 background space allows system DMA_1 to write 1: SMPU1 background space prohibits system DMA_1 writing								R/W		
b0	SMPU1BRP		SMPU1 read permission setting	background	permission	0: SMPU1 background space allows system DMA_1 to read 1: SMPU1 background space prohibits system DMA_1 reading								R/W		

When multiple area settings overlap, the priority is: Set Prohibit > Set Allow

12.4.9 SMPU2 area enable register MPU_S2RGE

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
S2RG15E	S2RG14E	S2RG13E	S2RG12E	S2RG11E	S2RG10E	S2RG9E	S2RG8E	S2RG7E	S2RG6E	S2RG5E	S2RG4E	S2RG3E	S2RG2E	S2RG1E	S2RG0E
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Reserved bit, read as 0, write 0 when writing	R/W											
b15	S2RG15E	SMPU2 area 15 enable	0: Area 15 of SMPU2 is invalid 1: Area 15 of SMPU2 is valid	R/W											
b14	S2RG14E	SMPU2 area 14 enable	0: Area 14 of SMPU2 is invalid 1: Area 14 of SMPU2 is valid	R/W											
b13	S2RG13E	SMPU2 area 13 enable	0: Area 13 of SMPU2 is invalid 1: Area 13 of SMPU2 is valid	R/W											
b12	S2RG12E	SMPU2 area 12 enable	0: Area 12 of SMPU2 is invalid 1: Area 12 of SMPU2 is valid	R/W											
b11	S2RG11E	SMPU2 area 11 enable	0: Area 11 of SMPU2 is invalid 1: Area 11 of SMPU2 is valid	R/W											
b10	S2RG10E	SMPU2 area 10 enable	0: Area 10 of SMPU2 is invalid 1: Area 10 of SMPU2 is valid	R/W											
b9	S2RG9E	SMPU2 Region 9 Enable	0: Area 9 of SMPU2 is invalid 1: Area 9 of SMPU2 is valid	R/W											
b8	S2RG8	SMPU2 Region 8 Enable	0: Area 8 of SMPU2 is invalid 1: Area 8 of SMPU2 is valid	R/W											
B7	S2RG7E	SMPU2 Region 7 Enable	0: Area 7 of SMPU2 is invalid 1: Area 7 of SMPU2 is valid	R/W											
b6	S2RG6E	SMPU2 Region 6 Enable	0: Area 6 of SMPU2 is invalid 1: Area 6 of SMPU2 is valid	R/W											
b5	S2RG5E	SMPU2 Region 5 Enable	0: Area 5 of SMPU2 is invalid 1: Area 5 of SMPU2 is valid	R/W											
b4	S2RG4E	SMPU2 Region 4 Enable	0: Area 4 of SMPU2 is invalid 1: Area 4 of SMPU2 is valid	R/W											
b3	S2RG3E	SMPU2 Region 3 Enable	0: Area 3 of SMPU2 is invalid 1: Area 3 of SMPU2 is valid	R/W											
b2	S2RG2E	SMPU2 Region 2 Enable	0: Area 2 of SMPU2 is invalid 1: Area 2 of SMPU2 is valid	R/W											
b1	S2RG1E	SMPU2 Region 1 Enable	0: Area 1 of SMPU2 is invalid 1: Area 1 of SMPU2 is valid	R/W											
b0	S2RG0E	SMPU2 area 0 enable	0: Area 0 of SMPU2 is invalid 1: Area 0 of SMPU2 is valid	R/W											

12.4.10 SMPU2 area write permission register MPU_S2RGWP

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
S2RG15 WP	S2RG14 WP	S2RG13 WP	S2RG12 WP	S2RG11 WP	S2RG10 WP	S2RG9 WP	S2RG8 WP	S2RG7 WP	S2RG6 WP	S2RG5 WP	S2RG4 WP	S2RG3 WP	S2RG2 WP	S2RG1 WP	S2RG0 WP
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Reserved bit, read as 0, write 0 when writing	R/W											
b15	S2RG15WP	SMPU2 area 15 write permission	0: Region 15 allows system DMA_2 writes 1: Region 15 disables system DMA_2 writes	R/W											
b14	S2RG14WP	SMPU2 area 14 write permission	0: Region 14 allows system DMA_2 writes 1: Region 14 disables system DMA_2 writes	R/W											
b13	S2RG13WP	SMPU2 area 13 write permission	0: Region 13 allows system DMA_2 writes 1: Region 13 disables system DMA_2 writes	R/W											
b12	S2RG12WP	SMPU2 area 12 write permission	0: Region 12 allows system DMA_2 writes 1: Region 12 disables system DMA_2 writes	R/W											
b11	S2RG11WP	SMPU2 area 11 write permission	0: Region 11 allows system DMA_2 writes 1: Region 11 disables system DMA_2 writes	R/W											
b10	S2RG10WP	SMPU2 area 10 write permission	0: Region 10 allows system DMA_2 writes 1: Region 10 disables system DMA_2 writes	R/W											
b9	S2RG9WP	SMPU2 area 9 write permission	0: Region 9 allows system DMA_2 writes 1: Region 9 disables system DMA_2 writes	R/W											
b8	S2RG8WP	SMPU2 area 8 write permission	0: Region 8 allows system DMA_2 writes 1: Region 8 disables system DMA_2 writes	R/W											
B7	S2RG7WP	SMPU2 area 7 write permission	0: Region 7 allows system DMA_2 writes 1: Region 7 disables system DMA_2 writes	R/W											
b6	S2RG6WP	SMPU2 area 6 write permission	0: Region 6 allows system DMA_2 writes 1: Region 6 disables system DMA_2 writes	R/W											
b5	S2RG5WP	SMPU2 area 5 write permission	0: Region 5 allows system DMA_2 writes 1: Region 5 disables system DMA_2 writes	R/W											
b4	S2RG4WP	SMPU2 area 4 write permission	0: Region 4 allows system DMA_2 writes 1: Region 4 disables system DMA_2 writes	R/W											
b3	S2RG3WP	SMPU2 area 3 write permission	0: Region 3 allows system DMA_2 to write 1: Region 3 disables system DMA_2 writes	R/W											
b2	S2RG2WP	SMPU2 area 2 write permission	0: Region 2 allows system DMA_2 writes 1: Region 2 disables system DMA_2 writes	R/W											
b1	S2RG1WP	SMPU2 area 1 write permission	0: Region 1 allows system DMA_2 writes 1: Region 1 disables system DMA_2 writes	R/W											
b0	S2RG0WP	SMPU2 area 0 write permission	0: Region 0 allows system DMA_2 writes 1: Region 0 disables system DMA_2 writes	R/W											

12.4.11 SMPU2 area read permission register MPU_S2RGRP

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
S2RG15RP	S2RG14RP	S2RG13RP	S2RG12RP	S2RG11RP	S2RG10RP	S2RG9RP	S2RG8RP	S2RG7RP	S2RG6RP	S2RG5RP	S2RG4RP	S2RG3RP	S2RG2RP	S2RG1RP	S2RG0RP
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Reserved bit, read as 0, write 0 when writing	R/W											
b15	S2RG15RP	SMPU2 area 15 read permission	0: Region 15 allows system DMA_2 reads 1: Region 15 disables system DMA_2 reads	R/W											
b14	S2RG14RP	SMPU2 area 14 read permission	0: Region 14 allows system DMA_2 reads 1: Region 14 disables system DMA_2 reads	R/W											
b13	S2RG13RP	SMPU2 area 13 read permission	0: Region 13 allows system DMA_2 reads 1: Region 13 disables system DMA_2 reads	R/W											
b12	S2RG12RP	SMPU2 area 12 read permission	0: Region 12 allows system DMA_2 reads 1: Region 12 disables system DMA_2 reads	R/W											
b11	S2RG11RP	SMPU2 area 11 read permission	0: Region 11 allows system DMA_2 reads 1: Region 11 disables system DMA_2 reads	R/W											
b10	S2RG10RP	SMPU2 area 10 read permission	0: Region 10 allows system DMA_2 reads 1: Region 10 disables system DMA_2 reads	R/W											
b9	S2RG9RP	SMPU2 area 9 read permission	0: Region 9 allows system DMA_2 reads 1: Region 9 disables system DMA_2 reads	R/W											
b8	S2RG8RP	SMPU2 area 8 read permission	0: Region 8 allows system DMA_2 reads 1: Region 8 disables system DMA_2 reads	R/W											
B7	S2RG7RP	SMPU2 area 7 read permission	0: Region 7 allows system DMA_2 reads 1: Region 7 disables system DMA_2 reads	R/W											
b6	S2RG6RP	SMPU2 area 6 read permission	0: Region 6 allows system DMA_2 reads 1: Region 6 disables system DMA_2 reads	R/W											
b5	S2RG5RP	SMPU2 area 5 read permission	0: Region 5 allows system DMA_2 reads 1: Region 5 disables system DMA_2 reads	R/W											
b4	S2RG4RP	SMPU2 area 4 read permission	0: Region 4 allows system DMA_2 reads 1: Region 4 disables system DMA_2 reads	R/W											
b3	S2RG3RP	SMPU2 area 3 read permission	0: Region 3 allows system DMA_2 reads 1: Region 3 disables system DMA_2 reads	R/W											
b2	S2RG2RP	SMPU2 area 2 read permission	0: Region 2 allows system DMA_2 reads 1: Region 2 disables system DMA_2 reads	R/W											
b1	S2RG1RP	SMPU2 area 1 read permission	0: Region 1 allows system DMA_2 reads 1: Region 1 disables system DMA_2 reads	R/W											
b0	S2RG0RP	SMPU2 area 0 read permission	0: Region 0 allows system DMA_2 reads 1: Region 0 disables system DMA_2 reads	R/W											

12.4.12 SMPU2 Control Register MPU_S2CR

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								SMPU2E		Reserved			SMPU2ACT[1:0]	SMPU2WP	SMPU2BRP	
Bit	Marking		Place name		Function										Read and write	
b31~b8	Reserved		-		Reserved bit, read as 0, write 0 when writing										R/W	
B7	SMPU2E		SMPU2 enable		0: SMPU2 is invalid 1: SMPU2 is valid										R/W	
b6~b4	Reserved		-		Reserved bit, read as 0, write 0 when writing										R/W	
b3~b2	SMPU2ACT[1:0]		SMPU2 action selection		Set the action when system DMA_2 access is prohibited 00: ignore (read access reads 0, write access is ignored) 01: Ignore + bus error 10: Ignore + Unmaskable Interrupt 11: reset										R/W	
b1	SMPU2BWP		SMPU2 background write permission settings		0: SMPU2 background space allows system DMA_2 to write 1: SMPU2 background space prohibits system DMA_2 writing										R/W	
b0	SMPU2BRP		SMPU2 background read permission setting		0: SMPU2 background space allows system DMA_2 to read 1: SMPU2 background space prohibits system DMA_2 reading										R/W	

When multiple area settings overlap, the priority is: Set Prohibit > Set Allow

12.4.13 FMPU region enable register MPU_FRGE

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								FRG7E	FRG6E	FRG5E	FRG4E	FRG3E	FRG2E	FRG1E	FRG0E	
Bit	Marking		Place name		Function										Read and write	
b31~b8	Reserved		-		Reserved bit, read as 0, write 0 when writing										R/W	
B7	FRG7E		FMPU Region 7 Enable		0: Area 7 of the FMPU is invalid 1: Region 7 of the FMPU is valid										R/W	
b6	FRG6E		FMPU Region 6 Enable		0: Region 6 of the FMPU is invalid 1: Region 6 of the FMPU is valid										R/W	
b5	FRG5E		FMPU Region 5 Enable		0: Area 5 of the FMPU is invalid 1: Region 5 of the FMPU is valid										R/W	
b4	FRG4E		FMPU Region 4 Enable		0: Region 4 of the FMPU is invalid 1: Region 4 of the FMPU is valid										R/W	
b3	FRG3E		FMPU Region 3 Enable		0: Region 3 of the FMPU is invalid 1: Area 3 of the FMPU is valid										R/W	
b2	FRG2E		FMPU Region 2 Enable		0: Area 2 of the FMPU is invalid 1: Area 2 of the FMPU is valid										R/W	
b1	FRG1E		FMPU Region 1 Enable		0: Area 1 of the FMPU is invalid 1: Region 1 of the FMPU is valid										R/W	
b0	FRG0E		FMPU Region 0 Enable		0: Area 0 of the FMPU is invalid 1: Area 0 of the FMPU is valid										R/W	

12.4.14 FMPU area write permission register MPU_FRGWP

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								FRG7WP P	FRG6WP P	FRG5WP P	FRG4WP P	FRG3WP P	FRG2WP P	FRG1WP P	FRG0WP P

Bit	Marking	Place name	Function	Read and write
b31~b8	Reserved	-	Reserved bit, read as 0, write 0 when writing	R/W
B7	FRG7WP	FMPU area 7 write permission	0: Region 7 allows USBFS-DMA writes 1: Region 7 disables USBFS-DMA writes	R/W
b6	FRG6WP	FMPU area 6 write permission	0: Region 6 allows USBFS-DMA writes 1: Region 6 disables USBFS-DMA writing	R/W
b5	FRG5WP	FMPU area 5 write permission	0: Region 5 allows USBFS-DMA writes 1: Region 5 disables USBFS-DMA writing	R/W
b4	FRG4WP	FMPU area 4 write permission	0: Region 4 allows USBFS-DMA writes 1: Region 4 disables USBFS-DMA writes	R/W
b3	FRG3WP	FMPU area 3 write permission	0: Region 3 allows USBFS-DMA writes 1: Region 3 disables USBFS-DMA writing	R/W
b2	FRG2WP	FMPU area 2 write permission	0: Region 2 allows USBFS-DMA writes 1: Region 2 disables USBFS-DMA writing	R/W
b1	FRG1WP	FMPU area 1 write permission	0: Region 1 allows USBFS-DMA writes 1: Region 1 disables USBFS-DMA writes	R/W
b0	FRG0WP	FMPU area 0 write permission	0: Region 0 allows USBFS-DMA writes 1: Region 0 disables USBFS-DMA writes	R/W

12.4.15 FMPU area read permission register MPU_FRGRP

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								FRG7RP P	FRG6RP P	FRG5RP P	FRG4RP P	FRG3RP P	FRG2RP P	FRG1RP P	FRG0RP P

Bit	Marking	Place name	Function	Read and write
b31~b8	Reserved	-	Reserved bit, read as 0, write 0 when writing	R/W
B7	FRG7RP	FMPU area 7 read permission	0: Region 7 allows USBFS-DMA reads 1: Region 7 disables USBFS-DMA read	R/W
b6	FRG6RP	FMPU area 6 read permission	0: Region 6 allows USBFS-DMA reads 1: Region 6 disables USBFS-DMA read	R/W
b5	FRG5RP	FMPU area 5 read permission	0: Region 5 allows USBFS-DMA reads 1: Region 5 disables USBFS-DMA read	R/W
b4	FRG4RP	FMPU area 4 read permission	0: Region 4 allows USBFS-DMA reads 1: Region 4 disables USBFS-DMA read	R/W
b3	FRG3RP	FMPU area 3 read permission	0: Region 3 allows USBFS-DMA read 1: Region 3 disables USBFS-DMA read	R/W
b2	FRG2RP	FMPU area 2 read permission	0: Region 2 allows USBFS-DMA reads 1: Region 2 disables USBFS-DMA read	R/W
b1	FRG1RP	FMPU area 1 read permission	0: Region 1 allows USBFS-DMA reads 1: Region 1 disables USBFS-DMA read	R/W
b0	FRG0RP	FMPU area 0 read permission	0: Region 0 allows USBFS-DMA reads 1: Region 0 disables USBFS-DMA reads	R/W

12.4.16 FMPU control register MPU_FCR

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								FMPUE	Reserved			FMPUACT[1:0]	FMPUBWP	FMPUBRP		
Bit	Marking		Place name			Function								Read and write		
b31~b8	Reserved		-			Reserved bit, read as 0, write 0 when writing								R/W		
B7	FMPUE		FMPU enable			0: FMPU is invalid 1: FMPU is valid								R/W		
b6~b4	Reserved		-			Reserved bit, read as 0, write 0 when writing								R/W		
b3~b2	FMPUACT[1:0]		FMPU action selection			Set the behavior when USBFS-DMA access is prohibited 00: ignore (read access reads 0, write access is ignored) 01: Ignore + bus error 10: Ignore + Unmaskable Interrupt 11: Reset								R/W		
b1	FMPUBWP		FMPU background write permission setting			0: FMPU background space allows USBFS-DMA writes 1: FMPU background space prohibits USBFS-DMA writing								R/W		
b0	FMPUBRP		FMPU background read permission setting			0: FMPU background space allows USBFS-DMA read 1: FMPU background space prohibits USBFS-DMA reading								R/W		

When multiple area settings overlap, the priority is set prohibit > set allow

12.4.17 HMPU area enable register MPU_HRGE

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								HRG7E	HRG6E	HRG5E	HRG4E	HRG3E	HRG2E	HRG1E	HRG0E	
Bit	Marking		Place name			Function								Read and write		
b31~b8	Reserved		-			Reserved bit, read as 0, write 0 when writing								R/W		
B7	HRG7E		HMPU Region 7 Enable			0: Area 7 of the HMPU is invalid 1: Region 7 of the HMPU is valid								R/W		
b6	HRG6E		HMPU Region 6 Enable			0: Area 6 of the HMPU is invalid 1: Region 6 of the HMPU is valid								R/W		
b5	HRG5E		HMPU Region 5 Enable			0: Area 5 of the HMPU is invalid 1: Region 5 of the HMPU is valid								R/W		
b4	HRG4E		HMPU Region 4 Enable			0: Region 4 of the HMPU is invalid 1: Region 4 of the HMPU is valid								R/W		
b3	HRG3E		HMPU Region 3 Enable			0: Region 3 of the HMPU is invalid 1: Region 3 of the HMPU is valid								R/W		
b2	HRG2E		HMPU Region 2 Enable			0: Area 2 of the HMPU is invalid 1: Region 2 of the HMPU is valid								R/W		
b1	HRG1E		HMPU Region 1 Enable			0: Area 1 of the HMPU is invalid 1: Region 1 of the HMPU is valid								R/W		
b0	HRG0E		HMPU Region 0 Enable			0: Area 0 of the HMPU is invalid 1: Area 0 of the HMPU is valid								R/W		

12.4.18 HMPU area write permission register MPU_HRGWP

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								HRG7WP P	HRG6WP P	HRG5WP P	HRG4WP P	HRG3WP P	HRG2WP P	HRG1WP P	HRG0WP P
Bit	Marking	Place name	Function	Read and write											
b31~b8	Reserved	-	Reserved bit, read as 0, write 0 when writing	R/W											
B7	HRG7WP	HMPU area 7 write permission	0: Region 7 allows USBHS-DMA writes 1: Region 7 disables USBHS-DMA writes	R/W											
b6	HRG6WP	HMPU area 6 write permission	0: Region 6 allows USBHS-DMA writes 1: Region 6 disables USBHS-DMA writes	R/W											
b5	HRG5WP	HMPU area 5 write permission	0: Region 5 allows USBHS-DMA writes 1: Region 5 disables USBHS-DMA writes	R/W											
b4	HRG4WP	HMPU area 4 write permission	0: Region 4 allows USBHS-DMA writes 1: Region 4 disables USBHS-DMA writes	R/W											
b3	HRG3WP	HMPU area 3 write permission	0: Region 3 allows USBHS-DMA writes 1: Region 3 disables USBHS-DMA writes	R/W											
b2	HRG2WP	HMPU area 2 write permission	0: Region 2 allows USBHS-DMA writes 1: Region 2 disables USBHS-DMA writes	R/W											
b1	HRG1WP	HMPU area 1 write permission	0: Region 1 allows USBHS-DMA writes 1: Region 1 disables USBHS-DMA writes	R/W											
b0	HRG0WP	HMPU area 0 write permission	0: Region 0 allows USBHS-DMA writes 1: Region 0 disables USBHS-DMA writes	R/W											

12.4.19 HMPU area read permission register MPU_HRGRP

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								HRG7RP P	HRG6RP P	HRG5RP P	HRG4RP P	HRG3RP P	HRG2RP P	HRG1RP P	HRG0RP P
Bit	Marking	Place name	Function	Read and write											
b31~b8	Reserved	-	Reserved bit, read as 0, write 0 when writing	R/W											
B7	HRG7RP	HMPU area 7 read permission	0: Region 7 allows USBHS-DMA read 1: Region 7 disables USBHS-DMA read	R/W											
b6	HRG6RP	HMPU area 6 read permission	0: Region 6 allows USBHS-DMA read 1: Region 6 disables USBHS-DMA read	R/W											
b5	HRG5RP	HMPU area 5 read permission	0: Region 5 allows USBHS-DMA read 1: Region 5 disables USBHS-DMA read	R/W											
b4	HRG4RP	HMPU area 4 read permission	0: Region 4 allows USBHS-DMA read 1: Region 4 disables USBHS-DMA read	R/W											
b3	HRG3RP	HMPU area 3 read permission	0: Region 3 allows USBHS-DMA read 1: Region 3 disables USBHS-DMA read	R/W											
b2	HRG2RP	HMPU area 2 read permission	0: Region 2 allows USBHS-DMA read 1: Region 2 disables USBHS-DMA read	R/W											
b1	HRG1RP	HMPU area 1 read permission	0: Region 1 allows USBHS-DMA reads 1: Region 1 disables USBHS-DMA reads	R/W											
b0	HRG0RP	HMPU area 0 read permission	0: Region 0 allows USBHS-DMA reads 1: Region 0 disables USBHS-DMA reads	R/W											

12.4.20 HMPU control register MPU_HCR

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								HMPUE	Reserved			HMPUACT[1:0]	HMPUBWP	HMPUBRP		
Bit	Marking		Place name			Function								Read and write		
b31~b8	Reserved		-			Reserved bit, read as 0, write 0 when writing								R/W		
B7	HMPUE		HMPU enable			0: HMPU is invalid 1: HMPU is valid								R/W		
b6~b4	Reserved		-			Reserved bit, read as 0, write 0 when writing								R/W		
b3~b2	HMPUACT[1:0]		HMPU action selection			Set the behavior when USBHS-DMA access is prohibited 00: ignore (read access reads 0, write access is ignored) 01: Ignore + bus error 10: Ignore + Unmaskable Interrupt 11: reset								R/W		
b1	HMPUBWP		HMPU background write permission setting			0: HMPU background space allows USBHS-DMA write 1: HMPU background space prohibits USBHS-DMA writing								R/W		
b0	HMPUBRP		HMPU background read permission setting			0: HMPU background space allows USBHS-DMA read 1: HMPU background space prohibits USBHS-DMA reading								R/W		

When multiple area settings overlap, the priority is set prohibit > set allow

12.4.21 EMPU area enable register MPU_ERGE

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								ERG7E	ERG6E	ERG5E	ERG4E	ERG3E	ERG2E	ERG1E	ERGOE	
Bit	Marking		Place name			Function								Read and write		
b31~b8	Reserved		-			Reserved bit, read as 0, write 0 when writing								R/W		
B7	ERG7E		EMPU Region 7 Enable			0: Area 7 of the EMPU is invalid 1: Region 7 of the EMPU is valid								R/W		
b6	ERG6E		EMPU Region 6 Enable			0: Region 6 of the EMPU is invalid 1: Region 6 of the EMPU is valid								R/W		
b5	ERG5E		EMPU Region 5 Enable			0: Area 5 of the EMPU is invalid 1: Region 5 of the EMPU is valid								R/W		
b4	ERG4E		EMPU Region 4 Enable			0: Region 4 of the EMPU is invalid 1: Region 4 of the EMPU is valid								R/W		
b3	ERG3E		EMPU Region 3 Enable			0: Area 3 of the EMPU is invalid 1: Region 3 of the EMPU is valid								R/W		
b2	ERG2E		EMPU Region 2 Enable			0: Area 2 of the EMPU is invalid 1: Region 2 of the EMPU is valid								R/W		
b1	ERG1E		EMPU Region 1 Enable			0: Area 1 of the EMPU is invalid 1: Region 1 of the EMPU is valid								R/W		
b0	ERG0E		EMPU Region 0 Enable			0: Area 0 of the EMPU is invalid 1: Area 0 of the EMPU is valid								R/W		

12.4.22 EMPU area write permission register MPU_ERGWP

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								ERG7WP P	ERG6WP P	ERG5WP P	ERG4WP P	ERG3WP P	ERG2WP P	ERG1WP P	ERG0WP P	
Bit	Marking	Place name	Function												Read and write	
b31~b8	Reserved	-	Reserved bit, read as 0, write 0 when writing												R/W	
B7	ERG7WP	EMPU area 7 permission	write	0: Region 7 allows ETH-DMA writes 1: Area 7 prohibits ETH-DMA writing												R/W
b6	ERG6WP	EMPU area 6 permission	write	0: Region 6 allows ETH-DMA writes 1: Region 6 prohibits ETH-DMA writing												R/W
b5	ERG5WP	EMPU area 5 permission	write	0: Region 5 allows ETH-DMA writes 1: Region 5 prohibits ETH-DMA writing												R/W
b4	ERG4WP	EMPU area 4 permission	write	0: Region 4 allows ETH-DMA writes 1: Region 4 prohibits ETH-DMA writing												R/W
b3	ERG3WP	EMPU area 3 permission	write	0: Region 3 allows ETH-DMA writes 1: Region 3 prohibits ETH-DMA writing												R/W
b2	ERG2WP	EMPU area 2 permission	write	0: Region 2 allows ETH-DMA writes 1: Region 2 prohibits ETH-DMA writing												R/W
b1	ERG1WP	EMPU area 1 permission	write	0: Zone 1 allows ETH-DMA writes 1: Area 1 disables ETH-DMA writing												R/W
b0	ERG0WP	EMPU area 0 permission	write	0: Region 0 allows ETH-DMA writes 1: Area 0 disables ETH-DMA writing												R/W

12.4.23 EMPU area read permission register MPU_ERGRP

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								ERG7RP P	ERG6RP P	ERG5RP P	ERG4RP P	ERG3RP P	ERG2RP P	ERG1RP P	ERG0RP P	
Bit	Marking	Place name	Function												Read and write	
b31~b8	Reserved	-	Reserved bit, read as 0, write 0 when writing												R/W	
B7	ERG7RP	EMPU region 7 read permission	read	0: Region 7 allows ETH-DMA reads 1: Region 7 disables ETH-DMA read												R/W
b6	ERG6RP	EMPU region 6 read permission	read	0: Region 6 allows ETH-DMA reads 1: Region 6 disables ETH-DMA read												R/W
b5	ERG5RP	EMPU region 5 read permission	read	0: Zone 5 allows ETH-DMA reads 1: Region 5 prohibits ETH-DMA reading												R/W
b4	ERG4RP	EMPU area 4 read permission	read	0: Region 4 allows ETH-DMA reads 1: Region 4 disables ETH-DMA reading												R/W
b3	ERG3RP	EMPU region 3 read permission	read	0: Region 3 allows ETH-DMA read 1: Region 3 prohibits ETH-DMA reading												R/W
b2	ERG2RP	EMPU region 2 read permission	read	0: Region 2 allows ETH-DMA reads 1: Region 2 disables ETH-DMA read												R/W
b1	ERG1RP	EMPU area 1 read permission	read	0: Zone 1 allows ETH-DMA reads 1: Region 1 disables ETH-DMA read												R/W
b0	ERG0RP	EMPU area 0 read permission	read	0: Region 0 allows ETH-DMA reads 1: Region 0 disables ETH-DMA read												R/W

12.4.24 EMPU control register MPU_ECR

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								EMPUE	Reserved			EMPUACT[1:0]	EMPUBWP	EMPUBRP	
Bit	Marking	Place name			Function								Read and write		
b31~b8	Reserved	-			Reserved bit, read as 0, write 0 when writing								R/W		
B7	EMPUE	EMPU enable			0: EMPU is invalid 1: EMPU is valid								R/W		
b6~b4	Reserved	-			Reserved bit, read as 0, write 0 when writing								R/W		
b3~b2	EMPUACT[1:0]	EMPU action selection			Set the action when access is prohibited in ETH-DMA 00: ignore (read access reads 0, write access is ignored) 01: Ignore + bus error 10: Ignore + Unmaskable Interrupt 11: Reset								R/W		
b1	EMPUBWP	EMPU background write permission setting			0: EMPU background space allows ETH-DMA write 1: EMPU background space prohibits ETH-DMA writing								R/W		
b0	EMPUBRP	EMPU background read permission setting			0: EMPU background space allows ETH-DMA read 1: EMPU background space prohibits ETH-DMA reading								R/W		

When multiple area settings overlap, the priority is set prohibit > set allow

12.4.25 IP Access Protection Register MPU_IPPR

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
BUSERRE	-	MSTPWRP	MSTPRDP	SYSCWRP	SYSCRDP	INTCWRP	INTCRDP	SRAMCWRP	SRAMCRDP	DMPUWRP	DMPURDP	RTCWRP	RTCRDP	BKSRAWRP	BKSRAMRDP
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
SWDTWRP	SWDTRDP	WDTWRP	WDTRDP	-	-	EFMWRP	EFMRDP	CRCWRP	CRCRD	TRNGWRP	TRNGRDP	HASHWRP	HASHRDP	AESWRP	AESRDP

Bit	Marking	Place name	Function	Read and write
b31	BUSERRE	bus error enabled	0: Ignore access to protected IP 1: A bus error is returned when an access to the protected IP occurs	R/W
b30	Reserved	-	Reserved bit, read as 0, write 0 when writing	R/W
b29	MSTPWRP	MSTP write protection	0: Allow write operations to PWC_FCG0/1/2/3, PWC_FCG0PC 1: Disable write operations to PWC_FCG0/1/2/3 and PWC_FCG0PC	R/W
b28	MSTPRDP	MSTP read protection	0: Allow read operations to PWC_FCG0/1/2/3, PWC_FCG0PC 1: Prohibit the read operation of PWC_FCG0/1/2/3, PWC_FCG0PC	R/W
b27	SYSCWRP	SYSC write protection	0: Write operations to RMU/CMU/PWC are allowed 1: Disable write operations to RMU/CMU/PWC Note: When BUSERRE is set to 1, writes to RMU/PWC will not cause bus errors	R/W
b26	SYSCRDP	SYSC read protection	0: Allow read operations to RMU/CMU/PWC 1: Disable read operations on RMU/CMU/PWC Note: When BUSERRE is set to 1, no bus errors will occur on reads of the RMU/PWC	R/W
b25	INTCWRP	INTC write protection	0: Write operations to INTC are allowed 1: Disable writing to INTC	R/W
b24	INTCRDP	INTC read protection	0: Enable read operations to INTC 1: Disable the read operation of INTC	R/W
b23	SRAMCWRP	SRAMC write protection	0: Write operation to SRAMC is allowed 1: Disable writing to SRAMC	R/W
b22	SRAMCRDP	SRAMC read protection	0: Allow read operation to SRAMC 1: Disable the read operation of SRAMC	R/W
b21	DMPUWRP	DMPU write protection	0: Write operations to the DMPU are allowed 1: Disable writes to DMPU	R/W
b20	DMPURDP	DMPU read protection	0: Allow read operations to DMPU 1: Disable read operations on DMPU	R/W
b19	RTCWRP	RTC write protection	0: Write operation to RTC is allowed 1: Disable writing to RTC	R/W
b18	RTCRDP	RTC read protection	0: Allow read operation to RTC 1: Disable read operations on RTC	R/W
b17	BKSRAWRP	BKSRAWRP	write 0: Write operation to BKSRAWRP is allowed 1: Disable writing to BKSRAWRP	R/W
b16	BKSRAMRDP	BKSRAMRDP	read 0: Read operation of BKSRAWRP is allowed 1: Disable the read operation of BKSRAWRP	R/W
b15	SWDTWRP	SWDTWRP	SWDT write protection 0: Write operation to SWDTWRP is allowed 1: Disable writing to SWDTWRP	R/W
b14	SWDTRDP	SWDTRDP	SWDT read protection 0: Read operation of SWDTWRP is allowed 1: Disable the read operation of SWDTWRP	R/W
b13	WDTWRP	WDTWRP	WDT write protection 0: Write operation to WDTWRP is allowed 1: Disable write operation to WDTWRP	R/W
b12	WDTRDP	WDTRDP	WDT read protection 0: Allow read operation of WDTWRP 1: Disable the read operation of WDTWRP	R/W
b11-b10	Reserved	-	Reserved bit, read as 0, write 0 when writing	R/W
b9	EFMWRP	EFMWRP	EFM write protection 0: Write operation to EFMWRP is allowed 1: Disable writing to EFMWRP	R/W
b8	EFMRDP	EFMRDP	EFM read protection 0: Read operation of EFMWRP is allowed 1: Disable the read operation of EFMWRP	R/W
B7	CRCWRP	CRCWRP	CRC write protection 0: Write operation to CRCWRP is allowed 1: Disable writing to CRCWRP	R/W
b6	CRCRD	CRCRD	CRC read protection 0: Read operation of CRCWRP is allowed 1: Disable the read operation of CRCWRP	R/W
b5	TRNGWRP	TRNGWRP	TRNG write protection 0: Write operations to TRNGWRP are allowed 1: Disable writing to TRNGWRP	R/W

b4	TRNGRPD	TRNG read protection	0: Read operations on TRNG are allowed 1: Disable read operations on TRNG	R/W
b3	HASHWRP	HASH write protection	0: Write operation to HASH is allowed 1: Disable writing to HASH	R/W
b2	HASHRDP	HASH read protection	0: Allow read operations on HASH 1: Disable the read operation of HASH	R/W
b1	AESWRP	AES write protection	0: Allow writes to AES 1: Disable writes to AES	R/W
b0	AESRDP	AES read protection	0: Allow read operations on AES 1: Disable read operations on AES	R/W

In privileged mode, the object IP can be read and written without being affected by this register.

13 Keyboard Scanning Control Module (KEYSCAN)

13.1 Introduction

This product is equipped with a keyboard scan control module (KEYSCAN) unit. The KEYSCAN module supports keyboard array (row and column) scanning, the column is driven by the independent scan output KEYOUT_m ($m=0\sim 7$), and the row KEYIN_n ($n=0\sim 15$) is input as EIRQ_n ($n=0\sim 15$) be detected. This module realizes the key recognition function through the line scan query method.

KEYSCAN main features:

- EIRQ0~EIRQ15 can be independently selected as row input of the keyboard array.
- KEYOUT can be selected by register.
- Output low level in turn at a certain interval to scan the keyboard array.
- Scan time can be set.
- Stop scanning when the EIRQ interrupt is detected, and locate the pressed key according to the SSR.INDEX value and the IRQ interrupt flag (INT_EIFR.INT_EIFR).

13.2 KEYSCAN system block diagram

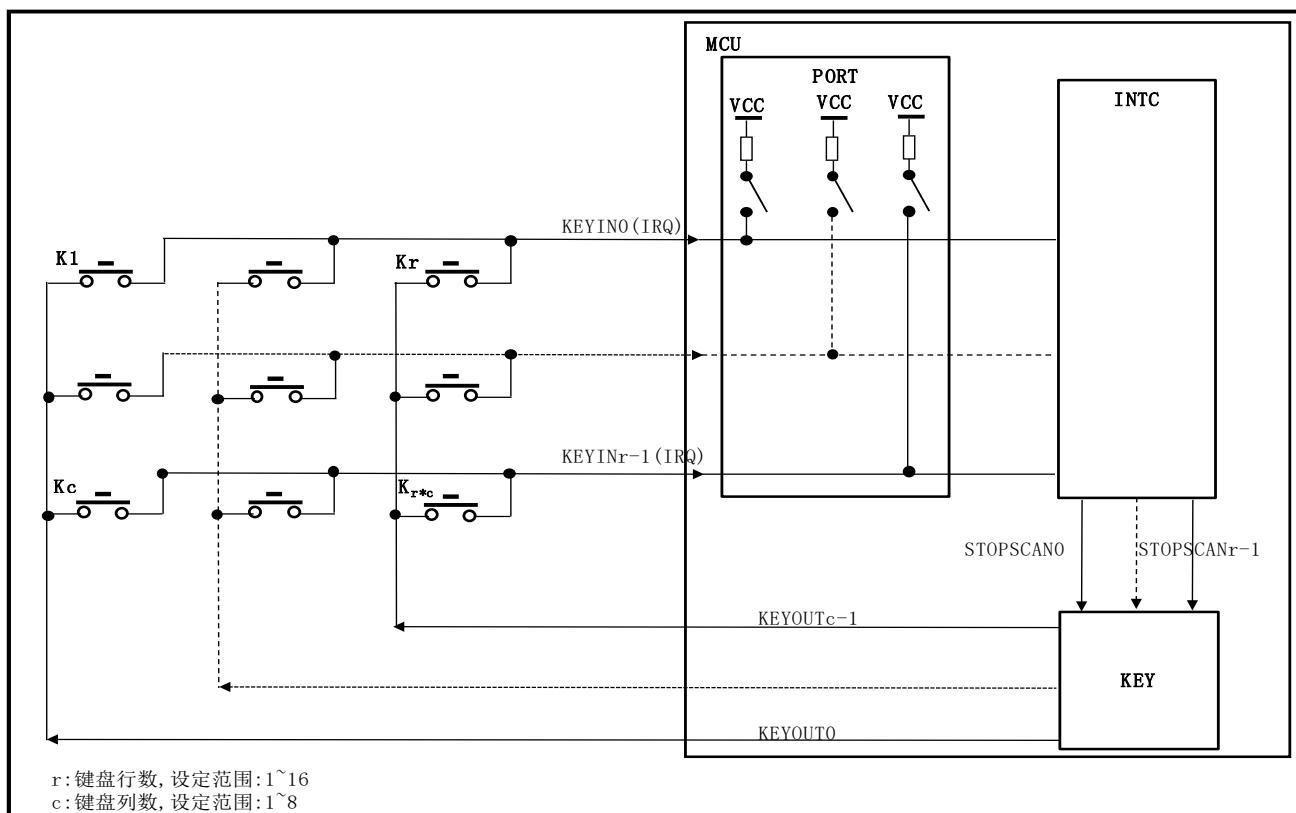


Figure 13-1 KEYSCAN system block diagram

13.3 Pin description

Table 13-1 KEYS CAN pin description

Pin name	Direction	Functional description
KEYINn	Input	Keyboard row input signal
KEYOUTm	Output	Keyboard column output signal

n:0~15 m:0~7

13.4 Functional description

This chapter will describe the keyboard scan function and key recognition function in detail.

13.4.1 Key recognition function

When a key is pressed, the row and column of the keyboard are short-circuited, and the row generates a falling edge, thereby generating the EIRQ interrupt flag, which is located by comparing the interrupt flag bit (INT_EIFR.INT_EIFR) and the value of SSR.INDEX[2:0] The currently pressed key.

Through the register SCR.KEYINSEL[15:0], KEYIN can be independently selected from EIRQ0~EIRQ15, and the KEYOUT pin can be selected through the register SCR.KEYOUTSEL[2:0], so that the number of rows and columns of the keyboard can be flexibly selected, the maximum It can support a keyboard array of 16 rows*8 columns.

13.4.2 Keyboard scan function

The keyboard scan function is: Continuously cyclically output a low level to the columns of the keyboard array, so that when a key is pressed, a corresponding EIRQ interrupt flag will be generated.

When SER.SEN is set to 1, KEYOUT0 outputs low level, KEYOUT1~KEYOUTn (n is set by SCR.KEYOUTSEL[2:0]) is HIZ, after the time set by SCR.T_LLEVEL[4:0], KEYOUT0~KEYOUTn pins are all HIZ, after the time set by SCR.T_HIZ[2:0], KEYOUT1 outputs low level, the other KEYOUT pins are HIZ, and so on. When a key is pressed and an EIRQ interrupt flag is generated, the keyboard scanning function stops, and the scan restarts automatically after the corresponding interrupt flag is cleared.

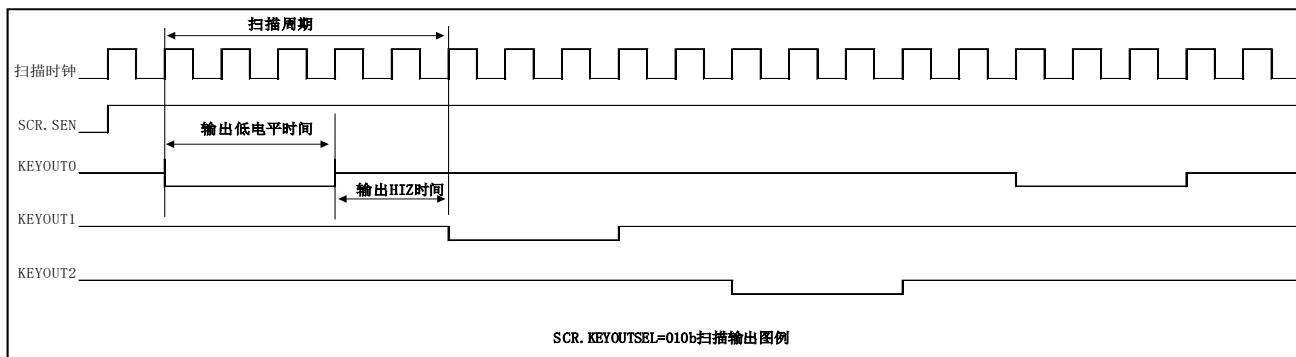


Figure 13-2 Schematic diagram of keyboard scanning function

13.4.3 Precautions for use

This module drives the keyboard column, and the keyboard row detection is realized by the external EIRQ function of the interrupt control module (INTC). EIRQ needs to select the falling edge detection, and enable the digital filtering function, and 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 the on-chip pull-up resistor is used, please refer to the PORT characteristics to select the appropriate scan time and filter time.

13.5 Register description

KEYSCAN_BASE_ADDR: 0x40050C00

Table 13-2 KEYS defense register list

Register name	Symbol	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 body register	KEYSCAN_SSR	0x08	32	0x0000 0000

13.5.1 KEYS defense Scan Control Register (KEYSCAN_SCR)

KEYSCAN Scan Control Register

offset address: 0x00

Reset value: 0x00000000

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]															
Bit	Marking	Place name	Function	Read and write											
b31~b29	T-HIZ[2:0]	Output HIZ time	HIZ time between KEYOUT output low level (number of scan clocks) Scanning period = output low level time + output HIZ time Setting value: HIZ cycles 000b: 4 001b: 8 010b: 16 011b: 32 100b: 64 101b: 256 110b: 512 111b: 1024 Note: SCR.T-HIZ[2:0] can only be set valid when SER.SEN=0	R/W											
b28~b24	T_LLEVEL[4:0]	output low time	Time between KEYOUT output low level (number of scan clocks) Scanning period = output low level time + output HIZ time Output low time = T_LLEVEL power of 2 number of scan clocks Note: SCR.T_LLEVEL[4:0] can only be set valid when SER.SEN=0, and the setting of 00000b and 00001b is prohibited, the maximum value that can be set is 11000b	R/W											
b23~b22	Reserved	-	Read as "0", write as "0"	R/W											
b21~b20	CKSEL[1:0]	Scan clock source selection bits	Scan clock source selection bits 00b: System clock HCLK 01b: Internal low-speed oscillator (LRC) 10b: External low speed oscillator (XTAL32) 11b: set ban Note: SCR.CKSEL[1:0] can only be set valid when SER.SEN=0	R/W											
b19	Reserved	-	Read as "0", write as "0"	R/W											
b18~b16	KEYOUTSEL[2:0]	output selection	KEYOUT output selection bit Set value: output selection 000b: prohibit 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	R/W											
b15~b0	KEYINSEL[15:0]	row input select bit	row input select bit, the selected row is used as the row of the keyboard array, and as EIRQn (n: 0~15) are detected KEYINSEL[n]=0: KEYINSEL[n] is not used as a row of the keyboard array KEYINSEL[n]=1: KEYINSEL[n] is used as the row of the keyboard array n range: 0~15 Note: SCR.KEYINSEL[15:0] can only be set valid when SER.SEN=0	R/W											

13.5.2 KEYS defense Scan Enable Register (KEYCAN_SER)

KEYSCAN Scan Enable Register

offset address: 0x04

Reset value: 0x00000000

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

Bit	Marking	Place name	Function	Read and write
b31~b1	Reserved	-	Read as "0", write as "0"	R/W
b0	SEN	scan enable bit	scan enable bit 0: Scanning disabled 1: scan enable	R/W

13.5.3 KEYS defense Scan Status Register (KEYCAN_SSR)

KEYSCAN Scan Status Register

offset address: 0x08

Reset value: 0x00000000

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]

Bit	Marking	Place name	Function	Read and write
b31~b3	Reserved	-	Read as "0", write as "0"	R/W
b2~b0	INDEX[2:0]	The current working 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: The SSR.INDEX[2:0] bits are read-only registers, and only the data read when SER.SEN=1 is meaningful	R

14 Internal clock calibrator (CTC)

14.1 Introduction

The Clock Trimming Controller (CTC) automatically calibrates the internal high-speed oscillator (HRC). Because the influence of working environment on HRC frequency may cause deviation, CTC can automatically adjust HRC frequency by hardware based on external high precision reference clock to obtain an accurate HRC clock.

The main features of CTC are as follows:

- Three external reference clock sources: XTAL, XTAL32 and CTCREF
- 16-bit calibration counter for frequency measurement with heavy load function
- 8-bit calibration deviation and 6-bit calibration values for frequency calibration
- Error interrupt to prompt calibration failure

14.2 Structural block diagram

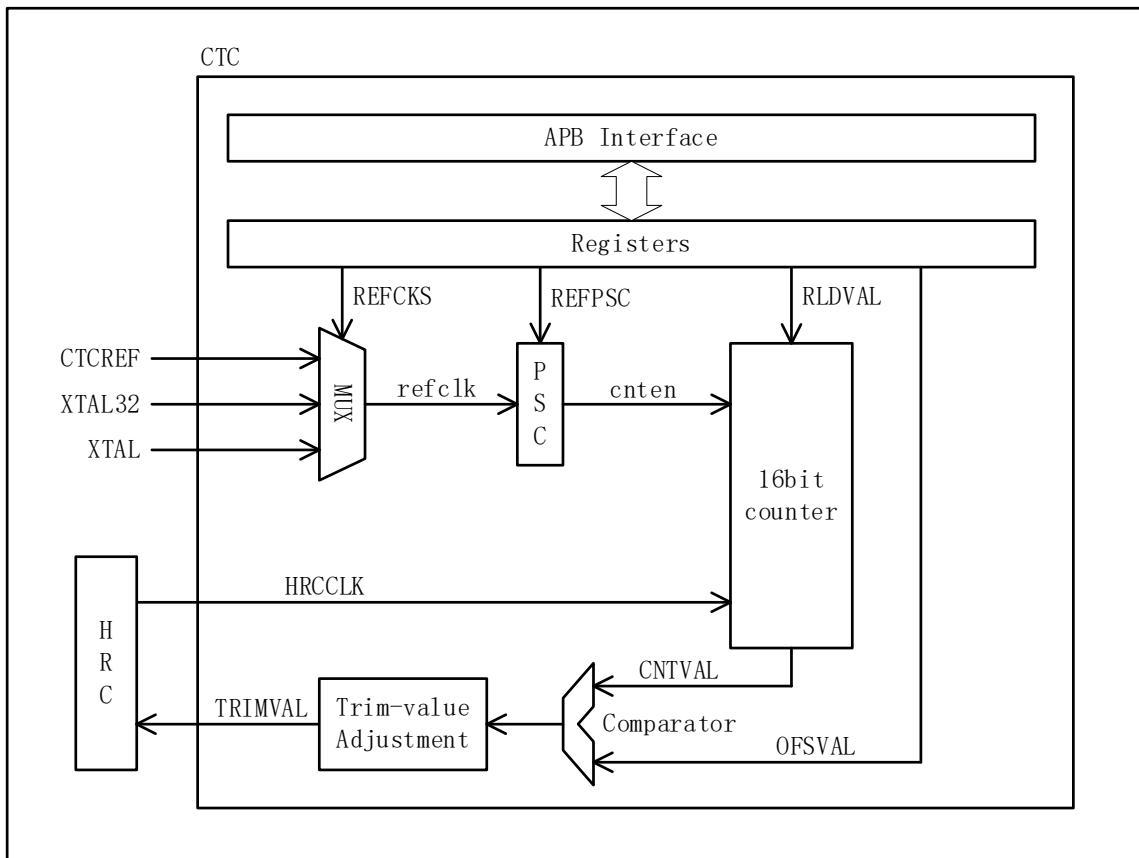


Figure 14-1 Basic block diagram of CTC

14.3 Functional description

14.3.1 Reference clock

The CTC has three external clock sources that can be used as the reference clock for calibrating the HRC frequency, namely the external high-speed clock (XTAL, ~24MHz), the external low-speed clock (XTAL32, 32.768kHz) and the external reference clock (CTCREF). You can select a reference clock by setting the REFCKS [1: 0] bit in the CTC _ CR1register.

The Jitter of HRC and the counting error in frequency measurement will affect the calibration accuracy. The effect of error can be reduced by setting the REFPSC [2: 0] bit in CTC _ CR1register to select suitable frequency division for reference clock. Table 14-1 andTable 14-2 is the error estimation value of reference clocks of different frequencies under different frequency division conditions. Please refer to Selecting an Appropriate Reference Clock Source and Frequency Division. If the frequency of the reference clock is not listed in the table, you can refer to the following formula to calculate the error. Basically, the measurement error should be made smaller than the calibration accuracy.

$$E = \frac{0.099\%}{\sqrt{(F_{hrc} \div F_{ref}) \times PSC \div 10^5}} + \frac{4}{(F_{hrc} \div F_{ref}) \times PSC}$$

Note:

F_{hrc} represents the target frequency of HRC

F_{ref} represents the frequency of the reference clock

PSC represents the number of reference clock dividers

(F_{hrc}/F_{ref}) × PSC indicates the count of the calibration counter when measuring the HRC frequency

Table 14-1 Measurement error when HRC target frequency is 20MHz

PSC \ Fref	32.768kHz	4MHz	8MHz	12MHz	24MHz
8	0.530%	Not recommended	Not recommended	Not recommended	Not recommended
32	0.244%	Not recommended	Not recommended	Not recommended	Not recommended
128	Irreducible	1.863%	3.000%	Not recommended	Not recommended
256	Irreducible	1.188%	1.863%	2.453%	Not recommended
512	Irreducible	0.775%	1.188%	1.540%	2.453%
1024	Irreducible	0.516%	0.775%	0.992%	1.540%
2048	Irreducible	0.348%	0.516%	0.653%	0.992%
4096	Irreducible	0.238%	0.348%	0.437%	0.653%

Note:

"Cannot be set" means that the setting exceeds the range of the calibration counter, and the calibration cannot be completed.

Note:

"Not recommended" means that the setting error is too large and cannot be accurately calibrated.

Table 14-2 Measurement error when HRC target frequency is 16MHz

PSC \ Fref	32.768kHz	4MHz	8MHz	12MHz	24MHz
8	0.603%	Not recommended	Not recommended	Not recommended	Not recommended
32	0.276%	Not recommended	Not recommended	Not recommended	Not recommended
128	Irreducible	2.165%	3.519%	Not recommended	Not recommended
256	Irreducible	1.369%	2.165%	2.866%	Not recommended
512	Irreducible	0.887%	1.369%	1.784%	2.866%
1024	Irreducible	0.587%	0.887%	1.140%	1.784%
2048	Irreducible	0.395%	0.587%	0.746%	1.140%
4096	Irreducible	0.269%	0.395%	0.497%	0.746%

Note:

"Cannot be set" means that the setting exceeds the range of the calibration counter, and the calibration cannot be completed.

Note:

"Not recommended" means that the setting error is too large and cannot be accurately calibrated.

14.3.2 Frequency calibration

After the CTCEN position of the CTC _ CR1register is 1, the 16-bit calibration counter loads RLDVAL from the CTC _ CR2register and counts down. The counting clock is provided by HRC. After that, if the count stop synchronization signal from the PSC is detected, the 16-bit calibration counter stops counting. If the PSC count stop synchronization signal is not always detected, the 16-bit calibration counter stops until the underflow is 0xFFFF and waits for the PSC count stop synchronization signal to arrive.

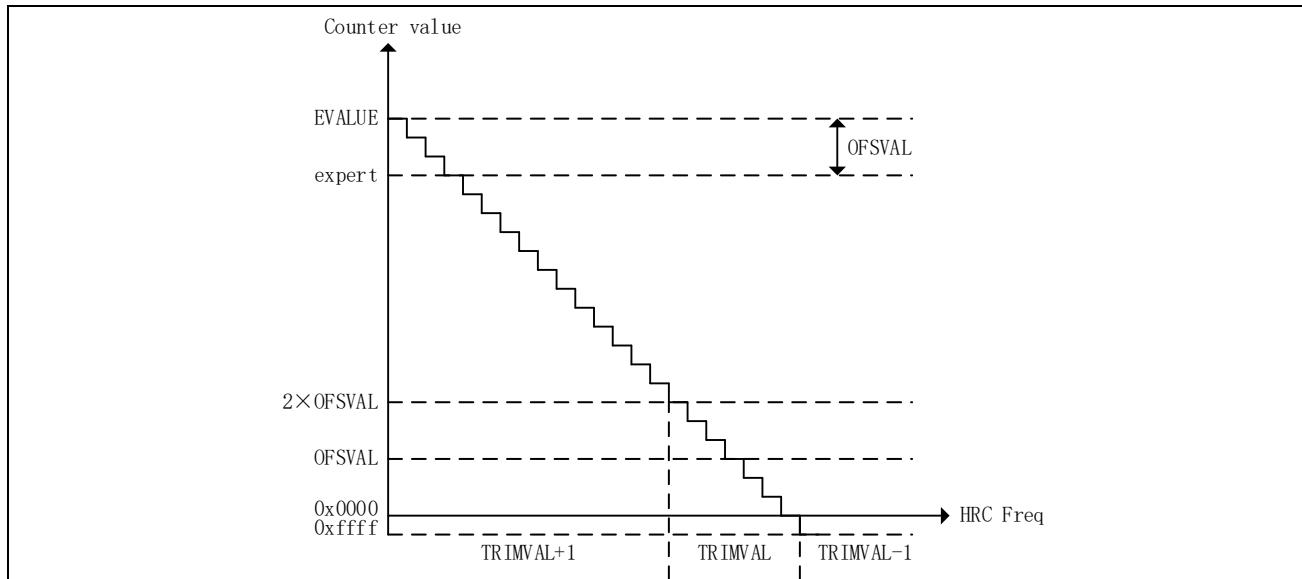


Figure 14-2 Schematic diagram of CTC calibration

After detecting the count stop synchronization signal from the PSC, the HRC frequency calibration function starts to execute. Since the calibration count offset OFSVAL is added when setting the calibration count overload value RLDVAL, if the 16-bit calibration counter value is greater than $2 \times$ OFSVAL and no underflow occurs, it means that the current frequency is slower than expected and that the value of TRMVAL [5: 0] in the CTC _ CR1register needs to be increased to increase the frequency. On the contrary, if the 16-bit calibration counter value underflows, the current frequency is faster than expected, and the TRMVAL value needs to be reduced to reduce the frequency.

- Stop counting when Counter > (OFSVAL × 2);
The value of TRMVAL in CTC _ CR1register is automatically increased by 1.
- $0 \leq$ Counter \leq (OFSVAL × 2);
The TRMVAL value in the CTC _ CR1register remains unchanged, and the TRIMOK bit in the CTC _ STRregister is set to 1.
- Counter = 0xFFFF;
The value of TRMVAL in CTC _ CR1register is automatically reduced by 1.

The median value of TRMVAL is 0x00. If the result of automatic incrementing by 1 is greater than 0x1F, the value of TRMVAL will remain 0x1F, and a calibration overflow occurs at the same time, and the TRMOVF bit in the CTC_STR register is set to 1. Conversely, if the result of automatic

decrement by 1 is less than 0x20, the value of TRMVAL will remain 0x20, and a calibration underflow will occur, and the TRMUDF bit in the CTC_STR register will be set to 1.

Calibration stops automatically when a calibration overflow or underflow occurs (CTCEN is reset automatically), and a calibration error interrupts if the ERRIE bit in the CTC_CR1 register is 1.

14.3.3 Programming guide

Take the example of calibrating the HRC to $(20 \pm 0.5\%)$ MHz using the XTAL32 divided by 8 as the reference clock.

Looking up the table, it can be seen that the frequency measurement error is 0.244%, which is less than 0.5% of the calibration allowable deviation.

- 1) Set CTC_CR2 = 0x132b0018, where
OFSVAL= $(20 \div 0.032768) \times 8 \times 0.5\% \approx 24$ (0x18)
RLDVAL= $(20 \div 0.032768) \times 8 + 24 \approx 4907$ (0x132b)
- 2) Set CTC_CR1 = 0x000000e0
TRMVAL = 0x00 (Calibrate from median)
CTCEN = 1 (start CTC)
ERRIE = 1 (Allow incorrect interrupt)
REFCKS=10b (select XTAL32 as the reference clock)
REFPSC = 000b (Select 8 frequency division of reference clock)
- 3) After a while, confirm the status of CTC_STR.TRIMOK
If TRIMOK=1, CTCEN is cleared
If TRIMOK = 0, return to (3)
- 4) Confirm that CTC_STR.CTCBSY=0, and follow up.

If an error interrupt occurs during the journey, the CTCEN bit is cleared automatically and the calibration stops due to a calibration overflow or a calibration underflow outside the calibrable range. In this case, please correct the OFSVAL, RLDVAL and reference clock settings and try the calibration again.

Requires attention: Frequency calibration is started or stopped by the CTCEN bit in the CTC_CR1 register. Therefore, do not change any other register settings after frequency calibration is started. To modify the settings, write the CTCEN bit to 0 and confirm that the CTCBSY bit in the CTC_STR register is 0.

14.4 Register Description

Base address: 0x40049C00

Table 14-3 List of CTC Registers

Register name	Symbol	Offset address	Bit width	Reset value
Clock calibration control register1	CTC_CR1	0x00	32	0x00000000
Clock calibration control register2	CTC_CR2	0x04	32	0x00000000
Clock calibration status register	CTC_STR	0x08	32	0x00000000

14.4.1 Clock Calibration Control Register1 (CTC_CR1)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16										
-	-	-	-	-	-	-	-	-	-	-	-	TRMVAL[5:0]													
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0										
-	-	-	-	-	-	-	-	CTC EN	ERR IE	REFCKS[1:0]	-	REFPSC[2:0]													

Bit	Marking	Place name	Function	Read and write
b31~b22	Reserved	-	Read as "0", write as "0"	R/W
b21~b16	TRMVAL[5:0]	HRC calibration value	The highest bit of TRMVAL is a symbol bit and the median value is 0. 0x20: -32 0x21: -31 0x3F: -1 0x00: 0 (median) 0x01: +1 0x1E: +30 0x1F: +31 CTCEN position 1 TRMVAL is read-only and automatically modified by hardware. If you need to modify it, please set the CTCEN bit to 0 and confirm that the CTC_STR.CTCBSY bit is 0 before modifying it.	R/W
b15~b8	Reserved	-	Read as "0", write as "0"	R/W
B7	CTCEN	Calibration enable	Used to start frequency calibration. 0: Stop frequency calibration 1: Start-up frequency calibration Condition 1: Software 1; Clear 0 condition: (1) Software clearing 0 (2) The hardware is cleared when the calibration is overflowed or underflowed;	R/W
b6	ERRIE	Error interrupt permission	Used to select whether error interrupt occurs when a calibration overflow or underflow occurs. 0: Prevent error interrupt 1: Allow error interrupt	R/W
b5~b4	REFCKS[1:0]	Reference clock selection	Used to select a reference timer for frequency calibration. 0xb: Select CTCREF as reference clock 10b: Select XTAL32 as reference clock 11b: Select XTAL as reference clock The value of REFCKS is not allowed to be modified at CTCEN position 1. If you need to modify it, please set the CTCEN bit to 0 and confirm that the CTC_STR.CTCBSY bit is 0 before modifying it.	R/W
b3	Reserved	-	Read as "0", write as "0"	R/W
b2~b0	REFPSC[2:0]	Reference clock division frequency	Used to select the frequency division of the reference clock. 000b: Reference clock divided by 8 001b: Reference clock divided by 32 010b: Reference clock divided by 128 011b: Reference clock divided by 256 100b: Reference clock divided by 512 101b: Reference clock divided by 1024 110b: Reference clock divided by 2048 111b: Reference clock divided by 4096 The value of REFPSC is not allowed to be modified at CTCEN position 1. If you need to modify it, please set the CTCEN bit to 0 and confirm that the CTC_STR.CTCBSY bit is 0 before modifying it.	R/W

14.4.2 Clock Calibration Control Register2 (CTC_CR2)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
RLDVAL[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	OFSVAL[7:0]

Bit	Marking	Place name	Function	Read and write
b31~b16	RLDVAL[15:0]	Calibration count overload	RLDVAL sets the 16-bit calibration counter, which counts down from the start of each calibration. $\text{RLDVAL} = (\text{Fhrc} \div \text{Fref}) + \text{OFSVAL}$ Fhrc: HRC standard frequency value; Fref: Reference clock frequency ÷ reference clock prescaler; OFSVAL: Calibration count deviation The value of RLDVAL is not allowed to be modified at CTCEN position 1. If you need to modify it, please set the CTCEN bit to 0 and confirm that the CTC_STR.CTCBSY bit is 0 before modifying it.	R/W
b15~b8	Reserved	-	Read as "0", write as "0"	R/W
b7~b0	OFSVAL[7:0]	Calibration count deviation	OFSVAL sets the permissible deviation value for HRC calibration. $\text{OFSVAL} = (\text{Fhrc} \div \text{Fref}) \times \text{TA}$ Fhrc: HRC standard frequency value; Fref: Reference clock frequency ÷ reference clock prescaler; TA: Calibration accuracy, such as 0.5%; CTCEN position 1 prevents OFSVAL from being modified. If you need to modify it, please set the CTCEN bit to 0 and confirm that the CTC_STR.CTCBSY bit is 0 before modifying it.	R/W

14.4.3 Clock calibration status register (CTC _ STR)

Reset value: 0x00000000

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
-	-	-	-	-	-	-	-	-	-	-	-	CTCBSY	TRMUDF	TRMOVF	TRIMOK

Bit	Marking	Place name	Function	Read and write
b31~b4	Reserved	-	Read as "0", write as "0"	R/W
b3	CTCBSY	CTC status flag	0: CTC is stopped 1: CTC is working	R
b2	TRMUDF	Calibration underflow mark	Calibration underflow indicates that TRMVAL has been reduced to 0x20 but has not yet been calibrated to the target range. Auto stop calibration when calibration underflow, error interrupt when ERRIE is 1. 0: No calibration underflow 1: Calibration underflow occurred	R
b1	TRMOVF	Calibration overflow mark	Calibration overflow indicates that TRMVAL has increased to 0x1F but has not yet been calibrated to the target range. Auto stop calibration when calibration overflow, error interrupt when ERRIE is 1. 0: No calibration overflow 1: Calibration overflow occurred	R
b0	TRIMOK	Calibration success mark	0: HRC frequency not calibrated to target range 1: HRC frequency calibrated to target range	R

15 DMA controller (DMA)

15.1 Introduction

DMA is used to transfer data between memory and peripheral function modules. It can exchange data between memories, between memory and peripheral function modules, or between peripheral function modules without CPU involvement.

- DMA bus is independent of CPU bus and transmitted according to AMBA AHB-Lite bus protocol.
- With 2 DMA control units, a total of 16 independent channels, can independently carry out different DMA transfers
- The transfer request of each channel is configured through the independent trigger source selection register
- One data block is transferred per request
- The minimum size of the data block is 1 data, and the maximum is 1024 data
- The width of each data can be configured as 8-bit, 16-bit or 32-bit
- Up to 65535 times of block transfer can be configured for one DMA transfer
- Source and destination addresses can be independently configured as fixed, increment, decrement, repeat or nonsequence (jump a specified offset) mode
- Three types of interrupts can be generated: Block transfer completion (DMA_BTC) interrupt, transfer completion(DMA_TC) interrupt, transfer error(DMA_ERR) interrupt. Each interrupt can be masked independently. Among them, DMA_BTC and DMA_TC can be used as an event output which can be used as a trigger source for other peripheral modules
- Support linked list transfer
- Support reconfigure transfer
- DMA module can be set to stop state to reduce power consumption when not in use

15.2 Module diagram

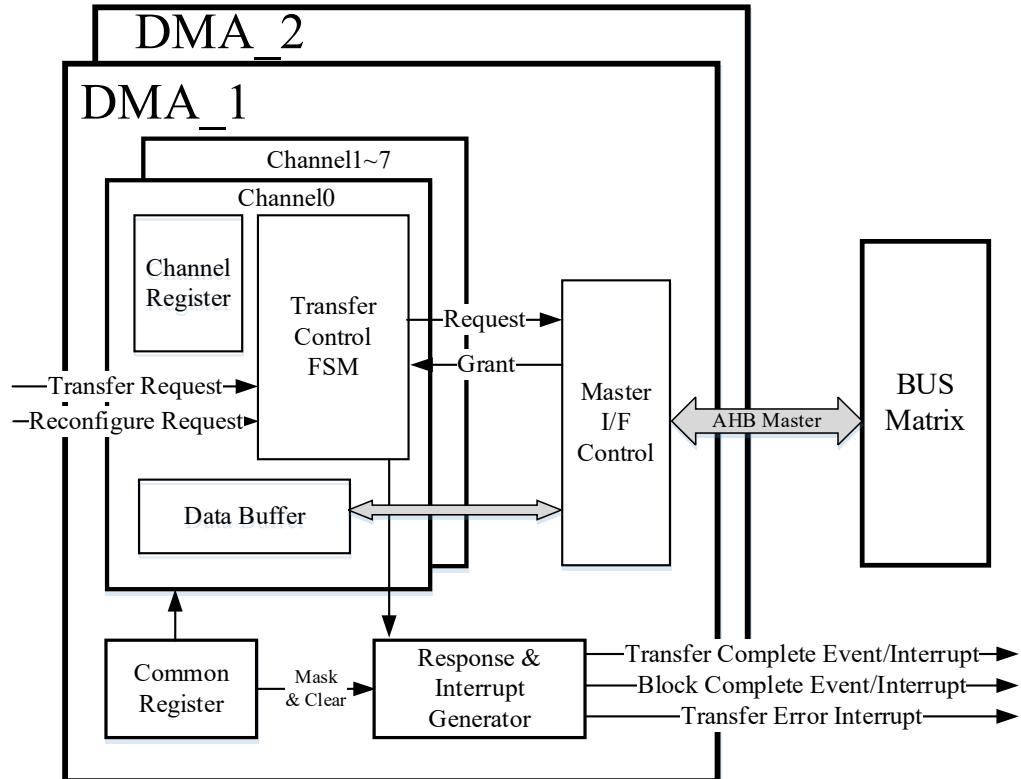


Figure 15-1 DMA structure diagram

15.3 Functional description

15.3.1 Enable DMA controller

After reset, DMA is in low power mode and clock is stopped. So, before using DMA, write 0 to PWC_FCG0.DMAx register bit to enable DMA clock first, and then write 1 to the DMA_EN.EN bit to enable DMA.

DMA transfer is started by the transfer request which is generated by AOS module, so the PWC_FCG0.AOS bit also needs to be written 0 to enable AOS.

When DMA is not used, or the chip needs to enter STOP mode, please set DMA_EN.EN to 0, and configure PWC_FCG0 to stop DMA clock. Please confirm that register DMA_CHSTAT.DMAACT is 0 to ensure that DMA has completed all transfers before writing 0 to DMA_EN.EN.

15.3.2 Channel selection and channel priority

Each DMA unit contains 8 channels, and each channel can be configured independently.

The priority order of the 8 channels is: Channel 0 > Channel 1 > Channel 2 > ... > Channel 7.

When multiple channels request DMA transfer, the highest priority channel will service first. But the channel that is already in transferring will not be interrupted, the high priority channel will start after the block transfer completion of the current running channel, if it wins the arbitration.

15.3.3 Start DMA

DMA transfer is started by transfer requests generated by peripheral module, these requests are configured through the Trigger Source Selection Register DMA_TRGSELx (x=0~7), each channel configures its transfer request source independently. When the transfer request asserts, and DMA module and the corresponding channel are enabled (DMA_EN.EN=1, DMA_CHEN.CHEN[x]=1), then channel x transfer is started.

Before using DMA, it is necessary to enable the AOS and DMA module by configuring register (PWC_FCG0).

15.3.4 Data block

The amount of transfer data corresponding to each DMA transfer request is represented by a data block. The size of the block is set by the Data Control Register DMA_DTCTLx.BLKSIZE, and a maximum of 1024 data can be set. The width of each data is determined by DMA_CHCTLx.HIZE, which can be 8-bit, 16-bit or 32-bit.

15.3.5 Address control

The source and destination addresses can be fixed, incremented, decremented, repeat or nonsequence.

fixed: The source address or destination address will be fixed during the transfer process.

Increment and decrement: The source address or destination address will increase or decrease according to the value of HSIZE after each data is transferred. For example, when HSIZE is 8-bit, the address will increase/decrease by 1 each time, when it is 16-bit, it will increase/decrease by 2 each time, and when it is 32-bit, it will increase/decrease by 4 each time.

Repeat: After the specified number of data is transferred, the source or destination address will reload the original setting address. The amount of data that needs to be transferred before the address reloading, that is, the size of the repeat area is set by the register DMA_RPT.

Nonsequence: After the specified amount of data is transferred, the source or destination address will jump a specified offset. The jump offset, and the amount of data to be transferred before jumping, that is, the size of the nonsequence area, is set by the registers DMA_SNSEQCTL/DMA_DNSEQCTL. When the conditions of address repeat and nonsequence jump are satisfied at the same time, address repeat is performed.

15.3.6 Number of transfers

The total number of data blocks transferred by a DMA channel is set by the CNT bit of the Data Control Register DMA_DTCTLx. The maximum number of transfers can be set to 65535. CNT loads to transfer counter after it is configured. Each time a data block is transferred, the transfer counter is decremented by 1. When the counter decreases to 0, it means that all the data of this channel is transferred completely, the channel enable bit DMA_CHEN.CHEN[x] is automatically cleared, and a transfer completion interrupt is generated. If DMA_DTCTLx.CNT is set to 0 at the beginning of the transfer, it means infinite transfers, each request triggers one block data transfer, but the channel transfer enable bit is not cleared, and no transfer completion (DMA_TC) interrupt is generated.

Note:

- After CHEN[x] is automatically cleared to 0, if you need to start the channel again, you need to set the channel configuration register once again and then write 1 to CHEN[x] to enable the channel. Otherwise, the subsequent transfer will start according to the end state of the previous transfer, that is, infinite transfer as CNT is 0x0, and the initial source/destination address is the value calculated according to the previous transfer.

15.3.7 Interrupt and event signal output

DMA controller can generate the following three types of interrupts:

Block transfer completion interrupt DMA_BTxC: Generated after completing a data block transfer.

Transfer completion interrupt DMA_TCx: Generated after completing the number of transfers set by register DMA_DTCTLx.CNT.

Transfer error interrupt DMA_ERR: Generated when the transfer request overflows (that is, the channel receives a transfer request again when this channel's previous request has not been processed yet), or when a bus error occurs during the transfer (such as accessing an illegal or protected address), where, A bus error will terminate the transfer immediately.

Except for the transfer request overflow error, other interrupts can be enabled or disabled through the register DMA_CHCTLx.IE. In addition, all interrupts are also equipped with independent MASK registers to mask the interrupts.

The above-mentioned DMA_BTxC and DMA_TCx interrupts can also be output as event signals and can be used as trigger sources for other peripheral circuits. The event output is controlled by the MASK register, but not controlled by the interrupt permission bit DMA_CHCTLx.IE. DMA_BTxC, DMA_TCx, DMA_ERR events will set the corresponding status register, not affected by DMA_CHCTLx.IE, or MASK register.

15.3.8 Linked list transfer

DMA controller supports linked list transfer, which is also called chain transfer. The chain transfer needs to configure 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, linked list pointer and other transfer control information.

DMA_SARx
DMA_DARx
DMA_DTCTLx
DMA_RPTx
DMA_SNSEQCTLx
DMA_DNSEQCTLx
DMA_LLPx
DMA_CHCTLx

The LLP is called the Linked List Pointer, and its value represents the first address of the next descriptor in the memory. When using chain transfer, first write 1 to LL PEN in the Channel Control Register DMA_CHCTLx to enable chain transfer, and write the descriptor information of the first transfer into the corresponding registers. And then write the descriptors of subsequent transfers

sequentially at the specified address in memory. When the chain transfer needs to be ended, write 0 to LLPEN of DMA_CHCTLx in the last descriptor, and the DMA controller will end the chain transfer after that transfer is completed.

When the transfer of a descriptor ends, the next descriptor specified by the LLP is loaded from the memory into the channel configuration register. Wait for the next transfer request to start the first transfer of the new descriptor. Or according to the setting of register DMA_CHCTLx.LLPRUN, start the first transfer directly after loading the new descriptor.

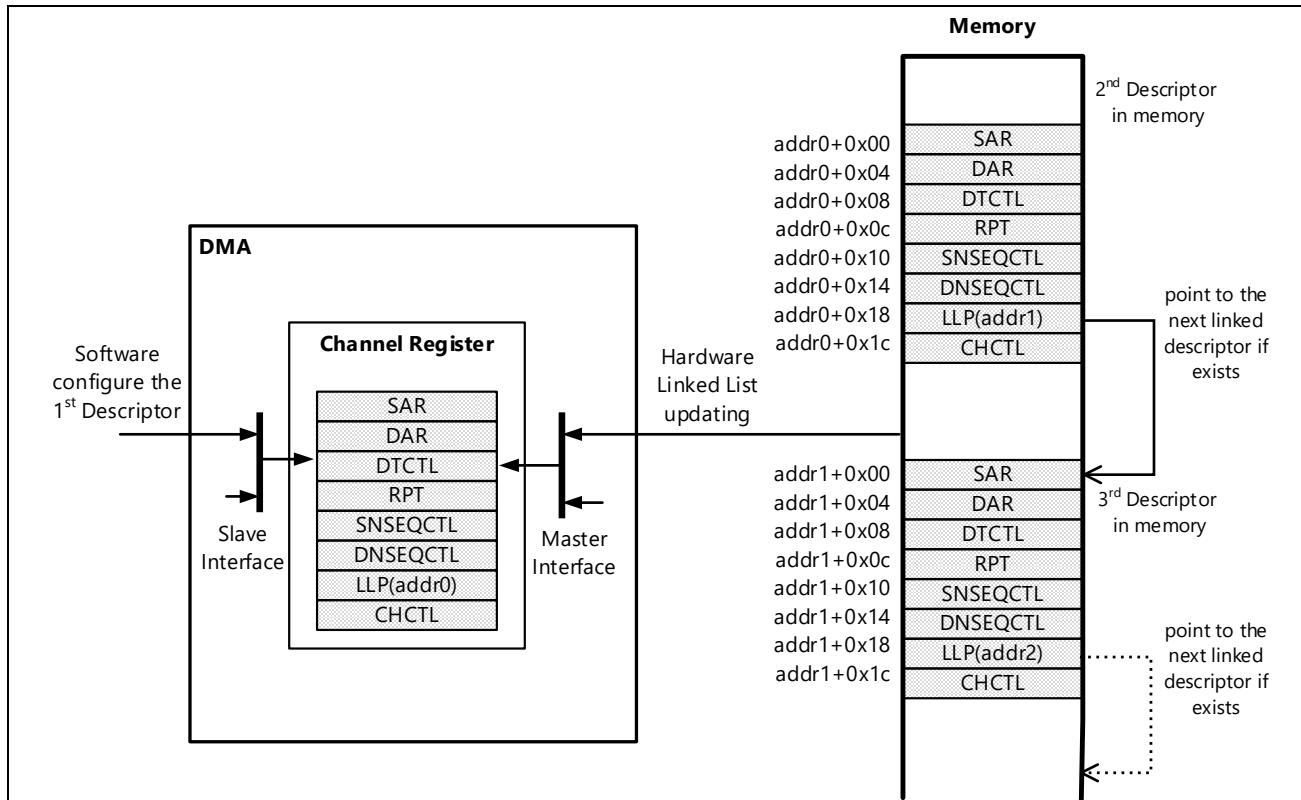


Figure 15-2 Linked list diagram

15.3.9 Nonsequence address mode

With nonsequence address mode, the source/destination address can jump to a discontinuous address at a specified offset after a certain amount of data is transmitted. Take the source address nonsequence mode as an example: First, set the channel control register DMA_CHCTLx.SNSEQEN to 1 to enable the nonsequence address mode. Write register DMA_CHCTLx.SINC to set the jump direction (jump forward or backward), and then write register DMA_SNSEQCTLx to specific jump condition and offset. The control of the destination address is similar to that of the source address, and the corresponding control registers are DMA_CHCTLx.DNSEQEN, DMA_CHCTLx.DINC, and DMA_DNSEQCTLx respectively. The transfer process is as shown in the following figure.

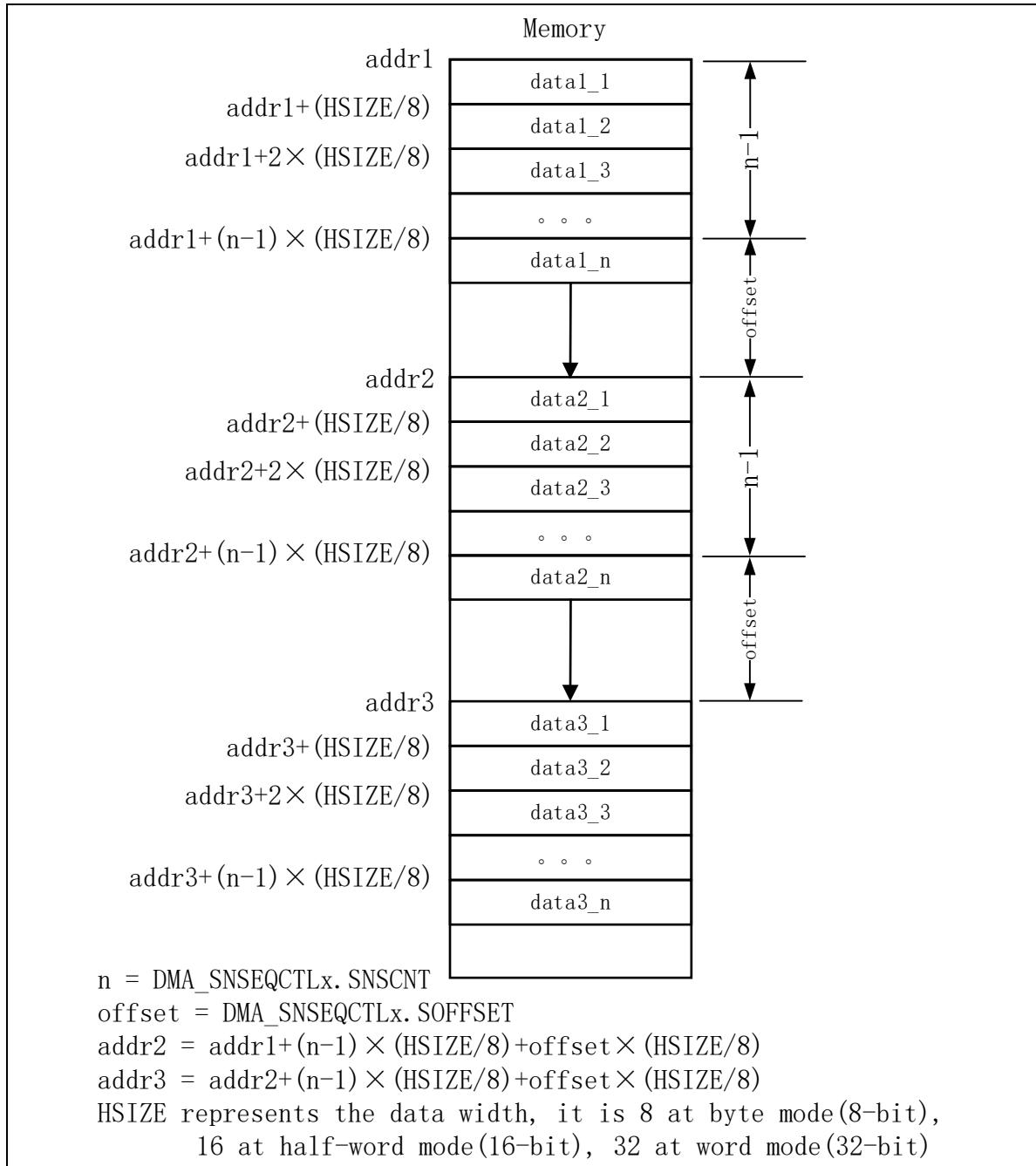


Figure 15-3 Example of source address nonsequence mode

15.3.10 Reconfigure transfer

The reconfigure transfer is an operation triggered by a DMA reconfigure request, it contains no data transfer, but updates the internal status registers of the corresponding channel for the next data transfer. Set register DMA_RCFGCTL.RCFGGEN to 1 to enable reconfigure transfer. The reconfigure request source is selected by the Trigger Source Selection Register DMA_TRGSELRC. When the selected reconfigure request source is input, the channel selected by the register DMA_RCFGCTL.RCFGCHS starts a reconfigure transfer.

There are three reconfigure modes: Linked list pointer (LLP) mode, nonsequence mode, and repeat mode.

When LLP mode is selected, the channel's descriptor and internal state are all updated with the new descriptor pointed by LLP. Subsequent transfer requests are executed according to the new descriptor.

When nonsequence mode or repeat mode is selected, the internal state of the channel is updated as described in the table below.

Table 15-1 Channel reset instructions

Reconfigure mode (register DMA_RCFGCTL)			Internal state of the channel		
CNTMD	SARMD	DARMD	Remaining transmission count	Source address of the next transmission	Destination address of the next transmission
0x0	arbitrary	arbitrary	Remain unchanged	Updated according to SARMD	Updated according to DARMD
0x1	0x0	arbitrary	Remain unchanged	Remain unchanged	Updated according to DARMD
	0x1		Update to the value after the next source address discontinuous jump occurs in normal state	Update to the first address of the next source address discontinuous transmission area	
	0x2, 0x3		Update to the value after the next source address overload occurs in the normal state	Update to the initial setting of register DMA_SARx	
0x2, 0x3	0x0	0x0	Remain unchanged	Updated according to SARMD	Remain unchanged
	arbitrary	0x1	Update to the value after the next target address discontinuous jump occurs in the normal state		Update to the first address of the next source address discontinuous transmission area
		0x2, 0x3	Update to the value after the next destination address overload occurs in the normal state		Update to the initial setting of register DMA_DARx

Note:

- When the reconfigure transfer function is active, this channel uses the registers DMA_RPTBx and DMA_SNSEQCTLBx, DMA_DNSEQCTLBx controls the address repeating

and nonsequence jumping. The registers DMA_RPTx and DMA_SNSEQCTLx, DMA_DNSEQCTLx are invalid.

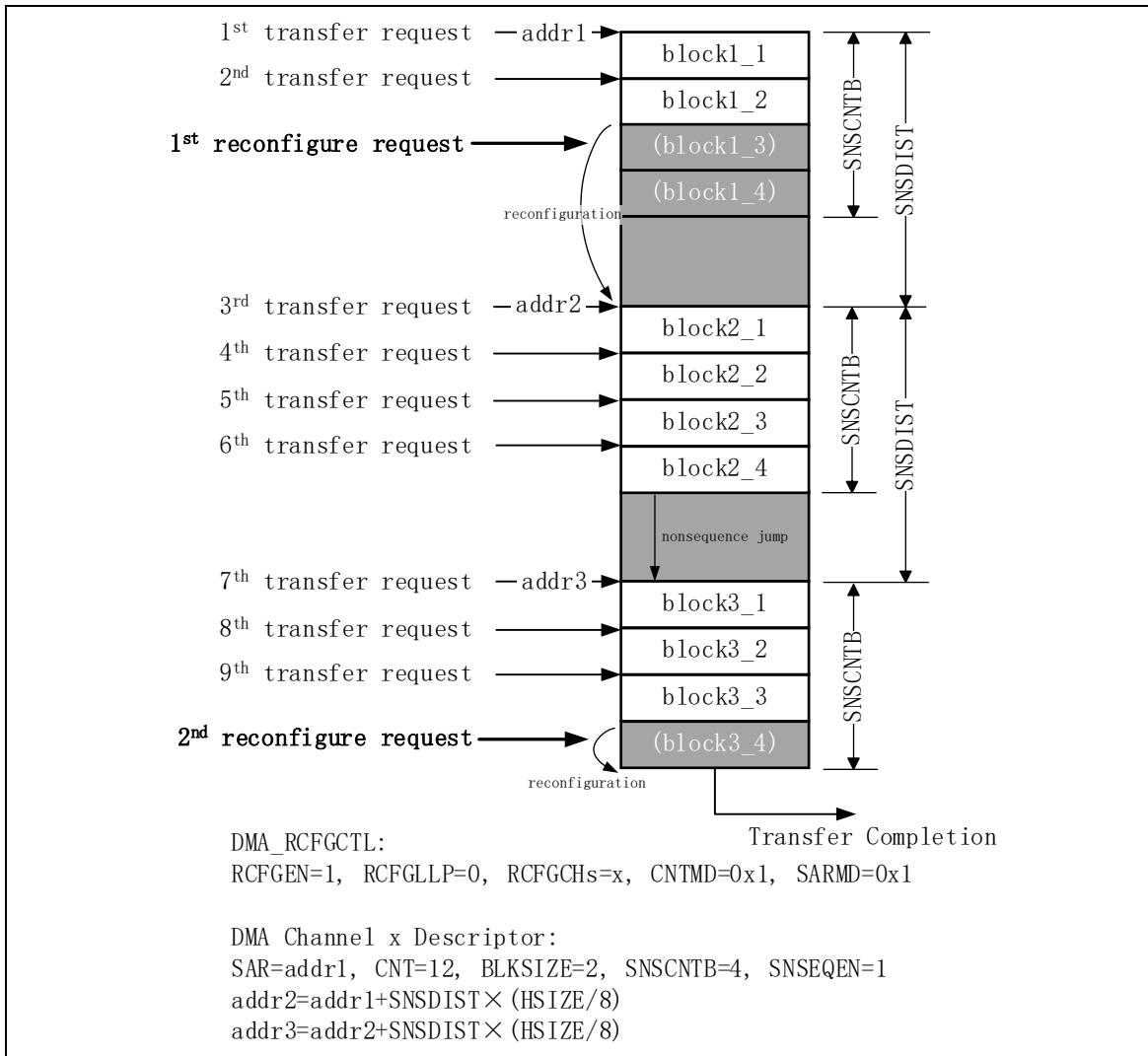


Figure 15-4 An example of the DMA reconfigure transfer

Figure 15-4 shows an example of reconfigure transfer. each transfer request starts the transfer of a data block. After the first reconfigure request is input, the controller skips the data blocks blk1_3 and blk1_4, and the source address is updated to the next nonsequence block base address, that is, addr2. After the second reconfigure transfer occurred, the transfer counter is updated to 0, that is, all data block transfers are completed, the channel enable bit is automatically cleared to 0, and a transfer completion (DMA_TCx) interrupt and event are generated.

15.3.11 Stop DMA transfer

During the transfer process, the Channel Enable Register DMA_CHEN.CHENx remains 1. In the case of Linked list transfer disabled, DMA_CHEN.CHENx automatically clears to 0 after all the number of transfers (specified by the Data Control Register DMA_DTCTLx.CNT) are completed. When LLP is enabled, DMA_CHEN.CHENx clears after all the transfers specified by the last linked descriptor are completed. During DMA transferring, if software write 1 to DMA_CHENCLR.CHENCLRx, the DMA will terminate the transfer after completing the current data read and write, and the DMA_CHEN.CHENx bit will be cleared to 0.

Note:

- The internal status of the breakpoint at software stop (write 1 to CHENCLR) is not saved, so when enable that channel again without modifying the channel configuration registers (descriptor), the entire stopped data block will be transferred again at the next transfer request.

15.4 Application examples

15.4.1 Memory-to-memory transfer

Target: Transfer 12 data from RAM address 0x20000000 to 0x20001000, data width is 32-bit.

1. Register setting

- Write 1 to DMA_EN.EN to enable DMA controller
- Select a channel, such as channel 0, and configure the channel descriptor registers:
 - Write 0x20000000 to DMA_SAR0, set the source address to RAM area
 - Write 0x20001000 to DMA_DAR0, set the destination address to RAM area
 - Write 0x00030004 to DMA_DTCTL0, set the size of the data block to 4 and the number of transfers to 3. After each transfer of one data block, a DMA_BTC0 interrupt is generated, and a DMA_TC0 interrupt is generated after 3 block transfers are completed.
 - Write 0x00000006 to DMA_RPT0, configure the source address repeat area size to 6, that is, reload the initial source address after completing the transfer of 6 data
 - Write 0x00001215 to DMA_CHCTL0:
 - * Disable linked list transfer
 - * Select increase mode for both source and destination address, and enable source address repeat mode
 - * Data width is word (32-bit)
 - * Enable interrupt
 - Configure DMA_TRGSEL0 to select software trigger as transfer request for DMA channel 0
 - Write 1 to DMA_CHEN.CHEN0 to enable channel 0
- Write 1 to Peripheral Event Software Triggers Register INTSFTTRG.SFTG of AOS module, sends the first software transfer request, DMA starts transferring

2. Transfer process

When software write 1 to INTSFTTRG.SFTG, the first transfer starts. After one data block (4 data) is transferred, the number of transfers DMA_MONDTCTL0.CNT is decremented by 1, and a block transfer completion interrupt is generated. The software can continue to write INTSFTTRG.SFTG in the interrupt subroutine to start the second transfer. In the second transfer, since the size of the source address repeat area is set to 6, the source address will be reloaded to the initial address 0x20000000 after transferring first 2 data, and continue to transfer the remaining 2 data. After the second transfer is completed, the transfer number DMA_MONDTCTL0.CNT is decremented by 1, and a block transfer completion interrupt is generated. The software can continue to write INTSFTTRG.SFTG in the interrupt subroutine to start the third transfer. After the third transfer is completed, the number of transfers DMA_MONDTCTL0.CNT is decremented to 0, that is, all transfers are completed, DMA

generates a block transfer completion interrupt and a transfer completion interrupt, and the channel enable bit DMA_CHEN.CHEN0 will be automatically cleared.

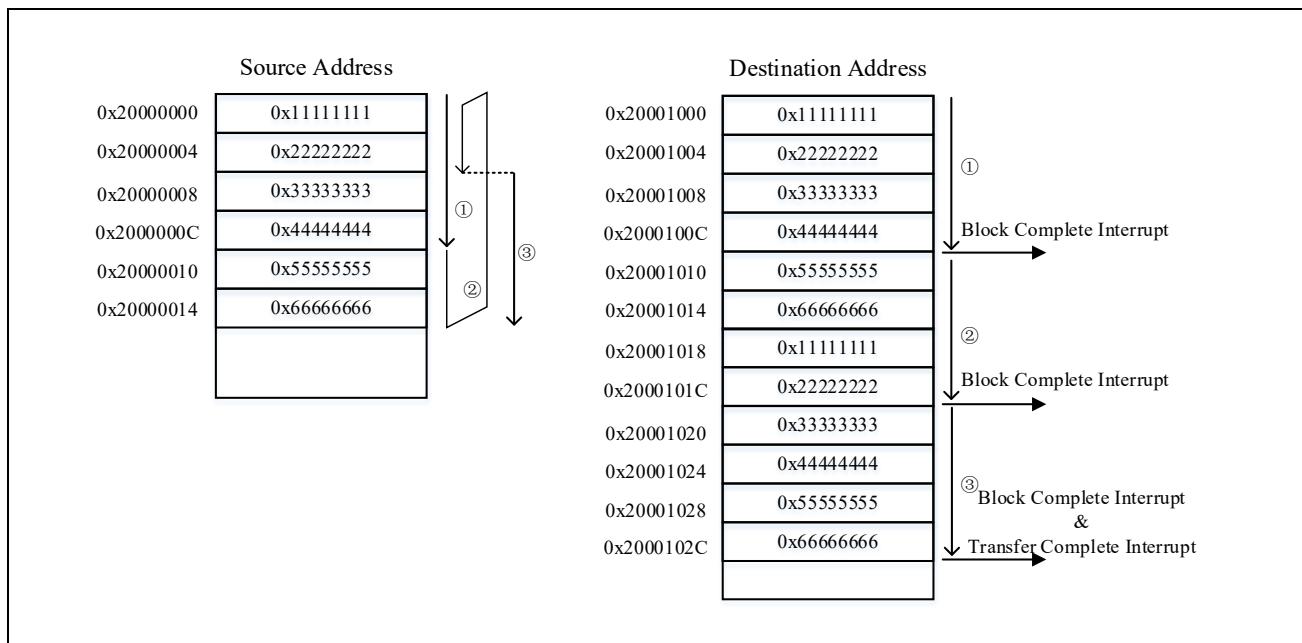


Figure 15-5 Application Example 1: Memory-to-Memory Transfer

15.4.2 Memory-to-peripheral transfer

Target: Transfer 10 16-bit data from the RAM address 0x20000000 to the transmit buffer register of the communication module. A DMA transfer request is generated each time the communication module sends out the data in the transmit buffer. When the last data is transferred, DMA generates a transfer complete interrupt.

1. Register setting

- Write 1 to DMA_EN.EN to enable DMA controller
- Configure the DMA_INTMSK1 register to mask the block transfer completion interrupt, and enable the transfer completion interrupt
- Select a channel, such as channel 0, and configure the channel descriptor registers:
 - Write 0x20000000 to DMA_SAR0, set the source address to RAM area
 - Write 0x40000000 to DMA_DAR0, set the destination address as the transmit buffer register address of peripheral module
 - Write 0x 000a0001 to DMA_DTCTL0, set the size of the data block to 1, the number of transfers to 10.
 - Write 0x00001101 to DMA_CHCTL0:
 - * Disable linked list transfer
 - * Select increment mode for source address, fixed mode for destination address
 - * Data width is half word (16-bit)
 - * Enable interrupt
 - Configure DMA_TRGSEL0, to select the transmit buffer empty event of the

communication module as the transfer request of DMA channel 0

- Write 1 to DMA_CHEN.CHEN0 to enable channel 0

2. Transfer process

After the channel is enabled, DMA waits for the transfer request from the communication module. After the transfer request is generated, the DMA transfers the data from the RAM to the transmit buffer register of the communication module and waits for the second transfer request from the communication module. Because the block transfer completion interrupt is masked, the DMA does not generate DMA_BTC0 interrupt. When all 10 data are transferred, the DMA generates a DMA_TC0 interrupt, and the channel enable bit DMA_CHEN.CHEN0 will be automatically cleared.

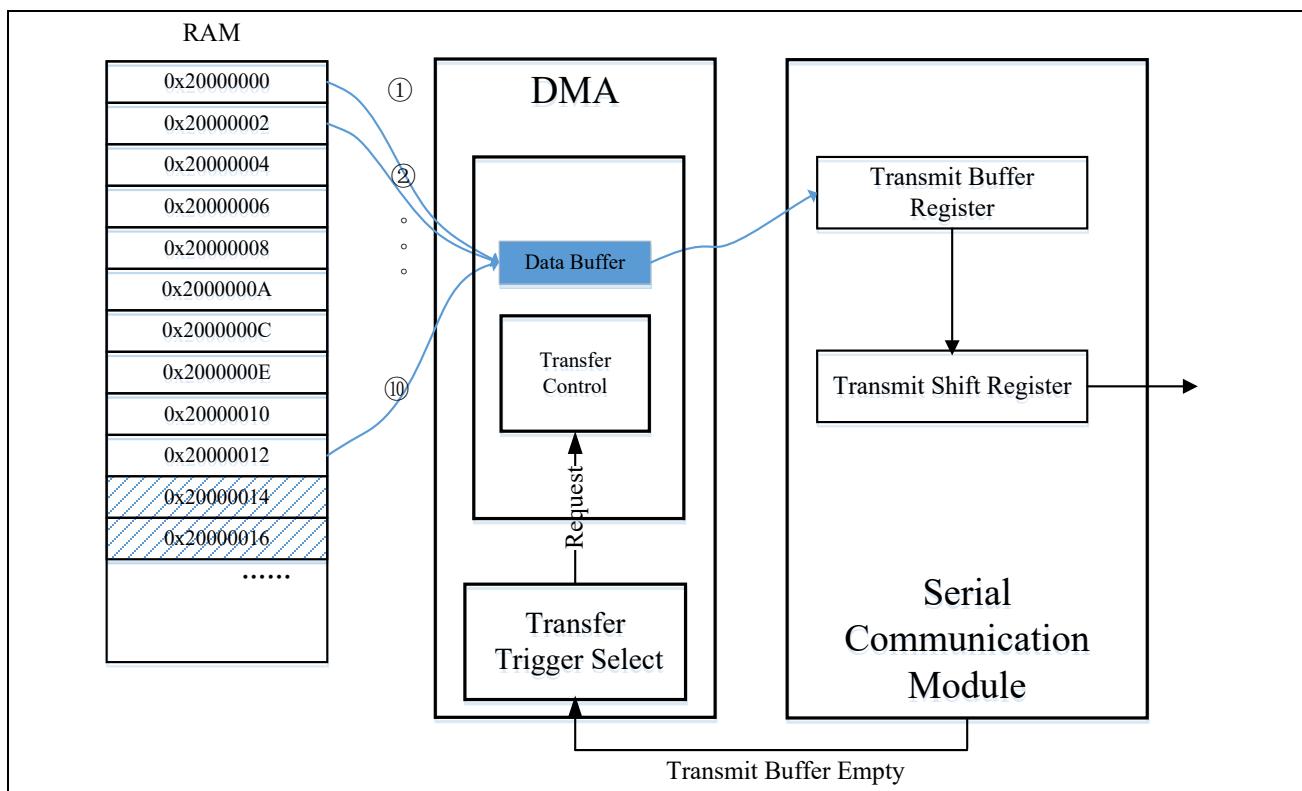


Figure 15-6 Application Example 2: Transfer from Memory to Peripheral module

15.4.3 Memory-to-memory with LLP enabled

1. Register setting

- Write 1 to DMA_EN.EN to enable DMA controller
- Select a channel, such as channel 0, and configure the channel descriptor registers (descriptor0):
 - Write 0x20000000 to DMA_SAR0, set the source address to RAM area
 - Write 0x20001000 to DMA_DAR0, set the destination address to RAM area
 - Write 0x0001000a to DMA_DTCTL0, set the size of data block to 10, the number of transfers to 1
 - Write 0x20002000 to DMA_LLPO, set the address of the second descriptor (descriptor1)

- Write 0x00001e05 to DMA_CHCTL0:
 - * Enable Linked list transfer
 - * Execute new transfer immediately after linked list descriptor loaded
 - * Select increment mode for source and destination address
 - * Data width is word (32-bit)
 - * Enable interrupt
- Configure the descriptor (descriptor1) for the second transfer in the 0x20002000 address of the RAM space, including:
 - Write 32-bit data 0x20000100 to address 0x20002000, which is the source address of the second transfer
 - Write 32-bit data 0x20001100 to address 0x20002004, which is the destination address of the second transfer
 - Write 32-bit data 0x00010014 to address 0x20002008, set the size of data block to 20, the number of transfers to 1
 - Write 32-bit data 0x20002020 to address 0x20002018, which is the address of the descriptor (descriptor2) of the third transmission
 - Write 0x00001d05 to address 0x2000201C:
 - * Enable Linked list transfer
 - * Execute new transfer immediately after linked list descriptor loaded
 - * Select increment mode for source and destination address
 - * Data width is half word (16-bit)
 - * Enable interrupt
- Configure the descriptor (descriptor2) of the third transfer in the 0x20002020 address of the RAM space, including:
 - Write 32-bit data 0x20000200 to address 0x20002020, which is the source address of the third transfer
 - Write 32-bit data 0x20001200 to address 0x20002024, which is the destination address of the third transfer
 - Write 32-bit data 0x00010028 to address 0x20002028, set the size of data block to 40
 - Write 32-bit data 0x0 to address 0x20002038, which is the address of the next descriptor, useless here because the next LLP is disabled.
 - Write 0x00001005 to address 0x2000203C:
 - * Disable Linked list transfer
 - * Select increment mode for source and destination address
 - * Data width is byte (8-bit)
 - * Enable interrupt
- Configure DMA_TRGSEL0, to select software trigger as the transfer request of DMA channel 0

- Write 1 to DMA_CHEN.CHEN0 to enable channel 0
 - Write 1 to AOS software triggers register INTSFTTRG.SFTG, sends a transfer request, DMA starts transferring
2. Transfer process

The software trigger starts DMA. After the first transfer is completed, the DMA reads descriptor 1 of the second transfer into the channel register. No DMA_BTC0 or DMA_TC0 interrupt is generated because the linked list transfer is enabled and immediately run mode is selected. The second transfer starts directly after the descriptor 1 is loaded into the channel register. After the second transfer is completed, descriptor 2 of the third transfer is loaded. The third transfer starts directly too. According to the configuration, this is the last transfer, no more linked descriptor exists. After the third transfer is completed, DMA_BTC0 and DMA_TC0 interrupts are generated because the interrupt is enabled, also, the channel enable bit DMA_CHEN.CHEN0 is automatically cleared.

15.5 Registers

15.5.1 Register list

Unit 1 BASE_ADDR: 0x40053000

Unit 2 BASE_ADDR: 0x40053400

Table 15-2 List of Registers

Register name	Symbol	Offset address	Bit width	Reset value
DMA enable register	DMA_EN	0x00	32	0x00000000
Interrupt state register 0	DMA_INTSTAT0	0x04	32	0x00000000
Interrupt state register 1	DMA_INTSTAT1	0x08	32	0x00000000
Interrupt mask register 0	DMA_INTMASK0	0x0C	32	0x00000000
Interrupt mask register 1	DMA_INTMASK1	0x10	32	0x00000000
Interrupt clear register 0	DMA_INTCLR0	0x14	32	0x00000000
Interrupt clear register 1	DMA_INTCLR1	0x18	32	0x00000000
Channel enable register	DMA_CHEN	0x1c	32	0x00000000
Channel enable clear register	DMA_CHENCLR	0x34	32	0x00000000
Transfer request status register	DMA_REQSTAT	0x20	32	0x00000000
Transfer status monitor register	DMA_CHSTAT	0x24	32	0x00000000
Reconfigure control register	DMA_RCFGCTL	0x2c	32	0x00000000
Source address register	DMA_SARx *1	0x40+0x40*x	32	0x00000000
Destination address register	DMA_DARx	0x44+0x40*x	32	0x00000000
Data control register	DMA_DTCTLx	0x48+0x40*x	32	0x00000001
Repeat control register	DMA_RPTx	0x4C+0x40*x	32	0x00000000
Repeat control register B	DMA_RPTBx			
Source address nonsequence control register	DMA_SNSEQCTLx	0x50+0x40*x	32	0x00000000
Source address nonsequence control register B	DMA_SNSEQCTLBx			
Destination address nonsequence control register	DMA_DNSEQCTLx	0x54+0x40*x	32	0x00000000
Destination address nonsequence control register B	DMA_DNSEQCTBx			
Linked list pointer register	DMA_LLpx	0x58+0x40*x	32	0x00000000
Channel control register	DMA_CHCTLx	0x5C+0x40*x	32	0x00001000
Source address monitor register	DMA_MONSARx	0x60+0x40*x	32	0x00000000
Destination address monitor register	DMA_MONDARx	0x64+0x40*x	32	0x00000000
Data control monitor register	DMA_MONDTCTLx	0x68+0x40*x	32	0x00000001
Repeat counter monitor register	DMA_MONRPTx	0x6C+0x40*x	32	0x00000000
Source nonsequence counter monitor register	DMA_MONSNSEQCTLx	0x70+0x40*x	32	0x00000000
Destination nonsequence counter monitor register	DMA_MONDNSEQCTLx	0x74+0x40*x	32	0x00000000

Register name	Symbol	Address	Bit width	Reset value
DMA1 transfer request selection register	DMA1_TRGSELx	0x40010814+0x4*x	32	0x000001ff
DMA2 transfer request selection register	DMA2_TRGSELx	0x40010834+0x4*x	32	0x000001ff
DMA1,2 shared reconfigure request select register	DMA_TRGSELRC	0x40010854	32	0x000001ff

Note *1: x=0-7

15.5.2 DMA Enable Register (DMA_EN)

Reset value: 0x00000000

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

Bit	Symbol	Bit name	Description	Read and write
b31-b1	Reserved	-	Read as "0", write as "0"	R/W
b0	EN	DMA enable bit	0: Disable DMA 1: Enable DMA	R/W

15.5.3 Interrupt State Register 0 (DMA_INTSTAT0)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16								
-	-	-	-	-	-	-	-	-	REQERR[7:0]														

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0								
-	-	-	-	-	-	-	-	-	TRNERR[7:0]														

Bit	Symbol	Bit name	Description	Read and write
b31-b24	Reserved	-	Read as "0", write as "0"	R/W
b23-b16	REQERR[7:0]	Request overflow error flag	0: No transfer request overflow error occurred on the channel 1: The transfer request overflow error occurred on the channel, that is, the transfer request input again when the previous request has not been processed yet	R
b15-b8	Reserved	-	Read as "0", write as "0"	R/W
b7-b0	TRNERR[7:0]	Transfer error flag	0: No transfer error occurred on the channel 1: Transfer error occur on the channel	R

15.5.4 Interrupt State Register 1 (DMA_INTSTAT1)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	BTC[7:0]

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	TC[7:0]

Bit	Symbol	Bit name	Description	Read and write
b31-b24	Reserved	-	Read as "0", write as "0"	R/W
b23-b16	BTC[7:0]	Block transfer completion flag	The flag sets after a block of data is transferred 0: No block transfer completion occurred on the channel 1: Block transfer completion occurred on the channel	R
b15-b8	Reserved	-	Read as "0", write as "0"	R/W
b7-b0	TC[7:0]	Transfer completion flag	The flag sets after the number of transfers set by register DMA_DTCTLx.CNT are all completed 0: No transfer completion occurred on the channel 1: Transfer completion occurred on the channel	R

15.5.5 Interrupt Mask Register 0 (DMA_INTMASK0)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MSKREQERR[7:0]

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MSKTRNERR[7:0]

Bit	Symbol	Bit name	Description	Read and write
b31-b24	Reserved	-	Read as "0", write as "0"	R/W
b23-b16	MSKREQERR[7:0]	Transfer request overflow error interrupt mask	0: Enable the generation of transfer request overflow interrupt 1: Disable the generation of transfer request overflow interrupt	R/W
b15-b8	Reserved	-	Read as "0", write as "0"	R/W
b7-b0	MSKTRNERR[7:0]	Transfer error interrupt mask	0: Enable the generation of transfer error interrupt 1: Disable the generation of transfer error interrupt	R/W

15.5.6 Interrupt Mask Register 1 (DMA_INTMASK1)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MSKBTC[7:0]

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MSKTC[7:0]

Bit	Symbol	Bit name	Description	Read and write
b31-b24	Reserved	-	Read as "0", write as "0"	R/W
b23-b16	MSKBTC[7:0]	Block transfer completion interrupt mask	0: Enable the generation of block transfer completion interrupt 1: Disable the generation of block transfer completion interrupt	R/W
b15-b8	Reserved	-	Read as "0", write as "0"	R/W
b7-b0	MSKTC[7:0]	Transfer completion interrupt mask	0: Enable the generation of transfer completion interrupt 1: Disable the generation of transfer completion interrupt	R/W

15.5.7 Interrupt Clear Register 0 (DMA_INTCLR0)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CLRREQERR[7:0]

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CLRTRNERR[7:0]

Bit	Symbol	Bit name	Description	Read and write
b31-b24	Reserved	-	Read as "0", write as "0"	R/W
b23-b16	CLRREQERR[7:0]	Clear transfer request overflow error flag	Write 0: Useless. Write 1: Clear bit DMA_INTSTAT0.REQERR. Read always as "0"	W
b15-b8	Reserved	-	Read as "0", write as "0"	R/W
b7-b0	CLRTRNERR[7:0]	Clear transfer error flag	Write 0: Useless. Write 1: Clear bit DMA_INTSTAT0.TRNERR. Read always as "0"	W

15.5.8 Interrupt Clear Register 1 (DMA_INTCLR1)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CLRBTC[7:0]

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CLRTC[7:0]

Bit	Symbol	Bit name	Description	Read and write
b31-b24	Reserved	-	Read as "0", write as "0"	R/W
b23-b16	CLRBTC[7:0]	Clear block transfer completion flag	Write 0: Useless. Write 1: Clear bit DMA_INTSTAT1.BTC. Read always as "0"	W
b15-b8	Reserved	-	Read as "0", write as "0"	R/W
b7-b0	CLRTC[7:0]	Clear transfer completion flag	Write 0: Useless. Write 1: Clear bit DMA_INTSTAT1.TC. Read always as "0"	W

15.5.9 Channel Enable Register (DMA_CHEN)

Reset value: 0x00000000

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[7:0]

Bit	Symbol	Bit name	Description	Read and write
b31-b8	Reserved	-	Read as "0", write as "0"	R/W
b7-b0	CHEN[7:0]	Channel enable bit	0: The channel is disabled 1: The channel is enabled Write 1 to CHEN[x] and set CHEN[x] to 1. Writing 0 to CHEN[x] has no effect. When writing 1 to the DMA_CHENCLR.CHENCLR[x] bit, CHEN[x] clears 0 and stops the ongoing transfer of channel x. The enable bit remains 1 during the transfer process, and will be automatically cleared to 0 after the number of transfers specified by register DMA_DTCTLx.CNT are completed. If DMA_DTCTLx.CNT is set to 0, it means an infinite transfer, CHEN[x] will not be cleared automatically and no DMA_TCx is generated. Note: Please write 1 to CHEN[x] to enable channel x after all 8 descriptor registers of this channel are set. When CHEN[x] is 1, the write operation to the descriptor registers of channel x is useless.	R/W

15.5.10 Channel Enable Clear Register (DMA_CHENCLR)

Reset value: 0x00000000

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
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CHENCLR[7:0]

Bit	Symbol	Bit name	Description	Read and write
b31-b8	Reserved	-	Read as "0", write as "0"	R/W
b7-b0	CHENCLR[7:0]	Clear channel enable bit	Write 0: Useless. Write 1: Clear bit DMA_CHEN.CHEN. Read always as "0" When write 1 to CHENCLR to a running channel, the transfer is stopped after the current data read/write is completed.	R/W

15.5.11 Reconfigure Control Register (DMA_RCFGCTL)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	CNTMD[1:0]	DARMD[1:0]	SARMD[1:0]			

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-	-	-	-		RCFGCHS[3:0]			-	-	-	-	-	-	RCFGLP	RCFGEN

Bit	Symbol	Bit name	Description	Read and write
b31-b22	Reserved	-	Read as "0", write as "0"	R/W
b21-b20	CNTMD[1:0]	Transfer counter reconfigure mode	0x0: Fixed mode 0x1: According to source address reconfigure mode, transfer counter subtracts the value of source nonsequence counter or repeat counter. 0x2, 0x3: According to destination address reconfigure mode, transaction counter subtracts the value of destination nonsequence counter or repeat counter.	R/W
b19-b18	DARMD[1:0]	Destination address reconfigure mode	0x0: Fixed mode 0x1: Nonsequence mode Update the destination address to the next nonsequence block base address. Only valid when the corresponding bit DMA_CHCTLx.DNSEQEN is 1. addr_base_next = addr_base + (DNSDIST x HSIZE(bit)/8) *: "addr_base" represents the current nonsequence block base address. 0x2, 0x3: Repeat mode Update the destination address to the initial value of DMA_DARx. Only valid when the corresponding bit DMA_CHCTLx.DRPTEN is1.	R/W
b17-b16	SARMD[1:0]	Source address reconfigure mode	0x0: Fixed mode 0x1: Nonsequence mode Update the source address to the next nonsequence block base address. Only valid when the corresponding bit DMA_CHCTLx.SNSEQEN is 1. addr_base_next = addr_base + (SNSDIST x HSIZE(bit)/8) *: "addr_base" represents the current nonsequence block base address. 0x2, 0x3: Repeat mode Update the source address to the initial value of DMA_SARx. Only valid when the corresponding bit DMA_CHCTLx.SRPTEN is1.	R/W
b15-b12	Reserved	-	Read as "0", write as "0"	R/W
b11-b8	RCFGCHS[3:0]	Reconfigure channel selection	0x0: Channel 0 0x1: Channel 1 ... 0x7: Channel 7 other: Reserved, setting prohibited	R/W
b7-b2	Reserved	-	Read as "0", write as "0"	R/W
b1	RCFGLLP	Enable Linked list reconfigure mode	0: Disable Linked list reconfigure mode 1: Enable Linked list reconfigure mode Note: When RCFGLLP is set to 1, the channel will reload the new descriptor in the memory, so the bits 16-25 of this register are useless.	R/W
b0	RCFGEN	Enable channel reconfigure	0: Disable reconfigure transfer 1: Enable reconfigure transfer	R/W

Note:

- Please set this register when DMA_EN.EN is 0. This register must be set before the first transfer of the selected reconfigure channel.

15.5.12 Transfer Request Status Register (DMA_REQSTAT)

Reset value: 0x00000000

b31	b30	b29	b28	b27	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[7:0]							
<hr/>															
Bit	Symbol	Bit name		Description								Read and write			
b31-b17	Reserved	-		Read as "0", write as "0"								R/W			
b16	RCFGREQ	Reconfigure request flag		This bit sets to 1 when a reconfigure request is input, it clears to 0 after the reconfigure transfer starts, or when reconfigure enable bit (DMA_RCFGCTL.RCFGEN) is written to 0. 0: No reconfigure request is waiting for processing 1: Reconfigure request is waiting for processing								R			
b15-b8	Reserved	-		Read as "0", write as "0"								R/W			
b7~b0	CHREQ[7:0]	Transfer request flag		Each bit corresponds to a channel. This bit sets to 1 when a transfer request is input, it clears to 0 after the channel transfer is started, or when the transfer enable bit (DMAEN or CHEN[x]) is cleared to 0. When this bit is 1 and the channel transfer request is input again, a transfer request overflow error occurs, and the second request is ignored, but this flag remains 1 and continues to wait for transfer. 0: No transfer request is waiting for processing for this channel 1: Transfer request is waiting for processing for this channel								R			

15.5.13 Transfer Status Monitor Register (DMA_CHSTAT)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
-	-	-	-	-	-	-	-	-	CHACT[7:0]							
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	RCFGACT	DMAACT	
<hr/>																
Bit	Symbol	Bit name		Description								Read and write				
b31-b24	Reserved	-		Read as "0", write as "0"								R/W				
b23-b16	CHACT[7:0]	Channel active flag		Each bit corresponds to a channel. 0: The channel is idle 1: The channel is transferring								R				
b15-b2	Reserved	-		Read as "0", write as "0"								R/W				
b1	RCFGACT	Reconfigure active flag		0: No reconfigure move is processing 1: Reconfigure move is processing								R				
b0	DMAACT	DMA active flag		0: DMA is not transferring 1: DMA is transferring								R				

15.5.14 Transfer Request Selection Register (DMA_TRGSELx) (x=0~7)

Reset value: 0x0000001ff

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
TRGSEL[8:0]															
Bit	Symbol	Bit name	Description												Read and write
b31	COMEN[1]	public trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger channel transfer 1: Allow the common trigger event of AOS_COMTRG2 to trigger channel transfer For common trigger events, please refer to the AOS chapter												R/W
b30	COMEN[0]	public trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to trigger channel transfer 1: Allow the common trigger event of AOS_COMTRG1 to trigger channel transfer For common trigger events, please refer to the AOS chapter												R/W
b29-b9	Reserved	-	Read as "0", write as "0"												R/W
b8-b0	TRGSEL[8:0]	Trigger source selection	Select the event number to start the corresponding channel for transfer. For specific numbers, please refer to [Note:].												R/W

15.5.15 Reconfigure Request Selection Register (DMA_TRGSELRC)

Reset value: 0x0000001ff

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
COMEN[1:0]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
TRGSEL[8:0]															
Bit	Symbol	Bit name	Description												Read and write
b31	COMEN[1]	Public trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger channel reconfigure 1: Allow the common trigger event of AOS_COMTRG2 to trigger channel reconfigure For common trigger events, please refer to the AOS chapter												R/W
b30	COMEN[0]	Public trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to trigger channel reconfigure 1: Allow the common trigger event of AOS_COMTRG1 to trigger a channel reconfigure For common trigger events, please refer to the AOS chapter												R/W
b29-b9	Reserved	-	Read as "0", write as "0"												R/W
b8-b0	TRGSEL[8:0]	Trigger source selection	Select the event number that triggers the channel to reconfigure. For specific numbers, please refer to [Note:]. DMA_1, DMA_2 share the same reconfigure request.												R/W

15.5.16 Source Address Register (DMA_SARx) (x=0~7)

Reset value: 0x00000000

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]															
Bit	Symbol	Bit name	Description										Read and write		
b31-b0	SAR[31:0]	Source address	Set source address Note: – When DMA_CHCTLx.HSIZE=0x1, the data width is 16-bit, SAR[0] is useless. When DMA_CHCTLx.HSIZE=0x2 or 0x3, the data width is 32-bit, SAR[1:0] is useless.										R/W		

15.5.17 Transfer Target Address Register (DMA_DARx) (x=0~7)

Reset value: 0x00000000

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]															
Bit	Symbol	Bit name	Description										Read and write		
b31-b0	DAR[31:0]	Destination address	Set destination address Note: – When DMA_CHCTLx.HSIZE=0x1, the data width is 16-bit, DAR[0] is useless. When DMA_CHCTLx.HSIZE=0x2 or 0x3, the data width is 32-bit, DAR[1:0] is useless.										R/W		

15.5.18 Data Control Register (DMA_DTCTLx) (x=0~7)

Reset value: 0x00000001

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]

Bit	Symbol	Bit name	Description	Read and write
b31-b16	CNT[15:0]	Number of transfers	The total number of transfers, each time a request starts the transfer of a data block, the transfer count is decremented by 1 when it is completed, and a transfer completion interrupt occurs when it decreases to 0. If it is set to 0, it means infinite transfer, that is, each transfer request start DMA to transfer a data block as normal, the transfer counter will remain unchanged as 0 when block transfer is completed, and no transfer completion interrupt will be generated.	R/W
b15-b10	Reserved	-	Read as "0", write as "0"	R/W
b9-b0	BLKSIZE[9:0]	Size of data block	Set the size of the data block, up to 1024 data can be configured. The width of each data is determined by the HSIZE bits in the DMA_CHCTLx 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

15.5.19 Repeat Control Register (DMA_RPTx) ($x=0\sim7$)

Reset value: 0x00000000

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]

Bit	Symbol	Bit name	Description	Read and write
b31-b26	Reserved	-	Read as "0", write as "0"	R/W
b25-b16	DRPT[9:0]	The number of data before destination address to reload	Set the number of data before destination address reloads the initial value of DMA_DARx. The destination repeat counter decrements by 1 after each data is transferred, when repeat counter decreases to 0, the address repeat occurs. When the register is set to 10, the address is reloaded after every 10 data transferred, and when it is set to 0, the address is reloaded after every 1024 data transferred. The width of each data is determined by the HSIZE bits in the DMA_CHCTLx register.	R/W
b15-b10	Reserved	-	Read as "0", write as "0"	R/W
b9-b0	SRPT[9:0]	The number of data before source address to reload	Set the number of data before source address reloads the initial value of DMA_SARx. The source repeat counter decrements by 1 after each data is transferred, when repeat counter decreases to 0, the address repeat occurs. When the register is set to 10, the address is reloaded after every 10 data transferred, and when it is set to 0, the address is reloaded after every 1024 data transferred. The width of each data is determined by the HSIZE bits in the DMA_CHCTLx register.	R/W

This register configures the size of the source and destination address repeat area. To use address repeat mode, it is necessary to write 1 to SRPTEN/DRPREN bits of the DMA_CHCTLx register, and configure the SINC/DINC bits of the DMA_CHCTLx register to make the address update method to increment or decrement mode. If fixed mode is selected, the address repeat mode is invalid.

This register is invalid when the reconfigure transfer is enabled for this channel.

15.5.20 Repeat Control Register B (DMA_RPTBx) (x=0~7)

Reset value: 0x00000000

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]

Bit	Symbol	Bit name	Description	Read and write
b31-b26	Reserved	-	Read as "0", write as "0"	R/W
b25-b16	DRPTB[9:0]	The number of data blocks before destination address to reload	Set the number of data blocks before destination address reloads the initial value of DMA_DARx. The destination repeat counter decrements by 1 after each data block is transferred, when repeat counter decreases to 0, the address repeat occurs. The data block size is determined by DMA_DTCTLx.BLKSIZE and DMA_CHCTLx.HSIZE.	R/W
b15-b10	Reserved	-	Read as "0", write as "0"	R/W
b9-b0	SRPTB[9:0]	The number of data blocks before source address to reload	Set the number of data blocks before source address reloads the initial value of DMA_SARx. The source repeat counter decrements by 1 after each data block is transferred, when repeat counter decreases to 0, the address repeat occurs. The data block size is determined by DMA_DTCTLx.BLKSIZE and DMA_CHCTLx.HSIZE.	R/W

This register configures the size of the source and destination address repeat area. To use address repeat mode, it is necessary to write 1 to SRPTEN/DRPREN bits of the DMA_CHCTLx register, and configure the SINC/DINC bits of the DMA_CHCTLx register to make the address update method to increment or decrement mode. If fixed mode is selected, the address repeat mode is invalid.

This register is only valid when the reconfigure transfer is enabled for this channel, and it replaces the register DMA_RPTx. It has no effect when the reconfigure transfer is disabled.

15.5.21 Source Address Nonsequence Control Cegister (DMA_SNSEQCTLx) ($x=0\sim7$)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
SNSCNT[11:0]												SOFFSET[19:16]				
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0	
SOFFSET[15:0]																
Bit	Symbol	Bit name	Description										Read and write			
b31-b20	SNSCNT[11:0]	The number of data before source address to jump	Set the number of data before source address to jump. The address after jump is calculated by adding or subtracting an offset (specified by field SOFFSET) from the current address. The source nonsequence counter decrements by 1 after each data is transferred, when nonsequence counter decreases to 0, the address jump occurs. When the register is set to 10, the address jumps after every 10 data transferred, and when it is set to 0, the address jumps after every 4096 data transferred. The width of each data is determined by register DMA_CHCTLx.HSIZE										R/W			
b19-b0	SOFFSET[19:0]	The offset for source address to jump	The offset is relative to the current address, which is the last transfer address before the jump. The jump direction is determined by the register DMA_CHCTLx.SINC. refer to Figure 15-3. if DMA_CHCTLx.SINC=0x1, $Current_Source_Address + SOFFSET \times (HSIZE(bit)/8)$ if DMA_CHCTLx.SINC=0x2 or 0x3, $Current_Source_Address - SOFFSET \times (HSIZE(bit)/8)$										R/W			

To use the source address nonsequence mode, it is necessary to write 1 to SNSQEN bit of the DMA_CHCTLx register, and configure the SINC bit of the DMA_CHCTLx register to make the address update method to increment or decrement mode. If fixed mode is selected, the address nonsequence mode is invalid.

This register is invalid when the reconfigure transfer is enabled for this channel.

15.5.22 Source Address Nonsequence Control Register B (DMA_SNSEQCTLBx) (x=0~7)

Reset value: 0x00000000

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]																
Bit	Symbol	Bit name	Description												Read and write	
b31-b20	SNSCNTB[11:0]	The number of data block before source address to jump	Set the number of data block before source address to jump. The address after jump is calculated by adding or subtracting an offset (specified by field SNSDIST) from the current nonsequence block base address. The source nonsequence counter decrements by 1 after each data block is transferred, when nonsequence counter decreases to 0, the address jump occurs. When the register is set to 10, the address jumps after every 10 data blocks transferred, and when it is set to 0, the address jumps after every 4096 data block transferred. The data block size is determined by DMA_DTCTLx.BLKSIZE and DMA_CHCTLx.HSIZE.												R/W	
b19-b0	SNSDIST[19:0]	The offset for source address to jump	The offset is relative to the current source nonsequence block base address, which is the address when nonsequence counter loads the SNSCNTB. The jump direction is determined by the register DMA_CHCTLx.SINC. refer to Figure 15-4 if DMA_CHCTLx.SINC=0x1, $\text{Current_Source_Nonsequence_Block_Base_Address} + \text{SNSDIST} \times (\text{HSIZE(bit)/8})$ if DMA_CHCTLx.SINC=0x2 or 0x3, $\text{Current_Source_Nonsequence_Block_Base_Address} - \text{SNSDIST} \times (\text{HSIZE(bit)/8})$												R/W	

To use the source address nonsequence mode, it is necessary to write 1 to SNSEQEN bit of the DMA_CHCTLx register, and configure the SINC bit of the DMA_CHCTLx register to make the address update method to increment or decrement mode. If fixed mode is selected, the address nonsequence mode is invalid.

This register is only valid when the reconfigure transfer is enabled for this channel, and it replaces the register DMA_SNSEQCTLx. It has no effect when the reconfigure transfer is disabled.

15.5.23 Destination Address Nonsequence Control Register (DMA_DNSEQCTLx) (x=0~7)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	DNSCNT[11:0]		DOFFSET[19:16]	
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0	DOFFSET[15:0]			
<hr/>																			
<hr/>																			
Bit	Symbol	Bit name	Description	Read and write															
b31-b20	DNSCNT[11:0]	The number of data before destination address to jump	Set the number of data before destination address to jump. The address after jump is calculated by adding or subtracting an offset (specified by field DOFFSET) from the current address. The destination nonsequence counter decrements by 1 after each data is transferred, when nonsequence counter decreases to 0, the address jump occurs. When the register is set to 10, the address jumps after every 10 data transferred, and when it is set to 0, the address jumps after every 4096 data transferred. The width of each data is determined by register DMA_CHCTLx.HSIZE	R/W															
b9-b0	DOFFSET[19:0]	The offset for destination address to jump	The offset is relative to the current address, which is the last transfer address before the jump. The jump direction is determined by the register DMA_CHCTLx.DINC. refer to Figure 15-3. if DMA_CHCTLx.DINC=0x1, <i>Current_Destination_Address</i> + DOFFSET × (HSIZE(bit)/8) if DMA_CHCTLx.DINC=0x2 or 0x3, <i>Current_Destination_Address</i> - DOFFSET × (HSIZE(bit)/8)	R/W															

To use the destination address nonsequence mode, it is necessary to write 1 to DNSEQEN bit of the DMA_CHCTLx register, and configure the DINC bit of the DMA_CHCTLx register to make the address update method to increment or decrement mode. If fixed mode is selected, the address nonsequence mode is invalid.

This register is invalid when the reconfigure transfer is enabled for this channel.

15.5.24 Destination Address Nonsequence control register B (DMA_DNSEQCTLBx) (x=0~7)

Reset value: 0x00000000

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]															
Bit	Symbol	Bit name	Description										Read and write		
b31-b20	DNSCNTB[11:0]	The number of data block before destination address to jump	Set the number of data block before destination address to jump. The address after jump is calculated by adding or subtracting an offset (specified by field DNSDIST) from the current nonsequence block base address. The destination nonsequence counter decrements by 1 after each data block is transferred, when nonsequence counter decreases to 0, the address jump occurs. When the register is set to 10, the address jumps after every 10 data blocks transferred, and when it is set to 0, the address jumps after every 4096 data block transferred. The data block size is determined by DMA_DTCTLx.BLKSIZE and DMA_CHCTLx.HSIZE.										R/W		
b19-b0	DNSDIST[19:0]	The offset for destination address to jump	The offset is relative to the current destination nonsequence block base address, which is the address when nonsequence counter loads the DNSCNTB. The jump direction is determined by the register DMA_CHCTLx.DINC. refer to Figure 15-4 if DMA_CHCTLx.DINC=0x1, $\text{Current Destination Nonsequence Block Base Address} + \text{DNSDIST} \times (\text{HSIZE(bit)} / 8)$ if DMA_CHCTLx.DINC=0x2 or 0x3, $\text{Current Destination Nonsequence Block Base Address} - \text{DNSDIST} \times (\text{HSIZE(bit)} / 8)$										R/W		

To use the destination address nonsequence mode, it is necessary to write 1 to SNSEQEN bit of the DMA_CHCTLx register, and configure the SINC bit of the DMA_CHCTLx register to make the address update method to increment or decrement mode. If fixed mode is selected, the address nonsequence mode is invalid.

This register is only valid when the reconfigure transfer is enabled for this channel, and it replaces the register DMA_SNSEQCTLx. It has no effect when the reconfigure transfer is disabled.

15.5.25 Linked List Pointer Register (DMA_LLPtr_x) ($x=0\sim7$)

Reset value: 0x00000000

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]															
Bit	Symbol	Bit name	Description										Read and write		
b31-b2	LLP[31:2]	Linked list pointer	The address of the linked descriptor. Low 2 bits are 0 fixed, 32-bit aligned.										R/W		
b1-b0	Reserved	-	Read as "0", write as "0"										R		

15.5.26 Channel Control Register (DMA_CHCTL_x) ($x=0\sim7$)

Reset value: 0x00001000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b199	b188	b177	b166
-															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-															
Bit	Symbol	Bit name	Description										Read and write		
b31-b13	Reserved	-	Read as "0", write as "0"										R/W		
b12	IE	Interrupt enable bit	Configure whether this channel generates an interrupt. 0: Disable this channel to generate interrupt 1: Enable this channel to generate interrupt										R/W		
b11	LLPRUN	Linked list transfer immediately run	Configure whether to start the transfer immediately after loaded the new descriptor. This field is valid only when LLPEN is 1. 0: Don not execute the new transfer immediately after loaded the linked list channel descriptor. 1: Execute the new transfer immediately after loaded the linked list channel descriptor.										R/W		
b10	LLPEN	Linked list enable	0: Disable linked list transfer 1: Enable linked list transfer										R/W		
b9-b8	HSIZE[1:0]	Width of data	0x0: 8-bit 0x1: 16-bit 0x2, 0x3: 32-bit										R/W		
B7	DNSEQEN	Destination address nonsequence mode enable	0: Disable destination address nonsequence mode 1: Enable destination address nonsequence mode										R/W		
b6	SNSEQEN	Source address nonsequence mode enable	0: Disable source address nonsequence mode 1: Enable source address nonsequence mode										R/W		
b5	DRPTEN	Destination address repeat mode enable	0: Disable destination repeat mode 1: Enable destination repeat mode										R/W		
b4	SRPTEN	Source address repeat mode enable	0: Disable source repeat mode 1: Enable source repeat mode										R/W		
b3-b2	DINC[1:0]	Destination address update mode	0x0: Fixed mode 0x1: Increment mode 0x2, 0x3: Decrement mode										R/W		
b1-b0	SINC[1:0]	Source address update mode	0x0: Fixed mode 0x1: Increment mode 0x2, 0x3: Decrement mode										R/W		

15.5.27 Channel Monitor Registers (DMA_MONSARx, DMA_MONDARx, DMA_MONDTCTLx, DMA_MONRPTx, DMA_MONSEQCTLx, DMA_MONDNSEQCTLx) (x=0~7)

These monitoring registers' structure is same with the corresponding channel descriptor register, but all are read-only registers. Unlike the 8 channel descriptor registers whose value remains unchanged during the transaction, The value of these registers update after each data block transfer is completed. The update content and method are as follows:

- DMA_MONSARx.SAR[31:0], DMA_MONDARx.DAR[31:0]: Update to the address of the next transfer, according to the fixed/increment/decrement/reload/nonsequence mode set by the channel descriptor.
- DMA_MONDTCTLx.CNT[15:0]: Subtract 1, if it is already 0, keep it as 0.
- DMA_MONRPTx.SRPT[9:0], DRPT[9:0]: When the channel reconfigure transfer is disabled, it is subtracted by DMA_DTCTL.BLKSIZE, and when it is decreases to 0, it reloads the register DMA_RPTx. When the channel reconfigure transfer is enabled, it decreases by 1, and reload DMA_RTPx when it decreases to 0.
- DMA_MONSEQCTLx.SNSCNT[11:0], DMA_MONDNSEQCTLx.DNSCNT[11:0]: When the channel reconfigure transfer is disabled, it is subtracted by DMA_DTCTL.BLKSIZE, and when it is decreases to 0, it reloads the register DMA_SNSEQCTLx/DMA_DNSEQCTLx. the channel reconfigure transfer is enabled, it decreases by 1, and reload DMA_SNSEQCTLBx/ DMA_DNSEQCTLBx when it decreases to 0.
- The monitor register bits other than the above remain the same value as the descriptor registers.

15.6 Cautions

- DMA registers only support 32-bit read and write operations, and 8/16-bit read and write operations are invalid.
- When a bus error occurs during a DMA transfer and another channel is in a waiting state, the DMA goes into a lock state and cannot respond to all subsequent channel transfer requests. Once in this state, the lock state cannot be unlocked by configuring the DMA's own registers. You need to perform a software reset or notify the peripheral circuit to reset the system.

To detect a DMA lock, query the error flag bit DMA_INTSTAT0.TRNERR[7:0] (which can be placed in DMA error interrupt handling). If the flag bit is 0, it indicates that no bus error occurred and the DMA is not locked. If the flag bit is not 0, continue to query the channel status bit DMA_CHSTAT.CHACT[7:0]. If CHACT[7:0]=0x0, the channel is not locked. If one of them, such as CHACT[x] of channel x, is 1 and remains 1 for a long time (longer than it takes for its normal transmission), and CHACT[x] remains 1 even if the corresponding channel license bit DMA_CHEN.CHEN[x] is cleared to 0 by software, DMA is locked.

To prevent DMA from entering a lock state, you need to avoid allowing DMA access to areas where bus errors occur, such as reserved address space, protected address space, etc. When you want to access areas that may have bus errors, you can disable other channels of the DMA unit to prevent other channels from waiting in case of bus errors.

16 Voltage comparator (CMP)

16.1 Introduction

The voltage comparator (Comparator, hereinafter referred to as CMP) is a peripheral module that compares two analog voltages. There are four comparison channels CMP1~4.

CMP main features:

- CMP1~4 can be independently compared for voltage
- Window comparison can be completed when CMP1, 2 or CMP3, 4 are used at the same time
- Both positive and negative terminal voltages have multiple input voltage sources for selection
- Optional sampling clock for digital noise filter
- The output can be switched on and off using the PWM waveform output by the timer
- Can generate events that trigger other peripherals to start
- Interrupt and wake-up from system stop mode
- The comparison result can be output to the external pin VCOUT

16.2 Functional block diagram

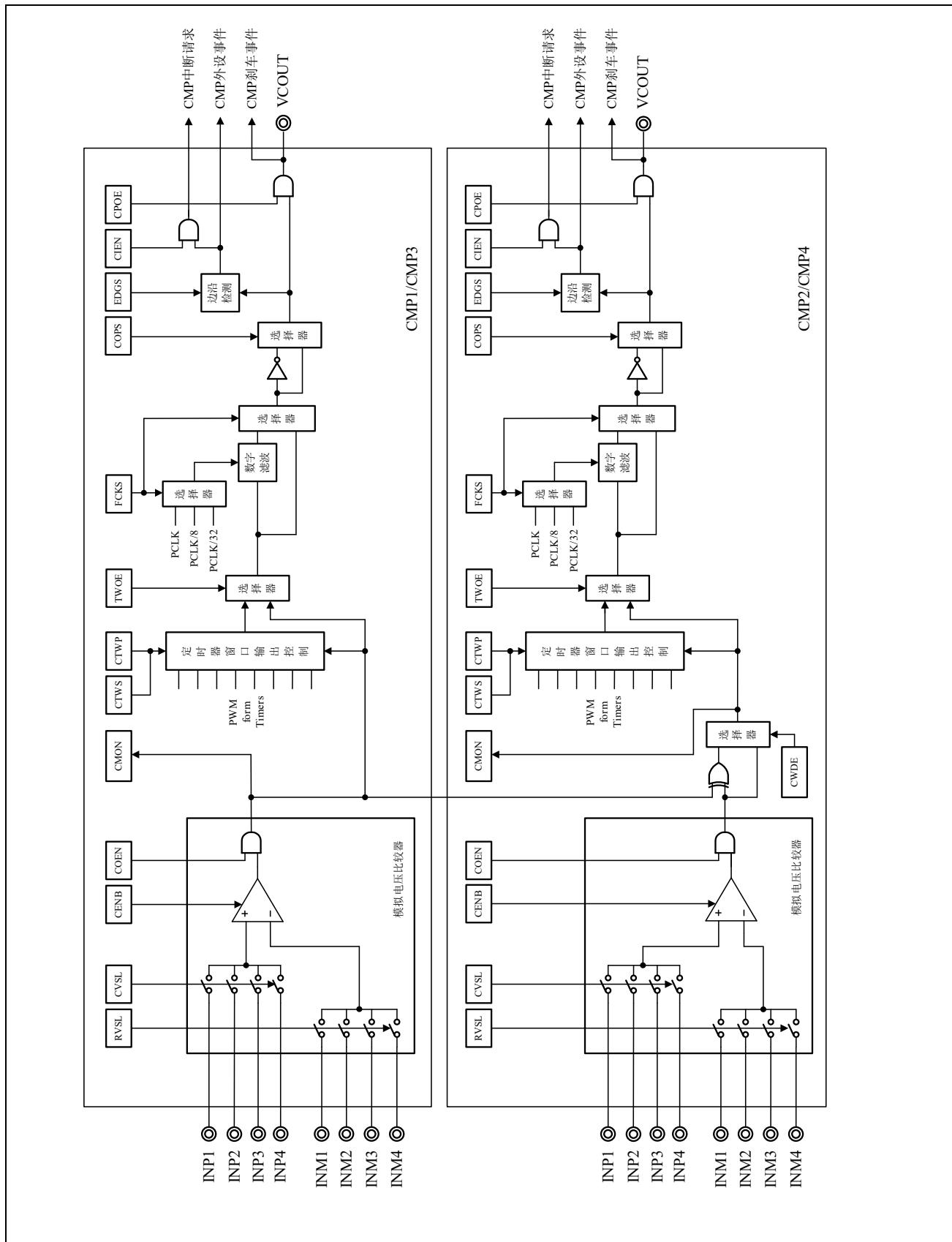


Figure 16-1 Functional block diagram

Table 16-1 CMP pin

Pin name	Input/Output	Function
AVCC0	Input	Analog power
AVSS0	Input	Analog ground
CMP1_INP2	Input	CMP1 positive analog input
CMP1_INP3	Input	CMP1 positive analog input
CMP1_INP4	Input	CMP1 positive analog input
CMP123_INM3	Input	CMP1, 2, 3 negative terminal analog input
CMP1_INM4	Input	CMP1 negative terminal analog input
CMP2_INP3	Input	CMP2 positive analog input
CMP2_INP4	Input	CMP2 positive analog input
CMP2_INM4	Input	CMP2 negative analog input
CMP3_INP2	Input	CMP3 positive analog input
CMP3_INP3	Input	CMP3 positive analog input
CMP3_INP4	Input	CMP3 positive analog input
CMP3_INM4	Input	CMP3 negative analog input
CMP4_INP3	Input	CMP4 positive terminal analog input
CMP4_INP4	Input	CMP4 positive terminal analog input
CMP4_INM3	Input	CMP4 negative terminal analog input
CMP4_INM4	Input	CMP4 negative terminal analog input
VCOUT1	Output	External port output of CMP1
VCOUT2	Output	External port output of CMP2
VCOUT3	Output	External port output of CMP3
VCOUT4	Output	External port output of CMP4
VCOUT	Output	External port output of CMP1~4

16.3 Functional description

16.3.1 General comparison mode

In normal comparison mode, CMP1~4 can operate independently. Take CMP1 as an example, the setting steps are as follows:

- 1) Write 0 to PWC_FCG3.bit3 and wait for at least 2us.
- 2) Disable the module stop function of CMP (write 0 in PWC_FCG3.bit8).
- 3) Set the CVSL bit of the CMP_PMSR register to select the positive terminal voltage;
Set the RVSL bit in the CMP_PMSR register to select the negative terminal voltage.
- 4) The CENB bit in the CMP_MDR register is set to 1 to enable the comparator.
- 5) Wait for the steady time tCMP of the analog voltage comparator.
- 6) Set the FCKS bit of the CMP_FIR register to set the digital filter;
Set the EDGS and CIEN bits of the CMP_FIR register to set interrupt and edge detection conditions.
- 7) Set the CPOE and COPS bits of the CMP_OCR register, and set the output of VCOUT;
Set the TWOE and TWOL bits of the CMP_OCR register and the CMP_TWSR and CMP_TWPR registers to set the timer window output.
- 8) The COEN bit of the CMP_OCR register is set to 1 to enable the comparator output.

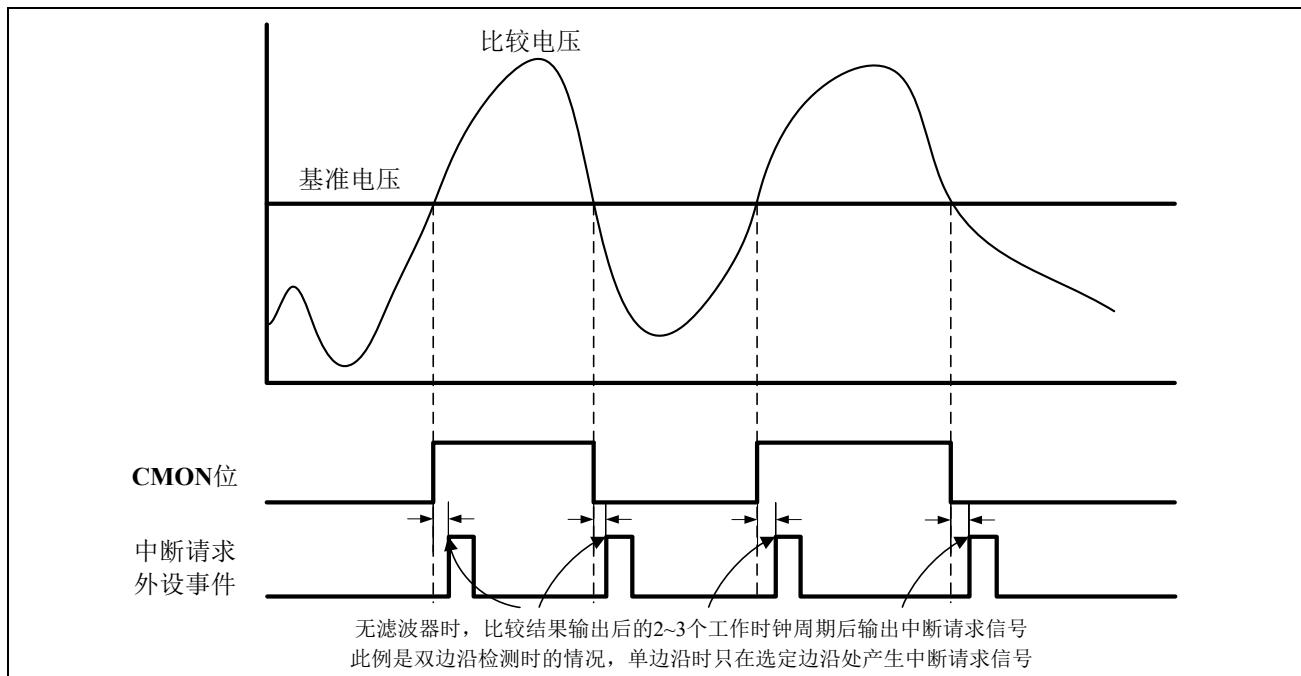


Figure 16-2 Working diagram of common comparison mode

Such as Figure 16-2, in normal comparison mode and positive output, the positive terminal voltage is higher than the negative terminal voltage, the CMON bit of the CMP_MDR register is 1; the positive terminal voltage is lower than the negative terminal voltage, the CMON bit is 0.

For CMP1 and CMP3, if INP2 or INP3 is selected as the positive terminal voltage, then the voltage input source needs to be selected by setting CMP_VISR. specific referenceTable 16-3 .

16.3.2 Window comparison mode

Window comparison is possible when CMP1, 2 or CMP3, 4 are running at the same time. Taking CMP1 and 2 as an example, the setting steps are as follows:

- 1) Write 0 to PWC_FCG3.bit3 and wait for at least 2us.
- 2) Disable the module stop function of CMP1 and 2 (write 0 in PWC_FCG3.bit8).
- 3) Set the CVSL bit of the CMP1_PMSR register to select the positive terminal voltage;
Set the RVSL bit of the CMP1_PMSR register to select the lower limit negative terminal voltage;
- 4) Set the CVSL bit of the CMP2_PMSR register to select the positive terminal voltage;
Note that the selected input voltage source must be the same as in (2).
Set the RVSL bit in the CMP2_PMSR register to select the upper limit negative terminal voltage.
- 5) CMP2_MDRregister CWDE bit is set to 1, select window comparison mode;
- 6) The CENB bit in the CMP1_MDR and CMP2_MDR registers is set to 1, enabling the comparator.
- 7) Wait for the steady time tCMP of the voltage comparator.
- 8) Set the FCKS bit of the CMP2_FIR register to set the digital filter;
Set the EDGS and CIEN bits of the CMP2_FIR register to set interrupt and edge detection conditions.
Set CPOE and COPS bits of CMP2_OCRregister and VCOUT output.
Set the TWOE and TWOL bits of the CMP2_OCR register and the CMP2_TWSR and CMP2_TWPR registers to set the timer window output.
- 9) The COEN bit of the CMP1_OCR, CMP2_OCR registers is set to 1 to enable the comparator output.

During window comparison, CMP2 or CMP4 completes operations such as monitoring, filtering, interrupting and outputting the comparison result. Such asFigure 16-3 as shown, if positive output is selected, when the positive terminal voltage is between the lower limit negative terminal voltage and the upper limit negative terminal voltage, the CMON bit of the CMP2_MDR register is 1; otherwise, the CMON bit is 0.

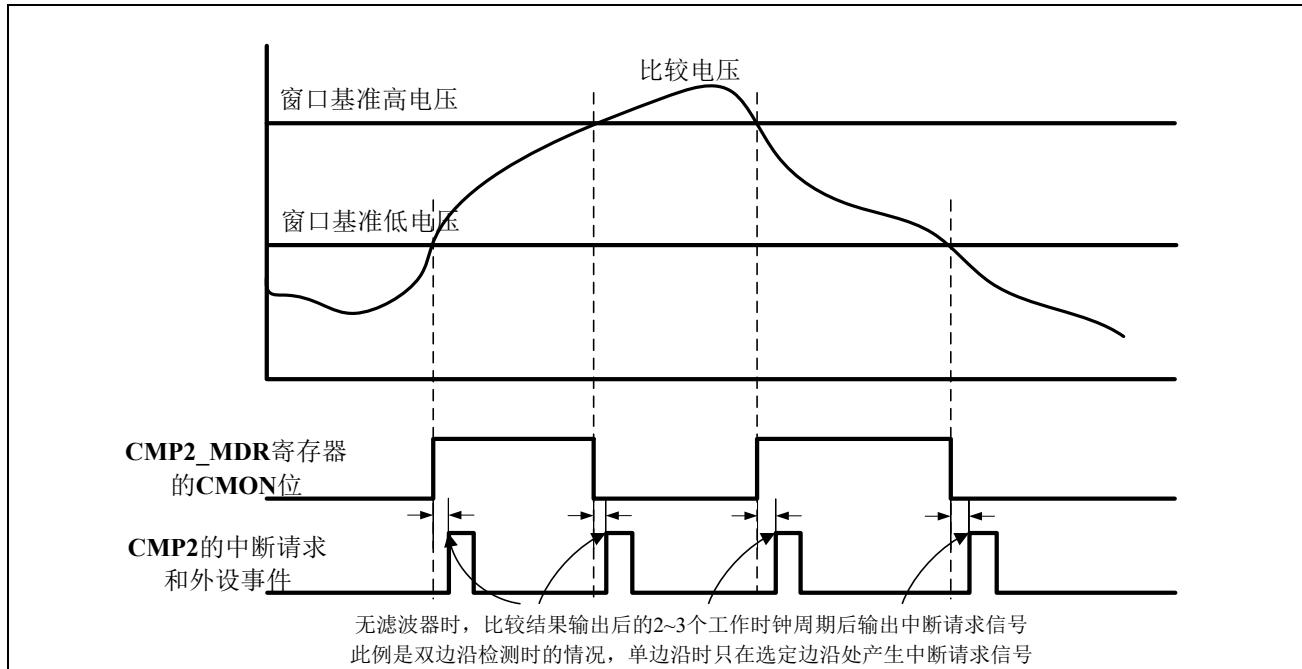


Figure 16-3 Window comparison mode working diagram

16.3.3 Timer window output

When the timer window output is inactive (TWOE bit in the CMP_OCR register is 0), the output of the voltage comparator is only controlled by the COEN bit in the CMP_OCR register. Output is allowed when COEN bit is 1, otherwise output is disabled and fixed to low level.

When the timer window output is active (TWOE bit in the CMP_OCR register is 1), the output of the voltage comparator is controlled by the timer window signal selected by the CMP_TWSR register in addition to the COEN bit in the CMP_OCR register. The polarity of the timer window signal when output is enabled can be set through the CMP_TWPR register, and the fixed level of the timer window output when output is disabled through the TWOL bit in the CMP_OCR register.

16.3.4 Digital filter

Each of the four compare channels is equipped with digital filters that noise filter the output of their respective voltage comparators. The digital filter samples the output of the voltage comparator according to the sampling clock. The next sampling clock with the same sampling level for three times will output the sampling value, otherwise it will keep the output unchanged. The sampling clock can be selected by the FCKS bit in the CMP_FIR register. When the FCKS bit is set to 00b, the digital filter is turned off.

16.3.5 Comparator interrupt

Comparator interrupt requests can be generated for each of the four compare channels. When using an interrupt, the CIEN bit of the CMP_FIR register must be set to 1, and the output edge of the comparator that generates the interrupt must be set through the EDGS bit (a value other than 2'b00) of the CMP_FIR register. If digital filtering and inverting output are also set, the EDGS bit is set to detect the edge of the signal after digital filtering and inverting output.

Comparator interrupts can also wake-up from low-power modes. Set the interrupt to be valid (CIEN=1) and disable digital filtering and edge detection, the interrupt will occur when the comparator output changes from low to high and wake up to stop the low-power mode. If the inverting output is also set, then an interrupt will occur when the inverting output goes from low to high and wake up from low power mode. After returning to working mode, set interrupt to invalid (CIEN = 0) before proceeding.

16.3.6 Peripheral triggering event

The same as the generation condition of the interrupt request, the event signal that triggers other peripherals is generated by detecting the output edge of the digital filter set by the CMP_FIR register. But unlike the interrupt request, the peripheral event is always output, regardless of the CIEN bit of the CMP_FIR register. The event trigger function must be enabled or disabled through the hardware trigger source register of the target peripheral.

16.3.7 External pin output

The comparison result after passing through the digital filter can be output to the external port VCOUT. Through the CPOE and COPS bits of the CMP_OCR register, whether to allow the output and the output polarity (normal phase output or reverse phase output) can be set respectively. The port corresponding to the reset VCOUT is a general port and input state, so it must also be set to VCOUT output via port register.

16.4 Precautions

16.4.1 Module stop function

CMP has module stop function, by setting module stop register can turn off the digital part of the module. CMP1,2 share a module stop control bit PWC_FCG3.bit8, CMP3,4 share a module stop control bit PWC_FCG3.bit9. CMP1~4 are in the stop state initially, and the respective registers can be accessed only when the module is set to work. For related register settings, please refer to the chapter on low power consumption.

16.4.2 The act of stopping a module.

When the CMP enters the module stop state in the working state, the comparator will continue to work, and the power consumption is equal to the working state. To further reduce power consumption, clear the CENB bit in the CMP_MDR register to "0".

16.4.3 The act of stopping a low-power mode.

When the chip enters the stop low-power mode, if the CMP is in operation, it will continue to operate after entering the stop low-power mode, which is equivalent to the level before entering the stop low-power mode. To further reduce power consumption, please clear the CENB bit in the CMP_MDR register to "0" before entering the stop low power mode.

16.5 Register description

CMP1 base address: 0x4004A000

CMP2 base address: 0x4004A010

CMP3 base address: 0x4004A400

CMP4 base address: 0x4004A410

Table 16-2 List of CMP registers

Register name	Symbol	Offset address	Bit width	Reset value
Comparator mode setting register	CMPx_MDR, x=1~4	0x00	8	0x00
Comparator Filter and Interrupt Control Register	CMPx_FIR, x=1~4	0x01	8	0x00
Comparator output control register	CMPx_OCR, x=1~4	0x02	8	0x00
Comparator positive and negative input selection register	CMPx_PMSR, x=1~4	0x03	8	0x00
Comparator Timer Window Select Register	CMPx_TWSR, x=1~4	0x04	16	0x0000
Comparator Timer Window Polarity Register	CMPx_TWPR, x=1~4	0x06	16	0x0000
Comparator Voltage Input Source Selection Register	CMPx_VISR, x=1, 3	0x08	16	0x0000

16.5.1 Comparator mode setting register (CMPx_MDR, x=1~4)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
CMON	-	-	-	-	-	CWDE	CENB

Bit	Marking	Place name	Function	Read and write
B7	CMON	Comparator result monitoring	General comparison mode 0: positive terminal voltage < negative terminal voltage 1: Positive terminal voltage > Negative terminal voltage Window compare mode (this bit is only valid for CMP2 and CMP4) 0: positive terminal voltage < lower limit negative terminal voltage, or Positive terminal voltage > upper limit negative terminal voltage 1: lower limit negative terminal voltage < positive terminal voltage < upper limit negative terminal voltage	R
b6~b2	Reserved	-	Read as "0", write as "0"	R/W
b1	CWDE	window comparison selection	0: General comparison mode 1: Window comparison mode Note: This bit is only present in CMP2 and CMP4. After the window comparison mode is set, the filtering, edge detection and output control of the window comparison result are all completed by CMP2 or CMP4.	R/W
b0	CENB	Comparator Work Permit	0: Voltage comparator stopped 1: Voltage comparator operation Note: Each time CENB is set from '0' to '1', it takes approximately 300 ns of work stabilization time to follow up.	R/W

16.5.2 Comparator Filter and Interrupt Control Register (CMPx_FIR, x=1~4)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	B0
-	CIEN	EDGS[1:0]		-	-	FCKS[1:0]	
Bit	Marking	Place name		Function			Read and write
B7	Reserved			Read as "0", write as "0"			R/W
b6	CIEN	Comparator interrupt enable		0: Disable comparator interrupt 1: Allow the comparator to interrupt			R/W
b5~b4	EDGS[1:0]	Comparator output edge Test selection		00: Do not detect edge of comparator output 01: Detect the rising edge of the comparator output 1 0: Detect the falling edge of the comparator output 1 1: Detect the rising and falling edges of the comparator output Note: Changing EDGS [1: 0] may cause a interrupt or peripheral trigger event, so set the register if the interrupt disable or peripheral trigger is invalid. After the register is set, clear the corresponding interrupt flag.			R/W
b3~b2	Reserved			Read as "0", write as "0"			R/W
b1~b0	FCKS[1:0]	Comparator result filtering Sampling selection		0: No noise filter 0 1: use noise filter, sample via PCLK 1 0: use noise filter, sample via PCLK/8 1 1: Use noise filter, sample via PCLK/32 Note: Please rewrite FCKS[1:0] when the comparator output is disabled (ie the COEN bit is "0"). When FCKS [1: 0] switches from '00b' to a different value, use the filtered output after 4 samples as a interrupt request or a peripheral trigger event. Changing FCKS [1: 0] may cause a interrupt or peripheral trigger event, so set the register if the interrupt disable or peripheral trigger is invalid. After the register is set, clear the corresponding interrupt flag.			R/W

16.5.3 Comparator output control register (CMPx_OCR, x=1~4)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	B0
-	-	-	TWOL	TWOE	CPOE	COPS	COEN
<hr/>							
Bit	Marking	Place name	Function			Read and write	
b7~b5	Reserved	-	Read as "0", write as "0"			R/W	
b4	TWOL	Timer window output Level selection	0: Timer window output is fixed at low level when output is disabled 1: Timer window output is fixed at high level when output is disabled			R/W	
b3	TWOE	Timer window output Enable	0: Close Timer Window Output Function (Normal Output Mode) 1: Open Timer Window Output Function			R/W	
b2	CPOE	VCOUT output allows	0: Disable VCOOUT output 1: Allow VCOOUT output			R/W	
b1	COPS	Comparator Output Polarity Selection	0: Comparator positive phase output 1: Comparator inverted output Note: Override COPS when comparator output is disabled (i.e. COEN bit is "0"). Changing the COPS bit may cause a interrupt or peripheral trigger event, so set the register if the interrupt disable or peripheral trigger is invalid. After the register is set, clear the corresponding interrupt flag.			R/W	
b0	COEN	Comparator output permissible	0: Disable comparator output (comparator output fixed to low level) 1: Allow comparator output			R/W	

16.5.4 Comparator positive and negative input selection register (CMPx_PMSR, x=1~4)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0	
			CVSL[3:0]	RVSL[3:0]				
Bit	Marking	Place name	Function					Read and write
b7~b4	CVSL[3:0]	Positive input selection	Select the positive input voltage of the comparator 0000: No positive terminal voltage input 0001: Select INP1 as the positive terminal voltage (seeTable 16-3) 0010: Select INP2 as the positive terminal voltage (see Table 16-3) 0100: Select INP3 as positive terminal voltage (for selected input see Table 16-3) 1000: Select INP4 as the positive terminal voltage (for selected input see Table 16-3) other: set ban Note: Please rewrite CVSL when the comparator output is disabled (ie the COEN bit is "0"). After each rewrite of CVSL, please wait 300ns before enabling the comparator output (ie the COEN bit is set to "1"). Changing CVSL may cause an interrupt or peripheral trigger event, so please set this register when the interrupt is disabled or the peripheral trigger function is disabled. After the register is set, clear the corresponding interrupt flag.					R/W
b3~b0	RVSL[3:0]	Negative input selection	Selection of the Negative Input Voltage of the Analog Voltage Comparator 0000: No negative terminal voltage input 0001: Select INM1 as the negative terminal voltage (see Table 16-4) 0010: Select INM2 as the negative terminal voltage (seeTable 16-4) 0100: Select INM3 as the negative terminal voltage (see Table 16-4) 1000: Select INM4 as the negative terminal voltage (for selected input seeTable 16-4) other: set ban Note: Please rewrite RVSL when the comparator output is disabled (ie the COEN bit is "0"). After each rewrite of RVSL, please wait 300ns before enabling the comparator output (ie the COEN bit is set to "1"). Changing RVSL may cause an interrupt or peripheral trigger event, so please set this register with interrupt disabled or peripheral trigger function disabled. After the register is set, clear the corresponding interrupt flag.					R/W

16.5.5 Comparator Voltage Input Source Selection Register (CMPx_VISR, x=1, 3)

CMP1_VISR Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
-	-	-	-	-	-	-	-
B7	b6	b5	b4	b3	b2	b1	B0
-	-	P3SL[1:0]			-	P2SL[2:0]	

Bit	Marking	Place name	Function	Read and write
b15~b6	Reserved	-	Read as "0", write as "0"	R/W
b5~b4	P3SL[1:0]	INP3 input selection	Select the input voltage when CMP1_PMSR.CVSL[3:0] is 3'b0100 00: No voltage input 01/10: Selected inputs seeTable 16-3 11: Prohibitions Note: Please rewrite P3SL when CMP1_PMSR.CVSL[2] is "0".	R/W
b3	Reserved	-	Read as "0", write as "0"	R/W
b2~b0	P2SL[2:0]	INP2 input selection	Select the input voltage when CMP1_PMSR.CVSL[3:0] is 3'b0010 000: No voltage input 001/010/100: Selected input see Table 16-3 other: set ban Note: Please rewrite P2SL when CMP1_PMSR.CVSL[1] is "0".	R/W

CMP3_VISR Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
-	-	-	-	-	-	-	-
B7	b6	b5	b4	b3	b2	b1	B0
-	-	P3SL[1:0]			-	P2SL[2:0]	

Bit	Marking	Place name	Function	Read and write
b15~b6	Reserved	-	Read as "0", write as "0"	R/W
b5~b4	P3SL[1:0]	INP3 input selection	Select the input voltage when CMP3_PMSR.CVSL[3:0] is 3'b0100 00: No voltage input 01/10: Selected voltage seeTable 16-3 11: Prohibitions Note: Please rewrite P3SL when CMP3_PMSR.CVSL[2] is "0".	R/W
b3	Reserved	-	Read as "0", write as "0"	R/W
b2~b0	P2SL[2:0]	INP2 input selection	Select the input voltage when CMP3_PMSR.CVSL[3:0] is 3'b0010 000: No voltage input 001/010/100: For selected voltage seeTable 16-3 other: set ban Note: Please rewrite P2SL when CMP3_PMSR.CVSL[1] is "0".	R/W

Table 16-3 Positive input voltage list

Compare channels	CVSL=0001b	CVSL=0010b		CVSL=0100b		CVSL=1000b
CMP1	PGA1_BP	P2SL=00 1b	PGA1	P3SL=01 b	CMP1_INP 3	CMP1_INP4
		P2SL=01 0b	PGA2	P3SL=10 b	CMP2_INP 3	
		P2SL=10 0b	CMP1_INP 2			
CMP2	PGA2_BP	PGA2		CMP2_INP3		CMP2_INP4
CMP3	PGA3_BP	P2SL=00 1b	PGA3	P3SL=01 b	CMP3_INP 3	CMP3_INP4
		P2SL=01 0b	PGA4	P3SL=10 b	CMP4_INP 3	
		P2SL=10 0b	CMP3_INP 2			
CMP4	PGA4_BP	PGA4		CMP4_INP3		CMP4_INP4

Note: In the table, PGA1_BP/PGA2_BP/PGA3_BP/PGA4_BP are respectively the unamplified voltage signals output by PGA1~4, and the PGA should also be set to be valid when using.

Table 16-4 Negative input voltage list

Compare channels	RVSL=0001b	RVSL=0010b	RVSL=0100b	RVSL=1000b
CMP1	DA1O1	DA1O2	CMP123_INM3	CMP1_INM4
CMP2	DA1O1	DA1O2	CMP123_INM3	CMP2_INM4
CMP3	DA2O1	DA2O2	CMP123_INM3	CMP3_INM4
CMP4	DA2O1	DA2O2	CMP4_INM3	CMP4_INM4

16.5.6 Comparator timer window selection register (CMPx_TWSR, x=1~4)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
CTWS15	CTWS14	CTWS14	CTWS12	CTWS11	CTWS10	CTWS9	CTWS8
B7	b6	b5	b4	b3	b2	b1	b0
CTWS7	CTWS6	CTWS5	CTWS4	CTWS3	CTWS2	CTWS1	CTWS0

Bit	Marking	Place name	Function	Read and write
b15~0	CTWS15~0	Timer window enable bit	When the timer window output is valid, select the timer PWM output as the comparison result output window. 0: Do not select the PWM output of the timer as the window signal 1: Select the PWM output of the timer as the window signal	R/W

Table 16-5 Timer Window PWM List

CTWS bit	CMP1	CMP2	CMP3	CMP4
15	TIM4_1_OWL	TIM4_2_OWL	TIM4_3_OWL	TIM4_3_OWL
14	TIM4_1_OWH	TIM4_2_OWH	TIM4_3_OWH	TIM4_3_OWH
13	TIM4_1_OVL	TIM4_2_OVL	TIM4_3_OVL	TIM4_3_OVL
12	TIM4_1_OVH	TIM4_2_OVH	TIM4_3_OVH	TIM4_3_OVH
11	TIM4_1_OUL	TIM4_2_OUL	TIM4_3_OUL	TIM4_3_OUL
10	TIM4_1_OUH	TIM4_2_OUH	TIM4_3_OUH	TIM4_3_OUH
9	TIM6_4_PWMA	TIM6_8_PWMA	TIM6_4_PWMB	TIM6_8_PWMB
8	TIM6_3_PWMA	TIM6_7_PWMA	TIM6_3_PWMB	TIM6_7_PWMB
7	TIM6_2_PWMA	TIM6_6_PWMA	TIM6_2_PWMB	TIM6_6_PWMB
6	TIM6_1_PWMA	TIM6_5_PWMA	TIM6_1_PWMB	TIM6_5_PWMB
5	TIMA_2_PWM3	TIMA_4_PWM3	TIMA_3_PWM3	TIMA_4_PWM3
4	TIMA_2_PWM2	TIMA_4_PWM2	TIMA_3_PWM2	TIMA_4_PWM2
3	TIMA_2_PWM1	TIMA_4_PWM1	TIMA_3_PWM1	TIMA_4_PWM1
2	TIMA_1_PWM3	TIMA_3_PWM3	TIMA_1_PWM3	TIMA_2_PWM3
1	TIMA_1_PWM2	TIMA_3_PWM2	TIMA_1_PWM2	TIMA_2_PWM2
0	TIMA_1_PWM1	TIMA_3_PWM1	TIMA_1_PWM1	TIMA_2_PWM1

16.5.7 Comparator Timer Window Polarity Register (CMPx_TWPR, x=1~4)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
CTWP15	CTWP14	CTWP13	CTWP12	CTWP11	CTWP10	CTWP9	CTWP8
B7	b6	b5	b4	b3	b2	b1	b0
CTWP7	CTWP6	CTWP5	CTWP4	CTWP3	CTWP2	CTWP1	CTWP0

Bit	Marking	Place name	Function	Read and write
b15~b0	CTWP15~0	Timer window polarity selection	CTWP15~0 are used to set the active level of the window signal selected by CMPx_TWSR. 0: Allow comparator output when window signal is low 1: Allow comparator output when window signal is high	R/W

17 Analog-to-digital conversion module (ADC)

17.1 Introduction

12-bit ADC is an analog-to-digital converter using the Successive Approximation Register (SAR) principle to convert analog input values (voltages) to discrete digital values. This MCU is equipped with 3 ADC units, among which units 1 and 2 support 16 channels, and unit 3 supports 20 channels, which can convert analog signals from external pins and inside the chip. The analog input channels can be arbitrarily combined into a sequence which can perform single scan conversion, or continuous scan conversion. The ADC module supports continuous multiple conversions for any specified channel and averages the conversion results. Analog watchdog monitors the conversion result of any specified channel and detects whether it exceeds the range set by the user.

ADC main characteristics

- High performance
 - Configurable 12-bit, 10-bit and 8-bit resolution
 - The frequency ratio of ADC digital interface clock PCLK4 and conversion clock PCLK2 (also called ADCLK) can be set to 1:1, 2:1, 4:1, 8:1, 1:2, 1:4
 - PCLK2 can choose a PLL clock that is asynchronous with the system clock HCLK. At this time, the frequency is PCLK4:PCLK2=1:1
 - PCLK2 frequency supports up to 60MHz
 - Sampling Rate: 2.5MSPS (PCLK2=60MHz, 12 bits, sampling 11 cycles, conversion 13 cycles)
 - Independent programming of sampling time for each channel
 - Independent data register of each channel
 - Data register can be configured as left/right alignment
 - Continuous multiple conversion and averaging function
 - Analog watchdog, monitoring conversion results
 - The ADC module can be set to stop when not in use
- Analog input channel
 - 28 external analog inputs, one ADC unit can support up to 20 channels
 - 2 internal analog inputs: Internal reference voltage, VBAT dividing voltage
- Conversion start condition
 - Software setup
 - Peripheral device synchronization trigger
 - External pin trigger
- Conversion mode
 - 2 scan sequences A and B, optionally specifying single-channel or multiple-channel
 - Sequence A single scan
 - Sequence A continuous scan

- Double sequence scan, sequence A and B independently select trigger source, sequence B has higher priority than A
- Synchronized mode (for devices with two or three ADCs)
- Interrupt and Event Output
 - Sequence A Scan End Interrupt and Event ADC_EOCA
 - Sequence B Scan End Interrupt and Event ADC_EOCB
 - Analog watchdog 0 compared interrupt and event ADC_CMP0
 - Analog watchdog 1 compared interrupt and event ADC_CMP1
 - The 4 above events can start DMA if the product supports DMA

17.2 ADC system block diagram

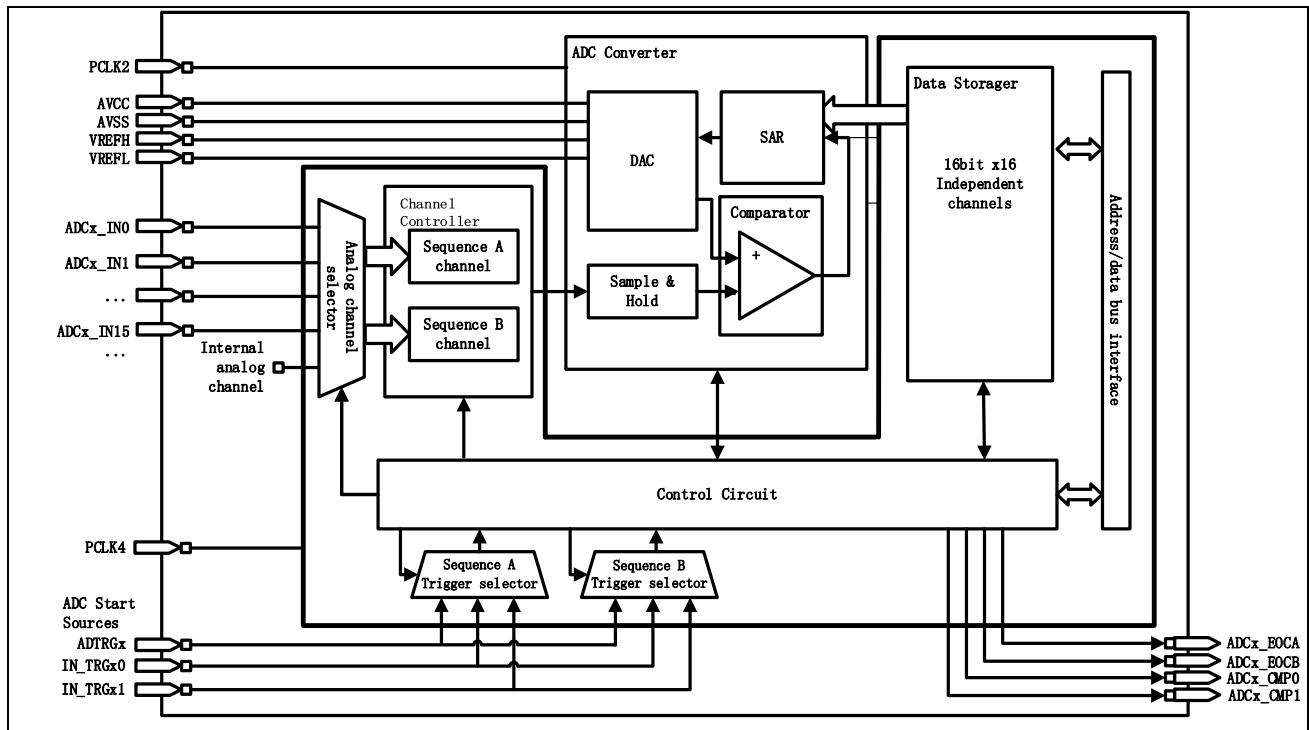


Figure 17-1 ADC block diagram

This chip contains 3 ADC module units, and the configuration of each unit is different. Please refer to the following table for details:

Table 17-1 Specifications of each ADC unit

Item		ADC1	ADC2	ADC3
Power supply		AVCC		
		AVSS/VREFL		
The reference voltage		VREFH		
Analog channel*1	CH0	ADC123_IN0	ADC123_IN0	ADC123_IN0
	CH1	ADC123_IN1	ADC123_IN1	ADC123_IN1
	CH2	ADC123_IN2	ADC123_IN2	ADC123_IN2
	CH3	ADC123_IN3	ADC123_IN3	ADC123_IN3
	CH4	ADC12_IN4	ADC12_IN4	ADC3_IN4
	CH5	ADC12_IN5	ADC12_IN5	ADC3_IN5
	CH6	ADC12_IN6	ADC12_IN6	ADC3_IN6
	CH7	ADC12_IN7	ADC12_IN7	ADC3_IN7
	CH8	ADC12_IN8	ADC12_IN8	ADC3_IN8
	CH9	ADC12_IN9	ADC12_IN9	ADC3_IN9
	CH10	ADC123_IN10	ADC123_IN10	ADC123_IN10
	CH11	ADC123_IN11	ADC123_IN11	ADC123_IN11
	CH12	ADC123_IN12	ADC123_IN12	ADC123_IN12
	CH13	ADC123_IN13	ADC123_IN13	ADC123_IN13
	CH14	ADC12_IN14	ADC12_IN14	ADC3_IN14
	CH15	ADC12_IN15/ Internal analog channel	ADC12_IN15/ Internal analog channel	ADC3_IN15/ Internal analog channel
	CH16	-	-	ADC3_IN16
	CH17	-	-	ADC3_IN17
	CH18	-	-	ADC3_IN18
	CH19	-	-	ADC3_IN19
Programmable Gain Amplifier PGA	PGA1/4	ADC123_IN0	ADC12_IN6	-
	PGA2	ADC123_IN1	-	-
	PGA3	ADC123_IN2	-	-
	COM	PGA123_VSS	PGA4_VSS	-
Dedicated sample and hold circuit SH	SH1	ADC123_IN0	-	-
	SH2	ADC123_IN1	-	-
	SH3	ADC123_IN2	-	-
hardware trigger source	external pins	ADTRG1	ADTRG2	ADTRG3
	periphery on- chip	IN_TRG10	IN_TRG20	IN_TRG30
		IN_TRG11	IN_TRG21	IN_TRG31

Note:

- The virtual channels CH0~CH15 in the ADC and the physical channel ADCx_INy (the actual analog input source) can be freely mapped by setting registers. This table shows the default mapping relationship after reset.

17.3 Functional description

17.3.1 ADC clock

The ADC module needs 2 clocks: Digital interface clock PCLK4 and analog circuit clock PCLK2. PCLK4 and PCLK2 are synchronous, and the frequency ratio can be set to 1:1, 2:1, 4:1, 8:1, 1:2, 1:4.

PCLK2 can choose a PLL clock source that is asynchronous with the system clock HCLK. At this time, PCLK4 and PCLK2 are synchronous and in same frequency.

Note:

- Please set the frequency of PCLK2 within 1MHz~60MHz.

17.3.2 Channel selection

The ADC module supports channel mapping, that is, the mapping between virtual channels in the module and actual physical channels. The virtual channel refers to the assumed channel in the ADC module. For example, if the register ADC_CHSELRA is set to 0x1, it means that sequence A selects the conversion CH0. This CH0 is the virtual channel, and the register ADC_DR0 is the conversion result register of the virtual channel CH0. The physical channel refers to the actual analog channel, including the analog input ADCx_INy of the external pin and the internal analog channels such as the internal reference voltage. The mapping between virtual channels and physical channels can be configured through the register ADC_CHMUXR, please refer to the register description for details. Unless otherwise specified in this chapter, channel n or CHn both represent virtual channels.

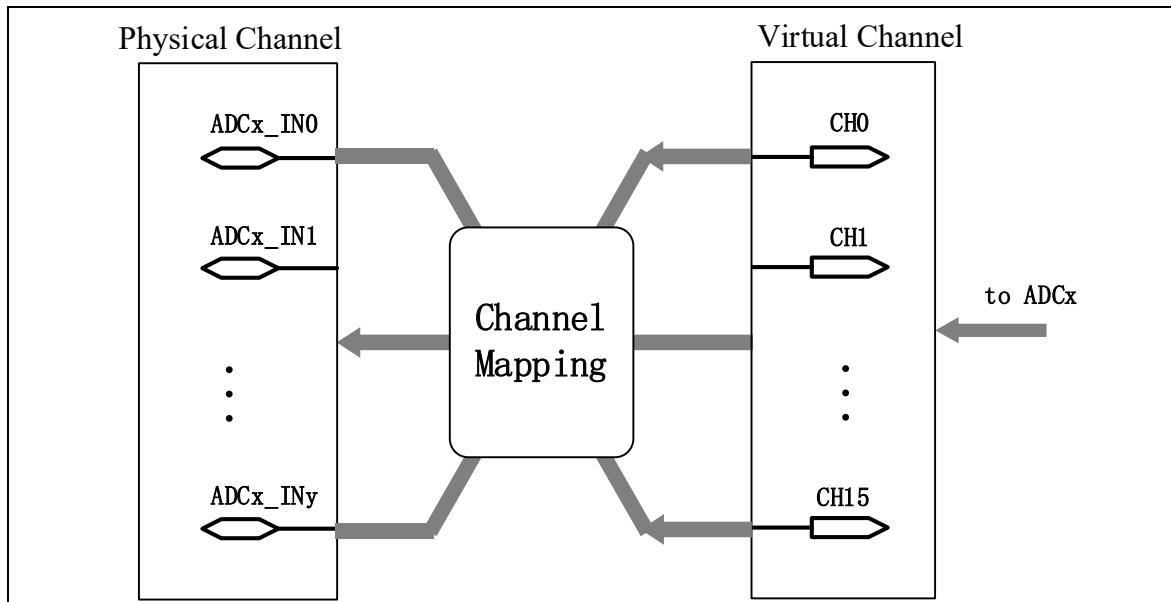


Figure 17-2 Channel Mapping Diagram

The ADC module has multiple channels that can be configured in two sequences: Sequence A, sequence B for conversion. Sequence A and B have their own independent channel selection registers ADC_CHSELRA, ADC_CHSELRB. Each bit of the register represents a channel. For example, writing 1 to bit0 means converting CH0, and writing 0 means not converting CH0. The two sequences can independently select any one or more channels for conversion. E.g: ADC_CHSELRA is set to 0x0055, ADCHSELRB is set to 0x0002, then when the trigger condition of sequence A occurs, the 4 channels of CH0, CH2, CH4 and CH6 will be converted in turn. When the trigger condition for sequence B occurs, CH1 is converted.

Among them, internal analog output and external input ADCx_IN 15 shares CH15. Setting ADC_CHMUX also changes the internal analog output and the mapping relationship between ADCx_IN15 and CHn. When the internal analog channel or ADCx_IN15 needs to be converted, ADC Extend Channel Select Register ADC_EXCHSELR needs to be set first, and then the corresponding virtual channel bit in ADC_CHSELRA/ B is written to 1 (write bit15 to 1 in the default mapping condition). When ADC_EXCHSELR is set to 0x0, it means to convert external analog input, when ADC_EXCHSELR is set to 0x1, it means to convert internal analog channel. For the selection of 2 internal analog signals, please refer to the description of the register PWC_PWRC4 in the Power Control (PWC) chapter.

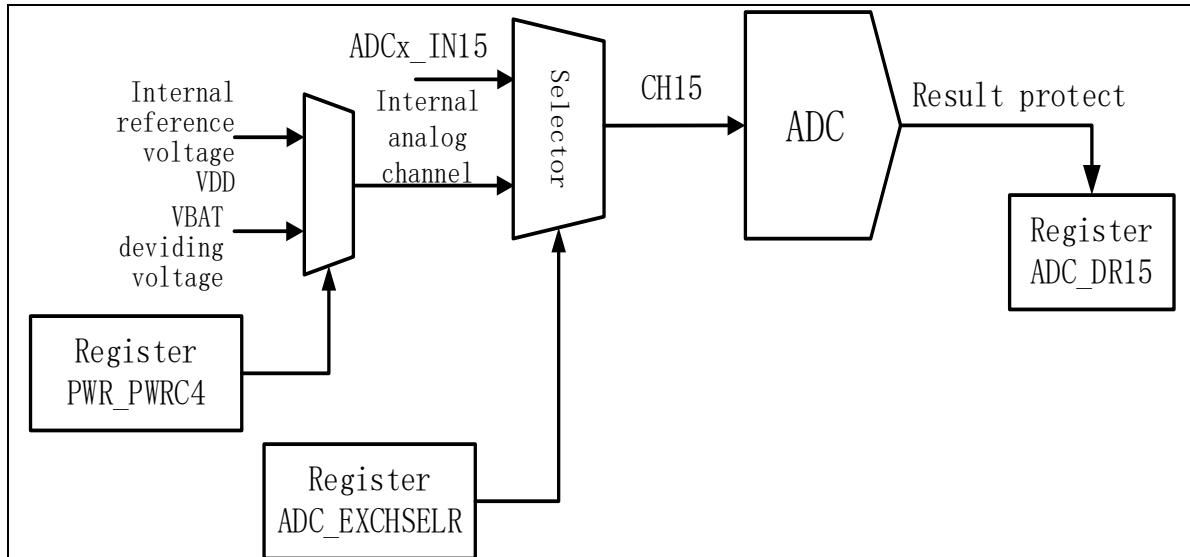


Figure 17-3 Internal analog channel selection

In addition, most physical channels can be input to multiple ADC units. For example, ADC123_IN0 can be input to three units of ADC_1, 2, and 3. ADC12_IN4 can be input to two units of ADC_1, 2. Combined with the multi-ADC synchronized mode and the channel mapping function, it is convenient to achieve high sampling rate conversion for the specified channel.

Note:

- Do not select the same channel in sequence A and B. For nonexistent channels, please do not set the corresponding registers and keep their reset states.

17.3.3 Trigger source selection

Sequence A and sequence B select their trigger source independently. Selectable trigger sources include external port ADTRGx, internal events IN_TRGx0, IN_TRGx1. Among them, the falling edge input of port ADTRGx is valid. IN_TRGx0, IN_TRGx1 are set by registers ADC_ITRGSELRO, 1, which can select rich event sources inside the chip. In addition, writing the register ADC_STR can generate a sequence A software trigger signal, which can only be used when the ADC is idle. The software trigger is independent of the setting of the trigger source selection register ADC_TRGSR.

17.3.4 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 sequence A start condition selected by the register ADC_TRGSR occurs, or the ADC_STR.START bit is written to 1, the ADC starts to sample and convert all channels selected in the sequence A channel selection register ADC_CHSELRA in turn, and the conversion results are stored in the corresponding data register ADC_DR. ADC_STR.START keep 1 during the ADC conversion process, and automatically be cleared to 0 when all channels are converted. The ADC enters the conversion idle state and waits for the next trigger condition.

Once all channels conversion complete, the sequence A conversion end flag bit ADC_EOCAF will be set to 1 and the sequence A conversion end event ADC_EOCA will generate, which can be used to start the DMA. If the ADC_ICR.EOCAIEN is 1, the interrupt request is generated if interrupt enable.

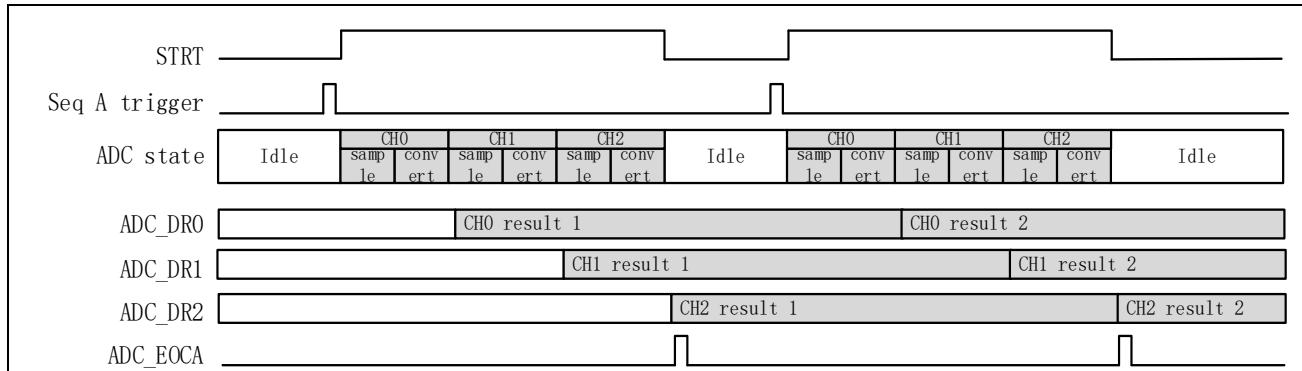


Figure 17-4 Sequence A single scan mode

Sequence A single scan mode software process:

1. Confirm that ADC_STR.START is 0, which means that the ADC is in conversion idle state.
2. Set A/D control register ADC_CR0.MS [1:0] to 00b to select sequence A single scan mode.
3. Set ADC Sequence A Channel Select Register ADC_CHSELRA.
4. Set the sampling time register ADC_SSTR.
5. Write 1 to ADC_STR.START to trigger sequence A or set register ADC_TRGSR to select sequence A trigger condition.
6. Poll the sequence A conversion end flag EOCAF.
7. Read each channel's data register ADC_DR.
8. Write 0 to clear the EOCAF flag bit to prepare for the next conversion.

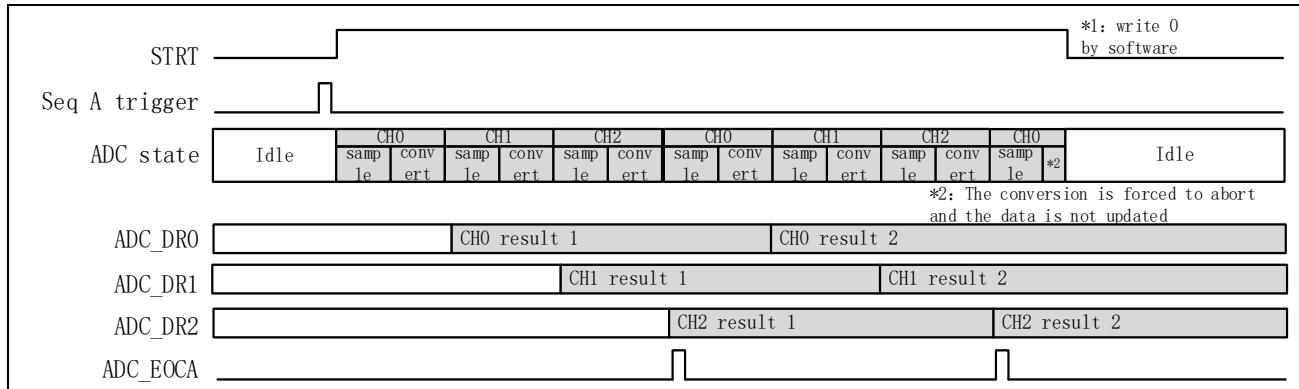
The CPU poll modes in steps 6 to 8 above can also be replaced by interrupt mode, which can be used to process the converted data or start DMA to read data by using the ADC_EOCA event.

17.3.5 Sequence A continuous scanning mode

The A/D control register AD_CR0_MS [1:0] is set to 01b to select the sequence A continuous scan mode.

Sequence A continuous scan mode is similar to sequence A single scan mode. The difference is that it does not enter conversion idle state after the last channel conversion complete in continuous scan mode, but restarts to convert sequence A and START bit will not be cleared to 0 automatically.

When a continuous scan needs to be stopped, write 0 to the START bit and poll START bit until it is 0 to determine if the ADC enters the conversion idle state.

**Figure 17-5 Continuous scan**

Sequence A Continuous Scan Mode software flow:

1. Confirm the ADC_STR.STRT is 0, which means that the ADC is in conversion idle state.
2. Set A/D control register ADC_CR0.MS [1:0] to 01b to select the sequence A continuous scan mode.
3. Set ADC Sequence A Channel Select Register ADC_CHSELRA.
4. Set the sampling time register ADC_SSTR.
5. Write 1 to ADC_STR.STRT to trigger sequence A or set register ADC_TRGSR to select sequence A trigger condition.
6. Poll the sequence A conversion end flag EOCAF.
7. Read each channel's data register ADC_DR.
8. Write 0 to clear EOCAF flag bit for next conversion.
9. Write 0 to the START bit when you do not need to continue the conversion, and poll the START bit to 0 to confirm that the ADC enters conversion idle state.

The CPU poll modes in steps 6 to 8 above can also be replaced by interrupt mode, which can be used to process the converted data or start DMA to read data by using the ADC_EOCA event.

Note:

- Since it is a continuous conversion, the interval between scans is shorter, especially when only one channel conversion is selected. It is recommended to use the ADC_EOCA event to start DMA read data in order to avoid data loss due to intemely processing in poll mode.

17.3.6 Double sequence scanning mode

The A/D control register ADC_CR0.MS [1:0] is set to 10b or 11b to select the double-sequence scan mode, that is, both sequence A and sequence B can be scanned by their 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 continuous scan mode, B is single scan mode.

For sequence A, the trigger source is selected by ADC_TRGSR.TRGSEL[2:0], and the converted channel is selected by ADC_CHSELRA. Sequence B selects the trigger source by ADC_TRGSR.TRGSELB[2:0], and selects the converted channel by ADC_CHSELRB.

When the conversion of all the channels of sequence A completes, the conversion end flag bit ADC_ISR.EOCAF of sequence A is set and the sequence A conversion end event ADC_EOCA is generated. If the interrupt permission bit ADC_ISR.EOCAIEN sets, the sequence A conversion end interrupt request is generated. When the conversion of all channels of sequence B completes, the conversion end flag bit ADC_ISR.EOCBF of sequence B sets, and the sequence B conversion end event ADC_EOCB is generated. If the interrupt permission bit ADC_ISCR.EOCBIEN sets, the sequence B conversion end interrupt is generated.

In double sequence scanning mode, sequence B will be prioritized when sequence A is competing with sequence B, that is, sequence B has higher priority than sequence A. Please refer to the table below for details.

Table 17-2 Various competitions of sequences A and B

A/D conversion	Trigger signal occurrence	Treatment	
		ADC_CR1.RSCHSEL=0	ADC_CR1.RSCHSEL=1
In Sequence A conversion	Sequence A triggering	Invalid trigger signal	
	Sequence B triggering	1) The sequence A conversion is interrupted and the sequence B conversion starts. 2) After the conversions of sequence B have completed, sequence A continues to be converted from the channel interrupted.	1) The sequence A conversion is interrupted and the sequence B conversion starts. 2) After the conversions of sequence B have completed, sequence A is reconverted from the first channel.
In Sequence B conversion	Sequence A triggering	All channel conversions of Sequence B have completed and sequence A conversion starts.	
	Sequence B triggering	Invalid trigger signal	
In ADC idle, sequence A and B are triggered simultaneously		Sequence B starts first, after all channel conversions have completed, sequence A conversion starts.	

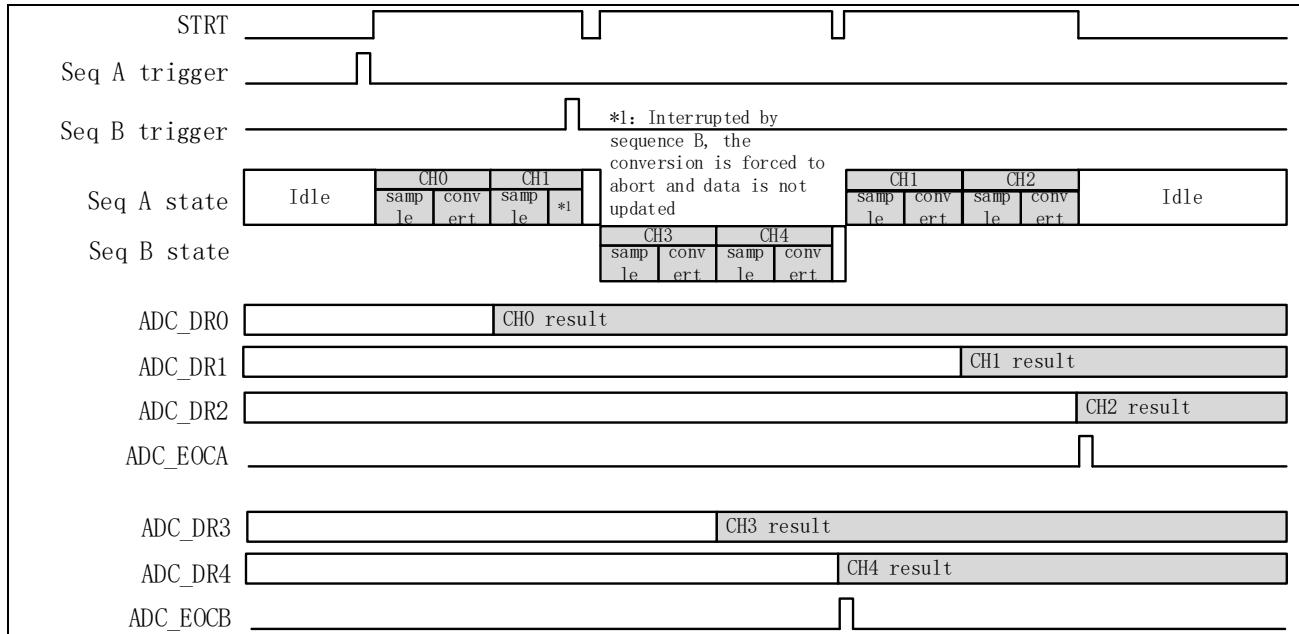


Figure 17-6 Double sequence scan mode (Sequence A restarts from interrupted channel)

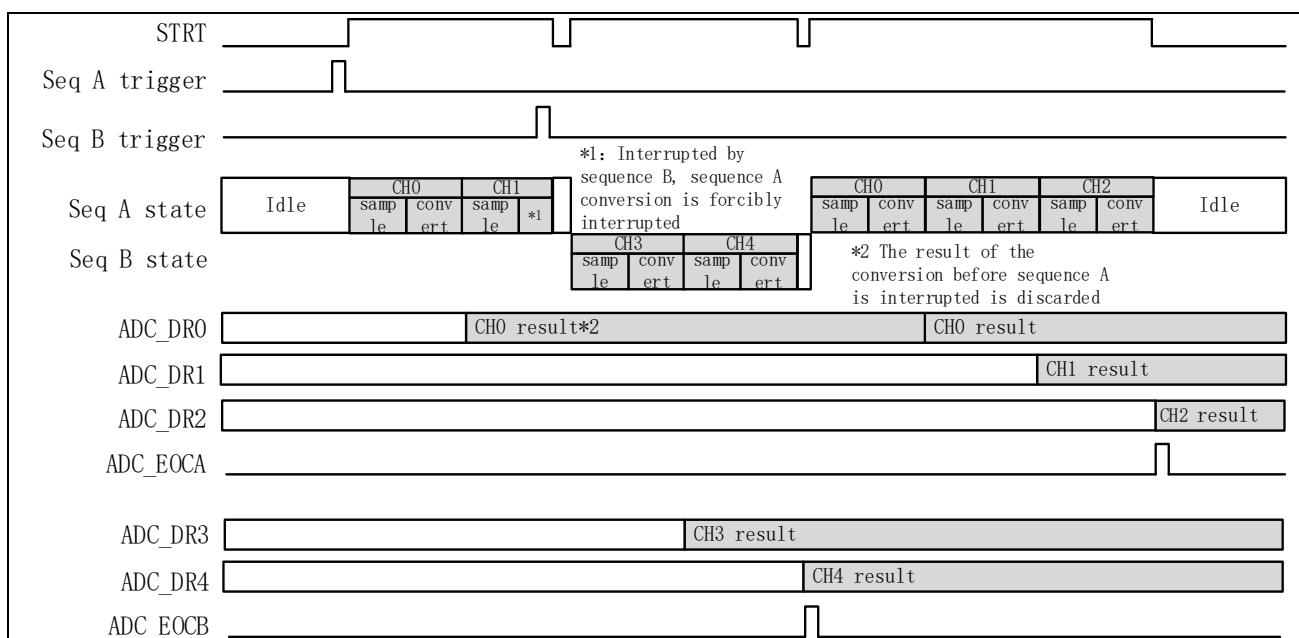


Figure 17-7 Double Sequence Scan Mode (Sequence A restarts from the first channel)

The software process of double sequence scanning mode:

1. Confirm that ADC_STR.STRT is 0, which means that the ADC is in conversion idle state.
2. Set A/D control register ADC_CR0.MS [1:0] to 10b or 11b to select dual sequence scan mode.
3. Set register ADC_CR1.RSCHSEL to select the restart mode after sequence A is broken.
4. Set ADC Sequence A Channel Select Register ADC_CHSELRA.
5. Set ADC Sequence B Channel Select Register ADC_CHSELRB.
6. Set ADC Sampling Time register ADC_SSTR.
7. Set the register ADC_TRGSR to select sequence A and B trigger conditions.

8. Process the converted data after the sequence A or B conversion completes by polling the EOCAF, EOCBF, or ADC_EOCA, or ADC_EOCB interrupt, or starting DMA .

Note:

- Do not select the same channel in sequence A and B. Do not select the same trigger source for sequences A and B.

17.3.7 Analog watchdog function

The analog watchdog function refers to comparing the conversion results at the end of the A/D conversion of the channel. This ADC supports two comparison windows: Compare window 0, Compare window 1. Taking comparison window 0 as an example, as shown in the figure below, if the conversion result is within the protection area, a watchdog comparison interrupt and event ADC_CMP0 will be generated.

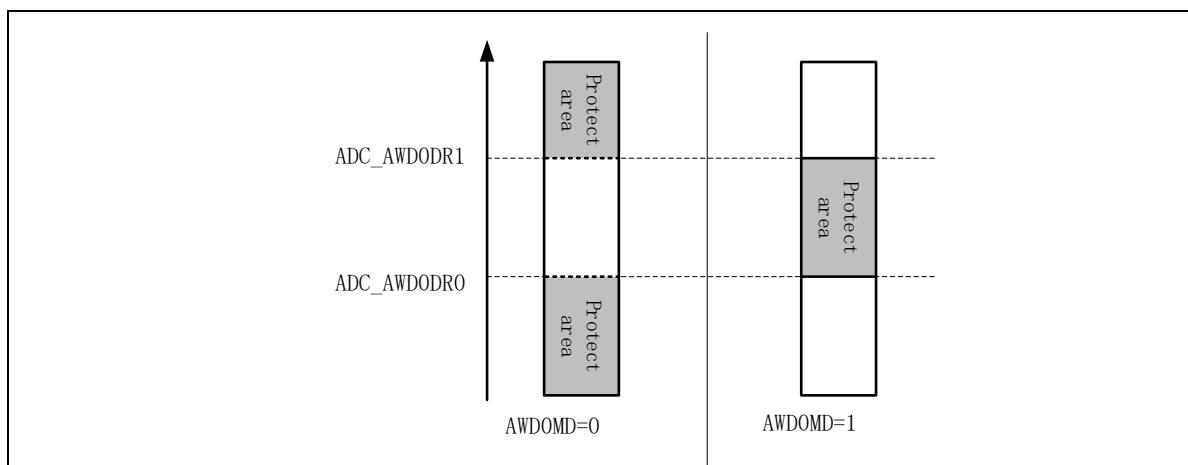


Figure 17-8 Analog Watchdog Protection Area (Compare Mode)

The software flow using the analog watchdog function is illustrated by comparison window 0:

1. Set threshold register ADC_AWD0DR0, ADC_AWD0DR1
2. Set the comparison channel register ADC_AWD0CHSR and select the channel to be compared
3. Set ADC_AWDCR.AWD0MD to select compare mode
4. Set the ADC_AWDCR.AWD0IEN interrupt permission bit.
5. Set ADC_AWDCR.AWD0EN to enable analog watchdog function
6. According to the previous text, set scan mode, start ADC for conversion.
7. Process the converted data after the sequence A or B conversion completes by polling ADC_AWDSR.AWD0F or ADC_EOCB interrupt, or starting DMA.

The comparison window 1 is used the same way as the comparison window 0.

Two comparison windows can be used in combination. When the window combination function is enabled and the conversion of the channel selected by window 1 ends, the output of the comparison interrupt ADC_CMP1 is no longer the comparison result of window 1 alone, but the comparison

results of windows 0 and 1 are logically OR, AND or XOR according to the settings. The software process of using the window combination comparison function is similar to that when the window is used alone. After setting the windows 0 and 1, add the ADC_AWDCR.AWDCM[1:0] register to select the combination mode.

17.3.8 Sampling time and conversion time of analog input

In single scan mode, A/D conversion can be triggered by software, internal trigger IN_TRGx0,1 and external pin trigger ADTRGx. The ADC module starts to sample and convert the analog channel after the scan conversion delay time (t_D). After all the conversion ends, the ADC module enters the idle state after the conversion delay time (t_{ED}), which means a scan ends. The continuous scanning mode is similar to a single scanning mode, except that there is no t_D time at the start of the second and subsequent scans .

The conversion time of a single channel is t_{CONV} ($t_{CONV} = t_{SPL} + t_{CMP}$). t_{SPL} represents the sampling time of the analog input, and the number of sampling cycles can be adjusted by setting register ADC_SSTR according to the input impedance. t_{CMP} represents successive comparison time, 13 PCLK2 for 12-bit precision, 11 PCLK2 for 10-bit precision, and 9 PCLK2 for 8-bit precision.

Time of a scan conversion is t_{SCAN} ($t_{SCAN} = t_D + \sum t_{CONV} + t_{ED}$). $\sum t_{CONV}$ represents the sum of the conversion time of all scanning channels. Since the sampling time t_{SPL} can be set independently, the conversion time t_{CONV} of each channel can be different.

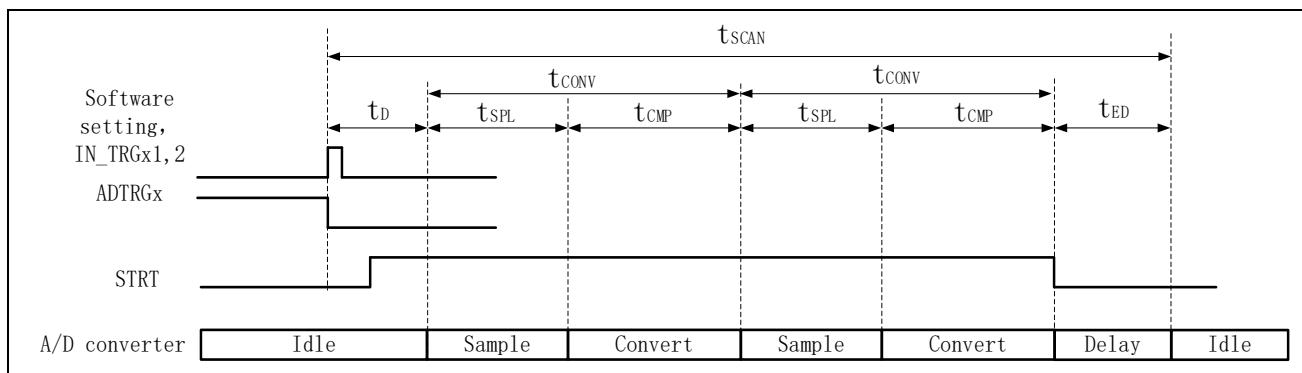


Figure 17-9 A/D conversion time

Table 17-3 AD conversion time

Marking		Note	Conditions				
			Synchronous peripheral trigger	Asynchronous Peripheral Trigger*Note	External pin trigger	Software triggering	
t_D		ADC is idle, start conversion	4 PCLK2	3 PCLK4 + 4 PCLK2 + 1 PCLK4_SYNC	2 PCLK4 + 4 PCLK2	3 PCLK2	
		Interrupted in sequence A conversion, starting sequence B conversion	5 PCLK2	4 PCLK4 + 5 PCLK2 + 1 PCLK4_SYNC	2 PCLK4 + 5 PCLK2	-	
t_{CONV}	t_{SPL}	Sampling time		ADSSTRx.SST [7:0] x PCLK2			
	t_{CMP}	Successive conversion time	12 bit resolution	13 PCLK2			
			10-bit resolution	11 PCLK2			
t_{ED}		Scan completion time		3 PCLK2			
t_{TD}		Minimum continuous trigger interval	$\sum t_{CONV} + 6 \text{ PCLK2}$				

Note:

- Asynchronous peripheral triggering refers to the ADC module selects PLL clock that is asynchronous to the system clock while the peripheral trigger is synchronous to the system clock. At this time, the clock of the peripheral module and the clock of the ADC module are asynchronous. PCLK4_SYNC represents the synchronous clock of the ADC module (that is, the clock set by CMU_SCFGR). At this time, PCLK4 and PCLK2 are the same, and both are asynchronous PLL clocks.

17.3.9 Automatic clearing function of A/D data register

When ADC_CR0.CLREN is '1' , A/D conversion data register ADC_DR is automatically cleared to '0x0000' after being read by the CPU or DMA.

Using this feature, you can detect whether the data register ADC_DR is updated. Examples are provided below.

- When ADC_CR0.CLREN is '0' , that means the automatic clear function is disabled, the analog quantity to be measured (0x0222) is not converted or the result 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 A/D conversion finishes terminating processing will read the unupdated (0x0111). In order to detect whether the A/D conversion value is valid, it is necessary to additionally store the previous conversion value in RAM, and judge by comparing the conversion result.

- If ADC_CR0.CLRE is "1", that means the function of automatic clearing is allowed, after the previous conversion result (0x0111) is read by CPU or DMA, the ADC_DR register will be automatically cleared to "0x0000". After A / D conversion, if the conversion result is not correctly transmitted to the ADC_DR register, ADC_DR register will hold "0x0000". At this time, if "0x0000" is read out in the interrupt processing, it will be easy to judge whether the A / D conversion data is correctly stored.

17.3.10 Converted data average calculation function

The A/D conversion average calculation function refers to the function of continuously performing 2, 4, 8, 16, 32, 64, 128 or 256 conversions on the same channel, and averaging the conversion results and saving them to the data register. Using the average calculation function can remove certain noise components to make the conversion result more accurate.

Register ADC_CR0.AVCNT[2:0] sets the number of continuous conversions, and register ADC_AVCHSEL selects any one or more channels to be averaged.

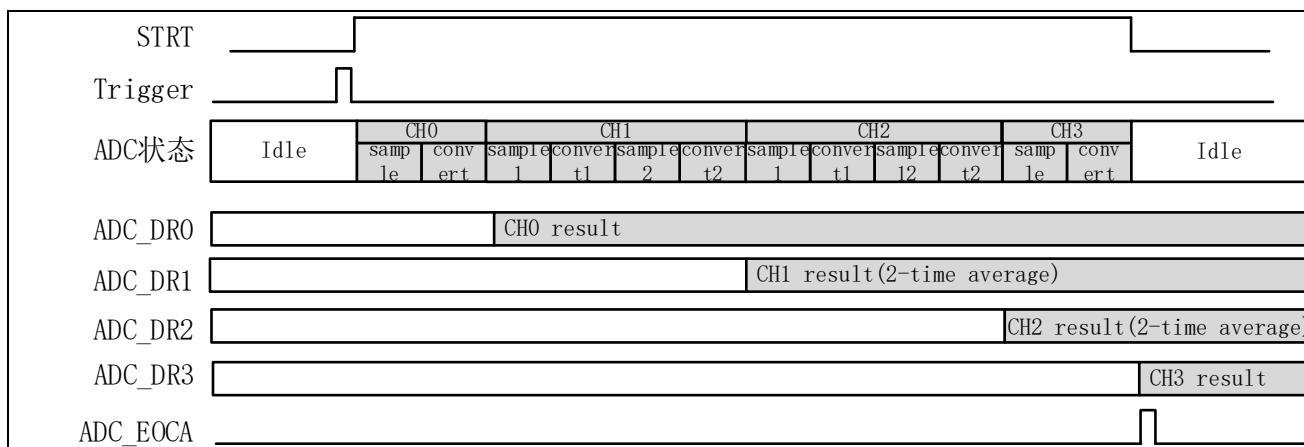


Figure 17-10 Conversion operation when the averaging function is valid

Figure 17-10 shows the single scan mode of sequence A that converts 4 channels CH0~CH3. CH1 and CH2 are set to 2-time average mode. During the scanning process, CH1 and CH2 will be converted twice in a row, and the averaged results will be saved to the data registers ADC_DR1 and ADC_DR2.

17.3.11 Programmable Gain Amplifier PGA

This MCU contains a programmable gain amplifier PGA, and the register ADC_PGACR can be set to enable the PGA circuit and select the gain multiple. The gain range is x2~x32. The analog input is first amplified by the PGA circuit, and then input to the ADC module for conversion.

Before using PGA, please set bit 3 of register PWC_FCG3 to allow PGA, and wait for 2us startup stabilization time.

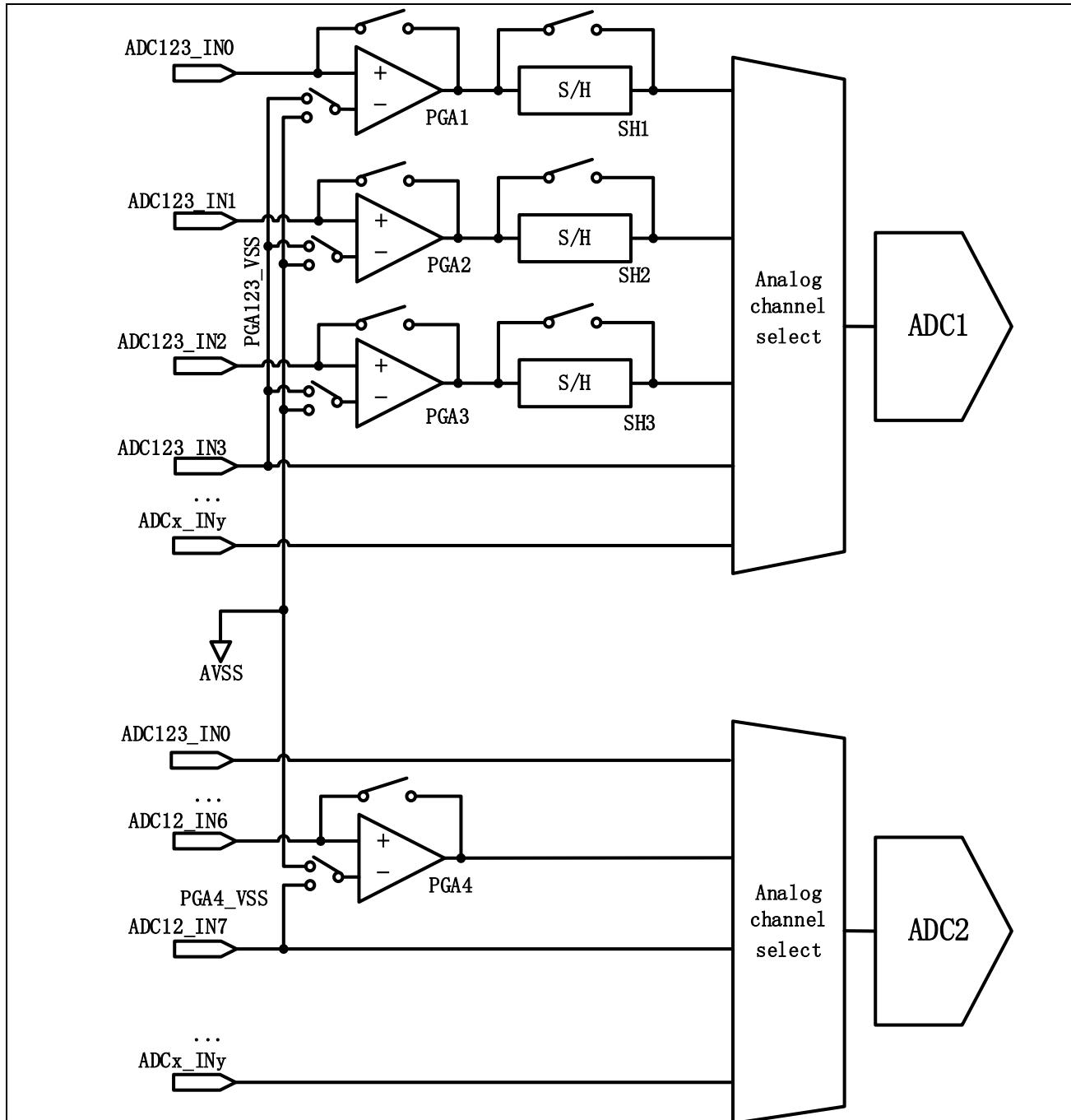


Figure 17-11 Schematic diagram of PGA and SH channels of ADC unit 1 and unit 2

As shown in the figure above, ADC1 includes 3 PGA channels called PGA1~3, which correspond to the physical channels ADC123_IN0~2 respectively. ADC2 includes 1 PGA channel called PGA4, which corresponds to the physical channel ADC12_IN6.

The reference ground of the PGA can be selected from the internal AVSS or the external PGAx_VSS. When selecting the external PGAx_VSS, please connect it to the same voltage as AVSS.

17.3.12 Dedicated sample and hold circuit SH

This MCU contains sample-and-hold circuits (SH) dedicated to three units. When the dedicated sample and hold circuit is enabled, each time the sequence starts, all SH valid channels are sampled at the same time, and then the ADC is started to perform A/D conversion on each channel in the sequence in turn. In continuous scan mode, the sequence inserts the SH sample time at the start of the second and subsequent scans.

Set the register ADC_SHCR.SHSEL bit to enable the SH circuit, and set the register ADC_SHCR.SHSST[7:0] to modify the sampling cycle of the SH circuit.

Before using SH, please set bit 3 of register PWC_FCG3 to allow SH, and wait for 2us startup stabilization time.

Refer to Figure 17-12 , ADC1 contains 3 channels of SH, which can be used in combination with PGA, that is, it is first amplified by PGA, then sampled and held by SH, and input to ADC for conversion. The PGA can also be disabled, the external pin input is directly sampled by SH, and then input to the ADC for conversion.

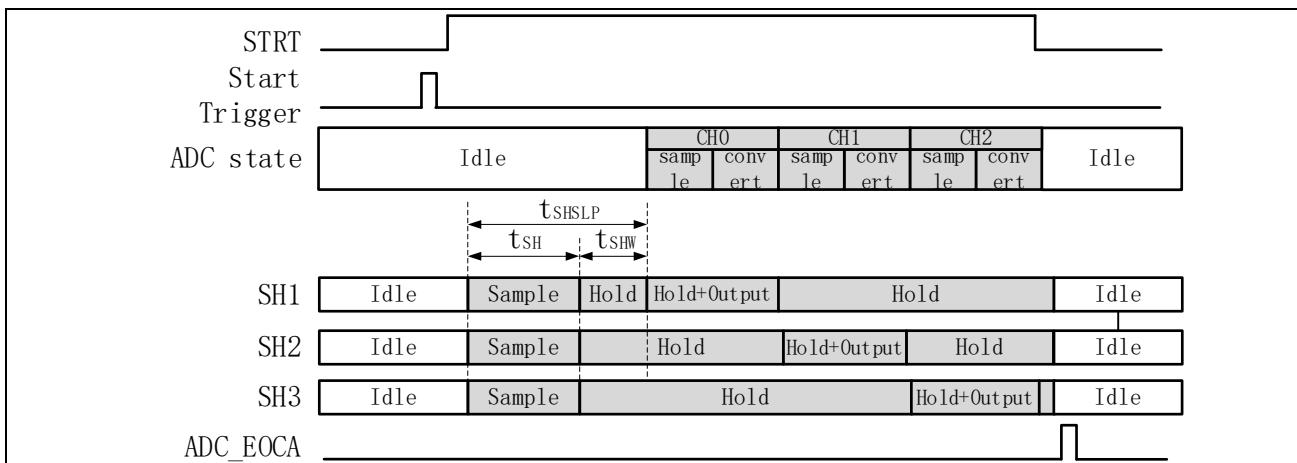


Figure 17-12 Dedicated sample-and-hold circuit is valid

As shown in the figure above, the dedicated sample and hold circuits SH1~3 of CH0~2 are all set to be valid. After the sequence is triggered, the SH1~3 circuits pre-sample the analog inputs of the channels CH0~2 at the same time, that is, the time t_{SHSLP} in the figure. Then perform A/D conversion on the analog voltage held by SH0~2 . $t_{SHSLP} = t_{SH} + t_{SHW}$. t_{SH} represents the sampling time of the dedicated sample and hold circuit, which is set by the register ADC_SHCR.SHSST[7:0], please set it above 0.4 μ s. t_{SHW} represents the waiting time after the sampling of the sampling and holding circuit is completed, which is 12 PCLK2 cycles.

Note:

- Do not use SH in sequence A when in double sequence scan mode. Because the sequence A may be interrupted by the sequence B, the sampling time of each dedicated sample and hold circuit is inconsistent and the unexpected conversion result is output.

- After the register ADC_SHCR.SHSEL[2:0] bit is written to 1, the SH circuit is valid, and the corresponding virtual channel CH0~2 is fixedly mapped to the physical channel ADC123_IN0~2, that is, when the SH circuit is enabled, the corresponding virtual channel does not support channel mapping.

17.3.13 Multi ADC synchronized mode

Since this chip contains two or three ADC uints, the ADC synchronized mode can be used.

In the multi ADC synchronized mode, ADC1 acts as the main control unit to synchronize the conversion of ADC2 and ADC3 through the trigger signal of ADC1. In this mode, all ADC units are just can be triggered by the trigger source selected by the ADC_TRGSR.TRGSELA[2:0] of ADC1, and the ADC_TRGSR.TRGSELA[2:0] setting of ADC2 and ADC3 is invalid. Besides, writing 1 to the ADC_STR.START register will not start conversion, that is, software startup is invalid.

When using multi ADC synchronized mode, please prohibit the sequence B action, so as not to disturb the synchronization.

Two ADC uints, ADC1 and ADC2, or three ADC uints, ADC1, ADC2, and ADC3, can be set to work together. Depending on the product specification, ADC3 may not be installed.

The ADC can be configured to the following four synchronized mode:

- One-Shot Parallel Trigger Mode
- One-Shot Delay Trigger Mode
- Cyclic Parallel Trigger Mode
- Cyclic Delay Trigger Mode

One-Shot Parallel Trigger Mode

The sequence A trigger condition of ADC1 triggers all ADC modules in synchronized mode simultaneously and only once.

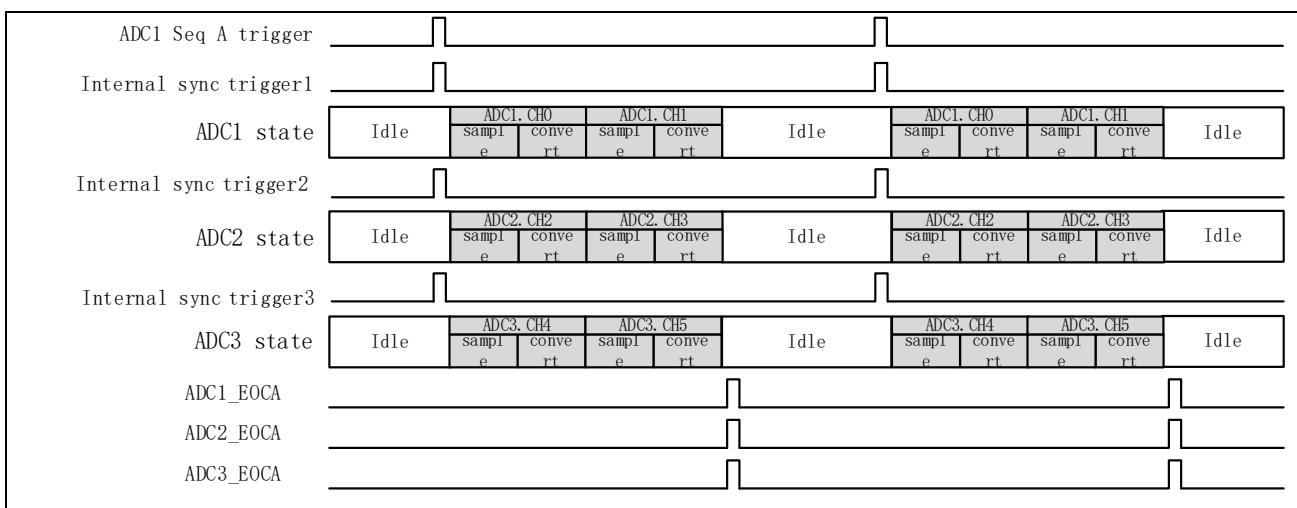


Figure 17-13 Single-Shot Parallel Trigger Mode (Triple ADCs)

Note:

- It is forbidden for multiple ADCs to convert the same analog input at the same time. One analog channel can only sample one ADC module at the same time, otherwise the accuracy is not guaranteed, the same below.

The software setting process of this mode is as follows:

1. Write 0 to Multi ADC Synchronous Mode Control Register ADC_SYNCCR.SYNCEN to confirm that the synchronized mode is disabled.
2. Set up ADC1 module
 - a) Confirm that ADC1_STR.STRT is 0 and ADC1 is in conversion idle state.
 - b) Set the control register ADC1_CR0.MS [1:0] to 00b: Sequence A single scan mode, or 01b: Sequence A continuous scanning mode
 - c) Set Sequence A Channel Select Register ADC1_CHSELRA
 - d) Set sampling time register ADC1_SSTR
 - e) Set sequence A trigger source selection register ADC1_TRGSR
3. Set up the ADC2 module
 - a) Confirm that ADC2_STR.STRT is 0 and ADC2 is in conversion idle state.
 - b) Set the control register ADC2_CR0.MS [1:0], the channel selection register ADC2_CHSELRA, and the channel sampling time register ADC2_SSTR.

Note:

- In order to ensure the synchronous operation of ADC2 and ADC1, the above registers should be set to the same value as the registers of ADC1. 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 consistent.

4. Set ADC3 module (when three ADCs work together)
 - a) Confirm that ADC3_STR.STRT is 0 and ADC3 is in conversion idle state.
 - b) Set the control register ADC3_CR0.MS [1:0], the channel selection register ADC3_CHSELRA, and the channel sampling time register ADC3_SSTR.

Note:

- The 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 the registers of ADC1 as far as possible.

5. Set Multi ADC Synchronous Mode Control Register ADC_SYNCCR.SYNCMD[2:0].
 - 010b: ADC1, ADC2 two ADCs work together.
 - 011b: ADC1, ADC2, ADC3 three ADCs work together.
6. Multi ADC Synchronous Mode Control Register ADC_SYNCCR.SYNCEN is written to 1, and the synchronized work is enabled.
7. Wait for ADC1 sequence A trigger source input, and process the result after ADC1, 2, 3

complete conversion.

One-Shot Delay Trigger Mode

Once ADC1 is triggered by sequence A triggering condition of ADC1, ADC2 will be triggered to start conversion after a set delay, and then the ADC3 will be triggered to start conversion after a set delay. Each ADC module will be triggered only once.

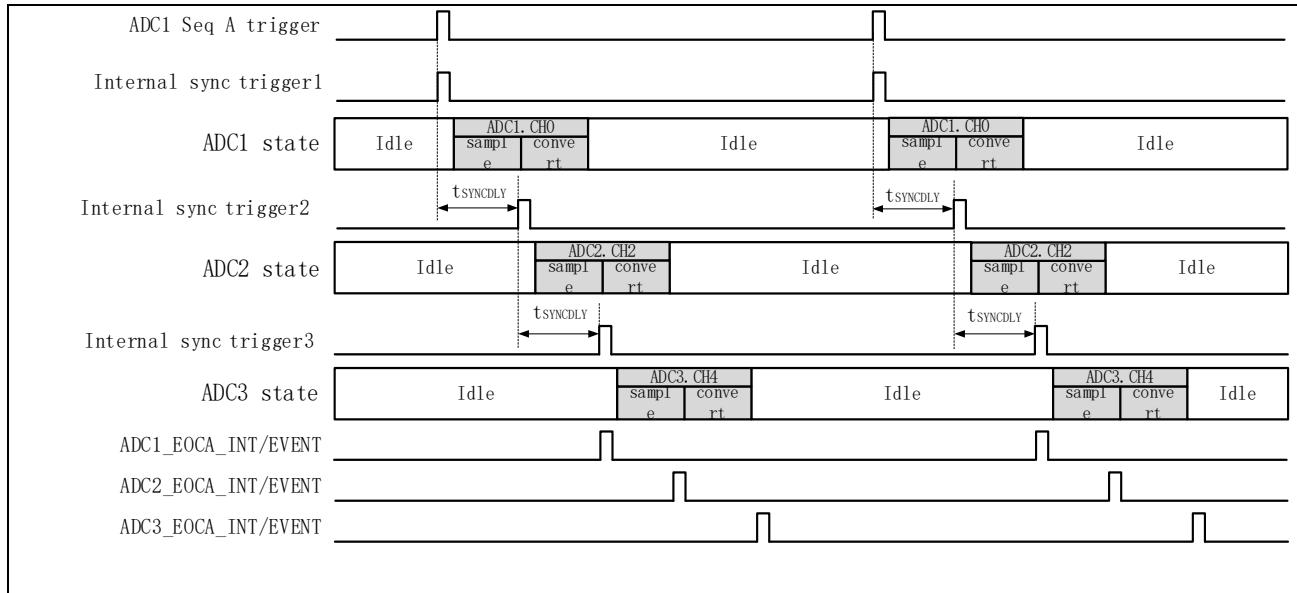


Figure 17-14 Single Delay Trigger Mode (Triple ADC)

Note:

- After the ADC1 sequence A trigger is input for the first time and before the ADC3 co-trigger occurs, inputting the ADC1 sequence A trigger again will be ignored.
- If each ADC unit converts the same analog channel, the sampling time needs to be staggered, that is, the delay time $t_{SYNCDLY}$ and the channel sampling time t_{SPL} must satisfy: $t_{SYNCDLY} > t_{SPL}$.

The software setting process of this mode is as follows:

1. Write 0 to Multi ADC Synchronous Mode Control Register ADC_SYNCCR.SYNCEN to confirm that the synchronized mode is disabled.
2. Set ADC1, 2, 3 modules (refer to single-shot parallel mode)
3. Set Multi ADC Synchronous Mode Control Register ADC_SYNCCR.SYNCDLY[7:0] to set the startup delay of the two ADCs.
4. Set Multi ADC Synchronous Mode Control Register ADC_SYNCCR.SYNCMD[2:0]
 - 000b: ADC1, ADC2 two ADCs work together.
 - 001b: ADC1, ADC2, ADC3 three ADCs work together.
5. Multi ADC Synchronous Mode Control Register ADC_SYNCCR.SYNCEN is written to 1, and the synchronized mode is enabled.
6. Wait for ADC1 sequence A trigger source input, and process the result after ADC1, 2, 3

complete conversion.

Cyclic Parallel Trigger Mode

Once the sequence A trigger condition of ADC1 triggers ADC1, all ADC uints in synchronized mode will be triggered simultaneously after each specified delay. Until the user actively software stops the ADC1 module, or disables the synchronized mode .

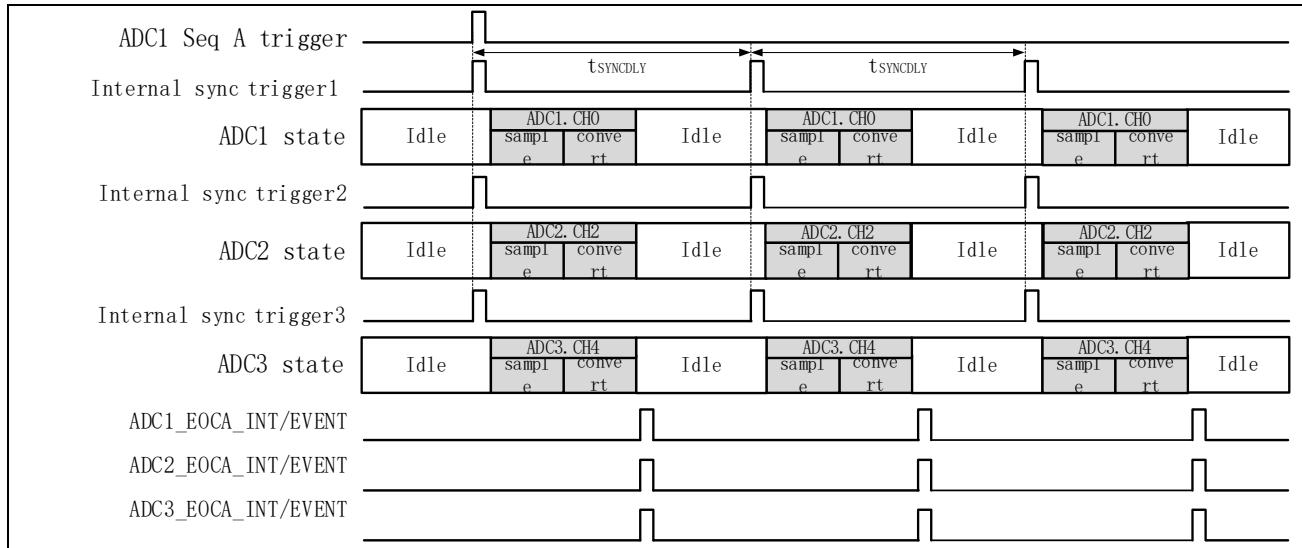


Figure 17-15 Cyclic Parallel Trigger Mode (Triple ADCs)

Note:

- The delay time $t_{SYNCDLY}$ and the time t_{SCAN} of one scan conversion must satisfy:
 $t_{SYNCDLY} > t_{SCAN}$.

The software setting process of this mode is as follows:

1. Write 0 to Multi ADC Synchronous Mode Control Register ADC_SYNCCR.SYNCEN to confirm that the synchronized mode is disabled.
2. Set ADC1, 2, 3, refer to single-shot parallel mode. ADC_CR0.MS [1:0] is set to 00b: Sequence A single scan mode
3. Set Multi ADC Synchronous Mode Control Register ADC_SYNCCR.SYNCDLY[7:0] to set the delay of each parallel trigger.
4. Set Multi ADC Synchronous Mode Control Register ADC_SYNCCR.SYNCMD[2:0]
110b: ADC1, ADC2 two ADCs work together.
111b: ADC1, ADC2, ADC3 three ADCs work together.
5. Multi ADC Synchronous Mode Control Register ADC_SYNCCR.SYNCEN is written to 1, and the synchronized mode is enabled.
6. Wait for ADC1 sequence A trigger source input, and process the result after ADC1, 2, 3 complete conversion.

Cyclic Delay Trigger Mode

Once the sequence A trigger condition of ADC1 triggers ADC1, ADC2, ADC3, ADC1, ADC2... are continuously triggered in sequence after each set delay until the user actively software stops the ADC1 module, or disables the synchronized mode.

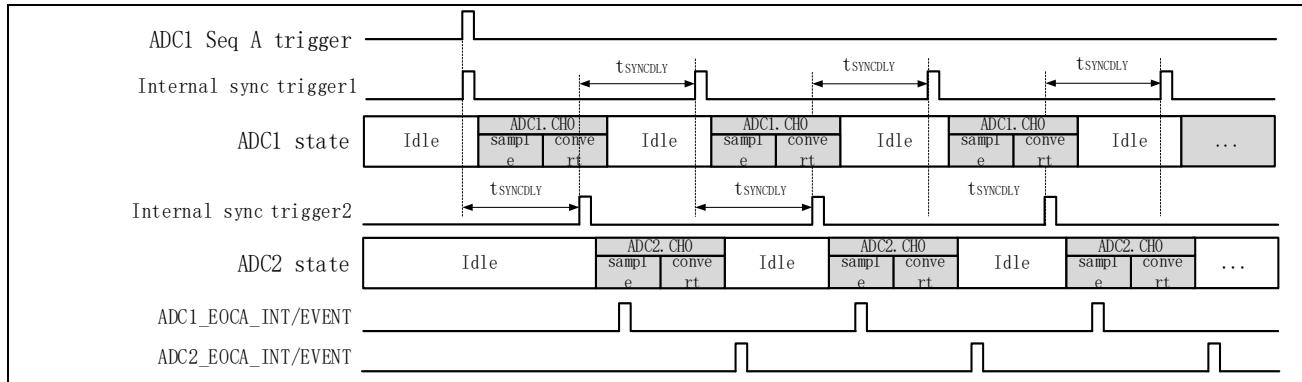


Figure 17-16 Cyclic Delay Trigger Mode (Two ADCs)

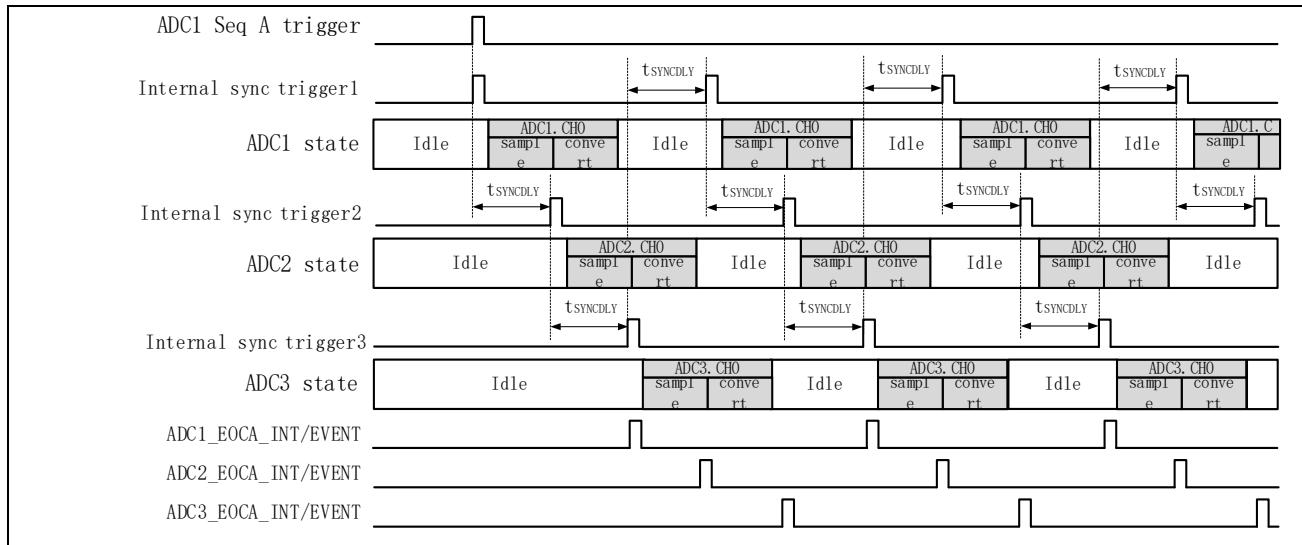


Figure 17-17 Cyclic Delay Trigger Mode (Triple ADC)

Note:

- When two ADCs work together, the delay time $t_{SYNCMDLY}$ and the time t_{SCAN} of one scan conversion must satisfy: $t_{SYNCMDLY} > t_{SCAN}/2$. When the three ADCs work together, the following conditions must be met: $t_{SYNCMDLY} > t_{SCAN}/3$. At the same time, if ADC1, ADC2, and ADC3 convert the same analog channel, the sampling time needs to be staggered, that is, $t_{SYNCMDLY} > t_{SPL}$.

The software setting process of this mode is as follows:

- Write 0 to Multi ADC Synchronous Mode Control Register ADC_SYNCCR.SYNCEN to confirm that the synchronized mode is disabled.
- Set ADC1, 2, 3 modules, reference loop parallel trigger mode.
- Set Multi ADC Synchronous Mode Control Register ADC_SYNCCR.SYNCMDLY[7:0] to set the

delay of each trigger.

4. Set Multi ADC Synchronous Mode Control Register ADC_SYNCCR.SYNCMD[2:0]
100b: ADC1, ADC2 two ADCs work together.
101b: ADC1, ADC2, ADC3 three ADCs work together.
5. Multi ADC Synchronous Mode Control Register ADC_SYNCCR.SYNCEN is written to 1, and the synchronized mode is enabled.
6. Wait for ADC1 sequence A trigger source input, and process the result after ADC1, 2, 3 complete conversion.

17.3.14 Interrupt and event signal output

The ADC module can produce the following four types of event outputs. When each event occurs and the corresponding interrupt permission register is valid, the interrupt application is also output.

1. Sequence A Scan End ADC_EOCA, corresponding to interrupt enable register ADC_ICR.EOCAIEN
2. Sequence B Scan End ADC_EOCB, corresponding to interrupt enable register ADC_ICR.EOCBIEN
3. Analog watchdog 0 ADC_CMP0, corresponding to interrupt enable register ADC_AWDCR.AWD0IEN
4. Analog watchdog 1 ADC_CMP1, corresponding to interrupt enable register ADC_AWDCR.AWD1IEN

These four event outputs can start other on-chip peripheral modules, including DMA transfer. DMA transfer can read A/D conversion results continuously without software intervention and completely implemented by hardware, which reduce CPU load. For DMA settings, refer to the DMA description section. The event signal output has nothing to do with the control of the interrupt enable bit, and will output as long as the condition occurs.

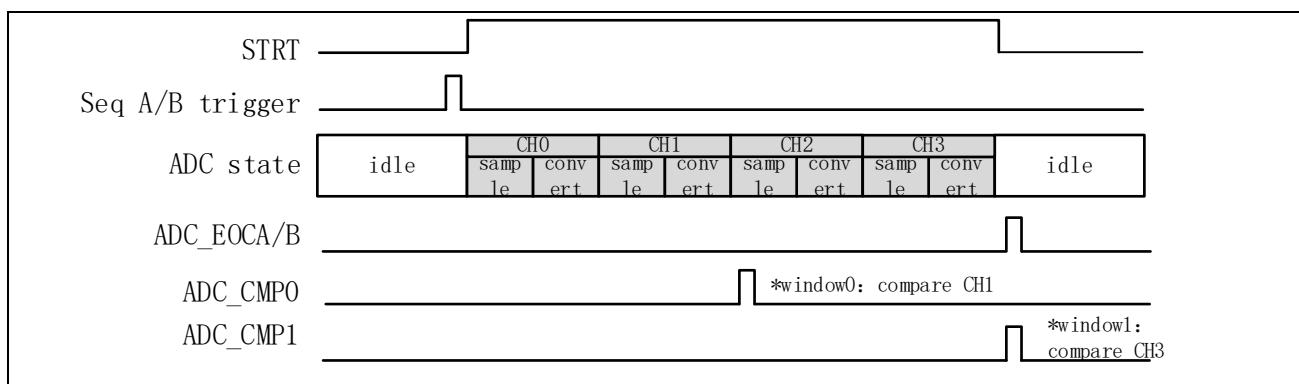


Figure 17-18 ADC Interrupt and Event Output Timing

17.4 Register description

17.4.1 Register overview

Unit 1 BASE_ADDR: 0x40040000

Unit 2 BASE_ADDR: 0x40040400

Unit 3 BASE_ADDR: 0x40040800

Table 17-4 ADC Register List 1/2

Register name	Symbol	Offset address	Bit width	Reset value
ADC Start Register	ADC_STR	0x00	8	0x00
ADC Control Register 0	ADC_CR0	0x02	16	0x0000
ADC Control Register 1	ADC_CR1	0x04	16	0x0000
ADC Trigger Select Register	ADC_TRGSR	0x0a	16	0x00000000
ADC Sequence A Channel Select Register	ADC_CHSELRA	0x0c	32	0x00000000
ADC Sequence B Channel Select Register	ADC_CHSELRB	0x10	32	0x00000000
ADC Average Channel Select Register	ADC_AVCHSELR	0x14	32	0x00000000
ADC Extend Channel Select Register	ADC_EXCHSELR	0x18	8	0x00
ADC Sampling Time register	ADC_SSTRx	0x20+x	8	0x0b
	ADC_SSTRL	0x30	8	0x0b
ADC Channel Mapping Register 0	ADC_CHMUXR0	0x38	16	0x3210
ADC Channel Mapping Register 1	ADC_CHMUXR1	0x3a	16	0x7654
ADC Channel Mapping Register 2	ADC_CHMUXR2	0x3c	16	0xba98
ADC Channel Mapping Register 3	ADC_CHMUXR3	0x3e	16	0xfedc
ADC Interrupt Flag Register	ADC_ISR	0x44	8	0x00
ADC Interrupt Enable Register	ADC_ICR	0x45	8	0x03
ADC Interrupt Flag Clear Register	ADC_ISCLRR	0x46	8	0x00
Multi ADC Synchronous Mode Control Register	ADC_SYNCCR	0x4c	16	0x0c00
ADC Channel Convert Data Register	ADC_DRy	0x50+2*y	16	0x0000
ADC Analog Watch Dog Control Register	ADC_AWDCR	0xa0	16	0x0000
ADC Analog Watch Dog Status Register	ADC_AWDSR	0xa2	8	0x00
ADC Analog Watch Dog Status Clear Register	ADC_AWDSCLRR	0xa3	8	0x00
ADC Analog Watch Dog Compare 0 Data Register	ADC_AWD0DR0	0xa4	16	0x0000
	ADC_AWD0DR1	0xa6	16	0xffff
ADC Analog Watch Dog 0 Channel Select Register	ADC_AWD0CHSR	0xa8	8	0x00
ADC Analog Watch Dog 1 Compare Data Register	ADC_AWD1DR0	0xac	16	0x0000
	ADC_AWD1DR1	0xae	16	0xffff
ADC Analog Watch Dog 1 Channel Select Register	ADC_AWD1CHSR	0xb0	8	0x00

Register name	Symbol	Offset address	Bit width	Reset value
ADC Dedicated SH Control Register	ADC_SHCR	0x1a	16	0x0018
ADC PGA 1 Control Register	ADC_PGACR1	0xc0	8	0x00
ADC PGA 2 Control Register	ADC_PGACR2	0xc1	8	0x00
ADC PGA 3 Control Register	ADC_PGACR3	0xc2	8	0x00
ADC PGA VSS Source Select Register	ADC_PGAVSSEN	0xc4	8	0x00

Table 17-5 ADC Register List 2/2

Register name	Symbol	Address	Bit width	Reset value
ADC1 trigger source select register0	ADC1_ITRGSEL0	0x40010894	32	0x000001ff
ADC1 trigger source select register1	ADC1_ITRGSEL1	0x40010898	32	0x000001ff
ADC2 trigger source select register0	ADC2_ITRGSEL0	0x4001089c	32	0x000001ff
ADC2 trigger source select register1	ADC2_ITRGSEL1	0x400108a0	32	0x000001ff
ADC3 trigger source select register0	ADC3_ITRGSEL0	0x400108a4	32	0x000001ff
ADC3 trigger source select register1	ADC3_ITRGSEL1	0x400108a8	32	0x000001ff

17.4.2 ADC Start Register (ADC_STR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
Bit	Symbol	Bit name	Description				Read and write
b7-b1	Reserved	—	Read 0, write 0				R/W
			0: Stop conversion 1: Start conversion Set condition: (1) Software setup (2) The selected trigger condition occurs. (3) A/D converting Reset condition: (1) Software write "0" (2) cleared to 0 automatically after conversion.				
b0	STRT	Start AD conversion	Note: <ul style="list-style-type: none"> — Writing 1 will generate a software trigger when STRT is 0 (ADC is idle), and then start sequence A. — Writing 1 is invalid when STRT is 1 (in ADC action). — Writing 0 when STRT is 1 will force the AD conversion to stop. If ADC_TRGSR is set to a value which is not 0x0 , while we do not want the ADC to restart, please set ADC_TRGSR to 0 first, and then write 0 to STRT. — Write 0 is invalid when STRT is 0. 				R/W

17.4.3 ADC 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]
<hr/>							
Bit	Symbol	Bit name	Description				Read and write
b15-b12	Reserved	—	Read 0, write 0				R/W
b10-b8	AVCNT[2:0]	Time selection	0 0 0: Average of 2 consecutive conversions 0 0 1: Average of 4 consecutive conversions 0 1 0: Average of 8 consecutive conversions 0 1 1: Average of 16 consecutive conversions 1 0 0: Average of 32 consecutive conversions 1 0 1: Average of 64 consecutive conversions 1 1 0: Average of 128 consecutive conversions 1 1 1: Average of 256 consecutive conversions				R/W
b7	DFMT	Data format	0: Right alignment of converted data 1: Left alignment of converted data				R/W
b6	CLREN	Data register automatic clear	0: Disable automatic clear 1: Enable automatic clear Note: After the CLREN bit is set, the register ADC_DRx will be automatically cleared after being read by the CPU, DMA, etc. The automatic clearing function is mainly used to detect whether the data register is updated.				R/W
b5-b4	ACCSEL[1:0]	Resolution selection	00: 12-bit resolution 01: 10-bit resolution 10: 8-bit resolution 11: Prohibitions				R/W
b3-b2	Reserved	—	Read 0, write 0				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: Sequence A single scan mode, Sequence B single scan mode 1 1: Sequence A continuous scan mode, sequence B single scan mode				R/W

Note:

- Set this register when ADC_STR. STRT is "0".

17.4.4 ADC 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	—	—

Bit	Symbol	Bit name	Description	Read and write
b15-b3	Reserved	—	Read 0, write 0	R/W
b2	RSCHSEL	Sequence A restart channel selection	0: After interrupted by sequence B, sequence A restarts to scan from the interrupted channel 1: After interrupted by sequence B, sequence A restarts to scan from the first channel	R/W
b1-b0	Reserved	—	Read 0, write 0	R/W

Note:

- Set this register when ADC _ STR. STRT is "0".

17.4.5 ADC Trigger Select Register (ADC_TRGSR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
TRGENB	—	—	—	—	—	TRGSELB[1]	TRGSELB[0]
B7	b6	b5	b4	b3	b2	b1	b0
TRGENA	—	—	—	—	—	TRGSELB[1]	TRGSELB[0]

Bit	Symbol	Bit name	Description	Read and write
b15	TRGENB	Sequence B trigger enable	0: Disable sequence B in-chip or external pin trigger 1: Enable sequence B in-chip or external pin trigger Note: When external pin trigger is valid, if ADTRGx changes from "High" to "Low" and a falling edge is detected, scan conversion starts, please keep "Low" for more than 1.5*PCLK4 cycles.	R/W
b14-b10	Reserved	—	Read 0, write 0	R/W
b9-b8	TRGSELB[1:0]	Sequence B trigger condition selection	When sequence B is valid (ADC_CRO.MS [1] = 1), trigger condition of sequence B 00b: ADTRGx 01b: IN_TRGx0 10b: IN_TRGx1 11b: IN_TRGx0 + IN_TRGx1 Note: Only valid when ADC_CRO.MS [1] = 1. Other mode settings are invalid. The interval between triggers must be greater than or equal to the scan period t_{SCAN} , and if less, the trigger is invalid.	R/W
B7	TRGENA	Sequence A trigger enable	0: Disable sequence A in-chip or external pin trigger 1: Enable sequence A in-chip or external pin trigger Note: When external pin trigger is valid, if ADTRGx changes from "High" to "Low" and a falling edge is detected, scan conversion starts, please keep "Low" for more than 1.5*PCLK4 cycles.	R/W
b6-b2	Reserved	—	Read 0, write 0	R/W
b1-b0	TRGSELB[1:0]	Sequence A trigger condition selection	Trigger condition for sequence A. 00b: ADTRGx (x=1~3, represents ADC unit number) 01b: IN_TRGx0 10b: IN_TRGx1 11b: IN_TRGx0 + IN_TRGx1 Note: When the ADC is idle and write 1 software trigger to ADC_STR.START, the A/D conversion will directly start and ignore the settings of TRGENA, TRGSELB[1:0]. The interval between triggers must be greater than or equal to the scan period t_{SCAN} , and if less, the trigger is invalid.	R/W

Note:

- Set this register when ADC _ STR. START is "0".

17.4.6 ADC Trigger Source Select Register (ADC_ITRGSEL0, ADC_ITRGSEL1)

Reset value: 0x0000001ff

b31	b30	b29	b28	b27	b26	b25	b24
COMEN[1]	COMEN[0]	—	—	—	—	—	—
b23	b22	b21	b20	b19	b18	b17	b16
—	—	—	—	—	—	—	—
b15	b14	b13	b12	b11	b10	b9	b8
—	—	—	—	—	—	—	TRGSEL[8]
B7	b6	b5	b4	b3	b2	b1	b0
TRGSEL[7]	TRGSEL[6]	TRGSEL[5]	TRGSEL[4]	TRGSEL[3]	TRGSEL[2]	TRGSEL[1]	TRGSEL[0]

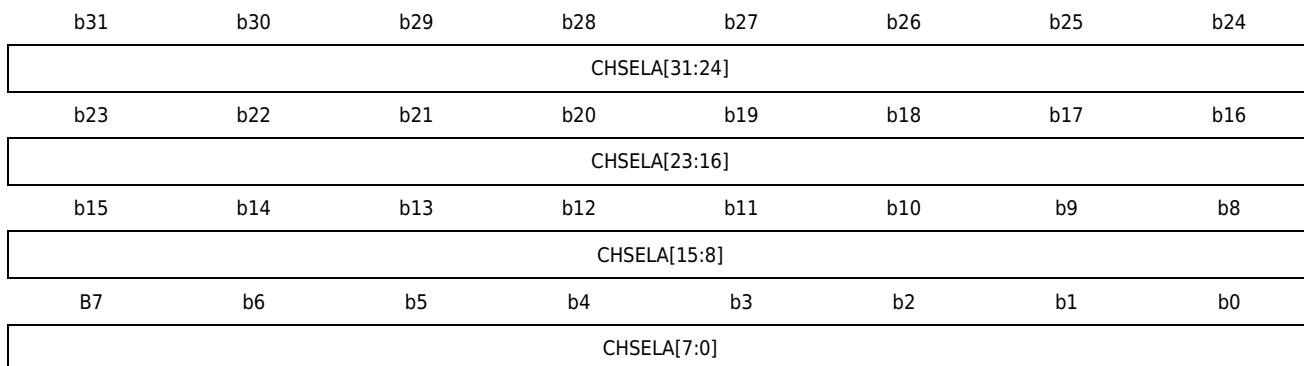
Bit	Symbol	Bit name	Description	Read and write
b31	COMEN[1]	public trigger enable	0: Disable the common trigger event selected by register AOS_COMTRG2 in internal trigger IN_TRGx0,1 1: Enable the common trigger event selected by register AOS_COMTRG2 in internal trigger IN_TRGx0,1 For common trigger events, please refer to the AOS chapter	R/W
b30	COMEN[0]	public trigger enable	0: Disable the common trigger event selected by register AOS_COMTRG1 in internal trigger IN_TRGx0,1 1: Enable the common trigger event selected by register AOS_COMTRG1 in internal trigger IN_TRGx0,1 For common trigger events, please refer to the AOS chapter	R/W
b29-b9	Reserved	—	Read 0, write 0	R/W
b8-b0	TRGSEL[8:0]	On-chip trigger source selection	refer to [Note:] to write the selected event number. According to COMEN[1:0] setting, IN_TRGx0,1 can contain up to 3 trigger sources selected by TRGSEL[8:0], AOSCOMTRG0, AOSCOMTRG1.	R/W

Note:

- Trigger source IN_TRGx0 is set by register ADCx_ITRGSEL0
- Trigger source IN_TRGx1 is set by register ADCx_ITRGSEL1

17.4.7 ADC Sequence A Channel Select Register(ADC_CHSELRA)

Reset value: 0x00000000



Bit	Symbol	Bit name	Description	Read and write
b31-b0	CHSEL[A31:0]	Converting channel selection	Channel selection for sequence A, each bit representing a channel, CHSEL[A31:0] representing a channel CHx and can set one or more channels. 0: No channel selected 1: Select Corresponding Channel The corresponding bit of no channel is the Reserved bit, which is 0 when read and 0 is written when writing. Note: Please do not select the same channel in Sequence A and Sequence B.	R/W

Note:

- Set this register when ADC _ STRT is "0".

17.4.8 ADC Sequence B Channel Select Register (ADC_CHSELRB)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24
CHSELB[31:24]							
b23	b22	b21	b20	b19	b18	b17	b16
CHSELB[23:16]							
b15	b14	b13	b12	b11	b10	b9	b8
CHSELB[15:8]							
B7	b6	b5	b4	b3	b2	b1	b0
CHSELB[7:0]							

Bit	Symbol	Bit name	Description	Read and write
b31-b0	CHSELB[31:0]	Converting channel selection	Channel selection for sequence B, each bit representing a channel, CHSELB [x] representing a channel CHx and can set one or more channels. Only valid in double sequence scan mode. 0: No channel selected 1: Select Corresponding Channel The corresponding bit of no channel is the Reserved bit, which is 0 when read and 0 is written when writing. Note: Please do not select the same channel in Sequence A and Sequence B.	R/W

Note:

- Set this register when ADC _ STR. STRT is "0".

17.4.9 ADC Average Channel Select Register (ADC_AVCHSELR)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24
AVCHSEL[31:24]							
b23	b22	b21	b20	b19	b18	b17	b16
AVCHSEL[23:16]							
b15	b14	b13	b12	b11	b10	b9	b8
AVCHSEL[15:8]							
B7	b6	b5	b4	b3	b2	b1	b0
AVCHSEL[7:0]							

Bit	Symbol	Bit name	Description	Read and write
b31-b0	AVCHSEL[31:0]	Average channel selection	<p>Each bit represents a channel, AVCHSEL[x] represents channel CHx, and any combination can be selected.</p> <p>0: No channel selected</p> <p>1: Select Corresponding Channels</p> <p>The corresponding bit of no channel is the Reserved bit, which is 0 when read and 0 is written when writing.</p> <p>Note: When AVCHSEL and the corresponding channel of ADC_CHSELRA or ADC_CHSELRB are selected at the same time, the channel will continuously perform A/D conversion for setting times during scanning, and the conversion results will be averaged and then updated into the data register. If the corresponding channel AVCHSEL is not set, the channel will perform a normal one-time conversion.</p>	R/W

Note:

- Set this register when ADC _ STR. STRT is "0".

17.4.10 ADC Extend Channel Select Register (ADC_EXCHSEL R)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
—	—	—	—	—	—	—	EXCHSEL

Bit	Symbol	Bit name	Description	Read and write
b7-b1	—	—	Read 0, write 0	R/W
b0	EXCHSEL	Extended channel selection	ADC _x _IN15 and on-chip analog channel selection 0: select external pin ADC _x _IN15 1: Select an internal analog channel. For the setting of an internal analog source, refer to the [Power controller (PWC)]	R/W

Note:

- Set this register when ADC _ STR. STRT is "0".

17.4.11 ADC Sampling Time register (ADC_SSTR x)

Reset value: 0x0b

B7	b6	b5	b4	b3	b2	b1	b0
SST[7:0]							

Bit	Symbol	Bit name	Description	Read and write
b7-b0	SST[7:0]	Number of sampling cycles	The number of sampling cycles can be set from 5 to 255 cycles. CH0~15 are set by ADC_SSTRx, x=0~15, and other channels are set by ADC_SSTRL. Note: The initial conversion state has 11 sampling cycles. If the frequency of PCLK2 is 50MHz, one sampling cycle is 20ns, so the sampling time is 0.22us. When the external input impedance R _{AIN} is too large and the sampling time is insufficient or the frequency of PCLK2 is low, the register can be set to adjust the sampling time. The sampling time should not be less than 5 cycles. $SST \geq (R_{AIN} + R_{ADC}) * C_{ADC} * \ln(2^{N+2}) * f_{ADC} + 1$ R _{AIN} represents external input impedance (Ω), R _{ADC} represents internal sampling switch resistance (Ω), C _{ADC} represents internal sample and hold capacitor (F), N represents AD resolution (12/10/8), f _{ADC} represents PCLK2 frequency (Hz). Refer to the description of electrical characteristics.	R/W

Note:

- Set this register when ADC _ STR. STRT is "0".
- The sampling time of the internal reference voltage should not be less than 1us.
- The sampling time of VBAT dividing voltage should not be less than 4us.
- When the dedicated sample and hold circuit SH is valid, the sampling time of the corresponding channel should not be less than 0.4 us

17.4.12 ADC Channel Mapping Register 0 (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]			

Bit	Symbol	Bit name	Description				Read and write
			The corresponding bit of the channel that does not exist is 0 when read, and 0 is written when writing				
			The mapping relationship to different ADC units CHx is as follows:				
CHxMUX[3:0]	Channel x mapping selection x=0~15	Set value	ADC1 mapping object	ADC2 mapping object	ADC3 mapping object		
		0x0	ADC123_IN0	ADC123_IN0	ADC123_IN0		
		0x1	ADC123_IN1	ADC123_IN1	ADC123_IN1		
		0x2	ADC123_IN2	ADC123_IN2	ADC123_IN2		
		0x3	ADC123_IN3	ADC123_IN3	ADC123_IN3		
		0x4	ADC12_IN4	ADC12_IN4	ADC3_IN4		
		0x5	ADC12_IN5	ADC12_IN5	ADC3_IN5		
		0x6	ADC12_IN6	ADC12_IN6	ADC3_IN6		
		0x7	ADC12_IN7	ADC12_IN7	ADC3_IN7		
		0x8	ADC12_IN8	ADC12_IN8	ADC3_IN8		
		0x9	ADC12_IN9	ADC12_IN9	ADC3_IN9		
		0xa	ADC123_IN10	ADC123_IN10	ADC123_IN10		
		0xb	ADC123_IN11	ADC123_IN11	ADC123_IN11		
		0xc	ADC123_IN12	ADC123_IN12	ADC123_IN12		
		0xd	ADC123_IN13	ADC123_IN13	ADC123_IN13		
		0xe	ADC12_IN14	ADC12_IN14	ADC3_IN14		
		0xf	ADC12_IN15	ADC12_IN15	ADC3_IN15		

Note: Please do not set to non-existing analog input.

R/W

Note:

- Set this register when ADC _ STR. STRT is "0".

17.4.13 ADC Interrupt Flag Register (ADC_ISR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
—	—	—	SASTPDF	—	—	EOCBF	EOCAF
<hr/>							
Bit	Symbol	Bit name	Description				Read and write
b7-b5	Reserved	—	Read 0, write 0				R/W
b4	SASTPDF	Sequence A is interrupted flag	Set the flag when the sequence A conversion is interrupted by the high-priority sequence B in double-sequence scan mode, This register bit is read-only				R
b3-b2	Reserved	—	Read 0, write 0				R/W
b1	EOCBF	Sequence B conversion completion flag	Set the flag when the conversion of all channels selected in Sequence B completes This register bit is read-only				R
b0	EOCAF	Sequence A conversion completion flag	Set the flag when the conversion of all channels selected in Sequence A completes This register bit is read-only				R

17.4.14 ADC Interrupt Enable Register (ADC_ICR)

Reset value: 0x03

B7	b6	b5	b4	b3	b2	b1	b0
—	—	—	—	—	—	EOCBIEN	EOCAIEN
<hr/>							
Bit	Symbol	Bit name	Description				Read and write
b7-b2	Reserved	—	Read 0, write 0				R/W
b1	EOCBIEN	Sequence B conversion complete interrupt enable	0: Disable sequence B conversion complete interrupt 1: Enable sequence B conversion complete interrupt				R/W
b0	EOCAIEN	Sequence A conversion complete interrupt enable	0: Disable sequence A conversion complete interrupt 1: Enable sequence A conversion complete interrupt				R/W

17.4.15 ADC Interrupt Flag Clear Register (ADC_ISCLRR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
—	—	—	CLRSASTPDF	—	—	CLREOCBF	CLREOCAF
<hr/>							
Bit	Symbol	Bit name	Description			Read and write	
b7-b5	Reserved	—	Read 0, write 0			R/W	
b4	CLRSASTPDF	Sequence A is interrupted flag reset	Writing 0 has no effect, writing 1 resets the SASTPDF status bit, Read always 0			R/W	
b3-b2	Reserved	—	Read 0, write 0			R/W	
b1	CLREOCBF	Sequence B conversion complete flag reset	Write 0 has no effect. Write 1 resets the EOCBF status bit. Read always 0			R/W	
b0	CLREOCAF	Sequence A conversion complete flag reset	Write 0 has no effect. Write 1 resets the EOCAF status bit. Read is always 0.			R/W	

17.4.16 Multi ADC Synchronous Mode Control Register (ADC_SYNCCR)

Reset value: 0x0c00

b15	b14	b13	b12	b11	b10	b9	b8
SYNCDLY[7:0]							
B7	b6	b5	b4	b3	b2	b1	b0
—	SYNCMD[2]	SYNCMD[1]	SYNCMD[0]	—	—	—	SYNCEN

Bit	Symbol	Bit name	Description	Read and write
b15-b8	SYNCDLY[7:0]	Sync delay time	<p>In delayed trigger mode, the startup delay time of two ADC is $t_{SYNCDLY}$. 0x1 means $t_{SYNCDLY} = 1 \times PCLK2$, 0xff means $t_{SYNCDLY} = 255 \times PCLK2$. Note: Set the 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 errors caused by multiple ADCs in the sampling state at the same time, and to prevent the ADC from triggering again before the conversion completed and resulting in synchronization failure. The recommended settings are as follows:</p> <p>Single Delay Trigger Mode: $t_{SYNCDLY} > t_{SPL}$</p> <p>Two ADC Cyclic Delay trigger mode: $t_{SYNCDLY} > t_{SPL}$, and $t_{SYNCDLY} > t_{SCAN} / 2$</p> <p>Three ADC Cyclic Delay Trigger Modes: $t_{SYNCDLY} > t_{SPL}$, and $t_{SYNCDLY} > t_{SCAN} / 3$</p> <p>Single-Shot Parallel Trigger Mode: The setting of this register is invalid.</p> <p>Cyclic Parallel Trigger Mode: $t_{SYNCDLY} > t_{SCAN}$</p>	R/W
B7	Reserved	—	Read 0, write 0	R/W
b6-b4	SYNCMD[2:0]	Sync mode selection	<p>SYNCMD[2] 0: single trigger 1: Cyclic trigger SYNCMD[1] 0: Delay trigger mode 1: Parallel trigger mode SYNCMD[0] 0: ADC1 and ADC2 work synchronously, ADC3 works independently 1: ADC1, ADC2 and ADC3 work synchronously</p> <p>Note: Set the register when SYNCEN is "0". When using single trigger, please set the ADC which is to be synchronized to sequence A single scan mode, or sequence A continuous scan mode. When using cyclic trigger mode, set the ADC to sequence A single scan mode.</p>	R/W
b3-b1	Reserved	—	Read 0, write 0	R/W
b0	SYNCEN	Synchronous Mode enable	<p>0: Disable Sync mode 1: Enable Sync mode</p> <p>Note: Synchronous mode only supports sequence A. Before writing 1 to SYNCEN, please turn off sequence B of several ADCs involved in synchronization (ADC_CR0.MS [1] = 0), select the same number of channels for sequence A, and set the same channel sampling time ADC_SSTRx. The process is in order to avoid the inconsistency of each ADC scan time t_{SCAN} which will cause subsequent synchronization failure. When software writes 0 to ADC1_STR STRT to force the conversion to stop, SYNCEN is automatically cleared to 0.</p>	R/W

Note:

- This register is only carried in the main control ADC (ADC1), and there is no such register in other ADC units.

17.4.17 ADC Channel Convert Data Register (ADC_DR)

ADC_DRx (ADC1 x=0~15, ADC2 x=0~15, ADC3 x=0~19) channel x data register

ADC_DR register is a read-only register used to store A/D conversion data for each channel. The reset value is 0x0000.

Conversion results are stored differently based on data alignment and conversion resolution.

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]

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]

Right alignment - 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]

Left alignment - 12-bit Resolution

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
												0	0	0	0

Left 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	0

Left alignment - 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

17.4.18 ADC Analog Watch Dog Control Register (ADC_AWDCR)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8
—	—	—	—	—	—	AWDCM[1]	AWDCM[0]
B7	b6	b5	b4	b3	b2	b1	b0
—	AWD1MD	AWD1IEN	AWD1EN	—	AWD0MD	AWD0IEN	AWD0EN

Bit	Symbol	Bit name	Description	Read and write
b15-b10	Reserved	—	Read 0, write 0	R/W
b9-b8	AWDCM[1:0]	Watchdog Window Combination Selection	00: Disable window combination, ADC_CMP1 output window 1 independent comparison result 01: Enable the combination of windows, ADC_CMP1 output the result that window 0 logic OR with window 1 10: Enable the combination of windows, ADC_CMP1 output the result that window 0 logic AND with window 1 11: Enable the combination of windows, ADC_CMP1 output the result that window 0 logic XOR with window1 Note: When using the window combination function, both window 0 and window 1 need to be enabled, that is, write 1 to AWD0EN and write 1 to AWD1EN. If the comparison channels selected by window 0 and window 1 are different, please ensure that the channel selected by window 1 is converted after the channel selected by window 0 during the scan conversion. The ADC_CMP1 interrupt or event is output at the end of the channel conversion selected for Window 1.	R/W
B7	Reserved	—	Read 0, write 0	R/W
b6	AWD1MD	Comparison mode of watchdog window 1	Window 1 Protected Area Selection 0: Conversion result < AWD1DR0, or conversion result > AWD1DR1 1: Conversion result ≥ AWD1DR0, and conversion result ≤ AWD1DR1.	R/W
b5	AWD1IEN	Watchdog Window 1 interrupt Enable	0: Disable Watchdog Window 1 comparison interrupt ADC_CMP1 1: Enable Watchdog Windows1 comparison interrupt ADC_CMP1	R/W
b4	AWD1EN	Watchdog Window 1 Compare Enable	0: Disable Watchdog Window 1 comparison 1: Enable Watchdog Window 1 comparison	R/W
b3	Reserved	—	Read 0, write 0	R/W
b2	AWD0MD	Watchdog Window 0 Comparison Mode	Window 0 protected area selection 0: Conversion result < AWD0DR0, or conversion result > AWD0DR1 1: Conversion result ≥ AWD0DR0, and conversion result ≤ AWD0DR1.	R/W
b1	AWD0IEN	Watchdog Window 0 interrupt enable	0: Disable Watchdog Window 0 comparison interrupt ADC_CMP0 1: Enable Watchdog Windows 0 comparison interrupt ADC_CMP0	R/W
b0	AWD0EN	Watchdog Window 0 Compare Enable	0: Disable Watchdog Window 0 comparison 1: Enable Watchdog Window 0 comparison	R/W

17.4.19 ADC Analog Watch Dog Status Register (ADC_AWDSR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
—	—	—	AWDCMF	—	—	AWD1F	AWD0F

Bit	Symbol	Bit name	Description	Read and write
b7-b5	Reserved	—	Read 0, write 0	R/W
b4	AWDCMF	Watchdog window combination comparison status bit	Set 1 when the watchdog window combination comparison function enables(that is, AWDCM [1: 0] = 01b/10b/11b) and window 0 and window 1 compare results meet the combination condition. Invalid write to this register bit.	R
b3-b2	Reserved	—	Read 0, write 0	R/W
b1	AWD1F	Watchdog Window 1 Comparative status bit	Set 1 when the comparison result met the comparison condition at the end of the selected channel conversion of Window 1. Invalid write to this register bit.	R
b0	AWD0F	Watchdog window 0 comparison status bit	Set 1 when the comparison result met the comparison condition at the end of the selected channel conversion of Window 0. Invalid write to this register bit.	R

17.4.20 ADC Analog Watch Dog Status Clear Register (ADC_AWDSCLRR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
—	—	—	CLRAWDCMF	—	—	CLRAWD1F	CLRAWD0F

Bit	Marking	Place name	Function	Read and write
b7-b5	Reserved	—	Read 0, write 0	R/W
b4	CLRAWDCMF	Watchdog Window Combination Comparison State Position Reset	Writing 0 has no effect. Writing 1 resets the AWDCMF status bit. Read always 0	R/W
b3-b2	Reserved	—	Read 0, write 0	R/W
b1	CLRAWD1F	Watchdog Window 1 Comparative State Reset	Writing 0 has no effect. Writing 1 resets the AWD1F status bit. Read always 0	R/W
b0	CLRAWD0F	Watchdog Window 0 Comparative State Reset	Writing 0 has no effect. Writing 1 resets the AWD1F status bit. Read always 0	R/W

17.4.21 ADC Analog Watch Dog Compare Data Register (ADC_AWD0DR0, ADC_AWD0DR1, ADC_AWD1DR0, ADC_AWD1DR1)

Reset value: ADC_AWD0DR0=0x0000, ADC_AWD0DR1=0xffff

ADC_AWD1DR0=0x0000, ADC_AWD1DR1=0xffff

b15	b14	b13	b12	b11	b10	b9	b8
AWDDR[15:8]							
B7	b6	b5	b4	b3	b2	b1	b0
AWDDR[7:0]							
Bit	Symbol	Bit name	Description				Read and write
B15-b0	AWDDR [15:0]	Comparative data	Comparative data				R/W

AWD0DR0 sets window 0 low threshold and AWD0DR1 sets window 0 high threshold.

AWD1DR0 sets window 1 low threshold, AWD1DR1 sets window 1 high threshold.

AWD0DR0, AWD0DR1, AWD1DR0, AWD1DR1 may differ in resolution (12-bit, 10-bit or 8-bit) and alignment (right or left alignment of data).

- Right alignment - 12-bit resolution low 12 bits [11: 0] available
- Right alignment - 10-bit resolution low 10 bits [9: 0] available
- Right alignment - 8-bit resolution low 8 bits [7: 0] available
- Left alignment - 12-bit resolution high 12 bits [15: 4] available
- Left alignment - 10-bit resolution high 10 bits [15: 6] available
- Left alignment - 8-bit resolution high 8 bits [15: 8] available

17.4.22 ADC Analog Watch Dog Channel Select Register (ADC_AWD0CHSR, ADC_AWD1CHSR)

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
—	—	—	AWDCH[4]	AWDCH[3]	AWDCH[2]	AWDCH[1]	AWDCH[0]
Bit	Symbol	Bit name	Description				Read and write
b7-b5	Reserved	—	Read 0, write 0				R/W
b4-b0	AWDCH[4:0]	Comparison channel selection of watchdogs	ADC_AWD0 CHSR for window 0 and ADC_AWD1 CHSR for window 1. 0x00: CH0 0x01: CH1 and so on Note: Please do not set to non-existing channel				R/W

17.4.23 ADC Dedicated SH Control Register(ADC_SHCR)

Reset value: 0x0018

b15	b14	b13	b12	b11	b10	b9	b8
—	—	—	—	—	SHSEL[2]	SHSEL[1]	SHSEL[0]
B7	b6	b5	b4	b3	b2	b1	b0
SHSST[7]	SHSST[6]	SHSST[5]	SHSST[4]	SHSST[3]	SHSST[2]	SHSST[1]	SHSST[0]

Bit	Symbol	Bit name	Description	Read and write
b15-b11	Reserved	—	Read 0, write 0	R/W
b10-b8	SHSEL[2:0]	the dedicated sample and hold circuit channel selection	0: Bypass the dedicated sample and hold circuit 1: Enable the dedicated sample and hold circuit Each bit corresponds to a channel. Note: In double sequence scan mode, do not use dedicated sample-and-hold circuits in sequence A. Because the sequence A may be interrupted by the sequence B, the sampling time of each dedicated sample and hold circuit is inconsistent and the unexpected conversion result will output.	
b7-b0	SHSST[7:0]	Sampling time selection	The sampling time can be selected from 4 to 255 cycles. Note: The initial conversion state has 24 sampling cycles. If the frequency of PCLK2 is 50MHz, one sampling cycle is 20ns, so the sampling time is 0.48us. When the input impedance is high and the sampling time is insufficient, or the PCLK2 frequency is low, the sampling time can be adjusted. The sampling time should not be less than 0.4us.	R/W

Note:

- Unit 1 (ADC1) supports SH1~3 total 3 channels, refer to Table 17-1 . Register ADC_SHCR.SHSEL[0] corresponds to SH1, SHSEL[1] corresponds to SH2, and SHSEL[2] corresponds to SH3.
- Unit 2 (ADC2) does not support SH , no this register.
- Unit 3 (ADC3) does not support SH , no this register.

17.4.24 ADC PGA Control Register(ADC_PGACRx)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
PGAGAIN[3:0]				PGACTL[3:0]			
Bit	Symbol	Bit name	Description				
b7-b4	PGAGAIN[3:0]	Gain setting	0 0 0 0: × 2.000				
			0 0 0 1: × 2.133				
			0 0 1 0: × 2.286				
			0 0 1 1: × 2.667				
			0 1 0 0: × 2.909				
			0 1 0 1: × 3.2				
			0 1 1 0: × 3.556				
			0 1 1 1: × 4.000				
			1 0 0 0: × 4.571				
			1 0 0 1: × 5.333				
			1 0 1 0: × 6.4				
			1 0 1 1: × 8				
			1 1 0 0: × 10.667				
			1 1 0 1: × 16				
			1 1 1 0: × 32				
Note: Other values are prohibited from being set							
b3-b0	PGACTL[3:0]	Amplifier control	0000: Disable amplifier 1110: Enable amplifier, the signal is amplified according to the set value of PGAGAIN[3:0] Note: Prohibits to set other values .				

Note:

- Unit 1 (ADC1) supports 3 channels PGA1~3, refer to Table 17-1 . Registers ADC1_PGACR1~3 correspond to control PGA1~3 respectively.
- Unit 2 (ADC2) supports 1 channel PGA4, refer to Table 17-1 . Register ADC2_PGACR1 corresponds to control PGA4.
- Unit 3 (ADC3) does not support PGA.

17.4.25 ADC PGA VSS Source Select Register (ADC_PGA_VSSEN)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
							PGAVSSEN[2:0]
Bit	Symbol	Bit name	Description				Read and write
b7-b3	Reserved	—	Read 0, write 0				R/W
b2-b0	PGAVSSEN[2:0]	Amplifier ground reference selection	0: Use external pin PGAx_VSS as PGA reference ground 1: Use the analog ground AVSS as the PGA reference ground Unit 1 ADC1_PGA_VSSEN.R.PGAVSSEN[0] selects PGA1 reference ground PGA123_VSS or AVSS Unit 1 ADC1_PGA_VSSEN.R.PGAVSSEN[1] selects PGA2 reference ground PGA123_VSS or AVSS Unit 1 ADC1_PGA_VSSEN.R.PGAVSSEN[2] selects PGA3 reference ground PGA123_VSS or AVSS Unit 2 ADC2_PGA_VSSEN.R.PGAVSSEN[0] select PGA4 reference ground PGA4_VSS or AVSS Unit 2 ADC2_PGA_VSSEN.R.PGAVSSEN[2:1] does not exist.	R/W			

Note:

- Unit 1, 2 (ADC1, 2) support this register, unit 3 (ADC3) does not have this register.

17.5 Precautions for use

17.5.1 Considerations when reading data register

A/D data register ADC_DR is available in half-word units. Do not access the data register in bytes.

17.5.2 Scan Finish Interrupt Handling Considerations

If the CPU does not read the first converted data in time, the second converted data will overwrite the first converted data.

17.5.3 Module Stop and Low Power Settings Considerations

By setting the register PWC_FCG, the ADC module can be set to stop to reduce power consumption. The initial ADC is stop mode. When the A/D module needs to work, please set the corresponding bit of the PWC_FCG register to cancel the stop, and wait for 1us before starting the A/D conversion.

Before setting the module to stop, please confirm that the A/D is in conversion stop, that is, the ADC_STR.START bit is "0".

Before setting the system to enter stop mode (STOP), please set the ADC to the module stop mode first.

For details, please refer to the Low Power Description section.

17.5.4 Pin Setting for A/D Converter Analog Channel Input

When the chip pin is set as A/D analog channel input, please disable the digital function (PCRxy.DDIS) of the corresponding pin first. Refer to the GPIO chapter.

17.5.5 Noise control

To prevent abnormal voltages such as surges from destroying the analog input pins, it is recommended to use **Electrical Characteristics in the Data Sheet** protection circuit shown in section.

18 Digital-to-analog converter DAC

18.1 Introduction

This MCU is equipped with two 12-bit digital-to-analog voltage converter units DAC1 and DAC2. Each DAC unit contains two D/A conversion channels, which can be converted independently or synchronously converted by synchronous update of conversion data. Each conversion channel features an output amplifier that can directly drive an external load without an external op amp. Independent pin input reference voltages VREFH and VREFL can be used to improve conversion accuracy.

DAC main features:

- Two DAC units, each with two D/A conversion channels
- 12-bit conversion data can be configured as left-aligned or right-aligned format
- Two conversion channels of the same DAC module can achieve simultaneous conversion
- Convert external data (from DCU) to output triangle wave and sawtooth wave
- Output amplification function, can directly drive external load
- A/D conversion priority mode reduces interference to A/D conversion
- Output available for voltage comparator as negative terminal voltage
- Independent pin input reference voltage VREFH/VREFL

18.2 Functional block diagram

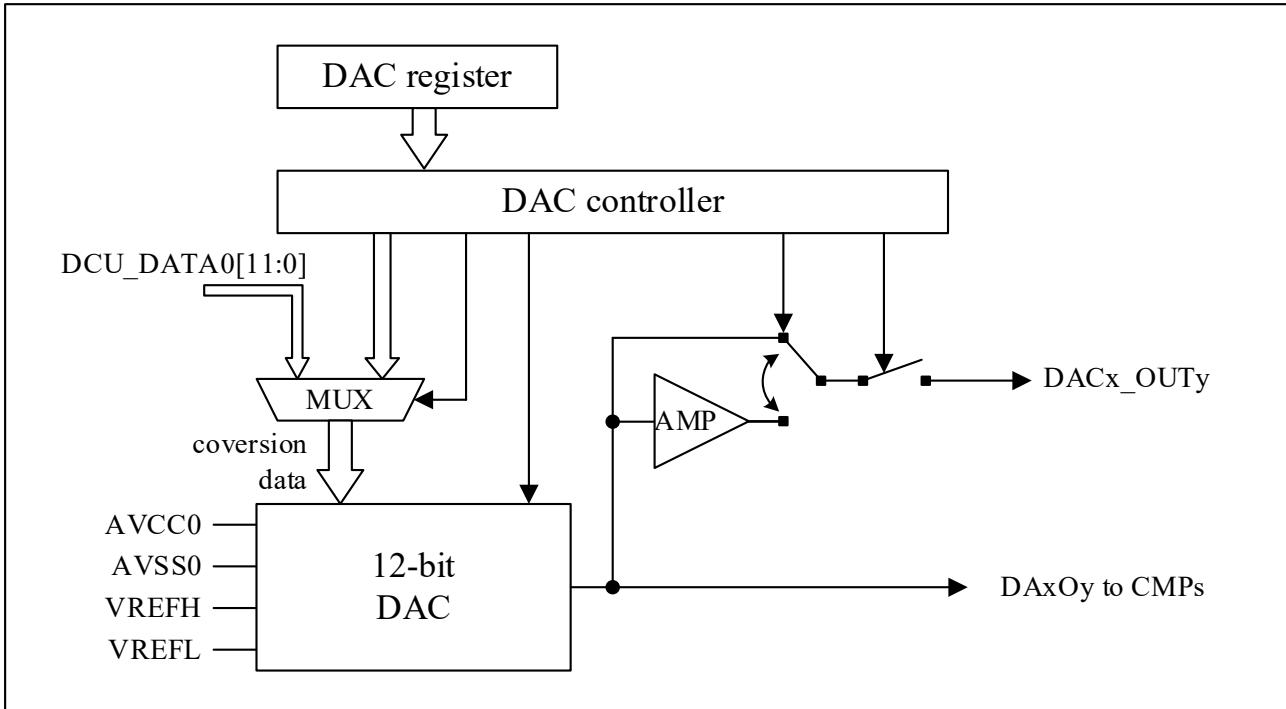


Figure 18-1 D/A conversion channel block diagram

Table 18-1 DAC pin

Pin name	Input/Output	Function
AVCC0	Input	Analog power
AVSS0	Input	Analog ground
VREFH	Input	reference voltage
VREFL	Input	reference place
DACx_OUTy	Output	D/A conversion analog output (x,y=1~2)

18.3 Functional description

18.3.1 D/A conversion

Each conversion channel can perform D/A conversion independently. When DACRx.DAyE(x,y=1,2) is set to 1, the D/A conversion of the corresponding channel starts and the conversion result is output from the DACx_OUTy port. If only the conversion result is used as the negative input voltage of the voltage comparator CMP, the DACx_OUTy output can be turned off through the DAOCR register.

The following is an example of single-channel conversion using DAC1_ch1, see the action sequence Figure 18-2 .

1. Set the D/A conversion data (DADR1_1) and data format (DACR1.DPSEL).
2. Set DACR1.DA1E to 1, D/A conversion starts. After the conversion time tDCONV, the conversion result is output from port DAC1_OUT1 and held until the value of DADR is overwritten or DACR1.DA1E is set to 0. The output analog voltage value can be calculated by the following formula:

$$\text{DACoutput} = \text{ConversionData}/4096 \times \text{VREFH}$$

3. When DACR1.DA1E is 1, rewriting the value of DADR will trigger a new D/A conversion. Similarly, after the conversion time tDCONV elapses, a new conversion result is output from the port DAC1_OUT1.
4. Set the DACR1.DA1E bit to 0, turn off DAC1_ch1, and output DAC1_OUT1 in a high-impedance state.

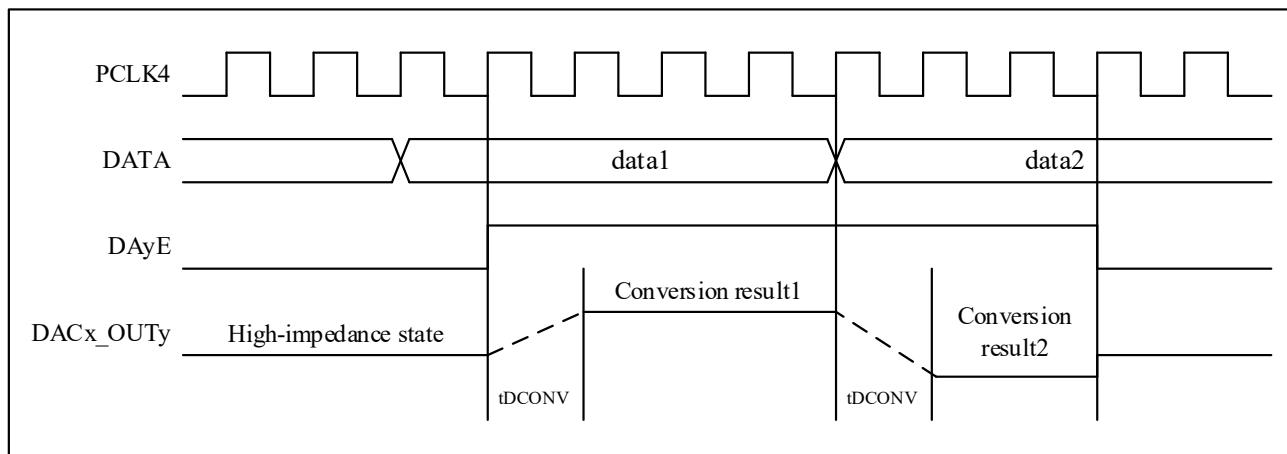


Figure 18-2 D/A conversion diagram

18.3.2 Synchronous conversion

Synchronous conversion of two conversion channels of the same DAC unit can be achieved by synchronous update of conversion data.

Below is an example of simultaneous conversion of two channels using DAC1.

1. Set D/A conversion data (DADR1_1, DADR1_2) and data format (DACR.DPSEL).
2. By setting DACR1.DAE to 1, the D/A conversion of both channels starts synchronously. After the conversion time tDCONV, the conversion results are output from ports DAC1_OUT1 and DAC1_OUT2 respectively and hold until the value of DADR is overwritten, or DACR1.DAE is set to 0.
3. When DACR1.DAE is 1, rewriting the values of DADR1_1 and DADR1_2 at the same time will trigger the two channels to start a new D/A conversion synchronously. Likewise, after the conversion time tDCONV has elapsed, a new conversion result is output from DAC1_OUT1 and DAC1_OUT2.
4. Set DACR1.DAE to 0, the two channels are turned off synchronously, and DAC1_OUT1 and DAC1_OUT output high-impedance state.

18.3.3 External data conversion

In addition to converting the data in the DADR, the DAC can also convert the data from the Data Computation Unit (DCU). Channel 1 and Channel 2 of DAC1 correspond to Unit 1 and Unit 2 of the DCU, respectively, and Channel 1 and Channel 2 of DAC2 correspond to Unit 3 and Unit 4 of the DCU, respectively. Setting DACRx.EXTDSLy to 1 selects to convert the lower 12-bit data from data register 0 of the corresponding DCU unit.

Use of output amplifier and A/D conversion priority mode is prohibited when converting external data.

18.3.4 A/D conversion priority mode

A brief inrush current may appear on the analog power supply when the DAC starts the D/A conversion, which can interfere with the A/D conversion in progress. The A/D conversion priority mode can effectively avoid the occurrence of such interference by changing the update timing of the D/A conversion data.

Set DAADPCR.ADOPEN to 1, the DAC enters the A/D conversion priority mode. At this time, if the value of DADR is rewritten during the A/D conversion, the D/A conversion will not start immediately, but will not start until the A/D conversion is completed. That is to say, from rewriting DADR to actually starting D/A conversion, it takes the longest time to wait for one A/D conversion. Therefore, the value of DADR does not agree with the analog output value during this period. However, if the ADC is stopped (ADCSR.ADST is 0) when DADR is rewritten, the D/A conversion will start after 2

ADCLK cycles. DAADPCR.ADPSLn ($n=0\sim 2$) is used to select the ADC channel with priority for A/D conversion.

When external data conversion is selected (DACRx.EXTDSLy=1), the A/D conversion priority mode is invalid.

The following takes DAC1_ch1 as an example to explain the setting steps of the A/D conversion priority mode. The action sequence is shown in Figure 18-3.

1. Verify that the ADC is stopped and set DACR1.ADPSL.
2. Make sure the ADC is stopped and set DACR1.DA1E to 1.
3. Write conversion data to DADR1_1.
 - When conversion data A is written into DADR1_1, ADCSR.ADST is 0, ADC is in stop state, and D/A conversion starts after 2 ADCLK cycles.
 - When conversion data B is written to DADR1_1, ADCSR.ADST is 1, ADC is performing A/D conversion, and D/A conversion will not start until A/D conversion is completed. The conversion data C has been written to DADR1_1 before the D/A conversion starts, so the conversion data C is finally converted, not the conversion data B.

In order to avoid the loss of conversion data, please check the D/A conversion status register (DAADPCR x .DAySF, $xy=1\sim 2$) first, and then rewrite the value of DADR after confirming that the current data conversion is completed.

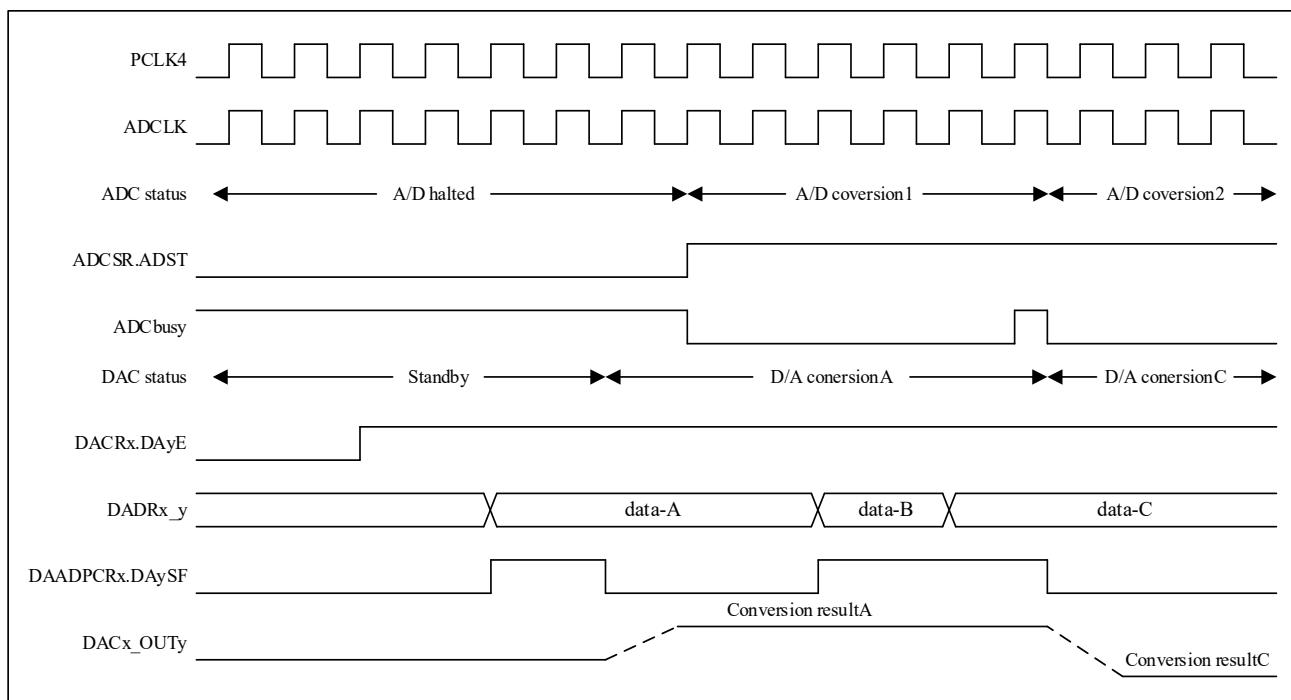


Figure 18-3 A/D conversion priority mode operation diagram

18.4 Precautions

18.4.1 Setting of the module stop function

You can use the module stop control register to set DAC1, 2 to be valid or stopped. After the system is reset, DAC1 and 2 are both stopped, and the register can be accessed only after the module is stopped. See [Operation Mode and Low Power Mode].

18.4.2 DAC operation when the module is stopped

If the DAC enters the module stop state during D/A conversion, the analog output will be maintained, and the current flowing through the analog power supply is the same as that during D/A conversion. To further reduce the power consumption when the module is stopped, set the DAyE and DAE bits of the DACRx to 0.

18.4.3 Stop DAC action in low power mode

If the system enters stop low-power mode during D/A conversion, the analog output will be held, and the current flowing through the analog power supply is the same as during D/A conversion. To further reduce power consumption in stop mode, set both the DAyE and DAE bits of the DACRx to 0.

18.4.4 DAC Action in Power-Down Low-Power Mode

If the system enters a power-down low-power mode during D/A conversion, the analog outputs will be placed in a high-impedance state.

18.4.5 Precautions for Using Output Amplifiers

Please use the following initialization procedure when using the output amplifier.

1. Write all 0s to DADRx_y.
2. Set DACRx.DAAMPy to 1.
3. Set DACRx.DAE or DAyE to 1.
4. After waiting for 3us, write the transformed data into DADRx_y.

Turning off the DAC can stop the amplifier from working. To use the amplifier again, repeat steps 1~4.

18.5 Register description

DAC1 base address: 0x40041000

Table 18-2 DAC1 register list

Register name	Symbol	Offset address	Bit width	Reset value
DAC1 Data Register 1	DADR1_1	0x00	16	0x0000
DAC1 Data Register 2	DADR1_2	0x02	16	0x0000
DAC1 Control Register	DACR1	0x04	16	0x0000
DAC1 A/D conversion priority control register	DAADPCR1	0x06	16	0x0000
DAC1 Analog Output Control Register	DAOCR1	0x1c	16	0x0000

DAC2 base address: 0x40041400

Table 18-3 DAC2 Register List

Register name	Symbol	Offset address	Bit width	Reset value
DAC2 Data Register 1	DADR2_1	0x00	16	0x0000
DAC2 Data Register 2	DADR2_2	0x02	16	0x0000
DAC2 Control Register	DACR2	0x04	16	0x0000
DAC2 A/D conversion priority control register	DAADPCR2	0x06	16	0x0000
DAC2 Analog Output Control Register	DAOCR2	0x1c	16	0x0000

18.5.1 DAC data register (DADR_{x_y}, x,y=1,2)

When DACR.DPSEL=0 (data right-aligned)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-	-	-	-												DADR[11:0]

Bit	Marking	Place name	Function	Read and write
b15~b12	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b11~b0	DADR[11:0]	Transform data	Conversion data set value	R/W

When DACR.DPSEL=1 (data left-aligned)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
												-	-	-	-

Bit	Marking	Place name	Function	Read and write
b15~b4	DADR[11:0]	Transform data	Conversion data set value	R/W
b3~b0	Reserved	-	Read 0 when reading, write 0 when writing	R/W

DADR is used to store the data of D/A conversion, one for each channel. As long as D/A conversion is enabled, the value in DADR is converted to an analog voltage and output from the analog port. A 32-bit operation on the DADR of the same unit enables dual-channel simultaneous conversion.

18.5.2 DAC Control Register (DACRx, x=1,2)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-	-	-	EXT DSL2	EXTS L1	DAA MP2	DAA MP1	DP SEL	-	-	-	-	-	DA2 E	DA1 E	DAE

Bit	Marking	Place name	Function	Read and write
b15~b13	Reserved	-	Read as "0", write as "0"	R/W
b12	EXTDSL2	Channel 2 external data selection	0: Select DADR as the conversion data for channel 2 1: Select external data as the conversion data of channel 2	R/W
b11	EXTDSL1	Channel 1 external data selection	0: Select DADR as the conversion data for channel 1 1: Select external data as the conversion data of channel 1	R/W
b10	DAAMP2	Channel 2 output amplifier enable	0: Disable channel 2 output amplifier 1: Enable channel 2 output amplifier	R/W
b9	DAAMP1	Channel 1 Output Amplifier Enable	0: Disable channel 1 output amplifier 1: Enable channel 1 output amplifier	R/W
b8	DPSEL	Data register format selection	0: Right-aligned format 1: Left-aligned format	R/W
b7~b3	Reserved	-	Read as "0", write as "0"	R/W
b2	DA2E	Channel 2 enable	This bit setting is valid when DAE=0 0: Channel 2 disabled 1: Channel 2 is enabled	R/W
b1	DA1E	Channel 1 enable	This bit setting is valid when DAE=0 0: Channel 1 disabled 1: Channel 1 is enabled	R/W
b0	DAE	Channel always enabled	0: Channels 1 and 2 are disabled 1: Channel 1, 2 enable	R/W

DACR is used to control D/A conversion and D/A output. see details

Table 18-4 .

When the DA1E bit is 0 and the DAE bit is also 0, the D/A conversion of channel 1 is stopped. At this time, no matter what the value of DAAMP1 is, the analog output of channel 1 is disabled, and the port is in a high-impedance state. When the DA2E bit is 0 and the DAE bit is also 0, the D/A conversion of channel 2 is stopped. At this time, no matter what the value of DAAMP2 is, the analog output of channel 2 is disabled, and the port is in a high-impedance state.

If the A/D conversion priority mode is selected, please set the DA2E, DA1E and DAE bits in the state of ADC stop (ADCSR.ADST=0). At the same time, in order to prevent the ADC from being accidentally activated, please set the trigger selection of the ADC to software trigger.

18.5.3 DAC Analog Output Control Register (DAOCR_x, x=1,2)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
DAO DIS2	DAO DIS1	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit	Marking	Place name	Function	Read and write
b15	DAODIS2	DAC _x _OUT2 port output control	1: Disable the DAC _x _OUT2 port to output analog voltage 0: Allow DAC _x _OUT2 port to output analog voltage	R/W
b14	DAODIS1	DAC _x _OUT1 port output control	1: Disable the DAC _x _OUT1 port to output analog voltage 0: Allow DAC _x _OUT1 port to output analog voltage	R/W
b13~b0	Reserved	-	Read as "0", write as "0"	R/W

Table 18-4 D/A conversion and analog output control

DAE	DAyE	DAAMP	DAODIS	D/A conversion	output amplifier	DAC_OUT Output	DAO Output
0	0	0/1	0/1	Stop	Stop	High impedance state	High impedance state
	1	0	0	start up	Stop	normal output	normal output
			1	start up	Stop	High impedance state	normal output
		1	0	start up	start up	Amplify output	normal output
			1	start up	start up	High impedance state	normal output
1	0	0	0	start up	Stop	normal output	normal output
		1	0	start up	Stop	High impedance state	normal output
		1	0	start up	start up	Amplify output	normal output
			1	start up	start up	High impedance state	normal output
1	0	0	0	start up	Stop	normal output	normal output
		1	0	start up	Stop	High impedance state	normal output
		1	0	start up	start up	Amplify output	normal output
			1	start up	start up	High impedance state	normal output

18.5.4 DAC A/D conversion priority control register (DAADPCR_x, x=1,2)

Reset value: 0x0000

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
ADP EN	-	-	-	-	-	DA2 SF	DA1 SF	-	-	-	-	-	ADC SL3	ADC SL2	ADC SL1

Bit	Marking	Place name	Function	Read and write
b15	ADPEN	A/D conversion priority mode selection	0: A/D conversion priority mode is invalid 1: A/D conversion priority mode is enabled	R/W
b14~b10	Reserved	-	Read as "0", write as "0"	R/W
b9	DA2SF	Channel 2 data update status	This flag only changes in A/D conversion priority mode 0: Channel 2 data is updated 1: Channel 2 data is being updated	R/W
b8	DA1SF	Channel 1 data update status	This flag only changes in A/D conversion priority mode 0: Channel 1 data is updated 1: Channel 1 data is being updated	R/W
b7~b3	Reserved	-	Read as "0", write as "0"	R/W
b2	ADCSL3	ADC_3 is preferred	0: Do not select ADC_3 as the priority object of A/D conversion 1: Select ADC_3 as the priority object of A/D conversion	R/W
b1	ADCSL2	ADC_2 is preferred	0: Do not select ADC_2 as the priority object of A/D conversion 1: Select ADC_2 as the priority object of A/D conversion	R/W
b0	ADCSL1	ADC_1 is preferred	0: Do not select ADC_1 as the priority object of A/D conversion 1: Select ADC_1 as the priority object for A/D conversion	R/W

DAADPCR is used for A/D conversion priority mode control.

When ADPEN is set to 1, the A/D conversion priority mode is valid. Please select the ADC unit as the priority object when setting ADPEN.

DAxSF(x=1,2) is the DA conversion status flag, which reflects whether the data in the current data register of this channel has been converted.

19 Temperature Sensor (OTS)

19.1 Introduction

On-chip Temperature Sensor (OTS for short) can obtain the temperature inside the chip to support Reliable operation of the system. OTS provides a set of digital quantities related to temperature, and the temperature value can be obtained by calculation. When not in use, it can be turned off by the module stop function to reduce system power consumption.

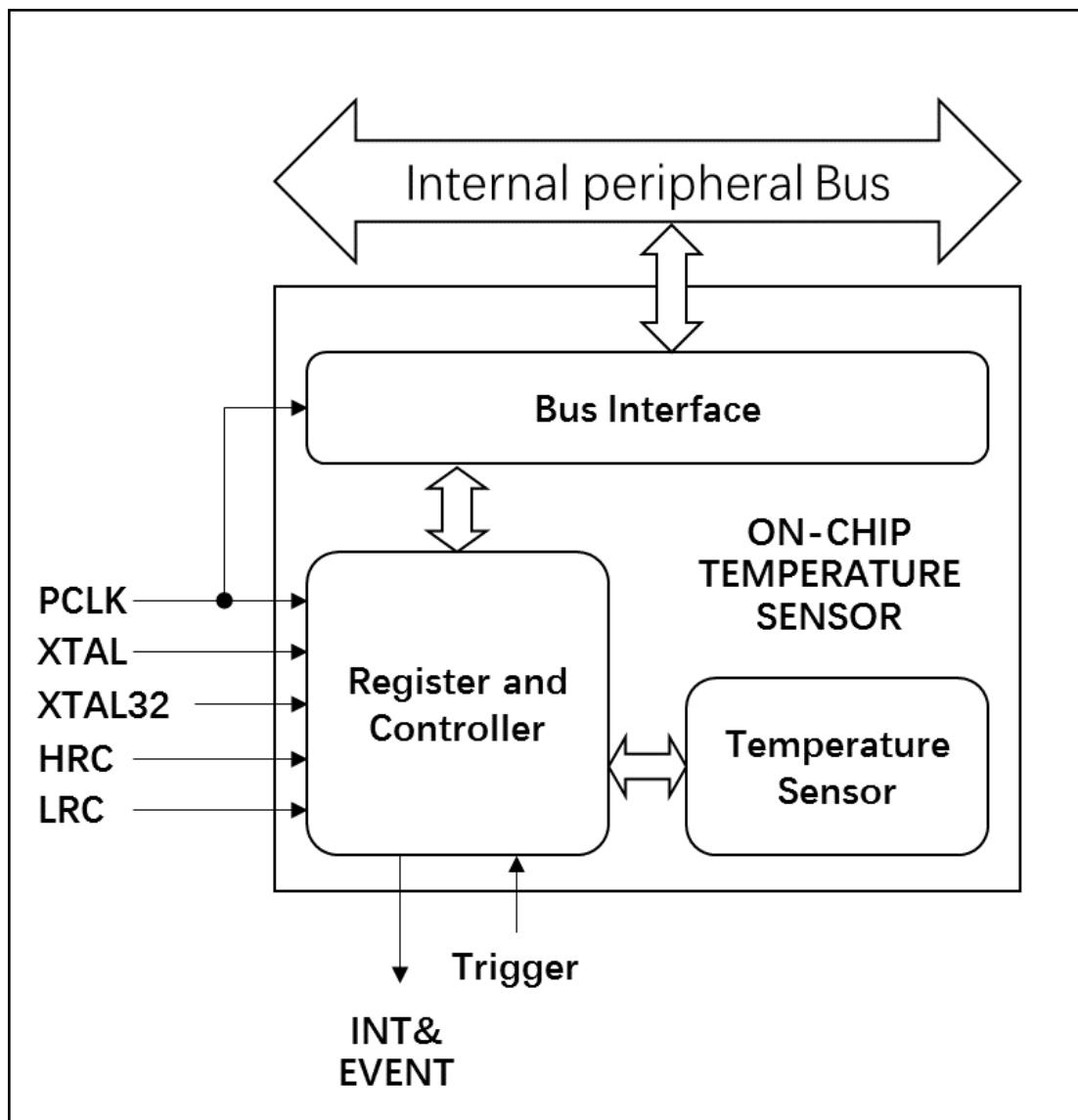


Figure 19-1 OTS functional block diagram

19.2 Usage notes

Before using the OTS to obtain the internal temperature of the chip, it must be calibrated first, and the measurement accuracy is related to the calibration method. Please refer to the following calibration experiments for calibration. When using, please turn off the module stop function and start the internal low-speed clock LRC. At the same time, please choose to start the internal high-speed clock HRC, external high-speed clock XTAL and external low-speed clock XTAL32 according to the usage. After setting the register OTS_CTL.OTSST to "1", the temperature measurement starts, and the OTSST bit will be automatically cleared after the temperature measurement is completed. After confirming that OTSST is "0", read the temperature parameters in registers OTS_DR1, 2 and OTS_ECR, and use the following calculation formula to obtain the temperature value.

$$T = K \times (1.7/D1 - 1/D2) \times E_{hrc} + M$$

[Parameter Description]

T: Temperature(°C)

K: Temperature slope (determined by calibration experiment)

D1: Temperature data 1 (read from register OTS_DR1)

D2: Temperature data 2 (read from register OTS_DR2)

E_{hrc} : HRC frequency error compensation data (read from register OTS_ECR)

M: Temperature offset (determined by calibration experiment)

The register OTS_CTL.OTSCK is used to select the temperature measurement clock. When HRC action is selected, its frequency error may affect the accuracy of the final calculated temperature. To eliminate this error, start XTAL32 before measuring temperature and use E_{hrc} when calibrating and calculating temperature. Omit E_{hrc} and show it as constant 1 when XTAL action is selected.

[Calibration experiment]

Please perform calibration experiments 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

$$A1: (1.7 / D1_{T1} - 1 / D2_{T1}) \times E_{hrcT1}$$

D1_{T1}, D1_{T1}, E_{hrcT1} are D1, D2, E_{hrc} measured at temperature T1 respectively;

$$A2: (1.7 / D1_{T2} - 1 / D2_{T2}) \times E_{hrcT2}$$

D1_{T2}, D1_{T2}, E_{hrcT2} are D1, D2, E_{hrc} measured at temperature T2 respectively;

As a reference, three groups of temperature data are preset inside the chip, and the data of two temperature points can be arbitrarily selected for calibration. Due to the deviation of the test environment temperature, the preset data is for reference only.

Table 19-1 OTS preset temperature data

Test temperature	Read mode		Test Conditions
	Temperature Data 1 (D1)	Temperature Data 2 (D2)	
T _a = -40 °C	OTS_PDR3[15:0]	OTS_PDR3[31:16]	XTAL=8MHz
T _a = 25 °C	OTS_PDR1[15:0]	OTS_PDR1[31:16]	XTAL=8MHz
T _a = 125 °C	OTS_PDR2[15:0]	OTS_PDR2[31:16]	XTAL=8MHz

E_{hrc} is a calibration value to eliminate the influence of the error of HRC itself on the measured temperature. It can be read from the register OTS_ECR. The usage method is as follows.

Table 19-2 How to use and set E_{hrc}

Action clock	Calibration experiment (calculating K,M)		Temperature measurement
(OTSCCK)	Use preset temperature data	Reacquire temperature data	(calculate T)
XTAL	E _{hrc} =1	E _{hrc} =1	E _{hrc} =1
HRC	E _{hrc} = f _{hrc} (MHz) / 0.032768	E _{hrc} =OTS_ECR[15:0]	E _{hrc} =OTS_ECR[15:0]

f_{hrc} is the standard frequency of HRC, namely 16MHz or 20MHz.

The register OTS_CTL.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 that 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 turning off to turning 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 triggered by other peripheral events. When using, please set the trigger target of the trigger source to OTS. When the temperature measurement is completed, an event that triggers the startup of other peripherals can also be generated. When using it, please set the register OTS_TRG to select the trigger target. When using the temperature measurement completion interrupt, please set the register OTS_CTL.OTSIE to "1".

19.3 Register description

Table 19-3 List of OTS Registers

Base address: 0x4004A800

Register name	Symbol	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

Base address: 0x40010600

Register name	Symbol	Offset address	Bit width	Reset value
OTS Preset Data Register 1	OTS_PDR1	0xe0	32	Indeterminate value
OTS Preset Data Register 2	OTS_PDR2	0xf4	32	Indeterminate value
OTS Preset Data Register 3	OTS_PDR3	0xf8	32	Indeterminate value

Base address: 0x40010800

Register name	Symbol	Address	Bit width	Reset value
OTS trigger source selection register	OTS_TRG	0x90	16	0x0000

19.3.1 OTS Control Register (OTS_CTL)

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-	-	-	-	-	-	-	-	-	-	-	-	TSST_P	OTS_IE	OTS_CK	OTS_ST

Reset value: 0x0000

Bit	Marking	Place name	Function	Read and write
b15~b4	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b3	TSSTP	Turn off the analog temperature sensor	Select whether to automatically turn off the analog temperature sensor after the temperature measurement is over 0: Do not turn off the analog temperature sensor 1: Turn off the analog temperature sensor	R/W
b2	OTSIE	interrupt enable bit	0: Disable temperature measurement end interrupt request 1: Allow temperature measurement end interrupt request	R/W
b1	OTSCK	clock select bits	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 temperature measurement 1: Start temperature measurement Set the "1" condition: (1) The software is set to "1" (2) The hardware trigger is set to "1" "0" condition: (1) Software "0" (2) Automatically clear "0" after temperature measurement	R/W

19.3.2 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

Bit	Marking	Place name	Function	Read and write
b15~b0	TSDC[15:0]	Temperature data D1	Temperature data D1 Automatically update after the temperature measurement is completed. Please confirm that OTS_CTL.OTSST is "0" before reading.	R

19.3.3 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

Bit	Marking	Place name	Function	Read and write
b15~b0	TSDC[15:0]	Temperature data D2	Temperature data D2 Automatically update after the temperature measurement is completed. Please confirm that OTS_CTL.OTSST is "0" before reading.	R

19.3.4 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

Bit	Marking	Place name	Function	Read and write
b15~b0	TSEC[15:0]	Error coefficient	Error coefficient Ehrc Automatically update after the temperature measurement is completed. Please confirm that OTS_CTL.OTSST is "0" before reading.	R

19.3.5 OTS Preset Temperature Data Registers (OTS_PDR1,2,3)

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TSPD2[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
TSPD1[15:0]															

Reset value: indeterminate value

Bit	Marking	Place name	Function	Read and write
b31~b16	TSPD2[15:0]	Preset temperature data D2	Preset temperature data for use in calibration. This data is measured at 8MHz, please convert it with the actual frequency of the operation clock selected by OTSCK before use. $D2 = TSPD2 \times f_{OTSCK} / 8$, f_{OTSCK} is the frequency of the actual operating clock, in MHz.	R
b15~b0	TSPD1[15:0]	Preset temperature data D1	Preset temperature data for use in calibration. This data is measured at 8MHz, please convert it with the actual frequency of the operation clock selected by OTSCK before use. $D1 = TSPD1 \times f_{OTSCK} / 8$, f_{OTSCK} is the frequency of the actual operating clock, in MHz.	R

19.3.6 OTS trigger source selection register (OTS_TRG)

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
TSTSL[8:0]															

Reset value: 0x0000

Bit	Marking	Place name	Function	Read and write
b15~b9	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b8~b0	TSTSL	Trigger source selection	Select the trigger source number when the hardware trigger starts Please refer to "10.3.2 Chapter of Note: Make settings, the interrupt event source marked with "V" in the "Can be selected as event" column in the table can be used as the trigger source.	R/W

20 Advanced Control Timer (Timer6)

20.1 Introduction

Advanced control timer 6 (Timer6) is a high-performance timer with a 16-bit/32-bit count width, which can provide rich and flexible combinations and various interrupts, events, and PWM outputs in various complex application scenarios. The timer supports two waveform modes of sawtooth waveform and triangular waveform, and can generate various PWM waveforms (unilaterally aligned independent PWM, bilaterally symmetrical independent PWM, bilaterally symmetrical complementary PWM, bilaterally asymmetrical PWM, etc.); software synchronization and hardware synchronization can be realized among units (synchronous start, stop, clear, update, etc.); each value register supports buffer function (single-level buffer and double-level buffer); supports pulse width measurement and period measurement; supports 2-phase quadrature encoding and 3-phase quadrature encoding; supports EMB control. This series of products is equipped with 8-unit Timer6 (U1~U4 are 32-bit timers; U5~U8 are 16-bit timers).

20.2 Basic block diagram

The basic functions and features of Timer6 are shown in Table 20-1.

Table 20-1 Basic functions and features of Timer6

Waveform mode	Sawtooth waveform (count up, count down), triangular waveform (count up and down)
Basic functions	• Input capture
	• Software synchronization
	• Hardware synchronization
	• Buffer function
	• Pulse width measurement
	• Period measurement
	• Quadrature encoding count
	• General PWM output
	• EMB control
Interrupt output	Count compare match interrupt
	Count period match interrupt
	Dead time error interrupt
Event output	Count compare match event
	Count period match event

The basic block diagram of Timer6 is shown in Figure 20-1. The "<t>" shown in the block diagram represents the unit number, that is, "<t>" is 1 to 8. When referring to "<t>" later in this chapter, they are all the unit number, and will not be repeatedly explained.

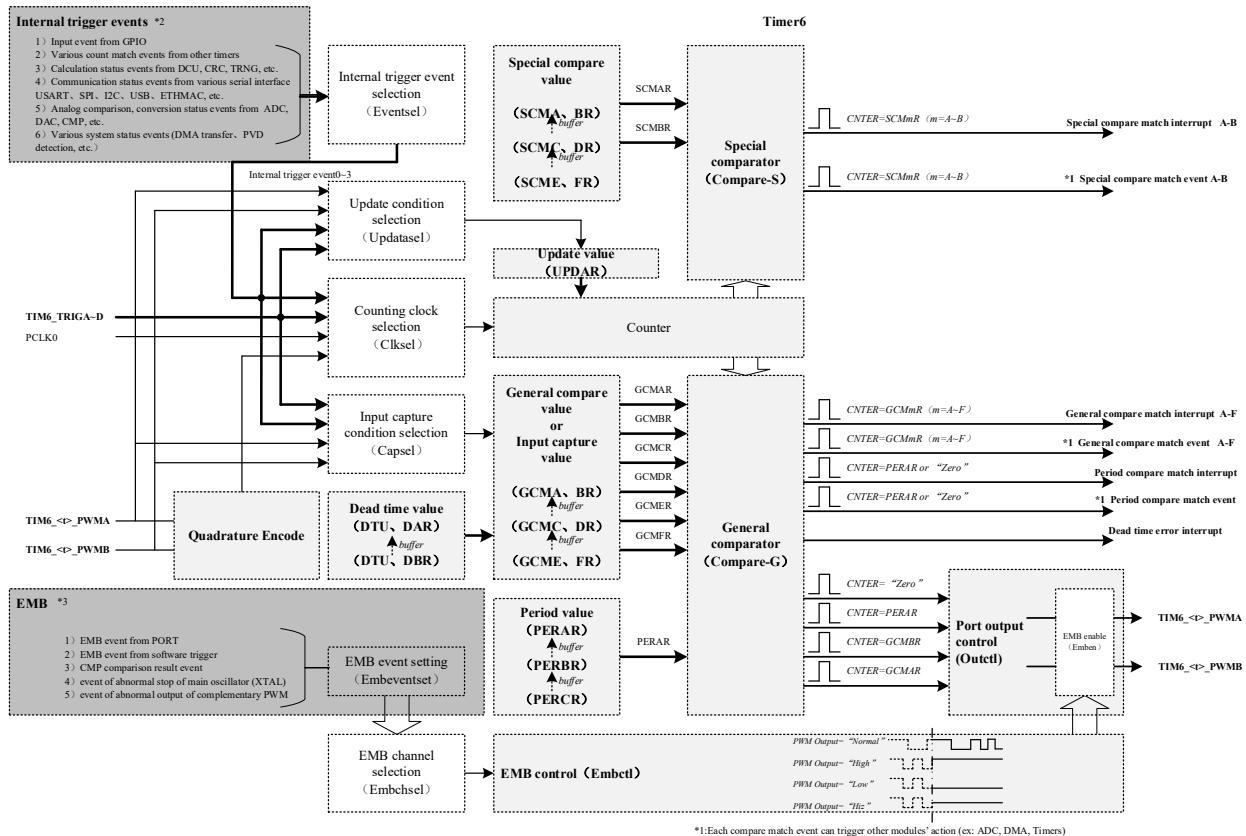


Figure 20-1 Basic block diagram of Timer6

Table 20-2 shows the list of input and output ports of Timer6.

Table 20-2 Timer 6 port list

Port name	Direction	Function
TIM6_<t>_PWMA	In or out	1) Quadrature encoding count clock input port or input capture port or compare output port 2) Hardware start, stop, clear, update condition input port
TIM6_<t>_PWMB		
TIM6_TRIGA	In	1) Hardware count clock input port or input capture port 2) Hardware start, stop, clear, update condition input port
TIM6_TRIGB		
TIM6_TRIGC		
TIM6_TRIGD		

20.3 Function description

20.3.1 Waveform mode

Timer6 has 2 basic counting waveform modes, sawtooth waveform mode and triangular waveform mode. The basic waveforms of the two modes are shown in figure 20-2 Sawtooth Waveform (count up) and figure 20-3 Triangular Waveform.

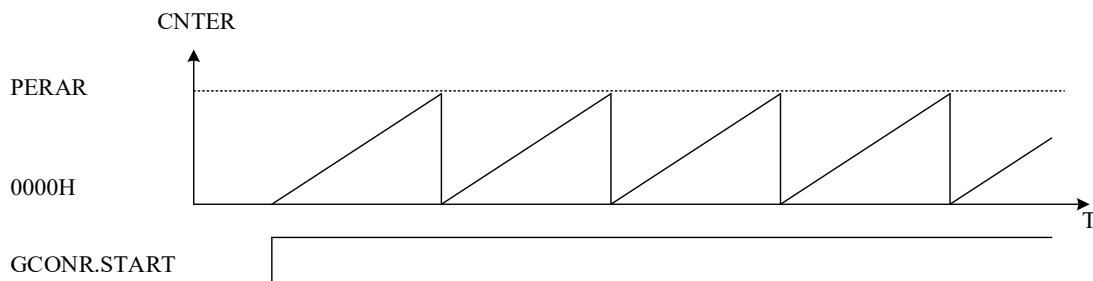


Figure 20-2 Sawtooth waveform (count up)

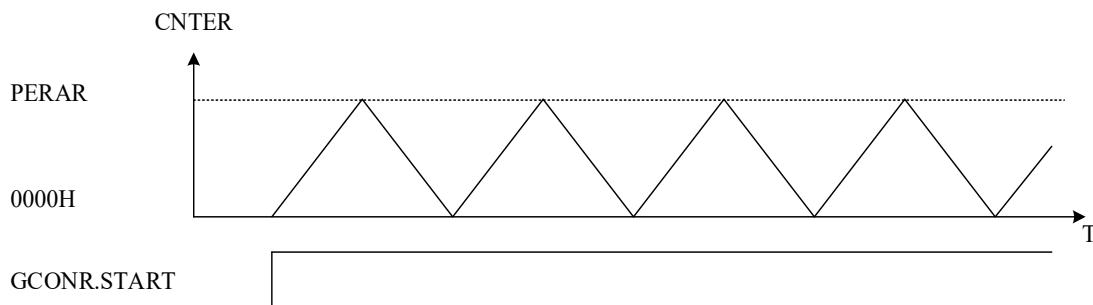


Figure 20-3 Triangular waveform

20.3.2 Clock source selection

The clock source of Timer6 has the following options:

- 1) 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 frequency division of PCLK0 and PCLK0 (set by GCONR.CKDIV[3:0])
- 2) Internal trigger event input 0~3 (set by HCUPR[11:8] or HCDOR[11:8])
- 3) Port input of TIM6_TRIGA~D (set by HCUPR[23:16] or HCDOR[23:16])
- 4) Quadrature encoding relation input of TIM6_<t>_PWMA and TIM6_<t>_PWMB (set by HCUPR[7:0] or HCDOR[7:0])

When clock source is selected to a) option, it's a software counting mode. When the clock source is selected to b), c) or d) option, it is a hardware counting mode. 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 a) clock is automatically invalid when b), c) or d) clock is selected.

20.3.3 Counting direction

The timer counting direction of Timer6 can be changed by software. In different waveform modes, the method of changing the counting direction is slightly different.

20.3.3.1 Sawtooth waveform counting direction

In the sawtooth waveform mode, the counting direction can be set during the timer counting or when it is stopped.

When GCONR.DIR=0 (counting down) is set during counting up, the timer will change to down counting mode after counting to overflow; when GCONR.DIR=1 (counting up) is set during counting down, the timer will change to up counting mode after counting to underflow.

If GCONR.DIR bit is set when counting is stopped, the setting of GCONR.DIR is reflected in the counting until overflow or underflow after counting starts.

20.3.3.2 Triangular waveform counting direction

In triangular waveform mode, the setting of the counting direction is invalid, and the counting direction is automatically changed when the counting reaches the counting peak point or the counting valley point.

20.3.4 Comparison output

Each unit of Timer6 has 2 comparison output ports (TIM6_<t>_PWMA, TIM6_<t>_PWMB), which can output the specified level when the count value matches with the compare value. The GCMAR and GCMBR registers correspond to the count compare values of TIM6_<t>_PWMA and TIM6_<t>_PWMB. When the count value of the timer and GCMAR are equal, the TIM6_<t>_PWMA or TIM6_<t>_PWMB port outputs the specified level.

The level of the TIM6_<t>_PWMA port can be set by PCNAR.STACA, PCNAR.STPCA, PCNAR.OVFCA, PCNAR.UDFCA, PCNAR.CMAU<D>A, PCNAR.FORCA bits of Port Control Register (PCNAR) when the counting starts, stops, overflows, and when count value matches, etc.

The level of the TIM6_<t>_PWMB port can be set by PCNBR.STACB, PCNBR.STPCB, PCNBR.OVFCB, PCNBR.UDFCB, PCNBR.CMAU<D>B, PCNBR.FORCB bits of Port Control Register (PCNBR) when the counting starts, stops, overflows, and when the count value matches, etc. Figure 20-4 is an example of the operation of the comparison output.

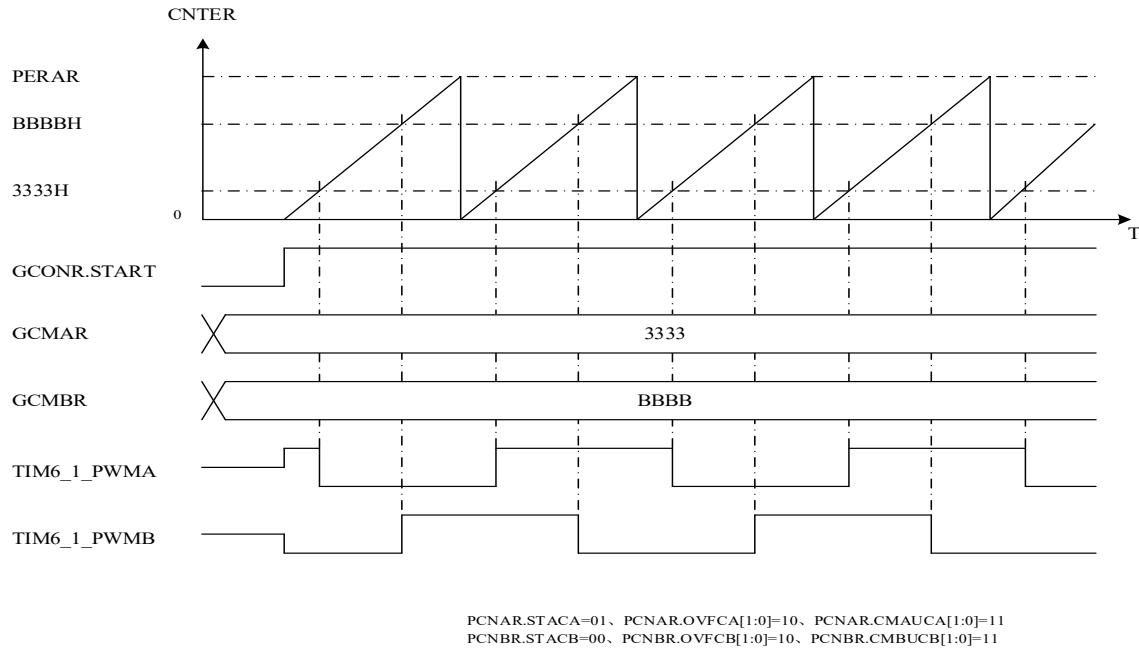


Figure 20-4 Comparison output action

20.3.5 Input capture

Each unit has a input capture function, with 2 groups of input capture registers (GCMAR, GCMBR), which are used to save the current count value captured. Set the CAPMDA bit of the Port Control Register (PCNAR) to 1 to enable the input capture function. When the corresponding input capture condition is set and the condition is valid, the current count value is saved to the corresponding capture register (GCMAR, GCMBR).

The input capture condition of each unit can be internal trigger event input 0~3, port input of TIM6_TRIGA~D, port input of TIM6_<t>_PWMA or TIM6_<t>_PWMB, etc. The detailed condition can be set by Hardware Input Capture Event Select Registers (HCPAR, HCPBR). Figure 20-5 is an example of input capture.

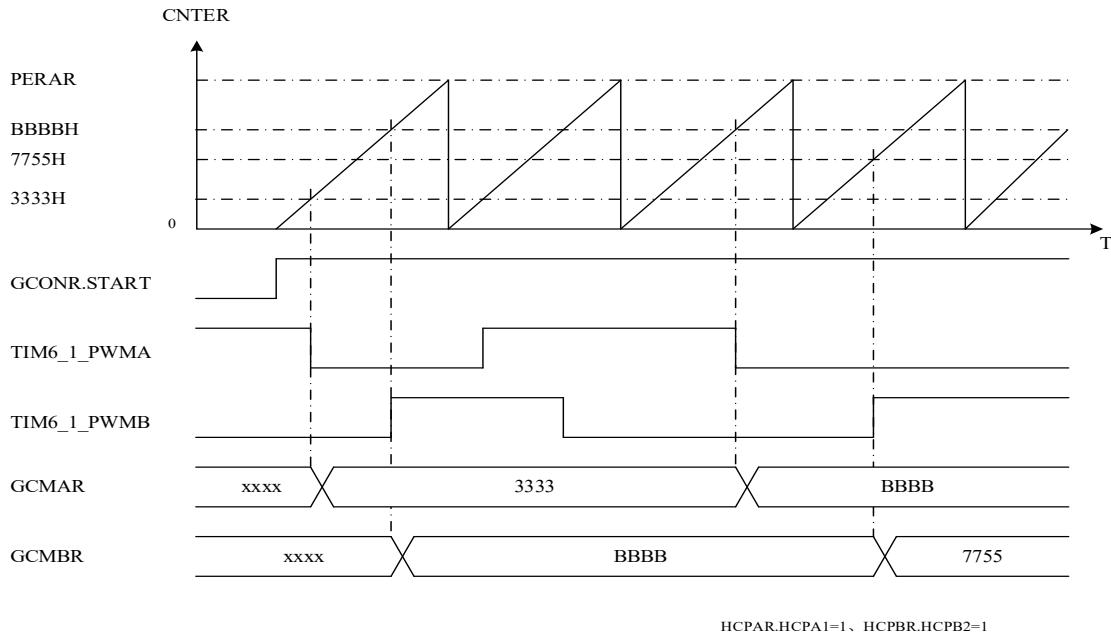


Figure 20-5 Input capture action

20.3.6 Counter update

Timer6 has a Update Value Register (UPDAR), which can update the count value of the Counter Value Register (CNTER) in real time when the count is stopped or counted.

The update condition of the count value is set by the corresponding bits in the Hardware Update Event Select Register (HUPDR) or the Software Sync Update Control Register (SUPDR). When the set update event is valid, the value of the Counter Value Register (CNTER) is updated to the value specified in the Update Value Register (UPDAR). Figure 20-6 is an example of the hardware update action during counting in the sawtooth waveform mode.

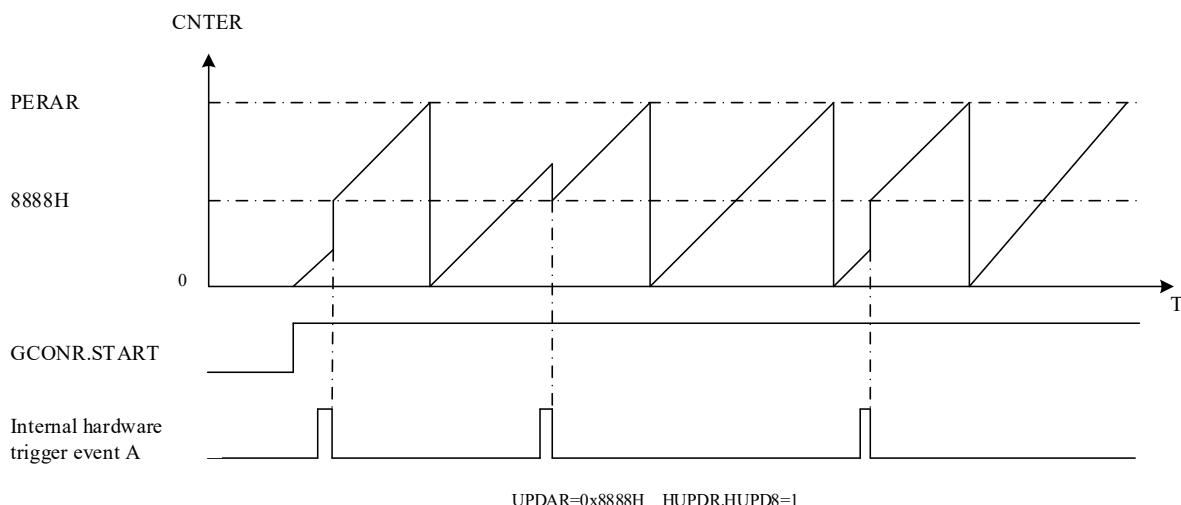


Figure 20-6 Hardware update action

20.3.7 Software synchronization

20.3.7.1 Software synchronous start

Each unit can realize the synchronous start of the target unit counter (CNTER) by setting the relevant bits of the Software Sync Start Control Register (SSTAR).

20.3.7.2 Software synchronous stop

Each unit can realize the synchronous stop of the target unit counter (CNTER) by setting the relevant bits of the Software Sync Stop Control Register (SSTPR).

20.3.7.3 Software synchronous clear

Each unit can realize the synchronous clear of the target unit counter (CNTER) by setting the relevant bits of the Software Sync Clear Control Register (SCLRR).

20.3.7.4 Software synchronous update

Each unit can realize the synchronous update of the target unit counter (CNTER) by setting the relevant bits of the Software Sync Update Control Register (SUPDR).

As Figure 20-7 shown, if set SSTAR.SSTA1=SSTAR.SSTA2=SSTAR.SSTA3=1, the software synchronous start of units 1~3 can be realized, set SSTPR.SSTP1=SSTPR.SSTP2=SSTPR.SSTP3=1, then the software synchronous stop of units 1~3 can be realized.

The software synchronization action related registers (SSTAR, SSTPR, SCLRR, SUPDR) are a group of registers that are independent of the units and shared by each unit. Each bit of this group's registers is only valid when writing 1, and invalid when writing 0. When reading the SSTAR register, the timer status of each unit (counting stopped or counting) is read out, and when reading SSTPR, SCLRR, or SUPDR, zero is read out.

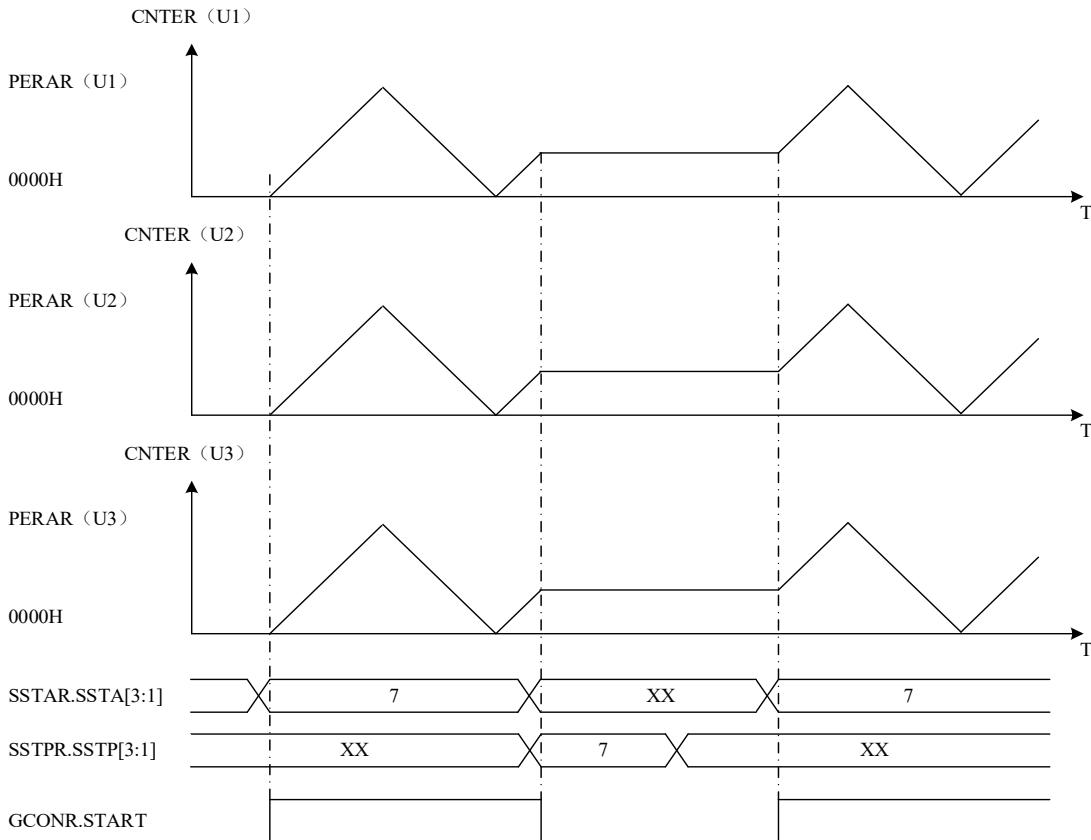


Figure 20-7 Software synchronization action

20.3.8 Hardware synchronization

In addition to 2 general input ports (TIM6_<t>_PWMA, TIM6_<t>_PWMB), each unit also has 4 general external trigger input ports (TIM6_TRIGA, TIM6_TRIGB, TIM6_TRIGC, TIM6_TRIGD) and 4 internal triggers event input conditions which can realize hardware synchronization between units.

The event source of the internal trigger event can be selected by the corresponding number setting in the Hardware Trigger Event Select Register (HTSSR0~3). For the detailed event correspondence, please refer to the INTC chapter. When using the internal trigger function, AOS function control bit of the Clock Control Register 0 (PWC_FCG0) should be set to 1 first.

20.3.8.1 Hardware synchronous start

Each unit can choose to start the timer by hardware, and the units 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 Select Register (HSTAR).

20.3.8.2 Hardware synchronous stop

Each unit can choose to stop the timer by hardware, and the units with the same hardware stop condition can realize synchronous stop when the stop condition is valid. The specific hardware stop condition is determined by the setting of the Hardware Stop Event Select Register (HSTPR).

20.3.8.3 Hardware synchronous clear

Each unit can choose to clear the timer by hardware, and the units with the same hardware clearing condition can realize synchronous clearing when the clearing condition is valid. The specific hardware clearing condition is determined by the setting of the Hardware Clear Event Select Register (HCLRR).

20.3.8.4 Hardware synchronous update

Each unit can choose to update the timer by hardware, and the units with the same hardware update condition can realize synchronous update when the update condition is valid. The specific hardware update condition is determined by the setting of the Hardware Update Event Select Register (HUPDR).

20.3.8.5 Hardware synchronous input capture

Each unit can choose to realize the input capture function by hardware, and the unit with the same input capture function condition can realize synchronous input capture when the input capture function condition is valid. The specific hardware input capture function condition is determined by the settings of the Hardware Input Capture Event Select Registers (HCPAR, HCPBR).

20.3.8.6 Hardware synchronization count

Each unit can choose to use the hardware input as CLOCK to count, and the units with the same hardware counting condition can realize synchronous counting when the hardware counting clock is valid. The specific hardware counting conditions are determined by the settings of the Hardware Count Up Event Select Register (HCUPR) and the Hardware Count Down Event Select Register (HCDOR).

When the hardware synchronous counting function is selected, only the external input clock source is selected, and the start, stop and clear actions of the timer are not affected. The start, stop, and clear of the timer also need to be set separately.

Figure 20-8 shows an example of the hardware synchronization operation of units 1 to 3.

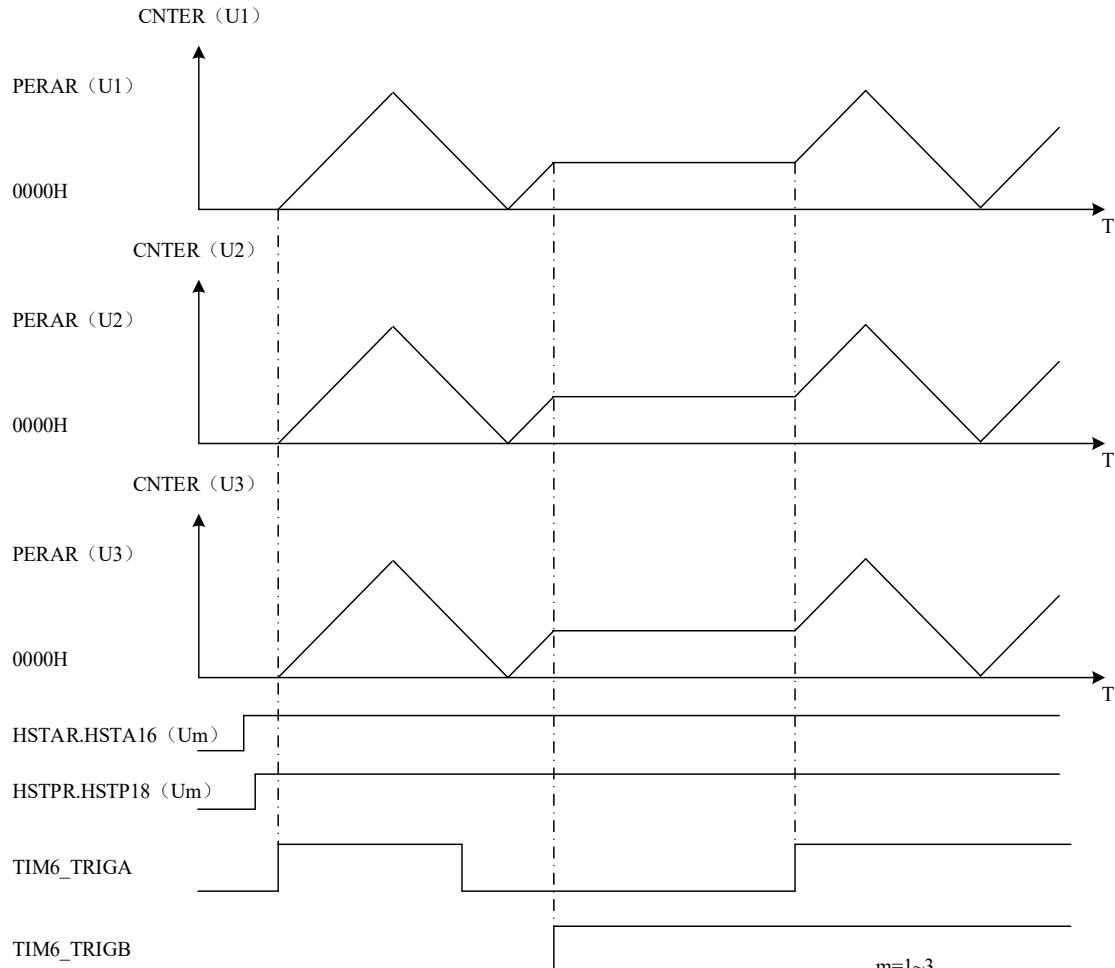


Figure 20-8 Hardware synchronization action

20.3.9 Pulse width measurement

When using the hardware trigger related functions of the TIM6_<t>_TRIGA~D port (refer to the hardware synchronization chapter), each unit can implement two independent pulse width measurement functions.

For example, if the hardware start condition of the counter is set to the rising edge of TIM6_<t>_TRIGA, the hardware clear condition, stop condition and the input capture condition of the GCMAR register are set to the falling edge of TIM6_<t>_TRIGA, then continuous pulse width measurement can be achieved. Figure 20-9 shows corresponding actions.

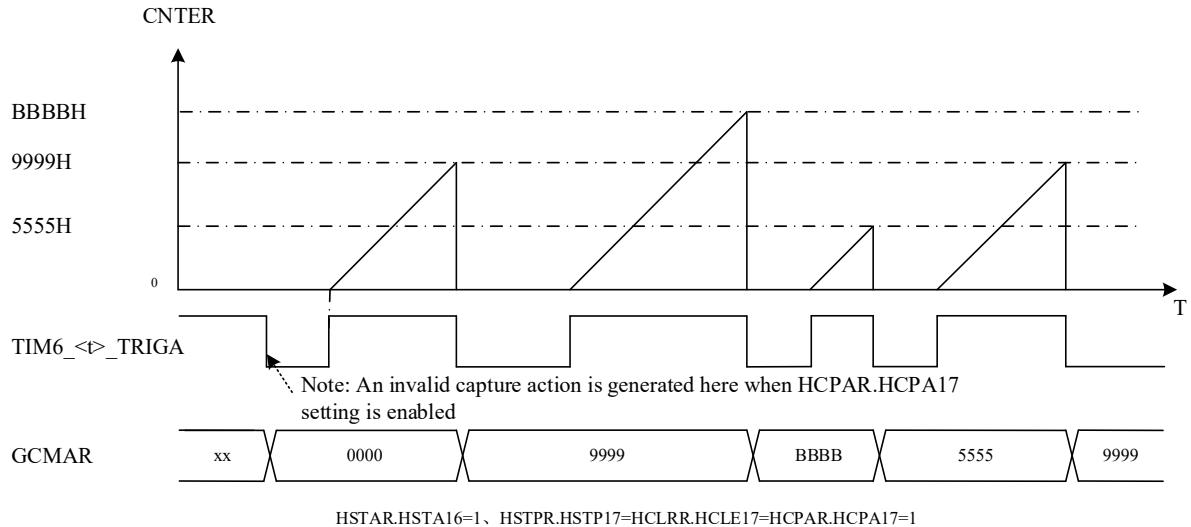


Figure 20-9 Pulse width measurement

20.3.10 Period measurement

When using the hardware trigger related functions of the TIM6_<t>_TRIGA~D port (refer to the hardware synchronization chapter), each unit can implement 2 independent period measurement functions.

For example, by setting the hardware start condition of the counter, the hardware clear condition, and the input capture condition of the GCMBR register to the rising edge of TIM6_<t>_TRIGB, continuous period measurement can be achieved. Figure 20-10 shows corresponding actions.

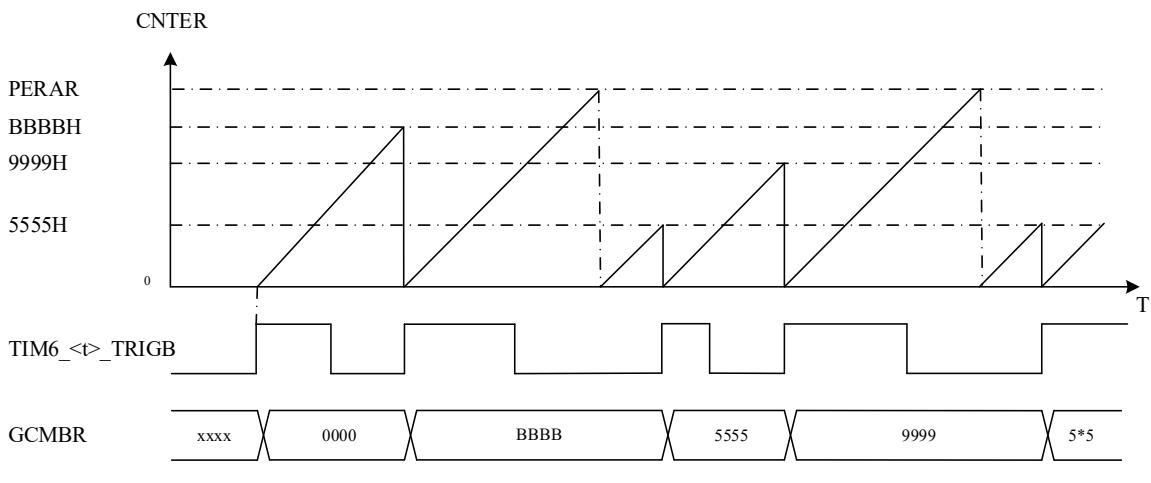


Figure 20-10 Period measurement

20.3.11 Buffer function

The period value, general compare value, special compare value, dead time value, etc. of Timer6 all have the buffer function, which can realize the period change, duty change, dead time change, etc. by hardware when the timer is counting. Period value, general compare value and special compare value have single buffer and double buffer functions, and dead time value has single buffer function.

20.3.11.1 Single buffer action

Single buffer action means that by setting the Buffer Control Register (BCONR.BENA<P><SPA><SPB>=1, BCONR.BSEA<P><SPA><SPB>=0), the DeadTime Control Register (DCONR.DTBENU<D>=1) , select the following events to happen at the buffer transfer time point:

1. The value of the Period Value Buffer Register (PERBR) is automatically transferred to the Period Value Register (PERAR)
2. The values of the General Compare Value Buffer Registers (GCMCR, GCMDR) are automatically transferred to the General Compare Value Registers (GCMAR, GCMBR) (when comparison output)
3. The values of the General Compare Value Registers (GCMAR, GCMBR) are automatically transferred to the General Compare Value Buffer Registers (GCMCR, GCMDR) (when input capture)
4. The values of the Special Compare Value Buffer Registers (SCMCR, SCMDR) are automatically transferred to the Special Compare Value Registers (SCMAR, SCMBR)
5. The value of the DeadTime Value Buffer Register (DTUBR, DTDBR) is automatically transferred to the DeadTime Value Register (DTUAR, DTDAR)

Figure 20-11 shows a timing chart of the single buffer action of the General Compare Value Register when the unit 1 is set to comparison output operation (PCNAR.CAPMDA=0). As can be seen from the figure, changing the value of the General Compare Value Register (GCMAR) during the count can adjust the output duty, and changing the value of the Period Value Register (PERAR) can adjust the output period.

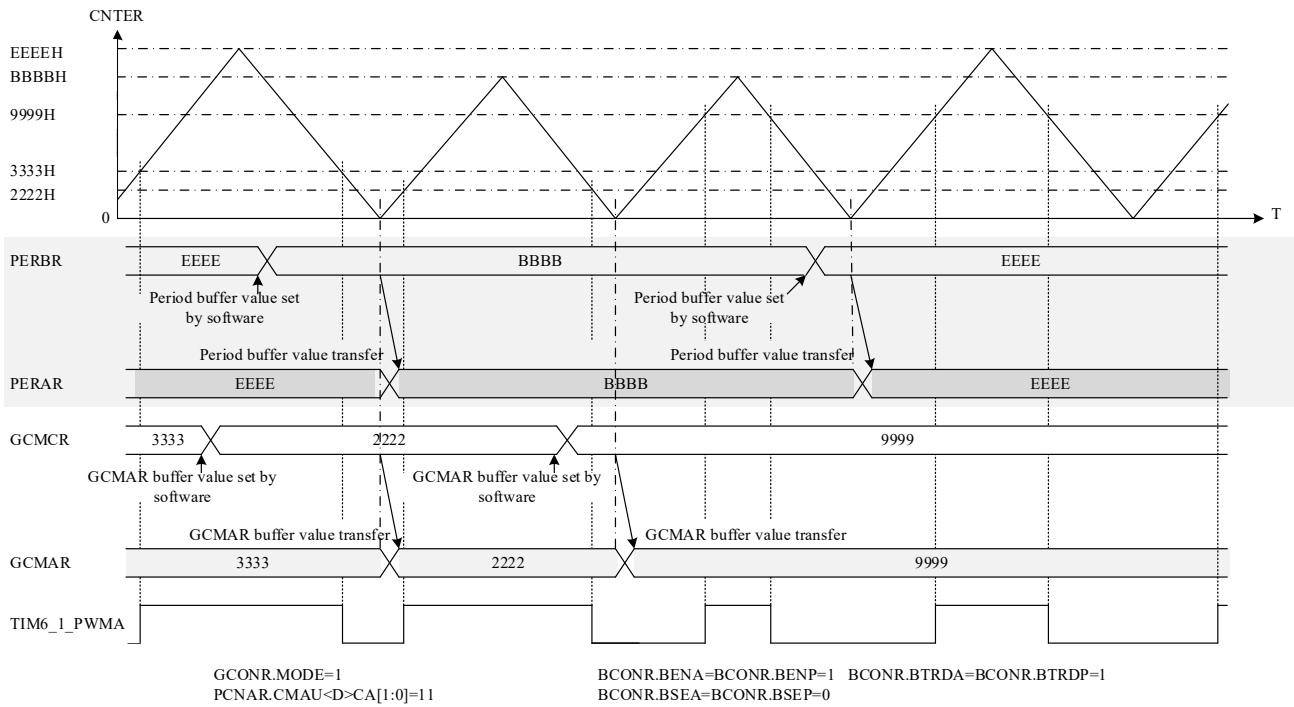


Figure 20-11 Timing of comparison output with Single buffer

20.3.11.2 Double buffer action

Double buffer action means that by setting the Buffer Control Register ($\text{BCONR.BENA}<\text{B}><\text{P}><\text{SPA}><\text{SPB}>=1$, $\text{BCONR.BSEA}<\text{B}><\text{P}><\text{SPA}><\text{SPB}>=1$), select the following events to happen at buffer transfer time point:

1. The value of the Period Value Buffer Register (PERBR) is automatically transferred to the Period Value Register (PERAR), and the value of the Period Value Double Buffer Register (PERCR) is automatically transferred to the Period Value Buffer Register (PERBR).
2. The values of the General Compare Value Buffer Registers (GCMCR, GCMDR) are automatically transferred to the General Compare Value Registers (GCMAR, GCMBR), and the values of the General Compare Value Double Buffer Registers (GCMER, GCMFR) are automatically transferred to the General Compare Value Buffer Registers (GCMCR, GCMDR) (when comparison output).
3. The values of the General Compare Value Buffer Registers (GCMCR, GCMDR) are automatically transferred to the General Compare Value Double Buffer Registers (GCMER, GCMFR), and the values of the General Compare Value Registers (GCMAR, GCMBR) are automatically transferred to the General Compare Value Buffer Registers (GCMCR, GCMDR) (when capturing input).
4. The value of the Special Compare Value Buffer Registers (SCMCR, SCMDR) is automatically transferred to the Special Compare Value Registers (SCMAR, SCMBR), and the value of the Special Compare Value Double Buffer Registers (SCMER, SCMFR) are automatically transferred to the Special Compare Value Buffer Registers (SCMCR, SCMDR). Figure 20-12

shows the timing diagram of the double buffer action when the internal trigger event 0 triggers the input capture.

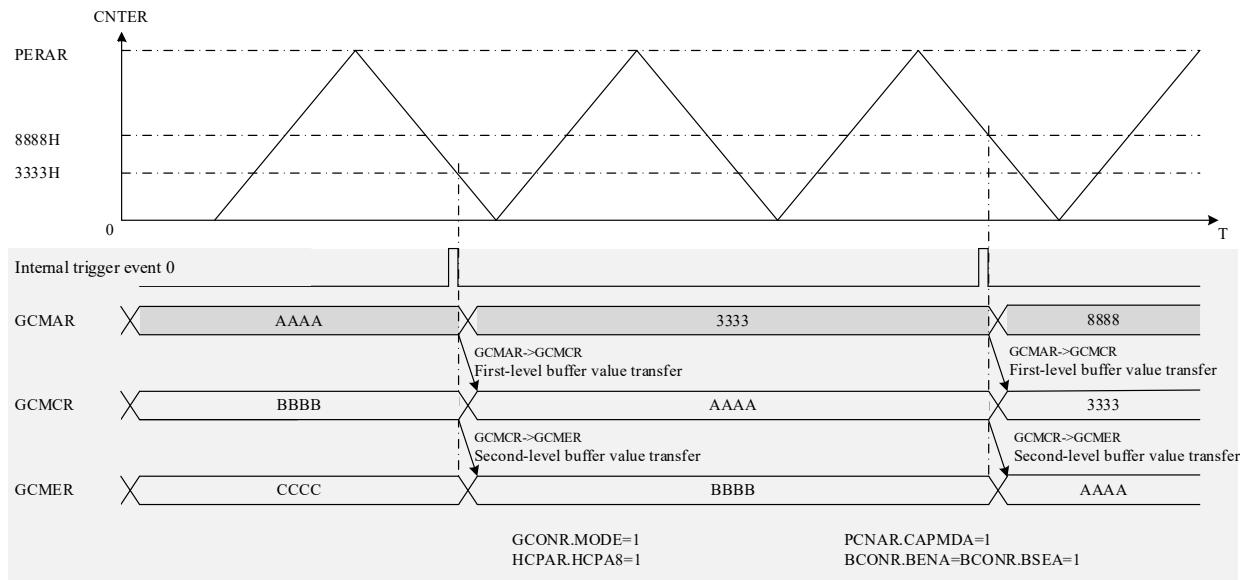


Figure 20-12 Timing of input capture with double buffer

20.3.11.3 Buffer transfer node

Buffer transfer in comparison output (sawtooth waveform)

When buffer function is enabled ($\text{BCONR.BENA} < \text{B} > < \text{P} > < \text{SPA} > < \text{SPB} > = \text{DCONR.DTBENU} < \text{D} > = 1$) && comparison output ($\text{PCNA} < \text{B} > \text{R.CAPMDA} < \text{B} > = 0$) && sawtooth waveform count mode ($\text{GCONR.MODE} = 0$), the buffer transfer of the period value, general compare value, special compare value, and dead time value occurs at the up counting overflow point or down counting underflow point.

Figure 20-13 shows the buffer operation of sawtooth waveform during counting up.

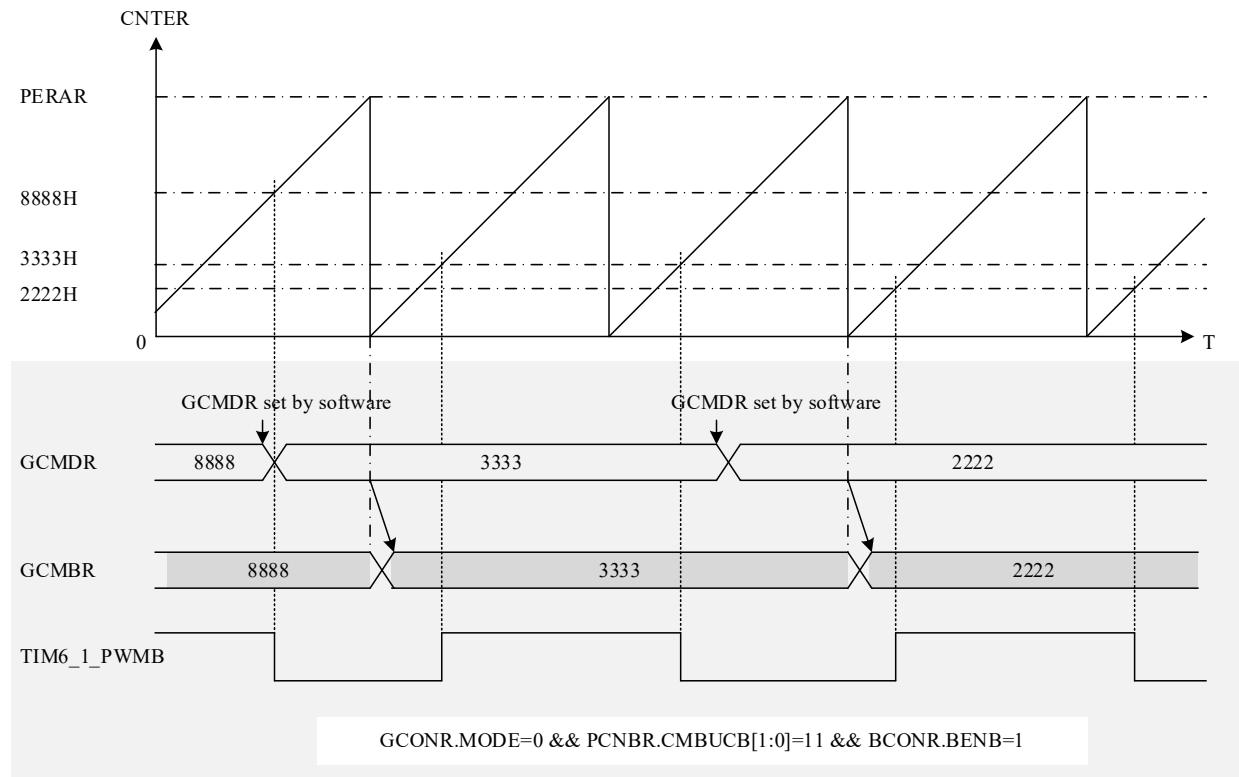


Figure 20-13 Count buffer action in sawtooth waveform mode

Note:

- In the sawtooth waveform count mode, if a clear action occurs, it is also regarded as a count overflow. For each value, a buffer transfer occurs once according to the corresponding buffer action setting status (single buffer, double buffer, etc.).
- In the hardware counting mode, if a clear action occurs, it is also regarded as a count overflow. For the period value and the general compare value, a buffer transfer occurs once according to the corresponding buffer action setting status (single buffer, double buffer, overflow transfer, underflow transfer, etc.), and no buffer transfer occurs for other values.

Buffer transfer in comparison output (triangular waveform)

When buffer function ($\text{BCONR.BENA}_{<\text{B}>} \text{<P>} \text{<SPA>} \text{<SPB>} = \text{DCONR.DTBENU}_{<\text{D}>} = 1$) is enabled && compare output count ($\text{PCNA}_{<\text{B}>} \text{.CAPMDA}_{<\text{B}>} = 0$) && triangular waveform count mode ($\text{GCONR.MODE} = 0$), the buffer transfer time points of period value, general compare value, special compare value and dead time value are determined by the corresponding register control bits. When $\text{BCONR.BTRUA}_{<\text{B}>} \text{<P>} \text{<SPA>} \text{<SPB>} = 1$ or $\text{DCONR.DTBTRU} = 1$, the corresponding buffer transfer occurs when the counter counts to the peak point of the triangular waveform; when $\text{BCONR.BTRDA}_{<\text{B}>} \text{<P>} \text{<SPA>} \text{<SPB>} = 1$ or $\text{DCONR.DTBTRD} = 1$, the corresponding buffer transfer occurs when the counter counts to the valley point of the triangular waveform.

Figure 20-14 shows the buffer action when the triangular waveform counts to the valley point. Figure 20-15 shows the buffer actions when the triangular waveform counts to the peak point and the valley point.

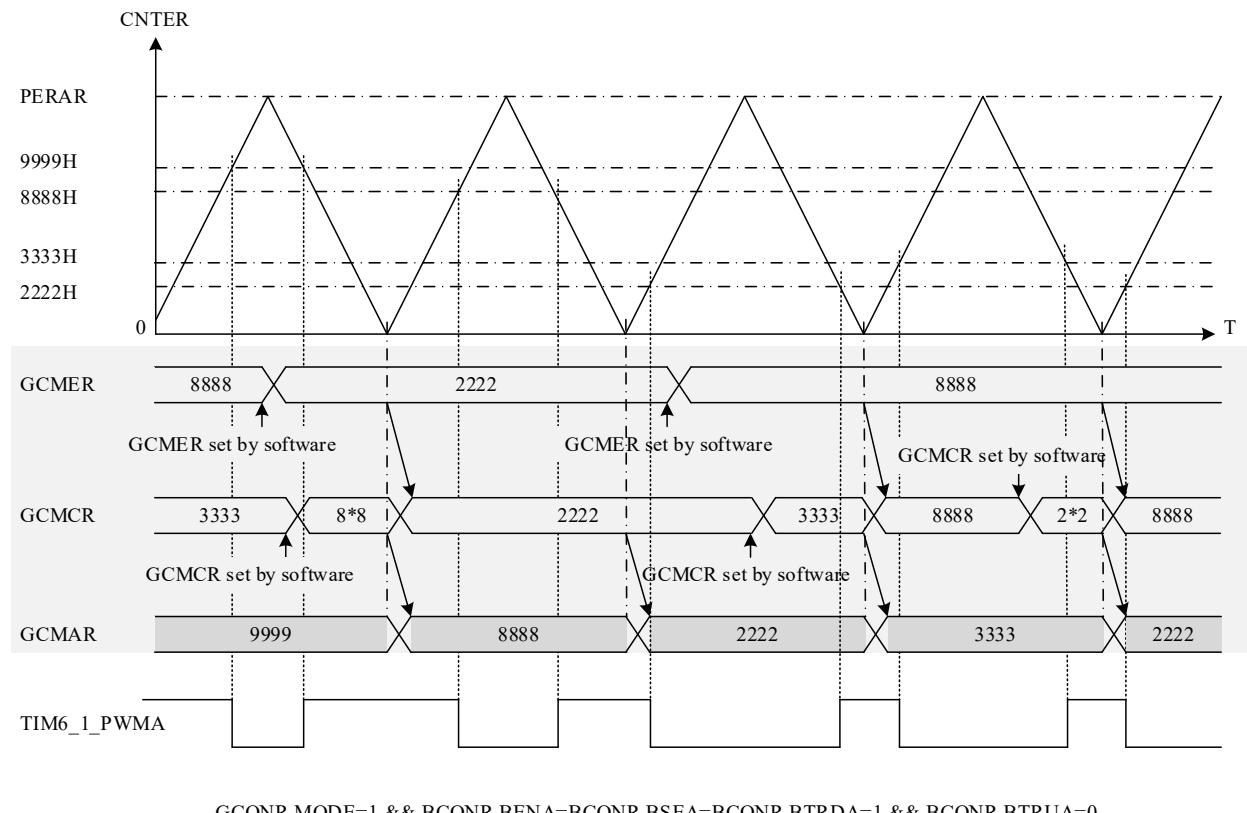


Figure 20-14 Count buffer action 1 in triangular waveform mode

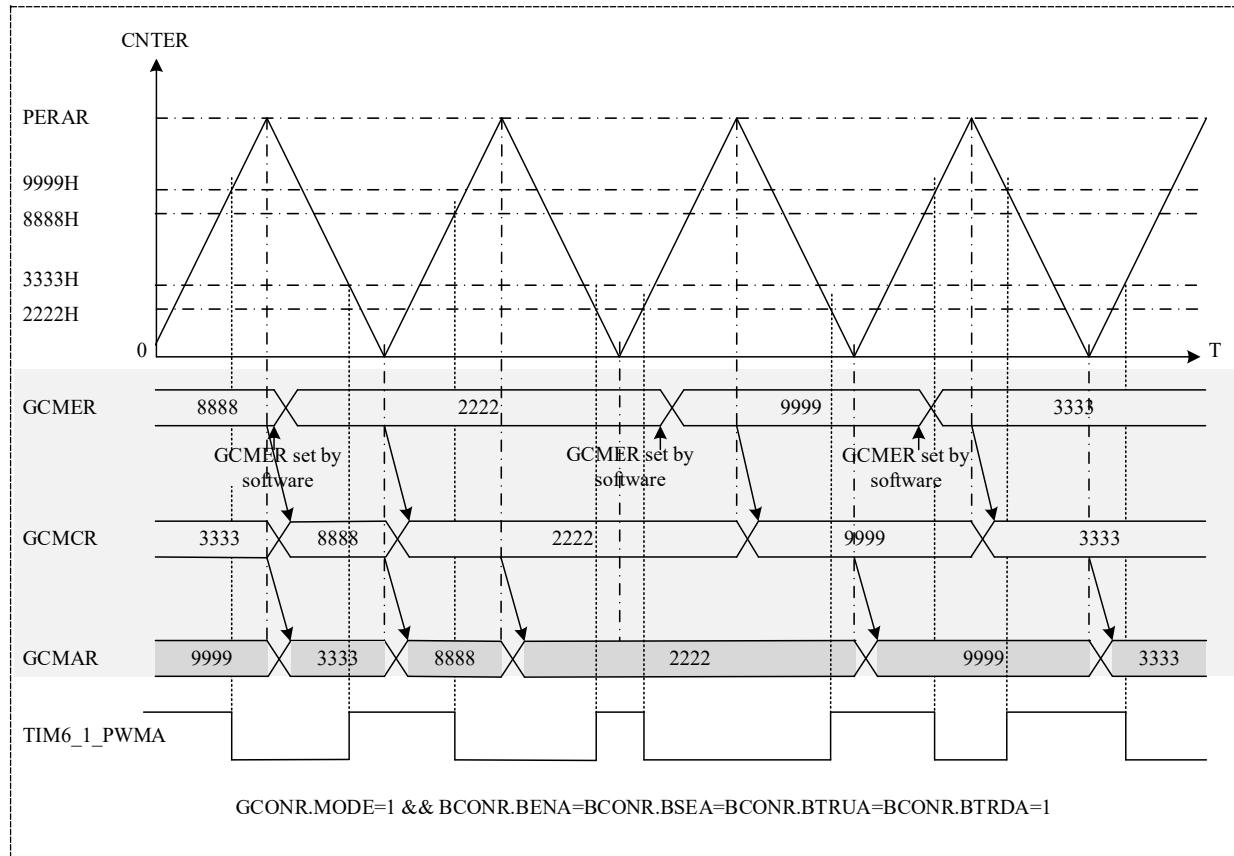


Figure 20-15 Count buffer action 2 in triangular waveform mode

Buffer transfer in Input capture

When the input capture action ($PCNA<1>R.CAPMDA<1>=1$) is valid, the general value supports the buffer function. If the buffer function is enabled ($BCONR.BENA<1>=1$), a buffer transfer occurs at the input capture action point. Input capture action can select single buffer function or double buffer function (set by $BCONR.BSEA<1>$).

20.3.12 Digital filtering

`TIM6_<t>_PWMA`, `TIM6_<t>_PWMB`, `TIM6_TRIGA~D` port inputs all have digital filtering function. `TIM6_<t>_PWMA` and `TIM6_<t>_PWMB` port filtering function can be enabled by setting the corresponding enable bits of the Filter Control General Port Register (FCNGR), and the filtering reference clock can also be set by the Filter Control General Port Register (FCNGR) when the filtering is valid; `TIM6_TRIGA~D` ports are a group of ports shared among units, the filter function of these ports can be enabled by setting the corresponding enable bit of the Filter Control Trigger Port Register (FCNTR), the filter reference clock can also be set by the Filter Control Trigger Port Register (FCNTR).

When the level on the port is sampled same for three times by filter sampling reference clock, the level is transferred into the module as an effective level. A level is sampled same for less than three times will be filtered out as external noise and will not be transferred into the module. Figure 20-16 shows filtering action.

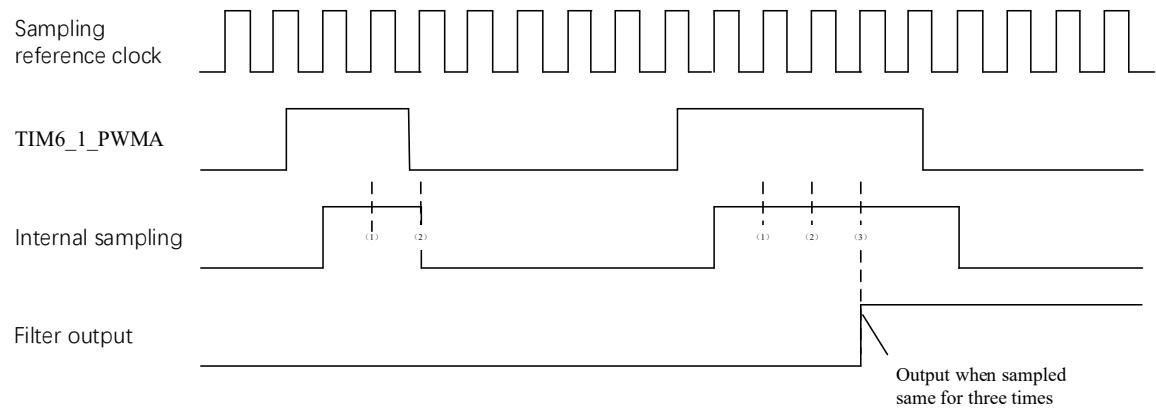


Figure 20-16 Filtering function at input capture port

20.3.13 General PWM output

20.3.13.1 Unilaterally aligned independent PWM output

In the sawtooth waveform counting mode (GCONR.MODE=0), the 2 ports TIM6_<t>_PWMA and TIM6_<t>_PWMB of each unit can output PWM waveform independently. If the same level change is set at the count overflow point, the unilaterally aligned independent PWM output can be realized. As Figure 20-17 shown.

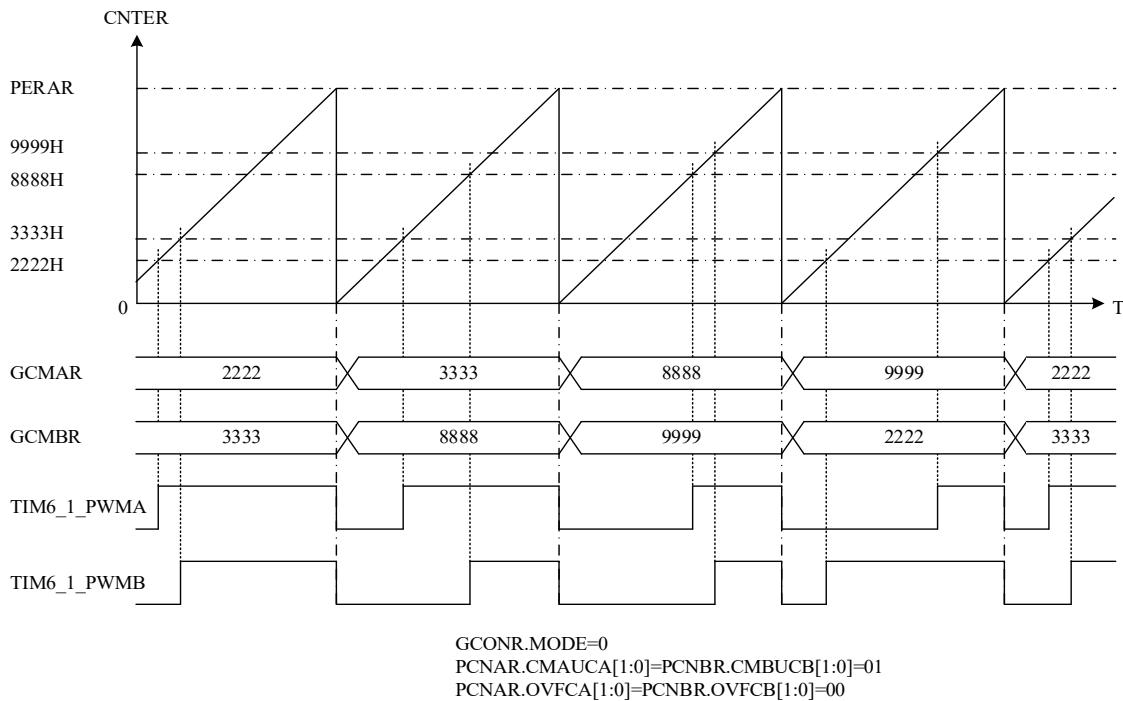


Figure 20-17 Unilaterally aligned independent PWM

20.3.13.2 Bilateral symmetrical independent PWM output

In the triangular waveform counting mode (GCONR.MODE=1), the two ports TIM6_<t>_PWMA and TIM6_<t>_PWMB of each unit can output PWM waveform independently. If level change is set at the count comparison matching point, and no level change is set at the count peak and valley points, bilateral symmetrical independent PWM output can be achieved. As Figure 20-18 shown, TIM6_<t>_PWMA port and TIM6_<t>_PWMB port realize bilateral symmetrical independent output PWM.

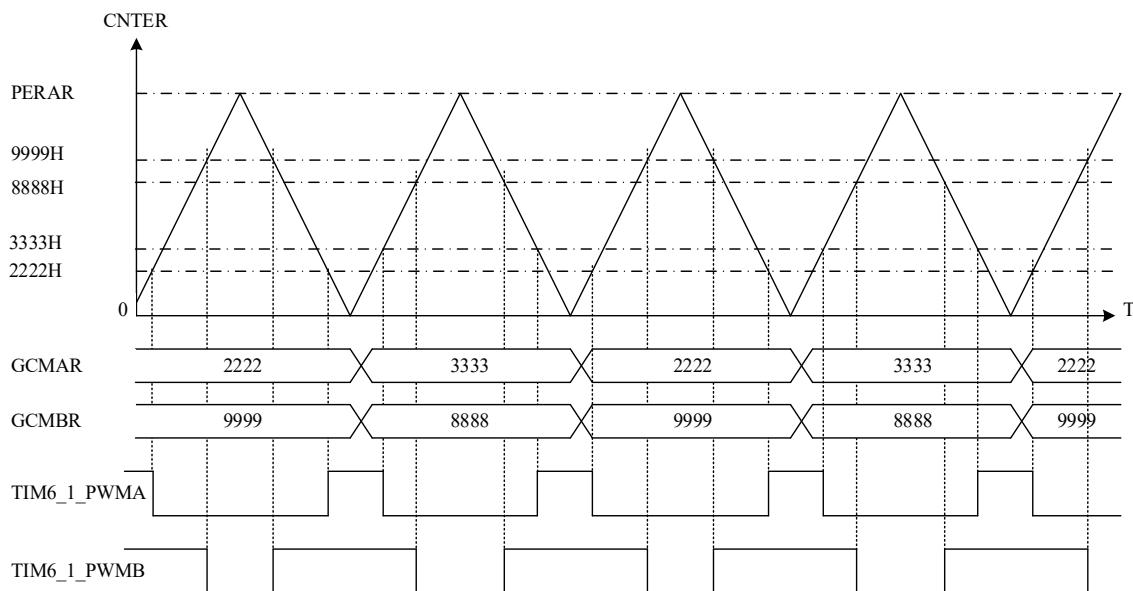


Figure 20-18 Bilateral symmetrical independent PWM

20.3.13.3 Bilateral symmetrical complementary PWM output

In the triangular waveform counting mode (GCONR.MODE=1), outputting a pair of complementary PWM waveforms from TIM6_<t>_PWMA and TIM6_<t>_PWMB ports can be realized by setting the level of the ports when the count starts, comparison matches and count overflows.

According to the different assignment methods of GCMBR value, the output of bilateral symmetrical complementary PWM can be divided into "complementary PWM output by software setting of GCMBR" and "complementary PWM output by hardware setting of GCMAR".

Complementary PWM output by software setting of GCMBR

The software setting method means that in the triangular waveform mode, the General Compare Value Register (GCMBR) used for the waveform output of the TIM6_<t>_PWMB port is directly written by the CPU, etc., and has no direct relationship with the value of GCMAR.

Figure 20-19 shows the example of complementary PWM waveform output by software setting of GCMBR.

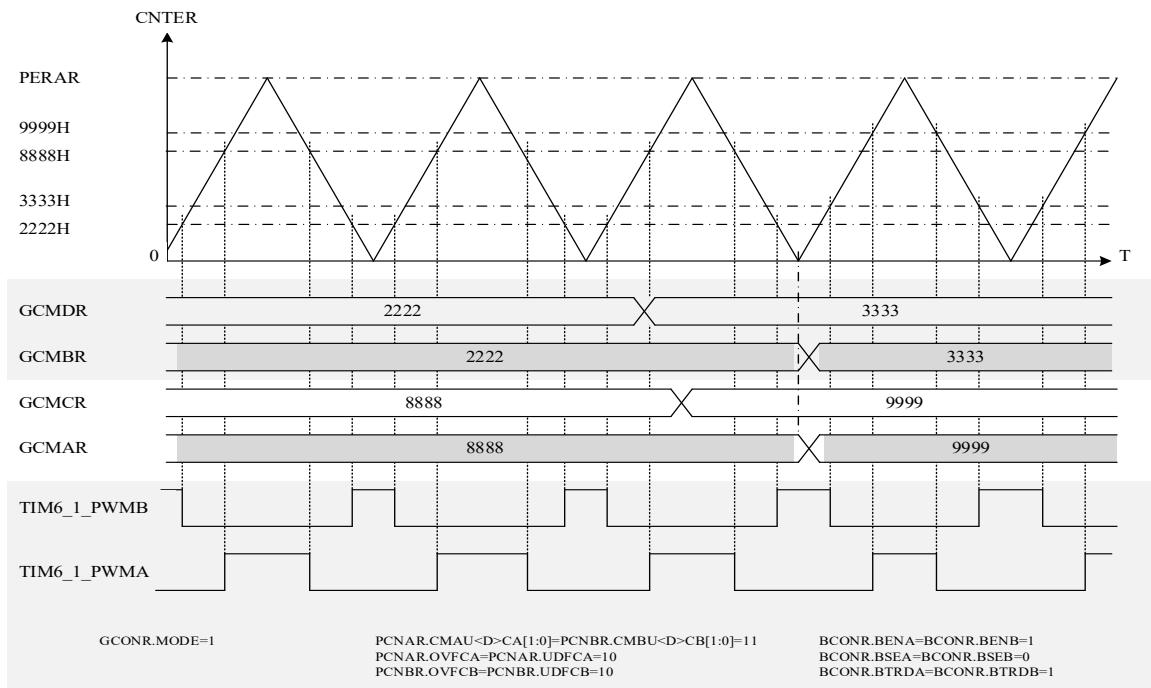
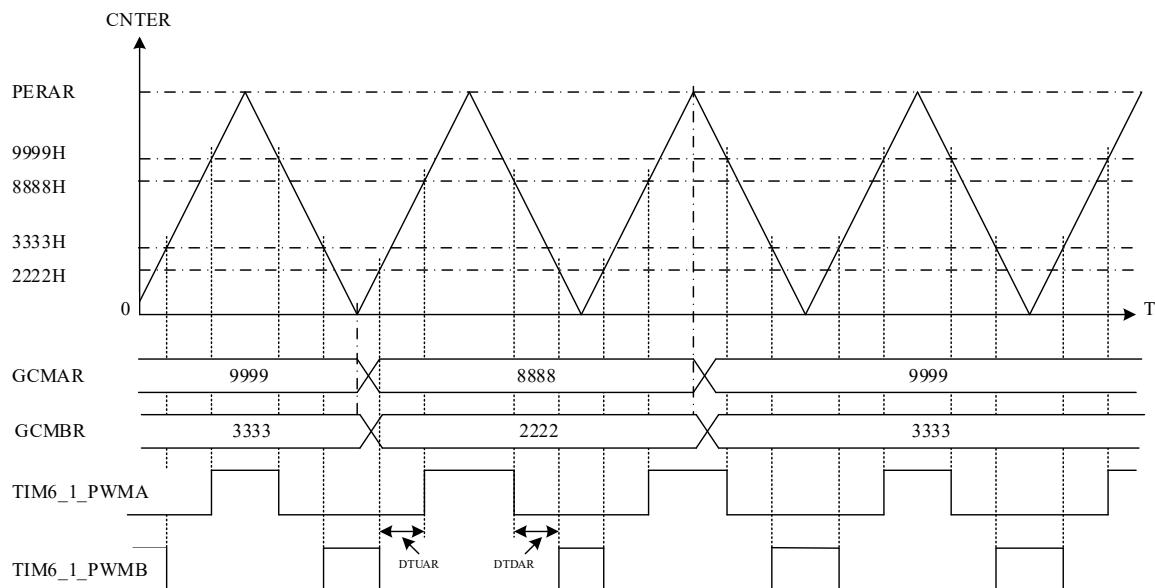


Figure 20-19 Complementary PWM waveform output by software setting of GCMBR

Complementary PWM output by hardware setting of GCMBR

The hardware setting method means that in the triangular waveform mode, the value of the General Compare Value Register (GCMBR) used for the waveform output of the TIM6_<t>_PWMB port is determined by the calculation of General Compare Value Register (GCMAR) and the DeadTime Value Register (DTU)<D>AR).

Figure 20-20 shows the example of complementary PWM waveform output by hardware setting of GCMBR.



When up counting, GCMBR=GCMAR-DTUAR; when down counting, GCMBR=GCMAR-DTDAR

GCONR.MODE=1

PCNAR.CMAU<D>CA[1:0]=PCNBR.CMBU<D>CB[1:0]=11
PCNAR.OVFCA=PCNAR.UDFCA=10
PCNBR.OVFCB=PCNBR.UDFCB=10

DCONR.DTCEN=1
DCONR.SPEA=1
DTUAR=DTDAR=0x6666H

Figure 20-20 Complementary PWM waveform output by hardware setting of GCMBR

20.3.13.4 Bilateral asymmetric PWM output

In the triangular waveform counting mode (GCONR.MODE=1), the two ports TIM6_<t>_PWMA and TIM6_<t>_PWMB of each unit can output PWM waveforms independently, and each port can perform the corresponding level output change by the value of General Compare Value Register (GCMAR, GCMBR). If the port level change of TIM6_<t>_PWMA is controlled respectively in the up counting period and the down counting period by the compare result of the GCMAR and GCMBR values, the bilateral asymmetric PWM can be realized on the TIM6_<t>_PWMA port. Figure 20-21 shows the bilateral asymmetric PWM output of the TIM6_<t>_PWMA port.

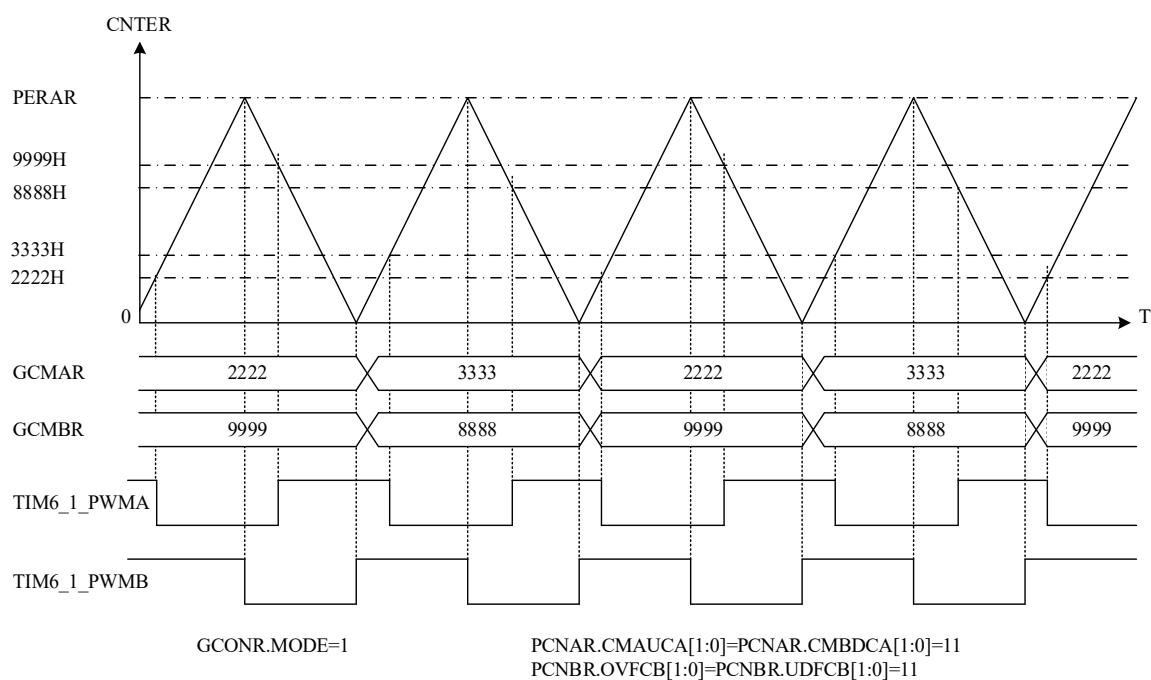


Figure 20-21 Bilateral asymmetrical PWM output

20.3.13.5 Interunit multi-phase PWM output

The TIM6_<t>_PWMA and TIM6_<t>_PWMB ports of each unit can output 2-phase PWM waveforms, multi-phase PWM waveform output can be realized by combination among multiple units and software and hardware synchronous actions . As Figure 20-22 shown, combination of unit 1, unit 2 and unit 3 outputs 6-phase unilaterally aligned independent PWM; as Figure 20-23 shown, combination of unit 1, unit 2 and unit 3 outputs 3-phase bilateral symmetrical complementary PWM.

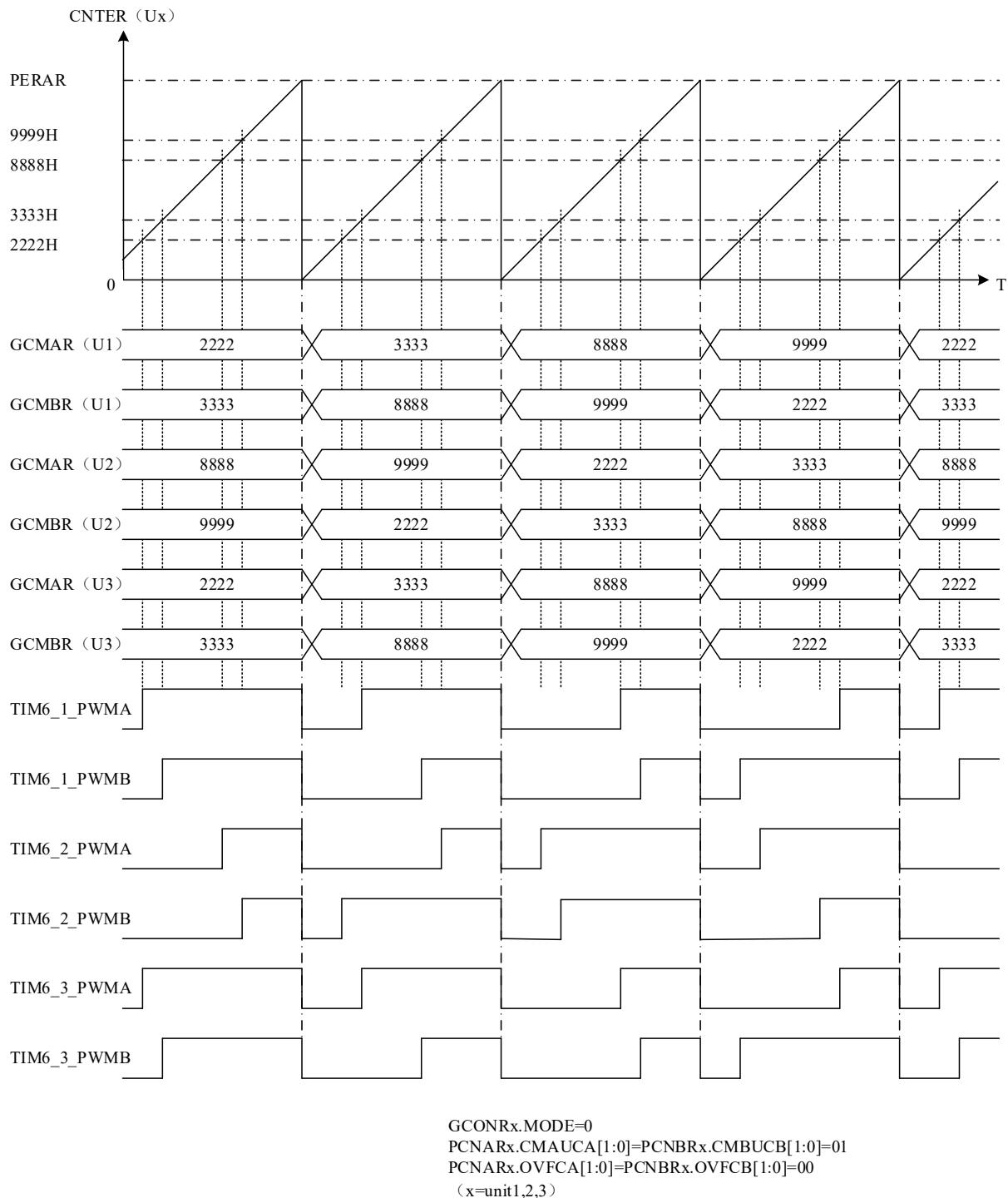


Figure 20-22 6-phase unilateral aligned independent PWM

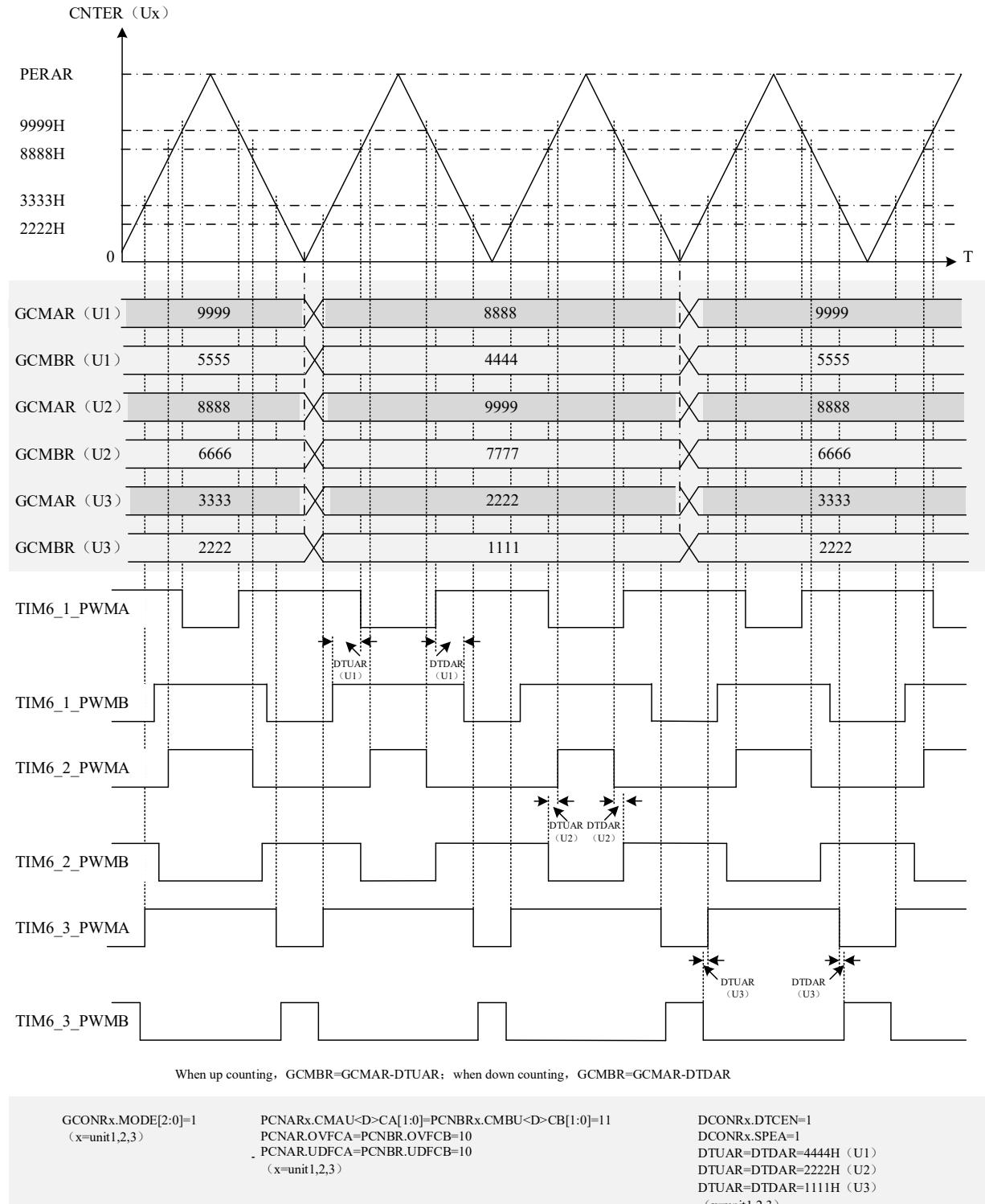


Figure 20-23 3-phase bilateral symmetrical complementary PWM with dead time

20.3.14 Periodic interval response

The two Special Compare Value Registers (SCMAR, SCMBR) of Timer6 can respectively output the special compare match interrupt A signal and the special compare match interrupt B signal to INTC to generate corresponding interrupts when the count comparison matches; at same time respectively output special compare match event A signal and the special compare match event B signal which are used to associate actions with other modules, mostly used to start ADC, etc.

The request signal of the interrupt and event can generate a valid request signal every several periods, that is, realize the periodic interval response. This function is enabled by setting the VPERR.PCNTE[1:0] bits and the VPERR.SPPERIA/B bits of the Valid Period Register (VPERR). Set the VPERR.PCNTS[2:0] bits to specify how many periods the request signal is valid once. In other periods, even if the count value and the value of the Special Compare Value Register SCMAR or SCMBR are equal, the valid request signal will not be output.

After this function is enabled, the period match interrupt and period match event in each waveform mode are only output in the valid period of the special compare match interrupt and event output (the period when STFLR.VPERNUM=0 in the figure below). Figure 20-24 shows an example of the operation of the periodic interval valid request signal.

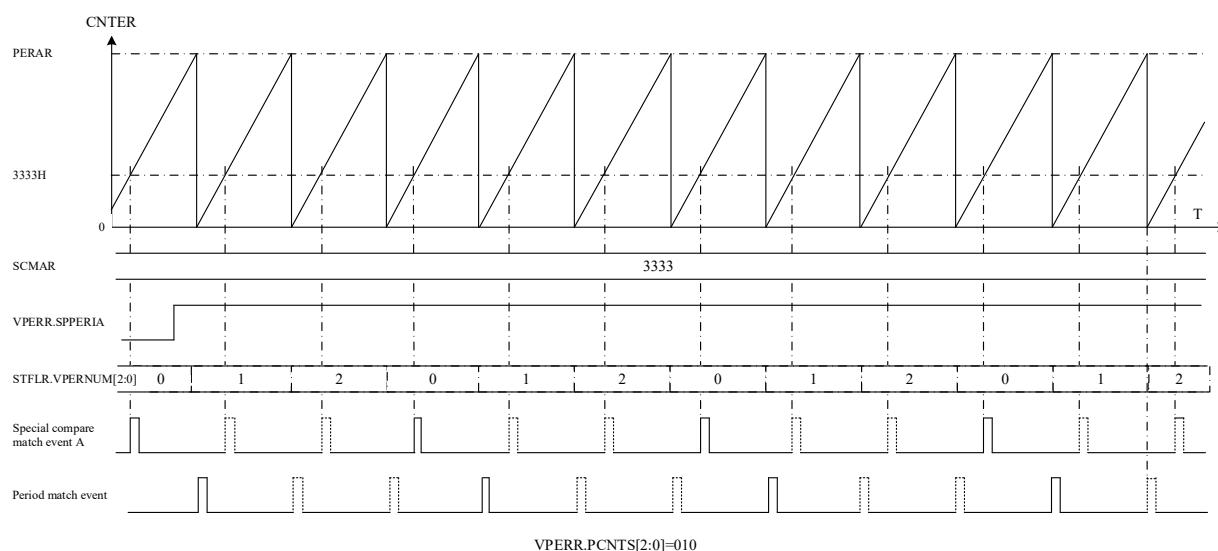


Figure 20-24 Action of periodic interval valid request signal

20.3.15 Quadrature encoding count

Considering the TIM6_<t>_PWMA input as the AIN input, the TIM6_<t>_PWMB input as the BIN input, and any of the TIM6_TRIGA~D inputs as the ZIN input, the Timer6 can realize the quadrature encoding count of the three inputs.

The single action of AIN and BIN of one unit can realize the position counting mode; the combined action of AIN, BIN and ZIN of two units can realize the revolution counting mode, in which one unit is used for position counting, and the other unit is used for revolution counting.

In revolution counting mode, the 8 units of Timer6 can be selected as position counting unit or revolution counting unit in any combination, but the combination is fixed as follows when the Z-phase mask function is enabled:

1. unit 1 and 2 are combined, unit 1 is used as the position counting unit and unit 2 is used as the revolution counting unit, they realize position counting and revolution counting respectively;
2. unit 3 and 4 are combined, unit 3 is used as the position counting unit and unit 4 is used as the revolution counting unit, they realize position counting and revolution counting respectively;
3. unit 5 and 6 are combined, and unit 5 is used as the position counting unit and unit 6 is used as the revolution counting unit , theu realize position counting and revolution counting respectively;
4. unit 7 and 8 are combined, unit 7 is used as the position counting unit and unit 8 is used as the revolution counting unit,they realize position counting and revolution counting respectively.

The counting conditions of AIN and BIN are realized by setting the quadrature relationship between TIM6_<t>_PWMA and TIM6_<t>_PWMB in the Hardware Count Up Event Select Register (HCUPR) and Hardware Count Down Event Select Register (HCDOR); the input action of ZIN realizes the clearing of position timer in position counting unit by setting the Hardware Clear Event Select Register (HCLRR), realizes the counting of revolution timer in revolution counting unit by setting the Hardware Count Up Event Select Register (HCUPR) .

20.3.15.1 Position counting mode

The quadrature encoding position counting mode refers to realizing the basic counting function, phase difference counting function and direction counting function according to the input of AIN and BIN.

Basic count

The basic counting action is to count according to the input clock of AIN or BIN port, as Figure 20-25 shown.

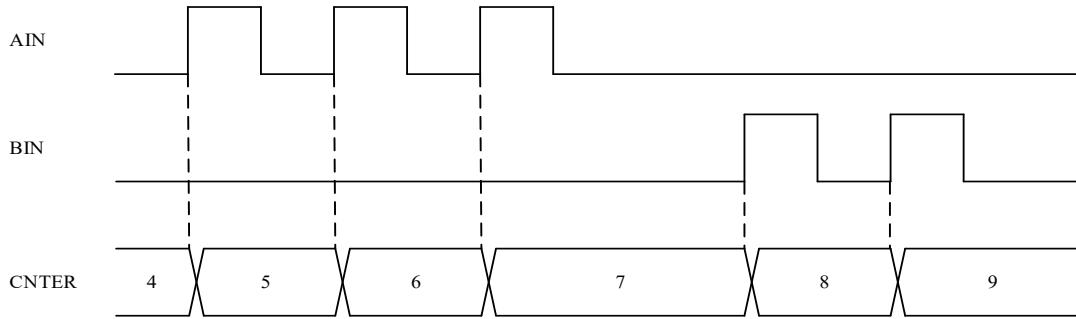
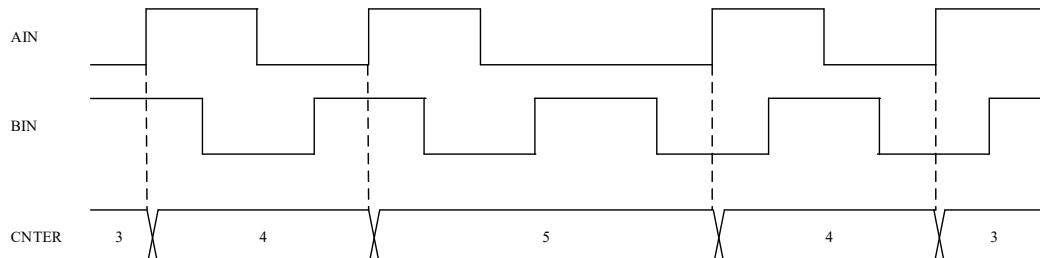


Figure 20-25 Position mode-basic counting

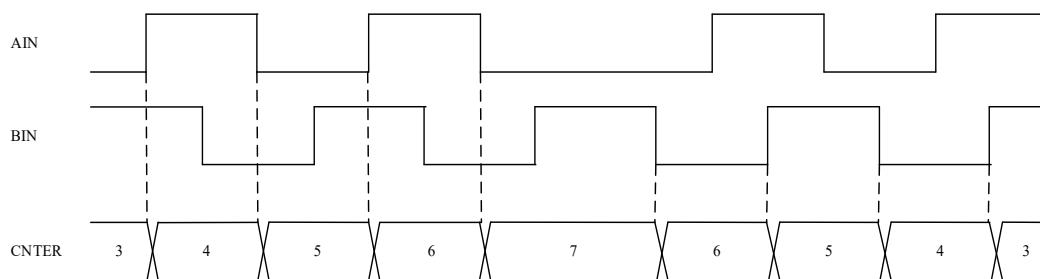
Phase difference count

Phase difference counting is to count according to the phase relationship between AIN and BIN. Depending on the difference of setting, 1 time counting, 2 times counting, 4 times counting, etc. can be realized, as Figure 20-26 ~Figure 20-28 shown.



HCUPR.HCUP6=1 && HCDOR.HCDO4=1

Figure 20-26 Position counting mode-phase difference counting (1 time counting)



HCUPR.HCUP6=HCUPR.HCUP5=1 && HCDOR.HCDO2=HCDOR.HCDO1=1

Figure 20-27 Position counting mode-phase difference counting (2 times counting)

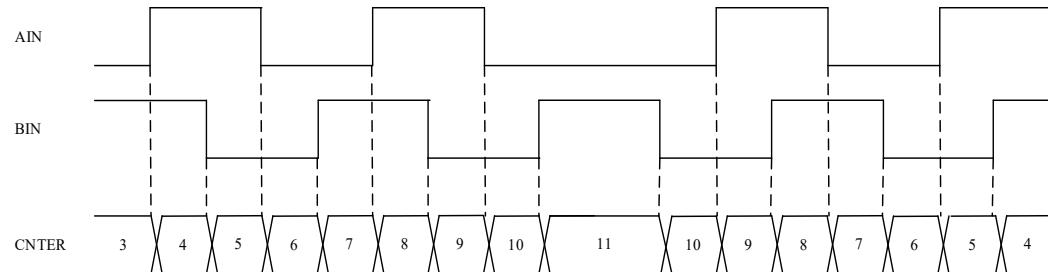


Figure 20-28 Position counting mode-phase difference counting (4 times counting)

Direction count

Direction counting refers to setting the input level of AIN as direction control, and using the input of BIN as counting clock, as Figure 20-29 shown.

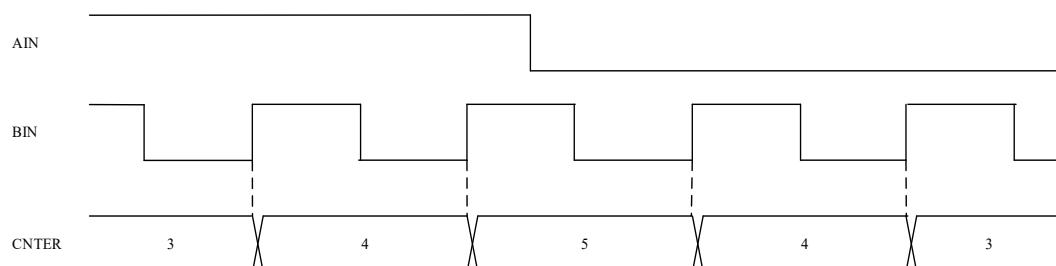


Figure 20-29 Position counting mode-direction counting

20.3.15.2 Revolution counting mode

Quadrature encoding revolution counting mode means that on the basis of AIN and BIN counting, input events of ZIN are added to realize the judgment of revolution number and so on. In revolution counting mode, according to the counting method of revolution timer, Z-phase counting function, position overflow counting function and mixed counting function can be realized.

Z-phase count

Z-phase counting refers to the counting action that revolution counting unit performs counting and clears the position counting unit according to the input of ZIN. as Figure 20-30 shown.

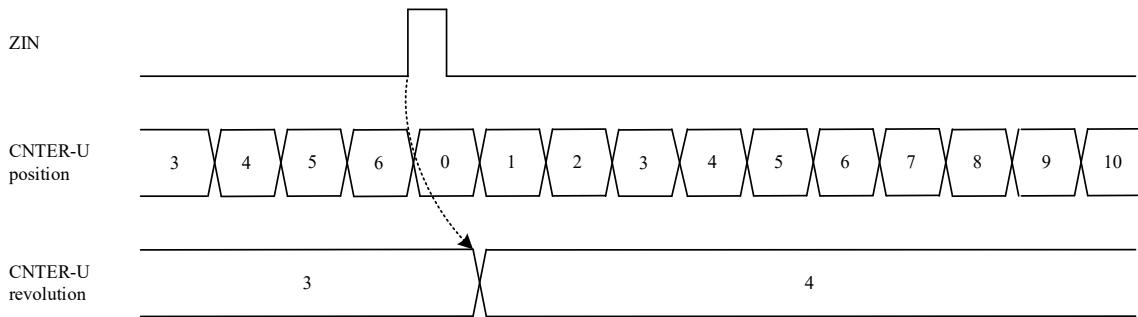


Figure 20-30 Revolution counting mode-Z phase counting

Position overflow/underflow count

Position overflow/underflow counting means that when the position counting unit count overflows or underflows, an overflow/underflow event is generated, which triggers the timer of the revolution counting unit to count once (in this counting mode, the input of ZIN does not perform the counting action of the revolution counting unit and the clearing action of the position counting unit).

The overflow/underflow event of the position counting unit can realize the counting of the revolution counting unit by selecting the internal trigger event interface, and the position overflow counting can be realized. Select count up (count down) event for the revolution counting unit by setting 1 bit of Bit8~Bit11 in hardware register (HCUPR or HCDOR), and set the event number in corresponding Hardware Trigger Event Select Register (HTSSR0~3) to the overflow or underflow event of the position counting unit. Refer to the INTC chapter for detailed event numbers. As Figure 20-31 shown.

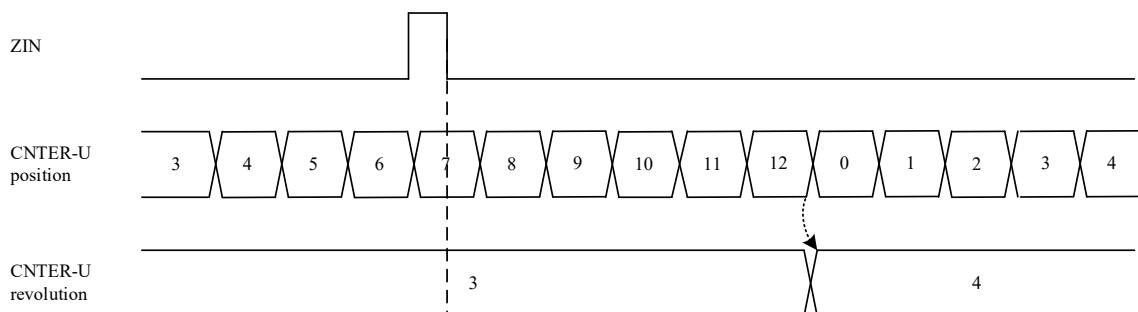


Figure 20-31 Revolution counting mode-position overflow counting

Mixed count

Mixed counting refers to the counting action in which the above two counting methods of Z-phase counting and position overflow/underflow counting are combined, and the realization method is also a combination of the above two counting methods. As Figure 20-32 shown.

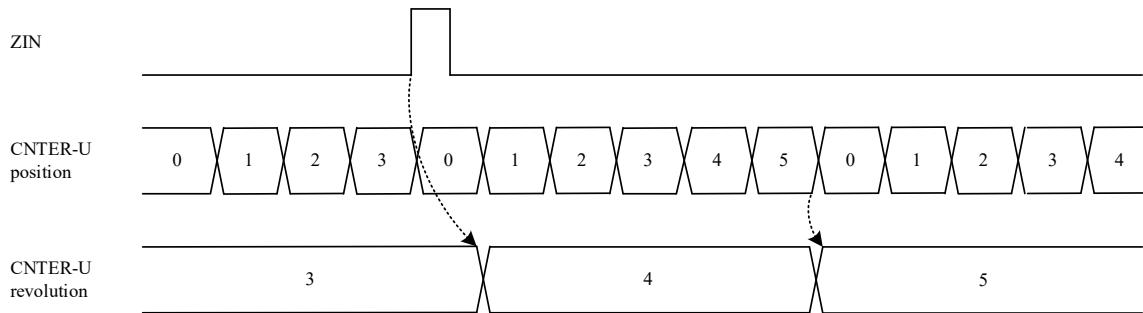


Figure 20-32 Revolution counting mode-mixed counting

20.3.15.3 Z-phase action mask

In the Z-phase counting function or mixed counting function of revolution counting mode, the valid input of ZIN can be masked it within a few periods after the overflow or underflow of the position timer (GCONR.ZMSKVAL[0:1]), and the counting of the revolution counting unit and the clearing of the position counting unit are not performed.

When GCONR.ZMSKPOS of the General Control Register (GCONR) of the position counting unit is 1, the Z-phase mask function of the position counting unit is enabled, and the number of periods of the Z-phase mask is set by GCONR.ZMSKVAL; When the revolution counting unit of the General Control Register (When GCONR.ZMSKREV of GCONR) is 1, the Z-phase mask function of the revolution counting unit is enabled.

Figure 20-33 shows the action in mixed counting in revolution counting mode, when there is a ZIN phase input within 4 count periods after the position counting unit count overflows, the action of the ZIN phase input is invalid, that is, the revolution counting unit does not count, and the position counting unit is not cleared; the next ZIN phase input operates normally.

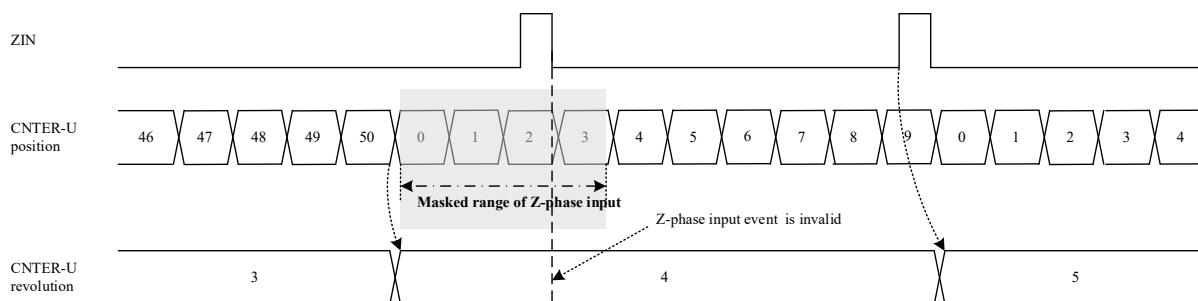


Figure 20-33 Revolution counting mode-mixed counting Z phase mask operation example 1

Figure 20-34 shows the action in mixed counting in revolution counting mode, at the 3rd period after the position counting unit count overflows, the counting direction changes, the set 4-period mask period becomes invalid at this time (actually ZIN phase mask function maintains 3 periods), and counter starts down-counting. After the count underflow occurs in the position counting unit, the ZIN phase mask function is re-opened and becomes invalid after 4 periods. During the ZIN

phase mask periods, the input function of the ZIN phase is invalid, that is, the revolution counting unit does not count, and the position counting unit is not cleared; the next ZIN phase input operates normally.

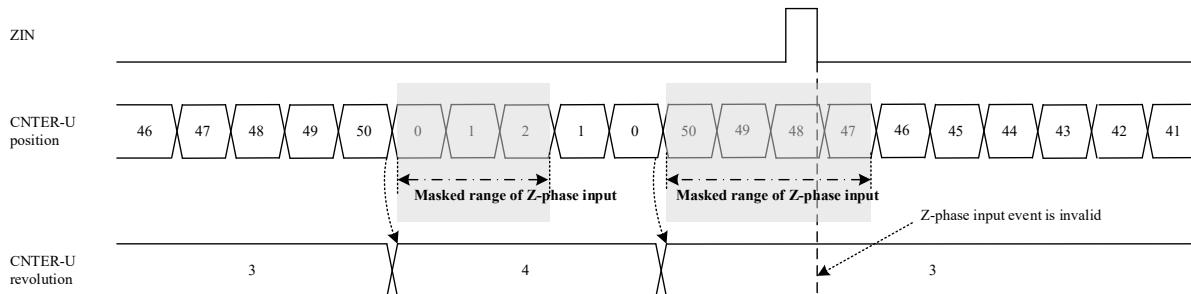


Figure 20-34 Revolution counting mode-mixed counting Z phase mask operation example 2

20.3.16 EMB control

Timer6 can protect and control the output state of the port, and fix the port state to a preset safe state when an abnormality occurs. All units have 4 common port output control interfaces. Each unit selects the EMB event connection interface to be used through the setting of the Port Control Register (PCNAR.EMBSA). This interface is connected to one group of EMB events from the output of the EMB module. At the same time, the abnormal events on the interface can be set from the EMB side (refer to the EMB chapter). When abnormal conditions are detected on these interfaces, the general PWM output can be controlled.

During the normal output period of the port, if the EMB event from the EMB is detected, the output state of the port can be changed to a preset state. When the EMB abnormal event occurs, the general PWM output port can change to output high-impedance state, output low level or output high level (depending on the setting of PCNAR.EMBCA). For example, PCNAR.EMBCA=01 is set, if an EMB event occurs during the normal output period of the TIM6_<t>_PWMA port, the output on the TIM6_<t>_PWMA port becomes a high-impedance state.

After the EMB event is invalid (the abnormal event connected to Timer6 from the EMB module disappears, and the signal becomes the normal level), the output of the PWM port can automatically return to the normal output. At this time, by the setting of the Port Control Register, the output of PWM can return to normal output immediately (PCNAR.EMBRA=00, this method is called One Shot method release); or return to normal output when the count reaches the next overflow point (PCNAR.EMBRA=01, 10, 11, this method is called Cycle By Cycle method release).

20.3.17 Typical application example

The following describes the basic settings of Timer6 related registers in several typical application situations for users' reference.

20.3.17.1 Basic count and interrupt operation

1. Set the period value (PERAR)
2. Set the required compare values, including general compare values (GCMAR~GCMFR), special compare values (SCMAR~SCMBR), etc.
3. Set the required interrupt enable bits, including count overflow interrupt (ICONR.INTENOVF), count underflow interrupt (ICONR.INTENUDF), count match interrupt (ICONR.INTENA~F, ICONR.INTENSAU, ICONR.INTENSAD, ICONR.INTENSBU, ICONR.INTENSBD), etc.
4. Set the internal counting clock frequency division (GCONR.CKDIV[3:0])
5. Set the waveform mode (GCONR.MODE)
6. Set the counting direction (only need to be set when in the sawtooth waveform mode GCONR.MODE=0)
7. Start the counter (GCONR.START=1)

20.3.17.2 Comparison output and interrupt operation

1. Set the period value (PERAR)
2. Set the compare value of each channel, including the general compare value A (GCMAR) and the general compare value B (GCMBR)
3. Set the required interrupt enable bits, including count overflow interrupt (ICONR.INTENOVF), count underflow interrupt (ICONR.INTENUDF), count match interrupt (ICONR.INTENA~B), etc.
4. Set the port output state of each channel in different count states (refer to the related control of bit17~bit0 of PCNAR or PCNBR)
5. Set the internal counting clock frequency division (GCONR.CKDIV[3:0])
6. Set the waveform mode (GCONR.MODE)
7. Set the counting direction (only need to be set when in the sawtooth waveform mode GCONR.MODE=0)
8. Set the comparison output mode of each channel (PCNAR.CAPMDA=0, PCNBR.CAPMDB=0)
9. Enable the output of each channel (PCNAR.OUTENA=1, PCNBR.OUTENB=1)
10. Start the counter (GCONR.START=1)

20.3.17.3 Input capture and interrupt action

1. Set the period value (PERAR)
2. Set the required interrupt enable bits, including count overflow interrupt

(ICONR.INTENOVF), count underflow interrupt (ICONR.INTENUDF), input capture interrupt (ICONR.INTENA~B), etc.

3. Set the input capture external condition of each channel (refer to all valid control bits of HCPAR or HCPBR. The valid control bits are independent of each other, and multiple conditions can be selected as the input capture condition of a channel at the same time)
4. Set the internal counting clock frequency division (GCONR.CKDIV[3:0])
5. Set the waveform mode (GCONR.MODE)
6. Set the counting direction (only need to be set when in the sawtooth waveform mode GCONR.MODE=0)
7. Set input capture mode (PCNAR.CAPMDA=1, PCNBR.CAPMDB=1)
8. Start the counter (GCONR.START=1)
9. Wait for the input capture condition to be generated, read the input capture value (GCMAR or GCMBR) of the corresponding channel or wait for the corresponding interrupt to be generated

20.3.17.4 Buffer transfer operation (period value)

1. Set the desired period value (PERAR, PERBR, PERCR)
2. Set single and double buffer transfer mode (BCONR.BSEP)
3. Set the buffer transfer time point (BCONR.BTRUP, BCONR.BTRDP, these two control bits are independent of each other and can be selected at the same time, both as the buffer transfer time point) (this setting is only valid in triangular waveform mode, and invalid in sawtooth waveform mode)
4. Set the internal counting clock frequency division (GCONR.CKDIV[3:0])
5. Set the waveform mode (GCONR.MODE)
6. Set the counting direction (only need to be set when in the sawtooth waveform mode GCONR.MODE=0)
7. Enable the buffer function (BCONR.BENP=1)
8. Start the counter (GCONR.START=1)
9. Wait for the corresponding buffer transfer time point, the buffer action occurs (PERBR->PERAR (when BCONR.BSEP=0), PERCR->PERBR->PERAR (when BCONR.BSEP=1))

20.3.17.5 Buffer transfer action (general compare value)

1. Set the desired compare values (GCMAR, GCMCR, GCMER, GCMBR, GCMDR, GCMFR)
2. Set the single and double buffer transfer mode of each channel (BCONR.BSEA, BCONR.BSEB)
3. Set the buffer transfer time point of each channel (BCONR.BTRUA, BCONR.BTRDA, BCONR.BTRUB, BCONR.BTRDB, the two control bits of each channel are independent of each other and can be selected at the same time as the buffer transfer time point) (this setting is only valid in triangular waveform mode, and invalid in sawtooth waveform mode)

4. Set the internal counting clock frequency division (GCONR.CKDIV[3:0])
5. Set the waveform mode (GCONR.MODE)
6. Set the counting direction (only need to be set when in the sawtooth waveform mode GCONR.MODE=0)
7. Enable the buffer function of each channel (BCONR.BENA=1, BCONR.BENB=1)
8. Start the counter (GCONR.START=1)
9. Wait for the corresponding buffer transfer time point set by each channel, and buffer action occurs (GCMCR->GCMAR (when BCONR.BSEA=0), GCMER->GCMCR->GCMAR (when BCONR.BSEA=1), GCMDR->GCMBR (when BCONR.BSEB=0), GCMFR->GCMDR->GCMBR (when BCONR.BSEB=1))

20.3.17.6 Buffer transfer operation (special compare value)

1. Set the desired special compare value (SCMAR, SCMCR, SCMER, SCMBR, SCMDR, SCMFR)
2. Set the single and double buffer transfer mode of each channel (BCONR.BSESPA, BCONR.BSESPB)
3. Set the buffer transfer time point of each channel (BCONR.BTRUSPA, BCONR.BTRDSPA, BCONR.BTRUSPB, BCONR.BTRDSPB, the two control bits of each channel are independent of each other and can be selected at the same time as the buffer transfer time point) (this setting is only valid in triangular waveform mode, and invalid in sawtooth waveform mode)
4. Set the internal counting clock frequency division (GCONR.CKDIV[3:0])
5. Set the waveform mode (GCONR.MODE)
6. Set the counting direction (only need to be set when in the sawtooth waveform mode GCONR.MODE=0)
7. Enable the buffer function of each channel (BCONR.BENSPA=1, BCONR.BENSPB=1)
8. Start the counter (GCONR.START=1)
9. Wait for the corresponding buffer transfer time point set by each channel, and buffer action occurs (SCMCR->SCMAR (when BCONR.BSESPA=0), SCMER->SCMCR->SCMAR (when BCONR.BSESPA=1), SCMDR->SCMBR (when BCONR.BSESPB=0), SCMFR->SCMDR->SCMBR (when BCONR.BSESPB=1))

20.3.17.7 Buffer transfer operation (dead time value)

1. Set the desired dead time value (DTUAR, DTUBR, DTDAR, DTDBR)
2. Set the buffer transfer time point (DCONR.DTBTRU, DCONR.DTBTRD, these two control bits are independent of each other and can be selected at the same time, both as the buffer transfer time point) (this setting is only valid in the triangular waveform mode, and invalid in the sawtooth waveform mode)
3. Set the internal counting clock frequency division (GCONR.CKDIV[3:0])
4. Set the waveform mode (GCONR.MODE)
5. Set the counting direction (only need to be set when in the sawtooth waveform mode)

- GCONR.MODE=0)
6. Enable the buffer function (DCONR.DTBENU=1, DCONR.DTBEND=1)
 7. Enable the hardware dead time function (DCONR.DTCEN=1)
 8. Start the counter (GCONR.START=1)
 9. Wait for the corresponding buffer transfer time point, the buffer action occurs (DTUBR->DTUAR, DTDBR->DTDAR)

20.3.17.8 Synchronous start action (software method)

1. Refer to steps a) ~ f) of the chapter [Basic count and interrupt operation] to set each unit that needs to be started synchronously
2. Start the counter synchronously (set the corresponding bit of the SSTAR register to 1, each unit corresponds to a register bit)

20.3.17.9 Synchronous start action (hardware method)

1. Set the period value (PERAR)
2. Set the required compare values, including general compare values (GCMAR~GCMFR), special compare values (SCMAR~SCMBR), etc.
3. Set the required interrupt enable bits, including count overflow interrupt (ICONR.INTENOVF), count underflow interrupt (ICONR.INTENUDF), count match interrupt (ICONR.INTENA~F, ICONR.INTENSAU, ICONR.INTENSAD, ICONR. INTENSBU, ICONR.INTENSBD), etc.
4. Set hardware start conditions (selected by HSTAR.HSTAx, x=8~31)
5. Enable hardware start (HSTAR.STAS=1)
6. Repeat the above steps a) ~ e) to set each unit that needs to be started synchronously (in each unit that needs to be started synchronously, the setting of step d) should be consistent)
7. Wait for the set trigger event to be generated, and confirm that the counters of each unit are started synchronously

20.3.17.10 Quadrature encoding counting action (2-phase)

1. Set the period value (PERAR)
2. Set the required compare values, including general compare values (GCMAR~GCMFR), special compare values (SCMAR~SCMBR), etc.
3. Set the required interrupt enable bits, including count overflow interrupt (ICONR.INTENOVF), count underflow interrupt (ICONR.INTENUDF), count match interrupt (ICONR.INTENA~F, ICONR.INTENSAU, ICONR.INTENSAD, ICONR. INTENSBU, ICONR.INTENSBD), etc.
4. Set the required hardware count up condition (selected by HCUPR.HCUPx, x=0~7)
5. Set the required hardware count down conditions (selected by HCDOR.HCDOx, x=0~7)

6. Start the counter (GCONR.START=1)
7. Wait for the set quadrature encoding count event to be generated, and confirm that the counter counts normally

20.3.17.11 Quadrature encoding counting action (3-phase)

1. Refer to steps a) ~ e) in the chapter [Quadrature encoding counting action (2-phase)] to set the position counting unit
2. Set the hardware clearing condition of the position counting unit (selected by HCLRR.HCLR x , $x=16\sim31$)
3. Enable position counting unit hardware clearing (HCLRR.CLES=1)
4. Set the period value (PERAR) of the revolution counting unit
5. Set the compare value of revolution counting unit, including general compare value (GCMAR~GCMFR), special compare value (SCMAR~SCMBR), etc.
6. Set the interrupt enable bits required by the revolution counting unit, including count overflow interrupt (ICONR.INTENOVF), count underflow interrupt (ICONR.INTENUDF), count match interrupt (ICONR.INTENA~F, ICONR.INTENSAU, ICONR.INTENSAD), ICONR.INTENSBU, ICONR.INTENSBD), etc.
7. Set the hardware count up condition 1 of the revolution counting unit (ZIN phase input) (selected by HCUPR.HCUP x , $x=16\sim31$, the set event here should be consistent with the event set by the position counting unit in step b))
8. Set the hardware count up condition 2 of the revolution counting unit (the overflow event input of the position counting unit) (select the internal hardware trigger event 0 through HCUPR.HCUP8)
9. Set the hardware count down condition of revolution counting unit (underflow event input of position counting unit) (select internal hardware trigger event 1 through HCDOR.HCD09)
10. Set the trigger source number in HTSSR0 as the count overflow event of the position counting unit (the overflow event number refers to the INTC chapter)
11. Set the trigger source number in HTSSR1 as the count underflow event of the position counting unit (the underflow event number refers to the INTC chapter)
12. Start counter of revolution counting unit (GCONR.START=1)
13. Start counter of position counting unit (GCONR.START=1)
14. Wait for the set AIN, BIN, ZIN phase count events to be generated, and confirm that the counter counts normally

20.3.17.12 Single PWM output

1. Refer to the setting of steps a) ~ j) in the chapter [Comparison output and interrupt operation] (the output states of the two PWM channels TIM6_<t>_PWMA and TIM6_<t>_PWMB in each unit can be set independently, forming two mutual unrelated single PWM output)

20.3.17.13 Complementary PWM outputs (Software Dead Time)

1. Set the periodic value (PERAR)
2. Set the general compare value A (GCMAR) and the general compare value B (GCMBR)
3. Set the required interrupt enable bits, including count overflow interrupt (ICONR.INTENOVF), count underflow interrupt (ICONR.INTENUDF), count match interrupt (ICONR.INTENA~B), etc.
4. Set the port output state in different count states (refer to the related control of bit17~bit0 of PCNAR and PCNBR, combined with the setting values of GCMAR and GCMBR. It is necessary to ensure that the complementary dead time is formed between the two PWM outputs)
5. Set the internal counting clock frequency division (GCONR.CKDIV[3:0])
6. Set the waveform mode to triangular waveform mode (GCONR.MODE=1)
7. Set the comparison output mode (PCNAR.CAPMDA=0, PCNBR.CAPMDB=0)
8. Enable output (PCNAR.OUTENA=1, PCNBR.OUTENB=1)
9. Start the counter (GCONR.START=1)

20.3.17.14 Complementary PWM output (Hardware Dead Time)

1. Set the period value (PERAR)
2. Set general compare value A (GCMAR), dead time value (DTUAR, DTDAR)
3. Set the required interrupt enable bits, including count overflow interrupt (ICONR.INTENOVF), count underflow interrupt (ICONR.INTENUDF), count match interrupt (ICONR.INTENA~B), dead time error interrupt (ICONR.INTENDTE) Wait
4. Set the port output state in different count states (refer to the related control of bit17~bit0 of PCNAR and PCNBR, combined with the setting values of GCMAR, DTUAR and DTDAR. It is necessary to ensure that the complementary dead time is formed between the two PWM outputs)
5. Set the internal counting clock frequency division (GCONR.CKDIV[3:0])
6. Set the waveform mode to triangular waveform mode (GCONR.MODE=1)
7. Set the comparison output mode of each channel (PCNAR.CAPMDA=0, PCNBR.CAPMDB=0)
8. Enable the output of each channel (PCNAR.OUTENA=1, PCNBR.OUTENB=1)
9. Enable the hardware dead time function (DCONR.DTCEN=1)
10. Start the counter (GCONR.START=1)

20.3.17.15 EMB monitoring and interruption action

1. Refer to steps a) ~ h) in the [Complementary PWM output (Software Dead Time)] chapter or steps a) ~ i) in the [Complementary PWM output (Hardware Dead Time)] chapter to set the complementary PWM output action.
2. Set the state of the PWM port (PCNAR.EMBCA, PCNBR.EMBCB) when the EMB event occurs

(select the corresponding protection state according to the system application)

3. Set the time point at which the PWM port resumes normal output when the EMB event becomes invalid (PCNAR.EMBRA, PCNBR.EMBRB)
4. Set the EMB event source (PCNAR.EMBSA, PCNBR.EMBSB) input from the EMB module
5. Set relevant registers of EMB module (including EMB interrupt enable register (EMB_INTEN0~3), EMB control register 1/2 (EMB_CTL1/2_0~3), EMB release select register (EMB_RLSSEL0~3), etc.)
6. Start the counter (GCONR.START=1), the EMB module monitors the system status in real time

20.3.18 Function Summary

In the sawtooth waveform mode and triangular waveform mode of Timer6, the summary table of main functions is as Table 20-3 shown.

Table 20-3 Comparison table of functions in different modes

PWM output function			Sawtooth waveform	Triangular waveform	Related main registers
Independent PWM output	Port Status Control	At start	Support	Support	PCNAR.STACA
		When stopped	Support	Support	PCNAR.STPCA
		When overflow	Support	Support	PCNAR.OVFCA
		When underflow	Support	Support	PCNAR.UDFCA
		When the count matches (Up Counting)	Support	Support	PCNAR.CMAUCA
		When the count matches (Down Counting)	Support	Support	PCNAR.CMADCA
	Buffer transfer	Period value	Single buffer	Support	PERBR->PERAR
			Double buffer	Support	PERCR->PERBR PERBR->PERAR
		Compare value	Single buffer	Support	GCMDR->GCMBR GCMCR->GCMAR
			Double buffer	Support	GCMFR->GCMDR GCMER->GCMCR GCMDR->GCMBR GCMCR->GCMAR
	EMB			Support	PCNAR.EMBCA
Complementary PWM output	Port Status Control	At start	Support	Support	PCNAR.STACA
		When stopped	Support	Support	PCNAR.STPCA
		When overflow	Support	Support	PCNAR.OVFCA
		When underflow	Support	Support	PCNAR.UDFCA
		When the count matches (Up Counting)	Support	Support	PCNAR.CMAUCA
		When the count matches (Down Counting)	Support	Support	PCNAR.CMADCA
	Buffer	Period	Single	Support	PERBR->PERAR

PWM output function				Sawtooth waveform	Triangular waveform	Related main registers
transfer	value	buffer				
		Double buffer	Support	Support	PERCR->PERBR PERBR->PERAR	
	Compare value	Single buffer	Support	Support	GCMDR->GCMBR GCMCR->GCMAR	
		Double buffer	Support	Support	GCMFR->GCMDR GCMER->GCMCR GCMDR->GCMBR GCMCR->GCMAR	
	Dead time value	Single buffer	Support	Support	DTUBR->DTUAR DTDBR->DTDAR	
	No dead time PWM output		Support	Support	GCMAR=GCMBR	
	PWM output with dead time	Software way	Support	Support	GCMAR≠GCMBR	
		Hardware way	Not support	Support	GCMBR=GCMAR-DTUAR GCNBR=GCMAR-DTDAR	
	EMB		Support	Support	PCNAR.EMBCA	

20.4 Interrupt and Event Description

20.4.1 Interrupt output

Timer6 includes 6 general count compare match interrupts (including 2 input capture interrupts), 2 special count compare match interrupts, 2 count period match interrupts, and 1 dead time error interrupt.

20.4.1.1 Count compare match interrupt

There are 6 general compare value registers (GCMAR~GCMFR), which can be compared with the count value to generate a comparison match. When the count comparison matches, the STFLR.CMAF~STFLR.CMFF bits in the Status Register (STFLR) will be set to 1 respectively. At this time, if the corresponding bit in INTENA~INTENF of the Interrupt Control Register (ICONR) is set to 1 to enable the interrupt, the corresponding interrupt request (TMR6_<t>_GCMA~F) will also be triggered.

The input capture action occurs when the input capture valid condition selected by the Hardware Input Capture Event Select Register (HCPAR, HCPBR) occurs. At this time, if the INTENA or INTENB bit of the Interrupt Control Register (ICONR) is set to 1 to enable the interrupt, the corresponding interrupt request (TMR6_<t>_GCMA~B) is triggered.

Two Special Compare Value Registers (SCMAR~SCMBR) can also be compared with the count value to generate a comparison match. When the count comparison matches, the STFLR.CMSPAF~CMSPBF bits in the Status Register (STFLR) will be set to 1 respectively. At this time, 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_<t>_SCMA~B) will also be triggered.

20.4.1.2 Count period match interrupt

When the sawtooth waveform counts up to the overflow point, the sawtooth waveform counts down to the underflow point, the triangular waveform counts to the valley point or the triangular waveform counts to the peak point, the STFLR.OVFF or STFLR.UDFF bit in the Status Register (STFLR) will be set to 1. At this time, if the ICONR.INTENOVF or ICONR.INTENUDF bit in the Interrupt Control Register (ICONR) is set to enable the interrupt, the count period match interrupt (TMR6_<t>_GOVF and TMR6_<t>_GUDF) can be triggered at the corresponding time point.

20.4.1.3 Dead time error interrupt

In the function of PWM output with dead time in triangular waveform mode, the General Compare Value Register (GCMBR) is loaded by the value calculated on the following formula:

- a) GCMBR=GCMAR-DTUAR (when up counting)
- b) GCMBR=GCMAR-DTDAR (when down counting)

If the calculated value exceeds period limit, a dead time error will be generated, STFLR.DTEF bit in the Status Register (STFLR) will be set to 1. At this time, if the INTENDTE bit in the Interrupt Control Register (ICONR) is set to enable the interrupt, the dead time error interrupt (TMR6_<t>_GDTE) will be triggered at this moment.

20.4.2 Event output

During the clock counting process, if a period match event (overflow point or underflow point of sawtooth waveform, count peak or valley point of triangular waveform), general count comparison match event and special count comparison match event occur, corresponding event output signals will be generated. The event output signal is used to trigger other modules if it is selected, such as ADC, DMA, etc.

The following figure is an example of the operation of the general comparison match interrupt A~F && events A~F, the special comparison match interrupt A~B && events A~B, and the period match interrupt && events of unit 1.

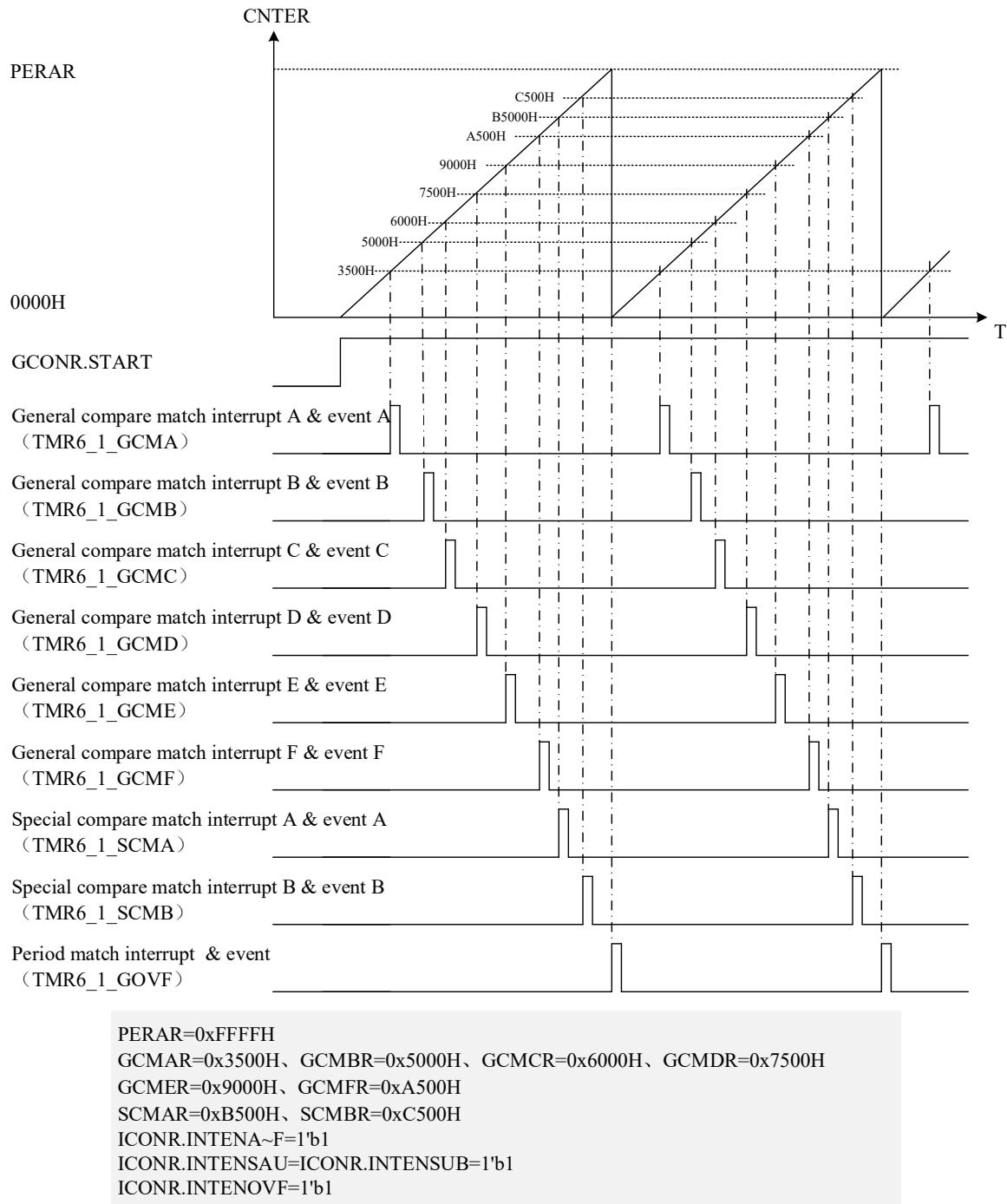


Figure 20-35 Example of interrupt & event output in sawtooth waveform mode

20.5 Register description

Table 20-4 is the register list of the Timer6 module.

BASE ADDR:

0x40018000 (U1), 0x40018400 (U2), 0x40018800 (U3), 0x40018C00 (U4)

0x40019000 (U5), 0x40019400 (U6), 0x40019800 (U7), 0x40019C00 (U8)

Table 20-4 Timer6 register list

Register name	Symbol	Offset	Bit width	Reset value
Counter Value Register	TMR6_CNTER	0x0000h	32	0x00000000h
Update Value Register	TMR6_UPDAR	0x0004h	32	0x00000000h
Period Value A Register	TMR6_PERAR	0x0040h	32	0x0000FFFFh
Period Value B Register	TMR6_PERBR	0x0044h	32	0x0000FFFFh
Period Value C Register	TMR6_PERCR	0x0048h	32	0x0000FFFFh
General Compare Value A Register	TMR6_GCMAR	0x0080h	32	0x0000FFFFh
General Compare Value B Register	TMR6_GCMBR	0x0084h	32	0x0000FFFFh
General Compare Value C Register	TMR6_GCMCR	0x0088h	32	0x0000FFFFh
General Compare Value D Register	TMR6_GCMDR	0x008Ch	32	0x0000FFFFh
General Compare Value E Register	TMR6_GCMER	0x0090h	32	0x0000FFFFh
General Compare Value F Register	TMR6_GCMFR	0x0094h	32	0x0000FFFFh
Special Compare Value A Register	TMR6_SCMAR	0x00C0h	32	0x0000FFFFh
Special Compare Value B Register	TMR6_SCMBR	0x00C4h	32	0x0000FFFFh
Special Compare Value C Register	TMR6_SCMCR	0x00C8h	32	0x0000FFFFh
Special Compare Value D Register	TMR6_SCMDR	0x00CCh	32	0x0000FFFFh
Special Compare Value E Register	TMR6_SCMER	0x00D0h	32	0x0000FFFFh
Special Compare Value F Register	TMR6_SCMFR	0x00D4h	32	0x0000FFFFh
DeadTime Value Up A Register	TMR6_DTUAR	0x0100h	32	0x0000FFFFh
DeadTime Value Down A Register	TMR6_DTDAR	0x0104h	32	0x0000FFFFh
DeadTime Value Up B Register	TMR6_DTUBR	0x0108h	32	0x0000FFFFh
DeadTime Value Down B Register	TMR6_DTDDBR	0x010Ch	32	0x0000FFFFh
General Control Register	TMR6_GCONR	0x0140h	32	0x00000002h
Interrupt Control Register	TMR6_ICONR	0x0144h	32	0x00000000h
Buffer Control Register	TMR6_BCONR	0x0148h	32	0x00000000h
DeadTime Control Register	TMR6_DCONR	0x014Ch	32	0x00000000h
Port Control A Register	TMR6_PCNAR	0x0154h	32	0x00000000h
Port Control B Register	TMR6_PCNBR	0x0158h	32	0x00000000h
Filter Control General Port Register	TMR6_FCNGR	0x015Ch	32	0x00000000h
Filter Control Trigger Port Register	TMR6_FCNTR	U1_base+0x03ECh	32	0x00000000h

Register name	Symbol	Offset	Bit width	Reset value
Valid Period Register	TMR6_VPERR	0x0160h	32	0x00000000h
Status Register	TMR6_STFLR	0x0164h	32	0x80000000h
Hardware Start Event Select Register	TMR6_HSTAR	0x0180h	32	0x00000000h
Hardware Stop Event Select Register	TMR6_HSTPR	0x0184h	32	0x00000000h
Hardware Clear Event Select Register	TMR6_HCLRR	0x0188h	32	0x00000000h
Hardware Update Event Select Register	TMR6_HUPDR	0x018Ch	32	0x00000000h
Hardware Input Capture A Event Select Register	TMR6_HCPAR	0x0190h	32	0x00000000h
Hardware Input Capture B Event Select Register	TMR6_HCPBR	0x0194h	32	0x00000000h
Hardware Count Up Event Select Register	TMR6_HCUPR	0x0198h	32	0x00000000h
Hardware Count Down Event Select Register	TMR6_HCDOR	0x019Ch	32	0x00000000h
Hardware Trigger Event Select Register 0	TMR6_HTSSR0	(0x40010858h)	32	0x000001FFh
Hardware Trigger Event Select Register 1	TMR6_HTSSR1	(0x4001085Ch)	32	0x000001FFh
Hardware Trigger Event Select Register 2	TMR6_HTSSR2	(0x40010860h)	32	0x000001FFh
Hardware Trigger Event Select Register 3	TMR6_HTSSR3	(0x40010864h)	32	0x000001FFh
Software Sync Start Control Register	TMR6_SSTAR	U1_base+0x03F0h	32	0x00000000h
Software Sync Stop Control Register	TMR6_SSTPR	U1_base+0x03F4h	32	0x00000000h
Software Sync Clear Control Register	TMR6_SCLRR	U1_base+0x03F8h	32	0x00000000h
Software Sync Update Control Register	TMR6_SUPDR	U1_base+0x03FCCh	32	0x00000000h

Note:

- The bit widths of the counter value register, general update value register and each value register of U1~U4 units are all 32 bits, and reset value of these registers is different from U5~U8 units. Among them, the reset value of the counter value register and the general update value register is 0x00000000h; the reset value of each value register is 0xFFFFFFFFh.
- Trigger event select registers (TMR6_HTSSR0~3) are 4 unit-independent registers, which are shared by 8 units of Timer6.
- The software sync registers (TMR6_SSTAR, TMR6_SSTPR, TMR6_SCLRR, TMR6_SUPDR) are 4 unit-independent registers, which are shared by the 8 units of Timer6.
- The filter control trigger port register (TMR6_FCNTR) is a unit-independent register that is shared by 8 units of Timer6.

20.5.1 Counter Value Register (TMR6_CNTER)

Reset value: 0x00000000h

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]															

Bit	Symbol	Bit name	Description	Read and write
b31~b16	Reserved	-	Read as "0"	R
b15~b0	CNT[15:0]	Count value	Current count value of timer	R/W

20.5.2 Update Value Register (TMR6_UPDAR)

Reset value: 0x00000000h

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
UPDA[15:0]															

Bit	Symbol	Bit name	Description	Read and write
b31~b16	Reserved	-	Read as "0"	R
b15~b0	UPDA[15:0]	Update value	Set the update value to be updated into the timer	R/W

20.5.3 Period Value Register (TMR6_PERmR) (m=A~C)

Reset value: 0x0000FFFFh

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]															

Bit	Symbol	Bit name	Description	Read and write
b31~b16	Reserved	-	Read as "0"	R
b15~b0	PERA-C[15:0]	Count period value	Set the count period value and the corresponding buffer value of each round of counting	R/W

20.5.4 General Compare Value Register (TMR6_GCMmR) (m=A~F)

Reset value: 0x0000FFFFh

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]															

Bit	Symbol	Bit name	Description	Read and write
b31~b16	Reserved	-	Read as "0"	R
b15~b0	GCMA-F[15:0]	Count compare reference value	Set the compare reference value. The matching signal is valid when compare reference value is equal to count value	R/W

20.5.5 Special Compare Value Register (TMR6_SCmR) (m=A~F)

Reset value: 0x0000FFFFh

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]															
<hr/>															
Bit	Symbol	Bit name			Description						Read and write				
b31~b16	Reserved	-			Read as "0"						R				
b15~b0	SCMA-F[15:0]	Special Compare Reference Value			Set the compare reference value and buffer value						R/W				

20.5.6 DeadTime Value Register (TMR6_DTmnR) (m=D, U&&n=A, B)

Reset value: 0x0000FFFFh

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]															
<hr/>															
Bit	Symbol	Bit name			Description						Read and write				
b31~b16	Reserved	-			Read as "0"						R				
b15~b0	DTU/DA-B[15:0]	Dead Time Value			Set dead time value and buffer value						R/W				

20.5.7 General Control Register (TMR6_GCONR)

Reset value: 0x00000002h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										ZMSK VAL[1:0]	ZMSK POS	ZMSK REV			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved										OV STP	CKDIV[3:0]	-	MODE	DIR	START
<hr/>															
Bit	Symbol	Bit name		Description										Read and write	
b31~b20	Reserved	-		Read as "0", write as "0"										R/W	
b19~b18	ZMSKVAL[1:0]	Masked period number of Z-phase input		Masked count period number of Quadrature-encoded Z-phase input 00: Z-phase input masking function is disabled 01: Z-phase input is masked within 4 count periods after position count overflow or underflow 10: Z-phase input is masked within 8 count periods after position count overflow or underflow 11: Phase Z input is masked within 16 count periods after position count overflow or underflow										R/W	
b17	ZMSKPOS	Position timer selection of Z-phase input		0: The unit acts as a position timer when the Z-phase is input, and the position timer clear function operates normally during the masked periods 1: The unit acts as a position timer when Z-phase is input, and the position timer clear function is invalid during the masked periods										R/W	
b16	ZMSKREV	Revolution timer selection of Z-phase input		0: The unit acts as a revolution timer when the Z-phase is input, and the revolution timer count function operates normally during the mask period. 1: The unit acts as a revolution timer when the Z-phase is input, and the revolution timer count function is invalid during the masked periods										R/W	
b15~b9	Reserved	-		Read as "0", write as "0"										R/W	
b8	OVSTP	Counting stop control of overflow or underflow		0: Counter continues counting after overflow or underflow 1: Counter stops counting after overflow or underflow										R/W	
b7~b4	CKDIV[2:0]	Counting clock selection		0000: PCLK0 0001: PCLK0/2 0010: PCLK0/4 0011: PCLK0/8 0100: PCLK0/16 0101: PCLK0/32 0110: PCLK0/64 0111: PCLK0/128 1000: PCLK0/256 1001: PCLK0/512 1010: PCLK0/1024										R/W	
b3	Reserved	-		Read as "0", write as "0"										R/W	
b2	MODE	Counting mode		0: Sawtooth waveform mode 1: Triangular waveform mode										R/W	
b1	DIR	Counting direction		0: count down 1: count up										R/W	
b0	START	Timer start		0: Timer off 1: Timer start Note: This bit automatically becomes 0 when a software stop condition or hardware stop condition is valid										R/W	

20.5.8 Interrupt Control Register (TMR6_ICONR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved												INTEN SBD	INTEN SBU	INTEN SAD	INTEN SAU
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved						INTEN DTE	INTEN UDF	INTEN OVF	INTEN F	INTEN E	INTEN D	INTEN C	INTEN B	INTEN A	
Bit	Symbol	Bit name	Description										Read and write		
b31~b20	Reserved	-	Read as "0", write as "0"										R/W		
b19	INTENSBD	Specail count-down interrupt enable B	0: During down counting, when the SCMBR register and the count value are equal, the interrupt is disabled. 1: During down counting, when the SCMBR register and the count value are equal, the interrupt is enabled										R/W		
b18	INTENSBU	Specail count-up interrupt enable B	0: During up counting, when the SCMBR register and the count value are equal, the interrupt is disabled 1: During up counting, when the SCMBR register and the count value are equal, the interrupt is enabled										R/W		
b17	INTENSAD	Specail count-down interrupt enable A	0: During down counting, when the SCMAR register and the count value are equal, the interrupt is disabled 1: During down counting, when the SCMAR register and the count value are equal, the interrupt is enabled										R/W		
b16	INTENSAU	Specail count-up interrupt enable A	0: During up counting, when the SCMAR register and the count value are equal, the interrupt is disabled 1: During up counting, when the SCMAR register and the count value are equal, the interrupt is enabled										R/W		
b15~b9	Reserved	-	Read as "0", write as "0"										R/W		
b8	INTENDTE	Dead Time Error Interrupt Enable	0: When the dead time error happens, the interrupt is disabled 1: When the dead time error happens, the interrupt is enabled										R/W		
B7	INTENUDF	Underflow interrupt enable	0: When the count underflows, the interrupt is disabled 1: When the count underflows, the interrupt is enabled										R/W		
b6	INTENOVF	Overflow interrupt enable	0: When the counter overflows, the interrupt is disabled 1: When the counter overflows, the interrupt is enabled										R/W		
b5	INTENF	Count match interrupt enable F	0: When the GCMFR register is equal to the count value, the interrupt is disabled 1: When the GCMFR register is equal to the count value, the interrupt is enabled										R/W		
b4	INTENE	Count match interrupt enable E	0: When the GCMER register is equal to the count value, the interrupt is disabled 1: When the GCMER register is equal to the count value, the interrupt is enabled										R/W		
b3	INTEND	Count match interrupt enable D	0: When the GCMDR register is equal to the count value, the interrupt is disabled 1: When the GCMDR register is equal to the count value, the interrupt is enabled										R/W		
b2	INTENC	Count match interrupt enable C	0: When the GCMCR register is equal to the count value, the interrupt is disabled 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: When the GCMBR register is equal to the count value, or when a capture input event occurs, the interrupt is disabled 1: When the GCMBR register is equal to the count value, or when a capture input event occurs, the interrupt is enabled										R/W		
b0	INTENA	Count Match interrupt enable A	0: When the GCMAR register is equal to the count value, or when a capture input event occurs, the interrupt is disabled 1: When the GCMAR register is equal to the count value, or when a capture input event occurs, the interrupt is enabled										R/W		

20.5.9 Buffer Control Register (TMR6_BCONR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved								BTRD SPB	BTRU SPB	BSE SPB	BEN SPB	BTRD SPA	BTRU SPA	BSE SPA	BEN SPA
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	BTRD P	BTRU P	BSE P	BEN P	BTRD B	BTRU B	BSE B	BEN B	BTRD A	BTRU A	BSE A	BEN A

Bit	Symbol	Bit name	Description	Read and write
b31~b24	Reserved	-	Read as "0", write as "0"	R/W
b23	BTRDSPB	Time setting of special compare reference value buffer transfer DB	0: In triangular waveform mode, buffer value is not transferred when counter counts to valley point 1: In triangular waveform mode, a buffer value transfer occurs when counter counts to the valley point. Note: In sawtooth waveform mode, control of this bit is not required	R/W
b22	BTRUSPB	Time setting of special compare reference value buffer transfer UB	0: In triangular waveform mode, buffer value is not transferred when counter counts to peak 1: In triangular waveform mode, a buffer value transfer occurs when counter counts to peak Note: In sawtooth waveform mode, control of this bit is not needed.	R/W
b21	BSESPB	Specail reference value buffer transfer selection B	0: Single buffer transfer (SCMDR->SCMBR) 1: Double buffer transfer (SCMFR->SCMDR->SCMBR)	R/W
b20	BENSPB	Specail reference value buffer transfer B	0: Buffer transfer is disabled 1: Buffer transfer is enabled	R/W
b19	BTRDSPA	Time setting of special compare reference value buffer transfer DA	0: In triangular waveform mode, buffer value is not transferred when counter counts to valley point 1: In triangular waveform mode, a buffer value transfer occurs when counter counts to the valley point. Note: In sawtooth waveform mode, control of this bit is not required	R/W
b18	BTRUSPA	Time setting of special compare reference value buffer transfer UA	0: In triangular waveform mode, buffer value is not transferred when counter counts to peak 1: In triangular waveform mode, a buffer value transfer occurs when counter counts to peak Note: In sawtooth waveform mode, control of this bit is not needed.	R/W
b17	BSESPA	Specail reference value buffer transfer selection A	0: Single buffer transfer (SCMCR->SCMAR) 1: Double buffer transmission (SCMER->SCMCR->SCMAR)	R/W
b16	BENSPA	Specail reference value buffer transfer A	0: Buffer transfer is disabled 1: Buffer transfer is enabled	R/W
b15~b12	Reserved	-	Read as "0", write as "0"	R/W
b11	BTRDP	Time setting of period value buffer transfer D	0: In triangular waveform mode, buffer value is not transferred when counter counts to valley point 1: In triangular waveform mode, a buffer value transfer occurs when counter counts to the valley point. Note: In sawtooth waveform mode, control of this bit is not needed	R/W
b10	BTRUP	Time setting of period value buffer transfer U	0: In triangular waveform mode, buffer value is not transferred when counter counts to peak 1: In triangular waveform mode, a buffer value transfer occurs when counter counts to peak Note: In sawtooth waveform mode, control of this bit is not needed	R/W
b9	BSEP	Period value buffer transfer selection	0: Single buffer transfer (PERBR->PERAR) 1: Double buffer transmission (PERCR->PERBR->PERAR) Note: The transfer time has nothing to do with the counting mode, only at the overflow point, underflow point of the sawtooth waveform or the valley point of the triangular wave	R/W
b8	BENP	Period value buffer transfer	0: Buffer transfer is disabled 1: Buffer transfer is enabled	R/W
b7	BTRDB	Time setting of general compare value buffer transfer DB	0: In triangular waveform mode, buffer value is not transferred when counter counts to valley point 1: In triangular waveform mode, a buffer value transfer occurs when counter counts to the valley point.	R/W

			Note: In sawtooth waveform mode, control of this bit is not needed	
b6	BTRUB	Time setting of general compare value buffer transfer UB	0: In triangular waveform mode, buffer value is not transferred when counter counts to peak 1: In triangular waveform mode, a buffer value transfer occurs when counter counts to peak Note: In sawtooth waveform mode, control of this bit is not needed	R/W
b5	BSEB	General compare value buffer transfer selection B	Comparation output function: 0: Single buffer transfer (GCMDR->GCMBR) 1: Double buffer transfer (GCMFR->GCMDR->GCMBR) Capture input function: 0: single buffer transfer (GCMBR->GCMDR) 1: Double buffer transfer (GCMBR->GCMDR->GCMFR)	R/W
b4	BENB	General compare value buffer transfer B	0: Buffer transfer is disabled 1: Buffer transfer is enabled	R/W
b3	BTRDA	Time setting of general compare value buffer transfer DA	0: In triangular waveform mode, buffer value is not transferred when counter counts to valley point 1: In triangular waveform mode, a buffer value transfer occurs when counter counts to the valley point. Note: In sawtooth waveform mode, control of this bit is not needed	R/W
b2	BTRUA	Time setting general compare value buffer transfer UA	0: In triangular waveform mode, buffer value is not transferred when counter counts to peak 1: In triangular waveform mode, a buffer value transfer occurs when counter counts to peak Note: In sawtooth waveform mode, control of this bit is not needed	R/W
b1	BSEA	General compare value buffer transfer selection A	Comparation output functions: 0: Single buffer transfer (GCMCR->GCMAR) 1: Double buffer transmission (GCMER->GCMCR->GCMAR) Capture input function: 0: Single buffer transfer (GCMAR->GCMCR) 1: Double buffer transmission (GCMAR->GCMCR->GCMER)	R/W
b0	BENA	General Compare Value Buffer Transfer A	0: Buffer transfer is disabled 1: Buffer transfer is enabled	R/W

20.5.10 DeadTime Control Register (TMR6_DCONR)

Reset value: 0x00000000h

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								DTB TRD	DTB TRU	DTB END	DTB ENU	-	-	SEPA	DTC EN
Bit	Symbol	Bit name		Description								Read and write			
b31~b8	Reserved	-		Read as "0", write as "0"								R/W			
b7	DTBTRD	Time setting of dead time value buffer transfer D		0: In triangular waveform mode, dead time buffer value is not transferred when counter counts to valley point 1: In triangular waveform mode, a dead time buffer value transfer occurs when counter counts to the valley point. Note: In sawtooth waveform mode, control of this bit is not needed								R/W			
b6	DTBTRU	Time setting of dead time value buffer transfer U		0: In triangular waveform mode, dead time buffer value is not transferred when counter counts to peak 1: In triangular waveform mode, a dead time buffer value transfer occurs when counter counts to peak Note: In sawtooth waveform mode, control of this bit is not needed								R/W			
b5	DTBEND	Dead time value buffer transfer D		0: Buffer transfer is disabled 1: Buffer transfer is enabled (DTDBR->DTDAR)								R/W			
b4	DTBENU	Dead time value buffer transfer U		0: Buffer transfer is disabled 1: Buffer transfer is enabled (DTUBR->DTUAR)								R/W			
b3~b2	Reserved	-		Read as "0", write as "0"								R/W			
b1	SEPA	Separation setting		0: DTUAR and DTDAR are set separately 1: The value of DTDAR and the value of DTUAR are automatically equal								R/W			
b0	DTCEN	Dead time function		0: Dead time function is disabled 1: Dead time function is enabled								R/W			

20.5.11 Port Control A Register (TMR6_PCNAR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CAP MDA	-	-	OUT ENA	-	-	EMBS A[1:0]	EMBR A[1:0]	EMBC A[1:0]	-	-	FORC A[1:0]				
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CMBDC A[1:0]		CMBUC A[1:0]		CMADC A[1:0]		CMAUC A[1:0]	UDFC A[1:0]	OVFC A[1:0]		STPC A[1:0]		STAC A[1:0]			

Bit	Symbol	Bit name	Description	Read and write
b31	CAPMDA	Function mode selection A	0: Comparison output function 1: Input Capture function	R/W
b30~b29	Reserved	-	Read as "0", write as "0"	R/W
b28	OUTENA	output enable A	0: TIM6 <t> PWMA port output is disabled (port acts as Timer6 function) 1: TIM6 <t> PWMA port output is enabled (port acts as Timer6 function)	R/W
b27~b26	Reserved	-	Read as "0", write as "0"	R/W
b25~b24	EMBSA[1:0]	EMB event channel selection A	00: Select EMB event channel 0 01: Select EMB event channel 1 10: Select EMB event channel 2 11: Select EMB event channel 3	R/W
b23~b22	EMBRA[1:0]	EMB release method selection A	00: When the selected channel EMB event is invalid, release TIM6 <t> PWMA port immediately (One Shot) 01: When the selected channel EMB event is invalid, release TIM6 <t> PWMA port until the counter counts to overflow (Cycle By Cycle 1) 10: When the selected channel EMB event is invalid, release TIM6 <t> PWMA port until the counter counts to underflow (Cycle By Cycle 2) 11: When the selected channel EMB event is invalid, release TIM6 <t> PWMA port until the counter counts to overflow or underflow (Cycle By Cycle 3)	R/W
b21~b20	EMBCA[1:0]	Port level setting at EMB event A	00: When an EMB event occurs on the selected channel, TIM6 <t> PWMA port outputs normally 01: When an EMB event occurs on the selected channel, TIM6 <t> PWMA port outputs a high-impedance state 10: When an EMB event occurs on the selected channel, TIM6 <t> PWMA port outputs a low level 11: When the selected channel has an EMB event, TIM6 <t> PWMA port outputs a high level	R/W
b19~b18	Reserved	-	Read as "0", write as "0"	R/W
b17~b16	FORCA[1:0]	Forced port level setting A	0x: invalid setting 10: When the next cycle starts, TIM6 <t> PWMA port output is set to low level 11: When the next cycle starts, TIM6 <t> PWMA port output is set to high level Note 1: The next period refers to the hardware counting mode or counting to the overflow point or underflow point in sawtooth waveform, counting to the valley point in the triangular waveform Note 2: This register bit can be used to achieve the control of 0% or 100% duty of PWM output .	R/W
b15~b14	CMBDCA[1:0]	Port level setting A to down-count && matching compare value B	00: During down counting, when the counter value is equal to GCMR, TIM6 <t> PWMA port output is set to low level 01: During down counting, when the counter value is equal to GCMR, TIM6 <t> PWMA port output is set to high level 10: during down counting, when the counter value equals to GCMR, TIM6 <t> PWMA port output maintains previous level 11: During down counting, when the counter value is equal to GCMR, TIM6 <t> PWMA port output is set to the inverted level	R/W
b13~b12	CMBUCA[1:0]	Port level setting A to up-count && matching compare value B	00: During up counting, when the counter value is equal to GCMR, TIM6 <t> PWMA port output is set to low level 01: During up counting, when the counter value is equal to GCMR, TIM6 <t> PWMA port output is set to high level 10: During up counting, when counter equals to GCMR, TIM6 <t> PWMA port output maintains previous level.	R/W

			11: During up counting, when the counter value is equal to GCMBR, TIM6_<t>_PWMA port output is set to the inverted level	
b11~b10	CMADCA[1:0]	Port level setting A to down- count && matching compare value A	00: During down counting, when the counter value is equal to GCMAR, TIM6_<t>_PWMA port output is set to low level 01: During down counting, when the counter value is equal to GCMAR, TIM6_<t>_PWMA port output is set to high level 10: During down counting, when counter value equals to GCMAR, TIM6_<t>_PWMA port output maintains previous level 11: During down counting, when counter value is equal to GCMAR, TIM6_<t>_PWMA port output is set to the inverted level	R/W
b9~b8	CMAUCA[1:0]	Port level setting A to up-count && matching compare value A	00: During up counting, when the counter value is equal to GCMAR, TIM6_<t>_PWMA port output is set to low level 01: During up counting, when the counter value is equal to GCMAR, TIM6_<t>_PWMA port output is set to high level 10: During up counting, when counter value equals to GCMAR, TIM6_<t>_PWMA port output maintains previous level 11: During up counting, when counter value is equal to GCMAR, TIM6_<t>_PWMA port output is set to the inverted level	R/W
b7~b6	UDFCA[1:0]	Port level setting A to count underflow	00: When the count underflows, TIM6_<t>_PWMA port output is set to low level 01: When the count underflows, TIM6_<t>_PWMA port output is set to high level 10: When the count underflows, TIM6_<t>_PWMA port output maintains the previous level 11: When the count underflows, TIM6_<t>_PWMA port output is set to the inverted level	R/W
b5~b4	OVFCA[1:0]	Port level setting A to count overflow	00: When the count overflows, TIM6_<t>_PWMA port output is set to low level 01: When the count overflows, TIM6_<t>_PWMA port output is set to high level 10: When the count overflows, TIM6_<t>_PWMA port output maintains the previous level 11: When the count overflows, TIM6_<t>_PWMA port output is set to the inverted level	R/W
b3~b2	STPCA[1:0]	Port level setting A to count stop	00: When count stops, TIM6_<t>_PWMA port output is set to low level 01: When count stops, TIM6_<t>_PWMA port output is set to high level 10: When count stops, TIM6_<t>_PWMA port output maintains the previous level 11: When count stops, TIM6_<t>_PWMA port output maintains the previous level	R/W
b1~b0	STACA[1:0]	Port level setting A to count start	00: When count starts, TIM6_<t>_PWMA port output is set to low level 01: When count starts, TIM6_<t>_PWMA port output is set to high level 10: When count starts, TIM6_<t>_PWMA port output maintains previous level 11: When count starts, TIM6_<t>_PWMA port output maintains previous level	R/W

20.5.12 Port Control B Register (TMR6_PCNBR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CAP MDB	-	-	OUT ENB	-	-	EMBS B[1:0]		EMBR B[1:0]		EMBC B[1:0]	-	-	FORC B[1:0]		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CMBDC B[1:0]		CMBUC B[1:0]		CMADC B[1:0]		CMAUC B[1:0]		UDFC B[1:0]		OVFC B[1:0]		STPC B[1:0]		STAC B[1:0]	

Bit	Symbol	Bit name	Description	Read and write
b31	CAPMDB	Function mode selection B	0: Comparison output function 1: Input Capture function	R/W
b30~b29	Reserved	-	Read as "0", write as "0"	R/W
b28	OUTENB	output enable B	0: TIM6 <t> PWMB port output is disabled (port acts as Timer6 function) 1: TIM6 <t> PWMB port output is enabled (port acts as Timer6 function)	R/W
b27~b26	Reserved	-	Read as "0", write as "0"	R/W
b25~b24	EMBSB[1:0]	EMB event channel selection B	00: Select EMB event channel 0 01: Select EMB event channel 1 10: Select EMB event channel 2 11: Select EMB event channel 3	R/W
b23~b22	EMBRB[1:0]	EMB release method selection B	00: When the selected channel EMB event is invalid, release TIM6 <t> PWMB port immediately (One Shot) 01: When the selected channel EMB event is invalid, release TIM6 <t> PWMB port until the counter counts to overflow (Cycle By Cycle 1) 10: When the selected channel EMB event is invalid, release TIM6 <t> PWMB port until the counter counts to underflow (Cycle By Cycle 2) 11: When the selected channel EMB event is invalid, release TIM6 <t> PWMB port until the counter counts to overflow or underflow (Cycle By Cycle 3)	R/W
b21~b20	EMBCB[1:0]	Port level setting B at EMB event	00: When an EMB event occurs on the selected channel, TIM6 <t> PWMB port outputs normally 01: When an EMB event occurs on the selected channel, TIM6 <t> PWMB port outputs a high-impedance state 10: When an EMB event occurs on the selected channel, TIM6 <t> PWMB port outputs a low level 11: When the selected channel has an EMB event, TIM6 <t> PWMB port outputs a high level	R/W
b19~b18	Reserved	-	Read as "0", write as "0"	R/W
b17~b16	FORCB[1:0]	Forced port level setting B	0x: invalid setting 10: When the next period starts, TIM6 <t> PWMB port output is set to low level 11: When the next period starts, TIM6 <t> PWMB port output is set to high level Note 1: The next period refers to the hardware counting mode or counting to the overflow point or underflow point in sawtooth waveform, counting to the valley point in the triangular waveform Note 2: This register bit can be used to achieve the control of 0% or 100% duty of PWM output .	R/W
b15~b14	CMBDCB[1:0]	Port level setting B to down-count && matching compare value B	00: During down counting, when the counter value is equal to GCMBR, TIM6 <t> PWMB port output is set to low level 01: During down counting, when the counter value is equal to GCMBR, TIM6 <t> PWMB port output is set to high level 10: During down counting, when the counter value equals to GCMBR, TIM6 <t> PWMB port output maintains previous level 11: During down counting, when the counter value is equal to GCMBR, TIM6 <t> PWMB port output is set to the inverted level	R/W
b13~b12	CMBUCB[1:0]	Port level setting B to up-count && matching compare value B	00: During up counting, when the counter value is equal to GCMBR, TIM6 <t> PWMB port output is set to low level 01: During up counting, when the counter value is equal to GCMBR, TIM6 <t> PWMB port output is set to high level 10: During up counting, when the counter equals to GCMBR, TIM6 <t> PWMB port output maintains previous	R/W

			level.	
			11: During the up counting, when the counter value is equal to GCMAR, TIM6_<t>_PWMB port output is set to the inverted level.	
b11~b10	CMADCB[1:0]	Port level setting B to down- count && matching compare value A	00: During down counting, when the counter value is equal to GCMAR, TIM6_<t>_PWMB port output is set to low level 01: During down counting, when the counter value is equal to GCMAR, TIM6_<t>_PWMB port output is set to high level 10: During down counting, when counter value equals to GCMAR, TIM6_<t>_PWMB port output maintains previous level 11: During down counting, when counter value is equal to GCMAR, TIM6_<t>_PWMB port output is set to the inverted level	R/W
b9~b8	CMAUCB[1:0]	Port level setting B to up-count && matching compare value A	00: During up counting, when the counter value is equal to GCMAR, TIM6_<t>_PWMB port output is set to low level 01: During up counting, when the counter value is equal to GCMAR, TIM6_<t>_PWMB port output is set to high level 10: During up counting, when counter value equals to GCMAR, TIM6_<t>_PWMB port output maintains previous level 11: During up counting, when counter value is equal to GCMAR, TIM6_<t>_PWMB port output is set to the inverted level	R/W
b7~b6	UDFCB[1:0]	Port level setting B to count underflow	00: When the count underflows, TIM6_<t>_PWMB port output is set to low level 01: When the count underflows, TIM6_<t>_PWMB port output is set to high level 10: When the count underflows, TIM6_<t>_PWMB port output maintains the previous level 11: When the count underflows, TIM6_<t>_PWMB port output is set to the inverted level	R/W
b5~b4	OVFCB[1:0]	Port level setting B to count overflow	00: When the count overflows, TIM6_<t>_PWMB port output is set to low level 01: When the count overflows, TIM6_<t>_PWMB port output is set to high level 10: When the count overflows, TIM6_<t>_PWMB port output maintains the previous level 11: When the count overflows, TIM6_<t>_PWMB port output is set to the inverted level	R/W
b3~b2	STPCB[1:0]	Port level setting B to count stop	00: When count stops, TIM6_<t>_PWMB port output is set to low level 01: When count stops, TIM6_<t>_PWMB port output is set to high level 10: When count stops, TIM6_<t>_PWMB port output maintains the previous level 11: When count stops, TIM6_<t>_PWMB port output maintains the previous level	R/W
b1~b0	STACB[1:0]	port level setting B to count start	00: When count starts, TIM6_<t>_PWMB port output is set to low level 01: When count starts, TIM6_<t>_PWMB port output is set to high level 10: When count starts, TIM6_<t>_PWMB port output maintains previous level 11: When count starts, TIM6_<t>_PWMB port output maintains previous level	R/W

20.5.13 Filter Control General Port Register (TMR6_FCNGR)

Reset value: 0x00000000h

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															
Bit	Symbol	Bit name	Description										Read and write		
b31~b7	Reserved	-	Read as "0", write as "0"										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	Input Capture port filter GB	0: Filter function of TIM6_<t>_PWMB input port is disabled 1: Filter function of TIM6_<t>_PWMB input port is enabled										R/W		
b3	Reserved	-	Read as "0", write as "0"										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	Input Capture port filtering GA	0: Filter function of TIM6_<t>_PWMA input port is disabled 1: Filter function of TIM6_<t>_PWMA input port is enabled										R/W		

20.5.14 Filter Control Trigger Port Register (TMR6_FCNTR)

Reset value: 0x00000000h

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
-	NOFI CKTD[1:0]	NOFI ENTD	-	NOFI CKTC[1:0]	NOFI ENTC	-	NOFI CKTB[1:0]	NOFI ENTB	-	NOFI CKTA[1:0]	NOFI ENTA				
Bit	Symbol	Bit name		Description								Read and write			
b31~b15	Reserved	-		Read as "0", write as "0"								R/W			
b14~b13	NOFICKTD[1:0]	Filter sampling reference clock selection TD		00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64								R/W			
b12	NOFIENTD	Input Capture port filter TD		0: Filter function of TIM6_TRIGD port input is disabled 1: Filter function of TIM6_TRIGD port input is enabled								R/W			
b11	Reserved	-		Read as "0", write as "0"								R/W			
b10~b9	NOFICKTC[1:0]	Filter sampling reference clock selection TC		00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64								R/W			
b8	NOFIENTC	Input Capture port filter TC		0: Filter function of TIM6_TRIGC port input is disabled 1: Filter function of TIM6_TRIGC port input is enabled								R/W			
b7	Reserved	-		Read as "0", write as "0"								R/W			
b6~b5	NOFICKTB[1:0]	Filter sampling reference clock selection TB		00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64								R/W			
b4	NOFIENTB	Input Capture port filter TB		0: Filter function of TIM6_TRIGB port input is disabled 1: Filter function of TIM6_TRIGB port input is enabled								R/W			
b3	Reserved	-		Read as "0", write as "0"								R/W			
b2~b1	NOFICKTA[1:0]	Filter sampling reference clock selection TA		00: PCLK0 01: PCLK0/4 10: PCLK0/16 11: PCLK0/64								R/W			
b0	NOFIENTA	Input Capture port filtering TA		0: Filter function of TIM6_TRIGA port input is disabled 1: Filter function of TIM6_TRIGA port input is enabled								R/W			

20.5.15 Valid Period Register (TMR6_VPERR)

Reset value: 0x00000000h

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							
<hr/>															
Bit	Symbol	Bit name		Description										Read and write	
b31~b21	Reserved	-		Read as "0", write as "0"										R/W	
<hr/>															
b20~b18	PCNTS[2:0]	Valid period selection		000: Valid period selection function is disabled 001: Valid every 2 period 010: Valid every 3 periods 011: Valid every 4 periods 100: Valid every 5 periods 101: Valid every 6 periods 110: Valid every 7 periods 111: Valid every 8 periods For example: PCNTS=010, Period N: valid, Period N+1: invalid, N+2: invalid, Period N+3: valid Refer to chapter "Periodic interval response" for details.										R/W	
<hr/>															
b17~b16	PCNTE[1:0]	Valid Period Counting Condition Selection		00: Valid period selection function is disabled 01: The overflow points and underflow point in sawtooth waveform, or valey point in triangular waveform act as the counting condition 10: The overflow points and underflow point in sawtooth waveform, or peak point in triangular waveform act as the counting condition 11: The overflow points and underflow point in sawtooth waveform , or peak point and valley point in triangular waveform act as the counting condition										R/W	
<hr/>															
b15~b10	Reserved	-		Read as "0", write as "0"										R/W	
<hr/>															
b9	SPPERIB	Specail signal valid period selection B		0: Valid period selection function is disabled 1: Valid period selection function is enabled										R/W	
<hr/>															
b8	SPPERIA	Specail signal valid period selection A		0: Valid period selection function is disabled 1: Valid period selection function is enabled										R/W	
<hr/>															
b7~b0	Reserved	-		Read as "0", write as "0"										R/W	

20.5.16 Status Register (TMR6_STFLR)

Reset value: 0x80000000h

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
-	-	-	CMSBDF	CMSBUF	CMSADF	CMSAUF	DTEF	UDFF	OVFF	CMFF	CMEF	CMDF	CMCF	CMBF	CMAF
Bit	Symbol	Bit name		Description								Read and write			
b31	DIRF	Counting direction		0: count down 1: count up								R			
b30~b24	Reserved	-		Read as "0", write as "0"								R			
b23~b21	VPERNUM[2:0]	Number of periods		The number of periods after counting when the valid period selection function is enabled								R			
b20~b13	Reserved	-		Read as "0", write as "0"								R			
b12	CMSBDF	Specail compare value match at down-count B		0: When counting down, the value of the SCMBR register is not equal to the count value 1: When counting down, the value of the SCMBR register is equal to the count value								R/W			
b11	CMSBUF	Specail compare value match at up-count B		0: When counting up, the value of the SCMBR register is not equal to the count value 1: When counting up, the value of the SCMBR register is equal to the count value								R/W			
b10	CMSADF	Specail compare value match at down-count A		0: When counting down, the value of the SCMAR register is not equal to the count value 1: When counting down, the value of the SCMAR register is equal to the count value								R/W			
b09	CMSAUF	Specail compare reference value match at up-count A		0: When counting up, the value of the SCMAR register is not equal to the count value 1: When counting up, the value of the 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 match		0: No underflow in sawtooth waveform or valley point in triangular wave 1: Underflow occurred in sawtooth waveform or valley point occurred in triangular wave								R/W			
b6	OVFF	overflow match		0: No overflow in sawtooth waveform or peak point in triangular waveform 1: Overflow occurred in sawtooth waveform or peak point occurred in triangular waveform								R/W			
b5	CMFF	counting match F		0: The value of the GCMFR register is not equal to the count value 1: The value of the GCMFR register is equal to the count value								R/W			
b4	CMEF	counting match E		0: The value of the GCMER register is not equal to the count value 1: The value of the GCMER register is equal to the count value								R/W			
b3	CMDF	counting match D		0: The value of the GCMDR register is not equal to the count value 1: The value of the GCMDR register is equal to the count value								R/W			
b2	CMCF	counting match C		0: The value of the GCMCR register is not equal to the count value 1: The value of the 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 the TIM6_<t>_PWMB capture input action has not occurred 1: The value of the GCMBR register is equal to the count value, or the TIM6_<t>_PWMB capture input action has completed								R/W			
b0	CMAF	counting match A		0: The value of the GCMAR register is not equal to the count value, and the TIM6_<t>_PWMA capture input action has not occurred 1: The value of the GCMAR register is equal to the count value, or the TIM6_<t>_PWMA capture input action has								R/W			

completed

20.5.17 Hardware Start Event Select Register (TMR6_HSTAR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										HSTA23	HSTA22	HSTA21	HSTA20	HSTA19	HSTA18
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HSTA11	HSTA10	HSTA9	HSTA8	STAS	-	-	-	HSTA3	HSTA2	HSTA1	HSTA0

Bit	Symbol	Bit name	Description	Read and write
b31~b24	Reserved	-	Read as "0", write as "0"	R/W
b23	HSTA23	Hardware start condition 23	Condition: A falling edge was sampled on TIM6_TRIGD port 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W
b22	HSTA22	Hardware start condition 22	Condition: A rising edge was sampled in TIM6_TRIGD port 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W
b21	HSTA21	Hardware start condition 21	Condition: A falling edge was sampled on TIM6_TRIGC port 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W
b20	HSTA20	Hardware start condition 20	Condition: A rising edge was sampled on TIM6_TRIGC port 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W
b19	HSTA19	Hardware start condition 19	Condition: A falling edge was sampled on TIM6_TRIGB port 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W
b18	HSTA18	Hardware start condition 18	Condition: A rising edge was sampled on TIM6_TRIGB port 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W
b17	HSTA17	Hardware start condition 17	Condition: A falling edge was sampled on TIM6_TRIGA port 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W
b16	HSTA16	Hardware start condition 16	Condition: A rising edge was sampled on TIM6_TRIGA port 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W
b15~b12	Reserved	-	Read as "0", write as "0"	R/W
b11	HSTA11	Hardware start condition 11	Condition: Internal hardware trigger event 3 is valid 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W
b10	HSTA10	Hardware start condition 10	Condition: Internal hardware trigger event 2 is valid 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W
b9	HSTA9	Hardware start condition 9	Condition: Internal hardware trigger event 1 is valid 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W
b8	HSTA8	Hardware start condition 8	Condition: Internal hardware trigger event 0 is valid 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W
b7	STAS	hardware start enable	0: Hardware start function is disabled 1: Hardware start function is enabled Note: When the hardware start function is enabled, the setting of SSTAR is invalid.	R/W
b6~b4	Reserved	-	Read as "0", write as "0"	R/W
b3	HSTA3	Hardware start condition 3	Condition: A falling edge was sampled on TIM6_<t>_PWMB port 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W
b2	HSTA2	Hardware start condition 2	Condition: A rising edge was sampled on TIM6_<t>_PWMB port 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W

b1	HSTA1	Hardware start condition 1	Condition: A falling edge was sampled on TIM6_<t>_PWMA port 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W
b0	HSTA0	Hardware start condition 0	Condition: A rising edge was sampled on TIM6_<t>_PWMA port 0: Hardware start is disabled when condition matches 1: Hardware start is enabled when condition matches	R/W

20.5.18 Hardware Stop Event Select Register (TMR6_HSTPR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved						HSTP 23	HSTP 22	HSTP 21	HSTP 20	HSTP 19	HSTP 18	HSTP 17	HSTP 16		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HSTP 11	HSTP 10	HSTP 9	HSTP 8	STPS	-	-	-	HSTP 3	HSTP 2	HSTP 1	HSTP 0

Bit	Symbol	Bit name	Description	Read and write
b31~b24	Reserved	-	Read as "0", write as "0"	R/W
b23	HSTP23	Hardware stop condition 23	Condition: A falling edge was sampled on TIM6_TRIGD port 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W
b22	HSTP22	Hardware stop condition 22	Condition: A rising edge was sampled on TIM6_TRIGD port 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W
b21	HSTP21	Hardware stop condition 21	Condition: A falling edge was sampled on TIM6_TRIGC port 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W
b20	HSTP20	Hardware stop condition 20	Condition: A rising edge was sampled on TIM6_TRIGC port 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W
b19	HSTP19	Hardware stop condition 19	Condition: A falling edge was sampled on TIM6_TRIGB port 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W
b18	HSTP18	Hardware stop condition 18	Condition: A rising edge was sampled on TIM6_TRIGB port 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W
b17	HSTP17	Hardware stop condition 17	Condition: A falling edge was sampled on TIM6_TRIGA port 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W
b16	HSTP16	Hardware stop condition 16	Condition: A rising edge was sampled on TIM6_TRIGA port 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W
b15~b12	Reserved	-	Read as "0", write as "0"	R/W
b11	HSTP11	Hardware stop condition 11	Condition: Internal hardware trigger event 3 is valid 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W
b10	HSTP10	Hardware stop condition 10	Condition: Internal hardware trigger event 2 is valid 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W
b9	HSTP9	Hardware stop condition 9	Condition: Internal hardware trigger event 1 is valid 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W
b8	HSTP8	Hardware stop condition 8	Condition: Internal hardware trigger event 0 is valid 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W
b7	STPS	hardware stop enable	0: Hardware stop function is disabled 1: Hardware stop function is enabled Note: When the hardware stop function is enabled, the setting of SSTPR is invalid.	R/W
b6~b4	Reserved	-	Read as "0", write as "0"	R/W
b3	HSTP3	Hardware stop condition 3	Condition: A falling edge was sampled on TIM6_<t>_PWMB port 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W
b2	HSTP2	Hardware stop condition 2	Condition: A rising edge was sampled on TIM6_<t>_PWMB port 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W

b1	HSTP1	Hardware stop condition 1	Condition: A falling edge was sampled on TIM6_<t>_PWMA port 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W
b0	HSTP0	Hardware stop condition 0	Condition: A rising edge was sampled on TIM6_<t>_PWMA port 0: Hardware stop is disabled when condition matches 1: Hardware stop is enabled when condition matches	R/W

20.5.19 Hardware Clear Event Select Register (TMR6_HCLRR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										HCLE23	HCLE22	HCLE21	HCLE20	HCLE19	HCLE18
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HCLE11	HCLE10	HCLE9	HCLE8	CLES	-	-	-	HCLE3	HCLE2	HCLE1	HCLE0

Bit	Symbol	Bit name	Description	Read and write
b31~b24	Reserved	-	Read as "0", write as "0"	R/W
b23	HCLE23	Hardware clear condition 23	Condition: A falling edge was sampled on TIM6_TRIGD port 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W
b22	HCLE22	Hardware clear condition 22	Condition: A rising edge was sampled on TIM6_TRIGD port 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W
b21	HCLE21	Hardware clear condition 21	Condition: A falling edge was sampled on TIM6_TRIGC port 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W
b20	HCLE20	Hardware clear condition 20	Condition: A rising edge was sampled on TIM6_TRIGC port 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W
b19	HCLE19	Hardware clear condition 19	Condition: A falling edge was sampled on TIM6_TRIGB port 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W
b18	HCLE18	Hardware clear condition 18	Condition: A rising edge was sampled on TIM6_TRIGB port 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W
b17	HCLE17	Hardware clear condition 17	Condition: A falling edge was sampled on TIM6_TRIGA port 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W
b16	HCLE16	Hardware clear condition 16	Condition: A rising edge was sampled on TIM6_TRIGA port 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W
b15~b12	Reserved	-	Read as "0", write as "0"	R/W
b11	HCLE11	Hardware clear condition 11	Condition: Internal hardware trigger event 3 is valid 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W
b10	HCLE10	Hardware clear condition 10	Condition: Internal hardware trigger event 2 is valid 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W
b9	HCLE9	Hardware clear condition 9	Condition: Internal hardware trigger event 1 is valid 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W
b8	HCLE8	Hardware clear condition 8	Condition: Internal hardware trigger event 0 is valid 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W
b7	CLES	Hardware clear enable	0: Hardware clear function is disabled 1: Hardware clear function is enabled Note: When hardware clear function is enabled, the setting of SCLR is invalid	R/W
b6~b4	Reserved	-	Read as "0", write as "0"	R/W
b3	HCLE3	Hardware clear condition 3	Condition: A falling edge was sampled on TIM6_<t>_PWMB port 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W
b2	HCLE2	Hardware clear condition 2	Condition: A rising edge was sampled on TIM6_<t>_PWMB port 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W

b1	HCLE1	Hardware clear condition 1	Condition: A falling edge was sampled on TIM6_<t>_PWMA port 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W
b0	HCLE0	Hardware clear condition 0	Condition: A rising edge was sampled on TIM6_<t>_PWMA port 0: Hardware clear is disabled when condition matches 1: Hardware clear is enabled when condition matches	R/W

20.5.20 Hardware Update Event Select Register (TMR6_HUPDR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved								HUPD23	HUPD22	HUPD21	HUPD20	HUPD19	HUPD18	HUPD17	HUPD16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HUPD11	HUPD10	HUPD9	HUPD8	UPDS	-	-	-	HUPD3	HUPD2	HUPD1	HUPD0
<hr/>															
Bit	Symbol	Bit name	Description								Read and write				
b31~b24	Reserved	-	Read as "0", write as "0"								R/W				
b23	HUPD23	Hardware refresh condition 23	Condition: A falling edge was sampled on TIM6_TRIGD port 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches								R/W				
b22	HUPD22	Hardware refresh condition 22	Condition: A rising edge was sampled on TIM6_TRIGD port 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches								R/W				
b21	HUPD21	Hardware refresh condition 21	Condition: A falling edge was sampled on TIM6_TRIGC port 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches								R/W				
b20	HUPD20	Hardware refresh condition 20	Condition: A rising edge was sampled on TIM6_TRIGC port 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches								R/W				
b19	HUPD19	Hardware refresh condition 19	Condition: A falling edge was sampled on TIM6_TRIGB port 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches								R/W				
b18	HUPD18	Hardware refresh condition 18	Condition: A rising edge was sampled on TIM6_TRIGB port 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches								R/W				
b17	HUPD17	Hardware refresh condition 17	Condition: A falling edge was sampled on TIM6_TRIGA port 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches								R/W				
b16	HUPD16	Hardware refresh condition 16	Condition: A rising edge was sampled on TIM6_TRIGA port 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches								R/W				
b15~b12	Reserved	-	Read as "0", write as "0"								R/W				
b11	HUPD11	Hardware refresh condition 11	Condition: Internal hardware trigger event 3 is valid 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches								R/W				
b10	HUPD10	Hardware refresh condition 10	Condition: Internal hardware trigger event 2 is valid 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches								R/W				

b9	HUPD9	Hardware refresh condition 9	Condition: Internal hardware trigger event 1 is valid 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches	R/W
b8	HUPD8	Hardware refresh condition 8	Condition: Internal hardware trigger event 0 is valid 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches	R/W
b7	UPDS	Hardware refresh enable	0: Hardware refresh function is disabled 1: Hardware refresh function is enabled Note: When hardware refresh function is enabled, the SUPDR setting is invalid	R/W
b6~b4	Reserved	-	Read as "0", write as "0"	R/W
b3	HUPD3	Hardware refresh condition 3	Condition: A falling edge was sampled on TIM6_<t>_PWMB port 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches	R/W
b2	HUPD2	Hardware refresh condition 2	Condition: A rising edge was sampled on TIM6_<t>_PWMB port 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches	R/W
b1	HUPD1	Hardware refresh condition 1	Condition: A falling edge was sampled on TIM6_<t>_PWMA port 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches	R/W
b0	HUPD0	Hardware refresh condition 0	Condition: A rising edge was sampled on TIM6_<t>_PWMA port 0: Hardware refresh is disabled when condition matches 1: Hardware refresh is enabled when condition matches	R/W

20.5.21 Hardware Input Capture A Event Select Register (TMR6_HCPAR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved								HCP A23	HCP A22	HCP A21	HCP A20	HCP A19	HCP A18	HCP A17	HCP A16
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 A3	HCP A2	HCP A1	HCP A0

Bit	Symbol	Bit name	Description	Read and write
b31~b24	Reserved	-	Read as "0", write as "0"	R/W
b23	HCPA23	Hardware Capture A Condition 23	Condition: A falling edge was sampled on TIM6_TRIGD port 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W
b22	HCPA22	Hardware Capture A Condition 22	Condition: A rising edge was sampled on TIM6_TRIGD port 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W
b21	HCPA21	Hardware Capture A Condition 21	Condition: A falling edge was sampled on TIM6_TRIGC port 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W
b20	HCPA20	Hardware Capture A Condition 20	Condition: A rising edge was sampled on TIM6_TRIGC port 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W
b19	HCPA19	Hardware Capture A Condition 19	Condition: A falling edge was sampled on TIM6_TRIGB port 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W
b18	HCPA18	Hardware Capture A Condition 18	Condition: A rising edge was sampled on TIM6_TRIGB port 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W
b17	HCPA17	Hardware Capture A Condition 17	Condition: A falling edge was sampled on TIM6_TRIGA port 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W
b16	HCPA16	Hardware Capture A Condition 16	Condition: A rising edge was sampled on TIM6_TRIGA port 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W
b15~b12	Reserved	-	Read as "0", write as "0"	R/W
b11	HCPA11	Hardware Capture A Condition 11	Condition: Internal hardware trigger event 3 is valid 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W
b10	HCPA10	Hardware Capture A Condition 10	Condition: Internal hardware trigger event 2 is valid 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W

b9	HCPA9	Hardware Capture A Condition 9	Condition: Internal hardware trigger event 1 is valid 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W
b8	HCPA8	Hardware Capture A Condition 8	Condition: Internal hardware trigger event 0 is valid 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W
b7~b4	Reserved	-	Read as "0", write as "0"	R/W
b3	HCPA3	Hardware Capture A Condition 3	Condition: A falling edge was sampled on TIM6_<t>_PWMB port 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W
b2	HCPA2	Hardware Capture A Condition 2	Condition: A rising edge was sampled on TIM6_<t>_PWMB port 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W
b1	HCPA1	Hardware Capture A Condition 1	Condition: A falling edge was sampled on TIM6_<t>_PWMA port 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W
b0	HCPA0	Hardware Capture A Condition 0	Condition: A rising edge was sampled on TIM6_<t>_PWMA port 0: Hardware capture A is disabled when condition matches 1: Hardware capture A is enabled when condition matches	R/W

20.5.22 Hardware Input Capture B Event Select Register (TMR6_HCPBR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved								HCPB23	HCPB22	HCPB21	HCPB20	HCPB19	HCPB18	HCPB17	HCPB16
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	HCPB11	HCPB10	HCPB9	HCPB8	-	-	-	-	HCPB3	HCPB2	HCPB1	HCPB0

Bit	Symbol	Bit name	Description	Read and write
b31~b24	Reserved	-	Read as "0", write as "0"	R/W
b23	HCPB23	Hardware Capture B Condition 23	Condition: A falling edge was sampled on TIM6_TRIGD port 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W
b22	HCPB22	Hardware Capture B Condition 22	Condition: A rising edge was sampled on TIM6_TRIGD port 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W
b21	HCPB21	Hardware Capture B Condition 21	Condition: A falling edge was sampled on TIM6_TRIGC port 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W
b20	HCPB20	Hardware Capture B Condition 20	Condition: A rising edge was sampled on TIM6_TRIGC port 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W
b19	HCPB19	Hardware Capture B Condition 19	Condition: A falling edge was sampled on TIM6_TRIGB port 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W
b18	HCPB18	Hardware Capture B Condition 18	Condition: A rising edge was sampled on TIM6_TRIGB port 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W
b17	HCPB17	Hardware Capture B Condition 17	Condition: A falling edge was sampled on TIM6_TRIGA port 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W
b16	HCPB16	Hardware Capture B Condition 16	Condition: A rising edge was sampled on TIM6_TRIGA port 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W
b15~b12	Reserved	-	Read as "0", write as "0"	R/W
b11	HCPB11	Hardware Capture B Condition 11	Condition: Internal hardware trigger event 3 is valid 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W
b10	HCPB10	Hardware Capture B Condition 10	Condition: Internal hardware trigger event 2 is valid 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W

b9	HCPB9	Hardware Capture B Condition 9	Condition: Internal hardware trigger event 1 is valid 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W
b8	HCPB8	Hardware Capture B Condition 8	Condition: Internal hardware trigger event 0 is valid 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W
b7~b4	Reserved	-	Read as "0", write as "0"	R/W
b3	HCPB3	Hardware Capture B Condition 3	Condition: A falling edge was sampled on TIM6_<t>_PWMB port 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W
b2	HCPB2	Hardware Capture B Condition 2	Condition: A rising edge was sampled on TIM6_<t>_PWMB port 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W
b1	HCPB1	Hardware Capture B Condition 1	Condition: A falling edge was sampled on TIM6_<t>_PWMA port 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W
b0	HCPB0	Hardware Capture B Condition 0	Condition: A rising edge was sampled on TIM6_<t>_PWMA port 0: Hardware capture B is disabled when condition matches 1: Hardware capture B is enabled when condition matches	R/W

20.5.23 Hardware Count Up Event Select Register (TMR6_HCUPR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved								HC UP23	HC UP22	HC UP21	HC UP20	HC UP19	HC UP18	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

Bit	Symbol	Bit name	Description	Read and write
b31~b24	Reserved	-	Read as "0", write as "0"	R/W
b23	HCUP23	Hardware Count Up Condition 23	Condition: A falling edge was sampled on TIM6_TRIGD port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W
b22	HCUP22	Hardware Count Up Condition 22	Condition: A rising edge was sampled on TIM6_TRIGD port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W
b21	HCUP21	Hardware Count Up Condition 21	Condition: A falling edge was sampled on TIM6_TRIGC port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W
b20	HCUP20	Hardware Count Up Condition 20	Condition: A rising edge was sampled on TIM6_TRIGC port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W
b19	HCUP19	Hardware Count Up Condition 19	Condition: A falling edge was sampled on TIM6_TRIGB port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	
b18	HCUP18	Hardware Count Up Condition 18	Condition: A rising edge was sampled on TIM6_TRIGB port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	
b17	HCUP17	Hardware Count Up Condition 17	Condition: A falling edge was sampled on TIM6_TRIGA port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	
b16	HCUP16	Hardware Count Up Condition 16	Condition: A rising edge was sampled on TIM6_TRIGA port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	
b15~b12	Reserved	-	Read as "0", write as "0"	R/W
b11	HCUP11	Hardware Count Up Condition 11	Condition: Internal hardware trigger event 3 is valid 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W
b10	HCUP10	Hardware Count Up Condition 10	Condition: Internal hardware trigger event 2 is valid 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W

b9	HCUP9	Hardware Count Up Condition 9	Condition: Internal hardware trigger event 1 is valid 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W
b8	HCUP8	Hardware Count Up Condition 8	Condition: Internal hardware trigger event 0 is valid 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W
b7	HCUP7	Hardware Count Up Condition 7	Condition: When TIM6_<t>_PWMB port is high, a falling edge was sampled on TIM6_<t>_PWMA port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W
b6	HCUP6	Hardware Count Up Condition 6	Condition: When TIM6_<t>_PWMB port is high, a rising edge was sampled on TIM6_<t>_PWMA port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W
b5	HCUP5	Hardware Count Up Condition 5	Condition: When TIM6_<t>_PWMB port is low, a falling edge was sampled on TIM6_<t>_PWMA port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W
b4	HCUP4	Hardware Count Up Condition 4	Condition: When TIM6_<t>_PWMB port is low, a rising edge was sampled on TIM6_<t>_PWMA port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W
b3	HCUP3	Hardware Count Up Condition 3	Condition: When TIM6_<t>_PWMA port is high, a falling edge was sampled on TIM6_<t>_PWMB port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W
b2	HCUP2	Hardware Count Up Condition 2	Condition: When TIM6_<t>_PWMA port is high, a rising edge was sampled on TIM6_<t>_PWMB port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W
b1	HCUP1	Hardware Count Up Condition 1	Condition: When TIM6_<t>_PWMA port is low, a falling edge was sampled on TIM6_<t>_PWMB port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W
b0	HCUP0	Hardware Count Up Condition 0	Condition: When TIM6_<t>_PWMA port is low, a rising edge was sampled on TIM6_<t>_PWMB port 0: Hardware Count Up is disabled when condition matches 1: Hardware Count Up is enabled when condition matches	R/W

20.5.24 Hardware Count Down Event Select Register (TMR6_HCDOR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved						HC DO23	HC DO22	HC DO21	HC DO20	HC DO19	HC DO18	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

Bit	Symbol	Bit name	Description	Read and write
b31~b24	Reserved	-	Read as "0", write as "0"	R/W
b23	HCDO23	Hardware count down condition 23	Condition: A falling edge was sampled on TIM6_TRIGD port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W
b22	HCDO22	Hardware count down condition 22	Condition: A rising edge was sampled on TIM6_TRIGD port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W
b21	HCDO21	Hardware count down condition 21	Condition: A falling edge was sampled on TIM6_TRIGC port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W
b20	HCDO20	Hardware count down condition 20	Condition: A rising edge was sampled on TIM6_TRIGC port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W
b19	HCDO19	Hardware count down condition 19	Condition: A falling edge was sampled on TIM6_TRIGB port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	
b18	HCDO18	Hardware count down condition 18	Condition: A rising edge was sampled on TIM6_TRIGB port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	
b17	HCDO17	Hardware count down condition 17	Condition: A falling edge was sampled on TIM6_TRIGA port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	
b16	HCDO16	Hardware count down condition 16	Condition: A rising edge was sampled on TIM6_TRIGA port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	
b15~b12	Reserved	-	Read as "0", write as "0"	R/W
b11	HCDO11	Hardware count down condition	Condition: Internal hardware trigger event 3 is valid 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W
b10	HCDO10	Hardware count down condition 10	Condition: Internal hardware trigger event 2 is valid 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W

b9	HCDO9	Hardware count down condition 9	Condition: Internal hardware trigger event 1 is valid 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W
b8	HCDO8	Hardware count down condition 8	Condition: Internal hardware trigger event 0 is valid 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W
b7	HCDO7	Hardware count down condition 7	Condition: When TIM6_<t>_PWMB port is high, a falling edge was sampled on TIM6_<t>_PWMA port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W
b6	HCDO6	Hardware count down condition 6	Condition: When TIM6_<t>_PWMB port is high, a rising edge was sampled on TIM6_<t>_PWMA port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W
b5	HCDO5	Hardware count down condition 5	Condition: When TIM6_<t>_PWMB port is low, a falling edge was sampled on TIM6_<t>_PWMA port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W
b4	HCDO4	Hardware count down condition 4	Condition: When TIM6_<t>_PWMB port is low, a rising edge was sampled on TIM6_<t>_PWMA port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W
b3	HCDO3	Hardware count down condition 3	Condition: When TIM6_<t>_PWMA port is high, a falling edge was sampled on TIM6_<t>_PWMB port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W
b2	HCDO2	Hardware count down condition 2	Condition: When TIM6_<t>_PWMA port is high, a rising edge was sampled on TIM6_<t>_PWMB port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W
b1	HCDO1	Hardware count down condition 1	Condition: When TIM6_<t>_PWMA port is low, a falling edge was sampled on TIM6_<t>_PWMB port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W
b0	HCDO0	Hardware count down condition 0	Condition: When TIM6_<t>_PWMA port is low, a rising edge was sampled on TIM6_<t>_PWMB port 0: Hardware count down is disabled when condition matches 1: Hardware count down is enabled when condition matches	R/W

20.5.25 Hardware Trigger Event Select Register (TMR6_HTSSRm) (m=0~3)

Reset value: 0x000001FFh

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]							

Bit	Symbol	Bit name	Description	Read and write
b31	COMEN[1]	Common trigger enable 1	0: Disable common trigger event of AOS_COMTRG2 to trigger Timer6 1: Enable common trigger event of AOS_COMTRG2 to trigger Timer6 Refer to the Automatic Operating System (AOS) chapter for details	R/W
b30	COMEN[0]	Common trigger enable 0	0: Disable common trigger event of AOS_COMTRG1 to trigger Timer6 1: Enable common trigger event of AOS_COMTRG1 to trigger Timer6 Refer to the Automatic Operating System (AOS) chapter for details	R/W
b29~b9	Reserved	-	Read as "0", write as "0"	R/W
b8~b0	TRGSEL	Trigger source selection	To write the trigger source number, refer to [Interrupt controller (INTC)] chapter	R/W

20.5.26 Software Sync Start Control Register (TMR6_SSTAR)

Reset value: 0x00000000h

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								SSTA 8	SSTA 7	SSTA 6	SSTA 5	SSTA 4	SSTA 3	SSTA 2	SSTA 1
Bit	Symbol	Bit name		Description								Read and write			
b31~b8	Reserved	-		Read as "0", write as "0"								R/W			
b7	SSTA8	Unit 8 Software Start		0: Software start is disabled 1: Software start is enabled								R/W			
b6	SSTA7	Unit 7 Software Start		0: Software start is disabled 1: Software start is enabled								R/W			
b5	SSTA6	Unit 6 Software Start		0: Software start is disabled 1: Software start is enabled								R/W			
b4	SSTA5	Unit 5 Software Start		0: Software start is disabled 1: Software start is enabled								R/W			
b3	SSTA4	Unit 4 Software Start		0: Software start is disabled 1: Software start is enabled								R/W			
b2	SSTA3	Unit 3 Software Start		0: Software start is disabled 1: Software start is enabled								R/W			
b1	SSTA2	Unit 2 Software Start		0: Software start is disabled 1: Software start is enabled								R/W			
b0	SSTA1	Unit 1 Software Start		0: Software start is disabled 1: Software start is enabled								R/W			

20.5.27 Software Sync Stop Control Register (TMR6_SSTPR)

Reset value: 0x00000000h

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								SSTP 8	SSTP 7	SSTP 6	SSTP 5	SSTP 4	SSTP 3	SSTP 2	SSTP 1
Bit	Symbol	Bit name		Description								Read and write			
b31~b8	Reserved	-		Read as "0", write as "0"								R/W			
b7	SSTP8	Unit 8 software stop		0: Software stop is disabled 1: Software stop is enabled								R/W			
b6	SSTP7	Unit 7 software stop		0: Software stop is disabled 1: Software stop is enabled								R/W			
b5	SSTP6	Unit 6 software stop		0: Software stop is disabled 1: Software stop is enabled								R/W			
b4	SSTP5	Unit 5 software stop		0: Software stop is disabled 1: Software stop is enabled								R/W			
b3	SSTP4	Unit 4 software stop		0: Software stop is disabled 1: Software stop is enabled								R/W			
b2	SSTP3	Unit 3 software stop		0: Software stop is disabled 1: Software stop is enabled								R/W			
b1	SSTP2	Unit 2 software stop		0: Software stop is disabled 1: Software stop is enabled								R/W			
b0	SSTP1	Unit 1 software stop		0: Software stop is disabled 1: Software stop is enabled								R/W			

20.5.28 Software Sync Clear Control Register (TMR6_SCLRR)

Reset value: 0x00000000h

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								SCLE 8	SCLE 7	SCLE 6	SCLE 5	SCLE 4	SCLE 3	SCLE 2	SCLE 1
Bit	Symbol	Bit name	Description								Read and write				
b31~b8	Reserved	-	Read as "0", write as "0"								R/W				
b7	SCLE8	Unit 8 software clear	0: Software clear is disabled 1: Software clear is enabled								R/W				
b6	SCLE7	Unit 7 software clear	0: Software clear is disabled 1: Software clear is enabled								R/W				
b5	SCLE6	Unit 6 software clear	0: Software clear is disabled 1: Software clear is enabled								R/W				
b4	SCLE5	Unit 5 software clear	0: Software clear is disabled 1: Software clear is enabled								R/W				
b3	SCLE4	Unit 4 software clear	0: Software clear is disabled 1: Software clear is enabled								R/W				
b2	SCLE3	Unit 3 software clear	0: Software clear is disabled 1: Software clear is enabled								R/W				
b1	SCLE2	Unit 2 software clear	0: Software clear is disabled 1: Software clear is enabled								R/W				
b0	SCLE1	Unit 1 software clear	0: Software clear is disabled 1: Software clear is enabled								R/W				

20.5.29 Software Sync Update Control Register (TMR6_SUPDR)

Reset value: 0x00000000h

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								SUPD 8	SUPD 7	SUPD 6	SUPD 5	SUPD 4	SUPD 3	SUPD 2	SUPD 1
Bit	Symbol	Bit name			Description						Read and write				
b31~b8	Reserved	-			Read as "0", write as "0"						R/W				
b7	SUPD8	Unit 8 software update			0: Software update is disabled 1: Software update is enabled						R/W				
b6	SUPD7	Unit 7 software update			0: Software update is disabled 1: Software update is enabled						R/W				
b5	SUPD6	Unit 6 software update			0: Software update is disabled 1: Software update is enabled						R/W				
b4	SUPD5	Unit 5 software update			0: Software update is disabled 1: Software update is enabled						R/W				
b3	SUPD4	Unit 4 software update			0: Software update is disabled 1: Software update is enabled						R/W				
b2	SUPD3	Unit 3 software update			0: Software update is disabled 1: Software update is enabled						R/W				
b1	SUPD2	Unit 2 software update			0: Software update is disabled 1: Software update is enabled						R/W				
b0	SUPD1	Unit 1 software update			0: Software update is disabled 1: Software update is enabled						R/W				

20.6 Notice for use

1. CPU cannot write to the CNTER register when the counter is counting, it can write to the CNTER register only when the counter stops.
2. When the corresponding buffer function of GCMCR~GCMFR is enabled, its interrupt output and event output are invalid.
3. When using the bilateral symmetrical complementary PWM output function, the user should calculate and set the relevant reference values and port output levels to ensure that the ports output the expected levels.
4. In some specific occasions, the control events of the counter may happen at the same time. At this time, the control priority to the counter (CNTER) change is shown in Table 20-5.

Table 20-5 Counter (CNTER) control priority

Timer control event	Counter (CTER) value	Priority
CPU writes to CNTER	Written value by CPU	high
Update event (software update or hardware update)	Value in UPAR	↓
Clear event (software clear or hardware clear)	In hardware counting mode or sawtooth waveform mode	Change to 0 or PERAR depending on the count direction
	In triangular wave mode	0
count by hardware event	normal count value	↓
count in triangular wave mode	normal count value	↓
count in sawtooth waveform mode	normal count value	Low

5. In some specific occasions, the control events of the PWM port output may happen at the same time. At this time, the control priority to output change of the two ports (TIM6_<t>_PWMA and TIM6_<t>_PWMB) is shown in Table 20-6, Table 20-7.

Table 20-6 PWMA port output control priority

PWMA port output control event	Port level	Priority
Event selected by EMB occurs	Level set by PCNAR.EMBCA	high
Set forced port output	Level set by PCNAR.FORCA	↓
When the count starts	Level set by PCNAR.STACA	↓
When the sawtooth waveform counts up	CNTER=PERAR	Level set by PCNAR.OVFCA
	CNTER=GCMAR	Level set by PCNAR.CMAUCA
	CNTER=GCMBR	Level set by PCNAR.CMBUCA
When the sawtooth waveform counts down	CNTER=0	Level set by PCNAR.UDFCA
	CNTER=GCMAR	Level set by PCNAR.CMADCA
	CNTER=GCMBR	Level set by PCNAR.CMBDCA
When the triangular wave counts up	CNTER=GCMAR	Level set by PCNAR.CMAUCA
	CNTER=GCMBR	Level set by PCNAR.CMBUCA
	CNTER=PERAR	Level set by PCNAR.OVFCA
When the triangular wave counts down	CNTER=GCMAR	Level set by PCNAR.CMADCA
	CNTER=GCMBR	Level set by PCNAR.CMBDCA
	CNTER=0	Level set by PCNAR.UDFCA
When count stops	Level set by PCNAR.STPCA	Low

Table 20-7 PWMB port output control priority

PWMB port output control event	Port level	Priority
Event selected by EMB occurs	Level set by PCNBR.EMBCB	high
Set forced port output	Level set by PCNBR.FORCB	↓
When the count starts	Level set by PCNBR.STACB	↓
When the sawtooth waveform counts up	CNTER=PERAR	Level set by PCNBR.OVFCB
	CNTER=GCMBR	Level set by PCNBR.CMBUCB
	CNTER=GCMAR	Level set by PCNBR.CMAUCB
When the sawtooth waveform counts down	CNTER=0	Level set by PCNBR.UDFCB
	CNTER=GCMBR	Level set by PCNBR.CMBDCB
	CNTER=GCMAR	Level set by PCNBR.CMADCB
When the triangular wave counts up	CNTER=GCMBR	Level set by PCNBR.CMBUCB
	CNTER=GCMAR	Level set by PCNBR.CMAUCB
	CNTER=PERAR	Level set by PCNBR.OVFCB
When the triangular wave counts down	CNTER=GCMBR	Level set by PCNBR.CMBDCB
	CNTER=GCMAR	Level set by PCNBR.CMADCB
	CNTER=0	Level set by PCNBR.UDFCB
When count stops	Level set by PCNBR.STPCB	Low

21 High Resolution PWM (HRPWM)

21.1 Introduction

High Resolution PWM (HRPWM) extends the resolution of Timer6's PWM signal. This module can generate up to 16 channels of high-resolution PWM waveforms, paired with TMR6.

The HRPWM has the following main features:

- Extended PWM signal resolution
- Can be used for adjusting PWM's duty cycle and phase
- Can be used for controlling the rising edge, falling edge, rising and falling edge
- Auto-calibration function, which can provide the unit delay

21.2 Basic block diagram

The block diagram of HRPWM is shown in Figure 21-1.

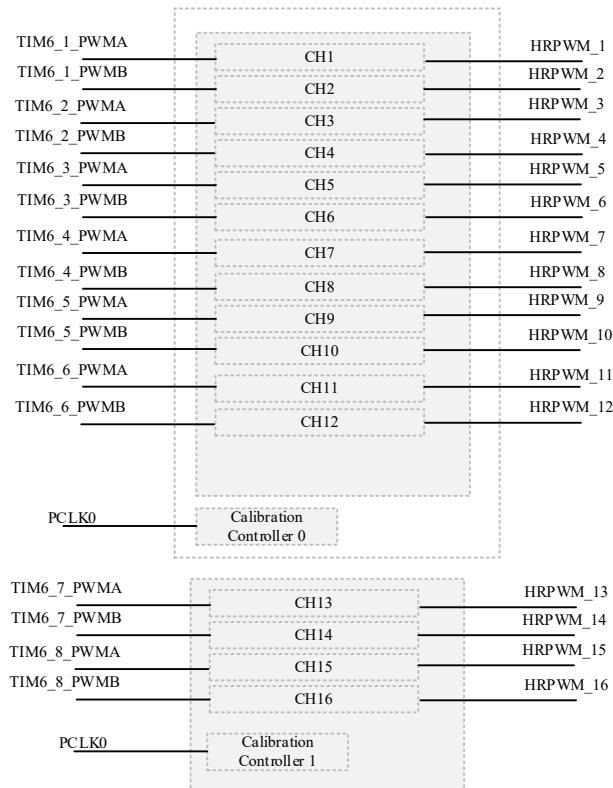


Figure 21-1 Basic block diagram of HRPWM

21.3 Functional description

21.3.1 Calibration function

This module contains two calibration units, among which calibration controller 0 is used for the unit delay calibration of channel 1- 12, and calibration controller 1 is used for the unit delay calibration of channel 13- 16. The calibration controllers are independent of channels 1-16, so the calibration process does not affect the function of channels 1-16. A calibration control uint generates a calibration code which indicates how many unit delays one PCLK0 can be divided into.

Calibration module usage process ($n=0/1$):

1. Configure and enable the PCLK0
2. Set HRPWM_CALCRn.CALEN to 1
3. Read HRPWM_CALCRn.ENDF until the result is 1
4. Read the value in HRPWM_CALCRn.CALCODE[7:0]

21.3.2 High Resolution PWM adjustment function

HRPWM can be configured by software to set the rising edge, falling edge, rising and falling edge adjustment of PWM signal respectively. The minimum adjustment scale is determined by the unit delay. The delay of the rising edge and the falling edge can be configured separately by software. The minimum delay is the unit delay, and the maximum is 256 times the minimum.

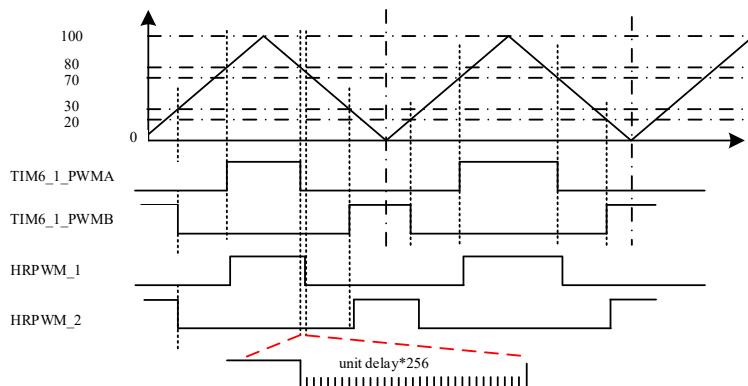


Figure 21-2 HRPWM adjustment waveform

Initial configuration HRPWM module process ($n=1-16$):

1. Perform the calibration according to usage process of the calibration function in 21.3.1, and get the calibration code
2. Configure HRPWM_CHnCR.NSEL to change the delay adjustment of the falling edge
3. Configure HRPWM_CHnCR.PSEL to change the delay adjustment of the rising edge
4. Configure HRPWM_CHnCR.NE to enable falling edge adjustment

5. Configure HRPWM_CHnCR.PE to enable rising edge adjustment
6. Configure HRPWM_CHnCR.EN=1 to enable HRPWM function

21.3.3 Note

1. When the PWM waveform is continuously output, HRPWM_CRn.NSEL and HRPWM_CRn.PSEL can only be dynamically configured at the counter peak and valley points of TIM6.
2. The frequency of PCLK0 must be higher than 120MHz.
3. After the chip enters STOP mode, TIM6_n_PWMA/TIM6_n_PWMB (n=1~6), HRPWM_1~HRPWM_12 output low level and will not maintain the state before STOP mode. If the output pins corresponding to these functions need to maintain the state before the STOP mode, the function selection register PFSR of the pin needs to be configured as a general GPIO function, and the general control register PCR of the pin needs to be set to the corresponding state.

21.4 Register description

Table 21-1 Shown is the register list of HRPWM module.

BASE ADDR: 0x4003C000 (HRPWM_BASE_ADDR)

Table 21-1 Register list

Register name	Symbol	Offset address	Bit width	Reset value
HRPWM Unit 1 Control Register	HRPWM_CR1	0x00	32	0x00000000h
HRPWM Unit 2 Control Register	HRPWM_CR2	0x04	32	0x00000000h
HRPWM Unit 3 Control Register	HRPWM_CR3	0x08	32	0x00000000h
HRPWM Unit 4 Control Register	HRPWM_CR4	0x0c	32	0x00000000h
HRPWM Unit 5 Control Register	HRPWM_CR5	0x10	32	0x00000000h
HRPWM Unit 6 Control Register	HRPWM_CR6	0x14	32	0x00000000h
HRPWM Unit 7 Control Register	HRPWM_CR7	0x18	32	0x00000000h
HRPWM Unit 8 Control Register	HRPWM_CR8	0x1c	32	0x00000000h
HRPWM Unit 9 Control Register	HRPWM_CR9	0x20	32	0x00000000h
HRPWM Unit 10 Control Register	HRPWM_CR10	0x24	32	0x00000000h
HRPWM Unit 11 Control Register	HRPWM_CR11	0x28	32	0x00000000h
HRPWM Unit 12 Control Register	HRPWM_CR12	0x2c	32	0x00000000h
HRPWM Unit 13 Control Register	HRPWM_CR13	0x30	32	0x00000000h
HRPWM Unit 14 Control Register	HRPWM_CR14	0x34	32	0x00000000h
HRPWM Unit 15 Control Register	HRPWM_CR15	0x38	32	0x00000000h
HRPWM Unit 16 Control Register	HRPWM_CR16	0x3c	32	0x00000000h
HRPWM Calibration Control Register 0	HRPWM_CALCR0	0x50	32	0x00000000h
HRPWM Calibration Control Register 1	HRPWM_CALCR1	0x54	32	0x00000000h

21.4.1 HRPWM Control Register (HRPWM_CRn, n=1...16)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EN	PE	NE							-						
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PSEL[7:0]								NSEL[7:0]							
Bit	Symbol	Bit name		Description								Read and write			
b31	EN	High Resolution PWM Enable		0: High-resolution PWM is invalid, the PWM signal is pass-through 1: High resolution PWM enable								R/W			
b30	PE	HRPWM rising edge adjustment enable		0: Disable the HRPWM to adjust rising edge 1: Enable the HRPWM to adjust rising edge								R/W			
b29	NE	HRPWM falling edge adjustment enable		0 : Disable the HRPWM to adjust the falling edge 1: Enable the HRPWM to adjust the falling edge								R/W			
b28-b16	Reserved	-		Read as 0, write 0 when writing								R/W			
b15-b8	PSEL[7:0]	HRPWM rising edge calibration selection		0x00h: Select a delay uint 0x01h: Select two delay uints 0xfeh: Select 255 delay uints 0xffh: Select 256 delay uints								R/W			
b7-b0	NSEL[7:0]	HRPWM falling edge calibration selection		0x00h: Select a delay uint 0x01h: Select two delay uints 0xfeh: Select 255 delay uints 0xffh: Select 256 delay uints								R/W			

21.4.2 HRPWM Calibration Control Register 0 (HRPWM_CALCRn, n=0,1)

Reset value: 0x00000000h

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
CALEN	-	ENDF	-	-	-	-	-	-	-	-	-	-	-	-	CALCODE[7:0]

Bit	Symbol	Bit name	Description	Read and write
b31-b16	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b15	CALEN	Calibration enable	0: Disable the calibration function 1: Enable the calibration function	R/W
b14-b13	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b12	ENDF	Calibration End Flag	0: Calibrating 1: Calibration completed	R
b11-b8	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b7-b0	CALCODE	Calibration value	The result of automatic calibration indicates how many delay units the calibration pclk consists of.	R/W

CALCR0 is the channel 1 to channel 12 calibration use register.

CALCR1 is the channel 13-channel 16 calibration use register.

22 General control timer (Timer4)

22.1 Introduction

General control timer 4 (Timer4) is a timer module for three-phase motor control. It provides three-phase motor control solutions for various applications. The timer supports triangular wave and sawtooth wave modes and generates various PWM waveforms. Supporting buffers function and EMB control. This series of products carry three units of Timer4.

22.2 Basic block diagram

The basic functions and features of Timer4 are shown in Table 22-1.

Table 22-1 Basic functions and features

Waveform mode	Sawtooth wave, triangular wave
Basic functions	• Direction of increment and decrement
	• Buffers function
	• General PWM output
	• Special event output to start ADC conversion
	• EMB control
Interrupt type	Count compare match interrupt
	Count period match interrupt
	Reload count match interrupt

In Figure 22-1, the basic architecture of the general control timer Timer4 is described. The "<t>" shown in the block diagram represents the unit number, that is, "<t>" is 1~3. When referring to "<t>" later in this chapter, it refers to the unit number and this will not be repeated; the PCLK shown in the block diagram refers to PCLK0, that is, the internal count clock of Timer4 is PCLK0.

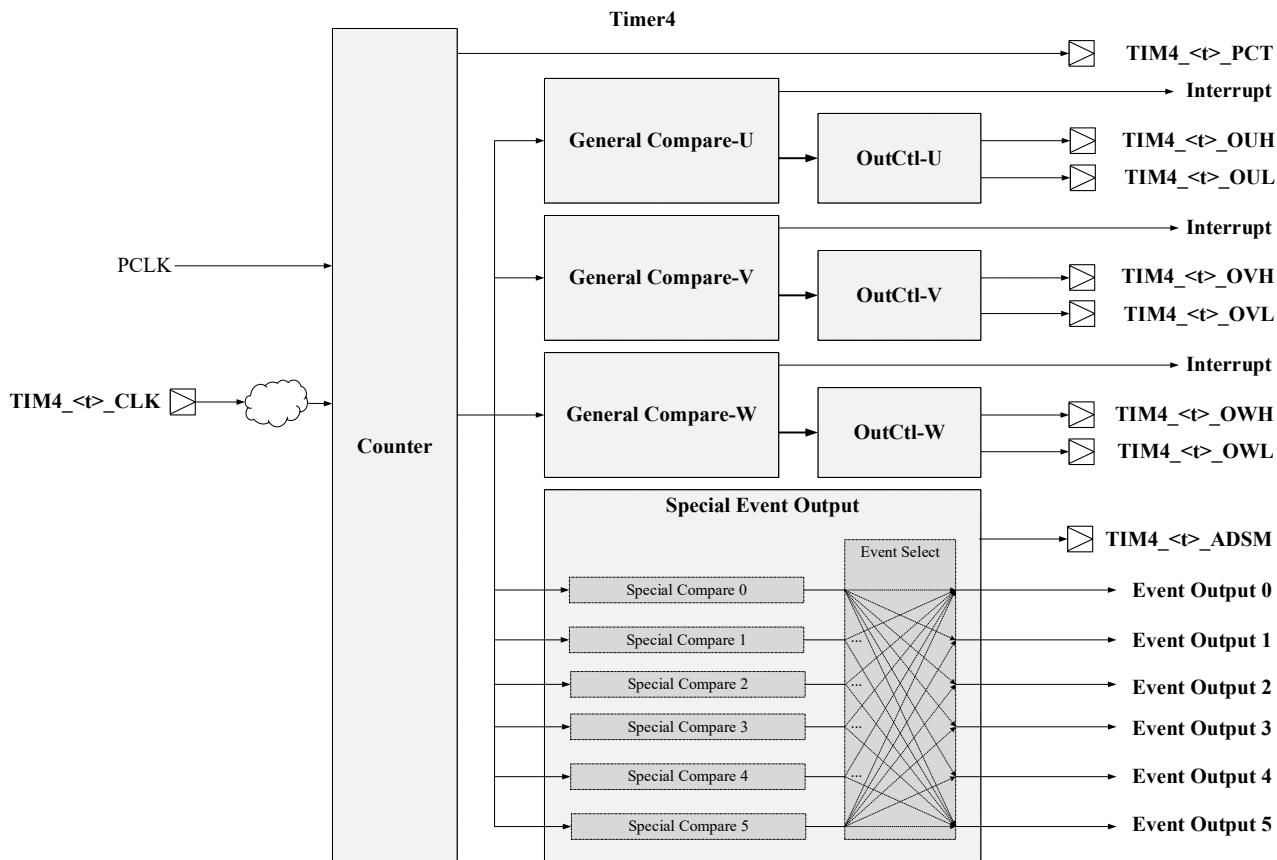


Figure 22-1 Basic block diagram of Timer4

Table 22-2 is the list of input and output ports of Timer4.

Table 22-2 Timer4 port list

Port name	Direction	Function
TIM4_<t>_CLK	in	Count clock input port
TIM4_<t>_OUH		
TIM4_<t>_OUL		
TIM4_<t>_OVH		
TIM4_<t>_OVL		
TIM4_<t>_OWH		
TIM4_<t>_OWL		
TIM4_<t>_PCT	out	PWM period output monitor
TIM4_<t>_ADSM	out	Special event output monitor

22.3 Function description

22.3.1 Waveform mode

Timer4 has two basic count wave modes: sawtooth wave mode and triangular wave mode. According to CCSR.MODE, two waveform modes can be implemented. Figure 22-2 and Figure 22-3 are the waveform diagrams of sawtooth wave and triangular wave respectively.

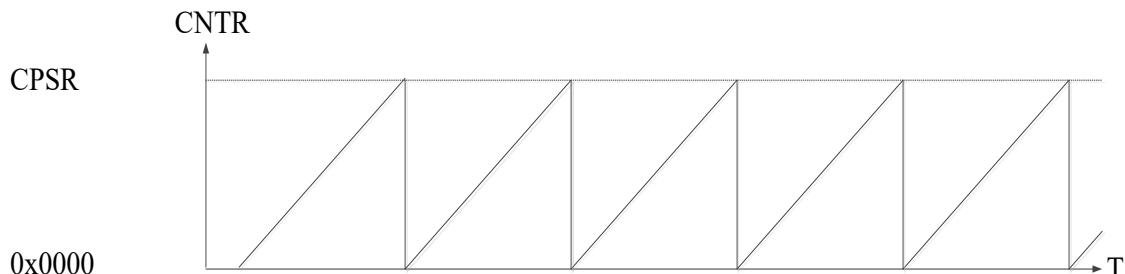


Figure 22-2 Timer4 sawtooth waveform

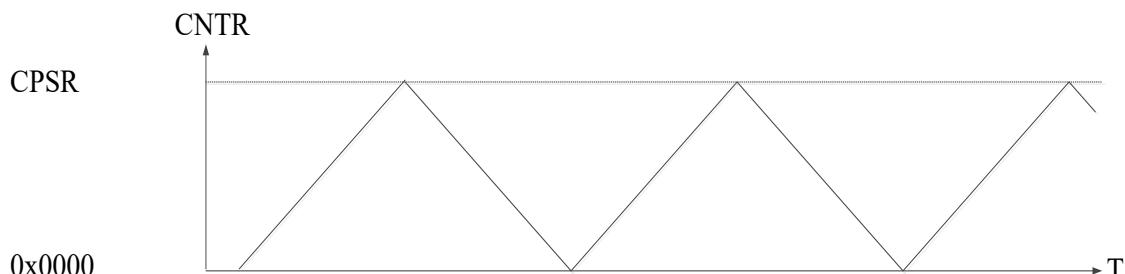


Figure 22-3 Timer4 triangular waveform

22.3.2 Count action

1. Sawtooth wave count operation and control process are shown as Figure 22-4.

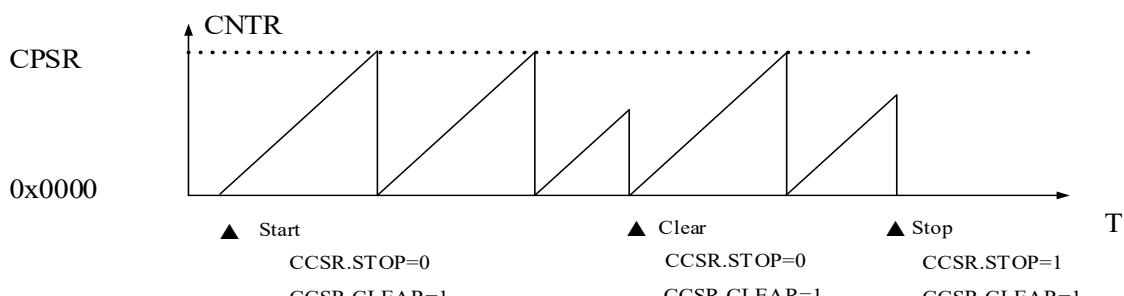


Figure 22-4 Timer4 sawtooth wave mode count action

- 1) Set mode CCSR.MODE=0.
- 2) Set the count peak in CPSR register.
- 3) Write CCSR.STOP = 0 and CCSR.CLEAR = 1, and the counter value (CNTR) is initialized to 0x0000 and starts up-count operation from 0x0000. After reach to the peak value(CPSR), the counter value returns to 0x0000. Then the operation is repeated in turn.

- 4) Count period = $(CPSR + 1) \times \text{Count clock period}$
 - 5) During counting, write CCSR.STOP = 0 and CCSR.CLEAR = 1 to initialize the counter value to 0x0000 and continue the count operation.
 - 6) During counting, write CCSR.STOP = 1 and CCSR.CLEAR = 1 to initialize the counter value to 0x0000 and stop the count operation.
2. Triangular wave count operation and control process are shown as Figure 22-5.

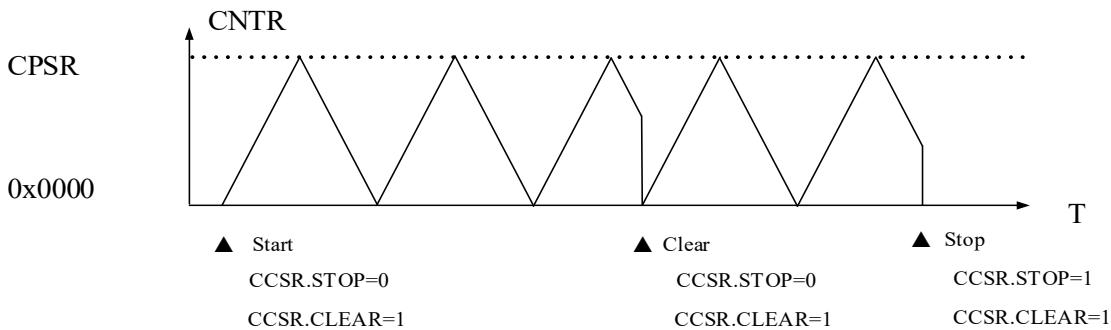


Figure 22-5 Timer4 triangle wave mode counting action

- 1) Set mode CCSR.MODE = 1.
- 2) Set the count peak in CPSR register.
- 3) Write CCSR.STOP = 0 and CCSR.CLEAR = 1, and the counter value (CNTR) is initialized to 0x0000 and starts the count operation. The up-count is performed from 0x0000 until the peak value(CPSR), then down-count is performed from the peak value until 0x0000.Following this, the count operation is repeated in turn.
- 4) Count period = $(CPSR) \times 2 \times \text{Count clock period}$
- 5) During counting, write CCSR.STOP=0 and CCSR.CLEAR=1 to initialize the counter value to 0x0000 and perform the up-count operation again, and then repeat the above operations.
- 6) During counting, write CCSR.STOP = 1 and CCSR.CLEAR = 1 to initialize the counter value to 0x0000 and stop the count operation.

22.3.3 Compare output

- Figure 22-6 is an example of the waveform output of the compare output module in the sawtooth wave mode.

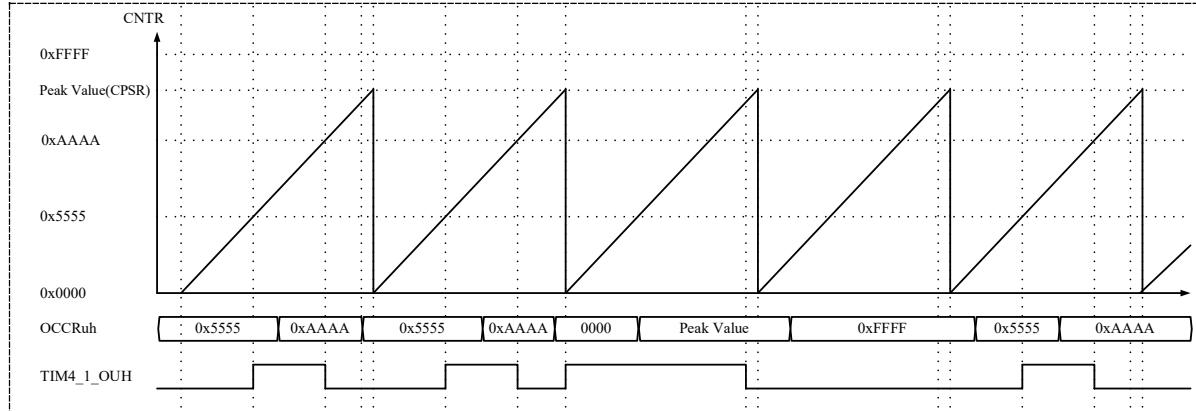


Figure 22-6 Sawtooth wave mode waveform output example

- Figure 22-7 is an example of the waveform output of the compare output module in the triangular wave mode.

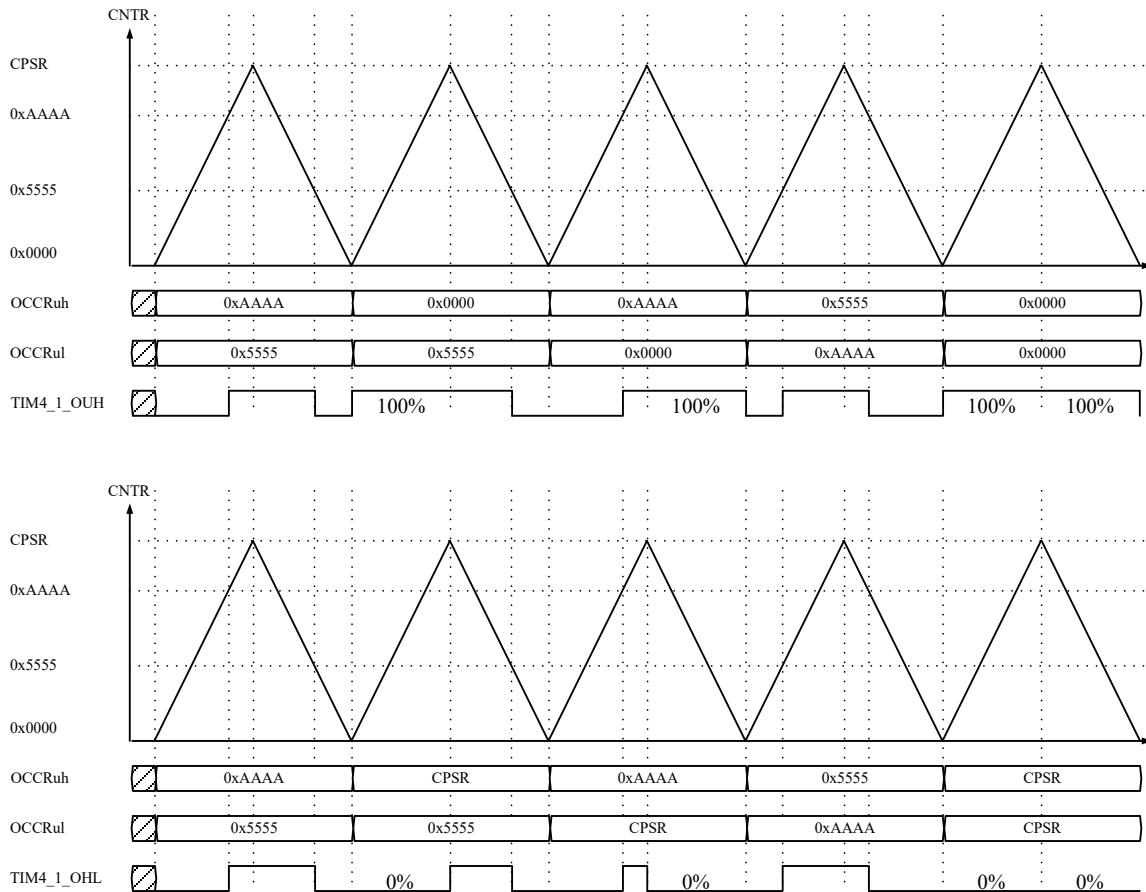


Figure 22-7 Triangular wave mode waveform output example

22.3.4 Buffer function

Timer4's Period Value Register (CPSR), General Compare Value Register (OCCR), General Gode Control Register (OCMR), Special Compare Value Register (SCCR) and Special Mode Control Register (SCMR) all have buffer function.

22.3.4.1 Period value register buffer function

CPSR has a buffer function register, and the written count peak data is first stored in the buffer register. Data is transferred from the buffer register to the CPSR register under the following conditions:

1. If buffer function is disabled (CCSR.BUFEN = 0), the written data are immediately transferred from buffer register to CPSR register.
2. If buffer function is enabled (CCSR.BUFEN = 1), the written data is transferred from buffer register to CPSR register when the counter stops (CCSR.STOP = 1), or when the counter count value is "0x0000".

Note:

- When reading data from CPSR, it 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 completes is not the most recently written value, but the last written CPSR value.

Figure 22-8 shows the operation of modifying the count peak CPSR when the buffer function is disabled in the sawtooth wave mode.

CCSR.BUFEN=0

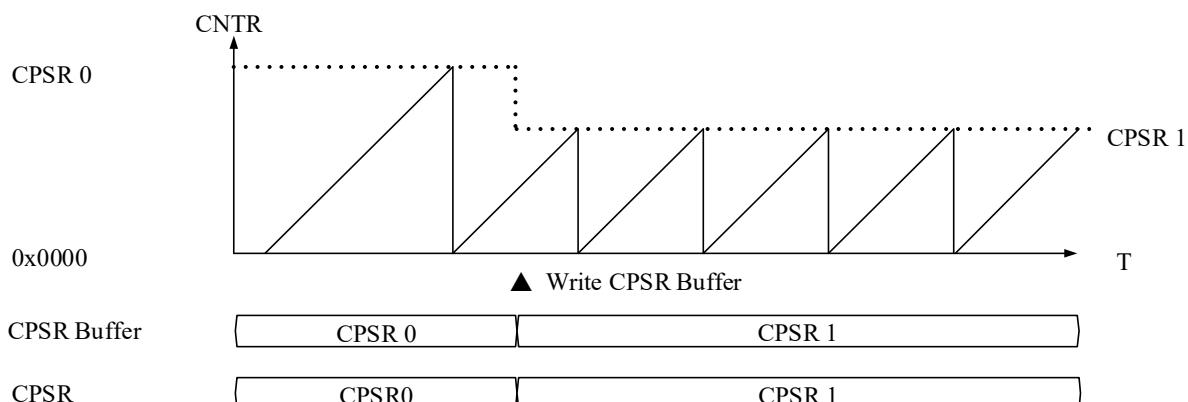


Figure 22-8 Modify the sawtooth count period when the buffer function is disabled

Note:

- When the buffer function is disabled, the written data is immediately transferred from buffer register to CPSR register, and the count period changes immediately after the write operation is completed. In this case, if the written value is less than the current count value, the counter continues to count up until it reaches "0xFFFF", which requires special attention.

Figure 22-9 shows the operation of modifying the count peak CPSR when the buffer function is enabled in the sawtooth wave mode .

CCSR.BUFEN=1

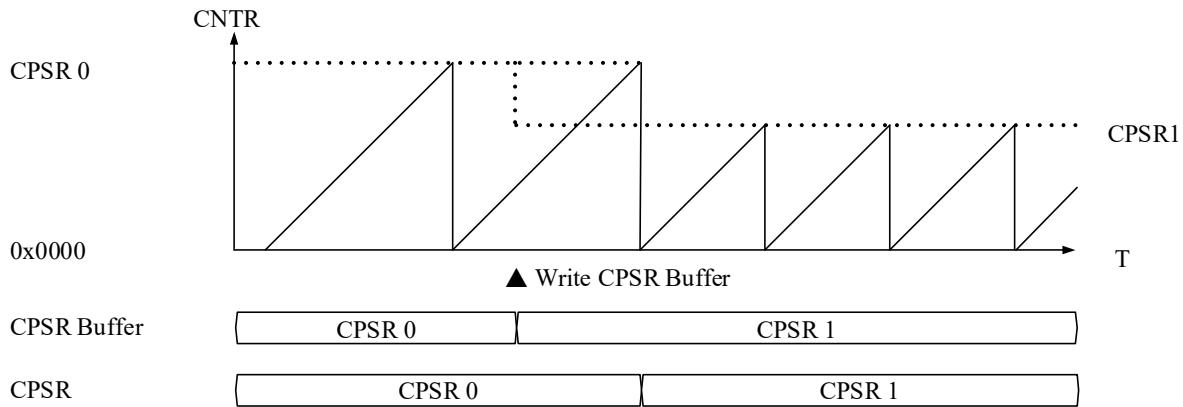


Figure 22-9 Modify the sawtooth count period when the buffer function is enabled

As shown in the figure, when the buffer function is enabled, the written data is transferred from the buffer register to CPSR register when the counter stops or when the counter count value is "0x0000".

Figure 22-10 shows the operation of modifying the count peak CPSR when the buffer function is enabled in the triangular wave mode.

CCSR.BUFEN=1

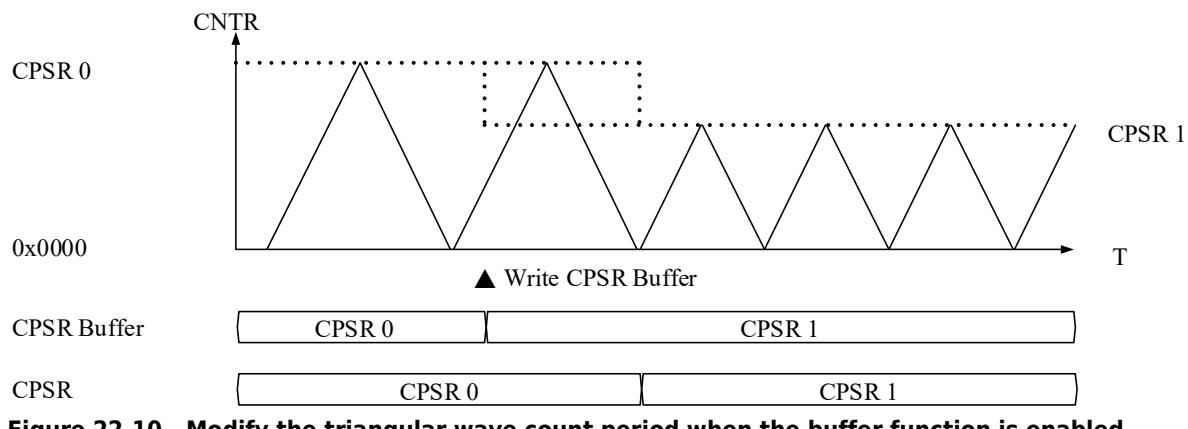


Figure 22-10 Modify the triangular wave count period when the buffer function is enabled

As shown in the figure, in the triangular wave mode, when the buffer function is enabled, the written data is transferred from the buffer register to CPSR register when the counter stops or the next counter value is "0x0000". The change of the counter period begins at the next counter period after the write operation completes.

22.3.4.2 General compare register buffer function

Both the General Compare Value Register (OCCR) and the General Mode Control Register (OCMR) have the buffer function. When the buffer function is enabled, the data are loaded to the OCCR and OCMR register at the specified transfer time. During counting, when the counter is at count peak

or bottom point, the OCCR buffer function can be used to change the compare value synchronously and the OCMR buffer function can be used to change the internal PWM output synchronously.

- When the period interval response link function is disabled, the buffer value is loaded to register at the specified count state. The load at this time is independent of the (overflow/underflow) interrupt mask counter.

Figure 22-11 is the waveform when the OCCR buffer function is enabled, the buffer value is loaded at counter zero value (OCER.CxBUFEN=01, x=L or H), and the period interval response link is disabled (OCER.LMCx=0, x=L or H).

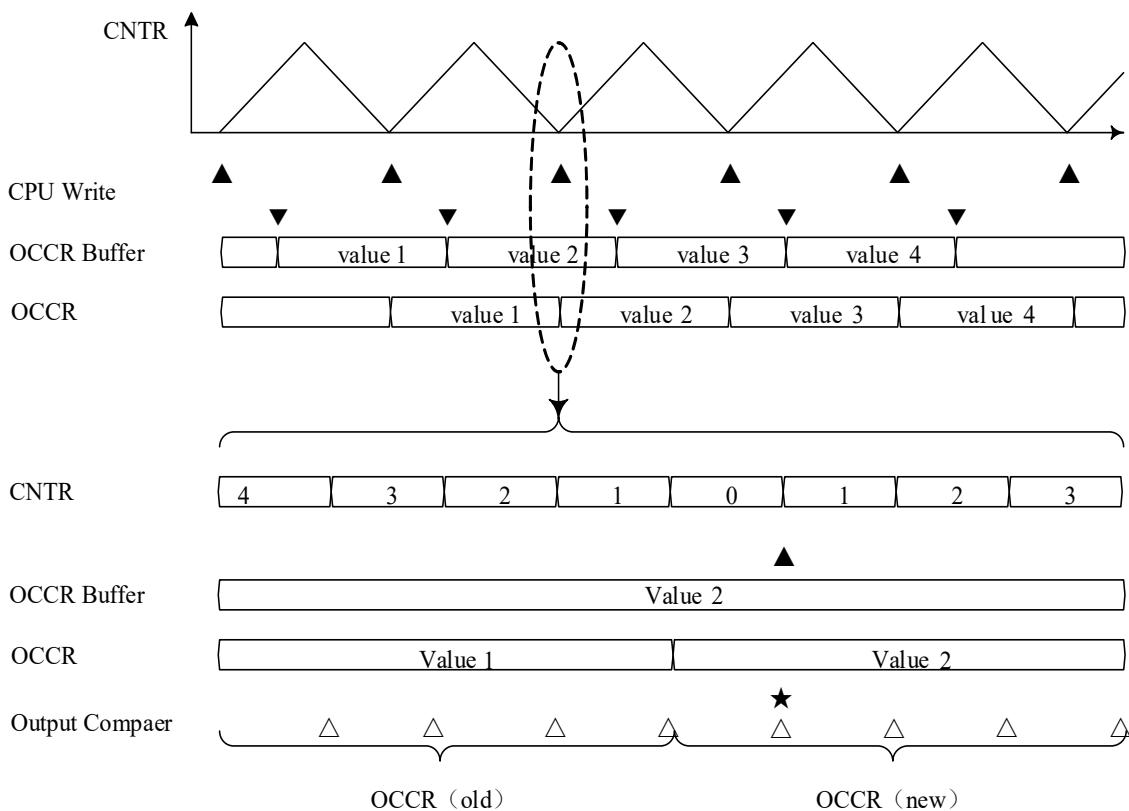


Figure 22-11 OCCR buffer data transfer (when the period interval response link is disabled)

The upper half of the diagram is a global diagram, and the lower half is an enlarged diagram during the transfer operation.

The counter is in triangular wave mode, and underflow interrupt occurs at the time of mark ▲. At the time of mark ▼, the CPU rewrites the OCCR register and the written data stored in the OCCR buffer register. After that, when the counter value is 0x0000, the data is loaded from the buffer register to the OCCR register, and the interrupt flag IRQZF is generated.

At the △ moment, the output compare is performed to change PWM output and set OCSR.OCFx bit ($x = L$ or H) according to the event that the set OCCR register value matches the count value. After the ★ time (count value = 0x0000), the port output performs the operation according to the new OCCR data and performs the operation according to the original OCCR data before the ★ time.

The diagram shows the OCCR buffer register transfer operation at the count bottom point, and the OCMR buffer register transfer operation is similar. And the transfer operation at the count peak is similar too. The new data takes effect immediately after the transfer time (the new written data will control the PWM output and interrupt flag).

- When the period interval response link is enabled, the buffer register transfer operation is performed when the buffer value is at the specified count state and the (overflow or underflow) interrupt mask counter value is 0.

Figure 22-12 is the waveform when the OCCR buffer function is enabled, the buffer value is loaded at counter zero value (OCER.CxBUFEN=01, x=L or H), and the period interval response link is enabled (OCER.LMCx=1, x=L or H).

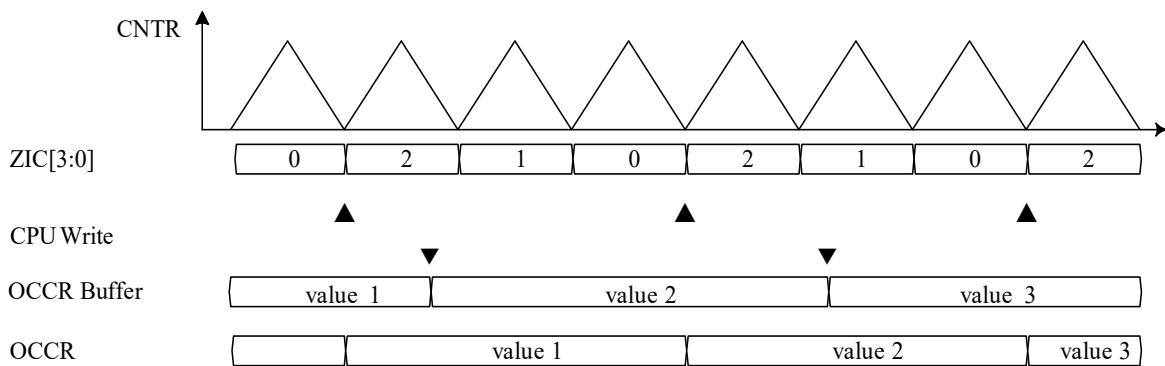


Figure 22-12 OCCR buffer data transfer (period interval response link is enabled)

The counter is in triangular wave mode, the underflow interrupt mask counter(CVPR.ZIC) is decremented from 2 to 0, and the underflow interrupt is generated at the time of the mark ▲. At the time of mark ▼, the CPU rewrites the OCCR register and the written data is stored in the OCCR buffer register. After that, when the counter value is 0x0000 and the underflow interrupt mask counter (CVPR.ZIC) is 0, the data is loaded from the buffer register to the OCCR register, and the interrupt flag IRQZF is generated.

The diagram shows the OCCR buffer register transfer operation at the count bottom point, and the OCMR buffer register transfer operation is similar. And the transfer operation at the count peak is similar too. The new data takes effect immediately after the transfer time (the new written data will control the PWM output and interrupt flag).

When using channel link operation mode, enabling both OCCRuh and OCCRul buffer function at the same time can produce various PWM output waveforms. Figure 22-13 shows the situation of changing the OCMR register value to generate different output waveforms of TIM4_<t>_OUL under the condition that the output compare registers OCCRuh and OCCRul remain unchanged.

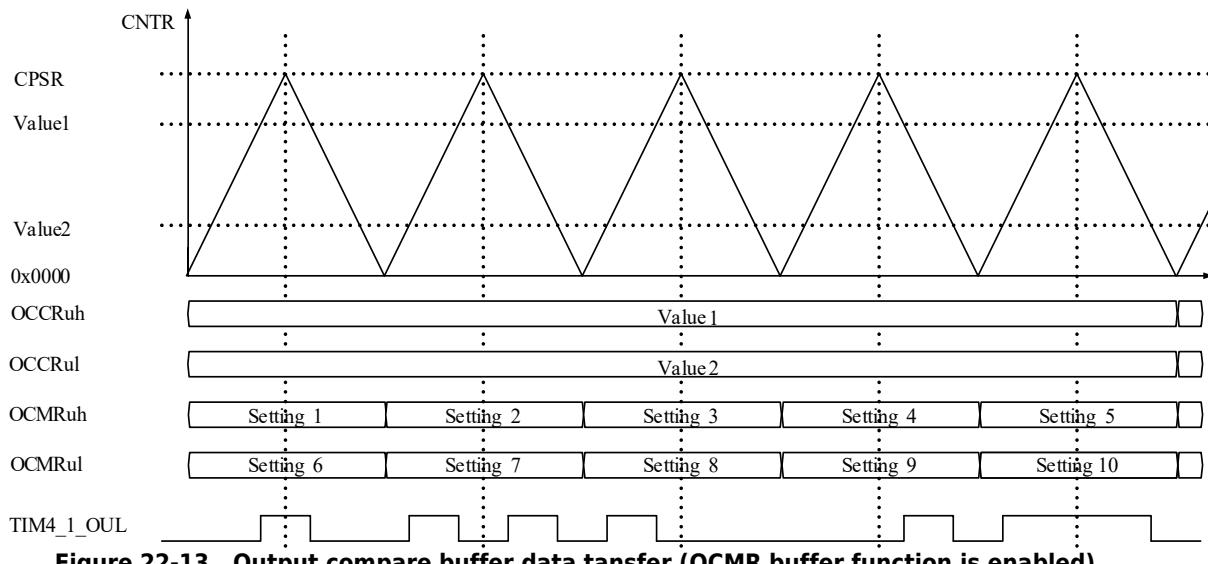


Figure 22-13 Output compare buffer data transfer (OCMR buffer function is enabled)

22.3.4.3 Special compare register buffer function

Both the Special Compare Value Register (SCCR) and the Special Mode Control Register (SCMR) have the buffer function. When the buffer function is enabled, the data written by the CPU transfers from SCCR and SCMR buffer registers to SCCR and SCMR registers at the specified count state.

- When the period interval response link is disabled, the buffer transfer operation is only related to the count state and is not affected by the (overflow/underflow) interrupt mask counter.

Figure 22-14 is the waveform when the register buffer function is enabled, the period interval response link is disabled (SCSR.LMC=0), and the buffer value is loaded to the SCCR and SCMR registers when the counter value is zero (SCSR.BUFEN=01).

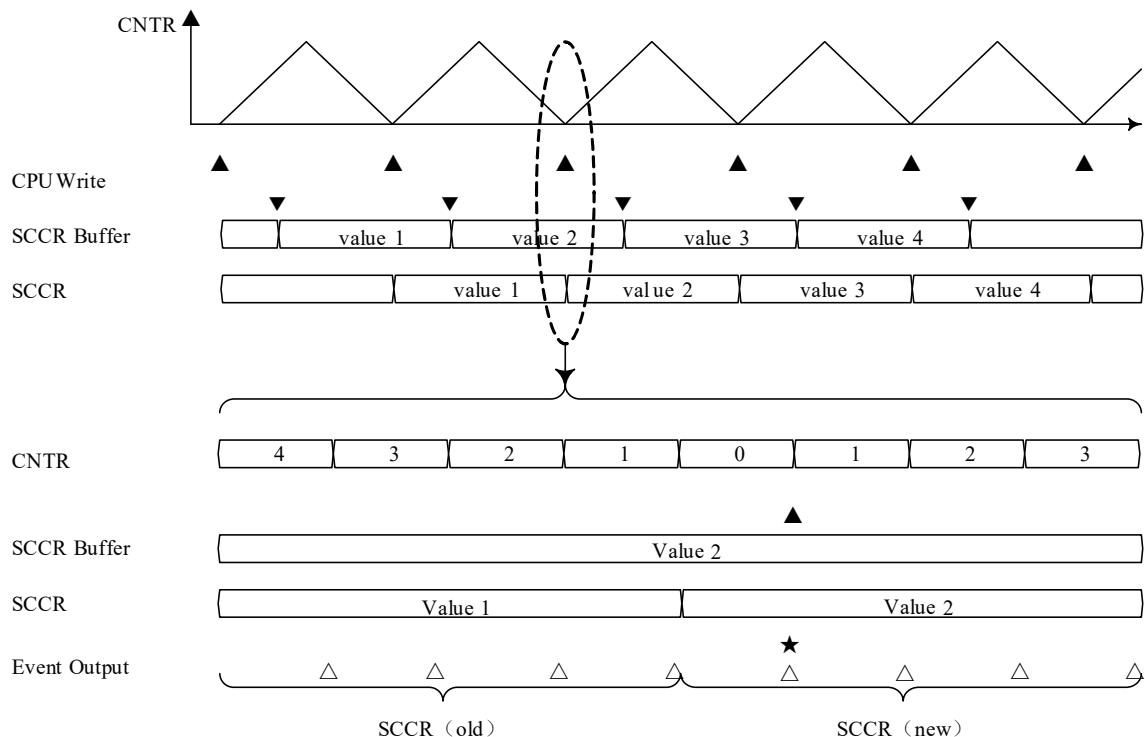


Figure 22-14 SCCR buffer transfer operation (when period interval response link is disabled)

The upper half of the diagram is a global diagram, and the lower half of the diagram is an enlarged diagram during the transfer operation.

The counter is in triangular wave count mode, and underflow interrupt is generated at the time of mark \blacktriangle . At the time of mark \blacktriangledown , the CPU rewrites the SCCR register and the written data is stored in the SCCR buffer register. After that, when the counter value is 0x0000, the data is loaded from buffer register to SCCR register, and the interrupt flag IRQZF is generated.

At the \triangle moment, the compare operation is performed according to the set SCCR register value and the counter value. After the \star time (counter value = 0x0000), the special event output performs the operation according to the new SCCR data and performs the operation according to the original SCCR data before the \star time.

The diagram shows the SCCR buffer register transfer operation at the count bottom point, and the SCMR buffer register transfer operation is similar. And the transfer operation at the count peak is similar too. The new data takes effect immediately after the transfer time (the new written data will control the special event output and interrupt flag).

- When the period interval response link is enabled, the buffer register transfer operation is performed when the buffer value is at the specified count state and the (overflow or underflow) interrupt mask counter value is 0.

Figure 22-15 is the waveform when the SCCR buffer function is enabled, the period interval response link is enabled (SCSR.LMC=1), and the buffer value is loaded to the SCCR and SCMR registers when the counter value is zero (SCSR.BUFEN=01).

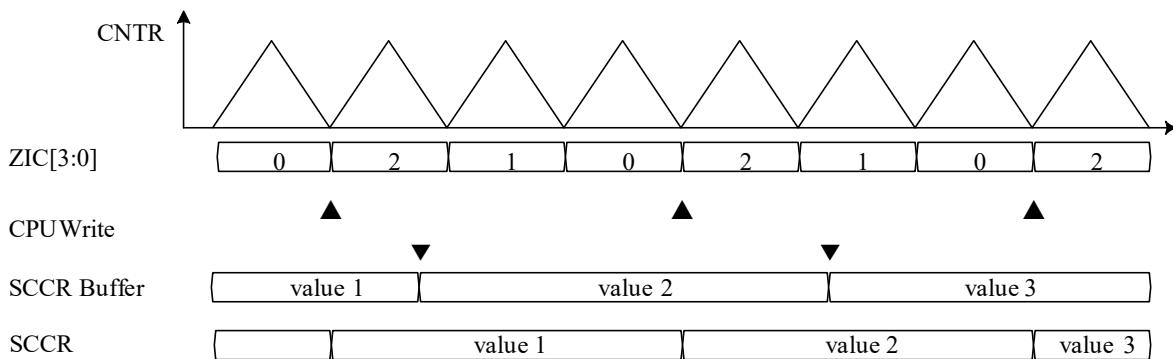


Figure 22-15 SCCR buffer transfer operation (when period interval response link is enabled)

The counter is in triangular wave mode, the underflow interrupt mask counter(CVPR.ZIC) is decremented from 2 to 0, and the underflow interrupt is generated at the time of the mark ▲. At the time of mark ▼, the CPU rewrites the SCCR register and the written data is stored in the SCCR buffer register. After that, when the counter count value is 0x0000 and the underflow interrupt mask counter (CVPR.ZIC) is 0, the data is loaded from buffer register to SCCR register, and the interrupt flag IRQZF is generated.

22.3.5 General PWM output

22.3.5.1 Independent PWM output

Various PWM output can be implemented by setting the compare value based on OCCRxh&OCCRxl, and the output states of ports based on OCMRxh&OCMRxl ($x = u, v, w$) respectively in the pass-through mode (POCR.PWMMD = 00). At this time, the PWM output of each port is independently controlled. Before using the independent PWM output mode, you need to set the MOE and OExy bits to enable the output of Timer4. For details, please refer to the PWM Status Control Register (PSCR). Figure 22-16 and Figure 22-17 are the independent PWM output examples of the sawtooth wave and triangular wave of unit 1 respectively.

Note:

- The pass-through mode refers to the internal output signal (in_opxh, in_opxl) generated by compare match of General Compare Value Register (OCCRxh, OCCRxl), is directly output to the corresponding ports (TIM4_<t>_OXH, TIM4_<t>_OXL) ($X = U, V, W, x = u, v, w$).

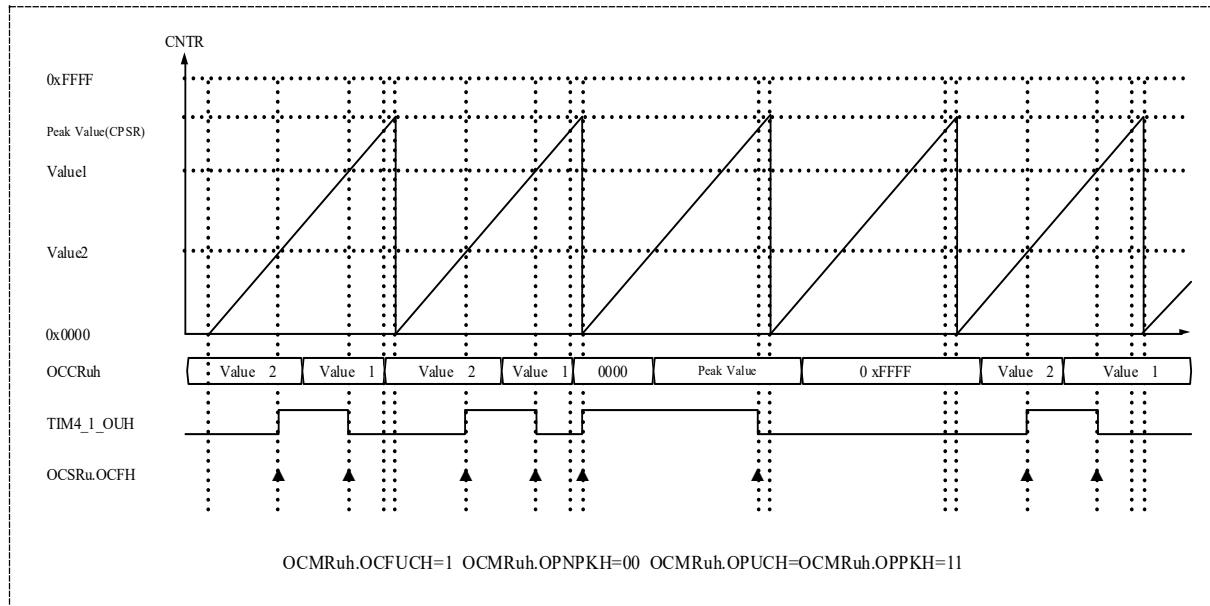


Figure 22-16 Sawtooth wave independent PWM output example

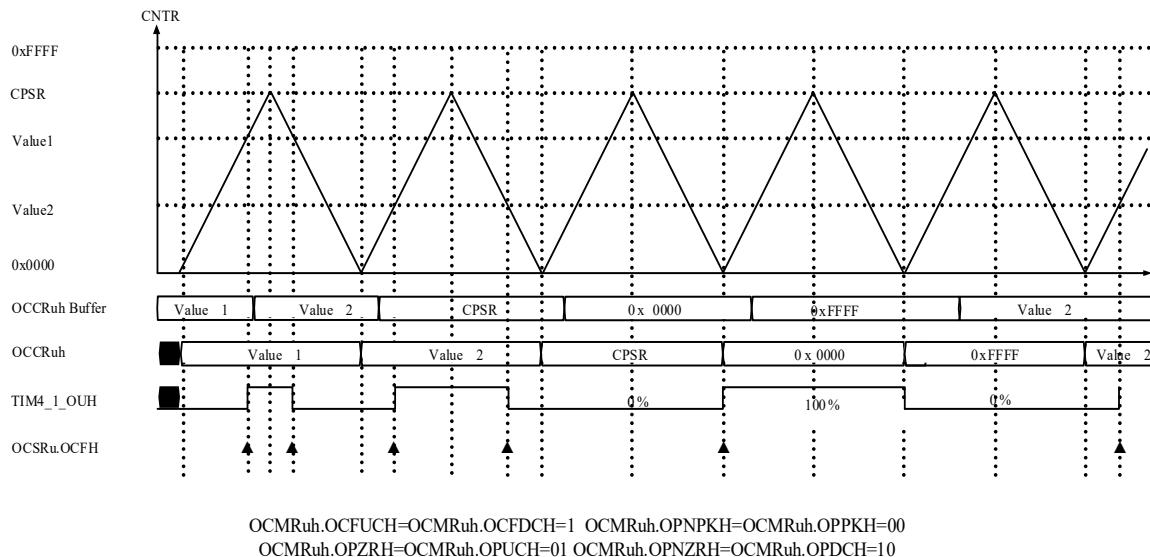


Figure 22-17 Triangular wave independent PWM output example

22.3.5.2 Extended PWM output

The output state of TIM4_<t>_OXL in the pass-through mode (POCR.PWMMD = 00) can also be set by the extended bit (bit32~16) in the OCMRxI register. The extended bit is related to the value of OCCRxh, thus implementing the extended PWM output on the TIM4_<t>_OXL port (X = U, V, W, x = u, v, w). Before using the extended PWM output mode, users need to set the MOE and OExy bits to enable the output of Timer4. For details, please refer to the PWM Status Control Register (PSCR). Figure 22-18 shows the PWM output of the TIM4_<t>_OUH and TIM4_<t>_OUL ports in this mode.

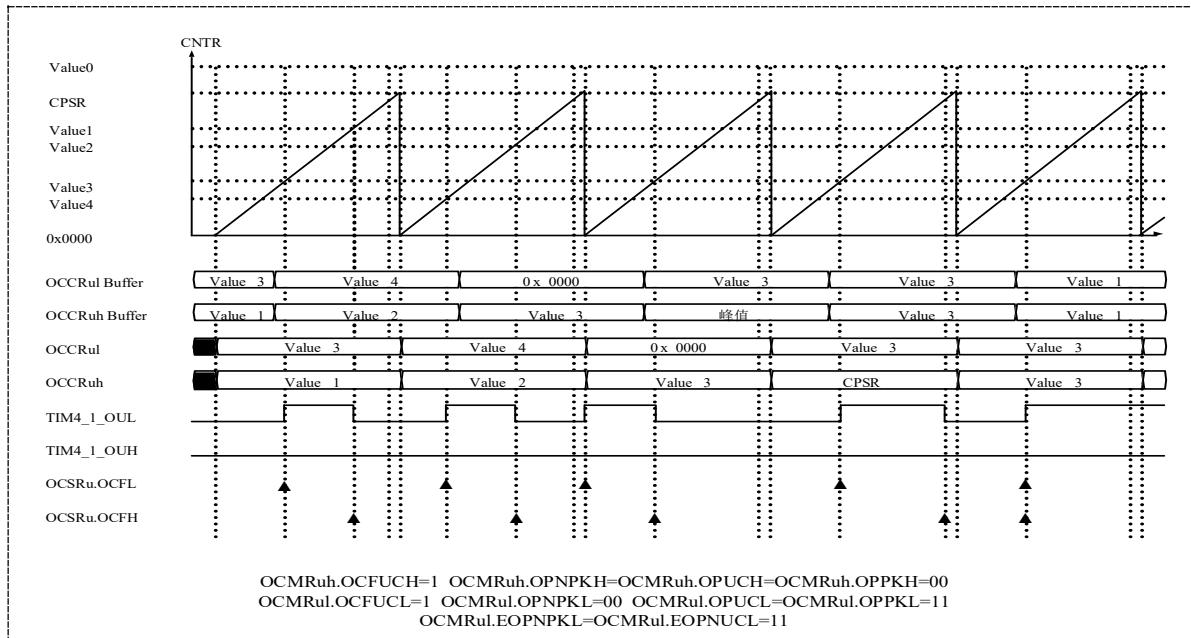


Figure 22-18 Sawtooth wave extended PWM output

Note:

- In independent PWM output mode, the port state of `TIM4_<t>_OXL` is set by bit15~bit4 of `OCMRxI` register, which is only related to the value of `OCCRxI` ($X = U, V, W, x = u, v, w$), for more detailed setting of independent or extened mode PWM output of `TIM4_<t>_OXL` port, please refer to `OCMRxI` register description and note.

22.3.5.3 Complementary PWM output

Realization of Complementary PWM Output by Software Setting

In the pass-through mode (POCR.PWMMD=00), directly set the value of OCCRxh and OCCRxl ($x=u, v, w$) to output a pair of complementary PWM waveforms to the ports, and 3 groups of ports can be set in the same way to achieve 3 phases complementary PWM outputs shown as Figure 22-19.

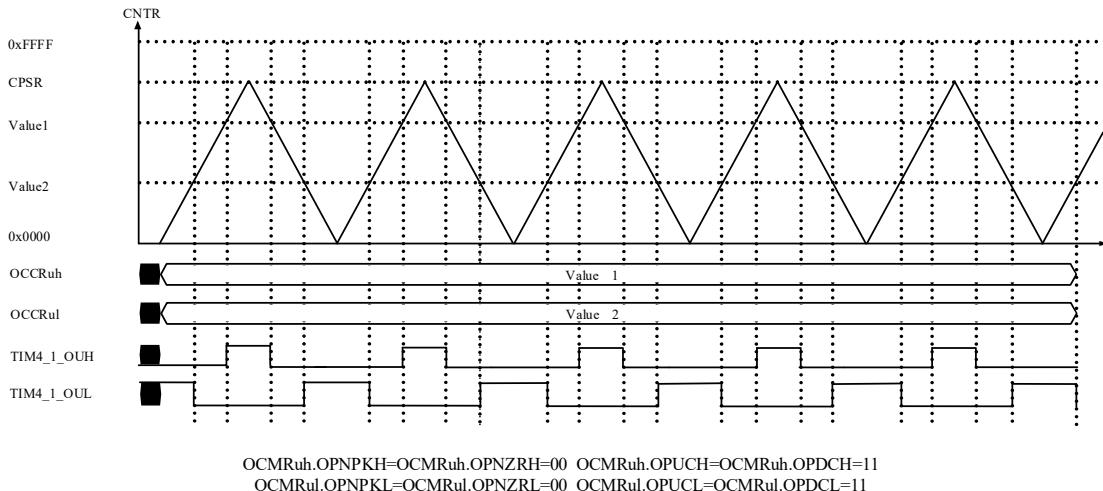


Figure 22-19 Software realizes complementary PWM output

Realization of Complementary PWM Output by Hardware Setting

In the dead-time timer mode (POCR.PWMMD = 01), the internal output signal (in_opxl) generated by the match of General Compare Value Register (OCCRxl) and the value of PWM Dead-time Control Register (PDAR/PDBR) can realize complementary PWM output by setting timing offset by hardware.

In this mode, the polarity of TIM4_<t>_OXH port output is the same as in_opxl, and the polarity of TIM4_<t>_OXL port output is opposite to in_opxl ($X = U, V, W, x = u, v, w$). Figure 22-20 is an example of a complementary PWM output in dead-time timer mode.

If the rising edge of in_opxl is detected, the output of TIM4_<t>_OXL becomes low, the dead-time counter loads the set value of PDBRx register and starts the decrement count. When the count value becomes 0x0000, the counter stops and TIM4_<t>_OXH outputs high level. If an in_opxl falling edge is detected, the TIM4_<t>_OXH output becomes low, the dead-time counter loads the set value of the PDARx register and starts the decrement count. When the count value becomes 0x0000, the counter stops and the TIM4_<t>_OXL outputs high level ($X = U, V, W, x = u, v, w$).

By setting PWM Dead-time Control Register PDAR and PDBR, output rising and falling dead time can be set accordingly.

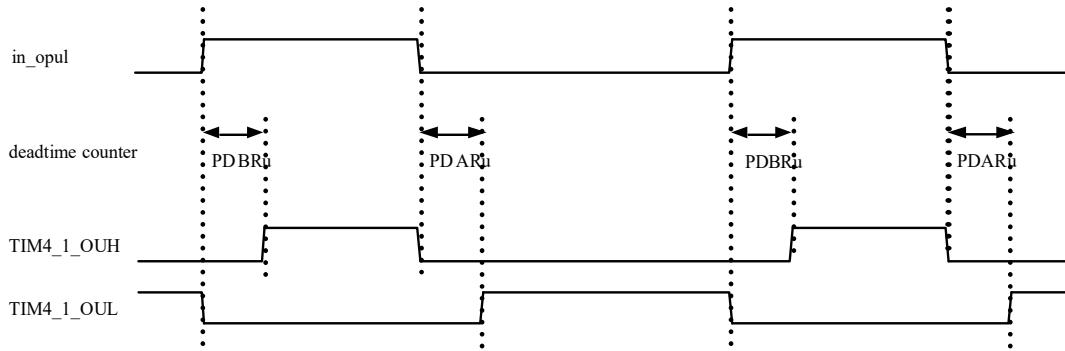


Figure 22-20 Complementary PWM output in dead-time timer mode

When the pulse width of in_opxl high level is less than the dead time set by PDBR, only the output of TIM4_<t>_OXL becomes low. TIM4_<t>_OXL output level becomes high if it passes the dead time set by PDAR register after the in_opxl falling edge. In this case, the TIM4_<t>_OXH output remains low.

When the pulse width of in_opxl low level is less than the dead time set by PDAR, only the output of TIM4_<t>_OXH becomes low. TIM4_<t>_OXH output level becomes high if it passes the dead time set by PDBR register after the in_opxl rising edge. In this case, the TIM4_<t>_OXL output remains low (X = U, V, W, x = u, v, w). Figure 22-21 shows that.

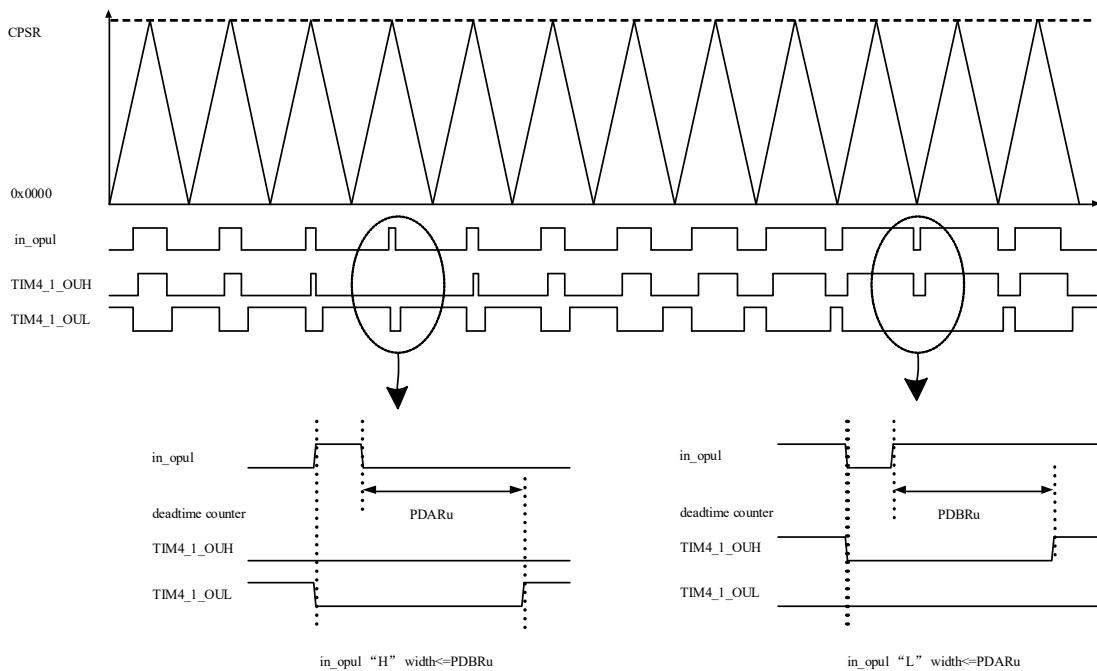


Figure 22-21 Waveform output in dead-time timer mode when the pulse width is abnormal

On the basis of the dead-time output mode by the hardware above, the pulse width of in_opxl signal can be monitored to realize the filtering control of in_opxl signal. This dead-time output

implementation of in_opxl with pulse width filtering is called dead-time counter filtering mode (POCR.PWMMD = 10) ($x = u, v, w$).

In dead-time counter filtering mode, the filter width is determined by the set value of PWM Filter Control Register (PFSRn). When the pulse width of in_opxl is more than the time set by PFSRn, the filter counter delays the output of in_opxl signal after the time set by PFSR, and then produces the complementary PWM output ($x = u, v, w$) in dead-time timer mode.

If the rising edge of in_opxl is detected, the filter counter loads the value of PFSR register and begins to measure the high-level width of in_opxl. When the high-level pulse width of in_opxl is more than the time set by PFSR, the TIM4_<t>_OXL output becomes low after the time set by PFSR, the dead time counter loads the set value of PDBR register and initiates the decrement count, and when the count value becomes 0x0000, the counter stops and the TIM4_<t>_OXH output becomes high. If the falling edge of in_opxl is detected, the filter counter loads the value of PFSR register and begins to measure the low-level width of in_opxl. When the low-level pulse width of in_opxl is more than the time set by PFSR, the TIM4_<t>_OXH output becomes low after the time set by PFSR, the dead time counter loads the set value of PDAR register and initiates the decrement count. When the count value becomes 0x0000, the counter stops and the TIM4_<t>_OXL output becomes high. When the level pulse width of in_opxl is less than the time set by PFSR, the output of TIM4_<t>_OXH and TIM4_<t>_OXL will remain unchanged ($X = U, V, W, x = u, v, w$).

Figure 22-22 is an example of complementary PWM output in dead time timer filter mode.

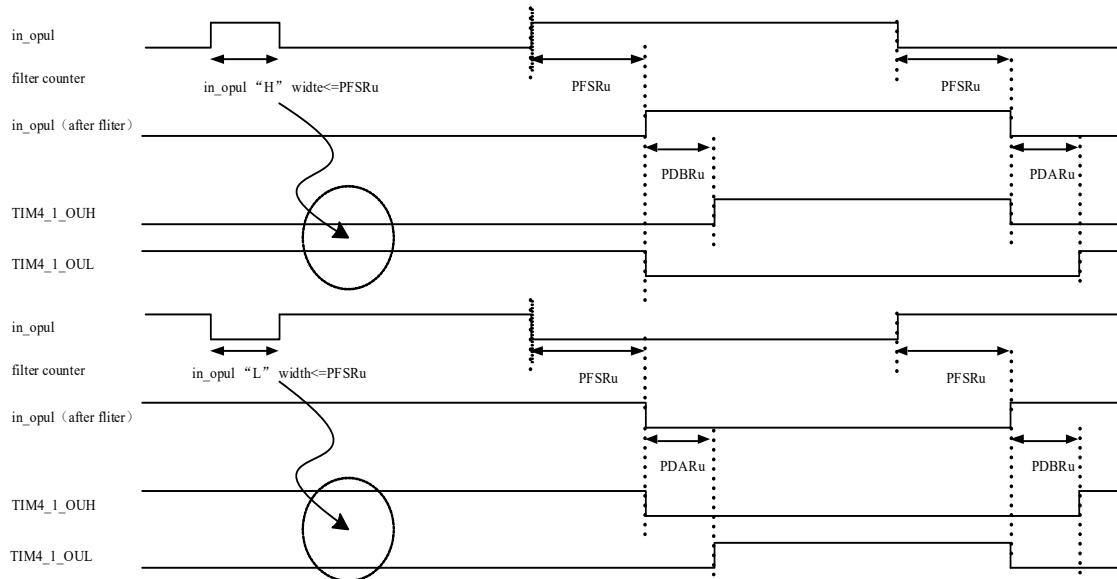


Figure 22-22 Complementary PWM output in dead time timer filter mode

Note:

- Before using the software or hardware complementary PWM output mode, you need to set the MOE and OExy bits to enable the output of Timer4. For detailed description, please refer to the PWM Status Control Register (PSCR).

22.3.6 Period interval response

The underflow interrupt mask counter is used to reduce (mask) the set times of the underflow flag bit (CCSR.IRQZF). The underflow interrupt mask counter (CVPR.ZIC[3:0]) operates as a decremental counter and loads the value set by CVPR.ZIM[3:0] at the count beginning, when CVPR.ZIC[3:0] = "0", the underflow flag (CCSR.IRQZF) is set to "1".

The overflow interrupt mask counter is used to reduce (mask) the set times of the overflow flag bit (CCSR.IRQPF). The overflow interrupt mask counter (CVPR.PIC[3:0]) operates as a decremental counter and loads the value set by CVPR.PIM[3:0] at the count beginning, when CVPR.PIC[3:0] = "0", the overflow flag (CCSR.IRQPF) is set to "1".

Figure 22-23 is the set timing diagram of IRQZF and IRQPF during period interval response.

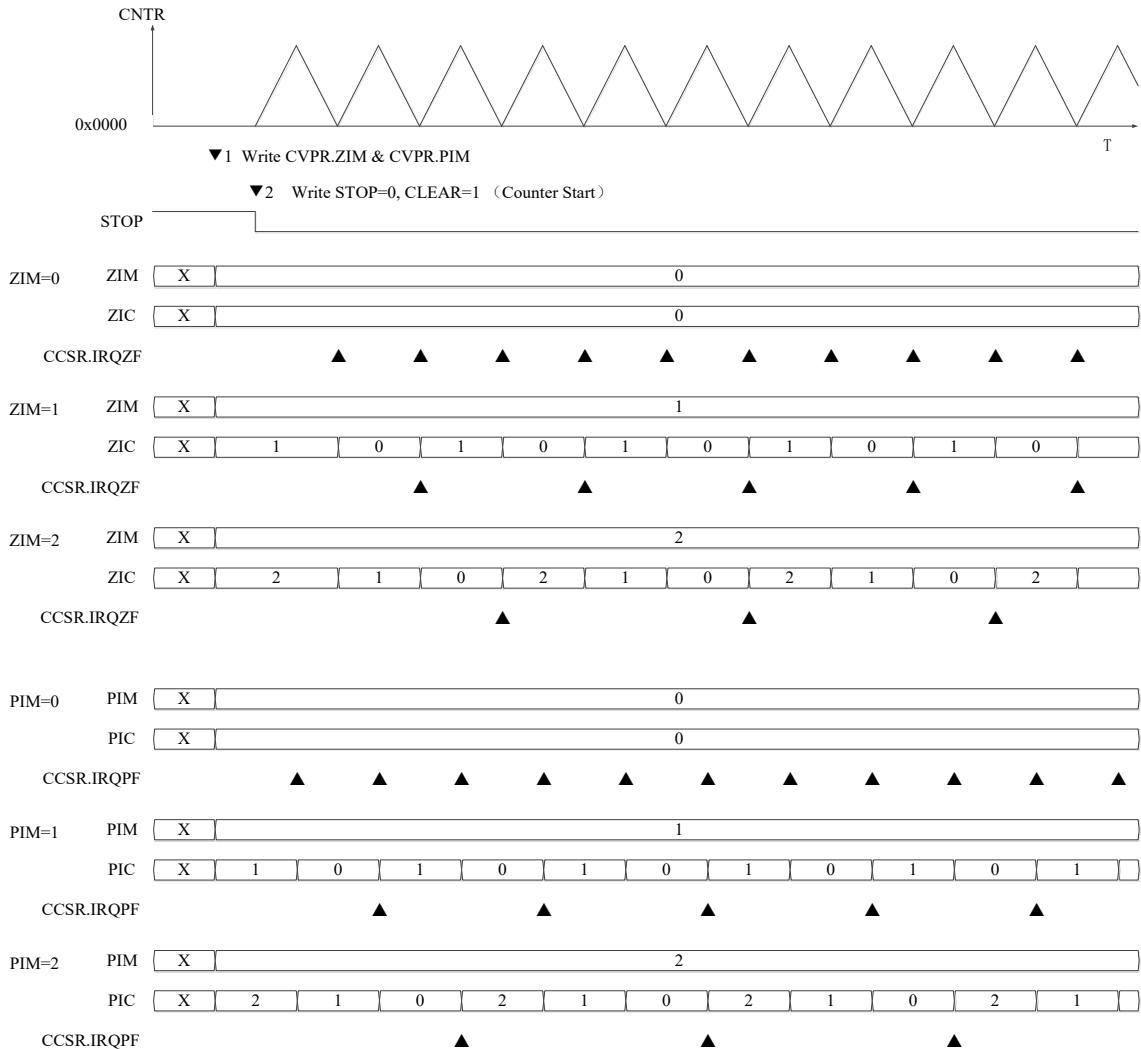


Figure 22-23 Period interval response timing diagram

▼1 When counter stops, write CVPR.ZIM and CVPR.PIM initial values, which are immediately loaded to internal counter (CVPR.ZIC, CVPR.PIC).

▼2 Initialize and start counter (STOP = 0 and CLEAR = 1), counter starts counting incrementally from zero after bus reset or software initialize CLEAR = 1, at this moment the CCSR.IRQZF flag is not set immediately, and then each time the count value of interrupt mask counter is 0x0000 and the count value of counter is 0x0000 and CPSR, the flag ▲ is the time when CCSR.IRQZF or CCSR.IRQPF is set.

Note:

- During the counter counting, if CVPR.ZIM and CVPR.PIM are written, the set values are not immediately loaded to interrupt mask counter (CVPR.ZIC and CVPR.PIC). If a software reset is written (CLEAR = 1), the written value of CVPR.ZIM and CVPR.PIM are immediately loaded as the initial value of interrupt mask counter.

The compare match event (special event output) of the Special Compare Value Register (SCCRm) also has the period interval response function. Figure 22-24 is a diagram of the period interval response output of a special event output. ($m = uh, ul, vh, vl, wh, wl$)

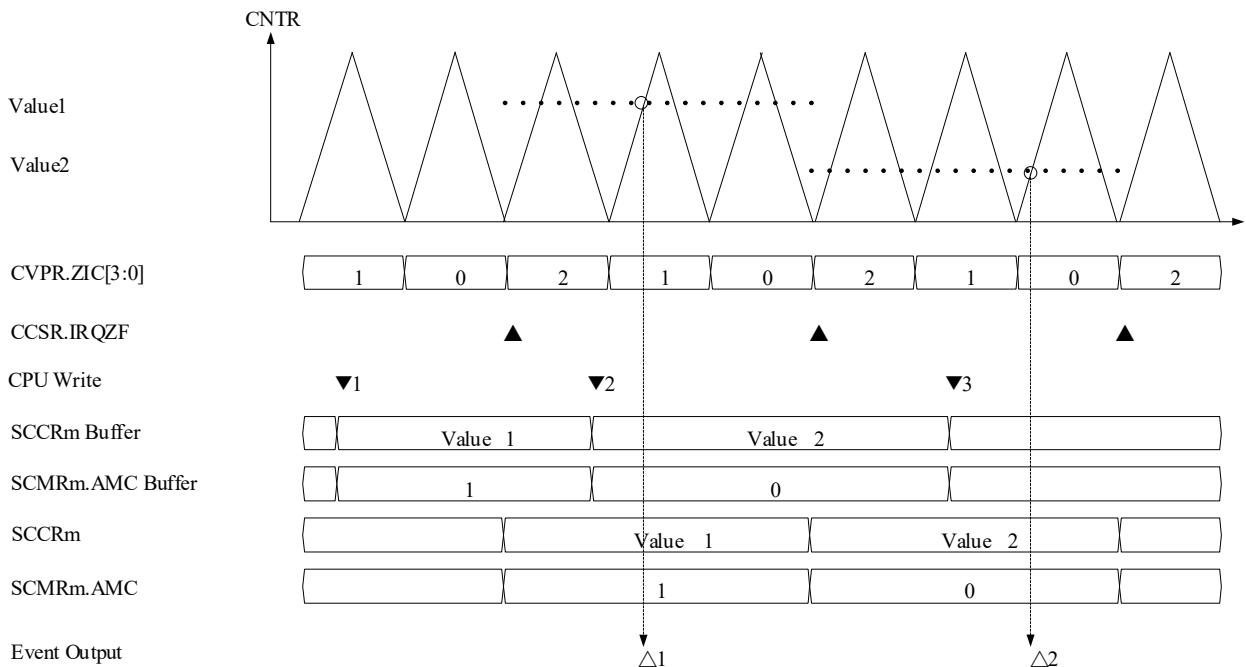


Figure 22-24 The period interval response of special event output signal

The counter is in triangular wave mode, and the underflow interrupt mask counter (CVPR.ZIC) is decremented from 2 to 0. Underflow interrupt occurs at the \blacktriangle moment.

At time $\blacktriangledown 1$, value 1 is written to the SCCRm buffer register. At the same time, MZCE=1, MPCE=0, AMC=0001 are written into the SCMRm buffer register. After that, the buffer register SCCRm and SCMRm transfer operations are performed at the \blacktriangle moment. Because MZCE = 1, AMC = 1 and count value = SCCRm = Value 1, special event output signal is set at time $\triangle 1$.

At time $\blacktriangledown 2$, value 2 is written to the SCCRm buffer register. At the same time, MZCE=1, MPCE=0, AMC=0000 are written into the SCMRm buffer register. After that, the buffer register SCCRm and SCMRm transfer operations are performed at the \blacktriangle moment. Because MZCE = 1, AMC = 0 and count value = SCCRm = Value 2, special event output signal is set at time $\triangle 2$.

22.3.7 EMB control

Each Timer4 unit has an output invalid event interface connected to the EMB event output from the EMB module. Abnormal condition events selected on this interface can be set in the EMB module (refer to EMB chapter).

During the normal output period of the 3 groups of PWM ports in each unit, if an abnormal EMB event from EMB module is detected, the PSCR.MOE bit is asynchronously cleared by hardware, and

the output state of each PWM port becomes a preset state. According to the setting of PSCR.OSxy, the preset port state can be output high-impedance state, output low level or output high level ($x=U, V, W, y=H, L$).

For example, if PSCR.OSUH=01, and an EMB event occurs during the normal output of the TIM4_<t>_OUH port of Timer4, the output on the TIM4_<t>_OUH port becomes high-impedance.

After the EMB abnormal event is cleared, if PSCR.AOE=1 at this time, PSCR.MOE will be automatically set by hardware, and then the PWM port returns to normal output; if PSCR.AOE=0 at this time, software needs to set PSCR.MOE to restore PWM port to normal output.

It should be noted that the asynchronous signal of MOE is used to control the PWM output enable on the hardware. When the value of MOE changes, it needs to wait several clock cycles to read the actual value on the bus. For the detailed operation method of EMB, please refer to EMB chapter.

22.3.8 Monitor output

Each Timer4 unit has a PWM period output monitor port TIM4_<t>_PCT and special event output monitor port TIM4_<t>_ADSM. These monitor ports are output to the outside, and users can flexibly control the application system according to the changing information of the ports.

The PWM period output monitor port TIM4_<t>_PCT is used to monitor the count status of the current counter. In the triangular wave mode, each time the peak or bottom is counted, a rollover occurs on the detection port TIM4_<t>_PCT.

Figure 22-25 is an example of a counting direction signal.

The special event output monitor port TIM4_<t>_ADSM can be used to monitor the special compare events of the internal counter. The special compare output channel that needs to be monitored is selected by the SCER.EVTRS bit of the Special Expand Control Register (SCER). When the special compare value on this channel is matched with the count value, the monitor port TIM4_<t>_ADSM will be set (output is high); when the counter on this channel counts to zero (sawtooth wave overflow and triangular wave counts to the bottom), the monitor port TIM4_<t>_ADSM is reset (output is low).

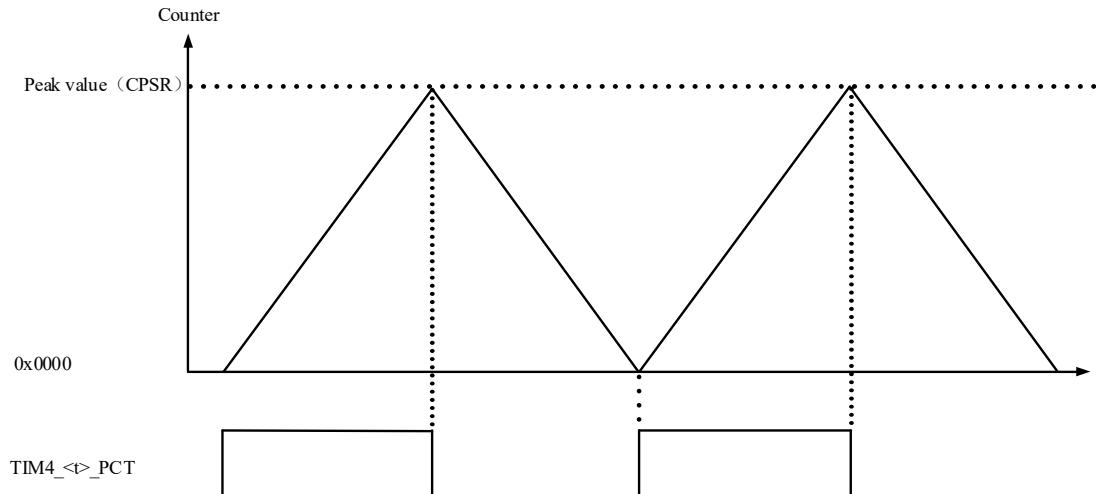


Figure 22-25 Example of counting direction signal output

22.4 Interrupt and Event Description

22.4.1 Count compare match interrupt

There are 6 General Compare Value Register (OCCRm), which can be compared with the count value to produce the compare match event signals. When the count compare matches, the OCSRn.OCFH and OCSRn.OCFL bits in the General Control Status Register (OCSRn) will be set to 1 respectively. At this time, if OCSRn.OCIEH and OCSRn.OCIEL are set to enable interrupts, the corresponding interrupt request (TMR4_<t>_GCMm, m= uh, ul, vh, vl, wh, wl; n=u,v,w) will also be triggered.

22.4.2 Count period match interrupt

The CCSR.IRQPF or CCSR.IRQZF bits of the Control Status Register (CCSR) will be set to 1 when the sawtooth wave counts up to the overflow point, the triangular wave counts to the bottom point or peak point. At this time, if the CCSR.IRQPEN or CCSR.IRQZEN bit is set to enable the interrupt, the count period match interrupt (TMR4_<t>_GOVF and TMR4_<t>_GUDF) can be triggered at the corresponding time point.

22.4.3 Reload count match interrupt

When the reload function is enabled, TMR4_PFSRn is used as the period count value of the reload timer. When the count is enabled, the value of register TMR4_PFSRn is reloaded to the initial value of the counter, and the decrement operation is performed. After one period is completed, a reload count interrupt request is generated, and the RCSR.RTIFU, RCSR.RTIFV, RCSR.RTIFW bits in the Reload Control Status Register (RCSR) will be set to 1 respectively. At this time, if the interrupt mask of RCSR.RTIDU, RCSR.RTIDV and RCSR.RTIDW is set to be invalid, the corresponding reload count match interrupt request (TMR4_<t>_RLOm, m=U, V, W) will also be triggered.

22.4.4 Special compare match event

The 6 special compare value registers (SCCRm) of Timer4 produce 6 special event output signals, which can be used to trigger other modules, such as starting ADC.

During the clock counting process, if a SCCRm count compare match event ($TMR4_{<t>}SCRM_m$, $m = uh, ul, vh, vl, wh, wl$) occurs, a corresponding valid request signal will be generated. The signal can be configured to any event EVT output signal (set by the SCSR.EVTOS bit) to trigger other modules.

The output of the event request signal can be selected to output in compare start mode or delay start mode. In compare start mode (SCSR.EVTMS = 0), the special event output signal is output directly after the SCCR count compare match event is generated. In delay start mode (SCSR.EVTMS = 1), if the OCCRxh or OCCRxl(selected by the SCSR.EVTDS bit, $x = u, v, w$) count compare match event is generated, special event signal is output after the time set by SCCR. Figure 22-26 is an example of the request output of the special event output signal in delay start mode.

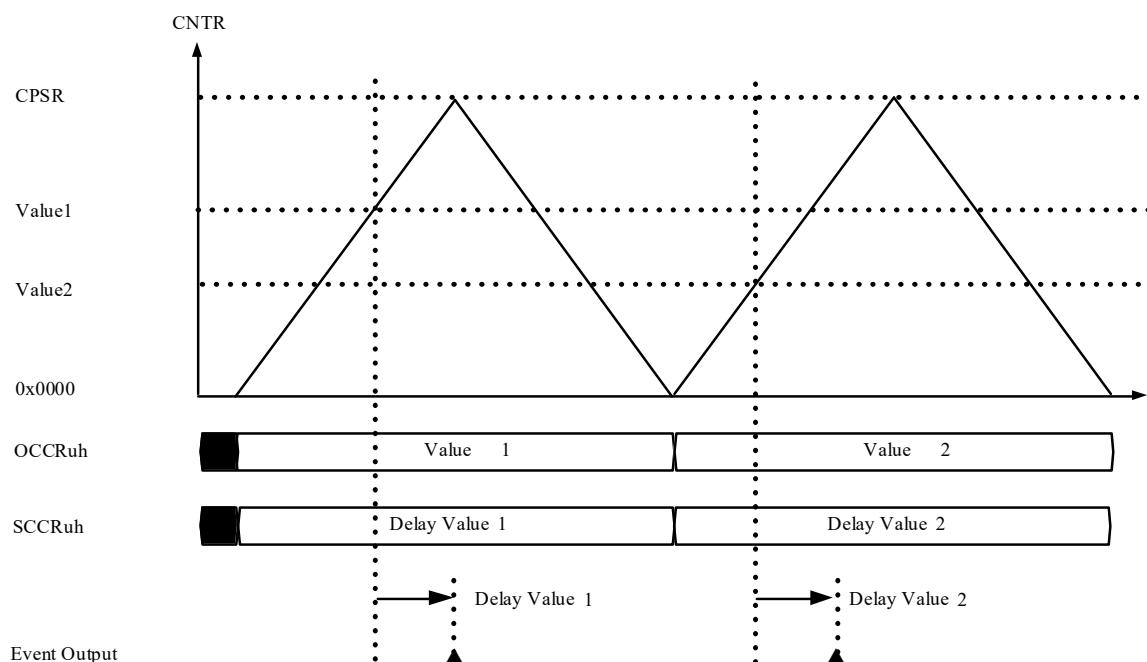


Figure 22-26 Output timing of special event output signal in delay start mode

Note:

- During delay counting, if an OCR count compare match event occurs again, the delay counter reloads the count value and decrements the count again. Therefore, if the OCR compare match event time interval is less than the set delay time in SCCR, the request signal output of the special event may never be generated.

22.5 Register description

Table 22-3 is the register list of the Timer4 module.

BASE ADDR: 0x40038000 (U1), 0x40038400 (U2), 0x40038800 (U3)

Table 22-3 Timer4 register list

Register name	Symbol	Offset	Bit width	Reset value
Counter Value Register	TMR4_CNTR	0x0046h	16	0x0000h
Period Value Register	TMR4_CPSR	0x0042h	16	0xFFFFh
Control Status Register	TMR4_CCSR	0x0048h	16	0x0040h
Valid Period Register	TMR4_CVPR	0x004Ah	16	0x0000h
General Compare Value UH Register	TMR4_OCCRuh	0x0002h	16	0x0000h
General Compare Value UL Register	TMR4_OCCRul	0x0006h	16	0x0000h
General Compare Value VH Register	TMR4_OCCRvh	0x000Ah	16	0x0000h
General Compare Value VL Register	TMR4_OCCRvl	0x000Eh	16	0x0000h
General Compare Value WH Register	TMR4_OCCRwh	0x0012h	16	0x0000h
General Compare Value WL Register	TMR4_OCCRwl	0x0016h	16	0x0000h
General Control Status U Register	TMR4_OCSRu	0x0018h	16	0xFF00h
General Control Status V Register	TMR4_OCSRv	0x001Ch	16	0xFF00h
General Control Status W Register	TMR4_OCSRw	0x0020h	16	0xFF00h
General Expand Control U Register	TMR4_OCERu	0x001Ah	16	0x0000h
General Expand Control V Register	TMR4_OCERv	0x001Eh	16	0x0000h
General Expand Control W Register	TMR4_OCERw	0x0022h	16	0x0000h
General Mode Control UH Register	TMR4_OCMRuh	0x0024h	16	0x0000h
General Mode Control UL Register	TMR4_OCMRul	0x0028h	32	0x00000000h
General Mode Control VH Register	TMR4_OCMRvh	0x002Ch	16	0x0000h
General Mode Control VL Register	TMR4_OCMRvl	0x0030h	32	0x00000000h
General Mode Control WH Register	TMR4_OCMRwh	0x0034h	16	0x0000h
General Mode Control WL Register	TMR4_OCMRwl	0x0038h	32	0x00000000h
Special Compare Value UH Register	TMR4_SCCRuh	0x00B2h	16	0x0000h
Special Compare Value UL Register	TMR4_SCCRul	0x00B6h	16	0x0000h
Special Compare Value VH Register	TMR4_SCCRvh	0x00BAh	16	0x0000h
Special Compare Value VL Register	TMR4_SCCRvl	0x00BEh	16	0x0000h
Special Compare Value WH Register	TMR4_SCCRwh	0x00C2h	16	0x0000h
Special Compare Value WL Register	TMR4_SCCRwl	0x00C6h	16	0x0000h
Special Control Status UH Register	TMR4_SCSRuh	0x00C8h	16	0x0000h
Special Control Status UL Register	TMR4_SCSRul	0x00CCh	16	0x0000h
Special Control Status VH Register	TMR4_SCSRvh	0x00D0h	16	0x0000h
Special Control Status VL Register	TMR4_SCSRvl	0x00D4h	16	0x0000h

Register name	Symbol	Offset	Bit width	Reset value
Special Control Status WH Register	TMR4_SCSRwh	0x00D8h	16	0x0000h
Special Control Status WL Register	TMR4_SCSRwl	0x00DCh	16	0x0000h
Special Expand Control Register	TMR4_SCER	0x00E4h	16	0xFF00h
Special Mode Control UH Register	TMR4_SCMRuh	0x00CAh	16	0xFF00h
Special Mode Control UL Register	TMR4_SCMRul	0x00CEh	16	0xFF00h
Special Mode Control VH Register	TMR4_SCMRvh	0x00D2h	16	0xFF00h
Special Mode Control VL Register	TMR4_SCMRvl	0x00D6h	16	0xFF00h
Special Mode Control WH Register	TMR4_SCMRwh	0x00DAh	16	0xFF00h
Special Mode Control WL Register	TMR4_SCMRwl	0x00DEh	16	0xFF00h
PWM Output Control U Register	TMR4_POCRu	0x0098h	16	0xFF00h
PWM Output Control V Register	TMR4_POCRv	0x009Ch	16	0xFF00h
PWM Output Control W Register	TMR4_POCRw	0x00A0h	16	0xFF00h
PWM Status Control Register	TMR4_PSCR	0x00E0h	32	0x05550000h
PWM Filter Control U Register	TMR4_PFSRu	0x0082h	16	0x0000h
PWM Filter Control V Register	TMR4_PFSRv	0x008Ah	16	0x0000h
PWM Filter Control W Register	TMR4_PFSRw	0x0092h	16	0x0000h
PWM Deadtime Control UA Register	TMR4_PDARu	0x0084h	16	0x0000h
PWM Deadtime Control UB Register	TMR4_PDBRu	0x0086h	16	0x0000h
PWM Deadtime Control VA Register	TMR4_PDARv	0x008Ch	16	0x0000h
PWM Deadtime Control VB Register	TMR4_PDBRv	0x008Eh	16	0x0000h
PWM Deadtime Control WA Register	TMR4_PDARw	0x0094h	16	0x0000h
PWM Deadtime Control WB Register	TMR4_PDBRw	0x0096h	16	0x0000h
Reload Control Status Register	TMR4_RCSR	0x00A4h	16	0x0000h

Note:

- In the following register description, m = uh, ul, vh, vl, wh, wl, n = u, v, w. m refers to the output control of corresponding ports TIM4_<t>_OUH, TIM4_<t>_OUL, TIM4_<t>_OVH, TIM4_<t>_OVL, TIM4_<t>_OWH, TIM4_<t>_OWL, etc. n refers to the output control of corresponding ports TIM4_<t>_OUx, TIM4_<t>_OVx, TIM4_<t>_OWx, where x = H or L, and H or L has corresponding symmetric bits in these registers, etc.

22.5.1 Counter Value Register (TMR4_CNTR)

Reset value: 0x0000h

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
CNTR[15:0]																
Bit	Symbol	Bit name										Description				Read and write
b15~b0	CNTR[15:0]	Counter current value										When the count stops, the counter count value can be initialized by writing a value to this register During counting, these bits indicates the current counter count value Note: A value cannot be written to this register while counting				R/W

22.5.2 Period Value Register (TMR4_CPSR)

Reset value: 0xFFFFh

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
CPSR[15:0]																
Bit	Symbol	Place name										Description				Read and write
b15~b0	CPSR[15:0]	General period reference										Counter count period value Note: When reading data from this address area, what is read is not the value of the buffer register, but the value of the CPSR register				R/W

22.5.3 Control Status Register (TMR4_CCSR)

Reset value: 0x0040h

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]								
Bit	Symbol	Bit name		Description												Read and write				
b15	ECKEN	Counter clock selection		0: Internal PCLK0 clock 1: External TIM4_<t>_CLK port input clock Note 1: This bit is set when counter stops Note 2: When using the external TIM4_<t>_CLK port to input the clock, after writing STOP= "1", the first edge of the external input clock is regarded as invalid. The counting action starts from the second edge, and both rising and falling edges are valid edges.												R/W				
b14	IRQZF	Underflow flag		0: No count underflow 1: Count underflow occurs Note 1: When the period interval response function is enabled, the position setting condition of this bit is determined by CVPR Note 2: When the counter is reset by the bus or written to CLEAR = '1', the IRQZF bit will not be set												R/W				
b13	IRQZEN	Underflow interrupt enable		0: Disable generate interrupt to CPU from IRQZF 1: Enable generate interrupt to CPU from IRQZF												R/W				
b12~b10	Reserved	-		Read as "0", write as "0"												R/W				
b9	IRQPF	Overflow flag		0: No count overflow 1: Count overflow Note 1: When the period interval response function is enabled, the position setting condition of this bit is determined by CVPR. Note 2: When the counter is reset by the bus or written to CLEAR = '1', the IRQPF bit will not be set												R/W				
b8	IRQOPEN	Overflow interrupt enable		0: Disable generating interrupt to CPU from IRQPF 1: Enable generate interrupt to CPU from IRQPF												R/W				
b7	BUFEN	CPSR buffer enable		0: Disable CPSR buffer function 1: Enable CPSR buffer function												R/W				
b6	STOP	Counter enable		0: Counter start 1: Counter stop												R/W				
b5	MODE	Waveform mode		0: Sawtooth wave mode (up-count only) 1: Triangular wave mode												R/W				
b4	CLEAR	Counter reset		0: No operation 1: Counter reset Note: This bit is always 0 when read												R/W				
b3~b0	CKDIV	Counter clock division		This bit indicates the counter clock division of the basic counter 0000: The count clock is PCLK0 0001: The count clock is PCLK0/2 0010: The count clock is PCLK0/4 0011: The count clock is PCLK0/8 0100: The count clock is PCLK0/16 0101: The count clock is PCLK0/32 0110: The count clock is PCLK0/64 0111: The count clock is PCLK0/128 1000: The count clock is PCLK0/256 1001: The count clock is PCLK0/512 1010: The count clock is PCLK0/1024 Please do not set other values Note: When the counting clock source is the input clock of the external TIM4_<t>_CLK port, the frequency division setting is invalid													R/W			

22.5.4 Valid Period Register (TMR4_CVPR)

Reset value: 0x0000h

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]							
<hr/>																			
Bit	Symbol	Bit name				Description								Read and write					
b15~b12	PIC[3:0]	Overflow interrupt mask counter				Read as the remaining number of currently masked overflow interrupts Note: For detailed function description, please refer to the Chapter: Period interval response.								R					
b11~b8	ZIC[3:0]	Underflow interrupt mask counter				Read as the remaining number of currently masked underflow interrupts Note: For detailed function description, please refer to the Chapter: Period interval response.								R					
b7~b4	PIM[3:0]	Overflow interrupt mask setting				Set the number of overflow interrupts for mask								R/W					
b3~b0	ZIM[3:0]	Underflow interrupt mask setting				Set the number of underflow interrupts for mask								R/W					

22.5.5 General Compare Value Register (TMR4_OCCRm)

Reset value: 0x0000h

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
OCCR[15:0]															
<hr/>															
Bit	Symbol	Bit name				Description								Read and write	
b15~b0	OCCR[15:0]	General compare value				General compare value Note: When reading data from this address area, what is read is not the value of the buffer register, but the value of the OCCR register								R/W	

22.5.6 General Control Status Register (TMR4_OCSRn)

Reset value: 0xFF00h

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	
Bit	Symbol	Bit name										Description				Read and write
b15~b8	Reserved	-										Read as "1", write as "1"				R/W
b7	OCFL	Count match flag L										0: Counter count value is not equal to OCCRxL set value 1: Counter count value is equal to OCCRxL set value (x=u, v, w) Note: This bit is valid when OCEL=1				R/W
b6	OCFH	Count match flag H										0: Counter count value is not equal to OCCRxh set value 1: Counter count value is equal to OCCRxh set value (x=u, v, w) Note: This bit is valid when OCEH=1				R/W
b5	OCIEL	Count match interrupt enable L										0: Disable interrupt when OCFL is set 1: Enable interrupt when OCFL is set				R/W
b4	OCIEH	Count match interrupt enable H										0: Disable interrupt when OCFH is set 1: Enable interrupt when OCFH is set				R/W
b3	OCPL	Port state L when the compare output is invalid										0: Output low level on TIM4_<t>_OxL when OCEL = 0 1: Output high level on TIM4_<t>_OxL when OCEL = 0 (x=U, V, W)				R/W
b2	OCPH	Port state H when the compare output is invalid										0: Output low level on TIM4_<t>_OxH when OCEH = 0 1: Output high level on TIM4_<t>_OxH when OCEH = 0 (x=U, V, W)				R/W
b1	OCEL	Port output selection L										0: The compare output is invalid. The port status of TIM4_<t>_OxL is determined by OCPL. 1: The compare output is valid. The port status of TIM4_<t>_OxL is determined by the setting of OCMRyl and the status of OCFL. (x=U, V, W, y=u, v, w)				R/W
b0	OCEH	Port output selection H										0: The compare output is invalid. The port status of TIM4_<t>_OxH is determined by OCPH. 1: The compare output is valid. The port status of TIM4_<t>_OxH is determined by the setting of OCMRyh and the status of OCFH. (x=U, V, W, y=u, v, w)				R/W

22.5.7 General Expand Control Register (TMR4_OCERn)

Reset value: 0x0000h

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	MCEC_L	MCEC_H	LMM_L	LMM_H	LMC_L	LMC_H	MLBUFEN[1:0]	MHBUFEN[1:0]	CLBUFEN[1:0]	CHBUFEN[1:0]				
Bit	Symbol	Bit name												Description	
b15~b14	Reserved	-												Read and write	
b13	MCECL	Extended control enable L												0: Disable extended control function of match conditions for count value and OCCRxL 1: Enable extended control function of match conditions for count value and OCCRxL (x=u, v, w) Note 1: This bit can only be modified when the output compare function is disabled (OCSR.OCEL = 0) Note 2: For details, please refer to the notes of TMR4_OCMRm register	
b12	MCECH	Extended control enable H												0: Disable extended control function of match conditions for count value and OCCRxH 1: Enable extended control function of match conditions for count value and OCCRxH (x=u, v, w) Note 1: This bit can only be modified when the output compare function is disabled (OCSR.OCEH = 0) Note 2: For details, please refer to the notes of TMR4_OCMRm register	R/W
b11	LMML	Period interval response link for general mode control L												0: Period interval response function link is disabled, OCMRxl buffer transfer is determined by MLBUFEN setting 1: Period interval response function link is enabled, OCMRxl buffer transfer is not only determined by MLBUFEN setting, but must satisfy CVPR.PIC[3:0]=0000 (when the count overflows) or CVPR.ZIC[3:0]= 0000 (when the count underflows) (x=u, v, w)	R/W
b10	LMMH	Period interval response link for general mode control H												0: Period interval response function link is disabled, OCMRhx buffer transfer is determined by MHBUFEN setting 1: Period interval response function link is enabled, OCMRhx buffer transfer is not only determined by MHBUFEN setting, but must satisfy CVPR.PIC[3:0]=0000 (when the count overflows) or CVPR.ZIC[3:0]= 0000 (when the count underflows) (x=u, v, w)	R/W
b9	LMCL	Period interval response link for general compare L												0: Period interval response function link is disabled, OCCRxL buffer transfer is determined by CLBUFEN setting 1: Period interval response function link is enabled, OCCRxL buffer transfer is not only determined by CLBUFEN setting, but must satisfy CVPR.PIC[3:0]=0000 (when the count overflows) or CVPR.ZIC[3:0]= 0000 (when the count underflows) (x=u, v, w)	R/W
b8	LMCH	Period interval response link for general compare H												0: Period interval response function link is disabled, OCCRxh cache transfer is determined by CHBUFEN setting 1: Period interval response function link is enabled, OCCRxh buffer transfer is not only determined by CHBUFEN setting, but must satisfy CVPR.PIC[3:0]=0000 (when the count overflows) or CVPR.ZIC[3:0]= 0000 (when the count underflows) (x=u, v, w)	R/W
b7~b6	MLBUFEN[1:0]	OCMRxl buffer transfer												00: OCMRxl buffer register values are written directly to OCMRxl 01: OCMRxl buffer register values are written to OCMRxl at count underflow 10: OCMRxl buffer register values are written to OCMRxl at count overflow 11: OCMRxl buffer register values are written to OCMRxl at count underflow or overflow (x=u, v, w)	R/W
b5~b4	MHBUFEN[1:0]	OCMRhx buffer transfer												00: OCMRhx buffer register values are written directly to OCMRhx	R/W

			01: OCMRxh buffer register values are written to OCMRxh at count underflow 10: OCMRxh buffer register values are written to OCMRxh at count overflow 11: OCMRxh buffer register values are written to OCMRxh at count underflow or overflow (x=u, v, w)	
b3~b2	CLBUFEN[1:0]	OCCRxl buffer transfer	00: OCCRxl buffer register values are written directly to OCCRxl 01: OCCRxl buffer register values are written to OCCRxl at count underflow 10: OCCRxl buffer register values are written to OCCRxl at count overflow 11: OCCRxl buffer register values are written to OCCRxl at count underflow or overflow (x=u, v, w)	R/W
b1~b0	CHBUFEN[1:0]	OCCRxh buffer transfer	00: OCCRxh buffer register values are written directly to OCCRxh 01: OCCRxh buffer register values are written to OCCRxh at count underflow 10: OCCRxh buffer register values are written to OCCRxh at count overflow 11: OCCRxh buffer register values are written to OCCRxh at count underflow or overflow (x=u, v, w)	R/W

22.5.8 General Mode Control Register (TMR4_OCMRm)

Reset value: 0x0000h

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 for OCMRuh, OCMRvh, OCMRwh

Bit	Symbol	Bit name	Description	Read and write
b15~b14	OPNZRH[1:0]	Underflow point port state H	Conditions: Count underflow & OCCRxh count mismatch (x=u, v, w) 00: TIM4_<t>_OyH port remains unchanged when condition is satisfied 01: TIM4_<t>_OyH port output high level when condition is satisfied 10: TIM4_<t>_OyH port output low level when condition is satisfied 11: TIM4_<t>_OyH port output invert when condition is satisfied (y=U, V, W)	R/W
b13~b12	OPNPKH[1:0]	Overflow point port state H	Conditions: Count overflow & OCCRxh count mismatch (x=u, v, w) 00: TIM4_<t>_OyH port remains unchanged when condition is satisfied 01: TIM4_<t>_OyH port output high level when condition is satisfied 10: TIM4_<t>_OyH port output low level when condition is satisfied 11: TIM4_<t>_OyH port output invert when condition is satisfied (y=U, V, W)	R/W
b11~b10	OPZRH[1:0]	Underflow point port state H	Conditions: Count underflow & OCCRxh count match (x=u, v, w) 00: TIM4_<t>_OyH port remains unchanged when condition is satisfied 01: TIM4_<t>_OyH port output high level when condition is satisfied 10: TIM4_<t>_OyH port output low level when condition is satisfied 11: TIM4_<t>_OyH port output invert when condition is satisfied (y=U, V, W)	R/W
b9~b8	OPUCH[1:0]	Up count port state H	Conditions: Count up & OCCRxh count match (x=u, v, w) 00: TIM4_<t>_OyH port remains unchanged when condition is satisfied 01: TIM4_<t>_OyH port output high level when condition is satisfied 10: TIM4_<t>_OyH port output low level when condition is satisfied 11: TIM4_<t>_OyH port output invert when condition is satisfied (y=U, V, W)	R/W
b7~b6	OPPKH[1:0]	Overflow point port state H	Conditions: Count overflow & OCCRxh count match (x=u, v, w) 00: TIM4_<t>_OyH port remains unchanged when condition is satisfied 01: TIM4_<t>_OyH port output high level when condition is satisfied 10: TIM4_<t>_OyH port output low level when condition is satisfied 11: TIM4_<t>_OyH port output invert when condition is satisfied (y=U, V, W)	R/W
b5~b4	OPDCH[1:0]	Down count port state H	Conditions: Count down & OCCRxh count match (x=u, v, w) 00: TIM4_<t>_OyH port remains unchanged when condition is satisfied 01: TIM4_<t>_OyH port output high level when condition is satisfied 10: TIM4_<t>_OyH port output low level when condition is satisfied	R/W

			11: TIM4_<t>_OyH port output invert when condition is satisfied (y=U, V, W)	
b3	OCFZRH	Underflow point OCFH state H	Conditions: Count underflow & OCCRxh count match (x=u, v, w) 0: OCSR.OCFH bit remains unchanged when condition is satisfied 1: OCSR.OCFH bit is set when condition is satisfied	R/W
b2	OCFUCH	Up count OCFH state H	Conditions: Count up & OCCRxh count match (x=u, v, w) 0: OCSR.OCFH bit remains unchanged when condition is satisfied 1: OCSR.OCFH bit is set when condition is satisfied	R/W
b1	OCFPKH	Overflow point OCFH state H	Conditions: Count overflow & OCCRxh count match (x=u, v, w) 0: OCSR.OCFH bit remains unchanged when condition is satisfied 1: OCSR.OCFH bit is set when condition is satisfied	R/W
b0	OCFDCH	Down count OCFH state H	Conditions: Count down & OCCRxh count match (x=u, v, w) 0: OCSR.OCFH bit remains unchanged when condition is satisfied 1: OCSR.OCFH bit is set when condition is satisfied	R/W

Reset value: 0x00000000h

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]		OCF ZRL		OCF UCL		OCF PKL		OCF DCL	

Note: This register bit description is used for OCMRul, OCMRvl, OCMRwl

Bit	Symbol	Bit name	Description	Read and write
b31~b30	EOPNZRL[1:0]	Extended underflow point port state L	Conditions: Count underflow & OCCRxl count mismatch & OCCRxh count match (x=u, v, w) 00: TIM4_<t>_OyL port remains unchanged when condition is satisfied 01: TIM4_<t>_OyL port output high level when condition is satisfied 10: TIM4_<t>_OyL port output low level when condition is satisfied 11: TIM4_<t>_OyL port output invert when condition is satisfied (y=U, V, W)	R/W
b29~b28	EOPNPKL[1:0]	Expanded overflow point port state L	Conditions: Count overflow & OCCRxl count mismatch & OCCRxh count match (x=u, v, w) 00: TIM4_<t>_OyL port remains unchanged when condition is satisfied 01: TIM4_<t>_OyL port output high level when condition is satisfied 10: TIM4_<t>_OyL port output low level when condition is satisfied 11: TIM4_<t>_OyL port output invert when condition is satisfied (y=U, V, W)	R/W
b27~b26	EOPZRL[1:0]	Extended underflow point port state L	Conditions: Count underflow & OCCRxl count match & OCCRxh count match (x=u, v, w) 00: TIM4_<t>_OyL port remains unchanged when condition is satisfied 01: TIM4_<t>_OyL port output high level when condition is satisfied 10: TIM4_<t>_OyL port output low level when condition is satisfied 11: TIM4_<t>_OyL port output invert when condition is satisfied (y=U, V, W)	R/W
b25~b24	EOPUCL[1:0]	Extended up count port status L	Conditions: Count up & OCCRxl count match & OCCRxh count match (x=u, v, w) 00: TIM4_<t>_OyL port remains unchanged when condition is satisfied	R/W

			01: TIM4_<t>_OyL port output high level when condition is satisfied 10: TIM4_<t>_OyL port output low level when condition is satisfied 11: TIM4_<t>_OyL port output invert when condition is satisfied (y=U, V, W)	
b23~b22	EOPPKL[1:0]	Expanded overflow point port state L	Conditions: Count overflow & OCCRxI count match & OCCRxh count match (x=u, v, w) 00: TIM4_<t>_OyL port remains unchanged when condition is satisfied 01: TIM4_<t>_OyL port output high level when condition is satisfied 10: TIM4_<t>_OyL port output low level when condition is satisfied 11: TIM4_<t>_OyL port output invert when condition is satisfied (y=U, V, W)	R/W
b21~b20	EOPDCL[1:0]	Expanded down count port status L	Conditions: Count down & OCCRxI count match & OCCRxh count match (x=u, v, w) 00: TIM4_<t>_OyL port remains unchanged when condition is satisfied 01: TIM4_<t>_OyL port output high level when condition is satisfied 10: TIM4_<t>_OyL port output low level when condition is satisfied 11: TIM4_<t>_OyL port output invert when condition is satisfied (y=U, V, W)	R/W
b19~b18	EOPNUCL[1:0]	Expanded up count port status L	Conditions: Count up & OCCRxI count mismatch & OCCRxh count match (x=u, v, w) 00: TIM4_<t>_OyL port remains unchanged when condition is satisfied 01: TIM4_<t>_OyL port output high level when condition is satisfied 10: TIM4_<t>_OyL port output low level when condition is satisfied 11: TIM4_<t>_OyL port output invert when condition is satisfied (y=U, V, W)	R/W
b17~b16	EOPNDCL[1:0]	Expanded down count port status L	Conditions: Count down & OCCRxI count mismatch & OCCRxh count match (x=u, v, w) 00: TIM4_<t>_OyL port remains unchanged when condition is satisfied 01: TIM4_<t>_OyL port output high level when condition is satisfied 10: TIM4_<t>_OyL port output low level when condition is satisfied 11: TIM4_<t>_OyL port output invert when condition is satisfied (y=U, V, W)	R/W
b15~b14	OPNZRL[1:0]	Underflow point port state L	Conditions: Count underflow & OCCRxI count mismatch & OCCRxh count mismatch (x=u, v, w) 00: TIM4_<t>_OyL port remains unchanged when condition is satisfied 01: TIM4_<t>_OyL port output high level when condition is satisfied 10: TIM4_<t>_OyL port output low level when condition is satisfied 11: TIM4_<t>_OyL port output invert when condition is satisfied (y=U, V, W)	R/W
b13~b12	OPNPKL[1:0]	Overflow point port status L	Conditions: Count overflow & OCCRxI count mismatch & OCCRxh count mismatch (x=u, v, w) 00: TIM4_<t>_OyL port remains unchanged when condition is satisfied 01: TIM4_<t>_OyL port output high level when condition is satisfied 10: TIM4_<t>_OyL port output low level when condition is satisfied 11: TIM4_<t>_OyL port output invert when condition is satisfied (y=U, V, W)	R/W
b11~b10	OPZRL[1:0]	Underflow point port state L	Conditions: Count underflow & OCCRxI count match & OCCRxh count mismatch (x=u, v, w) 00: TIM4_<t>_OyL port remains unchanged when condition is satisfied 01: TIM4_<t>_OyL port output high level when condition is satisfied	R/W

			10: TIM4_<t>_OyL port output low level when condition is satisfied 11: TIM4_<t>_OyL port output invert when condition is satisfied (y=U, V, W)	
b9~b8	OPUCL[1:0]	Up count port status L	Conditions: Count up & OCCRxI count match & OCCRxH count mismatch (x=u, v, w) 00: TIM4_<t>_OyL port remains unchanged when condition is satisfied 01: TIM4_<t>_OyL port output high level when condition is satisfied 10: TIM4_<t>_OyL port output low level when condition is satisfied 11: TIM4_<t>_OyL port output invert when condition is satisfied (y=U, V, W)	R/W
b7~b6	OPPKL[1:0]	Overflow point port status L	Conditions: Count overflow & OCCRxI count match & OCCRxH count mismatch (x=u, v, w) 00: TIM4_<t>_OyL port remains unchanged when condition is satisfied 01: TIM4_<t>_OyL port output high level when condition is satisfied 10: TIM4_<t>_OyL port output low level when condition is satisfied 11: TIM4_<t>_OyL port output invert when condition is satisfied (y=U, V, W)	R/W
b5~b4	OPDCL[1:0]	Down count port status L	Conditions: Count down & OCCRxI count match & OCCRxH count mismatch (x=u, v, w) 00: TIM4_<t>_OyL port remains unchanged when condition is satisfied 01: TIM4_<t>_OyL port output high level when condition is satisfied 10: TIM4_<t>_OyL port output low level when condition is satisfied 11: TIM4_<t>_OyL port output invert when condition is satisfied (y=U, V, W)	R/W
b3	OCFZRL	Underflow point OCFL state L	Conditions: Count underflow & OCCRxI count match (x=u, v, w) 0: OCSR.OCFL bit remains unchanged when condition is satisfied 1: OCSR.OCFL bit is set when condition is satisfied	R/W
b2	OCFUCL	Count up OCFL status L	Conditions: Count up & OCCRxI count match (x=u, v, w) 0: OCSR.OCFL bit remains unchanged when condition is satisfied 1: OCSR.OCFL bit is set when condition is satisfied	R/W
b1	OCFPKL	Overflow point OCFL state L	Conditions: Count overflow & OCCRxI count match (x=u, v, w) 0: OCSR.OCFL bit remains unchanged when condition is satisfied 1: OCSR.OCFL bit is set when condition is satisfied	R/W
b0	OCFDCL	Count down OCFL status L	Conditions: Count down & OCCRxI count match (x=u, v, w) 0: OCSR.OCFL bit remains unchanged when condition is satisfied 1: OCSR.OCFL bit is set when condition is satisfied	R/W

Note:

- When reading data from the local address area, it is not the value of the buffer register, but the value of the OCML register.
- TIM4_<t>_OyL can be determined by the count value of OCCRxI and counter (independent operation mode), or the count value of OCCRxH and counter, and the count value of OCCRxI and counter (link operation mode). Write the same 12-bit values to bit [31:20] and bit [15:4] of register OCML and write OCML[19:16] to "0000" at the same time. At this time, the output of TIM4_<t>_OyL is not affected by OCCRxH, but only determined by OCCRxI. This mode is called independent operation mode - channel yH is determined by OCCRxH and channel yL is determined by OCCRxI. If the above conditions are not satisfied

in independent operation mode, it is the link operation mode-channel yL output is affected by both OCCRxh and OCCRxl.

(x=u, v, w, y=U, V, W)

- When the following two conditions are satisfied, it is also considered to be in line with the conditions set by OCMRxh.OPPKH[7:6] (x=u, v, w):
 - 1) CCSR.MODE=1 && Count overflow && OCCRxh=0xFFFF
 - 2) OCERx.MCECH=1 && Count overflow && OCCRxh≥CNTR
- When the following two conditions are met, it is also considered to be in line with the conditions set by OCMRxl.OPPKL[7:6] (x=u, v, w):
 - 1) CCSR.MODE=1 && Count overflow && OCCRxl=0xFFFF
 - 2) OCERx.MCECL=1 && Count overflow && OCCRxl≥CNTR

22.5.9 Special Compare Value Register (TMR4_SCCRm)

Reset value: 0x0000h

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0		
SCCR[15:0]																	
Bit	Symbol	Bit name														Description	Read and write
b15~b0	SCCR[15:0]	Special compare value														Special compare value (compare start mode) or delay compare value (delay start mode) Note: When reading data from this address area, what is read is not the value of the buffer register, but the value of the SCCR register	R/W

22.5.10 Special Control Status Register (TMR4_SCSRm)

Reset value: 0x0000h

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]							
Bit	Symbol	Bit name		Description												Read and write				
b15	ZEN	Underflow point EVT enable		0: EVT does not operate when counting underflow 1: When counting underflow: EVTMS = 0 & SCCR Compare Match & SCMR Set Match, EVT Startup Output EVTMS = 1 & OCCR Compare Match & SCMR Set Match, EVT Delay Mode Startup												R/W				
b14	UEN	Up count EVT enable		0: EVT does not operate when counting up 1: When counting up: EVTMS = 0 & SCCR Compare Match & SCMR Set Match, EVT Startup Output EVTMS = 1 & OCCR Compare Match & SCMR Set Match, EVT Delay Mode Startup												R/W				
b13	PEN	Overflow point EVT enable		0: EVT does not operate when counting overflow 1: When counting overflow: EVTMS = 0 & SCCR Compare Match & SCMR Set Match, EVT Startup Output EVTMS = 1 & OCCR Compare Match & SCMR Set Match, EVT Delay Mode Startup												R/W				
b12	DEN	Down count EVT enable		0: EVT does not operate when counting down 1: Down count: EVTMS = 0 & SCCR Compare Match & SCMR Set Match, EVT Startup Output EVTMS = 1 & OCCR Compare Match & SCMR Set Match, EVT Delay Mode Startup												R/W				
b11~b10	Reserved	-		Read as "0", write as "0"												R/W				
b9	EVTDS	EVT delay object selection		0: Set OCCRxh as the delay compare match object in delay start mode 1: Set OCCRxl as the delay compare match object in delay start mode (x=u, v, w) Note: This bit has no effect when EVTMS=0												R/W				
b8	EVTMS	EVT mode selection		0: Compare start mode (CNTR and SCCR compare triggers) 1: Delay start mode (compare match events trigger after SCCR delay)												R/W				
b7~b6	Reserved	-		Read as "0", write as "0"												R/W				
b5	LMC	Period interval response link		0: Period interval response function link is disabled, SCCR buffer transfer is determined by BUFEN setting 1: Period interval response function link is enabled, SCCR buffer transfer is not only determined by BUFEN setting, but must satisfy CVPR.PIC[3:0]=0000 (when count overflows) or CVPR.ZIC[3:0]= 0000 (when count underflows)												R/W				
b4~b2	EVTOS[2:0]	EVT output selection		000: EVT Output of Special Evnet 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: EVT Output of Special Evnet 5 is Valid Please do not set other values												R/W				
b1~b0	BUFEN[1:0]	SCCR & SCMR buffer transfer		00: SCCR, SCMR buffer register values are written directly to SCCR, SCMR 01: SCCR, SCMR buffer register values are written to SCCR, SCMR when counting underflow 10: SCCR, SCMR buffer register values are written to SCCR, SCMR when counting overflow 11: SCCR, SCMR buffer register values are written to SCCR, SCMR when counting underflows or overflow												R/W				

22.5.11 Special Expand Control Register (TMR4_SCER)

Reset value: 0xFF00h

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Bit	Symbol	Reserved										PCTS	EVTRS[2:0]		
b15~b4	Reserved	-										Read as "0", write as "0"	R/W		
b3	PCTS	Count direction output enable										When the counter works in triangular wave mode, this bit controls the count direction signal output to the port 0: Disable count direction signal output to TIM4_<t>_PCT port 1: Enable count direction signal output to TIM4_<t>_PCT port	R/W		
b2~b0	EVTRS[2:0]	Special event output selection										000: Disable special event output to the port 001: EVT output of Special Event 0 to TIM4_<t>_ADSM port 010: EVT output of Special Event 1 to TIM4_<t>_ADSM port 011: EVT output of Special Event 2 to TIM4_<t>_ADSM port 100: EVT output of Special Event 3 to TIM4_<t>_ADSM port 101: EVT output of Special Event 4 to TIM4_<t>_ADSM port 110: EVT output of Special Event 5 to TIM4_<t>_ADSM port 111: Prohibitions	R/W		

22.5.12 Special Mode Control Register (TMR4_SCMRm)

Reset value: 0xFF00h

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
Bit	Symbol	Reserved										MPCE	MZCE	-	-	AMC[3:0]
b15~b8	Reserved	-										Read as "1", write as "1"	R/W			
b7	MPCE	Period interval response enable										0: Disable compare between AMC and CVPR.PIC 1: Enable compare between AMC and CVPR.PIC	R/W			
b6	MZCE	Period interval response enable										0: Disable compare between AMC and CVPR.ZIC 1: Enable compare between AMC and CVPR.ZIC	R/W			
b5~b4	Reserved	-										Read as "0", write as "0"	R/W			
b3~b0	AMC[3:0]	The period interval value of special event output										This bit sets the period interval value of the special event output function. When AMC and CVPR.PIC or CVPR.ZIC are equal, the special event output function is valid.	R/W			

Note:

- When reading data from the local address area, it is not the value of the buffer register, but the value of the SCMR register.

22.5.13 PWM Output Control Register (TMR4_POCRn)

Reset value: 0xFF00h

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]				
Bit	Symbol	Bit name		Description										Read and write	
b15~b8	Reserved	- Read as "1", write as "1"										R/W			
b7~b6	LVLS[1:0]	PWM output polarity control	00: Neither output of TIM4_<t>_OxH nor TIM4_<t>_OxL are inverted 01: Both output of TIM4_<t>_OxH and TIM4_<t>_OxL are inverted 10: Output of TIM4_<t>_OxH is inverted, and output of TIM4_<t>_OxL is not inverted. 11: Output of TIM4_<t>_OxH is not inverted, and output of TIM4_<t>_OxL is inverted.										R/W		
b5~b4	PWMMD[1:0]	PWM output mode	00: Pass-through mode 01: Dead-time timer mode 10: Dead-time timer filtering mode 11: Prohibitions										R/W		
b3	Reserved	- Read as "0", write as "0"										R/W			
b2~b0	DIVCK[2:0]	Count clock division	This bit indicates the count clock division of the filter counter and the dead-time counter. 000: The count clock is PCLK0 001: The count clock is PCLK0/2 010: The count clock is PCLK0/4 011: The count clock is PCLK0/8 100: The count clock is PCLK0/16 101: The count clock is PCLK0/32 110: The count clock is PCLK0/64 111: The count clock is PCLK0/128										R/W		

22.5.14 PWM Status Control Register (TMR4_PSCR)

Reset value: 0x05550000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
Reserved		OSWL[1:0]		OSWH[1:0]		OSVL[1:0]		OSVH[1:0]		OSUL[1:0]		OSUH[1:0]				
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
Reserved		AOE		MOE		ODT[1:0]		OEWL		OEWH		OEVL		OEVH	OEUL	OEUH

Bit	Symbol	Bit name	Description	Read and write
b31~b28	Reserved	-	Read as "0", write as "0"	R/W
b27~b26	OSWL[1:0]	TIM4_<t>_OWL output disable status	Conditions: 1. EMB event occurs; 2. Software writes MOE=0; 3. MOE=1&OEWL=0 00: When one of the above conditions is met, the TIM4_<t>_OWL port outputs normally 01: When one of the above conditions is met, the TIM4_<t>_OWL port output is Hi-z 10: When one of the above conditions is met, the TIM4_<t>_OWL port output is fixed to low level 11: When one of the above conditions is met, the TIM4_<t>_OWL port output is fixed to high level Note: This bit can only be modified when Timer4 is stopped (CCSR.STOP=1)	R/W
b25~b24	OSWH[1:0]	TIM4_<t>_OWH output disable status	Conditions: 1. EMB event occurs; 2. Software writes MOE=0; 3. MOE=1&OEWH=0 00: When one of the above conditions is met, the TIM4_<t>_OWH port outputs normally 01: When one of the above conditions is met, the TIM4_<t>_OWH port output is Hi-z 10: When one of the above conditions is met, the TIM4_<t>_OWH port output is fixed to low level 11: When one of the above conditions is met, the TIM4_<t>_OWH port output is fixed to high level Note: This bit can only be modified when Timer4 is stopped (CCSR.STOP=1)	R/W
b23~b22	OSVL[1:0]	TIM4_<t>_OVL output disable status	Conditions: 1. EMB event occurs; 2. Software writes MOE=0; 3. MOE=1&OEVL=0 00: When one of the above conditions is met, the TIM4_<t>_OVL port outputs normally 01: When one of the above conditions is met, the TIM4_<t>_OVL port output is Hi-z 10: When one of the above conditions is met, the TIM4_<t>_OVL port output is fixed to low level 11: When one of the above conditions is met, the TIM4_<t>_OVL port output is fixed to high level Note: This bit can only be modified when Timer4 is stopped (CCSR.STOP=1)	R/W
b21~b20	OSVH[1:0]	TIM4_<t>_OVH output disable status	Conditions: 1. EMB event occurs; 2. Software writes MOE=0; 3. MOE=1&OEVH=0 00: When one of the above conditions is met, the TIM4_<t>_OVH port outputs normally 01: When one of the above conditions is met, the TIM4_<t>_OVH port output is Hi-z 10: When one of the above conditions is met, the TIM4_<t>_OVH port output is fixed to low level 11: When one of the above conditions is met, the TIM4_<t>_OVH port output is fixed to high level Note: This bit can only be modified when Timer4 is stopped (CCSR.STOP=1)	R/W
b19~b18	OSUL[1:0]	TIM4_<t>_OUL output disable status	Conditions: 1. EMB event occurs; 2. Software writes MOE=0; 3. MOE=1&OEUL=0 00: When one of the above conditions is met, the TIM4_<t>_OUL port outputs normally 01: When one of the above conditions is met, the TIM4_<t>_OUL port output is Hi-z 10: When one of the above conditions is met, the TIM4_<t>_OUL port output is fixed to low level 11: When one of the above conditions is met, the TIM4_<t>_OUL port output is fixed to high level Note: This bit can only be modified when Timer4 is stopped (CCSR.STOP=1)	R/W

			Conditions: 1. EMB event occurs; 2. Software writes MOE=0; 3. MOE=1&OEUH=0 00: When one of the above conditions is met, TIM4_<t>_OUH Port normal output 01: When one of the above conditions is met, the TIM4_<t>_OUH port output is Hi-z 10: When one of the above conditions is met, the TIM4_<t>_OUH port output is fixed to low level 11: When one of the above conditions is met, the TIM4_<t>_OUH port output is fixed to high level Note: This bit can only be modified when Timer4 is stopped (CCSR.STOP=1)	
b17~b16	OSUH[1:0]	TIM4_<t>_OUH output disable status		R/W
b15~b10	Reserved	-	Read as "0", write as "0"	R/W
b9	AOE	Automatic output enable	0: After the EMB event is cleared, the MOE bit can only be set to 1 by software to restore the normal output of the PWM 1: After the EMB event is cleared, the MOE bit is automatically set to 1 by hardware, and the PWM output returns to normal Note: EMB events can be cleared by setting EMB_STATCLR or clearing EMB_SOE	R/W
b8	MOE	Main output enable	0: TIM4_<t>_Oxy port output state is specified by the OSxy bits (x=U, V, W, y=H, L) 1: TIM4_<t>_Oxy port output is enabled when the OExy bit is set to enable (x=U, V, W, y=H, L) Note: This bit is cleared by hardware immediately after an EMB event occurs. After the EMB event is cleared, this bit can be set to 1 by software or automatically by hardware depending on the setting of the AOE bit.	R/W
b7~b6	ODT	Port enable bit valid time	Conditions: After the value of the port enable bit OExy is changed (x=U, V, W, y=H, L) 0X: When the condition is met, it will take effect immediately 10: When the condition is met, it will take effect when the counter underflows next time 11: When the condition is met, it will take effect when the counter overflows next time	R/W
b5	OEWL	TIM4_<t>_OWL output enable	0: TIM4_<t>_OWL port output state is set by OSWL bit 1: When MOE=1, the TIM4_<t>_OWL port outputs normal PWM waveform	R/W
b4	OEWH	TIM4_<t>_OWH output enable	0: TIM4_<t>_OWH port output state is set by OSWH bit 1: When MOE=1, the TIM4_<t>_OWH port outputs normal PWM waveform	R/W
b3	OEVL	TIM4_<t>_OVL output enable	0: TIM4_<t>_OVL port output state is set by OSVL bit 1: When MOE=1, the TIM4_<t>_OVL port outputs normal PWM waveform	R/W
b2	OEVH	TIM4_<t>_OVH output enable	0: TIM4_<t>_OVH port output state is set by OSVH bit 1: When MOE=1, the TIM4_<t>_OVH port outputs normal PWM waveform	R/W
b1	OEUL	TIM4_<t>_OUL output enable	0: TIM4_<t>_OUL port output status is set by OSUL bit 1: When MOE=1, the TIM4_<t>_OUL port outputs normal PWM waveform	R/W
b0	OEUH	TIM4_<t>_OUH output enable	0: TIM4_<t>_OUH port output state is set by OSUH bit 1: When MOE=1, the TIM4_<t>_OUH port outputs normal PWM waveform	R/W

Table 22-4 is for relationship description of PWM port output status and register setting value .

Table 22-4 PWM port output status and register setting value

MOE bit	OExy bit	OSxy bit	TIM4_<t>_Oxy output status
1	0	00	Normal output (Timer4 driver)
		01	Hi-z
		10	Low level
		11	High level
	1	X	Normal output (Timer4 driver)
0	X	00	Normal output (Timer4 driver)
		01	Hi-z
		10	Low level
		11	High level

22.5.15 PWM Filter Control Register (TMR4_PFSRn)

Reset value: 0x0000h

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
PFSR[15:0]																
Bit	Symbol	Bit name										Description				Read and write
b15~b0	PFSR[15:0]	Filter initial value										Filter count initial value Note: When the dead-time timer filter mode is not selected in the PWM waveform output mode, the 16-bit filter counter is used as a 16-bit reload counter. At this time, the 16-bit filter counter can periodically generate interrupt output. This function has nothing to do with the PWM waveform generator function. .				R/W

22.5.16 PWM Deadtime Control Register (TMR4_PDARn)

Reset value: 0x0000h

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
PDA/BR[15:0]																
Bit	Symbol	Bit name										Description				Read and write
b15~b0	PDA/BR[15:0]	Dead time initial value										Dead time count initial value				R/W

22.5.17 Reload Control Status Register (TMR4_RCSR)

Reset value: 0x0000h

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	
Bit	Symbol	Bit name												Description		Read and write
b15	RTSW	Reload counter stop W												0: No operation 1: Stop reload counter W and clear RTIFW Note: This bit is always 0 when read		R/W
b14	RTEW	Reload counter start W												0: Reload counter W stop 1: Reload counter W start		R/W
b13	RTICW	Clear count match status W												0: No operation 1: Clear RTIFW flag bit Note: This bit is always 0 when read		R/W
b12	RTIFW	Count match status W												0: Reload counter count value does not match PFSRw 1: Reload counter count value matches PFSRw		R
b11	RTSV	Reload counter stop V												0: No operation 1: Stop reload counter V and clear RTIFV Note: This bit is always 0 when read		R/W
b10	RTEV	Reload counter start V												0: Reload counter V stop 1: Reload counter V start		R/W
b9	RTICV	Clear count match status V												0: No operation 1: Clear RTIFV flag bit Note: This bit is always 0 when read		R/W
b8	RTIFV	Count match status V												0: Reload counter count value does not match PFSRv 1: Reload counter count value matches PFSRv		R
b7	RTSU	Reload counter stop U												0: No operation 1: Stop reload counter U and clear RTIFU Note: This bit is always 0 when read		R/W
b6	RTEU	Reload counter start U												0: Reload counter U stop 1: Reload counter U start		R/W
b5	RTICU	Clear count match status U												0: No operation 1: Clear RTIFU flag bit Note: This bit is always 0 when read		R/W
b4	RTIFU	Count match status U												0: Reload counter count value does not match PFSRu 1: Reload counter count value matches PFSRu		R
b3	Reserved	-												Read as "0", write as "0"		R/W
b2	RTIDW	Reload interrupt mask W												0: When the reload function is enabled, the reload interrupt W output is valid. 1: When the reload function is enabled, the reload interrupt W output is invalid		R/W
b1	RTIDV	Reload interrupt mask V												0: When the reload function is enabled, the reload interrupt V output is valid. 1: When the reload function is enabled, the reload interrupt V output is invalid		R/W
b0	RTIDU	Reload interrupt mask U												0: When the reload function is enabled, the reload interrupt U output is valid. 1: When the reload function is enabled, the reload interrupt U output is invalid		R/W

23 Emergency Brake Module (EMB)

23.1 Introduction

The emergency brake module will generate control events to control the timer to stop outputting PWM signals to motor when certain conditions are met. The following factors are used to generate control events:

- Change of input level of external port
- Level of PWM output port occurs in phase (same high or same low)
- Voltage comparator comparison results
- External oscillator stop oscillation
- Write register software control

The EMB structure diagram is shown as follows Figure 23-1.

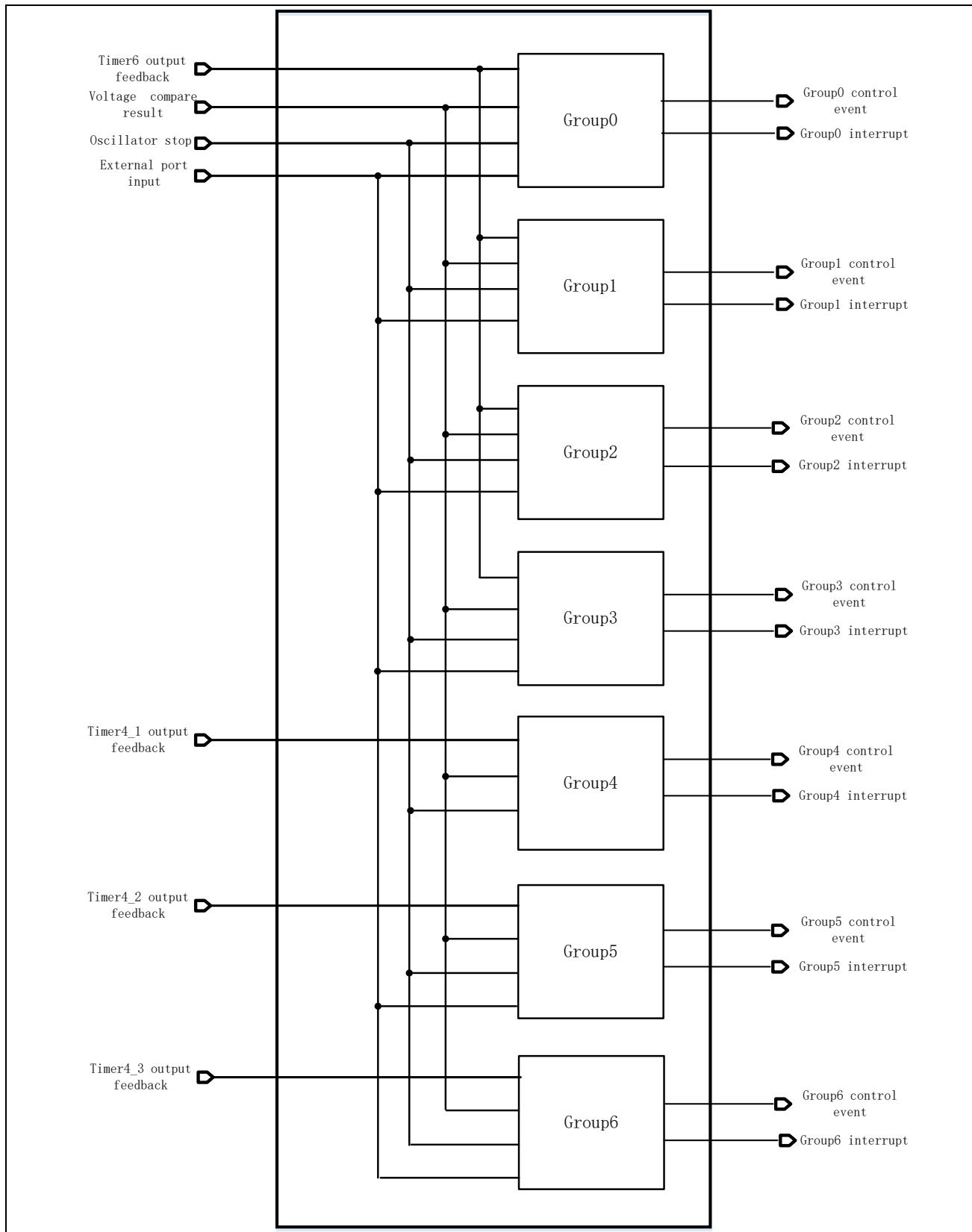


Figure 23-1 EMB structure diagram

23.2 Functional description

23.2.1 Overview

The EMB is used to output a control event signal to the timer module (Timer4, Timer6) with PWM function when certain conditions are met, and notify the timer module to close the current PWM output. The EMB module has 7 groups, of which group0~group3 is used to control Timer6, which is selected and used by Timer6 register setting, and group4~group6 is used to control Timer4, corresponding to 3 units of Timer4 respectively.

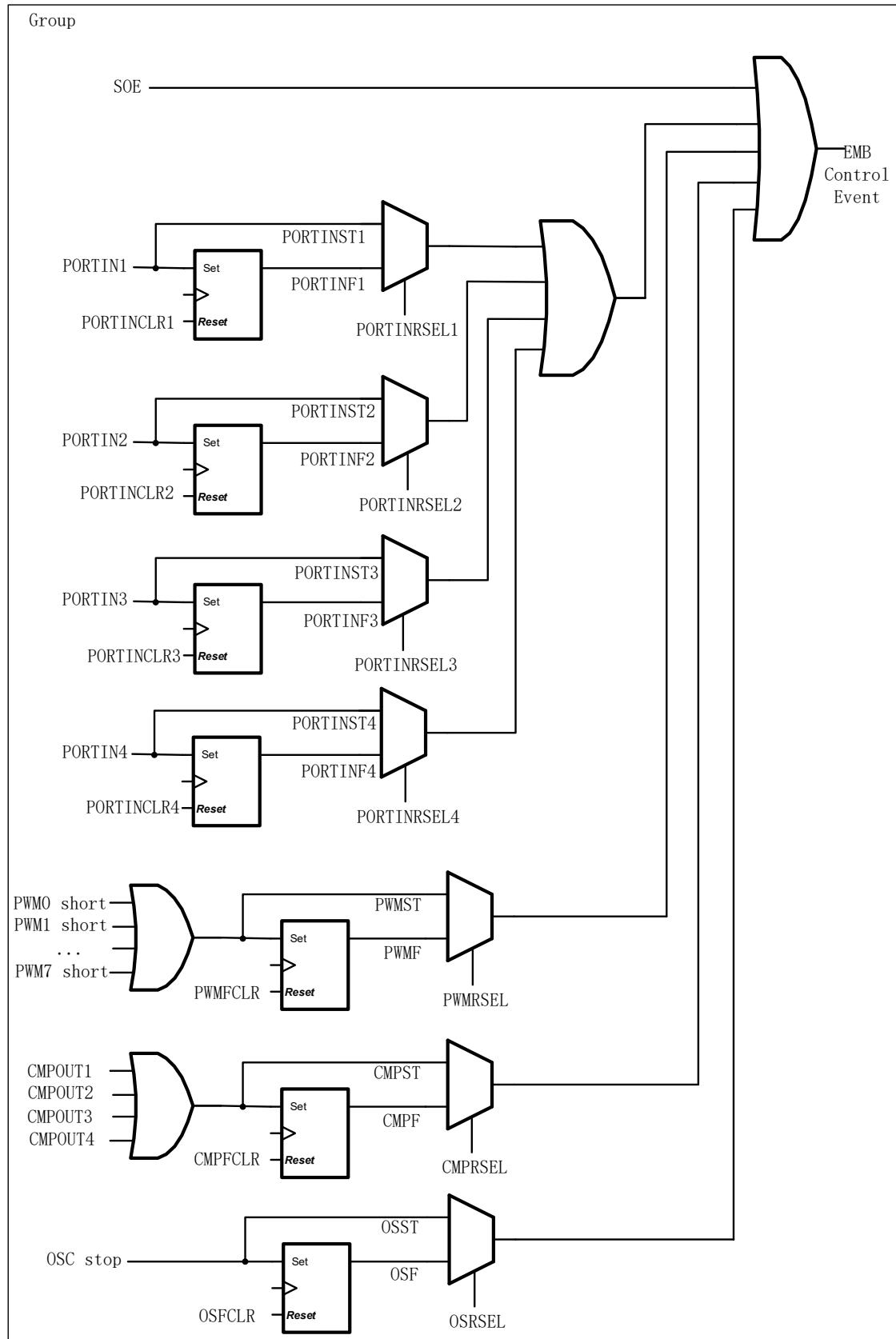


Figure 23-2 EMB Group Control

23.2.2 Control PWM signal output when external port input level changes

The EMB has 4 external ports for controlling the PWM signal output when the input level changes. Each group can independently set one or more of the 4 external ports to be valid, and the port assignments are shown in the table below.

Table 23-1 EMB port assignment

port	PIN name
POR1T	PA11,PB2,PC5,PD7
POR2T	PA6,PE15,PB11,PB12
POR3T	PA7, PC2, PD9
POR4T	PD4,PH2,PB15,PI4

When using the external port input level change to control the PWM signal output, firstly the EMB control register EMB_CTL1_x.PORTINENy ($x=0\sim6, y=1\sim4$) is set to be valid, and at the same time, the EMB control register EMB_CTL1_x.INVSELy ($x=0\sim6, y=1\sim4$) is configured. If EMB_CTL1_x.PORTINENy ($x=0\sim6, y=1\sim4$)=0, the control signal is generated when the port level is high or if EMB_CTL1_x.PORTINENy ($x=0\sim6, y=1\sim4$)=1, the control signal is generated when the port level is low. Enable the filter and set the filter clock through EMB_CTL2_x.NFENy ($x=0\sim6, y=1\sim4$) and EMB_CTL2_x.NFSELy ($x=0\sim6, y=1\sim4$) as required.

After the filter samples the input signal according to the filter clock, the filter adopts three comparisons that are consistent, that is, when the filter clock samples the same level three times on the port, the level is regarded as an effective level and transmitted to the inside of the module; The level that is less than 3 times consistent will be filtered out as external interference and will not be transmitted to the inside of the module.

When a valid level is input on the port, the port input state EMB_STATx.PORTINSTy ($x=0\sim6, y=1\sim4$) of the EMB status register is set, and the port input control flag bit EMB_STATx.PORTINFy ($x=0\sim6, y=1\sim4$) is set. An interrupt will be generated when EMB interrupt enable register EMB_INTENx.PORTINEN ($x=0\sim6$)=1.

The control event of EMB is selected by the EMB release mode selection register EMB_RLSSELx.PORTINRSELy ($x=0\sim6, y=1\sim4$) to output to Timer6 and Timer4 according to EMB_STATx.PORTINSTy ($x=0\sim6, y=1\sim4$) or EMB_STATx.PORTINFy ($x=0\sim6, y=1\sim4$). Timer6 and Timer4 can set the output port to high level, low level or high impedance state according to the register setting after receiving the control event.

If EMB_RLSSELx.PORTINRSELy ($x=0\sim6, y=1\sim4$)=1, when the port input level becomes invalid, EMB_STATx.PORTINSTy ($x=0\sim6, y=1\sim4$) will be automatically cleared, and the control event will be released immediately. If EMB_RLSSELx.PORTINRSELy ($x=0\sim6, y=1\sim4$)=0, when the port input level

becomes invalid, it is necessary to write the EMB status reset register EMB_STATCLR x .PORTINFCLR ($x=0\sim 6$) to clear EMB_STAT x .PORTI NF($x=0\sim 6$) for releasing the control event.

23.2.3 Stop PWM signal output when the level of PWM output port is in the same phase (same high or same low)

When using, firstly the EMB control register EMB_CTL1_ x .PWMSEN y ($x=0\sim 3$, $y=0\sim 7$ or $x=4\sim 6$, $y=0\sim 2$) is set to be valid, and select the effective level to be monitored through the EMB control register EMB_CTL2_ x .PWMLV y ($x=0\sim 3$, $y=0\sim 7$ or $x=4\sim 6$, $y=0\sim 2$).

After setting, EMB monitors the complementary PWM output signals of Timer6 and Timer4. When the output signals are the same high or the same low, the PWM output state EMB_STAT x .PWMST ($x=0\sim 6$) of the EMB status register is set, and the PWM output flag EMB_STAT x .PWMF ($x=0\sim 6$) is set. When EMB interrupt enable register EMB_INTEN x is set. An interrupt will be generated when PWMEN ($x=0\sim 6$)=1.

The control event of the EMB is selected by the EMB release mode selection register EMB_RLSSEL x .PWMRSEL ($x=0\sim 6$) to output to Timer6 and Timer4 according to EMB_STAT x .PWMST ($x=0\sim 6$) or EMB_STAT x .PWMF ($x=0\sim 6$). Timer6 and Timer4 can set the output port to high level, low level or high impedance state according to the register setting after receiving the control event.

If EMB_RLSSEL x .PWMRSEL($x=0\sim 6$)=1, when the PWM output is released from the same high or same low state, EMB_STAT x .PWMST($x=0\sim 6$) will be automatically cleared, and the control event will be released immediately. If EMB_RLSSEL x .PWMST($x=0\sim 6$)=0, when the PWM output is released from the same high or same low state, it is necessary to write the EMB state reset register EMB_STATCLR x .PWMFCLR ($x=0\sim 6$) to clear EMB_STAT x .PWMF ($x=0\sim 6$) for releasing the control event.

group0-3 can be used to monitor the complementary PWM output signal of Timer6, group4 is used to monitor the complementary PWM output signal of Timer4_1, group5 is used to monitor the complementary PWM output signal of Timer4_2, and group6 is used to monitor the complementary PWM output signal of Timer4_3.

Table 23-2 EMB Group Control Timer

Port name	Function	Correspond Group	EMB_CTL1 control bit	EMB_CTL2 control bit
TIM6_m_PWMA(m=1~8)	Complementary PWM output signal of Timer6	Group0 Group1 Group2 Group3	PWMSEN[7:0]	PWMLV[7:0]
TIM6_m_PWMB(m=1~8)				
TIM4_1_OUH	Complementary PWM output signal of Timer4_1	Group4	PWMSEN[2]	PWMLV[2]
TIM4_1_OUL			PWMSEN[1]	PWMLV[1]
TIM4_1_OVH			PWMSEN[0]	PWMLV[0]
TIM4_1_OVL				
TIM4_1_OWH				
TIM4_1_OWL				
TIM4_2_OUH	Complementary PWM output signal of Timer4_2	Group5	PWMSEN[2]	PWMLV[2]
TIM4_2_OUL			PWMSEN[1]	PWMLV[1]
TIM4_2_OVH			PWMSEN[0]	PWMLV[0]
TIM4_2_OVL				
TIM4_2_OWH				
TIM4_2_OWL				
TIM4_3_OUH	Complementary PWM output signal of Timer4_3	Group6	PWMSEN[2]	PWMLV[2]
TIM4_3_OUL			PWMSEN[1]	PWMLV[1]
TIM4_3_OVH			PWMSEN[0]	PWMLV[0]
TIM4_3_OVL				
TIM4_3_OWH				
TIM4_3_OWL				

23.2.4 Stop PWM signal output according to voltage comparator comparison result

Each group of EMB can send control event signals to Timer6 and Timer4 according to the 4 groups of comparison results of the voltage comparators. For the setting of the output result of the voltage comparator, please refer to the chapter on the voltage comparator.

When using, firstly the EMB control register EMB_CTL1_x.CMPENy ($x=0\sim6$) is set to be valid. When the voltage comparator comparison result flag is set, the EMB status register EMB_STATx.CMPST ($x=0\sim6$) is set, and the EMB status register EMB_STATx.CMPF ($x=0\sim6$) is set. An interrupt will be generated when CMPINTEN($x=0\sim6$)=1.

The control event of EMB is selected by EMB control PWM output release mode selection register EMB_RLSSELx.CMPRSELy ($x=0\sim6$) to output to Timer6 and Timer4 according to EMB_STATx.CMPST ($x=0\sim6$) or EMB_STATx.CMPF ($x=0\sim6$). Timer6 and Timer4 can set the output port to high level, low level or high impedance state according to the register setting after receiving the control event signal.

If EMB_RLSSELx.CMPRSEL($x=0\sim6$)=1, when the voltage comparator result becomes invalid, EMB_STATx.CMPST($x=0\sim6$) will be automatically cleared and the control event will be released immediately. If EMB_RLSSELx.CMPRSEL($x=0\sim6$)=0, when the voltage comparator result becomes invalid, you need to write the EMB status reset register EMB_STATCLRx.CMPFCLR($x=0\sim6$) to make EMB_STATx.CMPF ($x=0\sim6$) is cleared for releasing the control event .

23.2.5 Stop PWM signal output when external oscillator stops oscillating

Each group of EMB can send a notification signal to Timer6 and Timer4 when the external oscillator stops oscillating. Please refer to the CMU chapter for the setting of the external oscillator to stop oscillation.

When using, firstly the EMB control register EMB_CTL1_x.OSCSTPENy ($x=0\sim6$) is set to be valid. When the external oscillator stop oscillation flag is set, the oscillator state EMB_STATx.OSST($x=0\sim6$) of the EMB status register is set, and the EMB oscillator stops oscillation control flag EMB_STATx.OSF($x=0\sim6$) is set. An interrupt will also be generated when OSINTEN ($x=0\sim6$)=1.

The control event of EMB is selected by EMB release mode selection register EMB_RLSSELx.OSRSELy($x=0\sim6$) to output to Timer6 and Timer4 according to EMB_STATx.OSST($x=0\sim6$) or EMB_STATx.OSF ($x=0\sim6$). Timer6 and Timer4 can set the output port to high level, low level or high impedance state according to the register setting after receiving the control event signal.

If EMB_RLSSELx.OSRSEL($x=0\sim6$)=1, EMB_STATx.OSST ($x=0\sim6$) will be cleared automatically when the external oscillator stop oscillation flag becomes invalid, and the control event will be released immediately. If EMB_RLSSELx.OSRSEL($x=0\sim6$)=0, when the external oscillator stop oscillation flag

becomes invalid, it is necessary to write the EMB status reset register EMB_STATCLR x .OSFCLR($x=0\sim6$) to clear EMB_STAT x .OSF ($x=0\sim6$) for releasing the control event.

23.2.6 Write register software control PWM signal output

The EMB software output enable register (EMB_SOEx) ($x=0\sim6$) allows the user to send control signals to Timer6 and Timer4 by directly setting and clearing. When the software controls the PWM output, no interrupt request will be generated.

23.3 Register description

Table 23-3 List of EMB registers

Register Name	Symbol	Description	Offset address
EMB Control Register 1	EMB_CTL1	Enable PWM output control events and select port input valid level	0x0
EMB Control Register 2	EMB_CTL2	Filter of port input control events and select the active level of the PWM feedback signal	0x4
EMB Software Output Enable Register	EMB_SOE	Software generates PWM output control events	0x8
EMB Status Register	EMB_STAT	Indicate the state of PWM output control	0xC
EMB Status Clear Register	EMB_STATCLR	Clear the state of PWM output control	0x10
EMB Interrupt Enable Register	EMB_INTEN	Interrupt enable	0x14
EMB Release Select Register	EMB_RLSSEL	Select the release method for each PWM output control event	0x18

23.3.1 EMB Control Registers 1_0~3 (EMB_CTL1_0~3)

The register can only be written once after a reset

Address: 0x40017C00, 0x40017C20, 0x40017C40, 0x40017C60

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved				INVSEL4	INVSEL3	INVSEL2	INVSEL1	Reserved		PORTINEN4	PORTINEN3	PORTINEN2	PORTINEN1	PORTINEN1	PORTINEN1
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved		PWMSEN7	PWMSEN6	PWMSEN5	PWMSEN4	PWMSEN3	PWMSEN2	PWMSEN1	PWMSEN0	OSCSTPEN	CMPEN4	CMPEN3	CMPEN2	CMPEN1	CMPEN1
<hr/>															
Bit	Symbol	Bit name				Description					Read and write				
b31~b26	Reserved	-				Read "0" when reading, write "0" when writing					R/W				
b25	INVSEL4	Port 4 input active level selection				0: High level valid 1: Low level valid					R/W				
b24	INVSEL3	Port 3 input active level selection				0: High level valid 1: Low level valid					R/W				
b23	INVSEL2	Port 2 input active level selection				0: High level valid 1: Low level valid					R/W				
b22	INVSEL1	Port 1 input active level selection				0: High level valid 1: Low level valid					R/W				
b21~b20	Reserved	-				Read "0" when reading, write "0" when writing					R/W				
b19	PORTINEN4	Port 4 input control enable				0: Disable port input control 1: Enable port input control					R/W				
b18	PORTINEN3	Port 3 input control enable				0: Enable port input control 1: Disable port input control					R/W				
b17	PORTINEN2	Port 2 input control enable				0: Disable port input control 1: Enable port input control					R/W				
b16	PORTINEN1	Port 1 input control enable				0: Disable port input control 1: Enable port input control					R/W				
b15~b13	Reserved	-				Read "0" when reading, write "0" when writing					R/W				
b12	PWMSEN7	TIM6_8_PWMAB	in-phase	output	0: The output control is invalid when PWM is in-phase 1: The output control is valid when PWM is in-phase					R/W					
b11	PWMSEN6	TIM6_7_PWMAB	in-phase	output	0: The output control is invalid when PWM is in-phase 1: The output control is valid when PWM is in-phase					R/W					
b10	PWMSEN5	TIM6_6_PWMAB	in-phase	output	0: The output control is invalid when PWM is in-phase 1: The output control is valid when PWM is in-phase					R/W					
b9	PWMSEN4	TIM6_5_PWMAB	in-phase	output	0: The output control is invalid when PWM is in-phase 1: The output control is valid when PWM is in-phase					R/W					
b8	PWMSEN3	TIM6_4_PWMAB	in-phase	output	0: The output control is invalid when PWM is in-phase 1: The output control is valid when PWM is in-phase					R/W					
b7	PWMSEN2	TIM6_3_PWMAB	in-phase	output	0: The output control is invalid when PWM is in-phase 1: The output control is valid when PWM is in-phase					R/W					
b6	PWMSEN1	TIM6_2_PWMAB	in-phase	output	0: The output control is invalid when PWM is in-phase 1: The output control is valid when PWM is in-phase					R/W					
b5	PWMSEN0	TIM6_1_PWMAB	in-phase	output	0: The output control is invalid when PWM is in-phase					R/W					

			1: The output control is valid when PWM is in-phase	
b4	OSCSTPEN	Oscillator stop oscillating control enable	0: The output control is invalid when oscillator stops oscillating 1: The output control is valid when oscillator stops oscillating	R/W
b3	CMPEN4	CMP4 voltage comparator comparison result control enable	0: Voltage comparator output control is invalid 1: Voltage comparator output control is valid	R/W
b2	CMPEN3	CMP3 voltage comparator comparison result control enable	0: Voltage comparator output control is invalid 1: Voltage comparator output control is valid	R/W
b1	CMPEN2	CMP2 voltage comparator comparison result control enable	0: Voltage comparator output control is invalid 1: Voltage comparator output control is valid	R/W
b0	CMPEN1	CMP1 voltage comparator comparison result control enable	0: Voltage comparator output control is invalid 1: Voltage comparator output control is valid	R/W

23.3.2 EMB Control Registers 1_4~6 (EMB_CTL1_4~6)

The register can only be written once after a reset

Address: 0x40017C80, 0x40017CA0, 0x40017CC0

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved				INVSEL4	INVSEL3	INVSEL2	INVSEL1	Reserved		PORTINEN4	PORTINEN3	PORTINEN2	PORTINEN1		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				PWMSEN2	PWMSEN1	PWMSENO	OSCSTPEN	CMPEN4	CMPEN3	CMPEN2	CMPEN1				
<hr/>															
Bit	Symbol	Bit name			Description						Read and write				
b31~b26	Reserved	-			Read "0" when reading, write "0" when writing						R/W				
b25	INVSEL4	Port 4 input active level selection			0: High level valid 1: Low level valid						R/W				
b24	INVSEL3	Port 3 input active level selection			0: High level valid 1: Low level valid						R/W				
b23	INVSEL2	Port 2 input active level selection			0: High level valid 1: Low level valid						R/W				
b22	INVSEL1	Port 1 input active level selection			0: High level valid 1: Low level valid						R/W				
b21~b20	Reserved	-			Read "0" when reading, write "0" when writing						R/W				
b19	PORTINEN4	Port 4 input control enable			0: Disable port input control 1: Enable port input control						R/W				
b18	PORTINEN3	Port 3 input control enable			0: Disable port input control 1: Enable port input control						R/W				
b17	PORTINEN2	Port 2 input control enable			0: Disable port input control 1: Enable port input control						R/W				
b16	PORTINEN1	Port 1 input control enable			0: Disable port input control 1: Enable port input control						R/W				
b15~b8	Reserved	-			Read "0" when reading, write "0" when writing						R/W				
b7	PWMSEN2	TIM4_m_OUH/L in-phase output control enable (m=1~3)			0: The output control is invalid when PWM is in-phase 1: The output control is valid when PWM is in-phase						R/W				
b6	PWMSEN1	TIM4_m_OVH/L in-phase output control enable (m=1~3)			0: The output control is invalid when PWM is in-phase 1: The output control is valid when PWM is in-phase						R/W				
b5	PWMSENO	TIM4_m_OWH/L in-phase output control enable (m=1~3)			0: The output control is invalid when PWM is in-phase 1: The output control is valid when PWM is in-phase						R/W				
b4	OSCSTPEN	Oscillator stop oscillating control enable			0: The output control is invalid when oscillator stops oscillating 1: The output control is valid when oscillator stops oscillating						R/W				
b3	CMPEN4	CMP4 voltage comparator comparison result control enable			0: Voltage comparator output control is invalid 1: Voltage comparator output control is valid						R/W				
b2	CMPEN3	CMP3 voltage comparator comparison result control enable			0: Voltage comparator output control is invalid 1: Voltage comparator output control is valid						R/W				
b1	CMPEN2	CMP2 voltage comparator comparison result control enable			0: Voltage comparator output control is invalid 1: Voltage comparator output control is valid						R/W				
b0	CMPEN1	CMP1 voltage comparator comparison result control enable			0: Voltage comparator output control is invalid 1: Voltage comparator output control is valid						R/W				

23.3.3 EMB Control Registers 2_0~3 (EMB_CTL2_0~3)

The register can only be written once after a reset

Address: 0x40017C04, 0x40017C24, 0x40017C44, 0x40017C64

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved			NFEN4	NFSEL4[1:0]		NFEN3	NFSEL3[1:0]		NFEN2	NFSEL2[1:0]		NFEN1	NFSEL1[1:0]		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved						PWML V7	PWML V6	PWML V5	PWML V4	PWML V3	PWML V2	PWML V1	PWML V0		

Bit	Symbol	Bit name	Description	Read and write
b31~28	Reserved	-	Read "0" when reading, write "0" when writing	R/W
b27	NFEN4	Port 4 input digital filter enable	0: The filerter is invalid 1: The filerter is valid	R/W
b26~b25	NFSEL4[1:0]	Port 4 input digital filter clock selection	00: PCLK1 clock 01: PCLK1/8 clock 10: PCLK1/32 clock 11: PCLK1/128 clock	R/W
b24	NFEN3	Port 3 input digital filter enable	0: The filerter is invalid 1: The filerter is valid	R/W
b23~b22	NFSEL3[1:0]	Port 3 input digital filter clock selection	00: PCLK1 clock 01: PCLK1/8 clock 10: PCLK1/32 clock 11: PCLK1/128 clock	R/W
b21	NFEN2	Port 2 input digital filter enable	0: The filerter is invalid 1: The filerter is valid	R/W
b20~b19	NFSEL2[1:0]	Port 2 input digital filter clock selection	00: PCLK1 clock 01: PCLK1/8 clock 10: PCLK1/32 clock 11: PCLK1/128 clock	R/W
b18	NFEN1	Port 1 input digital filter enable	0: The filerter is invalid 1: The filerter is valid	R/W
b17~b16	NFSEL1[1:0]	Port 1 input digital filter clock selection	00: PCLK1 clock 01: PCLK1/8 clock 10: PCLK1/32 clock 11: PCLK1/128 clock	R/W
b15~b8	Reserved	-	Read "0" when reading, write "0" when writing	R/W
b7	PWMLV7	TIM6_8_PWMA/B output active level selection	0: Low level valid 1: High level valid	R/W
b6	PWMLV6	TIM6_7_PWMA/B output active level selection	0: Low level valid 1: High level valid	R/W
b5	PWMLV5	TIM6_6_PWMA/B output active level selection	0: Low level valid 1: High level valid	R/W
b4	PWMLV4	TIM6_5_PWMA/B output active level selection	0: Low level valid 1: High level valid	R/W
b3	PWMLV3	TIM6_4_PWMA/B output active valid level selection	0: Low level valid 1: High level valid	R/W
b2	PWMLV2	TIM6_3_PWMA/B output active level selection	0: Low level valid 1: High level valid	R/W
b1	PWMLV1	TIM6_2_PWMA/B output active level selection	0: Low level valid 1: High level valid	R/W
b0	PWMLV0	TIM6_1_PWMA/B output active level selection	0: Low level valid 1: High level valid	R/W

23.3.4 EMB Control Registers 2_4~6 (EMB_CTL2_4~6)

The register can only be written once after a reset

Address: 0x40017C84, 0x40017CA4, 0x40017CC4

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved		NFEN4	NFSEL4[1:0]	NFEN3	NFSEL3[1:0]	NFEN2	NFSEL2[1:0]	NFEN1	NFSEL1[1:0]						
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															
PWML V2	PWML V1	PWML V0													

Bit	Symbol	Bit name	Description	Read and write
b31~28	Reserved	-	Read "0" when reading, write "0" when writing	R/W
b27	NFEN4	Port 4 Input Digital Filter Enable	0: The filerter is invalid 1: The filerter is valid	R/W
b26~b25	NFSEL4[1:0]	Port 4 input digital filter clock selection	00: PCLK1 clock 01: PCLK1/8 clock 10: PCLK1/32 clock 11: PCLK1/128 clock	R/W
b24	NFEN3	Port 3 input digital filter enable	0: The filerter is invalid 1: The filerter is valid	R/W
b23~b22	NFSEL3[1:0]	Port 3 input digital filter clock selection	00: PCLK1 clock 01: PCLK1/8 clock 10: PCLK1/32 clock 11: PCLK1/128 clock	R/W
b21	NFEN2	Port 2 input digital filter enable	0: The filerter is invalid 1: The filerter is valid	R/W
b20~b19	NFSEL2[1:0]	Port 2 input digital filter clock selection	00: PCLK1 clock 01: PCLK1/8 clock 10: PCLK1/32 clock 11: PCLK1/128 clock	R/W
b18	NFEN1	Port 1 input digital filter enable	0: The filerter is invalid 1: The filerter is valid	R/W
b17~b16	NFSEL1[1:0]	Port 1 input digital filter clock selection	00: PCLK1 clock 01: PCLK1/8 clock 10: PCLK1/32 clock 11: PCLK1/128 clock	R/W
b15~b3	Reserved	-	Read "0" when reading, write "0" when writing	R/W
b2	PWMLV2	TIM4_m_OUH/L output active level selection (m=1~3)	0: Low level valid 1: High level valid	R/W
b1	PWMLV1	TIM4_m_OVH/L output active level selection (m=1~3)	0: Low level valid 1: High level valid	R/W
b0	PWMLV0	TIM4_m_OWH/L output active level selection (m=1~3)	0: Low level valid 1: High level valid	R/W

23.3.5 EMB Software Output Enable Register (EMB_SOEx) (x=0~6)

Address: 0x40017C08, 0x40017C28, 0x40017C48, 0x40017C68, 0x40017C88, 0x40017CA8, 0x40017CC8

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															
SOE															
Bit	Symbol	Bit name	Description										Read and write		
b31~b1	Reserved	-	Read "0" when reading, write "0" when writing										R/W		
b0	SOE	Software control output	0: PWM normal output 1: PWM stop output										R/W		

23.3.6 EMB Status Register (EMB_STATx) (x=0~6)

Address: 0x40017C0C, 0x40017C2C, 0x40017C4C, 0x40017C6C, 0x40017C8C, 0x40017CAC,
0x40017CCC

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16		
Reserved															PORTI NST4	PORTI NST3	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0		
PORT INST 2	PORTI NST1	Reserved	PORTI NF4	PORTI NF3	PORTI NF2	PORTI NF1	OSST	CMPST	PWM ST	Reser ved	OSF	CMPF	PWM SF	Reser ved			
Bit	Symbol	Bit name	Description												Read and write		
b31~18	Reserved	-	Read "0" when reading, write "0" when writing												R/W		
b17	PORTINST4	Port 4 input control status	0: Port 4 input control is in the inactive level state 1: Port 4 input control is in the active level state												R		
b16	PORTINST3	Port 3 input control status	0: Port 3 input control is in the inactive level state 1: Port 3 input control is in the active level state												R		
b15	PORTINST2	Port 2 input control status	0: Port 2 input control is in the inactive level state 1: Port 2 input control is in the active level state												R		
b14	PORTINST1	Port 1 input control status	0: Port 1 input control is in the inactive level state 1: Port 1 input control is in the active level state												R		
b13~b12	Reserved	-	Read "0" when reading, write "0" when writing												R/W		
b11	PORTINF4	Port 4 input control flag	Set to 1 when EMB_STATx.PORTINST4 is 1 Cleared to 0 when EMB_STATCLRx.PORTINFCLR4 is written to 1												R		
b10	PORTINF3	Port 3 input control flag	Set to 1 when EMB_STATx.PORTINST3 is 1 Cleared to 0 when EMB_STATCLRx.PORTINFCLR3 is written to 1												R		
b9	PORTINF2	Port 2 input control flag	Set to 1 when EMB_STATx.PORTINST2 is 1 Cleared to 0 when EMB_STATCLRx.PORTINFCLR2 is written to 1												R		
b8	PORTINF1	Port 1 input control flag	Set to 1 when EMB_STATx.PORTINST1 is 1 Cleared to 0 when EMB_STATCLRx.PORTINFCLR1 is written to 1												R		
b7	OSST	Oscillator Status	0: The oscillator has not stopped oscillating 1: The oscillator stops oscillating												R		
b6	CMPST	Voltage Comparator Status	Voltage comparator comparison results												R		
b5	PWMST	PWM output state	0: No PWM output in same phase 1: PWM output in same phase												R		
b4	Reserved	-	Read "0" when reading, write "0" when writing												R/W		
b3	OSF	Oscillator stop oscillation control flag	Set to 1 when EMB_STATx.OSST is 1 Cleared to 0 when EMB_STATCLRx.OSFCLR is written to 1												R		
b2	CMPF	Voltage Comparator Control Flag	Set to 1 when EMB_STATx.CMPST is 1 Cleared to 0 when EMB_STATCLRx.CMPFCLR is written to 1												R		
b1	PWMSF	PWM output in-phase control flag bit	Set to 1 when EMB_STATx.PWMST is 1 Cleared to 0 when EMB_STATCLRx.PWMSCLR is written to 1												R		
b0	Reserved	-	Read "0" when reading, write "0" when writing												R/W		

23.3.7 EMB Status Clear Register (EMB_STATCLR x) ($x=0\sim6$)

Address: 0x40017C10, 0x40017C30, 0x40017C50, 0x40017C70, 0x40017C90, 0x40017CB0, 0x40017CD0

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				PORTI NFCL R4	PORTI NFCL R3	PORTI NFCL R2	PORTI NFCL R1	Reserved				OSFC LR	CMPF CLR	PWM SFCL R	Reser ved				
<hr/>																			
Bit	Symbol	Bit name			Description								Read and write						
b31~b12	Reserved	-			Read "0" when reading, write "0" when writing								R/W						
b11	PORTINFCLR4	Clear flag EMB_STAT.PORTINF4			0: Nothing 1: When EMB_STAT.PORTINST4=0, EMB_STAT.PORTINF4 is cleared to 0 Reading this register bit has no effect								R/W						
b10	PORTINFCLR3	Clear flagEMB_STAT.PORTINF3			0: Nothing 1: When EMB_STAT.PORTINST3=0, EMB_STAT.PORTINF3 is cleared to 0 Reading this register bit has no effect								R/W						
b9	PORTINFCLR2	Clear flagEMB_STAT.PORTINF2			0: Nothing 1: When EMB_STAT.PORTINST2=0, EMB_STAT.PORTINF2 is cleared to 0 Reading this register bit has no effect								R/W						
b8	PORTINFCLR1	Clear flagEMB_STAT.PORTINF1			0: Nothing 1: When EMB_STAT.PORTINST1=0, EMB_STAT.PORTINF1 is cleared to 0 Reading this register bit has no effect								R/W						
b7~b4	Reserved	-			Read "0" when reading, write "0" when writing								R/W						
b3	OSFCLR	Clear flagEMB_STAT.OSF			0: Nothing 1: When EMB_STAT.OSST=0, EMB_STAT.OSF is cleared to 0 Reading this register bit has no effect								R/W						
b2	CMPFCLR	Clear flagEMB_STAT.CMPF			0: Nothing 1: When EMB_STAT.CMPST=0, EMB_STAT.CMPF is cleared to 0 Reading this register bit has no effect								R/W						
b1	PWMSCLR	Clear flagEMB_STAT.PWMSF			0: Nothing 1: When EMB_STAT.PWMST=0, EMB_STAT.PWMSF is cleared to 0 Reading this register bit has no effect								R/W						
b0	Reserved	-			Read "0" when reading, write "0" when writing								R/W						

23.3.8 EMB Interrupt Enable Register (EMB_INTENx) (x=0~6)

Address: 0x40017C14, 0x40017C34, 0x40017C54, 0x40017C74, 0x40017C94, 0x40017CB4, 0x40017CD4

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				PORTINTEN4	PORTINTEN3	PORTINTEN2	PORTINTEN1					OSINTEN	CMPINTEN	PWMSINTEN	Reserved
Reserved															
Bit	Symbol	Bit name	Description										Read and write		
b31~b12	Reserved	-	Read "0" when reading, write "0" when writing										R/W		
b11	PORINTEN4	Port 4 input control interrupt enable	0: Don't generate interrupt when EMB_STATx.PORTINF4=1 1: Generate interrupt when EMB_STATx.PORTINF4=1										R/W		
b10	PORINTEN3	Port 3 input control interrupt enable	0: Don't generate interrupt when EMB_STATx.PORTINF3=1 1: Generate interrupt when EMB_STATx.PORTINF3=1										R/W		
b9	PORINTEN2	Port 2 input control interrupt enable	0: Don't generate interrupt when EMB_STATx.PORTINF2=1 1: Generate interrupt when EMB_STATx.PORTINF2=1										R/W		
b8	PORINTEN1	Port 1 input control interrupt enable	0: Don't generate interrupt when EMB_STATx.PORTINF1=1 1: Generate interrupt when EMB_STATx.PORTINF1=1										R/W		
b7~b4	Reserved	-	Read "0" when reading, write "0" when writing										R/W		
b3	OSINTEN	Oscillator stop oscillation control interrupt enable	0: Don't generate interrupt when EMB_STATx.OSF=1 1: Generate interrupt when EMB_STATx.OSF=1										R/W		
b2	CMPINTEN	Voltage comparator comparison result control interrupt enable	0: Don't generate interrupt when EMB_STATx.CMPF=1 1: Generate interrupt when EMB_STATx.CMPF=1										R/W		
b1	PWMSINTEN	PWM output in-phase control interrupt enable	0: Don't generate interrupt when EMB_STATx.PWMSF=1 1: Generate interrupt when EMB_STATx.PWMSF=1										R/W		
b0	Reserved	-	Read "0" when reading, write "0" when writing										R/W		

23.3.9 EMB Release Select Register (EMB_RLSSELx) (x=0~6)

Address: 0x40017C18, 0x40017C38, 0x40017C58, 0x40017C78, 0x40017C98, 0x40017CB8, 0x40017CD8

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				PORTI NRSE L4	PORTI NRSE L3	PORTI NRSE L2	PORTI NRSE L1		Reserved		OSRS EL	CMPR SEL	PWM RSEL	Reser ved	
<hr/>															
Bit	Symbol	Bit name	Description											Read and write	
b31~b12	Reserved	-	Read "0" when reading, write "0" when writing											R/W	
b11	PORTINRSEL4	Port 4 input control release mode selection	0: Release PWM output control when EMB_STATx.PORTINF4=0 1: Release PWM output control when EMB_STATx.PORTINST4=0											R/W	
b10	PORTINRSEL3	Port 3 input control release mode selection	0: Release PWM output control when EMB_STATx.PORTINF3=0 1: Release PWM output control when EMB_STATx.PORTINST3=0											R/W	
b9	PORTINRSEL2	Port 2 input control release mode selection	0: Release PWM output control when EMB_STATx.PORTINF2=0 1: Release PWM output control when EMB_STATx.PORTINST2=0											R/W	
b8	PORTINRSEL1	Port 1 input control release mode selection	0: Release PWM output control when EMB_STATx.PORTINF1=0 1: Release PWM output control when EMB_STATx.PORTINST1=0											R/W	
b7~b4	Reserved	-	Read "0" when reading, write "0" when writing											R/W	
b3	OSRSEL	Oscillator stop oscillation control release mode selection	0: Release PWM output control when EMB_STATx.OSF=0 1: Release PWM output control when EMB_STATx.OSST=0											R/W	
b2	CMPRSEL	Comparator comparison result control release mode selection	0: Release PWM output control when EMB_STATx.CMPF=0 1: Release PWM output control when EMB_STATx.CMPST=0											R/W	
b1	PWMRSEL	PWM output control release mode selection	0: Release PWM output control when EMB_STATx.PWMSF=0 1: Release PWM output control when EMB_STATx.PWMST=0											R/W	
b0	Reserved	-	Read "0" when reading, write "0" when writing											R/W	

24 General Timer (TimerA)

24.1 Introduction

General Timer A (Timer A) is a timer with 16-bit count width and 4 channels PWM output. The timer supports two waveform modes of triangular wave and sawtooth wave, and can generate various PWM waveforms (unilaterally aligned PWM, bilaterally symmetrical PWM); supports synchronous start of counters; the Compare Value Register support the buffer function; support cascading between units to realize 32-bit counting; supports 2-phase quadrature encoding counting and 3-phase quadrature encoding counting. This series of products is equipped with 12 units of TimerA, which can achieve a maximum of 48 channels PWM output.

24.2 Basic block diagram

The basic functions and features of TimerA are shown in Table 24-1.

Table 24-1 Basic functions and features of TimerA

Waveform mode	Sawtooth wave, triangular wave
Basic functions	• Direction of increments and decrements
	• Synchronous start counter
	• Compare reference value buffer function
	• 32-bit cascade count
	• Quadrature encoding counting
	• 4-Channel PWM Output
	• Compare Matched Event Output
Interrupt type	• Compare match interrupt
	• Periodic matching interrupt

The basic block diagram of TimerA is shown in Figure 24-1. In the figure, "<t>" is the unit number, that is, "<t>" is 1~12. When referring to "<t>" later in this chapter, it refers to the unit number, and will not be repeated.

This series of products are equipped with 12 units of TimerA, in which the bus clock and count clock of unit 1 to unit 4 are PCLK0 (at this time, the PCLK shown in the block diagram refers to PCLK0); the bus clock and count clock of unit 5 to unit 12 are PCLK1 (at this time, the PCLK shown in the block diagram refers to PCLK1). The count frequency division clock source of each unit and the sampling reference clock source of port digital filtering are the same as its bus clock and count clock.

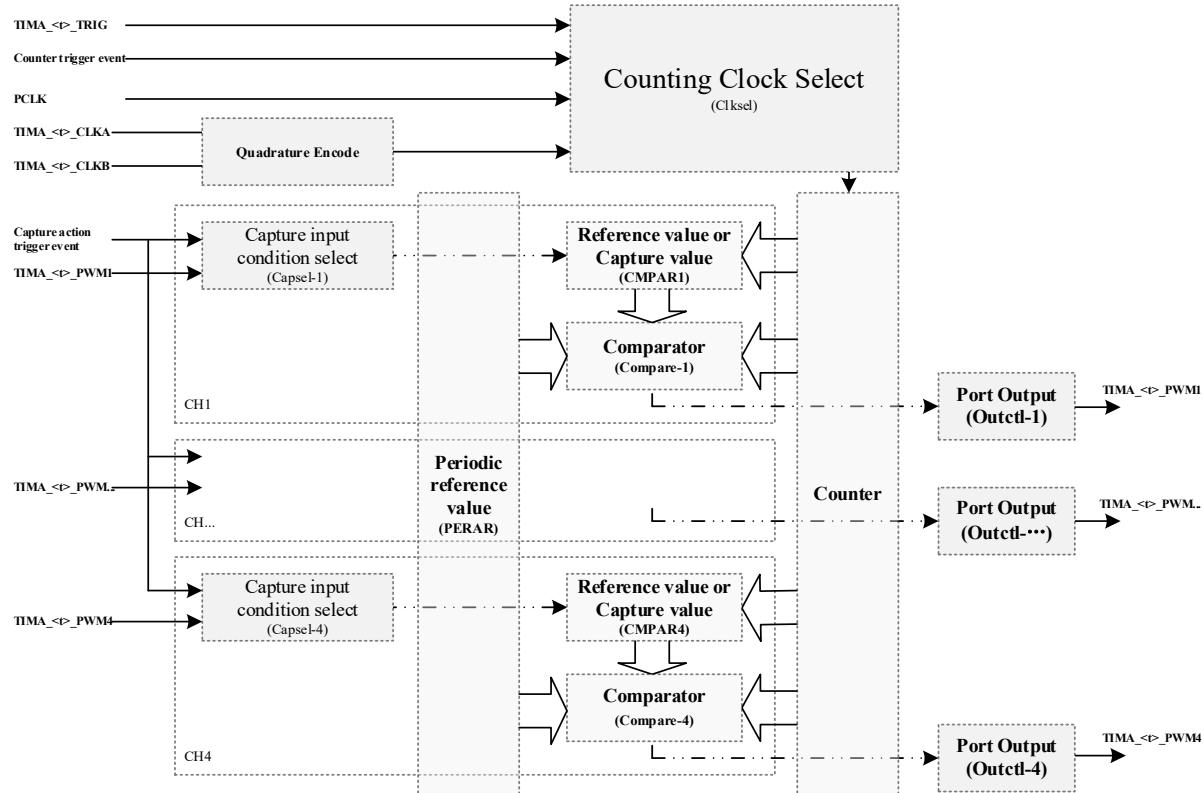


Figure 24-1 TimerA basic block diagram

Table 24-2 shown is the list of input and output ports of TimerA.

Table 24-2 TimerA port list

Port name	Direction	Function
TIMA_<t>_PWMr	in or out	Capture input event port or PWM output port (m=1~4)
TIMA_<t>_CLKA	in	Quadrature Coded Count Event Input Port
TIMA_<t>_CLKB		
TIMA_<t>_TRIG	in	Hardware Trigger Startup, Stop, Clear Event Input Port

24.3 Functional description

24.3.1 Waveform mode

TimerA has two basic counting wave modes: Sawtooth wave mode and triangular wave mode. The basic waveforms of the two waveform modes are follow in Figure 24-2, Figure 24-3.

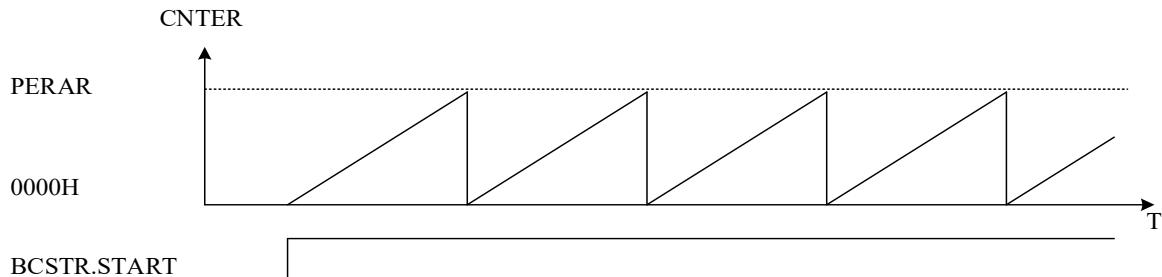


Figure 24-2 sawtooth waveform (increasing counting)

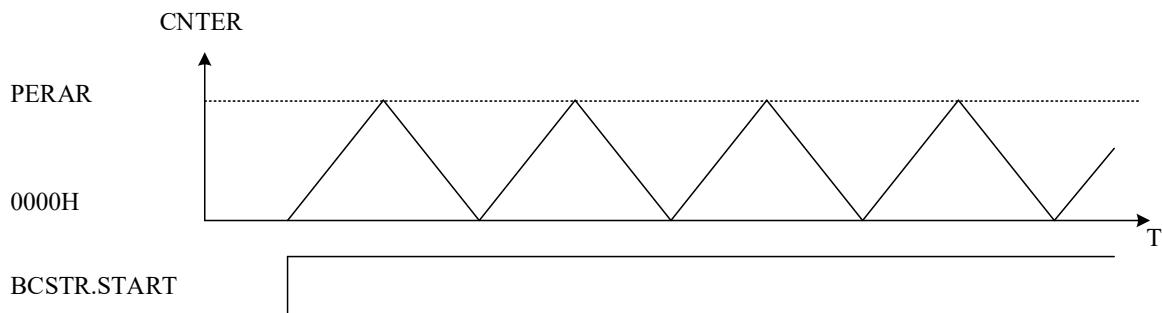


Figure 24-3 triangular wave

24.3.2 Clock source selection

TimerA's counting clock has the following options:

- a) 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 frequency division of PCLK (set by BCSTR.CKDIV[3:0])
- 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] settings)
- d) Count overflow or underflow event input for symmetric units (HCUPR [12: 11] or HCDOR [12: 11] settings)
- e) TIMA_<t>_CLKA, TIMA_<t>_CLKB port quadrature code input (HCUPR [7: 0] or HCDOR [7: 0] settings)

When the counting clock source selects a, it is the software counting mode, and when the counting clock source selects b, c, d and e, it is the hardware counting mode. When the counting clock is selected as d, it is mostly used for the revolution counting mode of three-phase quadrature encoding counting (refer to the chapters [Position Overflow Counting] and [Mixed Counting]), and can also be used for cascade counting. The above description shows that the clocks b, c, d and e

are independent of each other, can be set to be valid or invalid respectively, and the clock a is automatically invalid when clocks b, c, d and e are selected.

24.3.3 Compare output

Each TimerA unit contains 4-channel compare output (TIMA_<t>_PWMn), that outputs the specified level when the count value matches the compare reference value. TMRA_CMPARn register corresponds to the count compare reference value of TIMA_<t>_PWMn output port, respectively. When the count value of the timer is equal to TMRA_CMPARn, the TIMA_<t>_PWMn port outputs the specified level (n=1~4).

TIMA_<t>_PWMn port's count start level, count stop level, count compare match level, count cycle match level, etc., can be controlled through the STAC, STPC, CMPC, PERC, FORC bit of Port Control Register (PCONRn) (n=1~4). Figure 24-4 is an example of the compare output operation of unit 1.

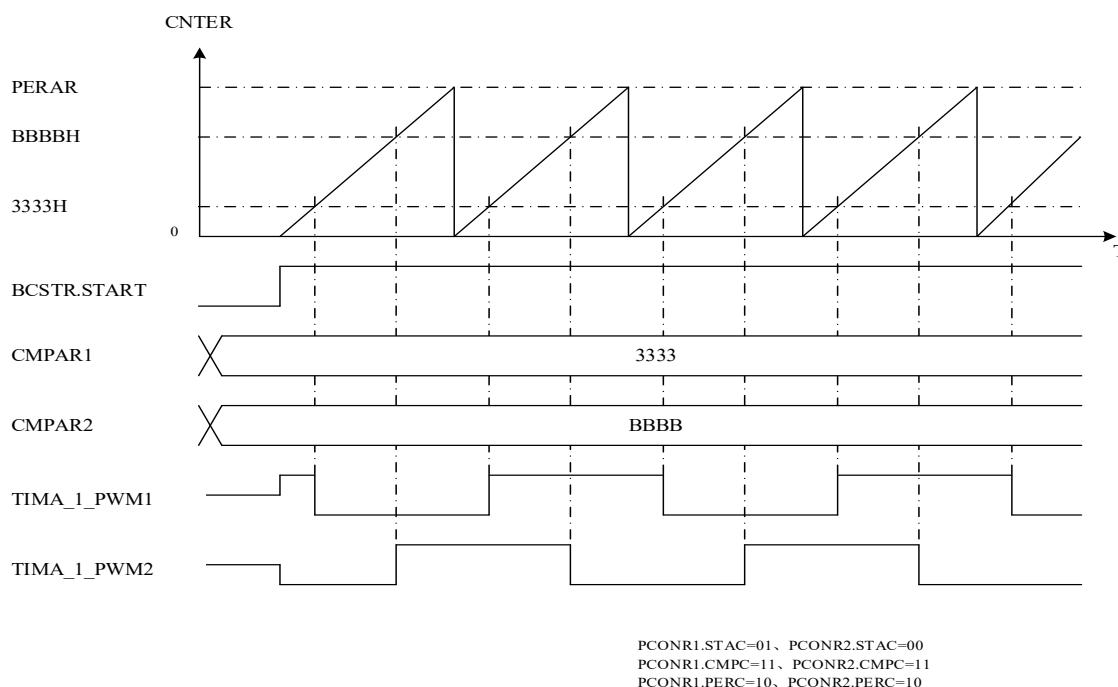


Figure 24-4 Compare output action

24.3.4 Capture input

Each PWM output channel of each TimerA unit has a capture input function to store the captured count values. Set the CCONR.CAPMD bit of the Capture Control Register (CCONRn) to 1 to make the capture input function valid. At this time, when the corresponding capture input condition is selected and the condition is valid, the current count value is saved to the corresponding register (CMPARn) (n=1~4).

The capture input condition can select the internal capture action trigger event (selected by the HTSSR0~3 registers, refer to the register description chapter for details), TIMA_<t>_PWMn port

input, etc. The specific condition selection can be selected by the HICP bit of the Capture Control Register (CCONRn) (n=1~4). Figure 24-5 is an example of capture input action.

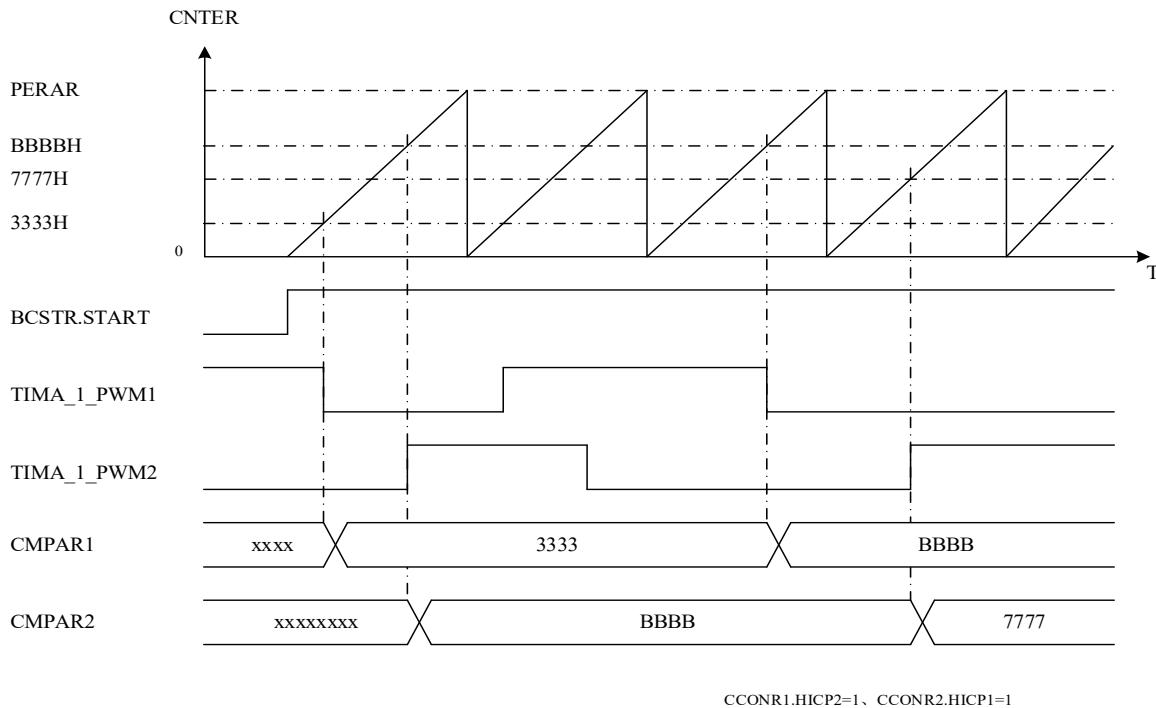


Figure 24-5 Capture input action

24.3.5 Synchronous startup

This product is equipped with 12 units of TimerA, which can realize software synchronous startup or hardware synchronous startup. When this unit is unit m, unit m can be selected to start synchronously with unit n (when m=2, 4, 6, 8, 10, 12, n=1, 3, 5, 7, 9, 11).

When the BCSTR.SYNST bit in unit m is set to 1, the synchronization function between unit m and unit n is effective. In this case, if the BCSTR.START bit of the unit n is 1, the counter of the synchronized unit (unit m) starts software synchronization counting. If any bit in the HCONR.HSTA1~0 of the hardware setting unit n is 1, and the corresponding hardware event of the unit n occurs, the counter of the synchronization unit (unit m) starts hardware synchronization counting. When the hardware synchronization count start function is selected, the corresponding bits of HCONR.HSTA1~0 of the synchronized unit (unit m) must also be set to be valid (when m=2, 4, 6, 8, 10, 12, n= 1, 3, 5, 7, 9, 11).

Figure 24-6 is an example of software synchronization startup when set BCSTR.SYNST=1 of unit 4.

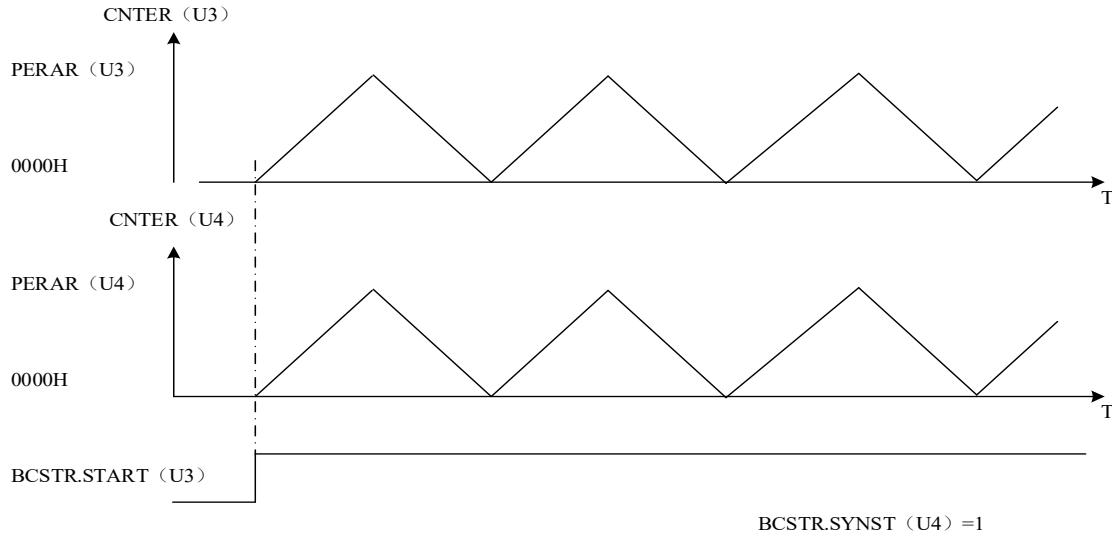


Figure 24-6 Software synchronization action

24.3.6 Digital filtering

Each TimerA unit input ports of TIMA_<t>_CLKA, TIMA_<t>_CLKB, TIMA_<t>_TRIG and TIMA_<t>_PWMr (capture input function) have digital filtering functions. The enabling of the filtering function of each port and the selection of the filtering clock can be realized by setting the corresponding bits of the Filter Control Register (FCONR) and the Capture Control Register (CCONRn) ($n=1\sim 4$).

When the filter sampling reference clock samples the same level three times on the port, the level is transferred to the module as an effective level. A level less than three times consistent will be filtered out as external interference and will not be transmitted to the inside of the module. Figure 24-7 is an example of the filtering action of the TIMA_1_CLKA port.

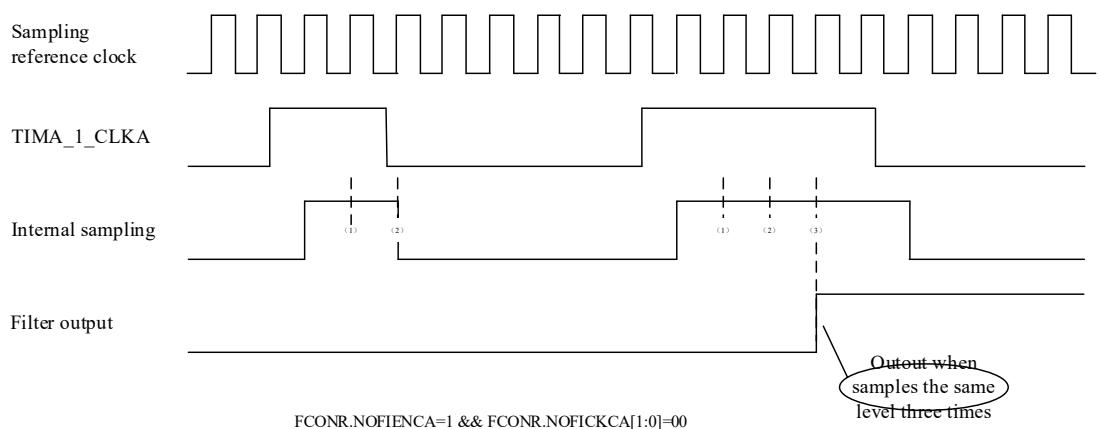


Figure 24-7 Filtering function of clock input port

24.3.7 Buffer function

A total of 4 Compare Value Registers (CMPARn) of TimerA can realize the buffer function in pairs ($n=1\sim 4$). That is, CMPAR2 is used as the buffer reference value of CMPAR1, and CMPAR4 is used as the buffer reference value of CMPAR3. The Buffer Control Register (BCONRm) respectively realizes the control of the four groups of buffer functions ($m=1\sim 2$).

When the BEN bit in the Buffer Control Register (BCONRm) is set, the buffer function becomes active ($m=1\sim 2$). A buffer transfer (CMPAR4/2->CMPAR3/1) occurs when the counter counts up to a certain point in time. The "specific point in time" has the following circumstances:

- In hardware counting mode, count to overflow point (BCSTR.DIR = 1) or underflow point (BCSTR.DIR = 0)
- In sawtooth wave counting mode (BCSTR.MODE = 0), the counter counts to the overflow point (BCSTR.DIR = 1) or to the underflow point (BCSTR.DIR = 0).
- When triangular wave counting mode (BCSTR.MODE=1), count to the peak point (when BCSTR.DIR=1 && BCONRn.BSE0=1) ($n=1\sim 2$)
- When triangular wave counting mode (BCSTR.MODE=1), count to the valley point (when BCSTR.DIR=0 && BCONRn.BSE1=1) ($n=1\sim 2$)
- In hardware counting mode or sawtooth wave counting mode, clear action occurs.

Figure24-8 is a schematic diagram of buffer transmission in sawtooth mode.

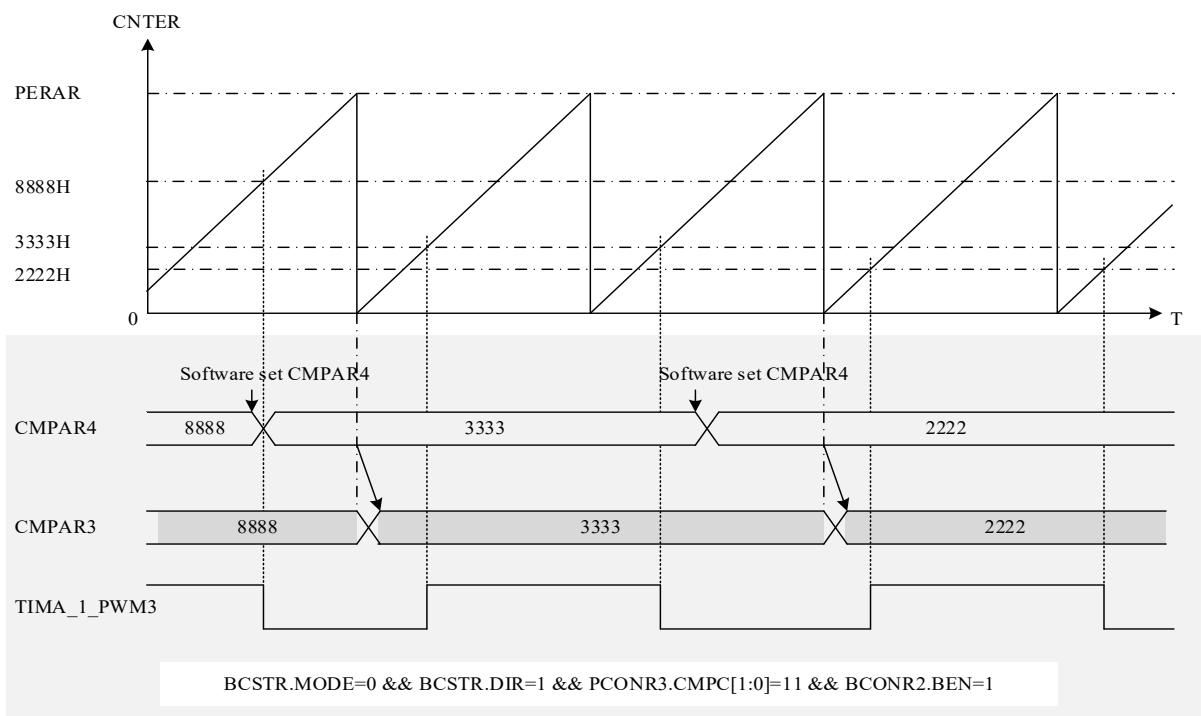


Figure 24-8 Buffer action in sawtooth mode

24.3.8 Cascade count

In the counting clock source selection chapter, when the clock source is selected d), the counting clock of this unit is selected as the count overflow (count overflow or count underflow) event of the symmetric unit, at this time, the two unit levels are combined and can realize 32-bit counter. In the cascade 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, when the triangular wave counts up mode (BCSTR.MODE=0, BCSTR.DIR=1), set the counting clock of unit 1 as PCLK, and set the counting clock source of unit 2 as the count overflow event of unit 1 (unit 2's TMRA_HCUPR.HCUP11=1), start the counting of units 1 and 2 (start unit 2 first, then start unit 1) to realize cascade counting. The CNTER of the unit 1 is the low 16-bit counter and the CNTER of unit 2 is the high 16-bit counter.

Figure 24-9 is a schematic diagram of the cascade count of units 1 and 2.

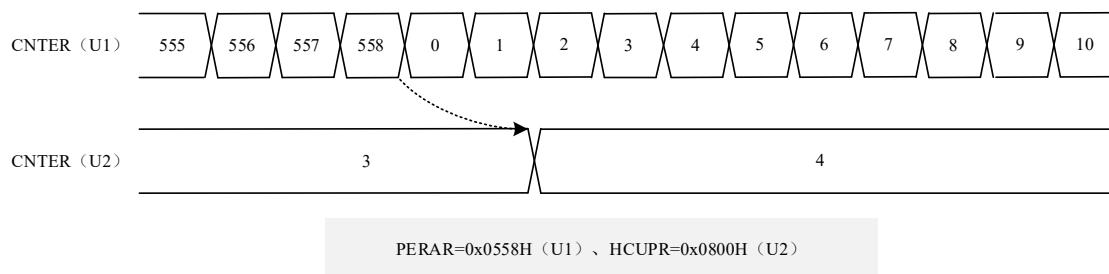


Figure 24-9 32-bit cascade counting action

24.3.9 PWM output

24.3.9.1 Unilaterally aligned PWM output

In sawtooth wave counting mode, each channel in a unit can be set by various port control settings to realize unilaterally aligned PWM output (aligned at counting cycle point). For example, by setting to toggle when the compare value matches (PCONR.CMPC=11) and to toggle when the period value matches (PCONR.PERC=11), the PWM output with unilaterally alignment can be generated in one cycle.

Figure 24-10 is an example of unilaterally aligned PWM output waveform in sawtooth mode.

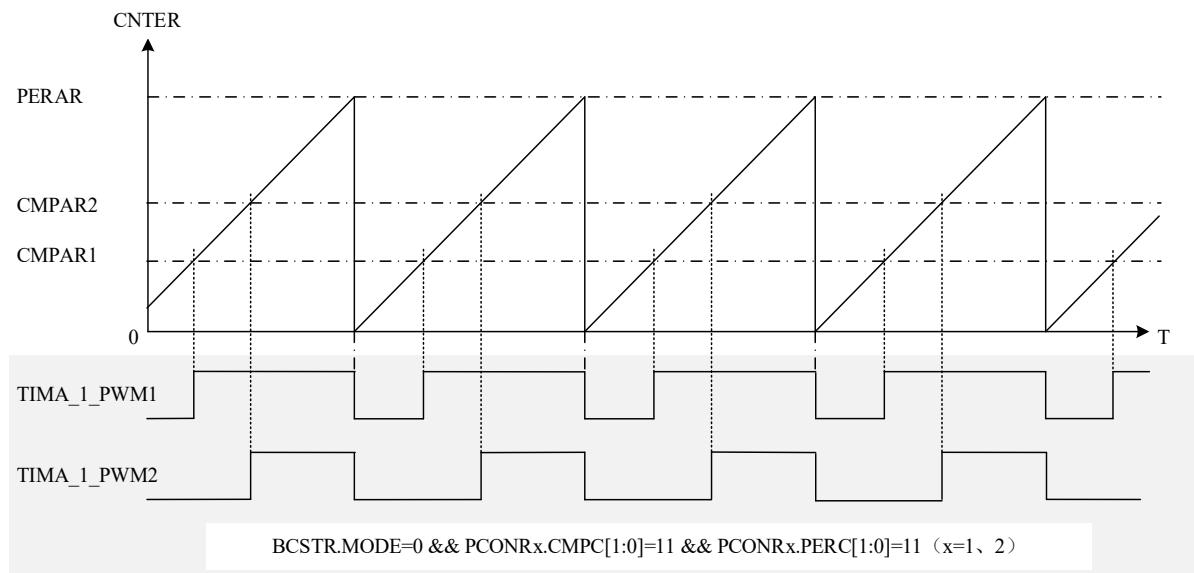


Figure 24-10 unilaterally aligned PWM output example

24.3.9.2 Bilateral symmetric PWM output

In the triangular mode, each channel in a unit can be set by various port controls to realize bilateral symmetric PWM output (symmetrical with counting peak). Independent PWM output or complementary PWM output can be realized according to the output relationship between channels.

Figure 24-11 is an example of bilateral symmetrical PWM output waveforms of channels 1, 2, 3, and 4 in triangular wave mode. Among them, channels 1 and 2 can be used as a pair of complementary PWM outputs.

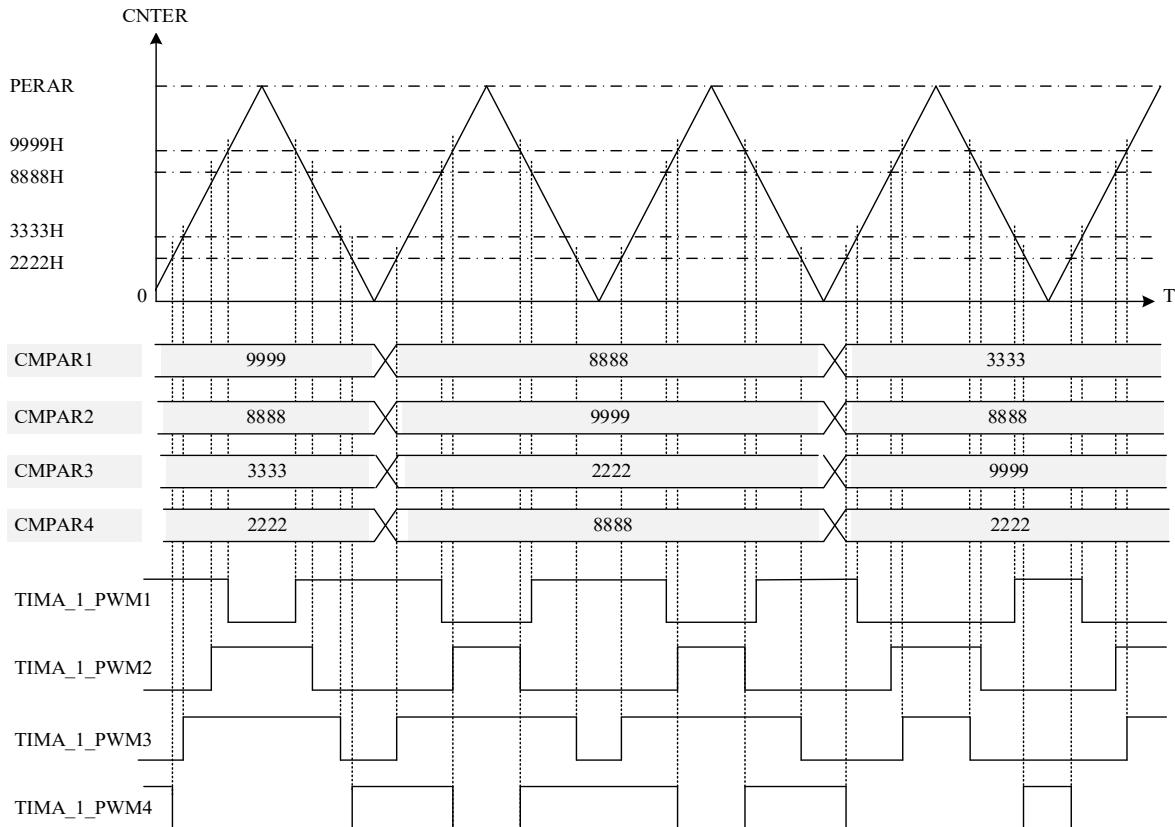


Figure 24-11 Bilateral Symmetrical PWM Output Example

24.3.10 Quadrature encoding counting

The TIMA_<t>_CLKA input is regarded as the AIN input, the TIMA_<t>_CLKB input is regarded as the BIN input, and the TIMA_<t>_TRIG input is regarded as the ZIN input, and TimerA can realize the quadrature encoding count of the three inputs.

The single action of AIN and BIN of each unit can realize the position counting mode; the combined action of AIN, BIN and ZIN of the two units can realize the revolution counting mode, and the unit used for position counting is called the position counting unit, the unit used for revolution counting is called revolution counting unit. In revolution counting mode, every two units are combined with each other (unit 1, 2; unit 3, 4; unit 5, 6; unit 7, 8; unit 9, 10; unit 11, 12). The position count unit and revolution count unit within the combination can be arbitrarily specified.

The counting conditions of AIN and BIN are enabled by setting the quadrature relationship between TIMA_<t>_CLKA and TIMA_<t>_CLKB in the Hardware Count Up Event Select Register (HCUPR) and Hardware Count Down Event Select Register (HCDOR); ZIN's Input action is through the clearing of the position timer by setting the clear enable bit of the Hardware Control Event Select Register (HCONR) of the position counting unit, and the counting of the revolution timer by setting the Hardware Count Up Event Select Register (HCUPR) of the revolution unit.

24.3.10.1 Position count mode

The quadrature encoding position counting mode refers to realizing the basic counting function, phase difference counting function and direction counting function according to the input of AIN and BIN.

Basic count

The basic counting action is to count according to the input clock of AIN or BIN port, as shown in Figure 24-12 below.

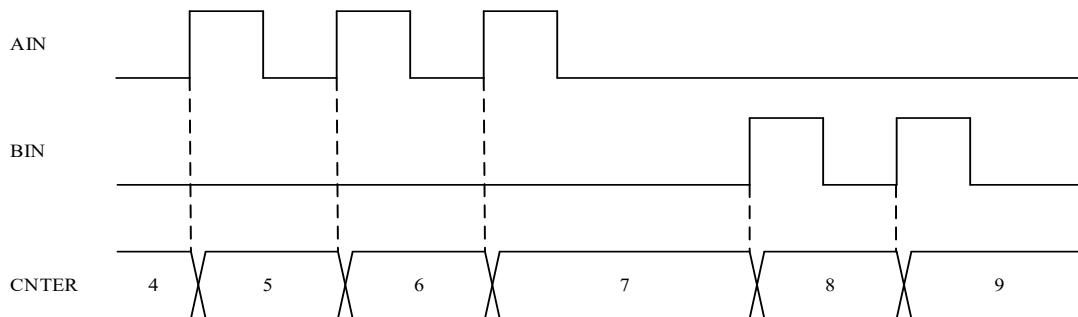
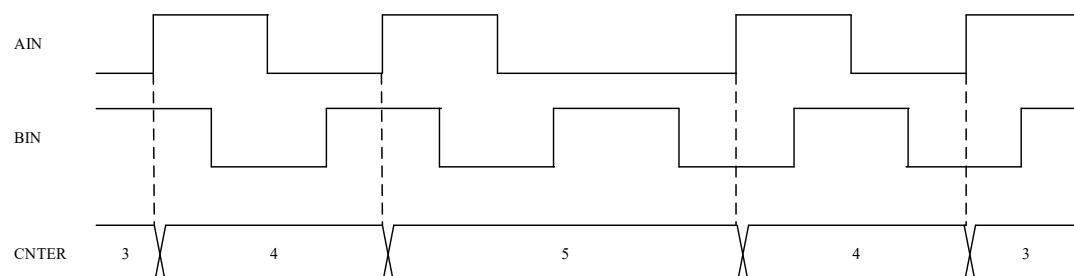


Figure 24-12 Position mode-basic counting

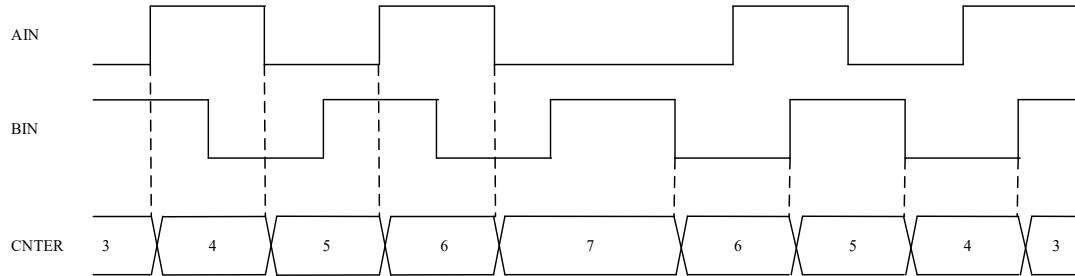
Phase difference count

Phase difference counts are counted according to the phase relationship between AIN and BIN. Depending on the setting, 1x count, 2x count, 4x count, etc. can be realized, as shown in Figure 24-13 ~ Figure 24-15 below.



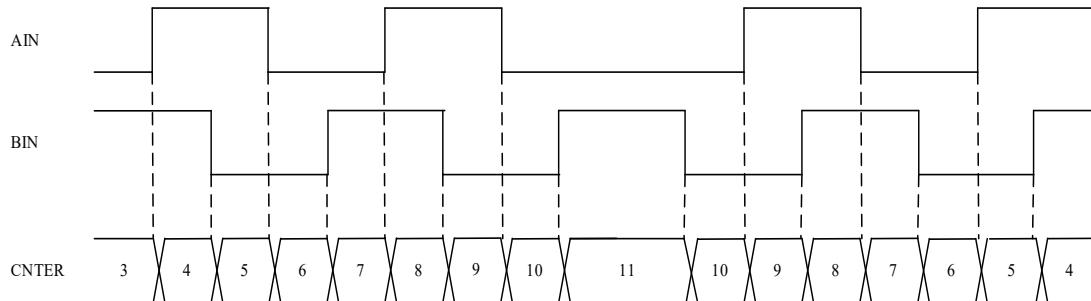
HCUPR.HCUP6=1 && HCDOR.HCDO4=1

Figure 24-13 Position counting mode-phase difference counting (1 times counting)



HCUPR.HCUP6=HCUPR.HCUP5=1 && HCDOR.HCDO2=HCDOR.HCDO1=1

Figure 24-14 Position counting mode-phase difference counting (2 times counting)

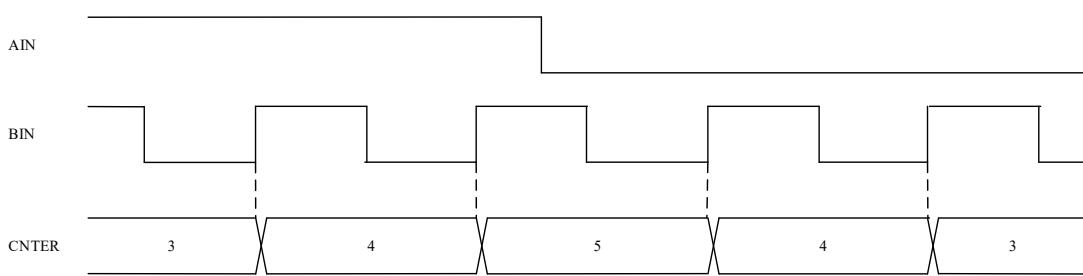


HCUPR.HCUP6=HCUPR.HCUP5=HCUPR.HCUP3=HCUPR.HCUP0=1 && HCDOR.HCDO7=HCDOR.HCDO4=HCDOR.HCDO2=HCDOR.HCDO1=1

Figure 24-15 Position counting mode-phase difference counting (4 times counting)

Direction count

Direction counting refers to setting the input state of AIN as direction control, and using the input of BIN as clock counting, as shown in follows Figure 24-16 shown below.



HCUPR.HCUP2=1 && HCDOR.HCDO0=1

Figure 24-16 Position counting mode-direction counting

24.3.10.2 Revolution count mode

Quadrature encoding revolution counting mode means that on the basis of AIN and BIN counting, input events of ZIN are added to realize the judgment of revolution circles and so on. In revolution counting mode, according to the counting method of revolution timer, Z-phase counting function, position overflow counting function and mixed counting function can be realized.

Z-phase count

Z-phase counting refers to the counting action of the revolution counting unit for counting, and clearing the position counting unit at the same time, according to the input of ZIN. As shown in Figure 24-17 below.

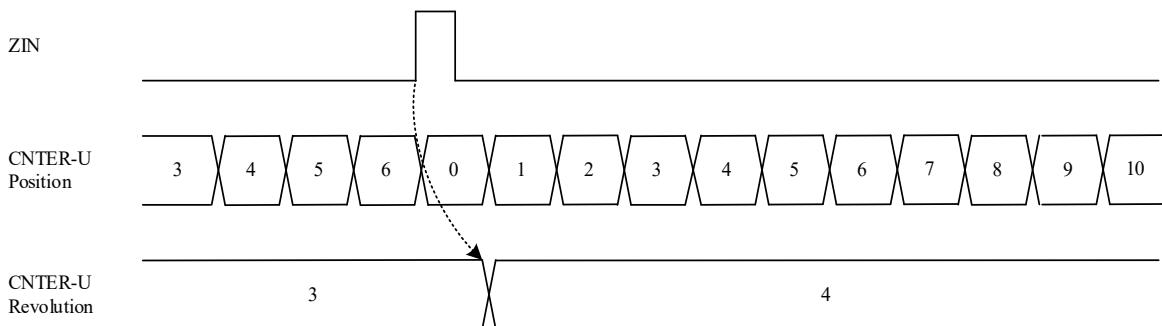


Figure 24-17 Revolution counting mode-Z phase counting

position overflow count

Position overflow counting means that when the position counting unit count overflows or underflows, an overflow event is generated, which triggers the timer of the revolution counting unit to count once (in this counting mode, the input of ZIN does not perform the counting action of the revolution counting unit and the clearing action of the position counting unit).

The increment (decrement) event bits 12~11 of the Hardware Count Up (Down) Event Select Register (HCUPR or HCDOR) of the revolution counting unit are enabled, and the overflow event of the position counting unit can trigger the revolution counting unit to realize one count. as follows Figure 24-18 shown.

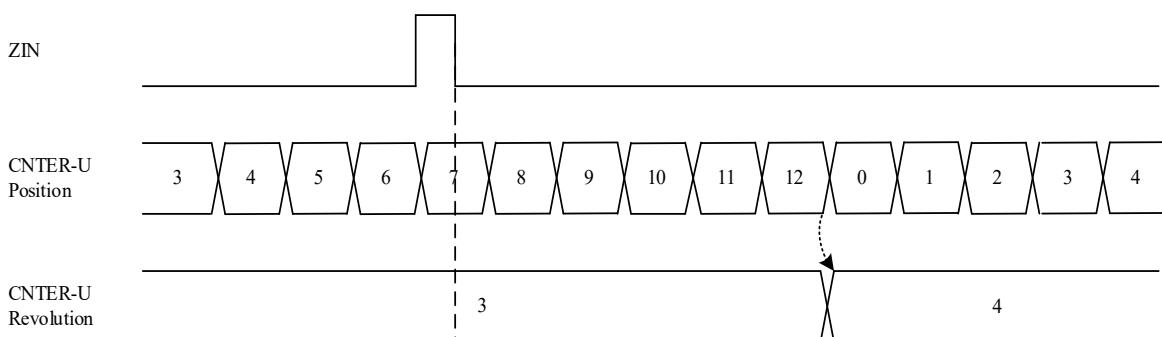


Figure 24-18 Revolution counting mode-position overflow counting

mixed count

Mixed counting refers to the counting action in which the above two counting methods of Z-phase counting and position overflow counting are combined, and the realization method is also a combination of the above two counting methods. as follows Figure 24-19 shown.

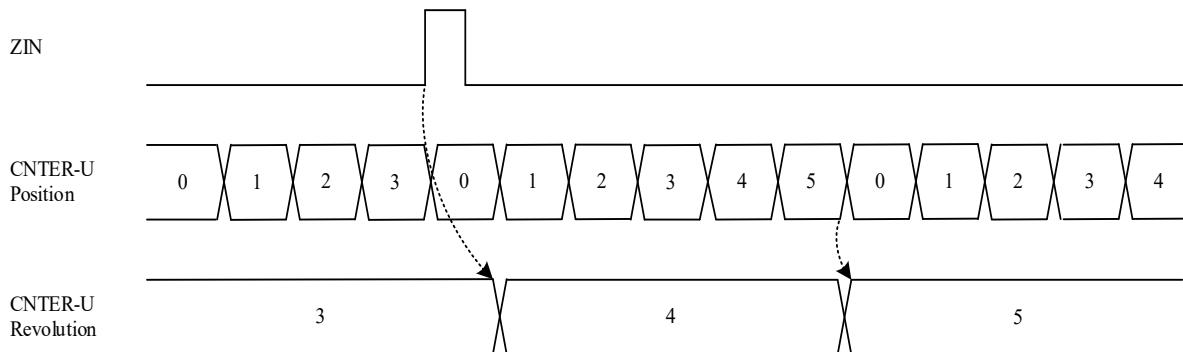


Figure 24-19 Revolution counting mode-mixed counting

24.4 Interrupt and Event Description

TimerA contains three interrupt outputs and three event outputs: One compare match interrupt and event, two period match interrupt and event.

24.4.1 Compare Matched Interrupt and Events

When a compare match occurs between the Compare Value Register (CMPARn) and the count value, the corresponding bit (STFLR.CMPFn) in the Status Flag Register (STFLR) will be set to 1. At this time, if the corresponding bit (ICONR.ITENn) of the Interrupt Control Register (ICONR) is set to 1, the corresponding interrupt request (TMRA_<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_<t>_CMP) will be triggered (n=1~4).

When the Capture Control Register (CCONRn) selected capture input valid condition occurs, the capture input action occurs. At this time, if the corresponding bit (ICONR.ITENn) of the Interrupt Control Register (ICONR) is set to 1, the corresponding interrupt request (TMRA_<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_<t>_CMP) will be triggered (n=1~4).

Compare-match interrupt and compare-match events of the four reference values within each unit are not output independently. Compare-match interrupt is aggregated into a interrupt by OR logic and output to the interrupt module (refer to INTC section), and compare-match events are aggregated into an event output by OR logic to select another module to trigger.

24.4.2 Periodic Matching Interrupt and Events

When the sawtooth wave mode counts up to the overflow point, the sawtooth wave mode counts down to the underflow point, and the triangular wave mode counts up to the valley or peak point, the OVFF or UDFF bit in the Basic Control and Status Register (BCSTR) will be set to 1. At this time, if the BCSTR.ITENOVF or BCSTR.ITENUDF bit is set to 1 to enable the interrupt, the periodic match interrupt (TMRA_<t>_OVF and TMRA_<t>_UDF) can be triggered at the corresponding cycle point and output to the interrupt module (INTC); the periodic match event has no corresponding enable bit control, and the periodic match event (TMRA_<t>_OVF and TMRA_<t>_UDF) output is triggered at the corresponding count cycle point to select and trigger other modules.

24.5 Register description

Table 24-3 shown is the register list of the TimerA module.

BASE ADDR:

0x4003A000 (U1), 0x4003A400 (U2), 0x4003A800 (U3),
 0x4003AC00 (U4), 0x40026000 (U5), 0x40026400 (U6),
 0x40026800 (U7), 0x40026C00 (U8), 0x40027000 (U9),
 0x40027400 (U10), 0x40027800 (U11), 0x40027C00 (U12)

Table 24-3 TimerA Register List

Register name	Symbol	Offset	Bit width	Reset value
Counter Value Register	TMRA_CNTER	0x0000h	16	0x0000h
Period Value Register	TMRA_PERAR	0x0004h	16	0xFFFFh
Compare Value Register 1	TMRA_CMPAR1	0x0040h	16	0xFFFFh
Compare Value Register 2	TMRA_CMPAR2	0x0044h	16	0xFFFFh
Compare Value Register 3	TMRA_CMPAR3	0x0048h	16	0xFFFFh
Compare Value Register 4	TMRA_CMPAR4	0x004Ch	16	0xFFFFh
Basic Control and Status Register	TMRA_BCSTR	0x0080h	16	0x0002h
Interrupt Control Register	TMRA_ICONR	0x0090h	16	0x0000h
Event Control Register	TMRA_ECONR	0x0094h	16	0x0000h
Filter Control Register	TMRA_FCONR	0x0098h	16	0x0000h
Status Flag Register	TMRA_STFLR	0x009Ch	16	0x0000h
Buffer Control Register 1	TMRA_BCONR1	0x00C0h	16	0x0000h
Buffer Control Register 2	TMRA_BCONR2	0x00C8h	16	0x0000h
Capture Control Register 1	TMRA_CCONR1	0x0100h	16	0x0000h
Capture Control Register 2	TMRA_CCONR2	0x0104h	16	0x0000h
Capture Control Register 3	TMRA_CCONR3	0x0108h	16	0x0000h
Capture Control Register 4	TMRA_CCONR4	0x010Ch	16	0x0000h
Port Control Register 1	TMRA_PCONR1	0x0140h	16	0x0000h
Port Control Register 2	TMRA_PCONR2	0x0144h	16	0x0000h
Port Control Register 3	TMRA_PCONR3	0x0148h	16	0x0000h
Port Control Register 4	TMRA_PCONR4	0x014Ch	16	0x0000h
Hardware Control Event Select Register	TMRA_HCONR	0x0084h	16	0x0000h
Hardware Count Up Event Select Register	TMRA_HCUPR	0x0088h	16	0x0000h
Hardware Count Down Event Select Register	TMRA_HCDOR	0x008Ch	16	0x0000h
Hardware Trigger Source Select Register 0	TMRA_HTSSR0	(0x40010880h)	32	0x000001FFh
Hardware Trigger Source Select Register 1	TMRA_HTSSR1	(0x40010884h)	32	0x000001FFh

Register name	Symbol	Offset	Bit width	Reset value
Hardware Trigger Source Select Register 2	TMRA_HTSSR2	(0x40010888h)	32	0x000001FFh
Hardware Trigger Source Select Register 3	TMRA_HTSSR3	(0x4001088Ch)	32	0x000001FFh

Note:

- The Hardware Trigger Source Select Registers (TMRA_HTSSR0~3) are 4 independent registers, which are shared by TimerA of 12 units. The specific allocation is shown in the table below.

Table 24-4 Internal trigger event HTSSR selection relationship correspondence table

Number of units	Event classification	The corresponding selection register
U1	Counter trigger event: HTSSR selection events referred to by HCONR, HCUPR, HCDOR	TMRA_HTSSR0
U2		TMRA_HTSSR1
U3		TMRA_HTSSR2
U4		TMRA_HTSSR3
U5		TMRA_HTSSR0
U6		TMRA_HTSSR1
U7		TMRA_HTSSR2
U8		TMRA_HTSSR3
U9		TMRA_HTSSR0
U10		TMRA_HTSSR1
U11		TMRA_HTSSR2
U12		TMRA_HTSSR3
U1	Capture action-triggered events: HTSSR selection event referred to in CCONR	TMRA_HTSSR1
U2		TMRA_HTSSR0
U3		TMRA_HTSSR3
U4		TMRA_HTSSR2
U5		TMRA_HTSSR1
U6		TMRA_HTSSR0
U7		TMRA_HTSSR3
U8		TMRA_HTSSR2
U9		TMRA_HTSSR1
U10		TMRA_HTSSR0
U11		TMRA_HTSSR3
U12		TMRA_HTSSR2

24.5.1 Counter Value Register (TMRA_CNTER)

Reset value: 0x0000h

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CNT[15:0]															
Bit	Symbol	Bit name		Description										Read and write	
b15~b0	CNT[15:0]	Count value		Current timer count										R/W	

24.5.2 Period Value Register (TMRA_PERAR)

Reset value: 0xFFFFh

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PER[15:0]															
Bit	Symbol	Bit name		Description										Read and write	
b15~b0	PER[15:0]	Count cycle value		Set the count cycle value for each round										R/W	

24.5.3 Compare Value Register (TMRA_CMPARm) (m=1~4)

Reset value: 0xFFFFh

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CMP[15:0]															
Bit	Symbol	Bit name		Description										Read and write	
b15~b0	CMP[15:0]	Count compare		Set compare reference value										R/W	

24.5.4 Basic Control and Status Register (TMRA_BCSTR)

Reset value: 0x0002h

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
UDF F	OVF F	ITEN UDF	ITEN OVF	-	-	-	OV STP	CKDIV[3:0]		SYN ST	MODE	DIR	START		
Bit	Symbol	Bit name												Read and write	
b15	UDFF	Underflow flag												R/W	
b14	OVFF	Overflow flag												R/W	
b13	ITENUDF	Underflow interrupt enable												R/W	
b12	ITENOVF	Overflow interrupt enable												R/W	
b11~b9	Reserved	-												R/W	
b8	OVSTP	Count overflow stop control												R/W	
b7~b4	CKDIV[3:0]	Counting clock selection												R/W	
b3	SYNST	Synchronous enable												R/W	
b2	MODE	Counting mode												R/W	
b1	DIR	Counting direction												R/W	
b0	START	Timer start												R/W	

24.5.5 Interrupt Control Register (TMRA_ICONR)

Reset value: 0x0000h

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved												IT EN4	IT EN3	IT EN2	IT EN1
Bit	Symbol	Bit name										Description			
b15~b4	Reserved	-										Read as "0", write as "0"			
b3	ITEN4	Count Match interrupt enable 4										0: When the CMPAR4 register is equal to the count value, or when a capture input event occurs, the interrupt is invalid 1: When the CMPAR4 register is equal to the count value, or when a capture input event occurs, the interrupt is enabled			
b2	ITEN3	Count Match interrupt enable 3										0: When the CMPAR3 register is equal to the count value, or when a capture input event occurs, the interrupt is invalid 1: The interrupt is enabled when the CMPAR3 register is equal to the count value, or when a capture input event occurs			
b1	ITEN2	Count Match interrupt enable 2										0: When the CMPAR2 register is equal to the count value, or when a capture input event occurs, the interrupt is invalid 1: When the CMPAR2 register is equal to the count value, or when a capture input event occurs, the interrupt is enabled			
b0	ITEN1	Count Match interrupt enable 1										0: When the CMPAR1 register is equal to the count value, or when a capture input event occurs, the interrupt is invalid 1: When the CMPAR1 register is equal to the count value, or when a capture input event occurs, the interrupt is enabled			

24.5.6 Event Control Register (TMRA_ECONR)

Reset value: 0x0000h

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
Reserved													ET EN4	ET EN3	ET EN2	ET EN1
Bit	Symbol	Bit name		Description										Read and write		
b15~b4	Reserved	-		Read as "0", write as "0"										R/W		
b3	ETEN4	Count Match Event Enabled 4		0: When the CMPAR4 register is equal to the count value, or when a capture input event occurs, the event output is invalid 1: When the CMPAR4 register is equal to the count value, or when a capture input event occurs, the event output is enabled										R/W		
b2	ETEN3	Count Match Event Enabled 3		0: When the CMPAR3 register is equal to the count value, or when a capture input event occurs, the event output is invalid 1: When the CMPAR3 register is equal to the count value, or when a capture input event occurs, the event output is enabled										R/W		
b1	ETEN2	Count Match Event Enabled 2		0: When the CMPAR2 register is equal to the count value, or when a capture input event occurs, the event output is invalid 1: When the CMPAR2 register is equal to the count value, or when a capture input event occurs, the event output is enabled										R/W		
b0	ETEN1	Count Match Event Enabled 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, the event output is enabled										R/W		

24.5.7 Filter Control Register (TMRA_FCONR)

Reset value: 0x0000h

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	
Bit	Symbol		Bit name		Description										Read and write
b15	Reserved		-		Read as "0", write as "0"										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 disabled 1: TIMA_<t>_CLKB port input filtering function enabled										R/W
b11	Reserved		-		Read as "0", write as "0"										R/W
b10~b9	NOFICKCA[1:0]		Filtering 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 disabled 1: TIMA_<t>_CLKA port input filtering function enabled										R/W
b7~b3	Reserved		-		Read as "0", write as "0"										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 filter TG		0: TIMA_<t>_TRIG port input filtering function disabled 1: TIMA_<t>_TRIG port input filtering function enabled										R/W

24.5.8 Status Flag Register (TMRA_STFLR)

Reset value: 0x0000h

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved												CMP F4	CMP F3	CMP F2	CMP F1
Bit	Symbol	Bit name		Description										Read and write	
b15-b4	Reserved	-		Read as "0", write as "0"										R	
b3	CMPF4	Count Match Flags 4		0: The value of the CMPAR4 register is not equal to the count value, and the TIMA_<t>_PWM4 capture completion action has not occurred 1: The value of the CMPAR4 register is equal to the count value, or the TIMA_<t>_PWM4 capture completion action occurs										R/W	
b2	CMPF3	Count Match Flags 3		0: The value of the CMPAR3 register is not equal to the count value, and the TIMA_<t>_PWM3 capture completion action has not occurred 1: The value of the CMPAR3 register is equal to the count value, or the TIMA_<t>_PWM3 capture completion action occurs										R/W	
b1	CMPF2	Count Match Flags 2		0: The value of the CMPAR2 register is not equal to the count value, and the TIMA_<t>_PWM2 capture completion action has not occurred 1: The value of the CMPAR2 register is equal to the count value, or the TIMA_<t>_PWM2 capture completion action occurs										R/W	
b0	CMPF1	Count Match Flags 1		0: The value of the CMPAR1 register is not equal to the count value, and the TIMA_<t>_PWM1 capture completion action has not occurred 1: The value of the CMPAR1 register is equal to the count value, or the TIMA_<t>_PWM1 capture completion action occurs										R/W	

24.5.9 Buffer Control Register (TMRA_BCONR_m) ($m=1\sim2$)

Reset value: 0x0000h

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0		
Reserved														BSE1	BSE0	BEN	
Bit	Symbol	Bit name		Description											Read and write		
b15~b3	Reserved	-		Read as "0", write as "0"											R/W		
b2	BSE1	Triangular wave buffer transmission selection 1		0: Buffed value is not transmitted when triangular mode counts to valley point 1: Buffed value is transmitted when triangular mode counts to valley point, i.e.: CMMAR _m -> CMMAR _n ($m=2, 4, n=1, 3$)												R/W	
b1	BSE0	Triangular wave buffer transmission selection 0		0: Buffed value is not transmitted when triangular mode counts to peak point. 1: Buffed value is transmitted when triangular mode counts to peak point, i.e.: CMMAR _m -> CMMAR _n ($m=2, 4, n=1, 3$)												R/W	
b0	BEN	Buffer enable		0: The Buffer Function of CMPAR _n reference value disable 1: The Buffer Function of CMPAR _n reference value enable ($n=1, 3$)												R/W	

24.5.10 Capture Control Register (TMRA_CCONR_m) ($m=1\sim 4$)

Reset value: 0x0000h

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	

Bit	Symbol	Bit name	Description	Read and write
b15	Reserved	-	Read as "0", write as "0"	R/W
b14~b13	NOFICKCP[1:0]	Filtering 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>_PWM _n port input filtering disabled 1: TIMA_<t>_PWM _n port input filtering enabled (n=1~4)	R/W
b11~b10	Reserved	-	Read as "0", write as "0"	R/W
b9	HICP4	Capture input condition enable 4	0: When the input of the TIMA_<t>_TRIG port is sampled to the falling edge, no capture input action occurs on channel m+1 1: When the input of the TIMA_<t>_TRIG port is sampled to the falling edge, channel m+1 generates a capture input action Note: This bit is only valid in the CCONR3 register. That is, after this bit is valid and the corresponding event occurs, under the condition of 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 input of the TIMA_<t>_TRIG port is sampled to the rising edge, no capture input action occurs on channel m+1 1: When the input of the TIMA_<t>_TRIG port is sampled to the rising edge, channel m+1 generates a capture input action Note: This bit is only valid in the CCONR3 register. That is, after this bit is valid and the corresponding event occurs, under the condition of CCONR4.CAPMD=1, the current counter value is captured and saved to CMPAR4, and STFLR.CMPF4 is set	R/W
b7	Reserved	-	Read as "0", write as "0"	R/W
b6	HICP2	Capture input condition enable 2	0: When an event specified in TMRA_HTSSR register occurs, the capture input action is not generated 1: When an event specified in TMRA_HTSSR register occurs, the capture input action is generated Note: For details, please refer to the notes in the register description chapter.	R/W
b5	HICP1	Capture input condition enable 1	0: When the input of TIMA_<t>_PWM _n port is sampled to the falling edge, the capture input action is not generated 1: When the input of TIMA_<t>_PWM _n port is sampled to the falling edge, the capture input action is generated (n=1~4)	R/W
b4	HICP0	Capture input condition enable 0	0: When the input of TIMA_<t>_PWM _n port is sampled to the rising edge, the capture input action is not generated 1: When the input of TIMA_<t>_PWM _n port is sampled to the rising edge, the capture input action is generated (n=1~4)	R/W
b3~b1	Reserved	-	Read as "0", write as "0"	R/W
b0	CAPMD	Functional mode selection	0: Compare output function 1: Capture input function	R/W

24.5.11 Port Control Register (TMRA_PCONRm) (m=1~4)

Reset value: 0x0000h

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]				
Bit	Symbol		Bit name	Description										Read and write	
b15~b13	Reserved		-	Read as "0", write as "0"										R/W	
b12	OUTEN		Output enable	0: When PWM output function, TIMA_<t>_PWMn port output is disabled 1: When PWM output Function, TIMA_<t>_PWMn port output is enabled (n=1~4)										R/W	
b11~b10	Reserved		-	Read as "0", write as "0"										R/W	
b9~b8	FORC[1:0]		Forced port status setting	0x: invalid setting 10: Starting from next cycle, TIMA_<t>_PWMn port output is set to low 11: Starting from next cycle, TIMA_<t>_PWMn port output is set to high (n=1~4) Note 1: The next cycle refers to the hardware counting mode or the sawtooth wave counting to the overflow point or underflow point, the triangular wave counting to the valley point Note 2: This register bit can be used to achieve 0% or 100% control of PWM output duty cycle.										R/W	
b7~b6	PERC[1:0]		Port status setting when period values match	00: When the count value is equal to PERAR, TIMA_<t>_PWMn port output is set to low 01: When the count value is equal to PERAR, TIMA_<t>_PWMn port output is set to high 10: When the count value is equal to PERAR, TIMA_<t>_PWMn port output remains the previous state 11: When the count value is equal to PERAR, TIMA_<t>_PWMn port output is set to inverted level (n=1~4)										R/W	
b5~b4	CMPC[1:0]		Port status setting when compare values match	00: When the count value is equal to CMPARn, TIMA_<t>_PWMn port output is set to low 01: When the count value is equal to CMPARn, TIMA_<t>_PWMn port output is set to high 10: When the count value is equal to CMPARn, TIMA_<t>_PWMn port output remains the previous state 11: When the count value is equal to CMPARn, TIMA_<t>_PWMn port output is set to inverted level (n=1~4)										R/W	
b3~b2	STPC[1:0]		Port status setting when count stops	00: When the count stops, TIMA_<t>_PWMn port output is set to low 01: When the count stops, TIMA_<t>_PWMn port output is set to high 10: When the count stops, TIMA_<t>_PWMn port output remains the previous state 11: When the count stops, TIMA_<t>_PWMn port output remains the previous state (n=1~4)										R/W	
b1~b0	STAC[1:0]		Port status setting at start of count	00: At the beginning of the count, TIMA_<t>_PWMn port output is set to low 01: At the beginning of the count, TIMA_<t>_PWMn port output is set to high 10: At the beginning of the count, TIMA_<t>_PWMn port output remains the previous state 11: At the beginning of the count, TIMA_<t>_PWMn port output remains the previous state (n=1~4) Note: This bit setting is only valid when the frequency is not divided (BCSTR.CKDIV=4'h0), other frequency divisions should be set to 2'b10 or 2'b11										R/W	

24.5.12 Hardware Control Event Select Register (TMRA_HCONR)

Reset value: 0x0000h

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
HCLE6	HCLE5	HCLE4	HCLE3	-	HCLE2	HCLE1	HCLE0	-	HSTP2	HSTP1	HSTP0	-	HSTA2	HSTA1	HSTA0

Bit	Symbol	Bit name	Description	Read and write
b15	HCLE6	Hardware clear condition 6	Conditions: TIMA_<t>_PWM3 port input sampled to falling edge 0: Hardware clear is invalid when conditions match 1: Hardware clear is valid when conditions match	R/W
b14	HCLE5	Hardware clear condition 5	Conditions: TIMA_<t>_PWM3 port input sampled to rising edge 0: Hardware clear is invalid when conditions match 1: Hardware clear is valid when conditions match	R/W
b13	HCLE4	Hardware clear condition 4	Conditions: When this unit is unit m, the input of the TRIG port of unit n is sampled to the falling edge (when m=1, 3, 5, 7, 9, 11, n=2, 4, 6, 8, 10, 12; when m = 2, 4, 6, 8, 10, 12, n=1, 3, 5, 7, 9, 11) 0: Hardware clear is invalid when conditions match 1: Hardware clear is valid when conditions match	R/W
b12	HCLE3	Hardware clear condition 3	Conditions: When this unit is unit m, the input of the TRIG port of unit n is sampled to the rising edge (when m=1, 3, 5, 7, 9, 11, n=2, 4, 6, 8, 10, 12; when m = 2, 4, 6, 8, 10, 12, n=1, 3, 5, 7, 9, 11) 0: Hardware clear is invalid when conditions match 1: Hardware clear is valid when conditions match	R/W
b11	Reserved	-	Read as "0", write as "0"	R/W
b10	HCLE2	Hardware clear condition 2	Conditions: The event specified in the TMRA_HTSSR register occurs 0: Hardware clear is invalid when conditions match 1: Hardware clear is valid when conditions match Note: For details, please refer to the notes in the register description chapter.	R/W
b9	HCLE1	Hardware clear condition 1	Conditions: TIMA_TRIG port input sampled to falling edge 0: Hardware clear is invalid when conditions match 1: Hardware clear is valid when conditions match	R/W
b8	HCLE0	Hardware clear condition 0	Conditions: TIMA_TRIG port input sampled to rising edge 0: Hardware clear is invalid when conditions match 1: Hardware clear is valid when conditions match	R/W
b7	Reserved	-	Read as "0", write as "0"	R/W
b6	HSTP2	Hardware stop condition 2	Conditions: The event specified in the TMRA_HTSSR register occurs 0: Hardware stop is invalid when conditions match 1: Hardware stop is valid when conditions match Note: For details, please refer to the notes in the register description chapter.	R/W
b5	HSTP1	Hardware stop condition 1	Conditions: TIMA_<t>_TRIG port input sampled to falling edge 0: Hardware stop is invalid when conditions match 1: Hardware stop is valid when conditions match	R/W
b4	HSTP0	Hardware stop condition 0	Conditions: TIMA_<t>_TRIG port input sampled to rising edge 0: Hardware stop is invalid when conditions match 1: Hardware stop is valid when conditions match	R/W
b3	Reserved	-	Read as "0", write as "0"	R/W
b2	HSTA2	Hardware startup condition 2	Conditions: The event specified in the TMRA_HTSSR register occurs 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match Note: For details, please refer to the notes in the register description chapter.	R/W
b1	HSTA1	Hardware startup condition 1	Conditions: 1) The input of the TIMA_<t>_TRIG port of this unit is sampled to the falling edge (the synchronous start function is invalid) 2) The TIMA_n_TRIG port input is sampled to the falling edge (the synchronous start function is valid) 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match	R/W

			Note: In condition 2), when the unit is 2, 4, 6, 8, 10, 12, $n=1, 3, 5, 7, 9, 11$; when the unit is 1, 3, 5, 7, 9, 11, this function is invalid	
b0	HSTA0	Hardware startup condition 0	Conditions: 1) The input of the TIMA_<t>_TRIG port of this unit is sampled to the rising edge (the synchronous start function is invalid) 2) The TIMA_n_TRIG port input is sampled to the rising edge (the synchronization start function is valid) 0: Hardware startup is invalid when conditions match 1: Hardware startup is valid when conditions match Note: In condition 2), when the unit is 2, 4, 6, 8, 10, 12, $n=1, 3, 5, 7, 9, 11$; when the unit is 1, 3, 5, 7, 9, 11, this function is invalid	R/W

24.5.13 Hardware Count Up Event Select Register (TMRA_HCUPR)

Reset value: 0x0000h

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

Bit	Symbol	Bit name	Description	Read and write
b15~b13	Reserved	-	Read as "0", write as "0"	R/W
b12	HCUP12	Hardware increment condition 12	Conditions: When this unit is unit m, the count underflow occurs in unit n (when m=1, 3, 5, 7, 9, 11, n=2, 4, 6, 8, 10, 12; when m=2, 4, 6, 8, 10, 12, n=1, 3, 5, 7, 9, 11) 0: Hardware increment is invalid when conditions match 1: Hardware increment is valid when conditions match	R/W
b11	HCUP11	Hardware increment condition 11	Conditions: When this unit is unit m, the count overflow occurs in unit n (when m=1, 3, 5, 7, 9, 11, n=2, 4, 6, 8, 10, 12; when m=2, 4, 6, 8, 10, 12, n=1, 3, 5, 7, 9, 11) 0: Hardware increment invalid when conditions match 1: Hardware increment is valid when conditions match	R/W
b10	HCUP10	Hardware increment condition 10	Conditions: The event specified in the TMRA_HTSSR register occurs 0: Hardware increment invalid when conditions match 1: Hardware increment is valid when conditions match Note: For details, please refer to the notes in the register description chapter.	R/W
b9	HCUP9	Hardware increment condition 9	Conditions: TIMA_<t>_TRIG port is sampled to the falling edge 0: Hardware increment invalid when conditions match 1: Hardware increment is valid when conditions match	R/W
b8	HCUP8	Hardware increment condition 8	Conditions: TIMA_<t>_TRIG port is sampled to the rising edge 0: Hardware increment invalid when conditions match 1: Hardware increment is valid when conditions match	R/W
b7	HCUP7	Hardware increment condition 7	Conditions: When the TIMA_<t>_CLKB port is high, the TIMA_<t>_CLKA port is sampled to the falling edge 0: Hardware increment invalid when conditions match 1: Hardware increment is valid when conditions match	R/W
b6	HCUP6	Hardware increment condition 6	Conditions: When the TIMA_<t>_CLKB port is high, the TIMA_<t>_CLKA port is sampled to the rising edge 0: Hardware increment invalid when conditions match 1: Hardware increment is valid when conditions match	R/W
b5	HCUP5	Hardware increment condition 5	Conditions: When the TIMA_<t>_CLKB port is low, the TIMA_<t>_CLKA port is sampled to the falling edge 0: Hardware increment invalid when conditions match 1: Hardware increment is valid when conditions match	R/W
b4	HCUP4	Hardware increment condition 4	Conditions: When the TIMA_<t>_CLKB port is low, the TIMA_<t>_CLKA port is sampled to the rising edge 0: Hardware increment invalid when conditions match 1: Hardware increment is valid when conditions match	R/W
b3	HCUP3	Hardware increment condition 3	Conditions: When the TIMA_<t>_CLKB port is high, the TIMA_<t>_CLKA port is sampled to the falling edge 0: Hardware increment invalid when conditions match 1: Hardware increment is valid when conditions match	R/W
b2	HCUP2	Hardware increment condition 2	Conditions: When the TIMA_<t>_CLKB port is high, the TIMA_<t>_CLKA port is sampled to the rising edge 0: Hardware increment invalid when conditions match 1: Hardware increment is valid when conditions match	R/W
b1	HCUP1	Hardware increment condition 1	Conditions: When the TIMA_<t>_CLKB port is low, the TIMA_<t>_CLKA port is sampled to the falling edge 0: Hardware increment invalid when conditions match 1: Hardware increment is valid when conditions match	R/W
b0	HCUP0	Hardware increment condition 0	Conditions: When the TIMA_<t>_CLKB port is low, the TIMA_<t>_CLKA port is sampled to the rising edge 0: Hardware increment invalid when conditions match 1: Hardware increment is valid when conditions match	R/W

24.5.14 Hardware Count Down Event Select Register (TMRA_HCDOR)

Reset value: 0x0000h

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

Bit	Symbol	Bit name	Description	Read and write
b15~b13	Reserved	-	Read as "0", write as "0"	R/W
b12	HCDO12	Hardware decrement condition 12	Conditions: When this unit is unit m, the count underflow occurs in unit n (when m=1, 3, 5, 7, 9, 11, n=2, 4, 6, 8, 10, 12; when m=2, 4 , 6, 8, 10, 12, n=1, 3, 5, 7, 9, 11) 0: Hardware decrement is invalid when conditions match 1: Hardware decrement is valid when conditions match	R/W
b11	HCDO11	Hardware decrement condition	Conditions: When this unit is unit m, the count overflow occurs in unit n (when m=1, 3, 5, 7, 9, 11, n=2, 4, 6, 8, 10, 12; when m=2, 4 , 6, 8, 10, 12, n=1, 3, 5, 7, 9, 11) 0: Hardware decrement is invalid when conditions match 1: Hardware decrement is valid when conditions match	R/W
b10	HCDO10	Hardware decrement condition 10	Conditions: The event specified in the TMRAHTSSR register occurs 0: Hardware decrement is invalid when conditions match 1: Hardware decrement is valid when conditions match Note: For details, please refer to the notes in the register description chapter.	R/W
b9	HCDO9	Hardware decrement condition 9	Conditions: TIMA_<t>_TRIG port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrement is valid when conditions match	R/W
b8	HCDO8	Hardware decrement condition 8	Conditions: TIMA_<t>_TRIG port is sampled to the rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrement is valid when conditions match	R/W
b7	HCDO7	Hardware decrement condition 7	Conditions: When the TIMA_<t>_CLKB port is high, the TIMA_<t>_CLKA port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrement is valid when conditions match	R/W
b6	HCDO6	Hardware decrement condition 6	Conditions: When the TIMA_<t>_CLKB port is high, the TIMA_<t>_CLKA port is sampled to the rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrement is valid when conditions match	R/W
b5	HCDO5	Hardware decrement condition 5	Conditions: When the TIMA_<t>_CLKB port is low, the TIMA_<t>_CLKA port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrement is valid when conditions match	R/W
b4	HCDO4	Hardware decrement condition 4	Conditions: When the TIMA_<t>_CLKB port is low, the TIMA_<t>_CLKA port is sampled to the rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrement is valid when conditions match	R/W
b3	HCDO3	Hardware decrement condition 3	Conditions: When the TIMA_<t>_CLKA port is high, the TIMA_<t>_CLKB port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrement is valid when conditions match	R/W
b2	HCDO2	Hardware decrement condition 2	Conditions: When the TIMA_<t>_CLKA port is high, the TIMA_<t>_CLKB port is sampled to the rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrement is valid when conditions match	R/W
b1	HCDO1	Hardware decrement condition 1	Conditions: When the TIMA_<t>_CLKA port is low, the TIMA_<t>_CLKB port is sampled to the falling edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrement is valid when conditions match	R/W
b0	HCDO0	Hardware decrement condition 0	Conditions: When the TIMA_<t>_CLKA port is low, the TIMA_<t>_CLKB port is sampled to the rising edge 0: Hardware decrement is invalid when conditions match 1: Hardware decrement is valid when conditions match	R/W

24.5.15 Hardware Trigger Source Select Register (TMRA_HTSSRm) (m=0~3)

Reset value: 0x000001FFh

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]							
<hr/>															
Bit	Symbol	Bit name		Description										Read and write	
b31	COMEN[1]	Common trigger enable 1		0: Disable the common trigger event of AOS_COMTRG2 to trigger TimerA 1: Enable the common trigger event of AOS_COMTRG2 to trigger TimerA See the Automatic Operating System (AOS) chapter for details										R/W	
b30	COMEN[0]	Common trigger enable 0		0: Disable the common trigger event of AOS_COMTRG1 to trigger TimerA 1: Enable the common trigger event of AOS_COMTRG1 to trigger TimerA See the Automatic Operating System (AOS) chapter for details										R/W	
b29~b9	Reserved	-		Read as "0", write as "0"										R/W	
b8~b0	TRGSEL	Trigger source selection		Trigger source number write, refer to INTC chapter										R/W	

25 General Timer (Timer2)

25.1 Introduction

General-purpose timer 2 (Timer2) is a basic timer that can realize synchronous counting and asynchronous counting. The timer contains 2 channels (CH-A and CH-B). Each channel has an output port, which can realize basic square wave output; each channel has 2 input ports, one is a clock input port, which can realize asynchronous counting of ports; the other is a trigger input port, which can realize timer start,stop,clear,count action and count value capture input; support pulse width measurement and period measurement. 4 units of Timer2 is carried in this series of product.

25.2 Basic block diagram

The basic functions and features of Timer2 are as follows:

Table 25-1 Basic functions and features of Timer2

Basic action	• Hardware-triggered counter start, stop, clear, count and capture inputs
	• Synchronous counting, asynchronous counting
	• Pulse width measurement
	• Period measurement
	• Square wave output
	• Event signal output
Interrupt type	• Count compare match interrupt
	• Count overflow interrupt

The basic block diagram of Timer2 is shown in Figure 25-1. The "<t>" shown in the block diagram indicates the unit number, that is, "<t>" is 1~4. When referring to "<t>" later in this chapter, it refers to the unit number, and will not be repeated here. The ANF on the TIM2_<t>_CLKA/B port input in the block diagram is an analog filter unit with a filter width of 40ns (typical conditions).

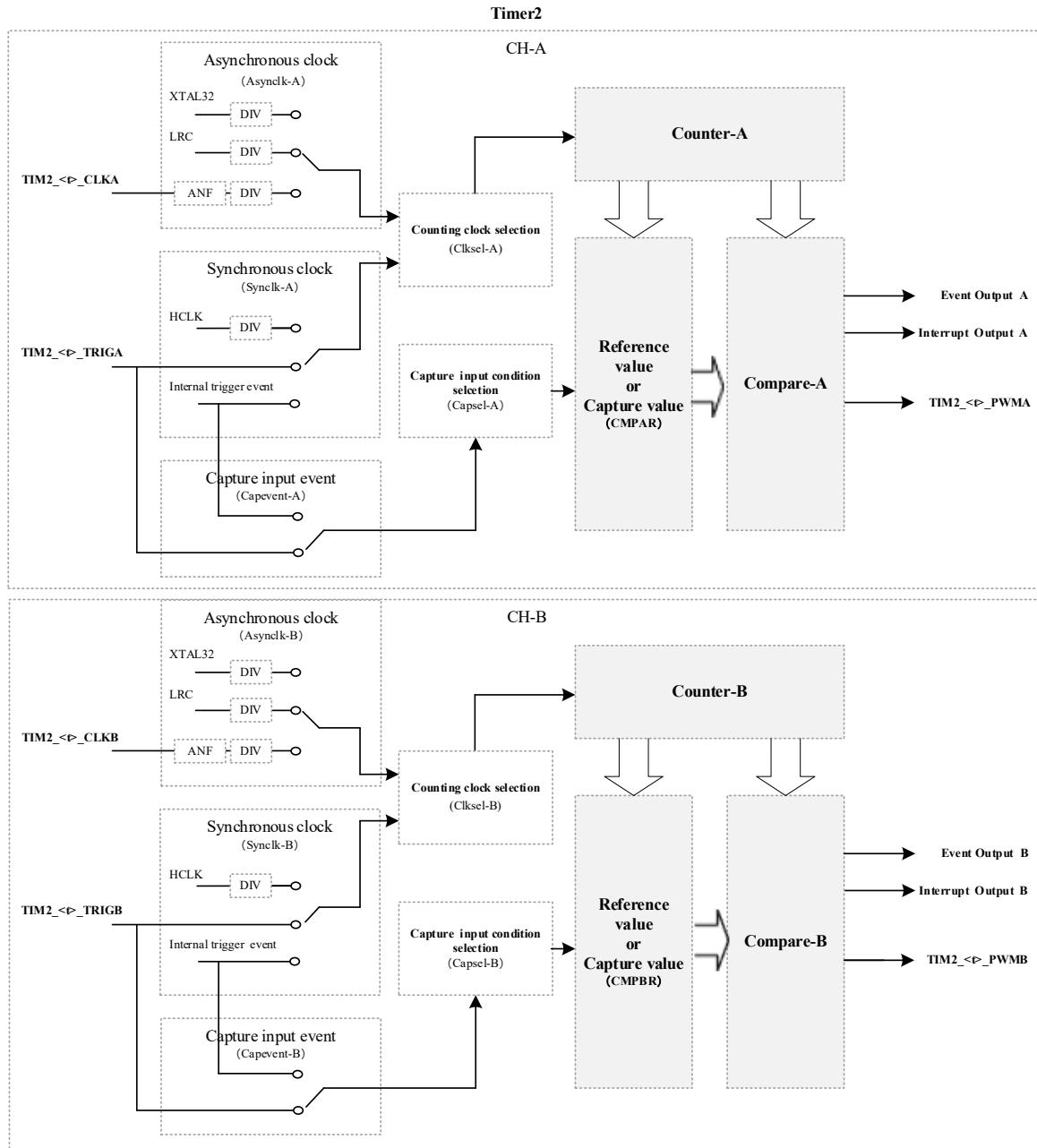


Figure 25-1 Timer2 basic block diagram

Table 25-2 shown is the list of input and output ports of Timer2.

Table 25-2 Timer2 port list

Port name	Direction	Function
TIM2_<t>_CLKA/B	in	Asynchronous clock input port
TIM2_<t>_TRIGA/B	in	1) Hardware start, stop, clear, counting input port 2) Capture input port
TIM2_<t>_PWMA/B	out	Square wave output port

25.3 Functional description

25.3.1 Clock source selection

Timer2 can be counted in synchronous or asynchronous mode.

The synchronous counting mode refers to the synchronous timing relationship between the timer count clock and the bus access clock (register read/write operation clock). Asynchronous counting mode means that the timer count clock and bus access clock (register read/write operation clock) are non-synchronous timing relationships. The status of timer, etc., may be changing when the register is read by asynchronous counting. Therefore, in the asynchronous counting mode, the register read operation must be implemented in the count stopped state.

25.3.1.1 Synchronous counting clock source

In synchronous counting mode ($\text{BCONR.SYNSA}_{[B]}=0$), the clock source can be selected from the following options ($\text{BCONR.SYNCLK A}_{[B]} [1:0]$ or $\text{BCONR.SYNCLK A}_{[B]} \text{T}[1:0]$ set up):

1. The 2, 4, 8, 16, 32, 64, 128, 256, 512, and 1024 frequency divisions of PCLK1 and PCLK1 are used as the synchronous count clock ($\text{BCONR.SYNCLK A}_{[B]} [1:0]=00$ & $\text{BCONR.CKDIV A}_{[B]} [3:0]$ setting)
2. The input valid select edge of $\text{TIM2}_{<t>}_{\text{TRIGA}}$, $\text{TIM2}_{<t>}_{\text{TRIGB}}$ port is sampled by the internal PCLK1 as the synchronous count clock ($\text{BCONR.SYNCLK A}_{[B]} [1:0]=01$ or 10)
3. Internal hardware trigger event input as synchronous count clock ($\text{BCONR.SYNCLK A}_{[B]} [1:0]=11$)
4. Count overflow event of Timer6 ($\text{BCONR.SYNCLKA}_{[B]} \text{T}[0]=1$)
5. Count underflow event of Timer6 ($\text{BCONR.SYNCLKA}_{[B]} \text{T}[1]=1$)

If clock source a is selected, the software counting mode is used. If clock source b, c, d, or e is selected, the hardware counting mode is used. The above description shows that the b, c, d and e clocks are independent of each other, can be set to be valid or invalid respectively, and the a clock is automatically invalid when b, c, d and e clocks are selected. When selecting d and e clock sources, please set PCLK0 and PCLK1 at the same frequency.

25.3.1.2 Asynchronous counting clock source

In asynchronous counting mode ($\text{BCONR.SYNS A}_{[B]}=1$), the clock source can be selected from the following options ($\text{BCONR.ASYNCLK A}_{[B]} [1:0]$ setting selection):

1. LRC clock source input and its 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 frequency division as asynchronous count clock ($\text{BCONR.ASYNCLK A}_{[B]}=00$ & $\text{BCONR.CKDIV A}_{[B]} [3:0]$ settings)
2. XTAL32 clock source input and its 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 frequency division as asynchronous count clock ($\text{BCONR.ASYNCLK A}_{[B]}=01$ & $\text{BCONR.CKDIV A}_{[B]}$)

- [3:0] settings)
3. TIM2_<t>_CLKA, TIM2_<t>_CLKB clock input port and its 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 division as asynchronous count clock (BCONR.ASYNCLK A=10 & BCONR.CKDIV A [3:0] settings)

25.3.2 Compare output

Timer2 has 2 output ports (TIM2_<t>_PWMA, TIM2_<t>_PWMB), which can realize basic square wave output.

The CMPAR and CMPBR registers correspond to the count compare reference values of TIM2_<t>_PWMA and TIM2_<t>_PWMB. When the timer count values CNTAR and CMPAR are equal, the TIM2_<t>_PWMA port outputs the specified level; when the timer count values CNTBR and CMPBR are equal, the TIM2_<t>_PWMB port outputs the specified level.

The count start level, stop level, and count compare match level of the TIM2_<t>_PWMA/B port can be determined by PCONR.STACA[1:0], PCONR.STPCA of the Port Control Register (PCONR). [1:0], PCONR.CMPCA[1:0] bit setting. Figure 25-2 shows an example of the compare output operation.

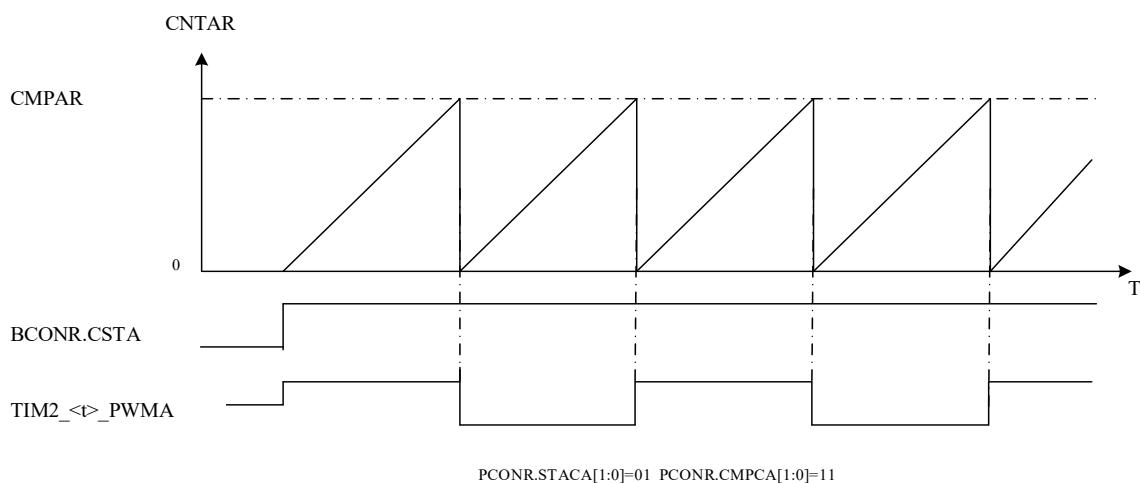


Figure 25-2 Compare output action

25.3.3 Hardware triggering

There are two types of hardware trigger actions of Timer2, one is the input event of the TIM2_<t>_TRIGA/B port, and the other is the internal hardware trigger event. Both kinds of events can realize the function of starting, stopping, clearing and hardware capture input. All kinds of function combination can also realize pulse width measurement and period measurement.

TIM2_<t>_TRIGA/B port input type selection (rising edge or falling edge) and function selection (start, stop, clear, capture input), function selection of internal hardware trigger events (start, stop, clear, capture input)) can be set by the corresponding bits in the Hardware Control Register (HCONR).

Timer2 has an internal hardware trigger source that can be selected by triggering the corresponding number setting in the Trigger Source Select Register (HTSSR). For specific event correspondence, refer to the INTC chapter. When using the internal hardware trigger function, you need to set the peripheral circuit trigger function enable bit of the Function Clock Control Register (PWC_FCG0) to 1 first.

25.3.3.1 Start stop and clear

The hardware start, stop and clear conditions of each channel are determined by the relevant settings of the Hardware Control Register (HCONR). Figure 25-3 is an example of the operation of starting and clearing by hardware.

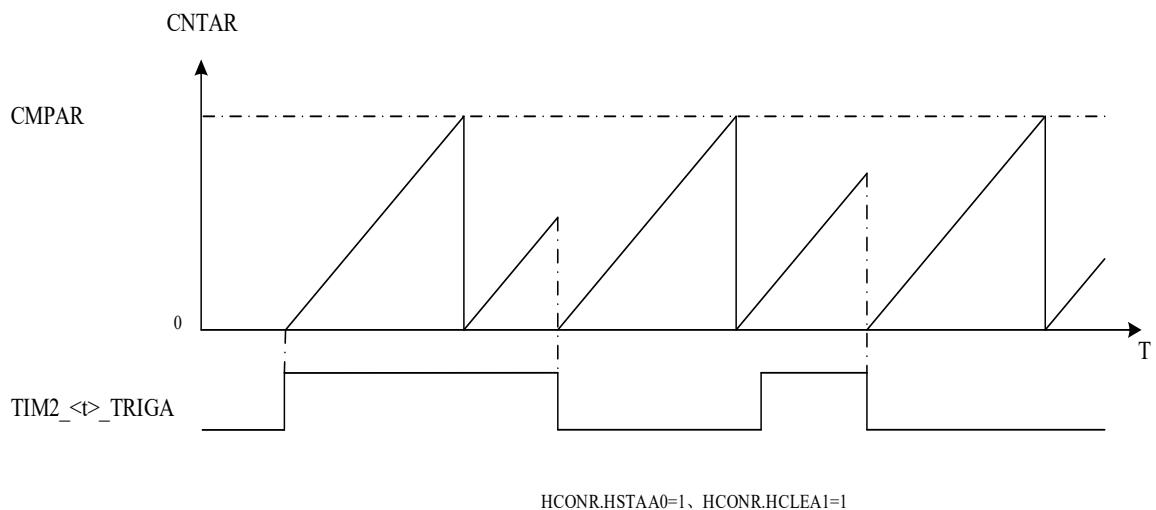


Figure 25-3 Hardware startup, reset action

25.3.3.2 Capture input action

When TIM2_<t>_TRIGA/B port input or internal trigger event is used as the capture input condition, the capture input function can be realized. When the capture input condition selected by the Hardware Control register (HCONR) is valid, the current count value is saved to the corresponding register (CMPAR, CMPBR). Figure 25-4 shows an example of the operation of capturing input.

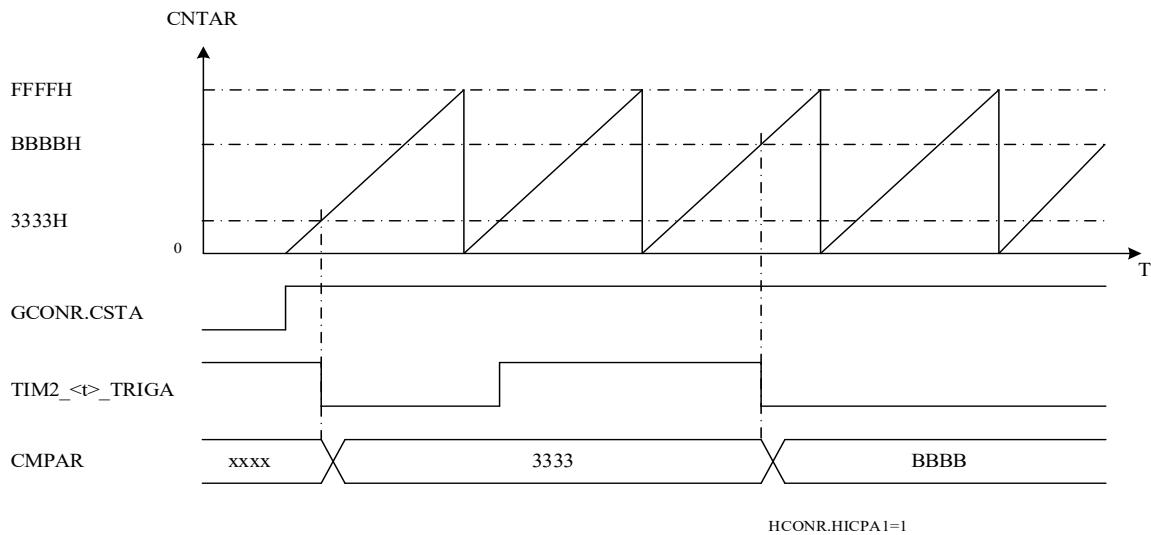


Figure 25-4 Capture input action

25.3.3.3 Pulse width measurement

Set the hardware start condition of channel A to the rising edge of TIM2_<t>_TRIGA, and set the hardware clear condition, stop condition and capture input condition to the falling edge of TIM2_<t>_TRIGA, so that continuous pulse width measurement can be achieved. The channel B can achieve the same function. corresponding actions such as Figure 25-5 shown.

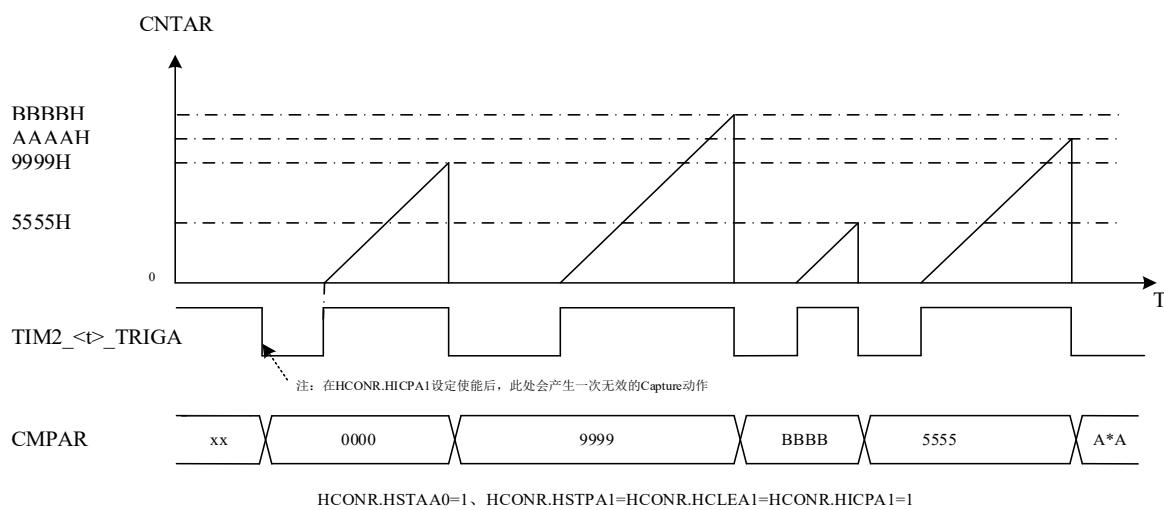


Figure 25-5 Pulse width measurement

25.3.3.4 Period measurement

By setting the hardware start condition, hardware clear condition and capture input condition of channel A to the same edge (rising or falling edge) of TIM2_<t>_TRIGA, continuous period width measurement can be achieved. The channel B can achieve the same function. corresponding actions such as Figure 25-6 shown.

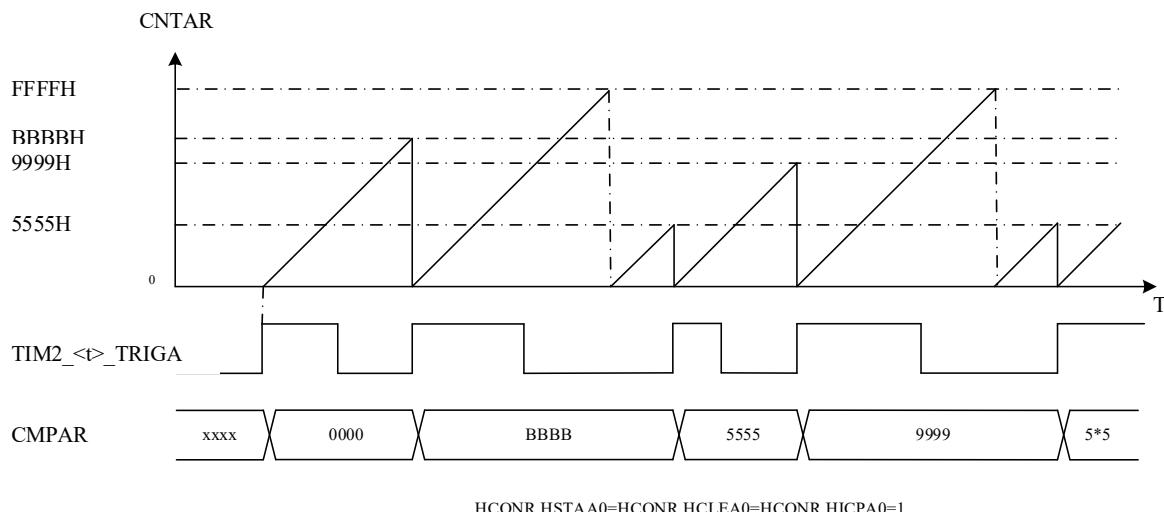


Figure 25-6 Period measurement

25.3.4 Digital filtering

The TRIG input port (TIM2_<t>_TRIGA/B) of Timer2 has a digital filtering function. The filter function of the corresponding port can be enabled by setting the enable bit of the Port Control Register (PCONR.NOFIENA). The reference clock for filtering is also set by the Port Control Register (PCONR.NOFICKA[1:0]).

When the filtered sampling reference clock is sampled to the same level three times on the port, the level is transferred to the module as an effective level. A level less than three times consistent will be filtered out as external interference and not transmitted to the module. Actions such as Figure 25-7 shown.

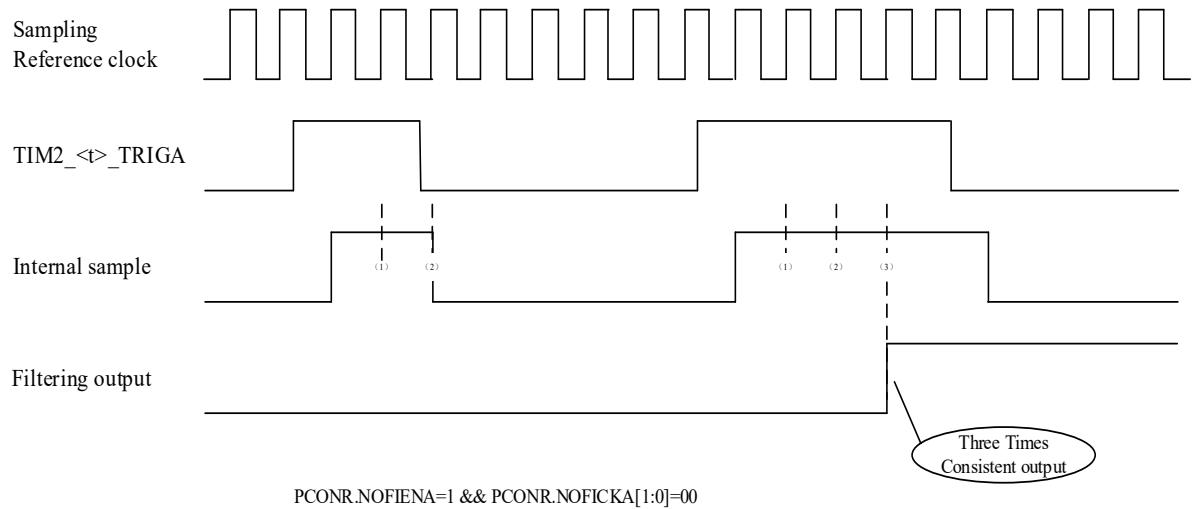


Figure 25-7 Digital filtering of the TRIG input port

25.4 Interrupt and Event Description

25.4.1 Interrupt output

A Timer2 contains 4 interrupts, which are the count compare match interrupt or capture input interrupt of channel A and channel B, and the count overflow interrupt of channel A and channel B.

There are 2 Compare Registers (CMPAR, CMPBR), which can be compared with the Counter Registers (CNTAR, CNTBR) to generate a compare match valid signal. When count compare match, the STFLR.CMFA bits in the Status Flags Register (STFLR) are set to 1, respectively. At this time, if the ICONR.CMENA bit of the Interrupt Control Register (ICONR) is set to enable the interrupt, the corresponding interrupt request (TMR2_<t>_CMPm, m=A, B) will also be triggered.

When the capture input valid condition selected by the Hardware Control Register (HCONR) is generated, the corresponding capture input action can be generated. At this time, if the ICONR.CMENA bit of the Interrupt Control Register (ICONR) is set to enable the interrupt, the corresponding interrupt request (TMR2_<t>_CMPm, m=A, B) is triggered.

A count overflow event occurs when the Counter Register (CNTAR, CNTBR) reaches 0xFFFF, and the STFLR.OVFA bit in the Status Flag Register (STFLR) will be set to 1 at this time. If the ICONR.OVENA bit in the Interrupt Control Register (ICONR) is set to enable the interrupt, the corresponding interrupt request (TMR2_<t>_OVFm, m=A, B) will also be triggered.

When the asynchronous counting mode is selected (except BCONR.ASYNCLKA[1:0]=2'b10), the compare match interrupt generated by the Compare Value A Register (CMPAR) of U1 can be used to wake up the system in low power mode. For details, please refer to the INTC chapter.

25.4.2 Event output

A Timer2 contains 4 events, which are the count compare match event of channel A and channel B or capture input event, and the count overflow event of channel A and channel B.

When a count comparison match or capture input action or count overflow occurs during the counting process, the corresponding event request (TMR2_<t>_CMPm or TMR2_<t>_OVFm, m=A, B) output signal will be generated, which can be used to select the trigger other modules.

25.5 Register description

Table 25-3 shown is the register list of the Timer2 module.

BASE ADDR:

0x40024800 (U1), 0x40024C00 (U2), 0x40025000 (U3), 0x40025400 (U4)

Table 25-3 Timer2 Register List

Register name	Symbol	Offset	Bit width	Reset value
Counter A Register	TMR2_CNTAR	0x0000h	32	0x00000000h
Counter B Register	TMR2_CNTBR	0x0004h	32	0x00000000h
Compare Value A Register	TMR2_CMPAR	0x0008h	32	0x0000FFFFh
Compare Value B Register	TMR2_CMPBR	0x000Ch	32	0x0000FFFFh
Basic Control Register	TMR2_BCONR	0x0010h	32	0x00000000h
Interrupt Control Register	TMR2_ICONR	0x0014h	32	0x00000000h
Port Control Register	TMR2_PCONR	0x0018h	32	0x00000000h
Hardware Control Register	TMR2_HCONR	0x001ch	32	0x00000000h
Hardware Trigger Source Select Register	TMR2_HTSSR	(0x40010874h)	32	0x000001FFh
Status Flag Register	TMR2_STFLR	0x0020h	32	0x00000000h

Note:

- The Hardware Trigger Source Select Register (TMR2_HTSSR) is an independent register that is shared by the 4 units of Timer2.

25.5.1 Counter Register (TMR2_CNTmR) (m=A, B)

Reset value: 0x00000000h

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[15:0]															

Bit	Symbol	Bit name	Description	Read and write
b31~b16	Reserved	-	Read as "0"	R
b15~b0	CNTA[15:0]	Counter value	Current timer count value	R/W

25.5.2 Compare Value Register (TMR2_CMPmR) (m=A, B)

Reset value: 0x0000FFFFh

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
CMPA[15:0]															

Bit	Symbol	Bit name	Description	Read and write
b31~b16	Reserved	-	Read as "0"	R
b15~b0	CMPA[15:0]	Compare value	Setting the count compare reference value to generate the Compare Match event	R/W

25.5.3 Basic Control Register (TMR2 _ BCONR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	SYN CLKBT[1:0]	ASYN CLKB[1:0]	SYN CLKB[1:0]			CKDIV B[3:0]			SYNS B	-	CAP MDB	CST B		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	SYN CLKAT[1:0]	ASYN CLKA[1:0]	SYN CLKA[1:0]			CKDIV A[3:0]			SYNS A	-	CAP MDA	CST A		
<hr/>															
Bit	Symbol	Bit name		Description										Read and write	
b31~b30	Reserved	-		Read as "0", write as "0"										R/W	
b29	SYNCLKBT[1]	Synchronous counting and clock source selection BT		Conditions: Count underflow occurs in unit n of Timer6 0: When the condition is matched, the timer2 unit m synchronization count is invalid 1: When the condition is matched, Timer2 unit m performs a synchronization count (When m=1, 2, 3, 4, n=2, 4, 6, 8)										R/W	
b28	SYNCLKBT[0]	Synchronous counting and clock source selection BT		Conditions: Count overflow occurs in unit n of Timer6 0: When the condition is matched, the timer2 unit m synchronization count is invalid 1: When the condition is matched, Timer2 unit m performs a synchronization count (When m=1, 2, 3, 4, n=2, 4, 6, 8)										R/W	
b27~b26	ASYNCLKB[1:0]	Asynchronous counting and clock source selection B		00: LRC 01: XTAL32 10: TIM2_<t>_CLKB clock input 11: Prohibitions										R/W	
b25~b24	SYNCLKB[1:0]	Synchronous counting and clock source selection B		00: PCLK1 01: TIM2_<t>_TRIGB rising edge (internal PCLK1 synchronization) 10: TIM2_<t>_TRIGB falling edge (internal PCLK1 synchronization) 11: Internal Trigger Event Input										R/W	
b23~b20	CKDIVB[3:0]	Count clock frequency division selection B		Counting clock frequency division options: 0000: Clock source 0001: Clock source/2 0010: Clock source/4 0011: Clock source/8 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 frequency-divided clock source can be various clock sources during asynchronous counting and PCLK1 during synchronous counting										R/W	
b19	SYNSB	Counting mode selection B		0: Synchronous counting mode 1: Asynchronous counting mode										R/W	
b18	Reserved	-		Read as "0", write as "0"										R/W	
b17	CAPMDB	Function mode selection B		0: Compare output function 1: Capture input function										R/W	
b16	CSTB	Timer startup B		0: Channel B timer is off 1: Channel B timer starts Note: This bit will automatically change to 0 when the hardware-triggered stop condition is valid.										R/W	
b15~b14	Reserved	-		Read as "0", write as "0"										R/W	
b13	SYNCLKAT[1]	Synchronous counting and clock source selection AT		Conditions: Count underflow occurs in unit n of Timer6 0: When the condition is matched, the timer2 unit m synchronization count is invalid 1: When the condition is matched, Timer2 unit m performs a synchronization count (When m=1, 2, 3, 4, n=1, 3, 5, 7)										R/W	
b12	SYNCLKAT[0]	Synchronous counting and clock source selection AT		Conditions: Count overflow occurs in unit n of Timer6 0: When the condition is matched, the timer2 unit m synchronization count is invalid 1: When the condition is matched, Timer2 unit m performs										R/W	

			a synchronization count (When m=1, 2, 3, 4, n=1, 3, 5, 7)	
b11~b10	ASYNCLKA[1:0]	Asynchronous counting with clock source selection A	00: LRC 01: XTAL32 10: TIM2_<t>_CLKA clock input 11: Prohibitions 00: PCLK1 01: TIM2_<t>_TRIGA rising edge (internal HCLK synchronization) 10: TIM2_<t>_TRIGA falling edge (internal HCLK synchronization) 11: Internal Trigger Event Input	R/W
b9~b8	SYNCLKA[1:0]	Synchronous counting with clock source selection A		R/W
b7~b4	CKDIVA[3:0]	Counting clock frequency division selection A	Counting clock frequency division options: 0000: Clock source 0001: Clock source/2 0010: Clock source/4 0011: Clock source/8 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 frequency-divided clock source can be various clock sources during asynchronous counting and PCLK1 during synchronous counting	R/W
b3	SYNSA	Counting mode selection A	0: Synchronous counting mode 1: Asynchronous counting mode	R/W
b2	Reserved	-	Read as "0", write as "0"	R/W
b1	CAPMDA	Functional mode selection A	0: Compare output function 1: Capture input function	R/W
b0	CSTA	Timer Startup A	0: Channel A timer is off 1: Channel A timer starts Note: This bit will automatically change to 0 when the hardware-triggered stop condition is valid.	R/W

25.5.4 Interrupt Control Register (TMR 2 _ ICONR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved												OVENB	CMENB		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved												OVENA	CMENA		
<hr/>															
Bit	Symbol	Bit name	Description												Read and write
b31~b18	Reserved	-	Read as "0", write as "0"												R/W
b17	OVENB	Count overflow interrupt enable B	0: When the count value (CNTBR) = 0xFFFFh, the interrupt is invalid 1: When the count value (CNTBR) = 0xFFFFh, the interrupt is enabled												R/W
b16	CMENB	Count match interrupt enable B	0: When the CMPBR register is equal to the count value (CNTBR), or when a capture input event occurs, the interrupt is invalid. 1: When the CMPBR register is equal to the count value (CNTBR), or when a capture input event occurs, the interrupt is enabled												R/W
b15~b2	Reserved	-	Read as "0", write as "0"												R/W
b1	OVENA	Count Overflow interrupt enable A	0: When the count value (CNTAR) = 0xFFFFh, the interrupt is disabled 1: When the count value (CNTAR) = 0xFFFFh, the interrupt is enabled												R/W
b0	CMENA	Count Match interrupt enable A	0: When CMPAR register is equal to the count value (CNTAR), or a capture input event occurs, the interrupt is invalid 1: When CMPAR register is equal to the count value (CNTAR), or a capture input event occurs, the interrupt is enabled.												R/W

25.5.5 Port Control Register (TMR2 _ PCONR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	NOFI CKB[1:0]	NOFI ENB	-	-	-	OUT ENB	-	-	-	CMP CB[1:0]	STP CB[1:0]	STA CB[1:0]			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	NOFI CKA[1:0]	NOFI ENA	-	-	-	OUT ENA	-	-	-	CMP CA[1:0]	STP CA[1:0]	STA CA[1:0]			

Bit	Symbol	Bit name	Description	Read and write
b31	Reserved	-	Read as "0", write as "0"	R/W
b30~b29	NOFICKB	Filter sampling reference clock selection B	00: Clock source 01: Clock source/4 10: Clock source/16 11: Clock source/64 Note: The clock source is various clock sources during asynchronous counting and PCLK1 during synchronous counting	R/W
b28	NOFIENB	Trigger Port Filter B	0: TIM2_<t>_TRIGB input port filtering function is invalid 1: TIM2_<t>_TRIGB input port filter function is enabled	R/W
b27~b25	Reserved	-	Read as "0", write as "0"	R/W
b24	OUTENB	Port output enable B	0: TIM2_<t>_PWMB port output is invalid 1: TIM2_<t>_PWMB port output is valid	R/W
b23~b22	Reserved	-	Read as "0", write as "0"	R/W
b21~b20	CMPCB[1:0]	Port state setting B when the compare value matches	00: When CNTBR is equal to CMPBR, the TIM2_<t>_PWMB port output is set to low level 01: When CNTBR is equal to CMPBR, the TIM2_<t>_PWMB port output is set to high level 10: When CNTBR is equal to CMPBR, the TIM2_<t>_PWMB port output maintains the previous state 11: When CNTBR is equal to CMPBR, the TIM2_<t>_PWMB port output is set to the inverted level	R/W
b19~b18	STPCB[1:0]	Port status setting B when counting is stopped	00: When counting stops, the TIM2_<t>_PWMB port output is set to low level 01: When counting stops, the TIM2_<t>_PWMB port output is set to high level 10: When counting stops, the TIM2_<t>_PWMB port output maintains the previous state 11: When counting stops, the TIM2_<t>_PWMB port output maintains the previous state	R/W
b17~b16	STACB[1:0]	Port state setting B when counting starts	00: When counting starts, the TIM2_<t>_PWMB port output is set to low level 01: When counting starts, the TIM2_<t>_PWMB port output is set to high level 10: When counting starts, the TIM2_<t>_PWMB port output maintains the previous state 11: When counting starts, the TIM2_<t>_PWMB port output maintains the previous state Note: This bit setting is only valid when the frequency is not divided (BCONR.CKDIVB=4'h0). For other frequency divisions, please set it to 2'b10 or 2'b11	R/W
b15	Reserved	-	Read as "0", write as "0"	R/W
b14~b13	NOFICKA	Filtering sampling reference clock selection A	00: Clock source 01: Clock source/4 10: Clock source/16 11: Clock source/64 Note: The clock source is various clock sources during asynchronous counting and PCLK1 during synchronous counting	R/W
b12	NOFIENA	Trigger Port Filter A	0: Invalid TIM2_<t>_TRIGA input port filtering function 1: TIM2_<t>_TRIGA input port filter enabled	R/W
b11~b9	Reserved	-	Read as "0", write as "0"	R/W
b8	OUTENA	Port Output Enable A	0: TIM2_<t>_PWMA port output invalid 1: TIM2_<t>_PWMA port output enabled	R/W
b7~b6	Reserved	-	Read as "0", write as "0"	R/W

b5~b4	CMPCA[1:0]	Port State Setting A When Compare Values Match	00: When CNTAR is equal to CMPAR, the TIM2_<t>_ PWMA port output is set to low level	R/W
			01: When CNTAR is equal to CMPAR, the TIM2_<t>_ PWMA port output is set to high level	
			10: When CNTAR is equal to CMPAR, the TIM2_<t>_ PWMA port output remains in the previous state	
			11: When CNTAR is equal to CMPAR, the TIM2_<t>_ PWMA port output is set to the inverted level	
b3~b2	STPCA[1:0]	Port Status Setting A when counting is Stopped	00: When counting stops, the TIM2_<t>_ PWMA port output is set to low level	R/W
			01: When counting stops, the TIM2_<t>_ PWMA port output is set to high level	
			10: When counting stops, the TIM2_<t>_ PWMA port output remains in the previous state	
			11: When counting stops, the TIM2_<t>_ PWMA port output remains in the previous state	
b1~b0	STACA[1:0]	Count port status setting A when counting starts	00: When counting starts, the TIM2_<t>_ PWMA port output is set to low level	R/W
			01: When counting starts, the TIM2_<t>_ PWMA port output is set to high level	
			10: When counting starts, the TIM2_<t>_ PWMA port output remains in the previous state	
			11: When counting starts, the TIM2_<t>_ PWMA port output remains in the previous state	
Note: This bit setting is only valid when the frequency is not divided (BCONR.CKDIVA=4'h0). For other frequency divisions, please set it to 2'b10 or 2'b11				

25.5.6 Hardware Control Register (TMR 2 _ HCONR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	HICP B2	HICP B1	HICP B0	-	HCLE B2	HCLE B1	HCLE B0	-	HSTP B2	HSTP B1	HSTP B0	-	HSTA B2	HSTA B1	HSTA B0
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	HICP A2	HICP A1	HICP A0	-	HCLE A2	HCLE A1	HCLE A0	-	HSTP A2	HSTP A1	HSTP A0	-	HSTA A2	HSTA A1	HSTA A0

Bit	Symbol	Bit name	Description	Read and write
b31	Reserved	-	Read as "0", write as "0"	R/W
b30	HICPB2	Hardware capture input condition B2	Conditions: Internal hardware trigger event is valid 0: Hardware capture input is invalid when conditions match 1: Hardware capture input is valid when conditions match	R/W
b29	HICPB1	Hardware capture input condition B1	Conditions: TIM2_<t>_TRIGB port sampled falling edge 0: Hardware capture input is invalid when conditions match 1: Hardware capture input is valid when conditions match	R/W
b28	HICPB0	Hardware capture input condition B0	Conditions: TIM2_<t>_TRIGB port sampled rising edge 0: Hardware capture input is invalid when conditions match 1: Hardware capture input is valid when conditions match	R/W
b27	Reserved	-	Read as "0", write as "0"	R/W
b26	HCLEB2	Hardware clear condition B2	Conditions: Internal hardware trigger event is valid 0: Hardware clear is invalid when conditions match 1: Hardware clear is valid when conditions match	R/W
b25	HCLEB1	Hardware clear condition B1	Conditions: TIM2_<t>_TRIGB port sampled falling edge 0: Hardware clear is invalid when conditions match 1: Hardware clear is valid when conditions match	R/W
b24	HCLEB0	Hardware clear condition B0	Conditions: TIM2_<t>_TRIGB port sampled rising edge 0: Hardware clear is invalid when conditions match 1: Hardware clear is valid when conditions match	R/W
b23	Reserved	-	Read as "0", write as "0"	R/W
b22	HSTPB2	Hardware Stop Condition B2	Conditions: Internal hardware trigger event is valid 0: Hardware stop is invalid when conditions match 1: Hardware stop is valid when conditions match	R/W
b21	HSTPB1	Hardware Stop Condition B1	Conditions: TIM2_<t>_TRIGB port sampled falling edge 0: Hardware stop is invalid when conditions match 1: Hardware stop is valid when conditions match	R/W
b20	HSTPB0	Hardware stop condition B0	Conditions: TIM2_<t>_TRIGB port sampled rising edge 0: Hardware stop is invalid when condition matches 1: Hardware stop is valid when conditions match	R/W
b19	Reserved	-	Read as "0", write as "0"	R/W
b18	HSTAB2	Hardware Start Condition B2	Conditions: Internal hardware trigger event is valid 0: Hardware start is invalid when conditions match 1: Hardware start is valid when conditions match	R/W
b17	HSTAB1	Hardware Start Condition B1	Conditions: TIM2_<t>_TRIGB port sampled falling edge 0: Hardware start is invalid when conditions match 1: Hardware start is valid when conditions match	R/W
b16	HSTAB0	Hardware Start Condition B0	Conditions: TIM2_<t>_TRIGB port sampled rising edge 0: Hardware start is invalid when conditions match 1: Hardware start is valid when conditions match	R/W
b15	Reserved	-	Read as "0", write as "0"	R/W
b14	HICPA2	Hardware Capture Input Condition A2	Conditions: Internal hardware trigger event is valid 0: Hardware capture input is invalid when conditions match 1: Hardware capture input is valid when conditions match	R/W
b13	HICPA1	Hardware Capture Input Condition A1	Conditions: TIM2_<t>_TRIGA port sampled falling edge 0: Hardware capture input is invalid when conditions match 1: Hardware capture input is valid when conditions match	R/W
b12	HICPA0	Hardware Capture Input Condition A0	Conditions: TIM2_<t>_TRIGA port sampled rising edge 0: Hardware capture input is invalid when conditions match 1: Hardware capture input is valid when conditions match	R/W
b11	Reserved	-	Read as "0", write as "0"	R/W
b10	HCLEA2	Hardware clear condition A2	Conditions: Internal hardware trigger event is valid 0: Hardware clear is invalid when conditions match 1: Hardware clear is valid when conditions match	R/W
b9	HCLEA1	Hardware clear condition A1	Conditions: TIM2_<t>_TRIGA port sampled falling edge 0: Hardware clear is invalid when conditions match	R/W

			1: Hardware clear is valid when conditions match	
b8	HCLEA0	Hardware clear condition A0	Conditions: TIM2_<t>_TRIGA port sampled rising edge 0: Hardware clear is invalid when conditions match 1: Hardware clear is valid when conditions match	R/W
b7	Reserved	-	Read as "0", write as "0"	R/W
b6	HSTPA2	Hardware stop condition A2	Conditions: Internal hardware trigger event is valid 0: Hardware stop is invalid when conditions match 1: Hardware stop is valid when conditions match	R/W
b5	HSTPA1	Hardware stop condition A1	Conditions: TIM2_<t>_TRIGA port sampled falling edge 0: Hardware stop invalid when condition match 1: Hardware stop is valid when conditions match	R/W
b4	HSTPA0	Hardware stop condition A0	Conditions: TIM2_<t>_TRIGA port sampled rising edge 0: Hardware stop is invalid when condition matches 1: Hardware stop is valid when conditions match	R/W
b3	Reserved	-	Read as "0", write as "0"	R/W
b2	HSTAA2	Hardware start condition A2	Conditions: Internal hardware trigger event is valid 0: Hardware start is invalid when conditions match 1: Hardware start is valid when conditions match	R/W
b1	HSTAA1	Hardware start condition A1	Conditions: TIM2_<t>_TRIGA port sampled falling edge 0: Hardware start is invalid when conditions match 1: Hardware start is valid when conditions match	R/W
b0	HSTAA0	Hardware start condition A0	Conditions: TIM2_<t>_TRIGA port sampled rising edge 0: Hardware start is invalid when conditions match 1: Hardware start is valid when conditions match	R/W

25.5.7 Hardware Source Trigger Select Register (TMR 2 _ HTSSR)

Reset value: 0x0000001FFh

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]							

Bit	Symbol	Bit name	Description	Read and write
b31	COMEN[1]	Common trigger enable 1	0: Disable the common trigger event of AOS_COMTRG2 to trigger Timer2 1: Enable the common trigger event of AOS_COMTRG2 to trigger Timer2 See the Automatic Operating System (AOS) chapter for details	R/W
b30	COMEN[0]	Common trigger enable 0	0: Disable the common trigger event of AOS_COMTRG1 to trigger Timer2 1: Enable the common trigger event of AOS_COMTRG1 to trigger Timer2 See the Automatic Operating System (AOS) chapter for details	R/W
b29~b9	Reserved	-	Read as "0", write as "0"	R/W
b8~b0	TRGSEL	Trigger source selection	To write the trigger source number, refer to [Interrupt controller (INTC)] chapter	R/W

25.5.8 Status Flag Register (TMR2 _ STFLR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved												OVFB	CMFB		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved												OVFA	CMFA		
Bit	Symbol	Bit name	Description	Read and write											
b31~b18	Reserved	-	Read as "0", write as "0"	R											
b17	OVFB	Count overflow flag B	0: The count value (CNTBR) does not count to 0xFFFFh 1: Count value (CNTBR) counts to 0xFFFFh	R/W											
b16	CMFB	Count match flag B	0: The value of the CMPBR register is not equal to the count value (CNTBR) and no capture input action has occurred 1: The value of the CMPBR register is equal to the count value (CNTBR) or a capture input action occurs	R/W											
b15~b2	Reserved	-	Read as "0", write as "0"	R											
b1	OVFA	Count overflow flag A	0: The count value (CNTAR) does not count to 0xFFFFh 1: Count value (CNTAR) counts to 0xFFFFh	R/W											
b0	CMFA	Count match flag A	0: The value of CMPAR register is not equal to the count value (CNTAR) and no capture input action occurs 1: The value of CMPAR register is equal to the count value (CNTAR) or captures the input action. occurs	R/W											

25.6 Precautions for use

1. In the asynchronous counting operation, it is necessary to set the BCONR.ASYNCLKA bit to select the asynchronous clock source, and then set the BCONR.SYNSA bit to select the asynchronous counting mode, and then start Timer2.
2. When asynchronous counting is selected, modify the counter value (CNTAR), compare value (CMPAR), start bit (BCONR.CSTA), status bit (STFLR.CMFA) When Timer2 receives the write action, it will only write the modified value into the corresponding register after 3 asynchronous counting clocks.
3. When asynchronous counting is selected, counter value (CNTAR), compare value (CMPAR), start bit (BCONR.CSTA), status bit (STFLR.CMFA)) to perform the write operation continuously, at least 3 asynchronous counting clocks interval are required.
4. When asynchronous counting is selected, set BCONR.SYNCLKA[1:0] to 0.

26 General Timer (Timer0)

26.1 Introduction

General-purpose timer 0 (Timer0) is a basic timer that can realize synchronous counting and asynchronous counting. The timer contains 2 channels (CH-A and CH-B) that can generate compare match events during counting. The event can trigger interrupt, or can be output as an event to control other modules. This product family has 2 units of Timer0.

26.2 Basic block diagram

The basic block diagram of Timer0 is shown in Figure 26-1.

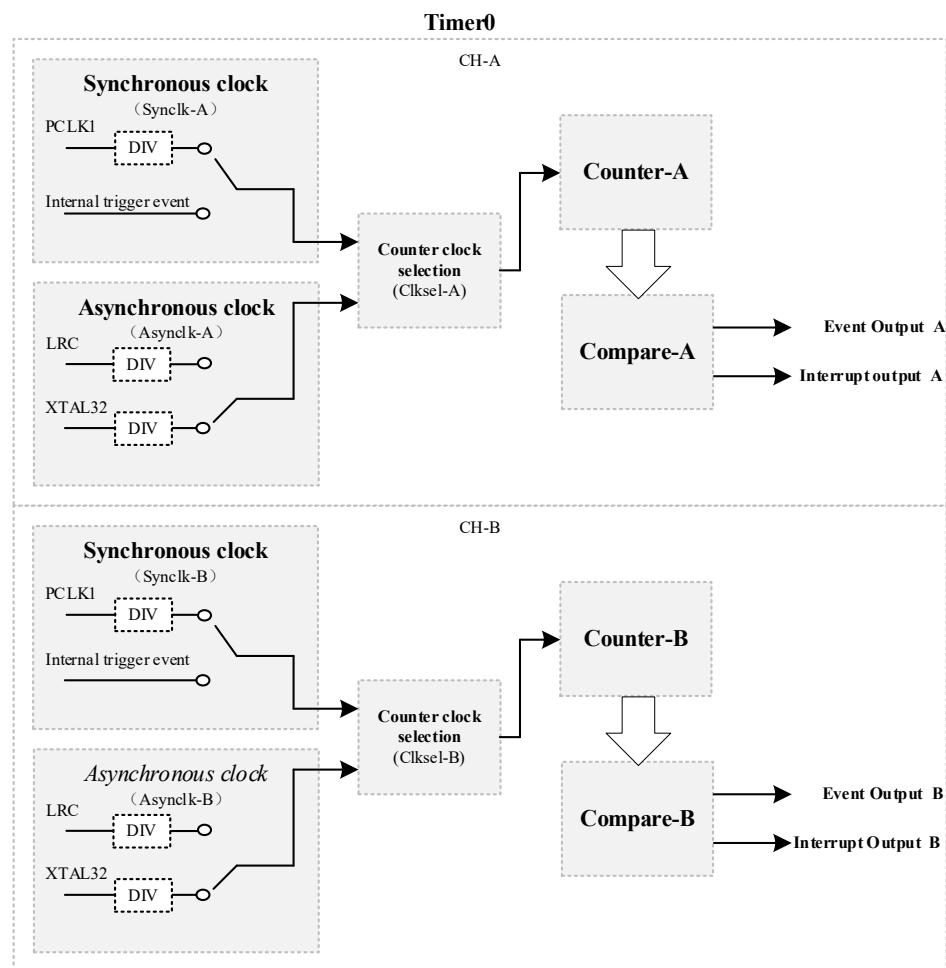


Figure 26-1 Timer0 basic block diagram

26.3 Functional description

26.4 Clock source selection

Timer0 can be counted in synchronous or asynchronous mode.

The synchronous counting mode refers to the synchronous timing relationship between the timer count clock and the bus access clock (register read/write operation clock). Asynchronous counting mode means that the timer count clock and bus access clock (register read/write operation clock) are non-synchronous timing relationships. The status of timer, etc., may be changing when the register is read by asynchronous counting. Therefore, in the asynchronous counting mode, the register read operation must be implemented in the count stopped state.

26.4.1.1 Synchronous counting clock source

In the synchronous counting mode (BCONR.SYNSA=0), the clock source can be selected from the following options (BCONR.SYNCLKA setting):

1. The 2, 4, 8, 16, 32, 64, 128, 256, 512, and 1024 frequency divisions of PCLK1 and PCLK1 are used as the synchronous counting clock source(BCONR.SYNCLK A=0 & BCONR.CKDIV A [3: 0] settings)
2. Internal hardware trigger event input as synchronous counting clock (BCONR.SYNCLK A=1)

26.4.1.2 Asynchronous counting clock source

In asynchronous counting mode (BCONR.SYNS A=1), the clock source can have the following options (BCONR.ASYNCLK A setting selection):

1. LRC clock source input and its 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 frequency division as asynchronous counting clock source (BCONR.ASYNCLK A=0 & BCONR.CKDIV A [3:0] settings)
2. XTAL32 clock source input and its 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 frequency division as asynchronous counting clock source (BCONR.ASYNCLK A=1 & BCONR.CKDIV A [3:0] settings)

26.4.2 Basic count

Each channel of Timer0 can set a compare reference count value, and generate a count compare match event when the counter value and the compare value are equal, as shown in Figure 26-2.

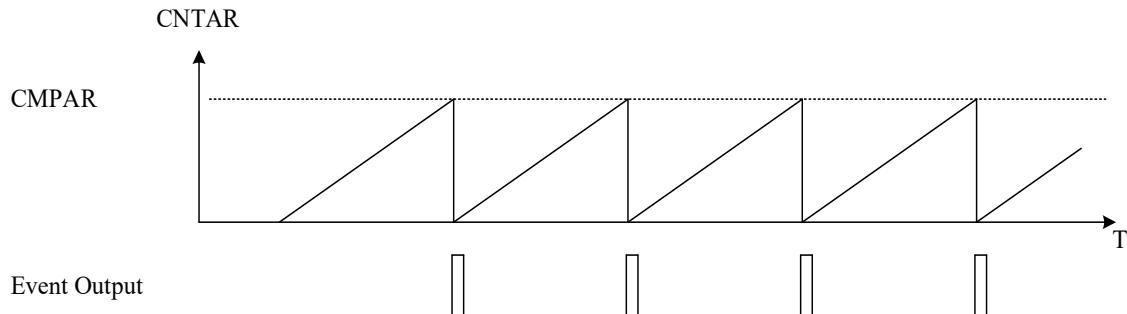


Figure 26-2 Timer0 count timing diagram

26.4.3 Hardware triggering

The two channels of Timer0 have a shared internal hardware trigger source, which can control the state of the timer (count, start, stop, clear) and capture input actions through the relevant settings of the Basic Control Register (BCONR).

The source selection of the hardware trigger is realized by entering the corresponding number into the Hardware Trigger Source Selection Register (HTSSR). For the specific event correspondence, refer to the INTC chapter. When using the internal hardware trigger function, the peripheral circuit trigger function bit of the Function Clock Control Register 0 (PWC_FCG0) needs to be enabled first.

26.5 Interrupt and Event Description

26.5.1 Interrupt output

A Timer0 contains 2 interrupt outputs, which are the count compare match interrupt of channel A and channel B or capture input interrupt.

There are 2 Compare Value Registers (CMPAR, CMPBR), which can be compared with the Counter Registers (CNTAR, CNTBR) to generate a compare match valid signal. When a count compare match, the STFLR.CMFA bits in the Status Flag Register (STFLR) are set to 1, respectively. At this time, if the BCONR.INTENA bit of the Basic Control Register (BCONR) is set to enable the interrupt, the corresponding interrupt request (TMR0_m_CMPn, m=1, 2; n=A, B) will also be triggered.

When the internal hardware triggers input as the capture input condition, the corresponding capture input action can be generated. At this time, if the BCONR.INTENA bit of the Basic Control Register (BCONR) is set to enable the interrupt, the corresponding interrupt request (TMR0_m_CMPn, m=1, 2; n=A, B) is triggered.

When the asynchronous counting mode is selected, the compare match interrupt generated by the Compare Value Register (CMPAR) of U1 can be used to wake up the system in the low power mode. For details, please refer to the INTC chapter.

26.5.2 Event output

A Timer0 contains 2 event outputs, which are count compare match events of channel A and channel B or capture input events.

When a count compare match or capture input action occurs during the counting process, the corresponding event request (TMR0_m_CMPn, m=1, 2; n=A, B) output signal will be generated respectively, which can be used to trigger other modules.

26.6 Register description

Table 26-1 shown is the register list of the Timer0 module.

BASE ADDR: 0x40024000 (U1), 0x40024400 (U2)

Table 26-1 Timer0 register list

Register name	Symbol	Offset	Bit width	Reset value
Counter A Register	TMR0_CNTAR	0x0000h	32	0x00000000h
Counter B Register	TMR0_CNTBR	0x0004h	32	0x00000000h
Compare Value A Register	TMR0_CMPAR	0x0008h	32	0x0000FFFFh
Compare Value B Register	TMR0_CMPBR	0x000ch	32	0x0000FFFFh
Basic Control Register	TMR0_BCONR	0x0010h	32	0x00000000h
Hardware Trigger Source Selection Register	TMR0_HTSSR	(0x40010870h)	32	0x000001FFh
Status Flag Register	TMR0_STFLR	0x0014h	32	0x00000000h

Note:

- The Hardware Trigger Source Selection Register (TMR0_HTSSR) is an independent register that is shared by 2 units of Timer0.

26.6.1 Counter Register (TMR0_CNTmR) (m=A~B)

Reset value: 0x00000000h

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[15:0]															

Bit	Symbol	Bit name	Description	Read and write
b31~b16	Reserved	-	Read as "0"	R
b15~b0	CNTA[15:0]	Counter value	Current timer count value.	R/W

26.6.2 Compare Value Register (TMR0_CMPmR) (m=A~B)

Reset value: 0x0000FFFFh

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
CMPA[15:0]															

Bit	Symbol	Bit name	Description	Read and write
b31~b16	Reserved	-	Read as "0"	R
b15~b0	CMPA[15:0]	Compare value	Setting the count compare reference value to generate the Compare Match event.	R/W

26.6.3 Basic Control Register (TMR0 _ BCONR)

Reset value: 0x00000000h

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	

Bit	Symbol	Bit name	Description	Read and write
b31	HICPB	Hardware trigger capture input B	Conditions: Internal hardware trigger event is valid 0: Capture input is invalid when conditions match 1: Capture input is valid when conditions match	R/W
b30	HCLEB	Hardware trigger clear B	Conditions: Internal hardware trigger event is valid 0: Timer clear is invalid when conditions match 1: Timer clear is valid when conditions match	R/W
b29	HSTPB	Hardware trigger stop B	Conditions: Internal hardware trigger event is valid 0: Timer stop is invalid when conditions match 1: Timer stop is valid when the conditions match	R/W
b28	HSTAB	Hardware trigger start B	Conditions: Internal hardware trigger event is valid 0: Timer start is invalid when conditions match 1: Timer start is valid when conditions match	R/W
b27	Reserved	-	Read as "0", write as "0"	R/W
b26	ASYNCLKB	Channel B asynchronous counting clock source selection	0: LRC 1: XTAL32	R/W
b25	SYNCLKB	Channel B synchronous counting clock source selection	0: PCLK1 1: Internal Hardware Trigger Event	R/W
b24	SYNSB	Channel B counting mode selection	0: Synchronous counting mode 1: Asynchronous counting mode	R/W
b23~b20	CKDIVB[3:0]	Channel B counting clock divider selection	Channel B counting clock frequency division options: 0000: Clock source 0001: Clock source/2 0010: Clock source/4 0011: Clock source/8 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 divided clock source can be various clock sources during asynchronous counting and PCLK1 during synchronous counting	R/W
b19	Reserved	-	Read as "0", write as "0"	R/W
b18	INTENB	Count match interrupt enable B	0: When the CMPBR register is equal to the count value (CNTBR), or when a capture input event occurs, the interrupt is invalid 1: When the CMPBR register is equal to the count value (CNTBR), or when a capture input event occurs, the interrupt is enabled	R/W
b17	CAPMDB	Function mode selection B	0: Select compare output function 1: Select capture input function	R/W
b16	CSTB	Timer start	0: Channel B timer is off 1: Channel B timer starts Note: This bit will automatically change to 0 when the hardware-triggered stop condition is valid.	R/W
b15	HICPA	Hardware trigger capture input A	Conditions: Internal hardware trigger event is valid 0: Capture input is invalid when conditions match 1: Capture input is valid when conditions match	R/W
b14	HCLEA	Hardware trigger clear A	Conditions: Internal hardware trigger event is valid 0: Timer clear is invalid when conditions match 1: Timer clear is valid when conditions match	R/W

b13	HSTPA	Hardware trigger stop A	Conditions: Internal hardware trigger event is valid 0: Timer stop is invalid when conditions match 1: Timer stop is valid when conditions match	R/W
b12	HSTAA	Hardware trigger start A	Conditions: Internal hardware trigger event is valid 0: Timer start is invalid when conditions match 1: Timer start is valid when conditions match	R/W
b11	Reserved	-	Read as "0", write as "0"	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 Trigger Event	R/W
b8	SYNSA	Channel A counting mode selection	0: Synchronous counting mode 1: Asynchronous counting mode	R/W
b7~b4	CKDIVA[3:0]	Channel A counting clock divider selection	Channel A counting clock frequency division options: 0000: Clock source 0001: Clock source/2 0010: Clock source/4 0011: Clock source/8 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 divided clock source can be various clock sources during asynchronous counting and PCLK1 during synchronous counting	R/W
b3	Reserved	-	Read as "0", write as "0"	R/W
b2	INTENA	Count Match interrupt enable A	0: CMPA Register is equal to the count value (CNTAR), or when a capture input event occurs, the interrupt is invalid. 1: CMPA Register is equal to the count value (CNTAR), or when a capture input event occurs, the interrupt is enabled	R/W
b1	CAPMDA	Functional mode selection A	0: Select compare output function 1: Select capture input function	R/W
b0	CSTA	Timer start	0: Channel A timer is off 1: Channel A timer starts Note: This bit will automatically change to 0 when the hardware-triggered stop condition is valid.	R/W

Note:

- The internal hardware trigger events (bit31~bit28 and bit15~bit12) mentioned in this register and the XTAL32 clock source (bit26 and bit10) in asynchronous counting mode are all provided by the USART module when the TIMEOUT function of the USART module is valid. Refer to the USART chapter for details.

26.6.4 Hardware Trigger Source Selection Register (TMR0 _ HTSSR)

Reset value: 0x000001FFh

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]							

Bit	Symbol	Bit name	Description	Read and write
b31	COMEN[1]	Common trigger enable 1	0: Disable the common trigger event of AOS_COMTRG2 to trigger Timer0 1: Enable the common trigger event of AOS_COMTRG2 to trigger Timer0 See the Automatic Operating System (AOS) chapter for details	R/W
b30	COMEN[0]	Common trigger enable 0	0: Disable the common trigger event of AOS_COMTRG1 to trigger Timer0 1: Enable the common trigger event of AOS_COMTRG1 to trigger Timer0 See the Automatic Operating System (AOS) chapter for details	R/W
b29~b9	Reserved	-	Read as "0", write as "0"	R/W
b8~b0	TRGSEL	Trigger source selection	To write the trigger source number, refer to [Interrupt controller (INTC)] chapter	R/W

26.6.5 Status Flag Register (TMR0 _ STFLR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															CMFB
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															CMFA
<hr/>															
Bit	Symbol	Bit name	Description										Read and write		
b31~b17	Reserved	-	Read as "0", write as "0"										R/W		
b16	CMFB	Count Match B	0: The value of the CMPBR register is not equal to the count value (CNTAR) and no capture input action occurs 1: The value of the CMPBR register is equal to the count value (CNTAR) or a capture input action occurs.										R/W		
b15~b1	Reserved	-	Read as "0", write as "0"										R/W		
b0	CMFA	Count Match A	0: The value of the CMPAR register is not equal to the count value (CNTAR) and no capture input action occurs. 1: The value of CMPAR register is equal to the count value (CNTAR) or a capture input action occurs.										R/W		

26.7 Precautions for use

1. In the asynchronous counting operation, it is necessary to set the BCONR.ASYNCLKA bit to select the asynchronous clock source, and then set the BCONR.SYNSA bit to select the asynchronous counting mode, and then start Timer0.
2. When asynchronous counting is selected, modify the counter value (CNTAR), compare value (CMPAR), start bit (BCONR.CSTA), status bit (STFLR.CMFA), When Timer0 receives the write action, it will only write the modified value into the corresponding register after 3 asynchronous counting clocks.
3. When asynchronous counting is selected, counter value (CNTAR), compare value (CMPAR), start bit (BCONR.CSTA), status bit (STFLR.CMFA) to perform the write operation continuously, at least 3 asynchronous counting clocks interval are required.
4. When asynchronous counting is selected, set BCONR.SYNCLKA to 0.

27 Real Time Clock RTC

27.1 Introduction

A real-time clock (RTC) is a counter that stores time information in BCD code format. Record specific calendar times from 00 to 99. Supports 12/24 hour two time systems, automatically calculates the number of days 28, 29 (leap year), 30 and 31 according to the month and year. Table 27-1 shown are its basic characteristics.

Table 27-1 Basic Specifications of RTC

Count clock source	External low speed oscillator (32.768kHz) RTC internal low speed oscillator (32.768kHz)
Basic functions	• BCD code represents seconds, minutes, hours, days, weeks, months, and years
	• Software start or stop
	• 12/24 hour system optional, automatic leap year recognition
	• Programmable Alarm Clock
	• Distributed/Uniformly Compensated 1Hz Clock Output
	• Clock Error Compensation Function
	• Timestamp function
	• Backup register reset function
Interrupt	Periodic interrupt
	Alarm interrupt
	Intrusion event interrupt

27.2 Basic block diagram

The basic block diagram of RTC is as followsFigure 27-1 shown.

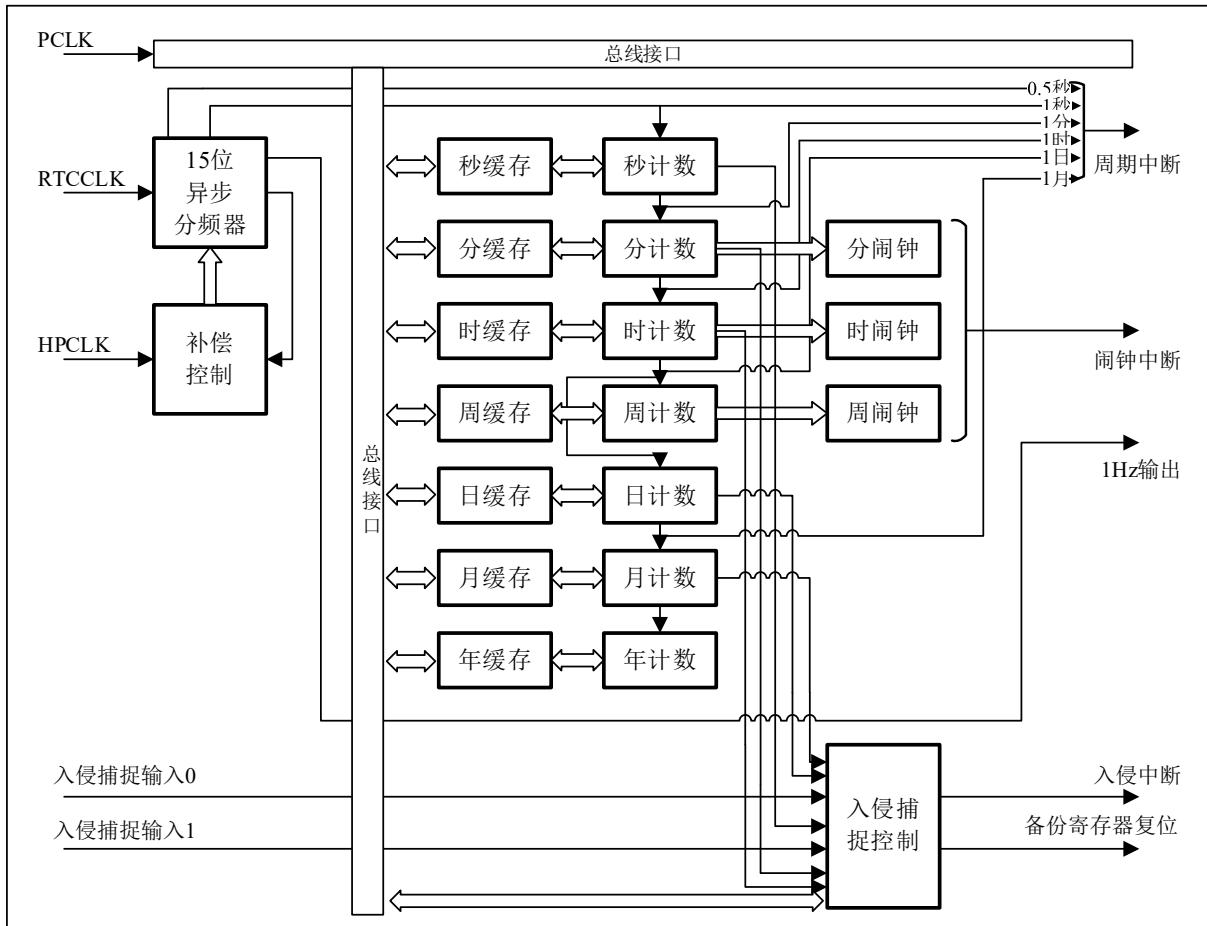


Figure 27-1 Basic block diagram of RTC

27.3 Functional description

27.3.1 Power-on settings

The RTC register value is not fixed after power-on reset. After power-on reset, the backup domain reset register PWC_VBATRSTR must be set first, and then the control register RTC_CR0.RESET bit must be set to reset all registers. Then set the control register, the initial value of the calendar, the alarm clock setting, and start the RTC. After the RTC is started, other external reset requests cannot reset the RTC, and the RTC will always be in a working state. The RTC_CR1.START bit of the control register can be set to "0" to stop the RTC operation. When setting the RTC, you must ensure that the clock source is stable.

27.3.2 RTC count start setting

1. After power-on, set RTC_CR0.RESET=0, after confirming that the RESET bit is "0", 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 external low-speed oscillator, and then set RTC_CR3 to select the RTC count clock source;
4. Set RTC_CR1, set time system, period, 1Hz clock output;
5. Set the calendar count register for seconds, minutes, hours, weeks, days, months, and years;
6. When clock error compensation is required, set the count clock error compensation registers RTC_ERRCRL and RTC_ERRCRH;
7. Clear register RTC_CR1, flag register bit in RTC_TPSR, and enable interrupt;
8. Set RTC_CR1.START=1 to start counting.

27.3.3 System low power mode switching

When the system switches to the low power consumption mode immediately after the RTC count starts, perform one of the following confirmations before switching the mode.

1. After RTC_CR1.START=1 is set, the mode switch 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, query RTC_CR2.RWEN=0, and switch the mode.

27.3.4 Read count register

1. After RTC_CR1.START=1 is set, after more than 2 RTC count clocks, set RTC_CR2.RWREQ=1 to read the calendar register request;
2. Query until RTC_CR2.RWEN=1;
3. Read out all or part of the second, minute, hour, week, day, month, year count register value;
4. Set RTC_CR2.RWREQ=0;
5. Query until RTC_CR2.RWEN=0.

27.3.5 Write count register

1. After RTC_CR1.START=1 is set, after more than 2 RTC count clocks, set RTC_CR2.RWREQ=1 to make a calendar register write request;
2. Query until RTC_CR2.RWEN=1;
3. Write all or part of the second, minute, hour, week, day, month, year count register value;
4. Set RTC_CR2.RWREQ=0. Note that all write operations must be completed within 1 second;
5. Query until RTC_CR2.RWEN=0.

27.3.6 Alarm setting

1. Set RTC_CR2.ALME=0, the alarm clock is disabled;
2. Set RTC_CR2.ALMIE=1, the alarm interrupt is allowed;
3. The minute alarm clock RTC_ALMMIN, the hour alarm clock RTC_ALMHOUR, and the weekly alarm clock RTC_ALMEEK are set;
4. Set RTC_CR2.ALME=1, the alarm clock is allowed;
5. wait for the alarm to be interrupted;
6. When the alarm occurs, RTC_CR2.ALMF=1, enter the alarm interrupt processing.

27.3.7 Clock Error Compensation

Due to the deviation of the external low-speed crystal oscillator under various temperature conditions, it is necessary to compensate for the error when high-precision counting results are required. Compensation method reference 27.5.15Clock Error Compensation Registers (RTC_ERRCRH, RTC_ERRCRL)].

27.3.8 1Hz output

The RTC can output a 1Hz clock and provides three precision output modes. The first one is the normal precision 1Hz output without clock compensation; the second one is the distributed compensation 1Hz output with average compensation every 32 seconds; Uniformly compensated 1Hz output. When the clock error compensation function is valid, RTC_ERRCRH.COMPEN=1, the

distributed compensation 1Hz output and the uniform compensation 1Hz output can be selected. in,

The normal precision 1Hz output settings are as follows:

- 1) Set RTC_CR0.RESET=0, after confirming that the RESET bit is "0", set RTC_CR0.RESET=1 to reset the calendar count register;
- 2) Set RTC_CR1.START=0, the counting stops;
- 3) 1Hz output pin setting;
- 4) RTC_CR1.ONEHZOE=1, the clock output is allowed;
- 5) Set RTC_CR1.START=1 to start counting;
- 6) Wait for more than 2 count cycles;
- 7) 1Hz output starts.

The distributed compensation 1Hz output settings are as follows:

- 1) Set RTC_CR0.RESET=0, after confirming that the RESET bit is "0", set RTC_CR0.RESET=1 to reset the calendar count register;
- 2) Set RTC_CR1.START=0, the counting stops;
- 3) 1Hz output pin setting;
- 4) RTC_CR1.ONEHZOE=1, the clock output is allowed;
- 5) Clock error compensation register RTC_ERRCRL.COMP[7:0] and RTC_ERRCRH.COMP[8] compensation number setting;
- 6) Clock error compensation register RTC_ERRCRH.COMPEN=1, error compensation is valid;
- 7) Set RTC_CR1.START=1 to start counting;
- 8) Wait for more than 2 count cycles;
- 9) 1Hz output starts.

The uniform compensation 1Hz output settings are as follows:

- 1) Set RTC_CR0.RESET=0, after confirming that the RESET bit is "0", set RTC_CR0.RESET=1 to reset the calendar count register;;
- 2) Set RTC_CR1.START=0, the counting stops;
- 3) RTC output pin setting;
- 4) RTC_CR1.ONEHZOE=1, the clock output is allowed;
- 5) RTC_CR1.ONEHZSEL=1, select to output uniform compensation 1Hz clock;
- 6) Clock error compensation register RTC_ERRCRL.COMP[7:0] and RTC_ERRCRH.COMP[8] compensation number setting;
- 7) Clock error compensation register RTC_ERRCRH.COMPEN=1, precision compensation is valid;
- 8) Set RTC_CR1.START=1 to start counting;
- 9) Wait for more than 2 count cycles;
- 10) 1Hz output starts.

27.3.9 Intrusion detection

When the intrusion detection RTC_TPCRn.TPEN(n=0,1) is valid, the external pins of the RTC can trigger the intrusion event detection function, the time stamp function and the backup register reset function respectively after detecting a valid intrusion event.

Intrusion event detection function: When an intrusion event occurs, the intrusion flag of the corresponding pin will be set up, and when the intrusion event occurs continuously, the overflow flag will be set up. When intrusion events 0 and 1 occur at the same time, the intrusion event 1 flag is reset after one RTC count clock is set, and the overflow flag is also set. When the RTC.TPCRn.TPIE (n=0,1) intrusion event interrupt is enabled, an intrusion detection interrupt will occur.

Timestamp function: When the time stamp function of RTC.TPCRn.TSTPE (n=0,1) is valid, the time stamp register will record the second, minute, hour, day and month calendar time when the intrusion event occurs.

Backup register reset: RTC.TPCRn.TPRSTE (n=0,1) When the backup register reset is enabled, the backup register will be reset after an intrusion event occurs. The backup register includes 32 32-bit registers for storing 128-byte backup data of the user.

Note:

- When the intrusion detection is valid, regardless of whether the time stamp function, the intrusion event interrupt and the backup register reset function are valid, the intrusion event flag RTC_TPSRn.TPFn (n=0,1) will be set after the event occurs.

27.3.10 Timestamp function

When the intrusion detection RTC_TPCRn.TPEN(n=0,1) is valid, and the time stamp recording function RTC_TPCRn.TSTPE (n=0,1) is valid, the external pin of the RTC will trigger the time stamp after detecting a valid intrusion event Features. The time stamp register will record seconds, minutes, hours, days, and months calendar time. A group of time stamp registers correspond to two intrusion event input pins respectively. When an intrusion event occurs on the pin for the first time, the intrusion flag is set and the time stamp is recorded. When the intrusion event occurs again, the overflow flag is set, but the time stamp register still retains the first recorded value. After the intrusion flag register is cleared, the time stamp will not be recorded again until intrusion detection occurs.

27.4 Interrupt Description

RTC supports 3 interrupt types. Timing alarm interrupt, periodic interrupt and intrusion detection interrupt.

27.4.1 Alarm interrupt

The alarm interrupt RTC_ALM, when ALMIE=1 of control register 2 (RTC_CR2) and ALME=1 of control register 2 (RTC_CR2), if the current calendar time and minute alarm register (RTC_ALMMIN), hour alarm register (RTC_ALMHOUR), weekly alarm register (RTC_ALMWEEK) is equal to trigger the alarm interrupt. Alarm configuration independent flag register bit RTC_CR2.ALMF, write "0" to RTC_CR2.ALMF bit to clear the alarm flag.

27.4.2 Periodic interrupt

Periodic interrupt RTC_PRD, when PRDIE=1 in control register 2 (RTC_CR2), after the selected period occurs, the periodic wake-up interrupt is triggered. The periodic interrupt configures the independent flag register RTC_CR2.PRDF, and the periodic flag can be cleared by writing "0" to the RTC_CR2.PRDF bit.

27.4.3 Intrusion detection interrupt

When intrusion detection is valid RTC_TPCRn.TPEN(n=0,1), and RTC.TPCRn.TPIE(n=0,1) intrusion event interrupt is enabled, after the external pin of RTC detects a valid intrusion event, it will occur Intrusion event interrupt. Since the two groups of intrusion detections share one interrupt, they can be distinguished by the intrusion event detection interrupt flag bit RTC_TPSR.TPFn (n=0,1).

27.5 Register description

Table 27-2 shown is the register list of the RTC module.

Register base address: 0x4004C000

Table 27-2 Register list

Register name	Symbol	Offset	Bit width	Reset value
Control register 0	RTC_CR0	0x0000h	8	Indefinite
Control register 1	RTC_CR1	0x0004h	8	Indefinite
Control register 2	RTC_CR2	0x0008h	8	Indefinite
Control register 3	RTC_CR3	0x000Ch	8	Indefinite
Second count register	RTC_SEC	0x0010h	8	Indefinite
Minute Count Register	RTC_MIN	0x0014h	8	Indefinite
Time count register	RTC_HOUR	0x0018h	8	Indefinite
Week count register	RTC_WEEK	0x001Ch	8	Indefinite
Day count register	RTC_DAY	0x0020h	8	Indefinite
Month count register	RTC_MON	0x0024h	8	Indefinite
Year count register	RTC_YEAR	0x0028h	8	Indefinite
Sub-alarm register	RTC_ALMMIN	0x002Ch	8	Indefinite
Time alarm register	RTC_ALMHOUR	0x0030h	8	Indefinite
Weekly Alarm Register	RTC_ALM WEEK	0x0034h	8	Indefinite
Clock Error Compensation Register	RTC_ERRCRH	0x0038h	8	Indefinite
Clock Error Compensation Register	RTC_ERRCRL	0x003Ch	8	Indefinite
Intrusion Control Register 0	RTC_TPCR0	0x0040h	8	Indefinite
Intrusion Control Register 1	RTC_TPCR1	0x0044h	8	Indefinite
Intrusion Status Register	RTC_TPSR	0x0048h	8	Indefinite
Second Timestamp Register	RTC_SECTP	0x004Ch	8	Indefinite
Sub-timestamp register	RTC_MINNTP	0x0050h	8	Indefinite
Time stamp register	RTC_HOURTP	0x0054h	8	Indefinite
Day time stamp register	RTC_DAYTP	0x0058h	8	Indefinite
Month Timestamp Register	RTC_MONTP	0x005Ch	8	Indefinite

27.5.1 Control Register 0 (RTC_CR0)

Reset value: Indefinite

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															
RESET															
Bit	Marking	Place name	Function	Read and write											
b31~b1	Reserved	-	Read as "0", write as "0"	R/W											
b0	RESET	RTC calendar counter reset	write status 0: The initialization register is invalid 1: The initialization register is valid Initialize all RTC registers. read status 0: Normal counting state or end of RTC software reset 1: RTC is in reset state	R/W											

27.5.2 Control Register 1 (RTC_CR1)

Reset value: Indefinite

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								START	ONEHZSEL	ONEHZOE	-	AMPM	PRDS[2:0] PRDS[1] PRDS[0]																																		
Bit	Marking	Place name		Function										Read and write																																	
b31~b8	Reserved	-		Read as "0", write as "0"										R/W																																	
b7	START	RTC count starts		0: RTC counting stops 1: RTC counting starts										R/W																																	
b6	ONEHZSEL	1Hz output selection		0: Distributed compensation 1Hz output 1: Uniform compensation 1Hz output <small>Note: When RTC_ERRCRH.COMPEN=1, the setting of this bit is valid.</small>										R/W																																	
b5	ONEHZOE	1Hz output license		0: 1Hz output disabled 1: 1Hz output permission										R/W																																	
b4	Reserved	-		Read as "0", write as "0"										R/W																																	
b3	AMPM	Timing selection		0: 12-hour clock 1: 24-hour clock										R/W																																	
Cycle selection settings:																																															
<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <th>PRDS[2]</th><th>PRDS[1]</th><th>PRDS[0]</th><th>Cycle selection</th></tr> <tr> <td>0</td><td>0</td><td>0</td><td>do not choose</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>every 1 second period</td></tr> <tr> <td>0</td><td>1</td><td>1</td><td>every 1 minute cycle</td></tr> <tr> <td>1</td><td>0</td><td>0</td><td>every 1 hour period</td></tr> <tr> <td>1</td><td>0</td><td>1</td><td>Every 1 day cycle (00:00:00 every day)</td></tr> <tr> <td>1</td><td>1</td><td>X</td><td>Every January cycle (00:00:00 on the 1st of each month)</td></tr> </table>																PRDS[2]	PRDS[1]	PRDS[0]	Cycle selection	0	0	0	do not choose	0	0	1	every 0.5 second cycle	0	1	0	every 1 second period	0	1	1	every 1 minute cycle	1	0	0	every 1 hour period	1	0	1	Every 1 day cycle (00:00:00 every day)	1	1	X	Every January cycle (00:00:00 on the 1st of each month)
PRDS[2]	PRDS[1]	PRDS[0]	Cycle selection																																												
0	0	0	do not choose																																												
0	0	1	every 0.5 second cycle																																												
0	1	0	every 1 second period																																												
0	1	1	every 1 minute cycle																																												
1	0	0	every 1 hour period																																												
1	0	1	Every 1 day cycle (00:00:00 every day)																																												
1	1	X	Every January cycle (00:00:00 on the 1st of each month)																																												
<small>Note: When writing cycle selection during START=1 counting, turn off cycle interrupt permission to prevent malfunction. And the relevant flag bits should be cleared after writing.</small>																																															

27.5.3 Control Register 2 (RTC_CR2)

Reset value: Indefinite

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	PRDF	RWEN	RWREQ	
Bit	Marking	Place name	Function													Read and write
b31~b8	Reserved	-	Read as "0", write as "0"													R/W
b7	ALME	Alarm function enabled	0: The alarm function is disabled 1: Alarm function license													R/W
b6	ALMIE	Alarm interrupt enable	0: Alarm interrupt disabled 1: Alarm interrupt permission													R/W
b5	PRDIE	Cycle interrupt enable	0: Periodic interrupt disabled 1: Periodic interrupt permission													R/W
b4	Reserved	-	Read as "0", write as "0"													R/W
b3	ALMF	alarm clock sign	0: The alarm clock does not match 1: Alarm matching Note: Valid when ALME=1, when the alarm clock matches, a count clock is set to "1". Write "0" to clear the flag, write "1" invalid.													R/W
b2	PRDF	cycle sign	0: cycle does not occur 1: Cycle occurs Note: After the set period occurs, this bit is set to "1". Write "0" to clear the flag, write "1" invalid													R/W
b1	RWEN	read/write enable	0: read/write disabled 1: read/write enabled Note: Calendar register read and write enable flag. Please make sure this bit is "1" before reading/writing. Calendar registers include second, minute, hour, week, day, month, and year count registers.													R/W
b0	RWREQ	read/write request	0: Normal counting mode 1: read/write request Note: When reading/writing the calendar register, please set this bit to "1" and request to read and write. Since the counter is counting continuously, please complete the read/write operation within 1 second and clear this bit to "0".													R/W

27.5.4 Control Register 3 (RTC_CR3)

Reset value: Indefinite

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	-	-	-	-
Bit	Marking	Place name	Function												Read and write
b31~b8	Reserved	-	Read as "0", write as "0"												R/W
b7	RCKSEL	RTC count clock selection	0: select external low-speed oscillator XTAL32 clock 1: Select the internal low-speed oscillator RTCLRC clock												R/W
b6	Reserved	-	Read as "0", write as "0"												R/W
b5	Reserved	-	Read as "0", write as "0"												R/W
b4	LCREN	Internal low speed oscillator enable	0: The internal low-speed oscillator RTCLRC is stopped 1: The internal low-speed oscillator RTCLRC works Note: When the low-speed oscillator is used as the RTC clock source, please set the LCREN bit to enable.												R/W
b3~b1	Reserved	-	Read as "0", write as "0"												R/W
b0	Reserved	-	Read as "0", write as "0"												R/W

27.5.5 Second Count Register (RTC_SEC)

Reset value: Indefinite

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]						
Bit	Marking	Place name	Function												Read and write
b31~b7	Reserved	-	Read as "0", write as "0"												R/W
b6~b4	SECD[2:0]	tens of seconds	Second tens count value												R/W
b3~b0	SECU[3:0]	seconds digit	Seconds digit count value												R/W

Represents 0-59 seconds in decimal count. Please write the decimal 0-59 BCD code, when writing the wrong value, the written value will be ignored.

27.5.6 Minute Count Register (RTC_MIN)

Reset value: Indefinite

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]			

Bit	Marking	Place name	Function	Read and write
b31~b7	Reserved	-	Read as "0", write as "0"	R/W
b6~b4	MIND[2:0]	tenths	10's digit count value	R/W
b3~b0	MINU[3:0]	one-digit	Minutes digit count value	R/W

Represents 0-59 points in decimal notation. Please write the decimal 0-59 BCD code, when writing the wrong value, the written value will be ignored.

27.5.7 Hour count register (RTC_HOUR)

Reset value: Indefinite

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]			

Bit	Marking	Place name	Function	Read and write
b31~b6	Reserved	-	Read as "0", write as "0"	R/W
b5~b4	HOURD[1:0]	tenth	10-digit count value	R/W
b3~b0	HOURU[3:0]	time unit	Hourly one-digit count value	R/W

In 24-hour time format, it means 0-23 hours. In the 12-hour time system, b5=0 means AM, then 01~12 means morning; b5=1 means PM, then 21~32 means afternoon.

Please set the correct decimal 0~23 or 01~12, 21~32 BCD code according to the value of the control bit AMPM. Values written out of range are ignored.

Refer to the table below for specific time:

24 hour clock	AMPM=1	12 hour clock	AMPM=0
Time	Register representation	Time	Register representation
00 hours	*	AM 12:00	12H
01:00	01H	AM 01	01H
02:00	02H	AM 02	02H
03:00	03H	AM 03	03H
04:00	04H	AM 04	04H
05:00	05H	AM 05	05H
06:00	06H	AM 06	06H
07:00	0000h	AM 07	0000h
08:00	0000h	AM 08	0000h
09:00	0000h	AM 09	09H
10 o'clock	10H	10 am	0000h
11:00	11H	11 am	0000h
12:00	12H	PM 12:00	32H
13:00	13H	PM 01	21H
14:00	14H	PM 02 hours	22H
15:00	15H	PM 03 hours	23H
16:00	16H	PM 04 hours	24H
17:00	17H	PM 05 hours	25H
18:00	0000h	PM 06 hours	26H
19:00	19H	PM 07	27H
20:00	20H	PM 08	28H
21:00	0000h	PM 09:00	29H
22 hours	0000h	10:00 PM	30H
23 hours	0000h	11:00 PM	31H

27.5.8 Day Count Register (RTC_DAY)

Reset value: Indefinite

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]				

Bit	Marking	Place name	Function	Read and write
b31~b6	Reserved	-	Read as "0", write as "0"	R/W
b5~b4	HOURD[1:0]	day ten	Day 10 count value	R/W
b3~b0	HOURU[3:0]	day unit	day count value	R/W

The decimal system represents days 1 to 31, and leap years and months are automatically calculated. The specific representation is as follows:

Month	Day count indication
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

27.5.9 Week Count Register (RTC_WEEK)

Reset value: Indefinite

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]	

Bit	Marking	Place name	Function	Read and write
b31~b3	Reserved	-	Read as "0", write as "0"	R/W
b2~b0	WEEK[2:0]	week	Week count value	R/W

Decimal 0~6 means Sunday~Saturday. Please write the correct decimal 0~6 BCD code, other values will be ignored. The corresponding relationship between the weekly count values is as follows:

Week	Week count representation
Sunday	00H
on Monday	01H
Tuesday	02H
Wednesday	03H
Thursday	04H
Friday	05H
Saturday	06H

27.5.10 Month Count Register (RTC_MON)

Reset value: Indefinite

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]					

Bit	Marking	Place name	Function	Read and write
b31~b5	Reserved	-	Read as "0", write as "0"	R/W
b4~b0	MON[4:0]	moon	Month count value	R/W

Decimal 1~12 means 1~12 months. Please write the correct decimal 1~12 BCD code, other values will be ignored.

27.5.11 Year Count Register (RTC_YEAR)

Reset value: Indefinite

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]		

Bit	Marking	Place name	Function	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
b7~b4	YEARD[3:0]	Year ten	Year ten-digit count value	R/W
b3~b0	YEARU[3:0]	year digit	Year one-digit count value	R/W

Decimal 0~99 means 0~99 years. Counting based on month rounds. Automatic calculation of leap years such as: 00, 04, 08, ..., 92, 96, etc. Please write the correct decimal year count value, writing an incorrect value will be ignored.

27.5.12 Sub-alarm register (RTC_ALMMIN)

Reset value: Indefinite

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]		ALMINU[3:0]			

Bit	Marking	Place name	Function	Read and write
b31~b7	Reserved	-	Read as "0", write as "0"	R/W
b6~b4	ALMMIND[3:0]	Alarm clock for ten	The ten-digit matching value of the sub-alarm clock	R/W
b3~b0	ALMINU[3:0]	Divide the alarm clock	Sub-alarm unit digit match value	R/W

Please set the BCD code of decimal 0~59. Write other values and no alarm match will occur.

27.5.13 Time alarm register (RTC_ALMHOUR)

Reset value: Indefinite

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]			

Bit	Marking	Place name	Function	Read and write
b31~b6	Reserved	-	Read as "0", write as "0"	R/W
b5~b4	ALMHOURD[1:0]	time alarm clock	The ten-digit matching value of the alarm clock	R/W
b3~b0	ALMHOURU[3:0]	time alarm clock	Hourly alarm one digit match value	R/W

Please set the correct alarm matching value according to the time system, otherwise the time alarm matching will not occur.

27.5.14 Weekly Alarm Register (RTC_ALMWEEK)

Reset value: Indefinite

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]					

Bit	Marking	Place name	Function	Read and write
b31~b7	Reserved	-	Read as "0", write as "0"	R/W
b6~b0	ALMWEEK[6:0]	weekly alarm clock	Weekly alarm matching value. b0~b6 correspond to Sunday~Saturday respectively, and when it is set to "1", it means that the alarm clock is valid on that day of the week. For example, b0=1, b5=1 means that the alarm clock setting 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.

27.5.15 Clock Error Compensation Registers (RTC_ERRCRH, RTC_ERRCRL)

Reset value: Indefinite

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]

Reset value: Indefinite

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

Bit	Marking	Place name	Function	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
b7	COMPEN	Compensation enable	0: Clock error compensation is disabled 1: Clock error compensation is enabled	R/W
B6~b1	Reserved	-	Read as "0", write as "0"	R/W
b0	COMP[8]	Compensation value	Set the compensation value together with COMP[7:0]	R/W

RTC_ERRCRL

Bit	Marking	Place name	Function	Read and write																																				
b31~b8	Reserved	-	Read as "0", write as "0"	R/W																																				
b7~b0	COMP[7:0]	Compensation value	By setting the compensation value, an accuracy compensation of +/-0.96ppm per second can be performed. The compensation value is a 9-bit 2's complement with a decimal point, and the last 5 digits are the fractional part. Compensable range -275.5ppm~+212.9ppm. Minimum resolution 0.96ppm. Please refer to the following table for specific compensation accuracy: <table border="1" style="margin-left: 20px;"> <thead> <tr> <th colspan="2">Compensation value setting</th><th>Compensation number</th></tr> <tr> <th>COMPEN</th><th>COMP[8:0]</th><th></th></tr> </thead> <tbody> <tr> <td rowspan="10" style="vertical-align: middle; text-align: center;">1</td><td>1</td><td>0 0 0 0 0 0 0 0 0 -275.5ppm</td></tr> <tr><td>1</td><td>0 0 0 0 0 0 0 0 1 -274.6ppm</td></tr> <tr><td>~</td><td>~ ~ ~ ~ ~ ~ ~ ~</td></tr> <tr><td>0</td><td>0 0 0 0 0 0 0 0 0 -30.5ppm</td></tr> <tr><td>~</td><td>~ ~ ~ ~ ~ ~ ~ ~</td></tr> <tr><td>0</td><td>0 0 0 0 1 1 1 1 1 -0.96ppm</td></tr> <tr><td>0</td><td>0 0 0 1 0 0 0 0 0 0 0 0 Oppm</td></tr> <tr><td>0</td><td>0 0 0 1 0 0 0 0 0 0 1 +0.96ppm</td></tr> <tr><td>~</td><td>~ ~ ~ ~ ~ ~ ~ ~</td></tr> <tr><td>0</td><td>0 0 1 0 0 0 0 0 0 0 0 0 +30.5ppm</td></tr> <tr> <td rowspan="5" style="vertical-align: middle; text-align: center;">0</td><td>~</td><td>~ ~ ~ ~ ~ ~ ~ ~</td></tr> <tr><td>0</td><td>1 1 1 1 1 1 1 1 0 +212.0ppm</td></tr> <tr><td>0</td><td>1 1 1 1 1 1 1 1 1 +212.9ppm</td></tr> <tr><td>X</td><td>X X X X X X X X invalid</td></tr> </tbody> </table>	Compensation value setting		Compensation number	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 0 1 -274.6ppm	~	~ ~ ~ ~ ~ ~ ~ ~	0	0 0 0 0 0 0 0 0 0 -30.5ppm	~	~ ~ ~ ~ ~ ~ ~ ~	0	0 0 0 0 1 1 1 1 1 -0.96ppm	0	0 0 0 1 0 0 0 0 0 0 0 0 Oppm	0	0 0 0 1 0 0 0 0 0 0 1 +0.96ppm	~	~ ~ ~ ~ ~ ~ ~ ~	0	0 0 1 0 0 0 0 0 0 0 0 0 +30.5ppm	0	~	~ ~ ~ ~ ~ ~ ~ ~	0	1 1 1 1 1 1 1 1 0 +212.0ppm	0	1 1 1 1 1 1 1 1 1 +212.9ppm	X	X X X X X X X X invalid	R/W
Compensation value setting		Compensation number																																						
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 0 1 -274.6ppm																																						
	~	~ ~ ~ ~ ~ ~ ~ ~																																						
	0	0 0 0 0 0 0 0 0 0 -30.5ppm																																						
	~	~ ~ ~ ~ ~ ~ ~ ~																																						
	0	0 0 0 0 1 1 1 1 1 -0.96ppm																																						
	0	0 0 0 1 0 0 0 0 0 0 0 0 Oppm																																						
	0	0 0 0 1 0 0 0 0 0 0 1 +0.96ppm																																						
	~	~ ~ ~ ~ ~ ~ ~ ~																																						
	0	0 0 1 0 0 0 0 0 0 0 0 0 +30.5ppm																																						
0	~	~ ~ ~ ~ ~ ~ ~ ~																																						
	0	1 1 1 1 1 1 1 1 0 +212.0ppm																																						
	0	1 1 1 1 1 1 1 1 1 +212.9ppm																																						
	X	X X X X X X X X invalid																																						

Compensation calculation description:

When the 1Hz clock is directly output in the default state, the compensation target value is calculated by measuring the accuracy of the clock.

Assuming the actual measured value is 0.9999888Hz, then:

$$\text{Actual vibration frequency} = 32768 \times 0.9999888 \approx 32767.63$$

Compensation target value = (actual vibration frequency - target frequency)/target frequency $\times 10^6$

$$\begin{aligned} &= (32767.96 - 32768) / 32768 \times 10^6 \\ &= -11.29 \text{ ppm} \end{aligned}$$

Set value calculation:

$$\text{COMP[8:0]} = \left(\frac{\text{Compensation target value[ppm]} \times 2^{15}}{10^6} \right)_{\text{Pick2complement}} + 0001.00000B$$

If the compensation target value is +20.3ppm, the corresponding register value is calculated as follows:

$$\begin{aligned} \text{COMP[8:0]} &= (20.3 \times 2^{15} / 10^6)_{\text{2's complement}} + 0001.00000B \\ &= (0.6651904)_{\text{2's complement}} + 0001.00000B \\ &= 0000.10101B + 0001.00000B \\ &= 0001.10101B \end{aligned}$$

If the compensation target value is -20.3ppm, the corresponding register value is calculated as follows:

$$\begin{aligned} \text{COMP[8:0]} &= (-20.3 \times 2^{15} / 10^6)_{\text{2's complement}} + 0001.00000B \\ &= (-0.6651904)_{\text{2's complement}} + 0001.00000B \\ &= 1111.01011B + 0001.00000B \\ &= 0000.01011B \end{aligned}$$

27.5.16 Intrusion Control Register 0 (RTC_TPCR0)

Reset value: Indefinite

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								TPENO	TSTPE0	TPIE0	TPRSTE0	TPNFO[1:0]	TPCT0[1:0]				
<hr/>																	
Bit	Marking	Place name	Function										Read and write				
b31~b8	Reserved	-	Read as "0", write as "0"										R/W				
b7	TPENO	Intrusion Detection Enable	0: The intrusion function on the RTCIC0 input pin is invalid 1: The intrusion function on the RTCIC0 input pin is valid										R/W				
b6	TSTPE0	Timestamp enabled	0: Time stamp function disabled 1: Timestamp function enabled										R/W				
b5	TPIE0	Intrusion Event Interrupt Enable	0: Intrusion event interrupt disabled 1: Intrusion event interrupt permission										R/W				
b4	TPRSTE0	Intrusion Event Reset Enable	0: Backup register reset disabled 1: Backup register reset permission										R/W				
b3~b2	TPNFO[1:0]	Filtering function	0X: Filter function is invalid 10: The RTCIC0 input pin detection performs 3 times of consistency detection and filtering according to the timing clock 11: The RTCIC0 input pin detection performs 3 times of consistency detection filtering according to the frequency division of the timing clock by 32										R/W				
b1~b0	TPCT0[1:0]	Active edge selection	00: RTCIC0 input pin is not detected 01: Valid when RTCIC0 input pin detects a rising edge 10: Valid when the RTCIC0 input pin detects a falling edge 11: Valid when RTCIC0 input pin detects rising edge or falling edge										R/W				

27.5.17 Intrusion Control Register 1 (RTC_TPCR1)

Reset value: Indefinite

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								TPEN1	TSTPE1	TPIE1	TPRSTE1	TPNF1[1:0]	TPCT1[1:0]				
Bit	Marking	Place name	Function										Read and write				
b31~b8	Reserved	-	Read as "0", write as "0"										R/W				
b7	TPEN1	Intrusion Detection Enable	0: The intrusion function on the RTCIC1 input pin is invalid 1: The intrusion function on the RTCIC1 input pin is valid										R/W				
b6	TSTPE1	Timestamp enabled	0: Time stamp function disabled 1: Timestamp function enabled										R/W				
b5	TPIE1	Intrusion Event Interrupt Enable	0: Intrusion event interrupt disabled 1: Intrusion event interrupt permission										R/W				
b4	TPRSTE1	Intrusion Event Reset Enable	0: Backup register reset disabled 1: Backup register reset permission										R/W				
b3~b2	TPNF1[1:0]	Filtering function	0X: Filter function is invalid 10: The RTCIC0 input pin detection performs 3 times of consistency detection and filtering according to the timing clock 11: The RTCIC0 input pin detection performs 3 times of consistency detection filtering according to the frequency division of the timing clock by 32										R/W				
b1~b0	TPCT1[1:0]	Active edge selection	00: RTCIC1 input pin is not detected 01: Valid when RTCIC1 input pin detects a rising edge 10: Valid when the RTCIC1 input pin detects a falling edge 11: Valid when RTCIC1 input pin detects rising or falling edge										R/W				

27.5.18 Intrusion Status Register (RTC_TPSR)

Reset value: Indefinite

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													TPOVF	TPF1	TPFO
Bit	Marking	Place name	Function											Read and write	
b31~b3	Reserved	-	Read as "0", write as "0"											R/W	
b2	TPOVF	Intrusion status overflow flag	0: Reset state or no tamper state overflow occurred 1: Intrusion overflow occurred <i>Note: The flag bit is cleared directly after writing "0".</i>											R/W	
b1	TPF1	Intrusion Status Flag 1	0: No intrusion event has occurred on the RTCIC1 input pin 1: Intrusion event on RTCIC1 input pin <i>Note: To prevent misoperation, this bit must be cleared when TPCT1[1:0]=00b and the intrusion detection edge is invalid. Please confirm that the status of this flag is 0 before setting the intrusion function to be valid.</i> <i>The flag bit is cleared directly after writing "0".</i>											R/W	
b0	TPFO	Intrusion status flag 0	0: No intrusion event has occurred on the RTCIC0 input pin 1: Intrusion event on RTCIC0 input pin <i>Note: To prevent misoperation, this bit must be cleared when TPCT0[1:0]=00b and the intrusion detection edge is invalid. Please confirm that the status of this flag is 0 before setting the intrusion function to be valid.</i> <i>The flag bit is cleared directly after writing "0".</i>											R/W	

27.5.19 Second Timestamp Register (RTC_SECTP)

Reset value: Indefinite

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								SECTPD[2:0]				SECTPU[3:0]							
Bit	Marking	Place name	Function											Read and write					
b31~b7	Reserved	-	Read as "0", write as "0"											R					
b6~b4	SECTPD[2:0]	tens of seconds	Second timestamp ten-digit count value											R					
b3~b0	SECTPU[3:0]	seconds digit	Second timestamp units count value											R					

27.5.20 Minute Timestamp Register (RTC_MINTP)

Reset value: Indefinite

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										MINTPD[2:0]		MINTPU[3:0]				
<hr/>																
Bit	Marking		Place name			Function										Read and write
b31~b7	Reserved		-			Read as "0", write as "0"										R
b6~b4	MINTPD[2:0]		tenths			Sub-timestamp ten-digit count value										R
b3~b0	MINTPU[3:0]		one-digit			Sub-timestamp one-digit count value										R

27.5.21 Time stamp register (RTC_HOURTP)

Reset value: Indefinite

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										HOURTPD[1:0]		HOURTPU[3:0]				
<hr/>																
Bit	Marking		Place name			Function										Read and write
b31~b6	Reserved		-			Read as "0", write as "0"										R
b5~b4	HOURTPD[1:0]		tenth			Time stamp ten-digit count value										R
b3~b0	HOURTPU[3:0]		time unit			Timestamp units count value										R

27.5.22 Day Timestamp Register (RTC_DAYTP)

Reset value: Indefinite

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										DAYTPD[1:0]	DAYTPU[3:0]				
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b6	Reserved	-	Read as "0", write as "0"	R											
b5~b4	HOURTPD[1:0]	day ten	Ten-digit count value of day and time stamp	R											
b3~b0	HOURTPU[3:0]	day unit	Day time stamp single digit count value	R											

27.5.23 Month Timestamp Register (RTC_MONTP)

Reset value: Indefinite

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										MONTP[4:0]					
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b5	Reserved	-	Read as "0", write as "0"	R											
b4~b0	MONTP[4:0]	moon	Month timestamp count value	R											

28 Watchdog Timer (WDT/SWDT)

28.1 Introduction

There are two Watchdog Timer. One is the dedicated Watchdog Timer (SWDT) whose counting clock source is dedicated internal RC (SWDTLRC: 10KHz), and the other is the general Watchdog Timer (WDT) whose counting clock source is PCLK3 . Dedicated Watchdog and general watchdog are 16-bit down counters used to monitor software faults caused by external disturbances or unforeseen logic conditions that cause the application to deviate from normal operation.

Both watchdogs support the window function. The window interval can be preset before the counting. When the count value is in the window interval, the counter can be refreshed and the counting restarts. The basic characteristics are shown as Table 28-1.

Table 28-1 Basic Characteristics of Watchdog Timer

Counting clock	SWDT: 1/16/32/64/128/256/2048 frequency division of SWDTLRC WDT: 4/64/128/256/512/1024/2048/8192 frequency division of PCLK3
Maximum overflow time	SWDT: 3.72hour WDT: 10.7s (PCLK3=50MHz)
Counting mode	Decrement count
Window function	Can set window interval, define the allowable interval of refresh action
Start-up mode	1) Hardware startup 2) Software startup
Stop conditions	1) Resetting 2) Underflow, refresh error occurs Start again: In the hardware startup mode, it starts automatically after the reset or interrupt request output In the software startup mode, set the refresh register again
Interrupt/Reset Conditions	1) Count underflow 2) Refresh error

28.2 Function description

28.2.1 Start the Watchdog

There are two ways to start the Watchdog timer: Hardware startup mode and software startup mode.

The hardware startup mode refers to reading the Watchdog timer setting information (ICG0 Register) from the main flash memory area during startup, and the counter starts counting automatically. Software startup mode means that after setting the control register, write the refresh register to completes the refresh action, and the counter starts counting.

28.2.2 Hardware startup mode

When bit 16 (WDTAUTS) and bit 0 (SWDTAUTS) of the ICG0 register are 0, it is the Hardware startup mode. When the hardware startup mode is selected, the relevant setting information of the WDT_CR and SWDT_CR registers is invalid.

In the hardware startup mode, the WDT/SWDT related settings (counting clock, window setting value, counting cycle, etc.) in the ICG0 register are loaded into the WDT/SWDT module during reset. After reset, the counter starts counting automatically according to the setting. Figure 28-1 shows an example of hardware startup mode.

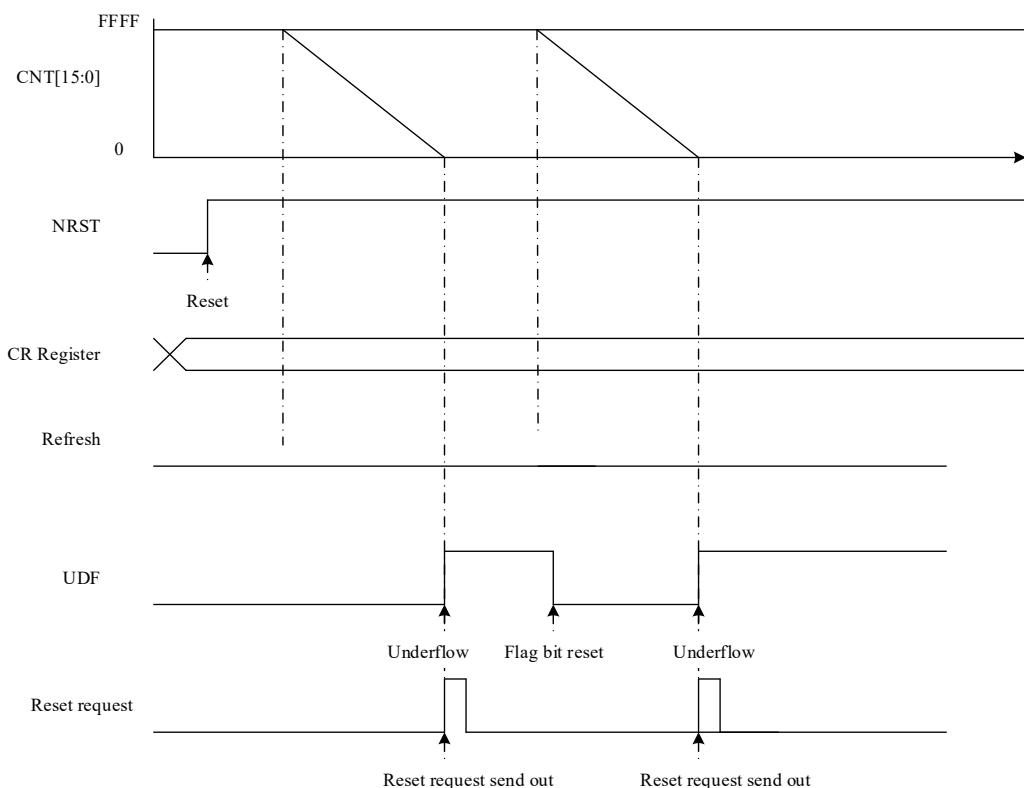


Figure 28-1 Hardware startup example

28.2.3 Software startup mode

When bit 16 (WDTAUTS) and bit 0 (SWDTAUTS) of the ICG0 Register are 1, start WDT/SWDT by setting refresh register as software startup mode. After reset, set the counting clock, window setting value, counting cycle, etc in the WDT_CR/SWDT_CR register, and then execute the refresh action, and the counter starts counting. WDT_CR/SWDT_CR can only be set once, and setting the write value again is invalid. Figure 28-2 shows an example of software startup mode.

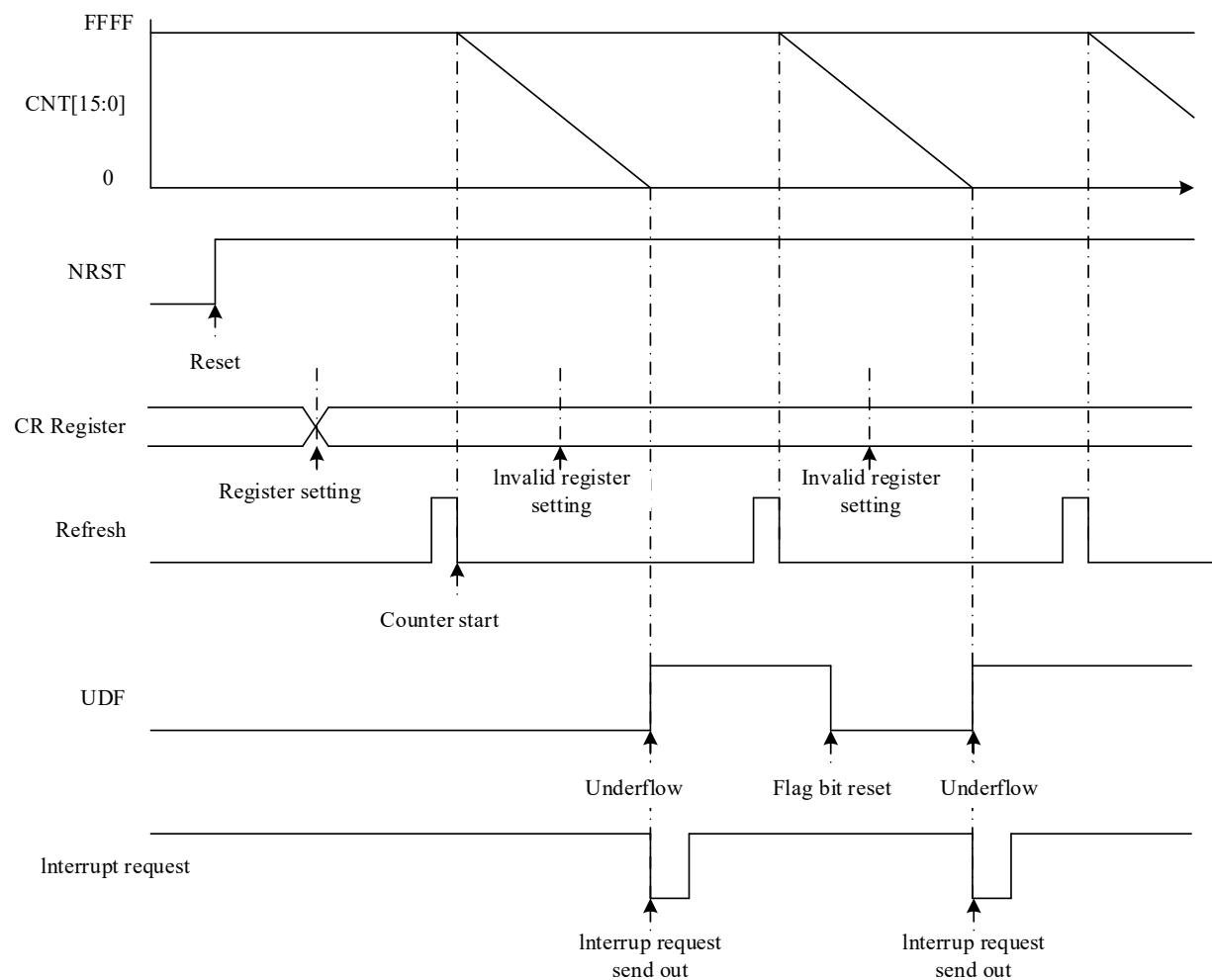


Figure 28-2 Software startup example

28.2.4 Refresh action

Write 0x0123 first, then write 0x3210 in the (S)WDT_RR register to complete a refresh operation, the WDT/SWDT starts counting (software start) or restarts counting.

The (S)WDT_RR register is written between 0x0123 and 0x3210. If reading other registers or reading the (S)WDT_RR register, the normal refresh action will not be affected.

Figure 28-3 shows the examples of various refresh action.

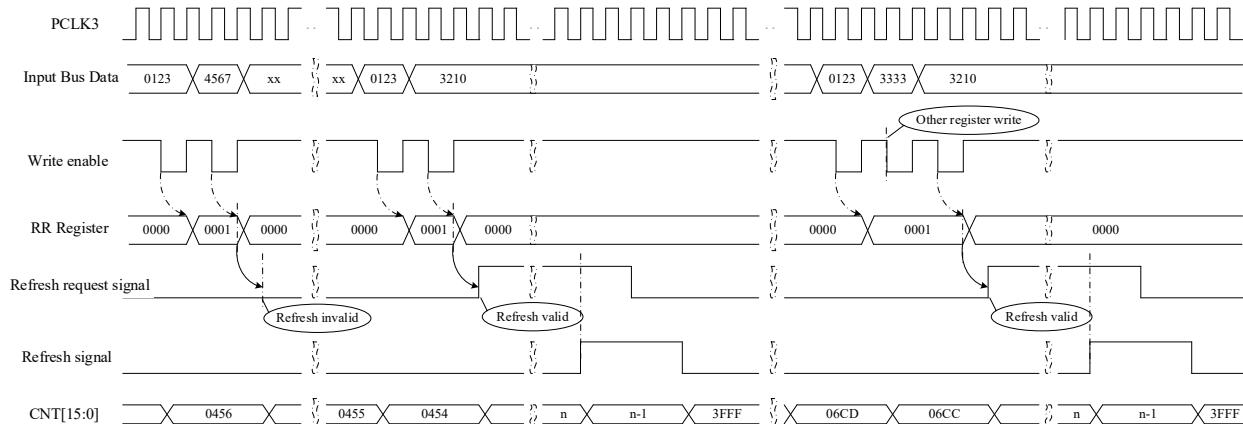


Figure 28-3 Timing examples of various refresh actions (action confirmation, falling edge of refresh request signal, etc.)

The refresh action requires four count cycles to update the count values, so write the refresh register in advance of four count values in the refresh lower window and underflow position. Check the status register for the count value.

28.2.5 Flag bit

The refresh error flag bit and the count underflow flag bit are maintained in the case of an interrupt request. After entering the interrupt, the cause of the interrupt can be confirmed by querying the flag bit. Flag bit reset: Read 1 before writing 0.

When the refresh error or count underflow flag bit is set, the Watchdog timer in hardware startup mode does not stop count; the Watchdog timer in software startup mode stop count. When writing "0" to the flag bit, the SWDT needs at most 3 SWDTLRC and 2 PCLK3 time before the register bit can be cleared; WDT needs at most 5 PCLK3 time before the register bit can be cleared. In addition, within a certain period of time when a refresh error or underflow error occurs, it is invalid to read "1" to the flag bit and clear it. This time is: 1 count cycle + 2 SWDTLRC (SWDT module); 1 count cycle + 2 PCLK3 (WDT module).

28.2.6 Interrupt reset

WDT/SWDT can choose to generate interrupt request or reset request when the counter counts underflow or refresh error. In the hardware startup mode, the interrupt request or the reset request is determined by the WDTITS/SWDTITS bit of the ICG0 register. In the software startup mode, the

interrupt request or the reset request is determined by setting the ITS bit of the WDT_CR/SWDT_CR register.

There are two types of interrupt reset generation conditions for WDT/SWDT. One is that counter counts generates underflow; One is to perform a refresh action outside the refresh allowed interval, resulting in a refresh error.

28.2.7 Count underflow

As shown in Figure 28-4 , the underflow occurs when counting down to zero.

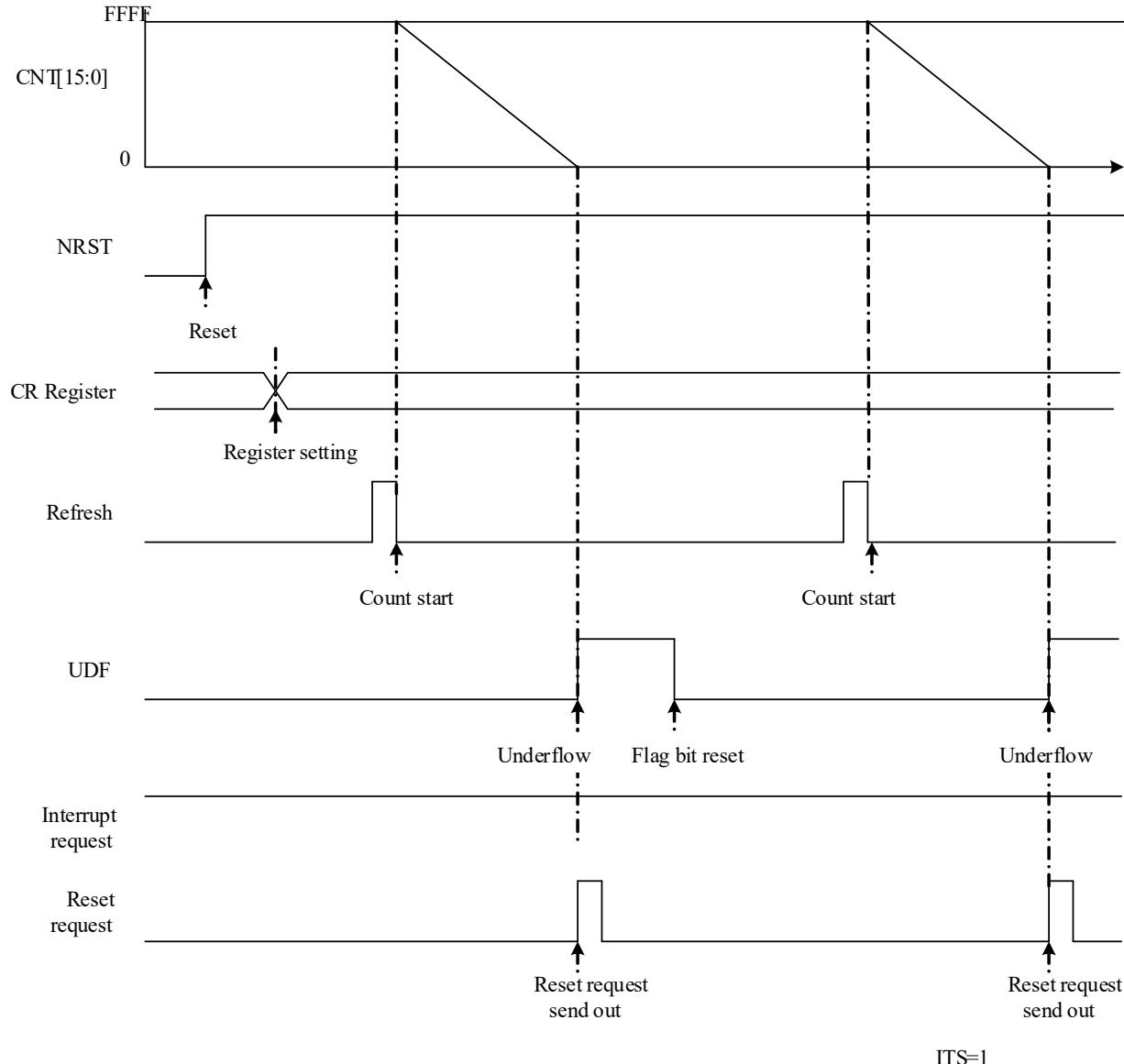


Figure 28-4 Counter underflow action example

28.2.8 Refresh error

After setting the window interval, the counter is refreshed and counted again only when the refresh action is executed in the window interval. A refresh error is generated when the refresh action is executed outside the window interval. Figure 28-5 shows an example of refresh action.

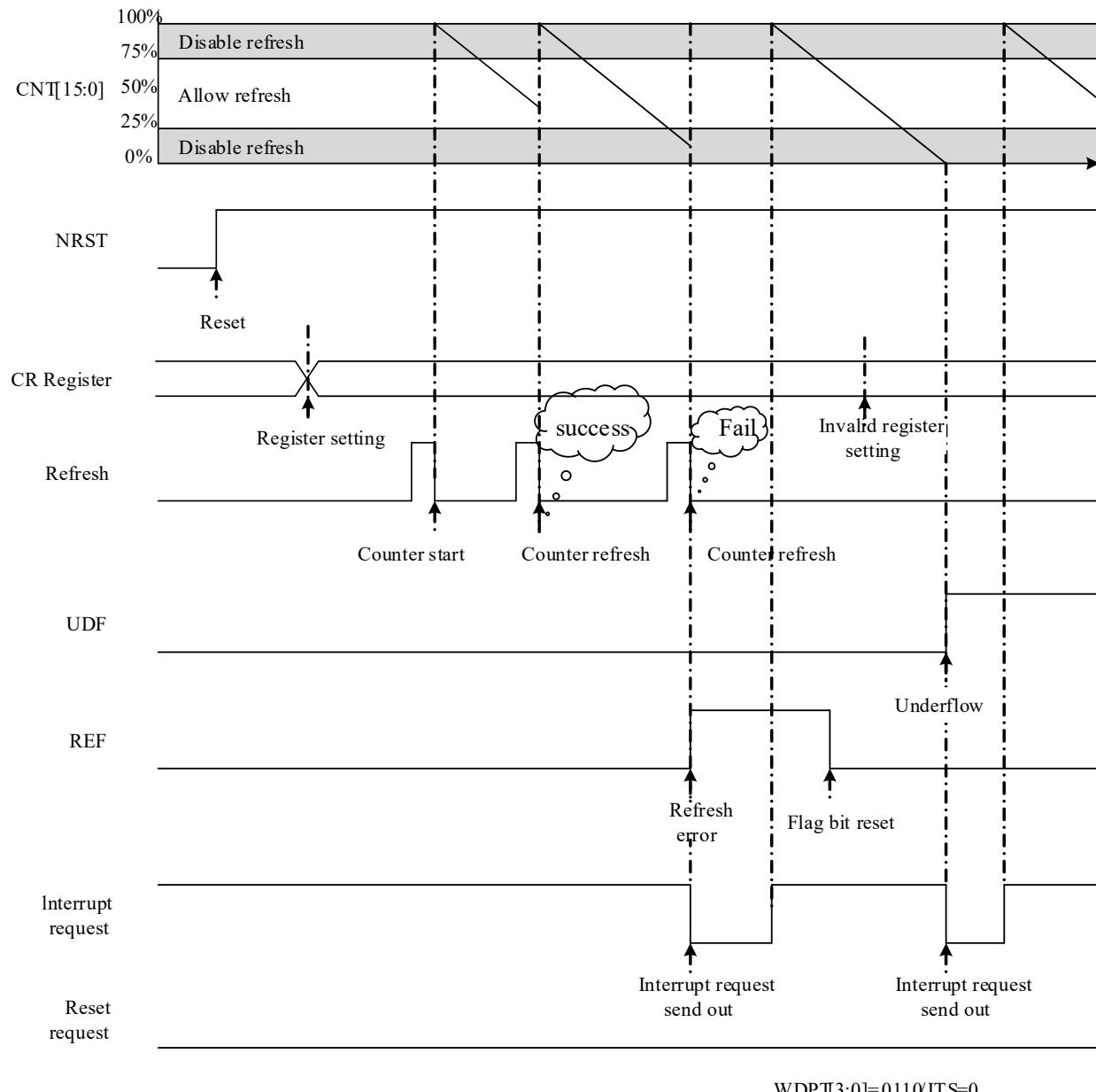


Figure 28-5 Example of counter refresh operation

28.3 Register description

Table 28-2 shown is the register list of the WDT and SWDT modules.

WDT_BASE_ADDR: 0x40049000

SWDT_BASE_ADDR: 0x40049400

Table 28-2 Register list

Register name	Symbol	Offset address	Bit width	Reset value
SWDT Control Register	SWDT_CR	0x00	32	0x8001_0FF3
SWDT State 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 State Register	WDT_SR	0x04	32	0x0000_0000
WDT Refresh Register	WDT_RR	0x08	32	0x0000_0000

28.3.1 Control Register (SWDT_CR, 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				CKS[3:0]				-	-	PERI[1:0]		

Bit	Symbol	Bit name	Description	Read and write
b31	ITS	Refresh Error/Overflow Interrupt/Reset Selection	0: Interrupt request 1: Reset request	R/W
b30~b17	Reserved	-	Read as "0", write as "0"	R/W
b16	SLPOFF	(S)WDT counting disabled in low power mode	WDT_CR: 0: WDT count permission in sleep mode 1: WDT counting is disabled in sleep mode SWDT_CR: 0: SWDT count permission in sleep/stop mode 1: SWDT count disabled in sleep/stop mode	R/W
b15~b12	Reserved	-	Read as "0", write as "0"	R/W
b11~b8	WDPT[3:0]	Percentage of areas allowed to refresh	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	WDT_CR: 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: Reservation SWDT_CR: 0000: SWDTCLK 0100: SWDTCLK/16 0101: SWDTCLK/32 0110: SWDTCLK/64 0111: SWDTCLK/128 1000: SWDTCLK/256 1011: SWDTCLK/2048 Other values: Reservation	R/W
b3~b2	Reserved	-	Read as "0", write as "0"	R/W
b1~b0	PERI[1:0]	Counting cycle	00: 256 cycle 01: 4096 cycle 10: 16384 cycle 11: 65536 cycle	R/W

28.3.2 State Register (SWDT_SR, WDT_SR)

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
CNT[15:0]															

Bit	Symbol	Bit name	Description	Read and write
b31~b18	Reserved	-	Read as "0", write as "0"	R/W
b17	REF	Refresh error flag	0: No refresh error 1: Refresh error occurred After reading 1 and writing 0 to this bit, the bit is cleared.	R/W
b16	UDF	Counting underflow flag	0: No count underflow 1: count underflow occurred After reading 1 and writing 0 to this bit, the bit is cleared.	R/W
b15~b0	CNT[15:0]	Count value	Counter current count	R/W

28.3.3 Refresh Register (SWDT_RR, WDT_RR)

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
RF[15:0]															

Bit	Symbol	Bit name	Description	Read and Write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15~b0	RF[15:0]	Refresh value	Write 0x0123 and 0x3210 in proper order to complete the refresh After the register is written with 0x0123h, the read register is 0x00000001; The value read out in other cases is 0x00000000.	R/W

28.4 Notice

When the SWDT action, the action frequency of the peripheral clock PCLK3 must be greater than or equal to 4 times of the counting clock frequency. the PCLK3 frequency \geq the count clock frequency $\times 4$.

29 Universal Synchronous Asynchronous Receiver/Transmitter (USART)

29.1 Introduction

This product is equipped with 10 units of Universal Synchronous Asynchronous Receiver/Transmitter Module (USART). Universal Synchronous Asynchronous Receiver/Transmitter (USART) can flexibly exchange full-duplex data with external devices; this USART supports Universal Asynchronous Receiver/Transmitter (UART), USART clock Synchronous communication interface, Smartcard interface (ISO/IEC7816-3) and LIN communication interface. Support modem operations (CTS/RTS), multiprocessor communication. Cooperate with Timer0 module to support UART receiver TIMEOUT function. USART_1 supports receiver wake-up STOP mode function via RX line.

The specific function distribution is as follows:

- UART: Full channel support
- Multiprocessor Communication: Full channel support
- Clock synchronization communication: Full channel support
- RX line wake-up STOP mode function: USART_1 support
- Fractional baud rate: USART_1, USART_2, USART_3, USART_4, USART_6, USART_7, USART_8, USART_9 support
- LIN: USART_5, USART_10 support
- Smartcard: USART_1, USART_2, USART_3, USART_4, USART_6, USART_7, USART_8, USART_9 support
- UART receiver timeout function: USART_1, USART_2, USART_6, USART_7 support

Key features of USART:

- Support full-duplex asynchronous communication, full-duplex clock synchronous communication
- Support LIN bus
- Support smartcard interface (ISO/IEC7816-3)
- Transmitter and receiver have independent enable bit
- Built-in double buffer
- LSB/MSB optional
- Modem operation supported (CTS/RTS)
- Transmission flags: Transmit data register empty, Transmit complete, receive data register full, receive error flag, UART receiver timeout flag, LIN wake-up signal detection flag, LIN break field detection flag, LIN bus error flag

Main features of UART:

- Data length programmable: 8 bits/9 bits
- The parity function can be configured: Odd parity/even parity/no parity
- The stop bits are configurable: 1 bit/2 bit
- Optional clock source: Internal clock source (clock generated by internal baud rate generator)/external clock source (clock input by USARTn_CK pin)
- Receive error: parity error, framing error, overrun error
- Support for multiprocessor communications
- Built-in digital filter
- Support receiving data TIMEOUT function
- USART_1 supports stop mode wake-up function
- Support full-duplex/half-duplex communication

Main features of clock synchronization mode:

- Data length: 8 bits
- Receive error: overrun error
- Clock source: Internal clock source (clock generated by internal baud rate generator)/external clock source (clock input by USARTn_CK pin)
- Support full duplex communication

The main features of the smartcard interface:

- Data length: 8 bits
- Automatically transmit an error signal when a checksum error is detected
- Support data retransmission

Key Features of LIN:

- Data length: 8 bits
- Detection of support wakeup signal
- Supports detection of 10/11-bit sync break field (BF)
- Support synchronization field measurement, register record measurement value
- Support 10/11/13/14 bit break field (BF) transmission
- Support bus collision detection
- Support loopback mode

29.2 USART system block diagram

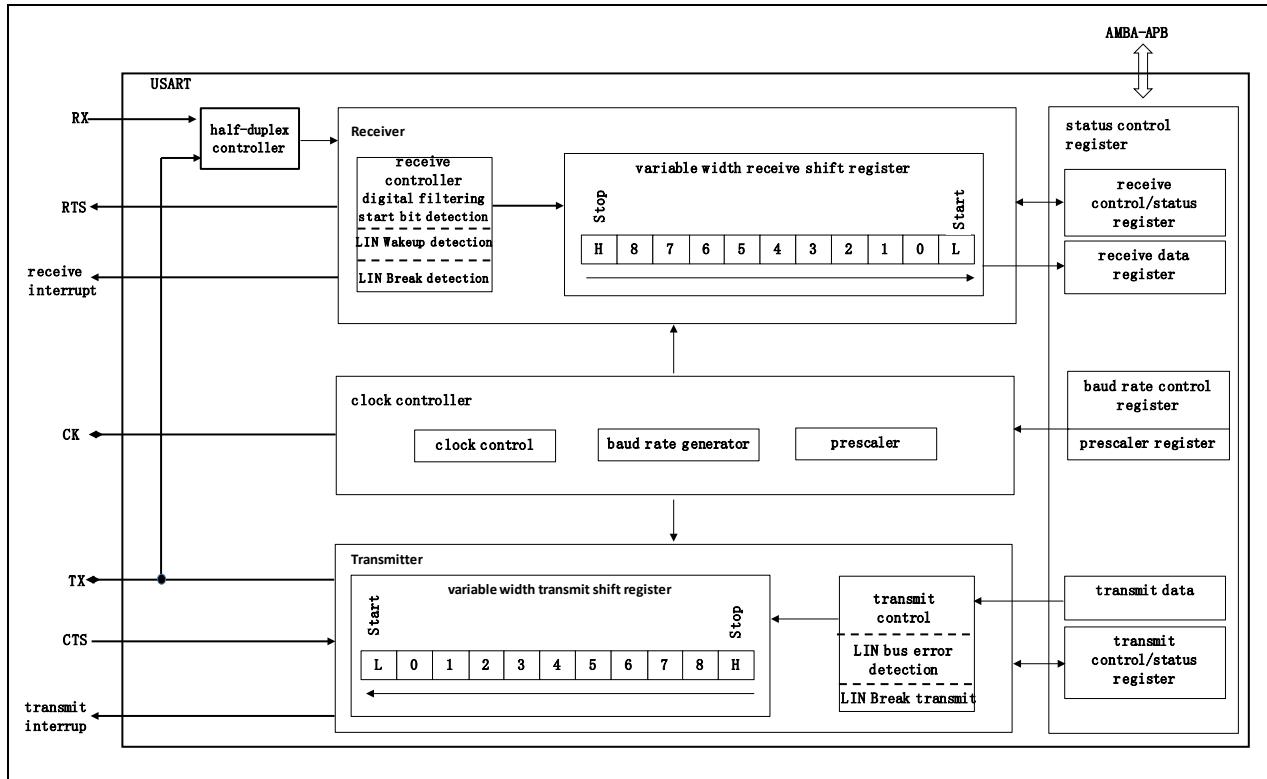


Figure 29-1 USART System Block Diagram

29.3 Pin description

Table 29-1 USART pin description

Pin name	Direction	Functional description
USARTn_CK	Input output	Clock
USARTn_TX	Input output	Transmit data pin Also acts as a receive data pin during half-duplex
USARTn_RX	Input	receive data pin
USARTn_CTS	Input	Modem operation pin clear transmit pin
USARTn_RTS	Output	Modem operation pin Request transmit pin

n: 1~10

29.4 Functional description

This chapter will describe the functions of UART, multiprocessor communication, smartcard, clock synchronization mode and LIN in detail.

29.4.1 UART

29.4.1.1 Clock

UART can select the clock generated by the internal baud rate generator (internal clock source) or the clock entered by the USARTn_CK pin (external clock source) as the clock source for communication.

Internal clock source

When the USARTn_CR2.CLKC[1:0] bits are set to 00b or 01b, the selected clock source is the internal clock source, that is, the clock generated by the internal baud rate generator.

When USARTn_CR2.CLKC[1:0]=00b, the USARTn_CK pin is not used as a clock pin, but can be used as a normal IO.

USARTn_CR2.CLKC [1: 0] = 01b outputs the clock at the same frequency as the communication baud rate from the USARTn_CK pin.

The clock source of the internal baud rate generator is selected as PCLK, PCLK/4, PCLK/16, PCLK/64 by the setting of USARTn_PR.PSC[1:0] bits.

External clock source

USARTn_CR2.CLKC [1: 0] bit is set to 10b or 11b, the clock source is selected as the external clock input from the USARTn_CK pin, and the input clock frequency is 16 times the baud rate (USARTn_CR1.OVER8 = 0) or 8 times (USARTn_CR1.OVER8 = 1).

Maximum baud rate

For internal clock source, the Baud Rate formula generated by the internal Baud Rate Generator is:

$$B = \frac{C}{8 \times (2 - \text{OVER8}) \times (\text{DIV_Integer} + 1)}$$

B: Baud rate unit: MBps

C: The clock set by USARTn_PR.PSC[1:0] bits (PCLK, PCLK/4, PCLK/16, PCLK/64) Unit: MHz

OVER8: USARTn_CR1.OVER8 setting

DIV_Integer: USARTn_BRR.DIV_Integer setting

The maximum baud rate is PCLK/8(MBps).

When the external clock source is used, the maximum frequency of the external input UART clock is required to be PCLK(MHz)/4, so when the clock source is the external input clock, the maximum

baud rate is PCLK/64(MBps) (when USARTn_CR1.OVER8=0) or PCLK /32 (MBps) (when USARTn_CR1.OVER8=1).

It should be noted that, in addition to the calculation method based on PCLK described above, the maximum communication baud rate of the UART also needs to refer to the maximum communication baud rate specified in the electrical characteristics chapter.

29.4.1.2 Data format

A frame of data in UART mode consists of start bits, data bits, parity bits, and stop bits.

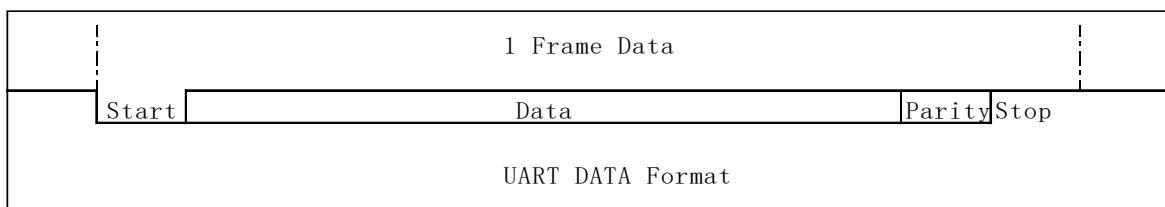


Figure 29-2 UART data format

Start bit

The start bit is fixed with a low level of one bit.

Data bit

Data bits can be configured as 8 bits or 9 bits.

Parity bit

The parity bits can be configured as 1-bit even parity or 1-bit odd parity or no parity bits.

Stop bit

Stops at a fixed high level and can be configured as 1 or 2 bits.

29.4.1.3 Modem operation

Modem operations include CTS and RTS functions. The RTS function is valid when USARTn_CR3.RTSE=1, and the CTS function is valid when USARTn_CR3.CTSE=1.

CTS function

The CTS function is to control the transmission of data through the input of the USARTn_CTS pin. Only when the USARTn_CTS pin is input low level can data be sent. If the USARTn_CTS input is high level during the process of transmitting data, the data being sent will not be affected.

RTS function

The RTS function refers to outputting a low level through the USARTn_RTS pin to request the other party to transmit data.

The USARTn_RTS pin outputting a low level needs to meet all the following conditions:

- Receive enable (USARTn_CR1.RE = 1) and not receiving data
- No unread received data in USARTn_DR.RDR register
- No receive errors, including frame error, parity error and overrun error

29.4.1.4 Transmitter

The transmitter can transmit 8-bit or 9-bit data, depending on the setting of the USARTn_CR1.M bits.

Transmitter enable bit (USARTn_CR1.TE) is set to 1. After writing the transmitted data, the transmitted data is serially output on the TX pin. The corresponding clock pulse can be output or not output at the USARTn_CK pin.

The order in which data is sent is: Start bit -> data bit (MSB/LSB) -> parity bit (with or without) -> stop bit.

The transmission data register USARTn_DR.TDR and the internal transmit shift register constitute a dual buffer structure, which can transmit data continuously.

To ensure the correctness of transmitting data, a request can only be written once by transmitting data register empty interrupt or DMA.

Transmitting data setting procedure

1. Set USARTn_CR1 register to reset
2. Set the pin that UART needs to use
3. Through USARTn_CR2.CLKC [1: 0] bit selection clock source
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3 register
5. Set USARTn_PR to select the prescaler value and USARTn_BRR register to set the communication baud rate (not required when clock source is external clock source).
6. Enable transmitter (USARTn_CR1. TE=1), if you need to use the transmit data register empty interrupt, set USARTn_CR1. TXEIE=1
7. Wait to Transmit data register empty, write communication data to USARTn_DR.TDR, transfer data to transmit shift register, transmit start
(When the CTS function is valid, when the USARTn_CTS input is low, the data is transferred to the transmit shift register, and the transmission starts)
8. Repeat Step 7 if you need to transmit data continuously
9. Confirm that the transmit is complete by confirming the USARTn_SR.TC bit. In the case of continuous data transmission and use of transmission interrupt, the last transmission data can be written through TI interrupt, and USARTn_CR1. TXEIE write 0, USARTn_CR1. TCIE writes 1, after the last frame of data is sent, the transmission is interrupted

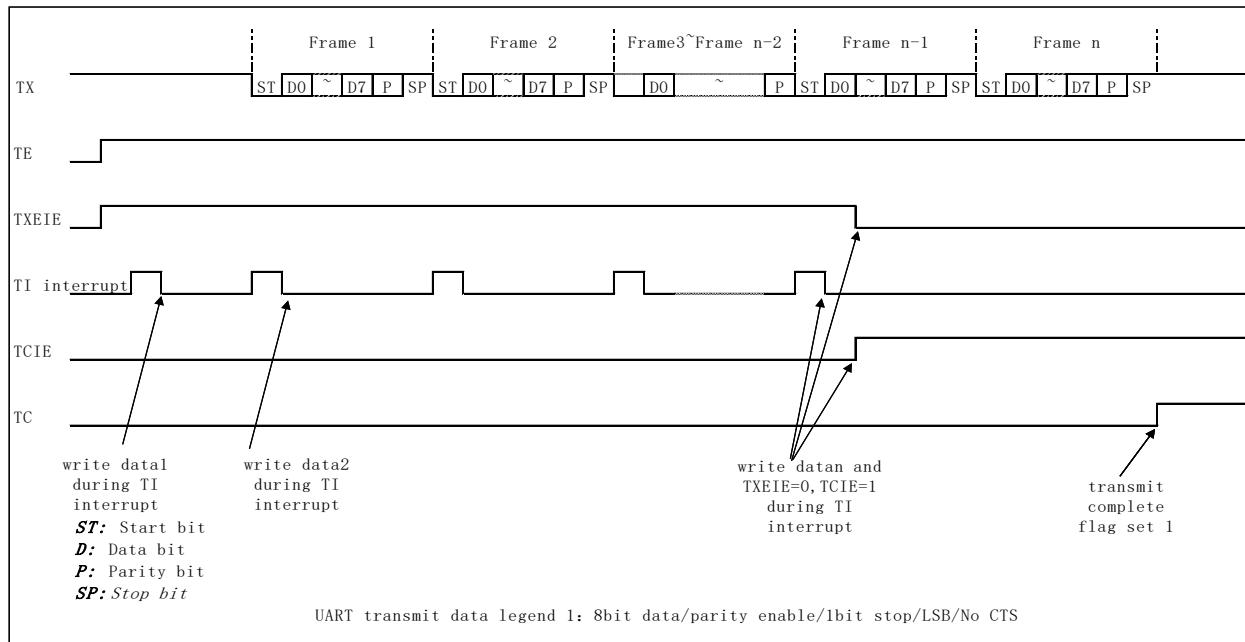


Figure 29-3 UART transmit data legend 1

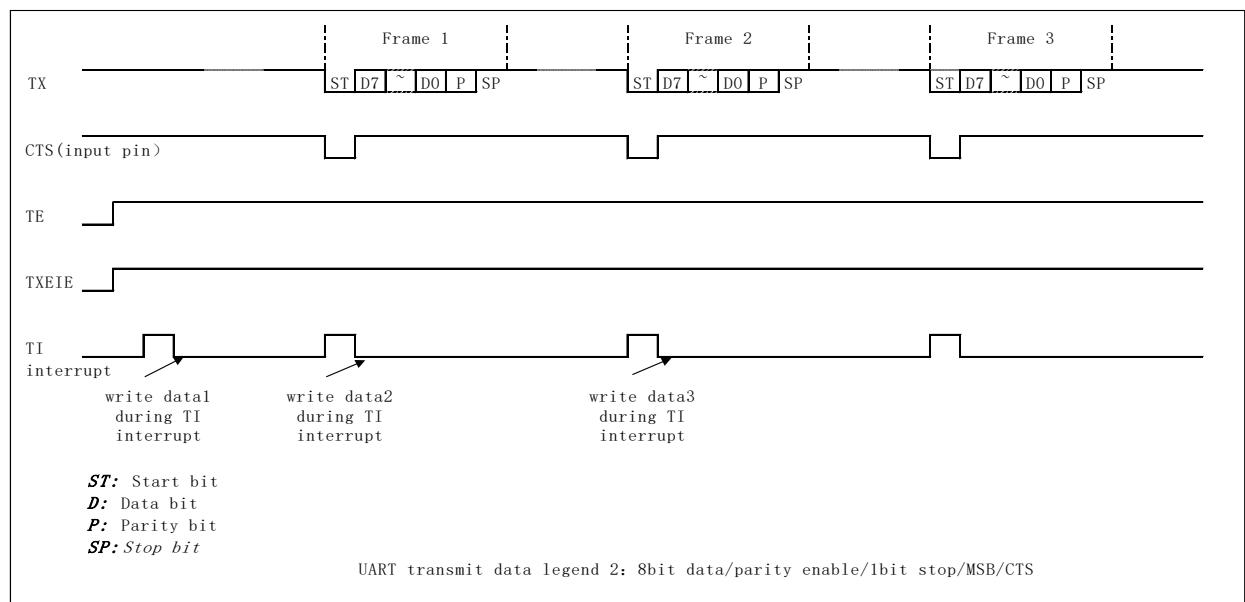


Figure 29-4 UART transmit data legend 2

Transmitter interrupt

The UART mode transmitter supports two types of interrupts, Transmit data register empty interrupt TI and transmission complete interrupt TCI.

TXEIE = 1, TI interrupt occurs when the USARTn_DR.TDR register value is sent to the transmit shift register.

TCIE = 1, TCI interrupt occurs when USARTn_DR.TDR register is not updated when the last bit of data is sent.

29.4.1.5 Receiver

Receivers can receive 8 or 9 bits of data depending on the set value of the USARTn_CR1.M bit. After the receiver sets the enable bit (USARTn_CR.RE) to 1 and detects the start bit, the data received on the RX pin is transferred to the receive shift register, and when a frame is filled, the data is transferred from the receive shift register to the receiving data register USARTn_DR.RDR.

The order in which data is received is: Start bit -> data bit (MSB/LSB) -> parity bit (with or without) -> 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.

A request can only read data once by receiving data register full interrupt or DMA read.

Start bit detection

Start bit detection can select low level mode or falling edge mode, depending on USARTn_CR1.SBS bit, low level detection when USARTn_CR1.SBS = 0, falling edge detection when USARTn_CR1.SBS = 1.

Sampling and reception tolerance

After detecting the start condition (low level or falling edge), USART synchronizes the received data based on the internal basic clock to start the data reception.

Data sampling in the center of the data, USARTn_CR1.OVER8 = 0 is sampled at the 8th internal base clock and USARTn_MR.OVER8 = 1 is sampled at the 4th internal base clock.

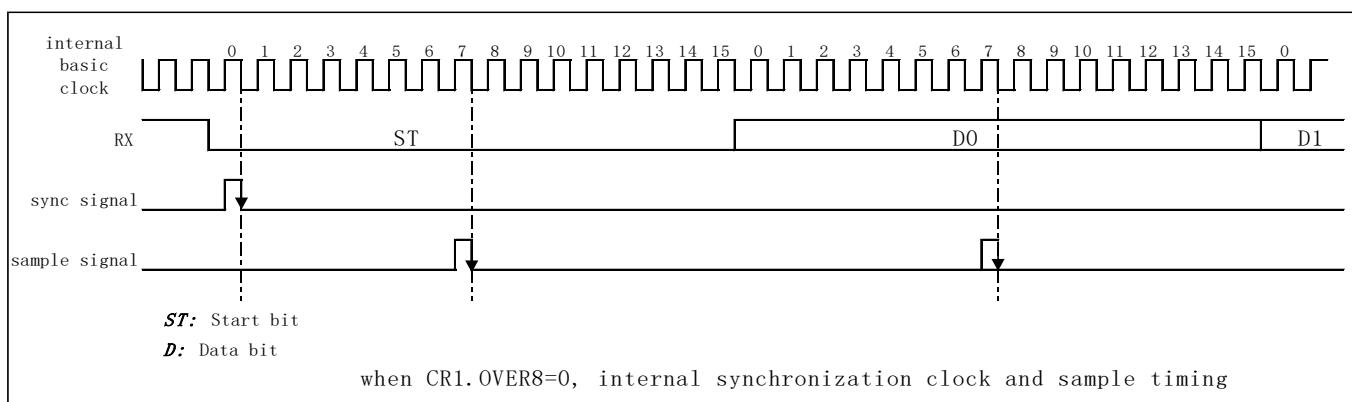


Figure 29-5 UART internal synchronization and sampling timing

The UART asynchronous receiver will work properly only if the total clock system skew is less than the tolerance of the UART receiver. Factors affecting the total deviation include:

- Deviation due to transmitter error (which also includes the local oscillator deviation of the transmitter)
- Error caused by quantization of baud rate of receiver

- Receiver local oscillator deviation
- Deviation caused by transmission lines

The maximum deviation allowed by the UART asynchronous receiver for correct reception of data depends on the following options:

- 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
- Defined by the OVER8 bit in the USART_CR1 register 8 bit or 16 bit oversampling
- Whether to use the fractional baud rate as defined by the FBME bit in the USART_CR1 register

Table 29-2 Tolerance of UART receiver when DIV_Fraction is 0

FL	OVER8bit=0	OVER8bit=1
10	4.375%	3.75%
11	3.97%	3.41%
12	3.646%	3.125%

Table 29-3 Tolerance of UART receiver when DIV_Fraction is not 0

FL	OVER8bit=0	OVER8bit=1
10	3.88%	3%
11	3.53%	2.73%
12	3.23%	2.5%

In a special case, when the RX pin receives a continuous high level for the data length FL time, Table 29-2 and Table 29-3 The data specified in may vary slightly.

Receiving data setting procedure

1. Set USARTn_CR1 register to reset
2. Set the pin that UART needs to use
3. Through USARTn_CR2.CLKC [1: 0] bit selection clock source
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3 register
5. Set USARTn_PR to select the prescaler value and USARTn_BRR register to set the communication baud rate (not required when clock source is external clock source).
6. Enable receiver (USARTn_CR1. RE=1), if you need to use receive interrupt, set USARTn_CR1. RIE=1
7. When the start bit is detected, the receiver transmits the received data to the shift register and checks the parity bit and stop bit
 - 1) When a parity error occurs, 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 sent to the USARTn_DR.RDR register and the USARTn_SR.FE flag is set.
 - 3) When an overrun error occurs, the data is lost and set to the USARTn_SR.ORE flag

- 4) When no error occurs, the received data is transferred to the USARTn_DR.RDR register, and the USARTn_SR.RXNE flag is set. After reading the received data, repeat step 7 to receive data continuously

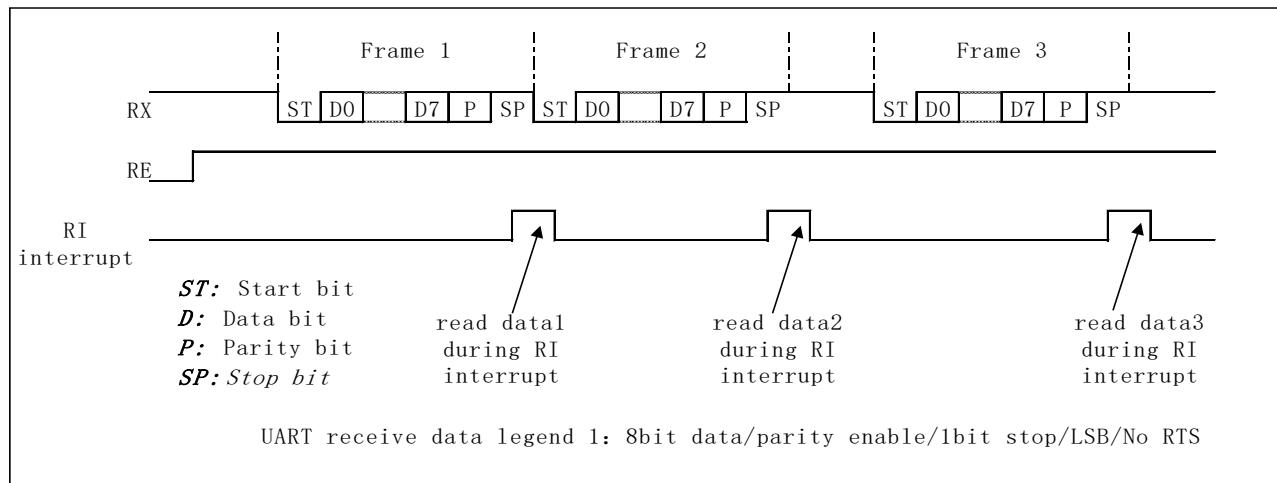


Figure 29-6 UART receive data legend 1

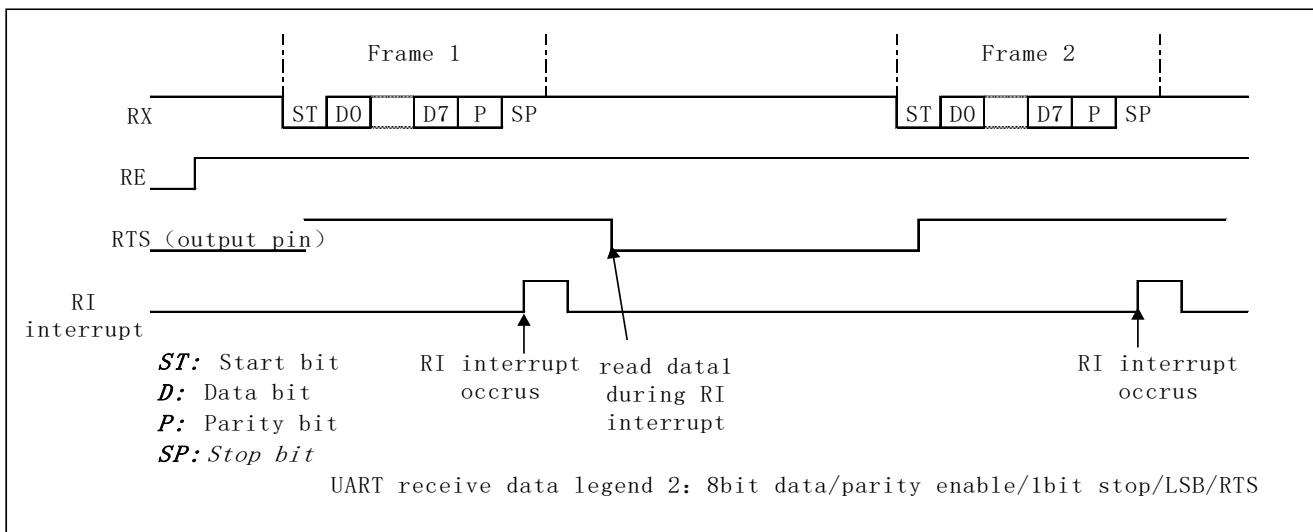


Figure 29-7 UART receive data legend 2

Error handling

There are three types of receiving errors at the time of receiving data: Overrun error (USARTn_SR.ORE), parity error (USARTn_SR.PE) and frame error (USARTn_SR.FE). Any reception error can no longer be received. You can restart data reception by clearing all error flags by writing the corresponding clear bit.

The overrun error occurs on condition that a new frame of data is received without the USARTn_DR.RDR register value being read, so the previous frame of data received should be read before the last bit of the current frame is received.

Parity errors occur on condition that parity errors occur.

Frame errors occur on the condition that the stop bit is low and only the first stop bit is checked in the case of two stop bits.

Data loss received when an overrun error occurs, RI interrupt does not occur.

When a parity error occurs, the data received is transferred to USARTn_DR.RDR, and the RI interrupt does not occur.

when a frame error occurs, the data received is transferred to USARTn_DR.RDR, and the RI interrupt does not occur.

Receiver interrupt

The UART mode receiver supports two interrupts, receive data register full interrupt RI and receive error interrupt EI.

USARTn_CR.RIE = 1, no receiving error occurred, and RI interrupt occurred when data was transferred from the receive shift register to the receive data register.

USARTn_CR.RIE=1, when an overrun error or a parity error or frame error occurs during reception, an EI interrupt occurs.

29.4.1.6 UART receive TIMEOUT function

When the UART receiving data stop bit is detected, the TIMEOUT counter starts. After the set TIMEOUT time (the setting unit is the number of received digits), when the next frame of received data is not detected, TIMEOUT will occur. If CR1.RE=1 at this time, Then the TIMEOUT status bit USARTn_SR.RTOF is set. If USARTn_CR1.RE=0 at this time, wait for the TIMEOUT status bit USARTn_SR.RTOF to be set after USARTn_CR1.RE=1.

The TIMEOUT counter adopts the counter of the Timer0 module, and the specific correspondence is as follows:

USART_1: Timer0 Unit1 Channel A

USART_2: Timer0 Unit1 channel B

USART_6: Timer0 Unit2 Channel A

USART_7: Timer0 Unit2 channel B

TIMEOUT function Timer0 comparison counter value setting

Timer0 is a 16-bit counter. The maximum frequency division of the counting clock can be 1024. The calculation formula of TMR0_CMPAR value setting is as follows:

$$\text{CMPA}_{<\text{B}>} \text{R} = \frac{\text{RTB}}{2^{\text{CKDIVA}_{<\text{B}>}}} - \alpha$$

CMPAR:TMR0_CMPAR register value. The calculation result must be rounded up.

α :The delay caused by Timer0 asynchronous counts synchronous circuit

Counting clock no frequency division, $\alpha= 7$

Counting clock 2 frequency division, $\alpha= 5$

Counting clock 4, 8, 16 frequency division, $\alpha= 3$

Counting clock 32 frequency division and above, $\alpha= 2$

RTB: Receive Timeout Bits, minimum = received frame length + $\alpha \times 2^{\text{CKDIVA}\langle B \rangle}$

There is some error between the actual TIMEOUT time and the RTB value, error $\leq 2^{\text{CKDIVA}\langle B \rangle}$.

CKDIRA $\langle B \rangle$: TMR0.BCONR.CKDIVA $\langle B \rangle$ bit register value

TIMEOUT function setting procedure

1. Set USARTn_CR1 register to reset
2. Set the pin that UART needs to use
3. Select the clock source through the USARTn_CR2.CLK[1:0] bits (if the internal clock source is selected, set CR2.CLKC[0]=1)
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3 register
5. Set USARTn_PR to select the prescaler value and USARTn_BRR register to set the communication baud rate (not required when clock source is external clock source).
6. USARTn_CR1.RTOE=1, if you need to use interrupt, set USARTn_CR1.RTOIE=1
7. Set TMR0.BCONR.CSTA $\langle B \rangle$ =0
8. Set TMR0_CNTA $\langle B \rangle$ R to 0, set the TMR0_CMPA $\langle B \rangle$ R register and
The TMR0.BCONR.CKDIVA $\langle B \rangle$ register determines the TIMEOUT time,
Set TMR0.BCONR.HCLEA $\langle B \rangle$ =1, TMR0.BCONR.HSTAA $\langle B \rangle$ =1,
TMR0.BCONR.ASYNCLKA $\langle B \rangle$ =1, TMR0.BCONR.SYNSA $\langle B \rangle$ =1
9. Enable receiver (USARTn_CR1.RE=1), if you need to use receive interrupt, set USARTn_CR1.RIE=1
10. When TIMEOUT is detected, follow these steps to disable the TMR0 timer and clear the USARTn_SR.RTOF status bit.
Set TMR0.BCONR.SYNSA $\langle B \rangle$ =0
Set TMR0.BCONR.CSTA $\langle B \rangle$ =0
Set TMR0.BCONR.SYNSA $\langle B \rangle$ =1
Clear the USARTn_SR.RTOF status bit by writing USARTn_CR1.CRTOF.

29.4.1.7 RX line wake-up stop mode function

When the UART communication is idle, the system can enter the stop mode to save current consumption. In the case of not changing the UART PORT setting, the UART unit 1 can wake up the system from the stop mode through the RX line. Specific steps are as follows:

1. When UART communication is idle, set USART_1_WUPI interrupt vector and INT_WUPEN. The RXWUEN bit enables the UART receive signal line wake-up stop mode function.
2. The system enters stop mode.
3. When the system detects the falling edge of the RX line, it returns from stop mode and disables the function in the USART_1_WUPI interrupt handler.

It should be noted that when the communication party needs to wake up the system, it needs to transmit a frame of wake-up data (0x00 is recommended), the data will not be received by the UART and the relevant flag will not be set. And the communication party needs to wait for the time required to wake up from the system stop mode before transmitting the UART communication data.

The UART RX line wake-up function can filter the noise on the RX line. For details, please refer to the USART_SYCTLREG register.

29.4.1.8 UART half-duplex communication mode

UART mode supports single-wire half-duplex mode, UART mode by setting USARTn_CR3.HDSEL = 1 enables single-wire half-duplex mode.

Single-wire half-duplex mode:

- TX and RX lines are connected from inside and no RX pins are used.
- In the absence of data transfer, the TX pin is released. Therefore, when single-line half-duplex mode is used, floating input without data transmission is avoided by pulling up the TX line.

In addition, half-duplex mode is similar to normal UART mode communication. Note that the transmitting process is not blocked by hardware. At USARTn_CR1.TE = 1, the transmitting takes place as soon as data is written to the USARTn_DR.TDR register. So conflicts on the line must be managed by software.

29.4.1.9 UART interrupt and Events

Table 29-4 UART Interrupt/Event Table

Functional name	Enable bit (interrupt only)	Flag bit	Can be used as an event source
Reception error interrupt	RIE	ORE,FE,PE	Yes
Received data register full interrupt	RIE	RXNE	Yes
Transmit data register empty interrupt	TXEIE	TXE	Yes
Transmit complete interrupt	TCIE	TC	Yes
TIMEOUT interrupt	RTOIE	RTOF	Yes
RX line wake-up stop mode interrupt	INT_WUPEN.RXWUEN	-	No

29.4.2 Multiprocessor communication

29.4.2.1 Feature brief

Multiprocessor communication mode refers to a communication mode in which multiple processors share communication lines. The processor is divided into transmitting and receiving stations, and each receiving station has its own ID. There are two types of transmitting data: Receiving station ID and communication data. By adding MPB bits to the data format to distinguish whether the receiving station ID or communication data is currently being sent. When the MPB bit is 0, the current frame is the communication data, and when the MPB bit is 1, the current frame is the ID of the receiving station. All receiving stations can receive the ID sent by the transmitting station and compare it with their ID. If it is consistent, the receiving data will be received, and if it is inconsistent, the receiving station will enter silent mode (neither receiving data nor setting the relevant flag) until the ID is received again.

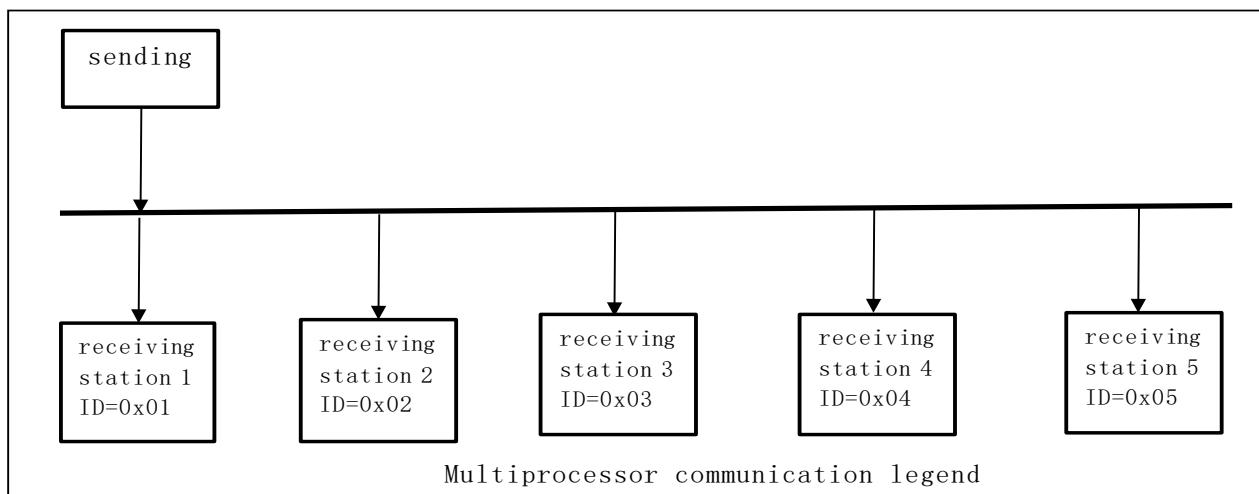


Figure 29-8 Multiprocessor Communication Legend

29.4.2.2 Data format

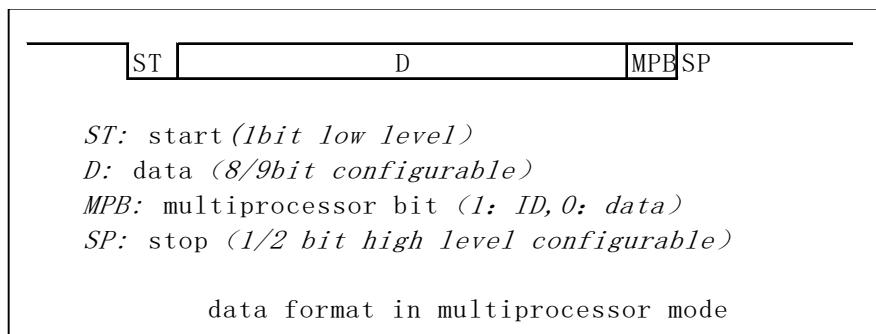


Figure 29-9 Multiprocessor Mode Data Format

29.4.2.3 Action description

In the Multiprocessor mode, the parity bit function is invalid, and the Multiprocessor bit function is added. The other functions such as clock and interrupt are the same as UART mode.

Transmitting station action

1. Set USARTn_CR1 register to reset
2. Set the pins to be used
3. Through USARTn_CR2.CLKC [1: 0] bit selection clock source
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3 register
5. Set USARTn_PR to select the prescaler value and USARTn_BRR register to set the communication baud rate (not required when clock source is external clock source).
6. Enable transmitter (USARTn_CR1. TE=1), if you need to use the transmit data register empty interrupt, set USARTn_CR1. TXEIE=1
7. Wait for transmit data register to be empty, set USARTn_DR.MPID bit to 1 (transmit ID), write ID to USARTn_DR, transmit ID
(When the CTS function is valid, when the USARTn_CTS input is low, the data is transferred to the transmit shift register, and the transmission starts)
8. Set USARTn_DR.MPID bit to 0 (transmit data), write data to USARTn_DR, transmit data
(When the CTS function is valid, the data is transferred to the transmit shift register when the USARTn_CTS input is low, and the transmission starts)
9. If you need to transmit the data continuously, repeat step 8. If you need to change the ID and then transmit data, repeat steps 7 and 8.
10. Confirm that the transmit is complete by confirming the USARTn_SR.TC bit. In the case of continuous data transmission and use of transmission interrupt, the last transmission data can be written through TI interrupt, and USARTn_CR1. TXEIE write 0, USARTn_CR1. TCIE writes 1, after the last frame of data is sent, the transmission complete interrupt generation

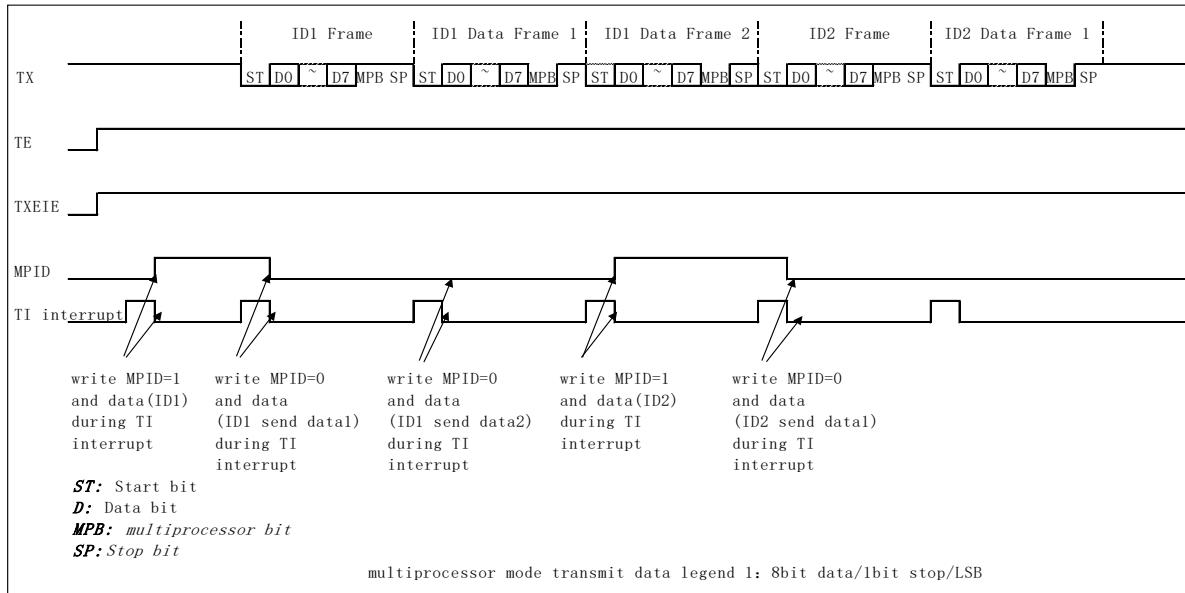


Figure 29-10 Example of transmitting data in multiprocessor mode

Receiving station action

In multiprocessor mode, the receiving station must ensure that each ID data is received and compared with its own ID. If it is consistent, the receiving station receives the data. If it is inconsistent, the receiving station enters silent mode (no data is received, and no related flag is set) until the next ID data is received. This function is implemented through the USARTn_CR1.SLME bit.

Data is normally received when USARTn_CR1.SLME=0.

When USARTn_CR1.SLME = 1, no data will be received unless data with MPB bit 1 (ID) is received, no RI interrupt will occur, and the error flags FE and ORE will not be set. When data with MPB bit 1 is received (ID), USARTn_CR1.SLME bits are cleared automatically, data are normally received and interruptd.

Action steps:

1. Set USARTn_CR1register to reset
2. Set the pins to be used
3. Through USARTn_CR1.CLKC [1: 0] bit selection clock source
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3 register
5. Set USARTn_PR to select the prescaler value and USARTn_BRRregister to set the communication baud rate (not required when clock source is external clock source).
6. USARTn_CR1.RE=1, USARTn_CR1.SLME=1 (waiting for receiving ID), if using receive interrupt, set USARTn_CR1.RIE=1
7. When the start bit is detected, the receiver receives the data to receive shift register and checks the USARTn_SR.MPB bit
8. If USARTn_SR.MPB = 1, USARTn_CR1.SLME bits are reset automatically and receive data

normally. The software compares the received ID with its own ID.

- 1) If the ID is consistent, the data received, interrupt generation, and error detection perform normally, the same as the data received by UART
- 2) If the IDs are inconsistent, the software writes the USARTn_CR1.SLME bit to 1 again, and repeats the action of 8

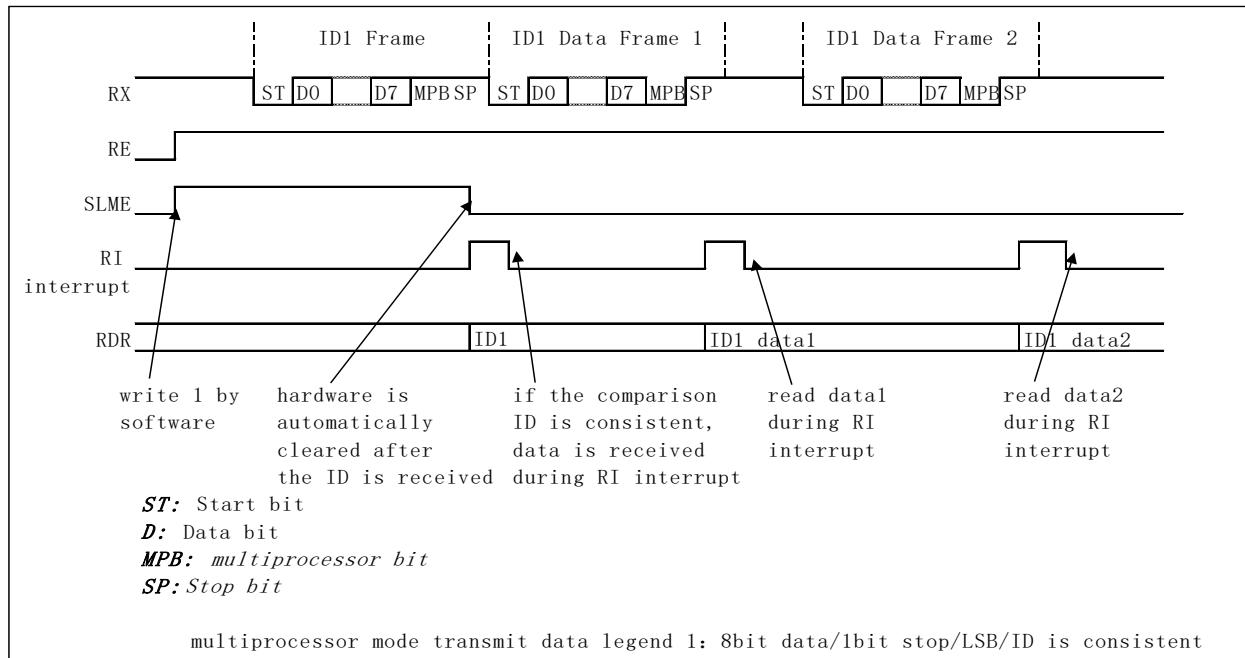


Figure 29-11 Figure 1 for receiving data in multiprocessor mode

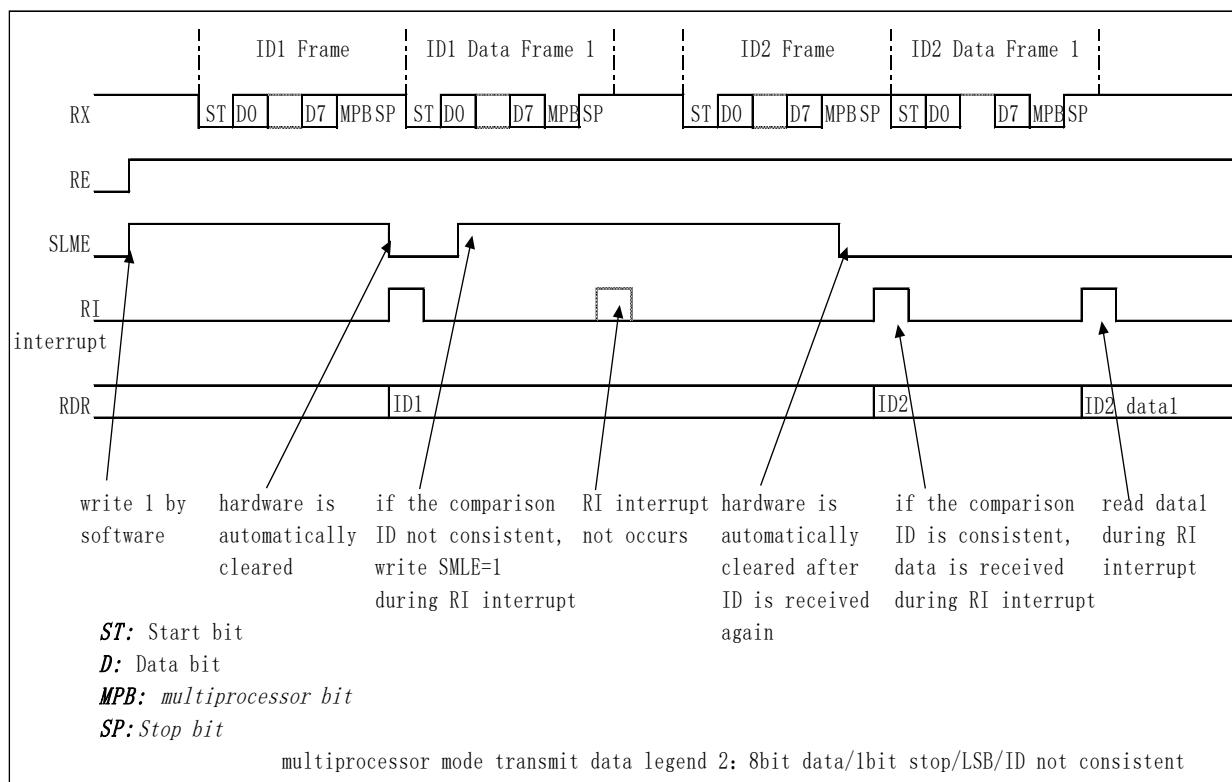


Figure 29-12 Figure 2 for receiving data in multiprocessor mode

29.4.2.4 Interrupt and events

The interrupt mode is the same as the UART mode except for no parity error.

Table 29-5 Multiprocessor Mode Interrupt/Event Table

Functional name	Enable bit (interrupt only)	Mark	Can be used as an event source
Reception error interrupt	RIE	ORE, FE	Yes
Received data register full interrupt	RIE	RXNE	Yes
Transmit data register empty interrupt	TXEIE	TXE	Yes
Transmit complete interrupt	TCIE	TC	Yes

29.4.3 UART-LIN

29.4.3.1 Feature brief

LIN is short for Local Interconnect Network, which is a low-speed (1-20 kbps) serial communication protocol to reduce the cost of automotive network.

29.4.3.2 LIN data format

A frame of LIN data is composed of start bit + 8 bit data + 1 stop bit, which is used to transmit and receive data in LSB mode. The data behavior on the LIN bus is shown below.

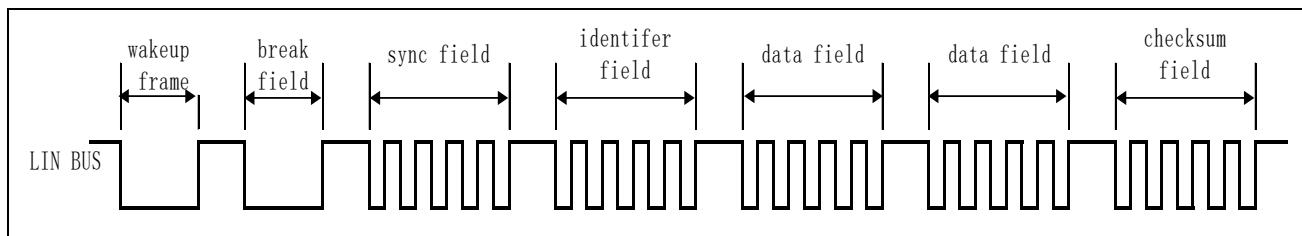


Figure 29-13 LIN bus data behavior

29.4.3.3 LIN transmit

Transmitting of Wake Signal Frames

By transmitting data 0x80, the corresponding signal frame is sent.

Break field transmitting

The synchronization break field indicates the beginning of a frame of data, and the hardware of this product supports the transmission of the LIN break field.

The width of the low level of the LIN break field is set by the register USARTn_CR2.SBK.

When USARTn_CR2.SBKM=0, write 1 to USARTn_CR2.SBK, and the hardware automatically transmits the break field.

When USARTn_CR2.SBKM=1, after writing 1 to USARTn_CR2.SBK, write data 0x00 to the transmit data register USARTn_DR.TDR to start transmitting the break field.

After the break field is sent, USARTn_CR2.SBK is automatically cleared.

Synchronization field transmitting

Transmitting Synchronous Field Data by Transmitting Data 0x55

Bus error detection

When USARTn_CR2.BEE=1, when the hardware detects that the data on the bus is inconsistent with the sent data, the USARTn_SR.BE flag is set, and when USARTn_CR2.BEIE=1, the corresponding interrupt is generated.

29.4.3.4 LIN Receive

Detection of Wake Signal Frames

When USARTn_CR2.WKUPE=1, the hardware automatically detects the wake-up signal, and when it detects that the low level width of the RX line is greater than or equal to 2.5 bits of data width (130uS at 19.2Kbps), the USARTn_SR.WKUP flag is set, and when USARTn_CR2.WKUPIE=1 , generate the corresponding interrupt.

The USARTn_CR2.WKUPE register is only set to 1 when the system needs to wait for the wake-up signal, and needs to be set to 0 in other cases.

Detection of Synchronous Break fields

The hardware of this product supports synchronization break field detection. When it is detected that the low level width of the communication line is greater than or equal to the value set by the USARTn_CR2.LBDL register, and the break delimiter is detected, the USARTn_SR.LBD flag is set. When USARTn_CR2.LBDIE=1, The corresponding interrupt is generated.

It should be noted that when the synchronization break field is detected, the USARTn_SR.FE flag will be set. When the USARTn_SR.LBD flag is cleared, the USARTn_SR.FE flag needs to be cleared, and the USARTn_DR.RDR register is read (the read value is 0x00).

Detection of Synchronous Fields and Baud Rate Measurement

When a break field is detected, the hardware automatically measures the frequency of the sync field. The clock of the measurement counter is set by the USARTn_PR.LBMPSC bit, and the value of the counter is stored in the USARTn_BTMC register. The selected counter clock frequency is divided by the USARTn_BTMC register value to obtain the baud rate of the master node.

It should be noted that the reading of the USARTn_BTMC register needs to be read after the synchronization field is received.

Receiving data setting procedure

1. Set the USARTn_CR1 register to the reset value
2. Set the pins required for the UART

3. Select the clock source through the USARTn_CR2.CLKC[1:0] bits
4. Set USARTn_CR1 (RE=1, RIE=0), USARTn_CR2, USARTn_CR3 registers

(Steps 5-6 below are for the detection of wake-up signal frames, skipped if not necessary)

5. Set USARTn_CR2.WKUP=1, wait for the wake-up signal
6. Wake-up signal detected, USARTn_CR2.WKUP=0, waiting for sync break field

(The following step 7 is the sync break field detection, skip it if it is not necessary)

7. After detecting the sync break field, clear the USARTn_SR.PE/FE/LBD flag and confirm that the received data is 0x00.

(The following steps 8~9 are the frequency measurement of the synchronization field and the calculation of the communication baud rate, skip it if it is not necessary)

8. If need to use receive interrupt, set USARTn_CR1. RIE=1
9. After the synchronization break field is detected and the synchronization field reception is completed, read the USARTn_BTMC register to calculate the baud rate, and confirm that the received data is 0x55

(The following steps 10~12 are the receiving identification field, data and checksum, the process is the same as the UART receiving data process)

10. Receive identification field
11. Receive data
12. Receive checksum

29.4.3.5 LIN interrupts and events

Table 29-6 LIN Interrupt/Event Table

Functional name	Enable bit (interrupt only)	Flag bit	Can be used as an event source
Error interrupt	RIE	ORE,FE	Yes
	BEIE	BE	Yes
Wake-up signal/break field detection interrupt	WKUPIE	WKUP	Yes
	LDBIE	LBD	Yes
Received data register full interrupt	RIE	RXNE	Yes
Transmit data register empty interrupt	TXEIE	TXE	Yes
Transmit complete interrupt	TCIE	TC	Yes

29.4.4 Smartcard

29.4.4.1 Connection diagram

Supports the smartcard communication protocol specified by ISO/IEC 7816-3. The following figure shows the connection diagram of smartcard mode.

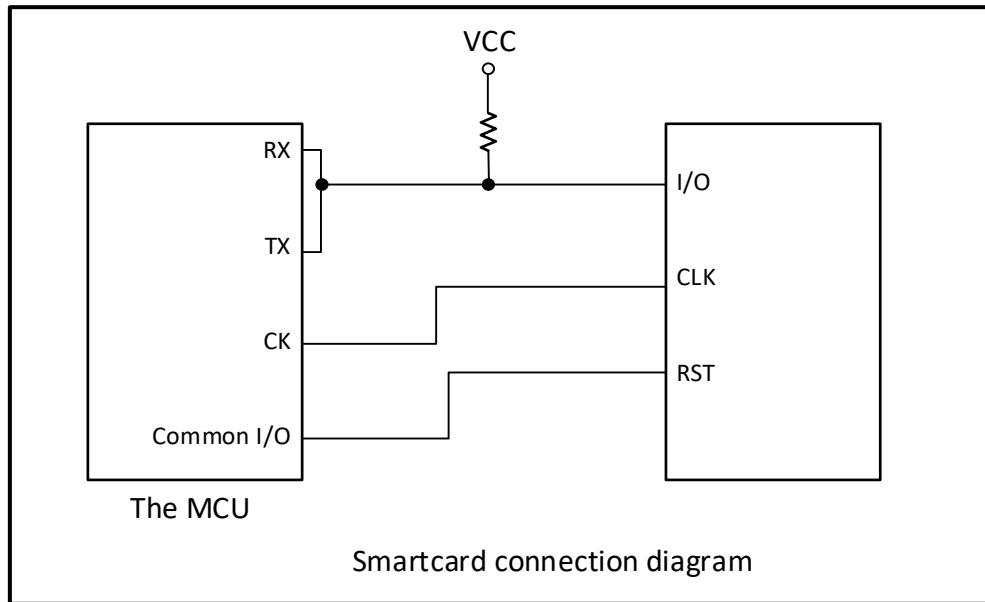


Figure 29-14 Smartcard Connection Diagram

29.4.4.2 Clock

Internal clock source

In smartcard mode, only the clock generated by the internal baud rate generator can be used as the clock source.

The basic clock number for one-bit data transmission is the set value of USARTn_CR3.BCN[2:0].

The clock output in smartcard mode is controlled by setting the register USARTn_CR2.CLKC[1:0] bits.

Maximum baud rate

For internal clock source, the Baud Rate formula generated by the internal Baud Rate Generator is:

$$B = \frac{C}{2 \times BCB \times (DIV_Integer + 1)}$$

B: Baud rate unit: MBps

C: The clock set by USARTn_PR.PSC[1:0] bits (PCLK, PCLK/4, PCLK/16, PCLK/64) Unit: MHz

DIV_Integer: USARTn_BRR.DIV_Integer setting

BCN: USARTn_CR3.BCN register setting value

When C is PCLK, DIV_Integer=0, BCN=0, the maximum baud rate is PCLK/64(MBps).

Sampling and reception tolerance

After detecting the drop of RX, the USART will clock synchronization the received data based on the internal base clock to start data reception. Received data will be sampled in the data center.

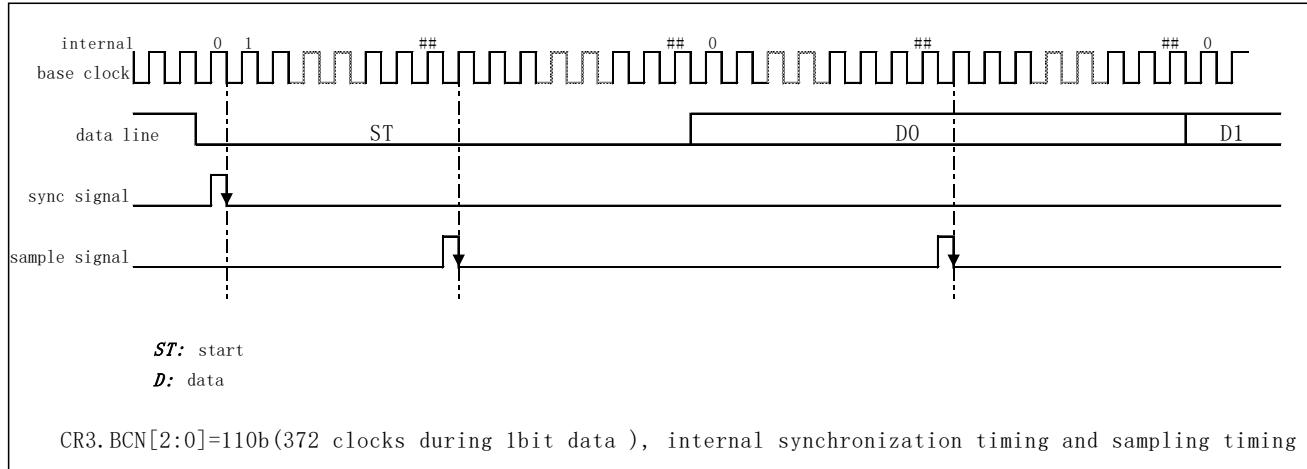


Figure 29-15 Smartcard Mode Synchronization Timing and Sampling Timing Diagram

The following formula is used to calculate the receiving tolerance:

$$RM[\%] = \left| 0.5 \times \left(1 - \frac{1}{BCN} \right) - 9.5CFD \right| \times 100$$

RM: Receive tolerance

BCN: The number of clocks required for one-bit data transmission (USARTn.CR3.BCN[2:0] set value)

CFD: clock frequency deviation

29.4.4.3 Data format

In smartcard mode, a frame of data consists of start bits, data bits and parity bits.

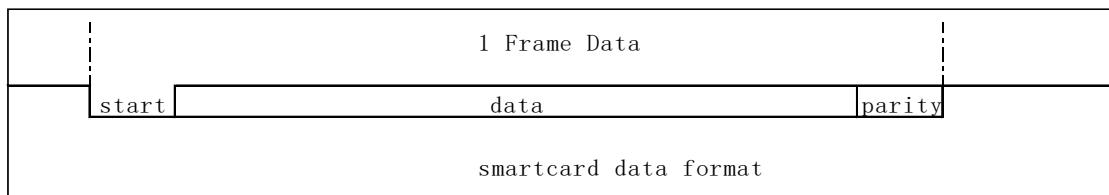


Figure 29-16 smartcard mode Data Format

Start bit

The start bit is fixed with a low level.

Data bit

The data bits are fixed to 8-bit data.

Parity bit

The parity bit needs to be configured as 1-bit even parity.

29.4.4.4 Initial setting procedure of smartcard

1. Set USARTn_CR1 register to reset
2. Set the pins to be used
3. Status register is confirmed, USARTn_SR register is set to reset value
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3 register
5. Set the USARTn_PR to select the prescaler value, and the 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, do not set TE and RE to 1 at the same time

When switching from the transmitting mode to the receiving mode, or switching from the receiving mode to the transmitting mode, the above steps 1 to 7 need to be reset.

29.4.4.5 Smartcard Mode Action Description

In smartcard mode, the flag bit of TI interrupt (transmit data register empty interrupt) is USARTn_SR.TC bit. TI interrupt is generated when USARTn_SR.TC=1 and USARTn_CR1.TXEIE=1.

Functional Overview

When transmitting data, there is a guard time of 2etu (Elementary Time Unit) or more between two frames of data (from the end of the parity bit to the start of the start bit of the next frame).

When transmitting data, if the error signal sent by the receiver is detected, the data will be automatically resent after 2etu.

When a parity error occurs in the received data, a low level of 1etu is sent, which is an error signal, and the timing of transmitting the error signal is 10.5etu after the start of reception.

Transmit instructions

1. After a frame of data is sent, if an error signal sent by the receiver is detected, USARTn_SR.FE will be set to 1 (if USARTn_CR.RIE=1, an error interrupt will occur), the USARTn_SR.TC flag will not be set to 1, and the data will be automatically retransmitted. The USARTn_SR.FE bit must be cleared before the next frame parity bit is received.
2. After a frame of data is sent without error, the USARTn_SR.TC flag is set, and a TI interrupt occurs when USARTn_CR1.TXEIE=1. By writing data again, data can be sent continuously.

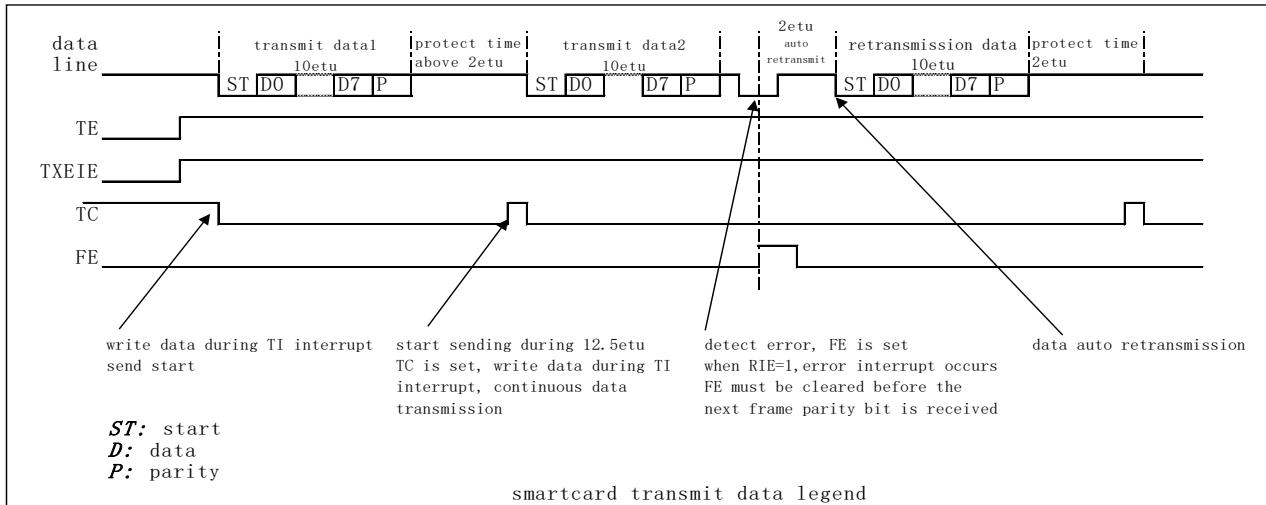


Figure 29-17 Example of transmitting data in smartcard mode

Receive instructions

- When receiving data, if a parity error is detected, USARTn_SR.PE is set, and an EI interrupt occurs when the interrupt is enabled. The USARTn_SR.PE bit needs to be cleared before the next frame parity bit is received.
- When a parity error occurs, a low level of 1etu will be sent, that is, an error signal, requiring the transmitter to resend the data.
- To receive data normally, you can read the received data through RI interrupt and receive it continuously.
- When receiving data, detecte the overrun error.

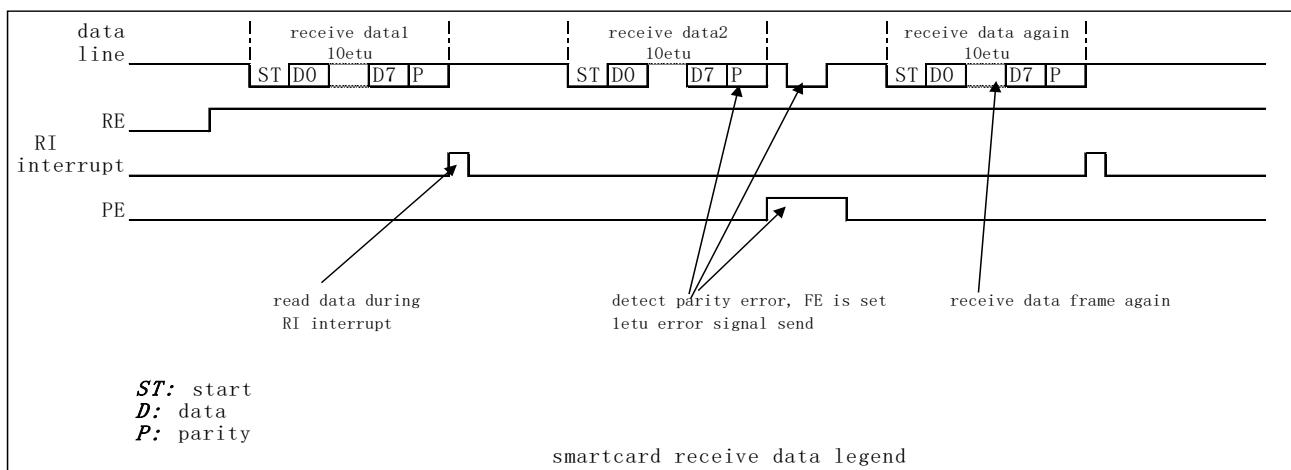


Figure 29-18 Example of receiving data in smartcard mode

29.4.4.6 Interrupt and events

Table 29-7 Smartcard Mode Interrupt/Event Table

Functional name	Enable bit (interrupt only)	Flag bit	Can be used as an event source
Error interrupt	RIE	ORE,PE,FE	Yes
Receive data full interrupt	RIE	RXNE	Yes
Transmit data register empty interrupt	TXEIE	TC	Yes

29.4.5 Clock synchronization mode

29.4.5.1 Clock

The clock synchronization mode can select 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 synchronization clock is output from the USARTn_CK pin, and a frame of data outputs eight clock pulses. When neither transmitting nor receiving data, the clock output is fixed at a high level.

External clock source

The external clock source is the input clock from the USARTn_CK pin as the communication clock.

Maximum baud rate

For internal clock source, the Baud Rate formula generated by the internal Baud Rate Generator is:

$$B = \frac{C}{4 \times (\text{DIV_Integer} + 1)}$$

B: Baud rate unit: MBps

C: The clock set by USARTn_PR.PSC[1:0] bits (PCLK, PCLK/4, PCLK/16, PCLK/64) Unit: MHz

DIV_Integer: USARTn_BRR.DIV_Integer setting

When the internal clock source is used, when C is PCLK and DIV_Integer=1, the maximum baud rate is PCLK/8(MBps). Note that the DIV_Integer must not be set to 0 in clock synchronous mode.

When the external clock source is used, the maximum frequency of the external input clock is required to be PCLK(MHz)/6, so the maximum baud rate is PCLK/6(MBps).

It should be noted that, in addition to the calculation method based on PCLK described above, the maximum communication baud rate in the synchronous mode also needs to refer to the maximum communication baud rate specified in the electrical characteristics chapter.

29.4.5.2 Data format

One frame of data in clock synchronization mode is composed of eight bits, the transmission and reception of one frame of data requires 8 synchronization clock pulses . When transmitting 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 synchronization clock is fixed at a high level when there is no data transfer. After the last bit is sent, the communication line keeps the last bit value.

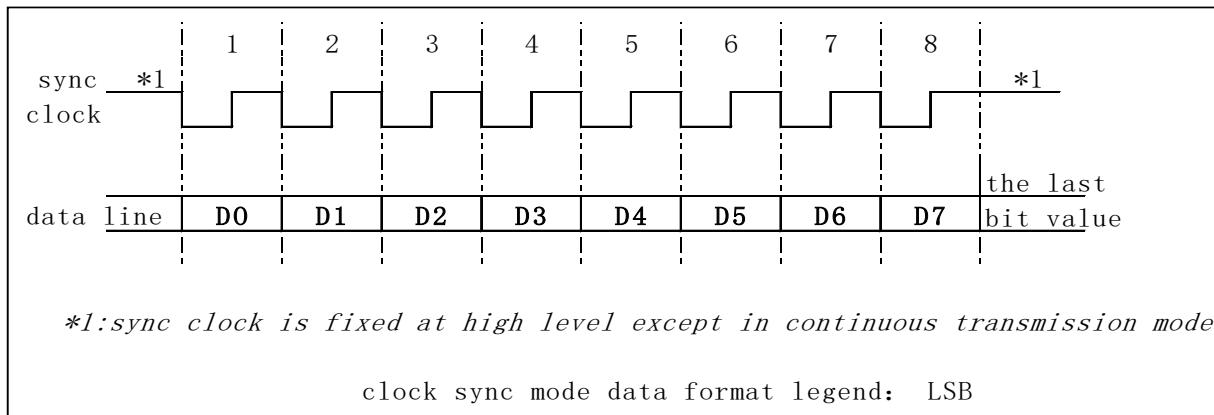


Figure 29-19 Clock Sync Mode Data Format

29.4.5.3 Modem operation

Modem operations include CTS and RTS functions. The RTS function is valid when USARTn_CR3.RTSE=1, and the CTS function is valid when USARTn_CR3.CTSE=1.

CTS function

The CTS function is to control the transmission of data through the input of the USARTn_CTS pin. Only when the USARTn_CTS pin is input low level can data be sent. If the USARTn_CTS input is high level during the process of transmitting data, the data being sent will not be affected.

RTS function

The RTS function refers to outputting a low level through the USARTn_RTS pin to request the other party to transmit data.

The USARTn_RTS pin outputting a low level needs to meet all the following conditions:

- Receive enable (USARTn_CR1.RE = 1) and not receiving data
- 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 error

If all the above conditions cannot be met at the same time, USARTn_RTS will output a high level.

29.4.5.4 Transmitter

Transmitter enable bit (USARTn_CR1.TE) When set to 1, the data in the transmit shift register is serially output in the USARTn_TX pin and the corresponding clock pulse is output in the USARTn_CK pin.

The transmission data register USARTn_DR.TDR and the internal transmit shift register constitute a dual buffer structure, which can transmit data continuously.

To ensure the correctness of transmitting data, a request can only be written once by transmitting data register empty interrupt or DMA.

Transmitting data setting procedure

1. Set USARTn_CR1, USARTn_SR1 register to reset
2. Set the pins to be used
3. Through USARTn_CR2.CLKC [1: 0] bit selection clock source
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3 register
5. Set USARTn_PR to select the prescaler value and USARTn_BRR register to set the communication baud rate (not required when clock source is external clock source).
6. Enable transmitter (USARTn_CR1. TE=1), if you need to use the transmit data register empty interrupt, set USARTn_CR1. TXEIE=1
7. Wait to transmit data register empty, write communication data to USARTn_DR.TDR, transfer data to transmit shift register, transmit start
(When the CTS function is valid, when the USARTn_CTS input is low, the data is transferred to the transmit shift register, and the transmission starts)
8. Repeat Step 7 if you need to transmit data continuously
9. Confirm that the transmit is complete by confirming the USARTn_SR.TC bit. In the case of continuous data transmission and use of transmission interrupt, the last transmission data can be written through TI interrupt, and USARTn_CR1. TXEIE write 0, USARTn_CR1. TCIE is written to 1, after the last data transmission is completed, a transmission completion interrupt is generated.

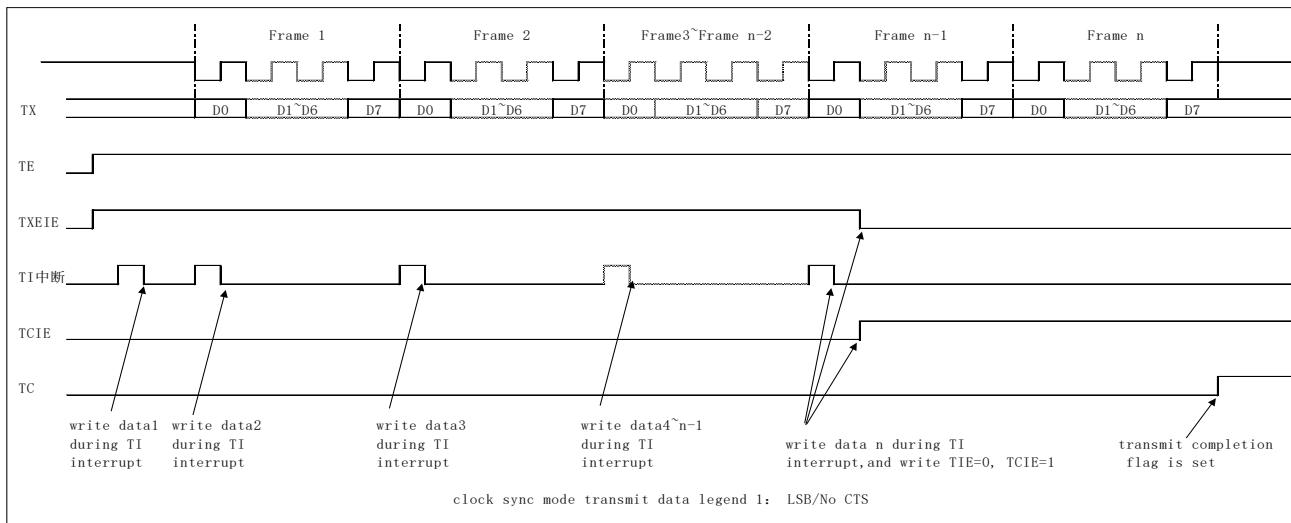


Figure 29-20 Example 1 of transmitting data in clock synchronization mode

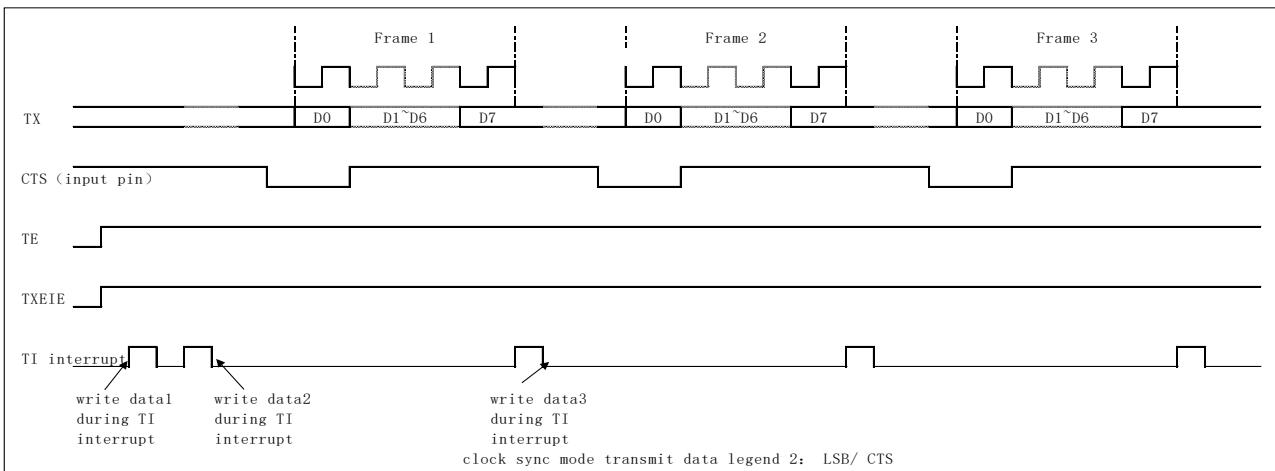


Figure 29-21 Example 2 of transmitting data in clock synchronization mode

Transmitter interrupt

The Clock Synchronization Mode Transmitter supports two types of interrupts, Transmit data register empty interrupt TI and Transmit complete interrupt TCI.

TXEIE = 1, USARTn_DR.TDR register value sent to transmit shift register TI interrupt occurs.

TCIE = 1, TCI interrupt occurs when USARTn_DR.TDR register is not updated when the last bit of data is sent.

29.4.5.5 Receiver

Receiving data setting procedure

1. Set USARTn_CR1, USARTn_SR register to reset
2. Set the pins to be used
3. Through USARTn_CR2.CLKC [1: 0] bit selection clock source
4. Set USARTn_CR1, USARTn_CR2, USARTn_CR3register

5. Set USARTn_PR to select the prescaler value and USARTn_BRRregister to set the communication baud rate (not required when clock source is external clock source).
6. Enable receiver (USARTn_CR1. RE=1), if you need to use receive interrupt, set USARTn_CR1. RIE=1
(When using the RTS function, USARTn_RTS outputs a low level after RE=1)
7. Synchronize with the input synchronization clock or internally generated synchronization clock to start receiving data, receiving data to the receive shift register.
 - 1) When an overrun error occurs, the data is lost and set to the USARTn_SR.ORE flag
 - 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 currently received data is read before the last bit of the next frame of data is received. Repeat step 7 to achieve continuous Receive data function.

(When using the RTS function, USARTn_RTS outputs a low level after the data is read)

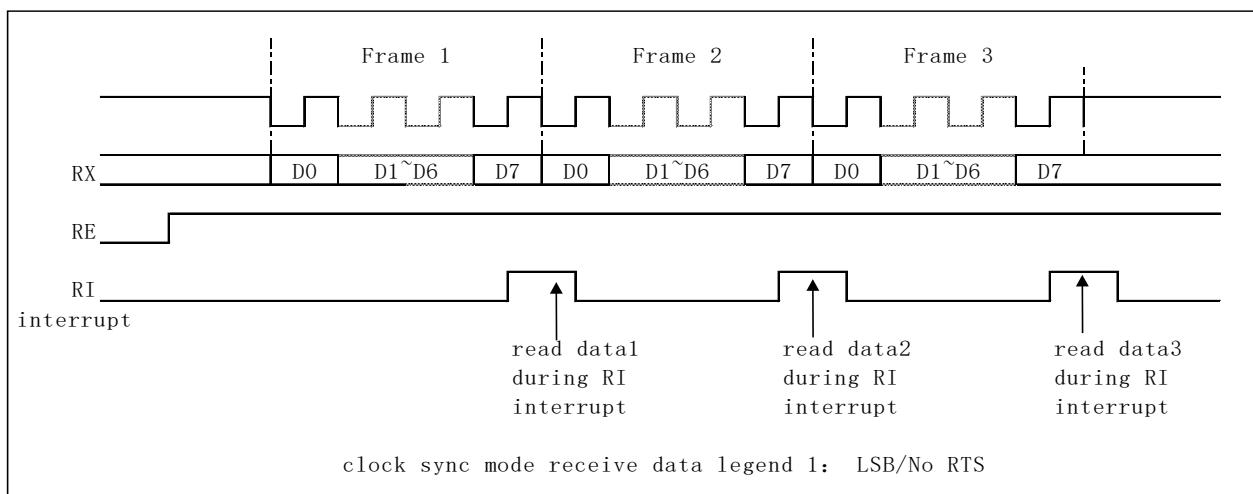


Figure 29-22 Clock synchronization mode receiving data legend 1

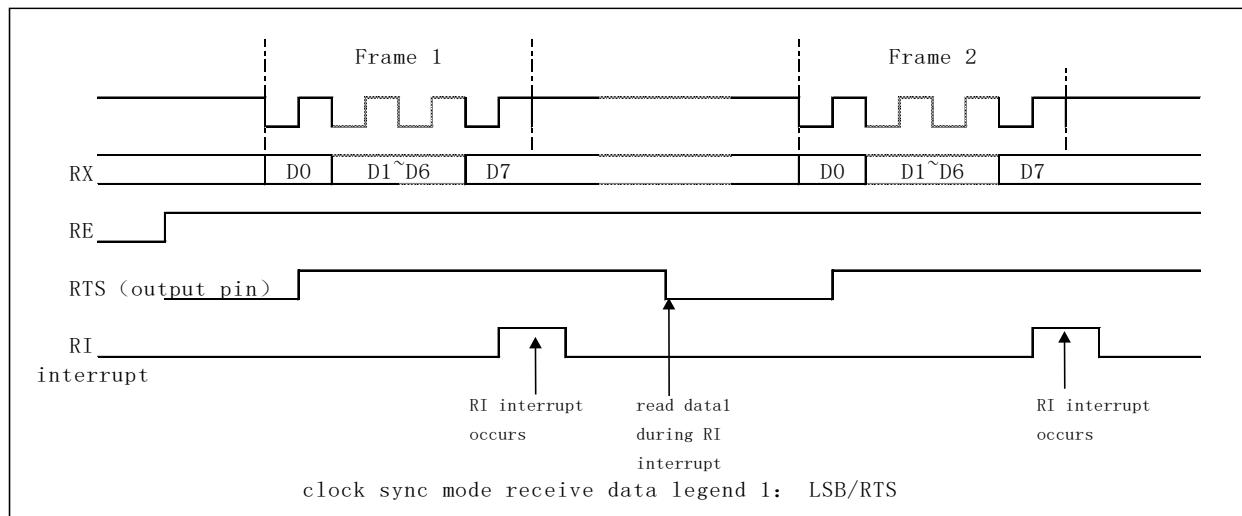


Figure 29-23 Clock synchronization mode receiving data legend 2

Error handling

An overrun error (USARTn_SR.ORE) is received while receiving data in clock synchronization mode. When a reception error occurs, data cannot be received or sent. You can restart the data transfer by clearing the error flag by writing the corresponding register clear bit.

The overrun error occurs when the USARTn_DR.RDR value is not read and new data is received, so the data received in the previous frame should be read before the last bit of the current frame is received. Data loss received when an overrun error occurs, RI interrupt does not occur.

Receiver interrupt

The clock synchronization mode receiver supports two kinds of interrupts, receive data register full interrupt RI and receive error interrupt EI.

RIE = 1, RI interrupt occurs when data is transferred from the receive shift register to the receive data register.

RIE=1, the EI interrupt occurs when an error occurs in the received data (overrun error).

29.4.5.6 Transmit and receive data at the same time

The USART clock synchronization mode supports full-duplex operation while transmitting and receiving data. A command is needed to write RE, TE, RIE and TXEIE into one while transmitting and receiving data. The other setting flow is the same as that of transmitter and receiver.

29.4.5.7 Clock Synchronization Mode Interrupt and Events

Table 29-8 Clock Synchronized Mode Interrupt/Event Table

Interrupt name	Enable bit (interrupt only)	Flag bit	Can be used as an event source
Reception error interrupt	RIE	ORE	Yes
Received data register full interrupt	RIE	RXNE	Yes
Transmit data register empty interrupt	TXEIE	TXE	Yes
Transmit complete interrupt	TCIE	TC	Yes

29.4.6 Digital filtering function

When USARTn_CR1.NFE = 1, the built-in digital filter function is effective. The digital filter is only effective in UART mode and removes noise from the received data line RX.

The built-in digital filter can filter out noise which less than 3/16 (USARTn_CR1.OVER8=0) or 3/8 (USARTn_CR1.OVER8=1) width of one bit of data.

If the digital filter starts again after the clock stops, the digital filter continues to work from the state left when the clock stops.

USARTn_CR.TE = 0 and USARTn_CR.RE = 0 reset the Flip-Flop state inside the digital filter to 1.

29.4.7 Interrupt

The following table gives the USART overall interrupt:

Table 29-9 USART overall interrupt list

Interrupt name	Symbol	Note
Error interrupt	USART_n_EI n=1~10	UART/Multiprocessor/Smartcard /Clock Synchronous Mode Receive Error Interrupt LIN error interrupt
Received data register full interrupt	USART_n_RI n=1~10	UART/Multiprocessor/LIN/Smartcard /Clock Synchronous Mode Receive Data Register Full Interrupt LIN error interrupt
Transmit data register empty interrupt	USART_n_TI n=1~10	UART/Multiprocessor/LIN/Smartcard /Clock Synchronous Mode Transmit Data Register Empty Interrupt
Transmit complete interrupt	USART_n_TCI n=1~10	UART/Multiprocessor/LIN/Smartcard /Clock Synchronous Mode Transmit Complete Interrupt
Wake-up signal/break field detection interrupt	USART_n_BRWKPI n=5/10	LIN Wakeup Signal/Break Field Detection Interrupt
UART receive TIMEOUT interrupt	USART_n_RTO n=1/2/6/7	UART receive TIMEOUT interrupt
RX line wake-up stop mode interrupt	USART_1_WUPI	USART_1 RX line wake-up stop mode interrupt

29.5 Register description

This chapter will describe the USART module control status and other related registers in detail. It should be noted that the function register bit is only valid when the corresponding function is configured for the channel, otherwise it defaults to 0.

The base address is as follows:

USART_1_BASE_ADDR: 0x4001CC00

USART_2_BASE_ADDR: 0x4001D000

USART_3_BASE_ADDR: 0x4001D400

USART_4_BASE_ADDR: 0x4001D800

USART_5_BASE_ADDR: 0x4001DC00

USART_6_BASE_ADDR: 0x40020C00

USART_7_BASE_ADDR: 0x40021000

USART_8_BASE_ADDR: 0x40021400

USART_9_BASE_ADDR: 0x40021800

USART_10_BASE_ADDR: 0x40021C00

Table 29-10 USART Register List

Register name	Offset address	Reset value
USART Status Register (USART_SR)	0x00	0x0000 00C0
USART Data Register (USART_DR)	0x04	0x0000 01FF
USART Baud Rate Register (USART_BRR)	0x08	0x0000 FF00 (USART_5/10) 0x0000 FFFF (USART_1/2/3/4/6/7/8/9)
USART Control Register1 (USART_CR1)	0x0C	0x8000 0000
USART Control Register2 (USART_CR2)	0x10	0x0000 0600
USART Control Register3 (USART_CR3)	0x14	0x0000 0000
USART Prescaler Register (USART_PR)	0x18	0x0000 0000
USART LIN Baudrate Measuring Counter (USART_LBMC)	0x1C	0x0000 0000

29.5.1 USART status register (USART_SR)

USART Status Register

offset address: 0x00

Reset value: 0x000000C0

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
-	-	-	-	-	LBD	WKUP	RTOF	TXE	TC	RXNE	BE	ORE	-	FE	PE

Bit	Symbol	Bit name	Description	Read and write
b31~b17	Reserved	-	Read as "0", write as "0"	R/W
b16	MPB	Multiprocessor position	Multiprocessor bit flag 0: Current received data is communication data 1: Current received data is ID Note: MPB bit is only valid in multiprocessor mode	R
b15~b11	Reserved	-	Read as "0", write as "0"	R/W
b10	LBD	LIN break field detection flag	LIN break field detection flag (only valid for USART_5/USART_10) 0: LIN Break field not detected 1: LIN Break field detected LBD set condition When CR2.WKUPE=0, it is detected that the low level width of the RX line is greater than or equal to the width set by the CR2.LBDL register, and the break delimiter is detected clear condition Control register CR1.CLBD bit write 1 Note: RE=0 does not reset the LBD bit	R
b9	WKUP	LIN Wakeup signal flag	LIN Wakeup signal detection flag (only valid for USART_5/USART_10) 0: Wakeup signal not detected 1: Wakeup signal detected WKUP set condition When CR2.WKUPE=1, it is detected that the low level width of the RX line is greater than or equal to 2.5 bits of data width (130uS at 19.2Kbps) clear condition Control register CR1.CWKUP bit write 1 Note: RE=0 does not reset the WKUP bit After detecting that the WKUP signal is 1, it means that the LIN node has exited from the sleep mode. After setting the register CR2.WKUPE=0, the subsequent break field detection and other operations are performed.	R
b8	RTOF	UART receiver TIMEOUT flag	UART receiver TIMEOUT flag (only valid for USART_1/USART_2/USART_6/USART_7) 0: No UART receiver TIMEOUT 1: UART receiver TIMEOUT occurs RTOF set condition •No new received data is detected after the set time has elapsed from the detection of the STOP bit of the last frame of data RTOF clear condition •Control register CR1.CRTOF bit write Note: RTOF is set by hardware and only when CR1.RE=1 and CR1.RTOE=1. When CR1.RE=0, the TIMEOUT function is valid, but RTOF is not set to 1.	R
b7	TXE	Transmit data register empty	Transmit data register empty flag The TXE bit is valid in UART/Clock Sync mode/LIN mode. 0: Data not transmitted to shift register, transmit data register is not empty 1: Data transfer to shift register, Transmit data register empty Note: The TXE bit is set and cleared by hardware. When the data is not transferred to the shift register, the hardware will clear TXE to 0. When the data is transferred to the shift register, the hardware will set TXE to 1.	R
b6	TC	Transmission complete	Transmission complete flag 0: Transmission is not complete 1: Transmission is complete	R

<p>UART mode, clock synchronization mode TC set condition <ul style="list-style-type: none"> ● TE = 0 Transmit Disabled ● When the last bit of a frame of data is sent, the value of the sent data register is not updated TC clear condition <ul style="list-style-type: none"> ● When TE = 1, write the transmitting data to the transmitting data register <p>Smartcard mode TC set condition <ul style="list-style-type: none"> ● TE = 0 Transmit Disabled ● After a certain time has elapsed after the last 1 byte of data is sent, FE=0 and the value of the transmit data register is not updated. The specific timing of TC setting is: 2.5-bit time elapses after the parity bit is sent TC clear condition <ul style="list-style-type: none"> ● When TE = 1, write the transmitting data to the transmitting data register <p>Note: When the TE bit changes from 0 to 1, TC remains at 1</p> </p></p>			
b5	RXNE	Received data register not empty	<p>Received data register not empty 0: Data is not received 1: Received data is ready to be read</p> <p>Note: The RXNE bit is set and cleared by hardware. When received data is ready to be read, the hardware will set RXNE to 1. After reading the received data, the hardware will clear RXNE to 0.</p>
b4	BE	LIN bus check error flag	<p>LIN bus check error flag 0: No bus error 1: A bus error occurred BE set condition <ul style="list-style-type: none"> ● When transmitting data, it is detected that the bus data is inconsistent with the transmitting data, and the detection position is 13/16 of each bit (16-bit oversampling) BE clear condition <ul style="list-style-type: none"> ● Control register CR1.CBE bit write 1 <p>Note: TE=0 does not reset the BE bit When BE=1, no new data is sent after the data being sent is complete. The software needs to clear the BE flag and restart the transmitting action.</p> </p>
b3	ORE	Reception overrun error	<p>Reception overrun error flag bit 0: No Overrun error 1: Overrun error is detected ORE set condition <ul style="list-style-type: none"> ● A new frame of data was received when the received data register was not read ORE clear condition <ul style="list-style-type: none"> ● Control register CR1.CORE bit write 1 <p>Note: RE=0 does not reset the ORE bit Data received before ORE = 1 is maintained, and data received when ORE = 1 is lost ORE = 1 does not continue to receive data and cannot transmit data in clock synchronization mode</p> </p>
b2	Reserved	-	Read as "0", write as "0" R/W
b1	FE	Framing error	<p>Framing error flag bit 0: No framing error is detected 1: Framing error is detected UART mode FE set condition <ul style="list-style-type: none"> ● The stop bit of the received data frame is low, and only the first stop bit is checked in the case of two stop bits. FE clear condition <ul style="list-style-type: none"> ● Control register CR1.CFE bit write 1 <p>Note: In UART mode, RE=0 does not reset the FE bit Data received at FE = 1 will be retained but RI interrupt will not occur and cannot continue to receive data after FE = 1 smartcard mode FE set condition <ul style="list-style-type: none"> ● Sampling to low level error signal flag FE clear condition <ul style="list-style-type: none"> ● Control register CLR.CFE bit write 1 <p>Note: In smartcard mode, RE=0 does not reset the FE bit</p> </p></p>
b0	PE	Parity error	<p>Parity error flag 0: No parity error is detected 1: Parity error is detected PE set condition <ul style="list-style-type: none"> ● When a parity error occurs in the received data PE clear condition <ul style="list-style-type: none"> ● Control register CR1.CPE bit write 1 <p>Note: RE=0 does not reset the PE bit</p> </p>

Data received at PE = 1 will be retained but RI interrupt will not occur
and cannot continue to receive data after PE = 1

29.5.2 USART data register (USART_DR)

USART Data Register

offset address: 0x04

Reset value: 0x000001FF

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]

Bit	Symbol	Bit name	Description	Read and write
b31~b25	Reserved	-	Read as "0", write as "0"	R/W
b24~b16	RDR[8:0]	Receive data register	Receive data register Note: The highest bit RDR[8] is only valid in UART mode and the data length is set to 9 bits	R
b15~b10	Reserved	-	Read as "0", write as "0"	R/W
b9	MPID	Multiprocessor mode ID bit	In multiprocessor mode, the select bit for transmitting communication data or ID 0: Transmit data 1: Transmit ID Note: MPID bit is only valid in multiprocessor mode, other modes must be set to reset value	R/W
b8~b0	TDR[8:0]	Transmit data register	Transmit data register Note: The highest bit TDR[8] is only valid in UART mode and the data length is set to 9 bits	R/W

29.5.3 USART baud rate register (USART_BRR)

USART Bit Rate Register

offset address: 0x08

Reset value: 0x0000FF00/0x0000FFFF (see register list for details)

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]				
Bit	Symbol	Bit name	Description										Read and write		
b31~b16	Reserved	-	Read as "0", write as "0"										R/W		
b15~b8	DIV_Integer[7:0]	Integer frequency division register Note: DIV_Integer[7:0] can only be set when TE=0&RE=0 (transmit/receive disabled)	Integer frequency division register Note: DIV_Integer[7:0] can only be set when TE=0&RE=0 (transmit/receive disabled)										R/W		
b7	Reserved	-	The reset value needs to be written when writing										R/W		
b6~b0	DIV_Fraction[6:0]	Fractional divider register Note: DIV_Fraction[6:0] can only be set when TE=0&RE=0 (transmit/receive disabled), and the set value is valid only when FBME=1 This register is invalid when USART_5/USART_10 is read out as 0	Fractional divider register Note: DIV_Fraction[6:0] can only be set when TE=0&RE=0 (transmit/receive disabled), and the set value is valid only when FBME=1 This register is invalid when USART_5/USART_10 is read out as 0										R/W		

Table 29-11 Baud rate calculation formula (fractional baud rate is invalid FBME=0)

Mode	Formula for calculating baud rate	Error E(%) calculation formula
UART mode Multiprocessor mode	$B = \frac{C}{8 \times (2^{OVER8}) \times (DIV_Integer + 1)}$	$E(%) = \left\{ \frac{C}{(8 \times (2 - OVER8) \times (DIV_Integer + 1) \times B)} - 1 \right\} \times 100$
Clock synchronization mode	$B = \frac{C}{4 \times (DIV_Integer + 1)}$	-
smartcard mode	$B = \frac{C}{2 \times BCN \times (DIV_Integer + 1)}$	$E(%) = \left\{ \frac{C}{(2 \times BCN \times (DIV_Integer + 1) \times B)} - 1 \right\} \times 100$

B: Baud rate, Unit: Mbps

C: The clock set by PR.PSC[1:0] bits, Unit: MHz

BCN: CR3.BCN register setting value

Table 29-12 Baud rate calculation formula (fractional baud rate valid FBME=1)

Pattern	Formula for calculating baud rate	Error E(%) calculation formula
UART mode Multiprocessor mode	$B = \frac{C \times (128 + DIV_Fraction)}{8 \times (2^{OVER8}) \times (DIV_Integer + 1) \times 256}$	$E(%) = \left\{ \frac{C \times (128 + DIV_Fraction)}{(8 \times (2 - OVER8) \times (DIV_Integer + 1) \times 256 \times B)} - 1 \right\} \times 100$
Clock synchronization mode	$B = \frac{C \times (128 + DIV_Fraction)}{4 \times (DIV_Integer + 1) \times 256}$	
smartcard mode	$B = \frac{C \times (128 + DIV_Fraction)}{2 \times BCN \times (DIV_Integer + 1) \times 256}$	$E(%) = \left\{ \frac{C \times (128 + DIV_Fraction)}{(2 \times BCN \times (DIV_Integer + 1) \times 256 \times B)} - 1 \right\} \times 100$

B: Baud rate, Unit: Mbps

C: The clock set by PR.PSC[1:0] bits, Unit: MHz

BCN: CR3.BCN register setting value

29.5.4 USART control register1 (USART_CR1)

USART Control Register 1

offset address: 0x0C

Reset value: 0x80000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
SBS	NFE	FBME	ML	-	-	-	MS	CLB D	CWK UP	CBE	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	-	TXEIE	TCIE	RIE	SLME	TE	RE	RTOIE	RTOE

Bit	Symbol	Bit name	Description	Read and write
b31	SBS	Start bit detection mode for receiving data in UART Mode	Start bit detection mode for receiving data in UART Mode 0: Start bit detection mode is RX pin low level 1: Start bit detection method is RX pin falling edge Note: The SBS bit must hold the reset value when not in UART mode SBS bits can only be set at TE = 0 & RE = 0 (Transmit/Receive Disabled)	R/W
b30	NFE	Digital filter enable bit	Digital filter enable bit 0: Disable digital filtering 1: Enable data filtering Note: NFE bit must hold reset value when not in UART mode NFE bits can only be set at TE = 0 & RE = 0 (Transmit/Receive Disabled)	R/W
b29	FBME	Fractional baud rate function enabled	Fractional baud rate function enable bit 0: Disable 1: Enable Note: The FBME bit can only be set when TE=0&RE=0 (transmit/receive disabled) This register is invalid when USART_5/USART_10 is read out as 0	R/W
b28	ML	MSB/LSB selection bit	In UART mode/clock synchronization mode/smartcard mode, MSB/LSB mode selection bit 0: LSB mode 1: MSB mode Note: The ML bit can only be set when TE=0&RE=0 (transmit/receive disabled)	R/W
b27~b25	Reserved	-	Read as "0", write as "0"	R/W
b24	MS	Communication mode selection bit	Communication mode selection bit 0: UART mode 1: Clock synchronization mode Note: The MS bit can only be set when TE=0&RE=0 (transmit/receive disabled), and the smartcard mode MS needs to be written to the reset value	R/W
b23	CLBD	LBD clear bit	LBD clear bit (only valid for USART_5/USART_10) 0: do not clear the LBD flag 1: clear the LBD flag Note: Write 1 to the CLBD bit to clear the LBD flag, and return 0 when read	R/W
b22	CWKUP	WKUP clear bit	WKUP clear bit (only valid for USART_5/USART_10) 0: do not clear the WKUP flag 1: clear the WKUP flag Note: Write 1 to the CKWUP bit to clear the WKUP flag, and return 0 when read	R/W
b21	CBE	BE clear bit	BE clear bit (only valid for USART_5/USART_10) 0: do not clear the BE flag 1: clear the BE flag Note: Write 1 to CBE bit to clear the BE flag, and return 0 when read	R/W
b20	CRTOF	RTOF clear bit	RTOF clear bit (only valid for USART_1/USART_2/USART_6/USART_7) 0: do not clear RTOF flag 1: clear RTOF flag Note: Write 1 to the CRTOF bit to clear the RTOF flag, and return 0 when read	R/W
b19	CORE	ORE clear bit	ORE clear bit 0: do not clear ORE flag 1: clear ORE flag Note: Write 1 to the CORE bit clears the ORE flag, and returns 0 when	R/W

read				
b18	Reserved	-	Read as "0", write as "0"	R/W
b17	CFE	FE clear bit	FE clear bit 0: do not clear FE flag 1: clear FE flag Note: Write 1 to CFE bit to clear the FE flag, and return 0 when read	R/W
b16	CPE	PE clear bit	PE clear bit 0: do not clear PE Mark 1: Clear PE flag Note: Write 1 to CPE bit to clear the PE flag, and return 0 when read	R/W
b15	OVER8	UART over-sampling mode	UART over-sampling mode setting, i.e. the basic number of clocks during one-bit data transmission 0: 16 bits 1: 8 bits Note: The OVER8 bit must hold the reset value when not in UART mode OVER8 bits can only be set at TE = 0 & RE = 0 (Transmit/Receive Disabled)	R/W
b14~b13	Reserved	-	Read as "0", write as "0"	R/W
b12	M	Data length	In UART mode, transmit/receive data length 0: 8 bits 1: 9 bits Note: M bit must remain reset when not in UART mode M bits can only be set when TE = 0 & RE = 0 (Transmit/Receive Disabled)	R/W
b11	Reserved	-	Read as "0", write as "0"	R/W
b10	PCE	Parity enable bit	Parity enable bit in UART mode 0: No parity 1: parity enabled Note: The PCE bit must be 1 in smartcard mode, and the PCE bit must remain at the reset value in clock synchronous mode PCE bits can only be set at TE = 0 & RE = 0 (Transmit/Receive Disabled)	R/W
b9	PS	Parity bit	Parity selection bit in UART mode 0: Even parity 1: Odd Parity Note: The PS bit can only be set when TE=0&RE=0 (transmit/receive disabled), and the PS bit is only valid when PCE=1	R/W
b8	Reserved	-	Read as "0", write as "0"	R/W
b7	TXEIE	Transmit data register empty interrupt enable bit	Transmit data register empty interrupt enable bit 0: TI interrupt request is disabled, TI interrupt does not occur 1: TI interrupt request is enabled, TI interrupt occurs Note: Write TXEIE=1 when TE=0, you need to wait for the TI interrupt to occur when TE=1 Write TXEIE=1 when TE=1, you need to wait for SR.TC=1 to write	R/W
b6	TCIE	Transmission complete interrupt enable	Transmission complete interrupt enable 0: TCI interrupt request is disabled, TCI interrupt does not occur 1: TCI interrupt request enabled, TCI interrupt occurs	R/W
b5	RIE	Receive interrupt enable	Receive interrupt enable 0: Receive interrupt request is disabled, RI and EI interrupts do not occur 1: Receive interrupt request is enabled, RI and EI interrupts occur	R/W
b4	SLME	Silent mode enable bit	Silent mode enable bit when multiprocessor operation 0: Normal mode 1: Silent mode When SLME = 1, the communication data with MPB 0 will not be read from the receive shift register to the Receive data register and the error flags ORE and FE bit are not set. When receiving ID data with MPB 1, SLME automatically clears and starts normal data receiving action. Note: The SLME bit is only valid in the UART multiprocessor mode. In other modes, this bit must remain at the reset value.	R/W
b3	TE	Transmitter enable	Transmitter enable 0: Transmitter is disabled 1: Transmitter is enabled Note: In clock synchronization mode, the TE bit can only be written to 1 when TE=0&RE=0 (transmit/receive disabled).	R/W
b2	RE	Receiver enable	Receiver enable 0: Receiver is disabled 1: Receiver is enabled Note: In clock synchronization mode, the RE bit can only be written to 1 when TE=0&RE=0 (transmit/receive disabled)	R/W

b1	RTOIE	UART TIMEOUT interrupt enable bit	UART TIMEOUT interrupt enable bit (only valid for USART_1/USART_2/USART_6/USART_7) 0: UART TIMEOUT interrupt request is disabled, RTOI interrupt does not occur 1: UART TIMEOUT interrupt request is enabled, RTOI interrupt occurs	R/W
b0	RTOE	UART TIMEOUT function enable bit	UART TIMEOUT function enable bit (only valid for USART_1/USART_2/USART_6/USART_7) 0: UART TIMEOUT function disabled 1: UART TIMEOUT function enabled	R/W

29.5.5 USART control register2 (USART_CR2)

USART Control Register 2

offset address: 0x10

Reset value: 0x00000600

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	SBK M	SBK
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	LINE N	STOP	CLKC[1:0]	-	-	WKU PE	SBKL[1:0]	LBDL	LBDI E	BEE	BEIE	WKU PIE	MPE		

Bit	Symbol	Bit name	Description	Read and write
b31~b18	Reserved	-	Read as "0", write as "0"	R/W
b17	SBKM	Break field transmitting mode	Break field transmitting mode selection (only valid for USART_5/USART_10) 0: Write 1 to the SBK bit to automatically transmit the break field 1: After the SBK bit is written to 1, write 0x00 to the USARTn_DR.TDR register to start transmitting the break field Note: In LIN mode, the SBKM bit can only be written to 1 when TE=0&RE=0 (transmit/receive disabled)	R/W
b16	SBK	Break field transmit enable bit	Break field transmission enable bit (only valid for USART_5/USART_10) 0: No break field is sent 1: Set the transmission break field according to the SBKM bit SBK SBKM break field transmission 0 0 break field is not transmit 0 1 break field is not transmit 1 0 When SBK is written to 1, the break field is automatically sent. After the transmission is completed, SBK is automatically cleared. 1 1 Write data 0x00 to USARTn_DR.TDR, start transmitting break field, after transmitting, SBK is automatically cleared	R/W
b15	Reserved	-	Read as "0", write as "0"	R/W
b14	LINEN	LIN function enabled	In UART mode, LIN function enable bit (only valid for USART_5/USART_10) 0: Disable LIN function 1: Enable LIN function Note: The LINEN bit can only be set when TE=0&RE=0 (transmit/receive disabled)	R/W
b13	STOP	Stop bits	Stop bit length setting in UART mode 0: 1 stop bit 1: 2 stop bit Note: The STOP bit must hold the reset value when not in UART mode The STOP bit can only be set at TE = 0 & RE = 0 (Transmit/Receive Disabled)	R/W
b12~b11	CLKC[1:0]	Clock control bit	UART mode 00b: The clock source is the clock generated by the internal baud rate generator. The clock is not output to the USARTn_CK pin. The USARTn_CK pin can be used as a common IO. 01b: The clock source is the clock generated by the internal baud rate generator, the clock is output to the USARTn_CK pin, and the output clock frequency is the same as the baud rate 10b or 11b: The clock source is an external input clock, and the frequency of the input clock is 16 times the baud rate (OVER8=0) or 8 times (OVER8=1) Clock synchronization mode 00b or 01b: The clock source is the clock generated by the internal baud rate generator, which is output to the USARTn_CK pin 10b or 11b: The clock source is an external input clock, and the frequency and baud rate of the input clock are the same smartcard 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 a normal IO. 01b: The clock source is the clock generated by the internal baud rate generator, and the clock is output to the CK pin	R/W

			10b or 11b: set ban Note: The CLKC[1:0] bits can only be set when TE=0&RE=0 (transmit/receive disabled)	
b10~b9	Reserved	-	Need to keep the reset value 11b, write 11b when writing	R/W
b8	WKUPE	LIN wake-up signal detection enable bit	LIN wake-up signal detection enable bit (only valid for USART_5/USART_10) 0: Disable wake-up signal detection 1: Enable wake-up signal detection Note: Only when the system needs wake-up signal detection, set WKUPE to 1. After detecting the wake-up signal, software needs to clear WKUPE.	R/W
b7~b6	SBKL[1:0]	LIN break field low level width setting bit	LIN break field low level width setting bit (only valid for USART_5/USART_10) 00b: 10 bits 01b: 11 bits 10b: 13 bits 11b: 14 bits Note: The SBKL bit must hold the reset value when not in UART-LIN mode The SBKL bit can only be set when TE=0&RE=0 (transmit/receive disabled)	R/W
b5	LBDL	LIN break field low level width detection threshold setting bit	LIN break field low level width detection threshold setting bit (only valid for USART_5/USART_10) 0: ≥ 10 digits 1: ≥ 11 digits Note: The LBDL bit must hold the reset value when not in UART-LIN mode The LBDL bit can only be set when TE=0&RE=0 (transmit/receive disabled)	R/W
b4	LBDIE	LIN Break field Detection Interrupt Enable Bit	LIN Break field detection interrupt enable bit (only valid for USART_5/USART_10) 0: LIN break field detection interrupt request is disabled, no interrupt will occur 1: LIN break field detection interrupt request is enabled and an interrupt occurs Note: LBDIE bit must hold reset value when not in UART-LIN mode	R/W
b3	BEE	LIN bus error detection function enable bit	LIN bus error detection function enable bit (only valid for USART_5/USART_10) 0: Disable LIN bus error detection function 1: Enable LIN bus error detection function Note: BEE bit must hold reset value when not in UART-LIN mode The BEE bit can only be set when TE=0&RE=0 (transmit/receive disabled)	R/W
b2	BEIE	LIN bus error interrupt enable bit	LIN bus error interrupt enable bit (only valid for USART_5/USART_10) 0: LIN bus error interrupt request is disabled, no interrupt will occur 1: LIN bus error interrupt request is enabled, an interrupt occurs Note: LBDIE bit must hold reset value when not in UART-LIN mode	R/W
b1	WKUPIE	LIN wake-up signal detection interrupt enable bit	LIN wake-up signal detection interrupt enable bit (only valid for USART_5/USART_10) 0: LIN wake-up signal detection interrupt request is disabled, no interrupt will occur 1: LIN wake-up signal detection interrupt request is enabled and an interrupt occurs Note: IN non-UART-LIN mode WKUPIE bit must hold reset value	R/W
b0	MPE	Multiprocessor function	In UART mode, the multiprocessor function enables the bit 0: Disable 1: Enable Note: MPE bit must hold reset value when not in UART mode MPE bits can only be set when TE = 0 & RE = 0 (Transmit/Receive Disabled)	R/W

29.5.6 USART control register3 (USART_CR3)

USART Control Register 3

offset address: 0x14

Reset value: 0x00000000

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
-	-	-	-	-	-	CTSE	RTSE	-	-	SCE N	LOO P	HDS EL	-	-	-

Bit	Symbol	Bit name	Description	Read and write
b31~b24	Reserved	-	Read as "0", write as "0"	R/W
b23~b21	BCN[2:0]	number of base clocks	In smartcard mode, the basic clock number during one-bit data transfer BCN[2:0] the number of basic clocks during one-bit data transmission 000b 32 001b 64 010b Prohibitions 011b 128 100b Prohibitions 101b 256 110b 372 111b Prohibitions	R/W
			Note: The BCN[2:0] bits must hold the reset value in non-smartcard mode BCN[2:0] bits can only be set when TE=0&RE=0 (transmit/receive disabled) USART_5/USART_10 does not support this function, read as 0	
b20~b10	Reserved	-	Read as "0", write as "0"	R/W
b9	CTSE	CTS function enable	CTS function enable 0: CTS function is disabled 1: CTS function is enabled Note: The CTSE bit can only be set when TE=0&RE=0 (transmit/receive disabled)	R/W
b8	RTSE	RTS function enable	RTS function enable 0: RTS function is disabled 1: RTS function is enabled Note: RTSE bit can only be set when TE=0&RE=0 (transmit/receive disabled)	R/W
b7~b6	Reserved	-	Read as "0", write as "0"	R/W
b5	SCEN	Smartcard Mode Enable Bit	Smartcard Mode Enable Bit 0: Disable smartcard mode 1: Enable smartcard mode Note: SCEN bit must remain at reset value in non-smartcard mode SCEN bit can only be set when TE=0&RE=0 (transmit/receive disabled) USART_5/USART_10 does not support this function, read as 0	R/W
b4	LOOP	LIN loopback mode enable bit	LIN loopback mode enable bit 0: Normal mode 1: Loopback mode Note: The LOOP bit can only be set when TE=0&RE=0 (transmit/receive disabled)	R/W
b3	HDSEL	UART Single Line half-duplex Mode Enable Bit	UART Single Line Half-duplex Mode Enable Bit 0: UART full duplex mode 1: UART half-duplex mode Note: The HDSEL bit can only be set when TE=0&RE=0 (transmit/receive disabled)	R/W
b2~b0	Reserved	-	Read as "0", write as "0"	R/W

29.5.7 USART prescaler register (USART_PR)

USART prescaler register

offset address: 0x18

Reset value: 0x00000000

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
-	-	-	-	-	-	-	-	-	-	-	-	LBMPSC[1:0]	PSC[1:0]		

Bit	Symbol	Bit name	Description	Read and write
b31~b2	Reserved	-	Read as "0", write as "0"	R/W
b3~b2	LBMPSC[1:0]	LIN baud rate measurement counter clock source selection	LIN baud rate measurement counter clock source selection (only valid for USART_5/USART_10) 00: PCLK 01: PCLK/2 10: PCLK/4 11: PCLK/8 Note: LBMPSC[1:0] bits can only be set when TE=0&RE=0 (transmit/receive disabled)	R/W
b1~b0	PSC[1:0]	Prescaler value	Prescaler value selection bit when the clock source is the clock generated by the internal clock source 00: PCLK 01: PCLK/4 10: PCLK/16 11: PCLK/64 Note: PSC[1:0] bits can only be set when TE=0&RE=0 (transmit/receive disabled)	R/W

29.5.8 USART LIN baudrate measuring counter (USART_LBMC)

USART LIN Baudrate Measuring Counter
offset address: 0x1C

Reset value: 0x00000000

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
LBMC[15:0]															

Bit	Symbol	Bit name	Description	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15~b0	LBMC[15:0]	LIN baud rate measurement result counter	LIN baud rate measurement result counter (only valid for USART_5/USART_10) The LIN slave node uses the clock source selected by the USARTn_PR.LBMPSC register to measure the frequency of the synchronization field sent by the master node. The count clock frequency selected by USARTn_PR.LBMPSC is divided by the value of LBMC to obtain the measured baud rate of LIN communication. Note: The value of LBMC is only meaningful after the synchronization field is received.	R

29.5.9 USART1 filter control register (USART1_NFC)

USART1 Noise Filtering Control Counter
address: 0x4005541C

Reset value: 0x00000000

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
-	-	-	-	-	-	-	-	-	-	-	-	-	USA RT1_ NFE	USASRT1_NFS [1:0]	

Bit	Symbol	Bit name	Description	Read and write
b31~b3	Reserved	-	Read as "0", write as "0"	R/W
b2	USART1_NFE	USART1 Filter Enable	USART_1 filter enable This register is used to control the switching of the analog filter on the RX line in STOP mode 0: Analog filter off 1: Analog filter is on, please refer to the setting of USART1_NFS[1:0] for the filter range	R/W
b1~b0	USART1_NFS	USART1 filter selection	USART_1 filter selection register This bits is used to control the filtering range of the analog filter on the RX line in STOP mode 00b: Filter width gear 1 01b: Filter width gear 2 10b: Filter width gear 3 11b: Filter width gear 4 For the specific value of each gear, please refer to the chapter "Electrical Characteristics of RX Filter Characteristics in USART1 STOP Mode".	R/W

29.6 Precautions for use

29.6.1 UART considerations

Transmitter

In UART mode, when the transmitter is disabled (USARTn_CR1.TE = 0), then the TX pin can be used as a common IO and the output value and direction can be set. If output is 0, frame errors are generated by the receiver, which interrupts the data transfer. If output 1, the receiver cannot detect the start bit to start the data transfer.

Receiver

When a frame error occurs in UART mode, the software detects whether the subsequent RX line is at a low level, thus determining whether the transmitter wants to interrupt the transmission.

If the receiving data start bit detection mode is low level detection, the receiving error continues to receive all low level data after clearing the error flag, and the receiving error will occur again.

29.6.2 Clock synchronization mode considerations

When using an external input clock to transmit data, the update of USARTn_DR.TDR needs to be completed before the clock is entered. After data is written, at least one bit of data time needs to be waited before the clock input.

When data is sent consecutively, the next frame of data needs to be updated before the last bit of the current frame is sent.

29.6.3 Other considerations

In order to prevent the TX communication line Hi-Z from being sent in the Disable state, the following methods can be used:

- Communication line pull-up
- At the end of transmitting data, before USARTn_CR1.TE=0, set the TX pin to normal IO output
- Before transmitting data, after USARTn_CR1.TE=1, set IO to TX function

30 Inter-integrated circuit bus (I2C)

30.1 Introduction

I2C (Integrated Circuit Bus) controller is used as the interface between the microcontroller and the I2C serial bus. The I2C controller implements the whole I2C-BUS protocol including multi-master, arbitration and so on. Standard mode and fast mode are supported. SMBus is also supported.

I2C main features:

- Both I2C bus and SMBus are supported
- Both master and slave functions are supported
- Setup time, Hold time and Idle time relative to the transfer baudrate are guaranteed
- Supports standard-mode (up to 100 Kbps), fast-mode (up to 400 Kbps)
- The start condition, restart condition, and stop condition are automatically generated, and the start condition, restart condition, and stop condition of the bus can be detected.
- Support 7-bit and 10-bit address format.
- General Call address, SMBus master address, SMBus device default address, SMBus alarm address can be detected
- Acknowledge bit detection and generation are supported in master and slave mode
- Clock stretching supported
- Arbitration supported
- SCL timeout detection
- Analog Noise Filter and programmable digital noise filter supported
- Communication error, receive buffer full, transmit buffer empty, frame transmit done, and address match interrupts are supported

30.2 I2C System Block Diagram

30.2.1 System block diagram

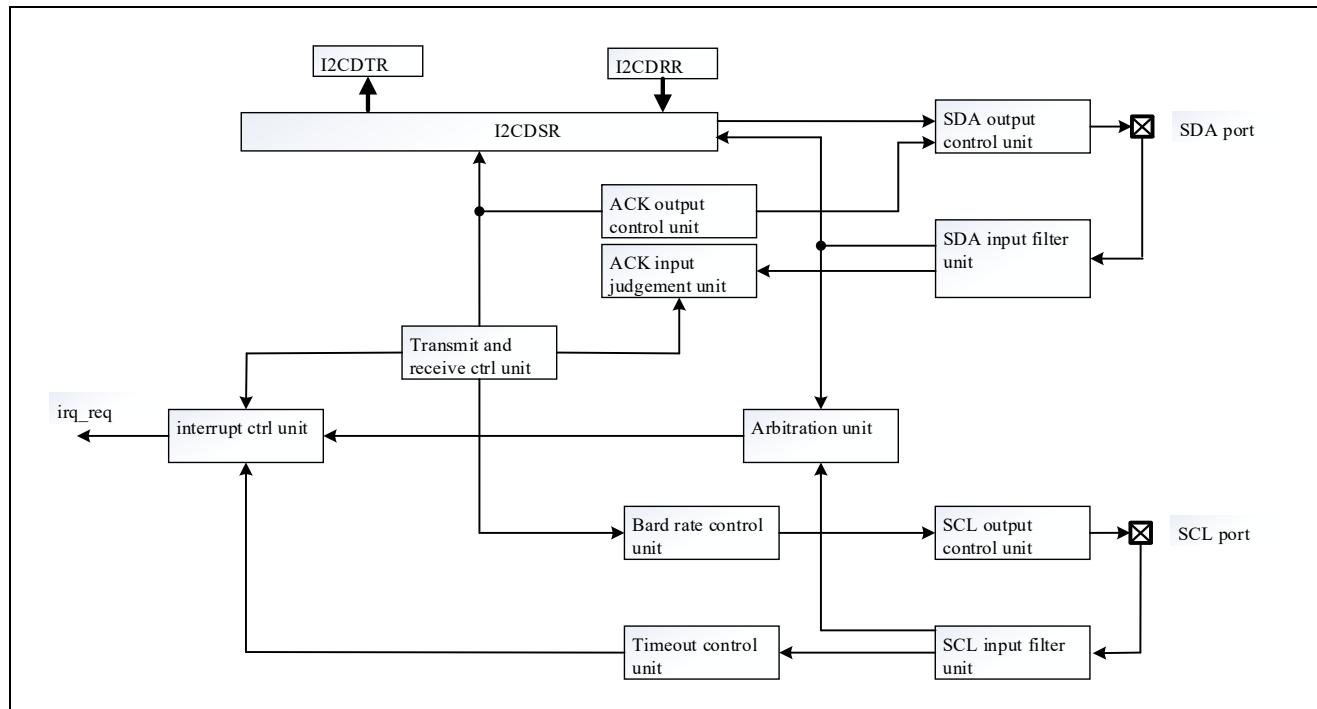


Figure 30-1 I2C System Block Diagram

30.2.2 Structural diagram

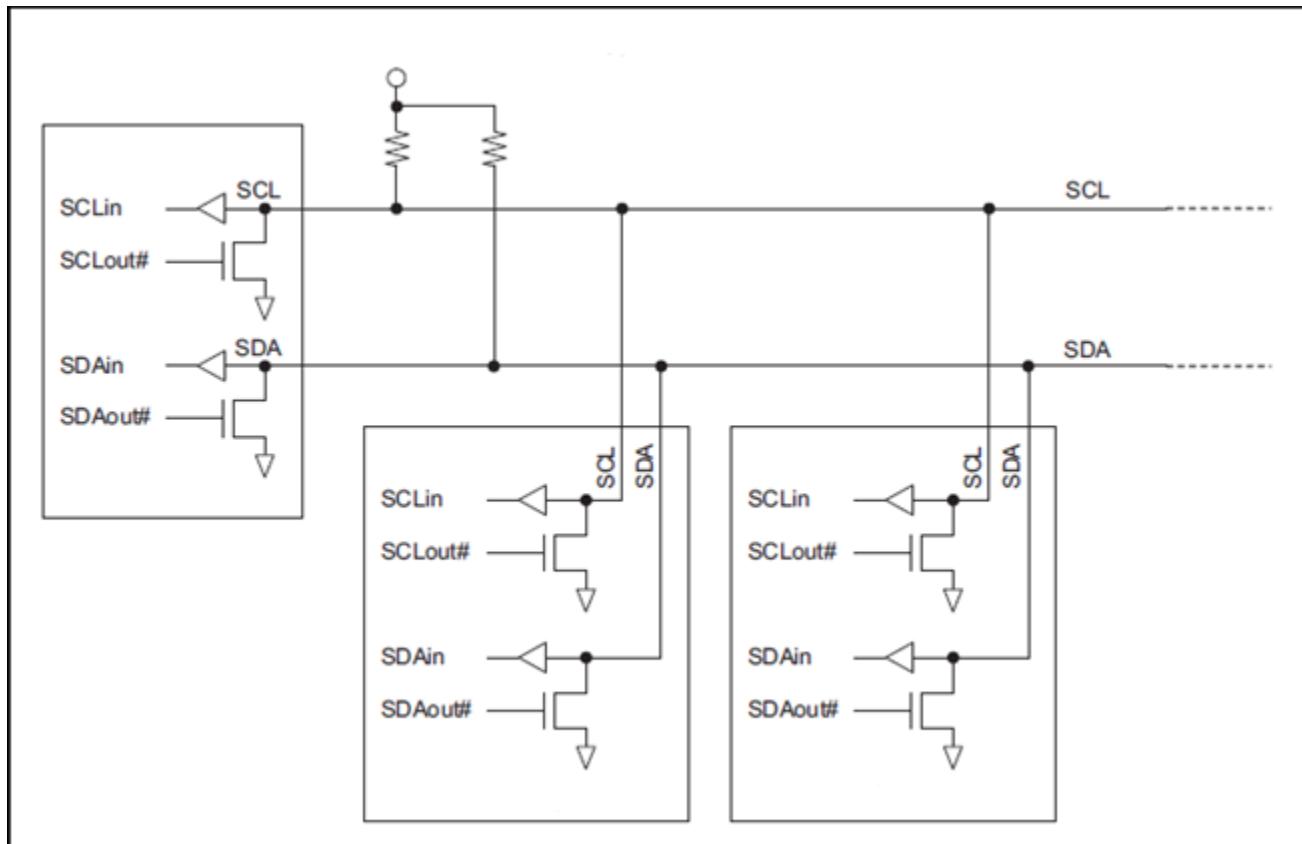


Figure 30-2 Structure example of I2C bus

Table 30-1 Input/output pins

Pin name	Input/Output	Function
SCL	Input/Output	Input/output pin of serial clock
SDA	Input/Output	Input/output pin of serial clock

When the I2C bus is selected, the SCL/SDA input level is Schmitt level (compatible with CMOS level).

When selecting SMBus, set the SCL/SDA input level to CMOS level by using the PCRxy register (PCRxy.CINSEL) in the General IO(GPIO) section.

30.3 Action description

This section provides a description of the functionalities of the I²C module.

30.3.1 I²C protocol

I²C bus consists of a clock line (SCL) and a data line (SDA). All connected devices must be drain open output. The SCL and SDA lines require external pull-up resistors, the value of which depends on the system application

In general, a complete communication process consists of the following four parts:

1. Start condition
2. Address transfer
3. Data transfer
4. Stop condition

The following figure is the timing diagram of the I²C bus.

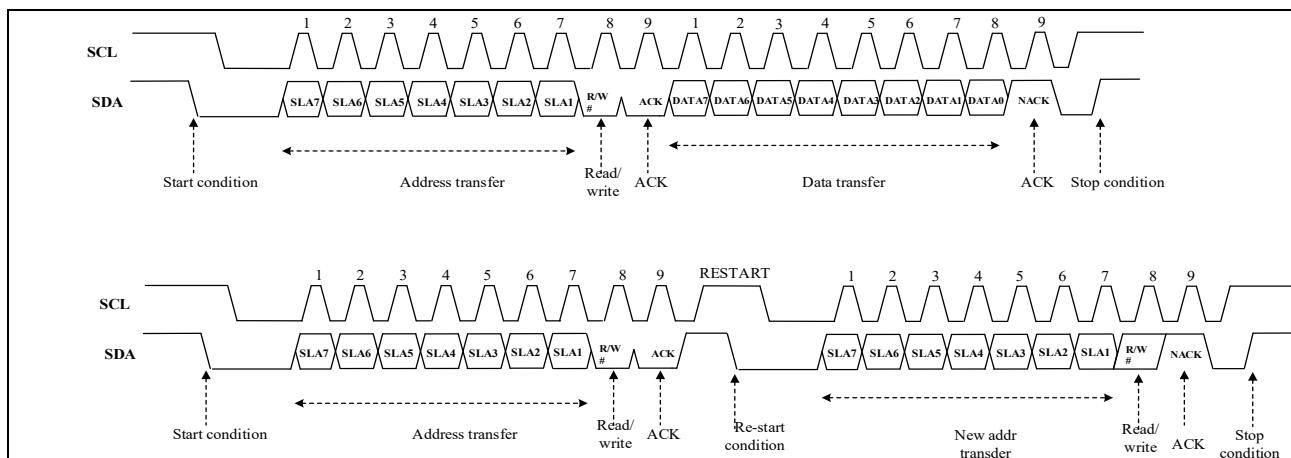


Figure 30-3 Timing diagram for the I²C bus

30.3.1.1 Start condition

When the master on the bus does not drive the bus, the bus enters the idle state. Both SCL and SDA are high. Devices on the bus in idle state can start the communication by sending the start condition.

When the I²C_SR.BUSY flag is "0", if the START bit is set to "1", a start condition will be generated. If a start condition is detected, the I²C_SR.BUSY flag and I²C_SR.STARTF flag are automatically set to "1" and the START bit is automatically set to "0". At this time, if the SDA value sent after START bit setting to "1" is the same as the SDA line, and the start condition is detected, it is considered that the start condition is correctly issued by the START bit. After the I²C_SR.MSL bit and I²C_SR.TRA bit are automatically set to "1" and then it changed to the master transmit mode. In addition, I²C_SR.TEMPTYF automatically becomes "1" because the TRA bit is "1". Next writing the slave address to the I²C_DTR register will send the slave address.

30.3.1.2 Address transfer

The frame after the start condition or restart condition is an address frame that specifies the object address for master communication. The specified slave is always valid until the stop condition is sent.

The upper 7 bits of the address frame are the slave addresses. The 8th bit of the address frame determines the direction of the data frame.

- The 7-bit addressing format is shown in the following figure [7-bit address format]
In master transmit mode, the 8th bit of the first address is set to '0'
In master receive mode, the 8th bit of the first address is set to '1'
- The 10-bit addressing format is shown in the following figure [10-bit address format]
In master transmit mode, the first frame sends header sequence (11110XX0, where XX represents the 2bits upper address of the 10-bit address) , and the second frame sends the 8bits low-order address.
In master receive mode, the first frame sends header sequence (11110XX0, where XX represents the 2bits upper address of the 10-bit address) , and the second frame sends the 8bits low-order address. Next, a restart condition is sent, and then a frame header sequence is sent (11110XX1, where XX represents the 2bits upper address of the 10-bit address).

7bit address format



10bit address format

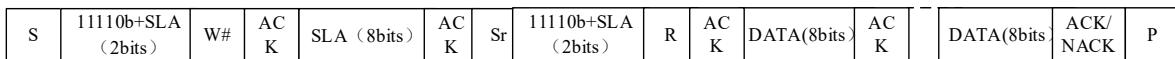


Figure 30-4 I2C bus data format

S: Indicates the start condition.

SLA: Indicates the slave address.

R/W#: Indicates the direction of the frame. when R/W # is "1" the data transmission is from slave to master; When R/W # is "0", the data transmission is from the master to the slave.

Sr: Indicates a restart condition.

DATA: Represents data sent and received

P: Indicates a stop condition.

30.3.1.3 Data transfer

After the address match, the master or slave transmits the data one by one according to the direction defined by R/W.

All data transmitted after the address frame is regarded as a data frame. Even the lower 8-bit address in a 10-bit address format is regarded as a data frame.

The length of the data frame is 8 bits. SDA changes during the low level of SCL, SDA is held during the high level of SCL, and one bit shifted every SCL clock cycle. The 9th SCL clock after the data frame is an acknowledge bit and is a handshake signal to the transmitter.

If the slave receives data on the bus and does not respond to the master in the 9th clock cycle, the slave must send a NACK. If the master receives data on the bus, it sends a NACK in the 9th SCL clock cycle, and the slave stops sending data after receiving this NACK.

Whether the master or the slave sends NACK, the data transfer interrupts. The master can do any of the following:

- Send stop condition to release bus
- Send the restart condition to start a new communication.

Master transmit data

In master transmit mode, the master outputs SCL clock and transmit data, the slave receives data from the master and responds an acknowledge bit. The sequence of master sending data is shown in the following figure.

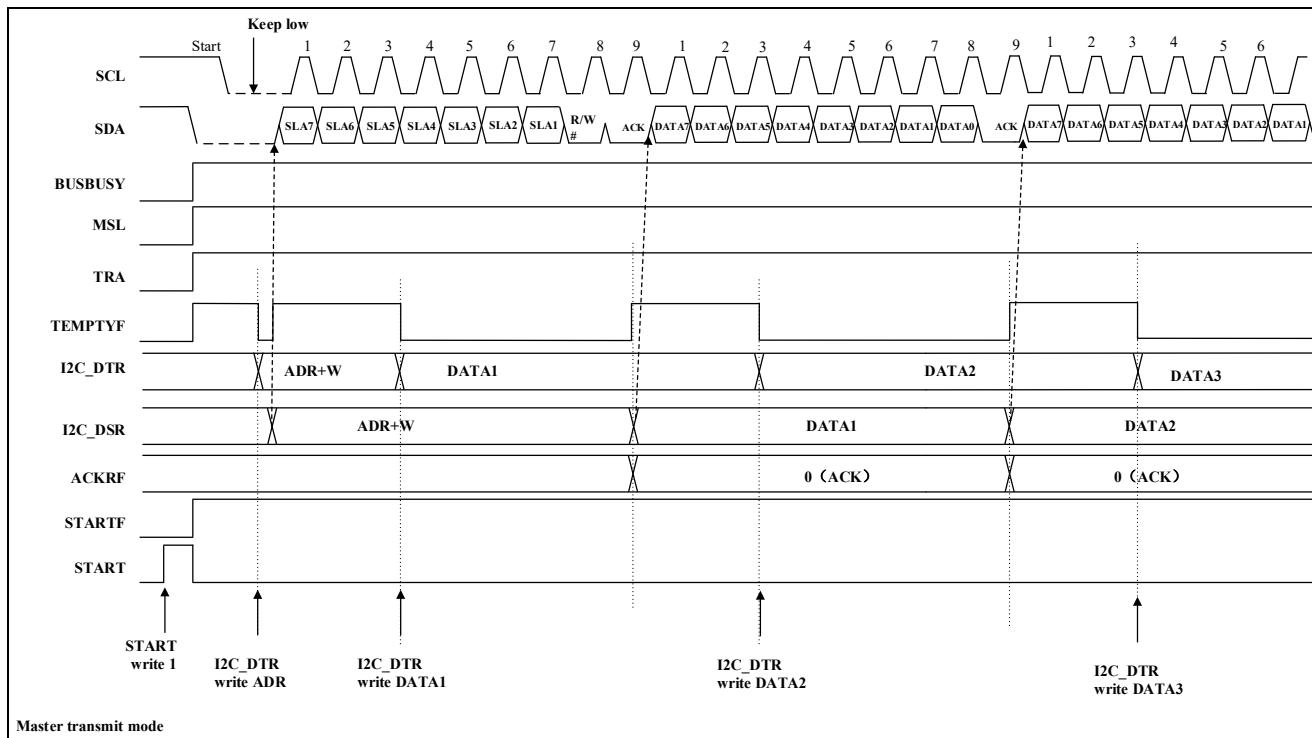


Figure 30-5 Sequence diagram of the master sending data in 7-bit address format

Master receive data

In Master receive mode, the master outputs SCL clocks, receives slave data, and responds acknowledge bits. The sequence of receiving data by the master is shown in the following figure.

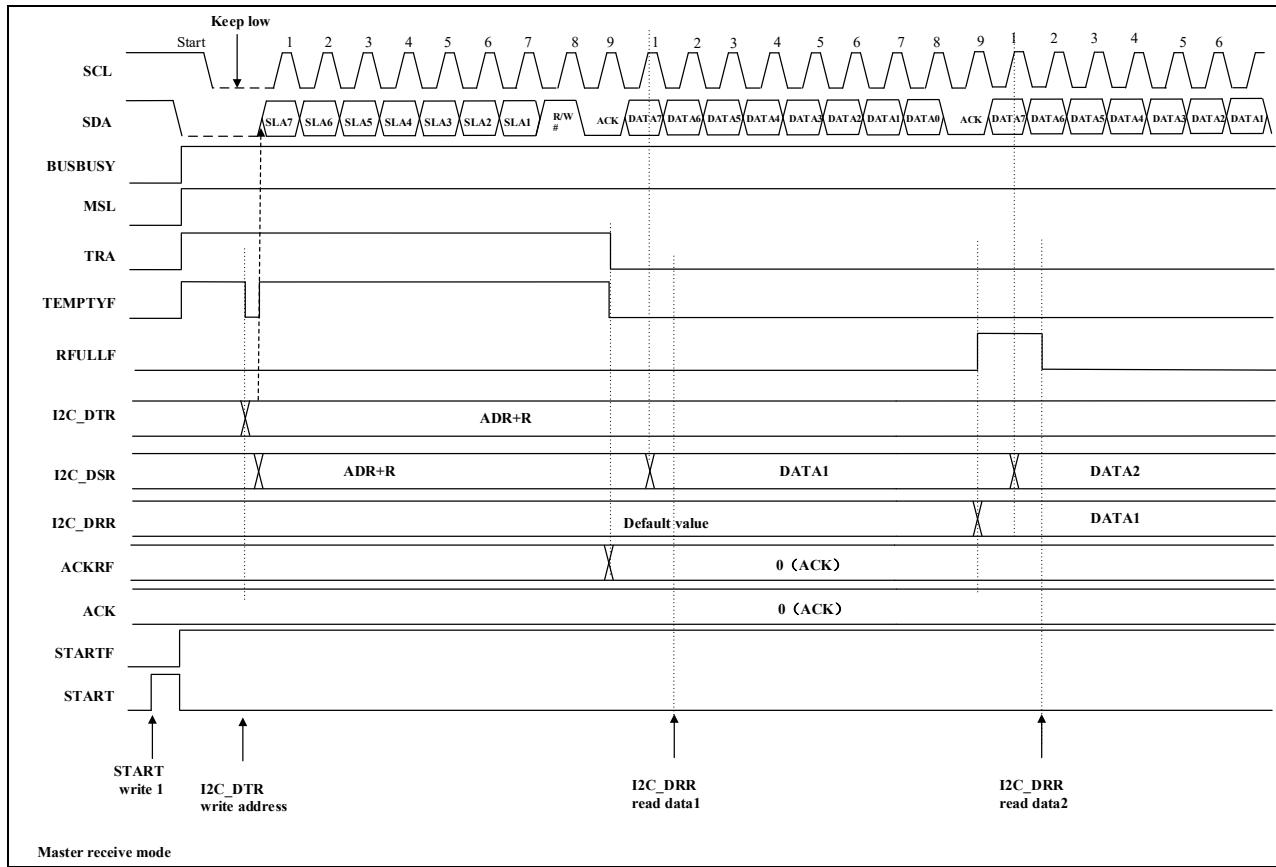


Figure 30-6 Timing diagram of master receiving data in 7-bit address format

Slave transmit data

In slave transmit mode, the slave receives SCL clock from the master. This product sends data for the slave and receives a reply from the master. The sequence of slave sending data is shown in the following figure.

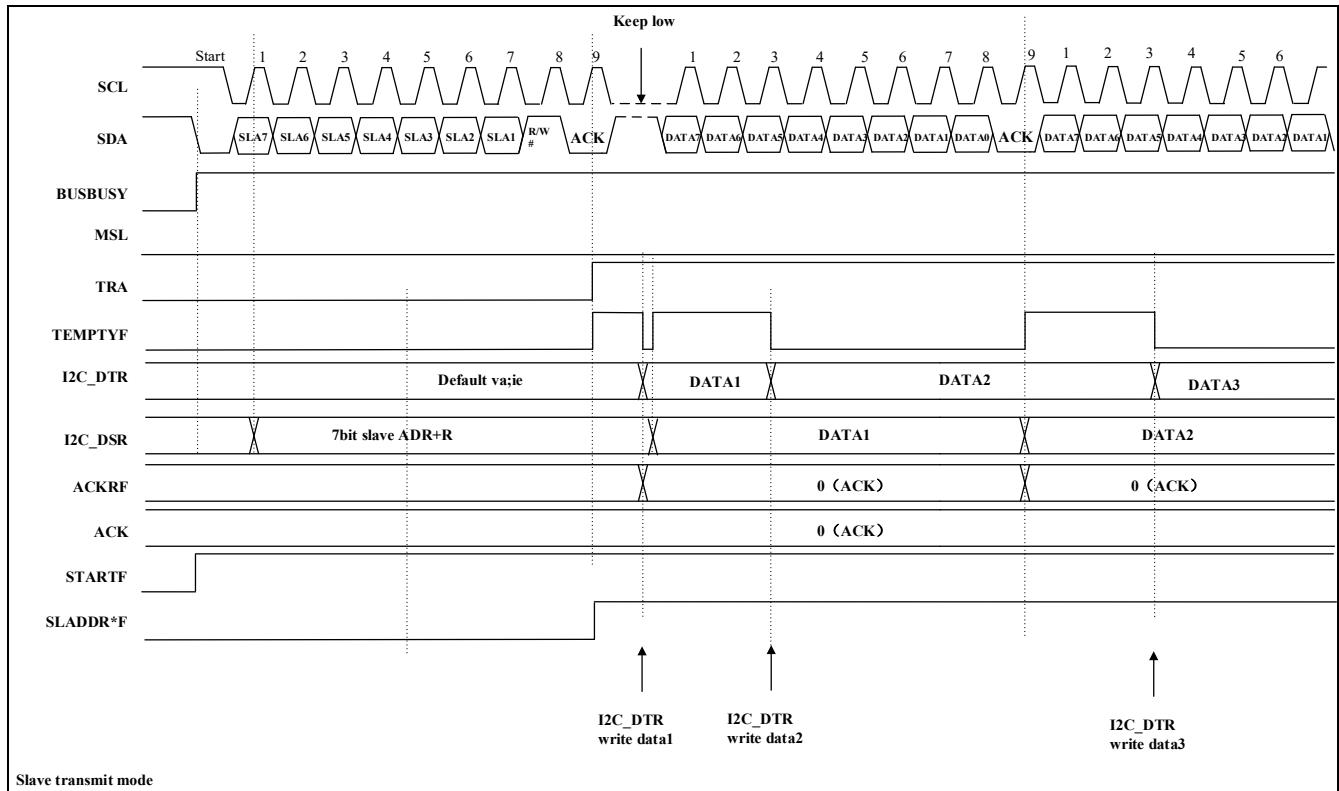


Figure 30-7 Slave Transmit Mode Timing Chart in 7-bit Address Format (Example)

Slave receive data

In slave receive mode, the slave receives the SCL clock and the data from the master, and responds the acknowledge bits to the master. The sequence of receiving data from the master is shown in the following figure.

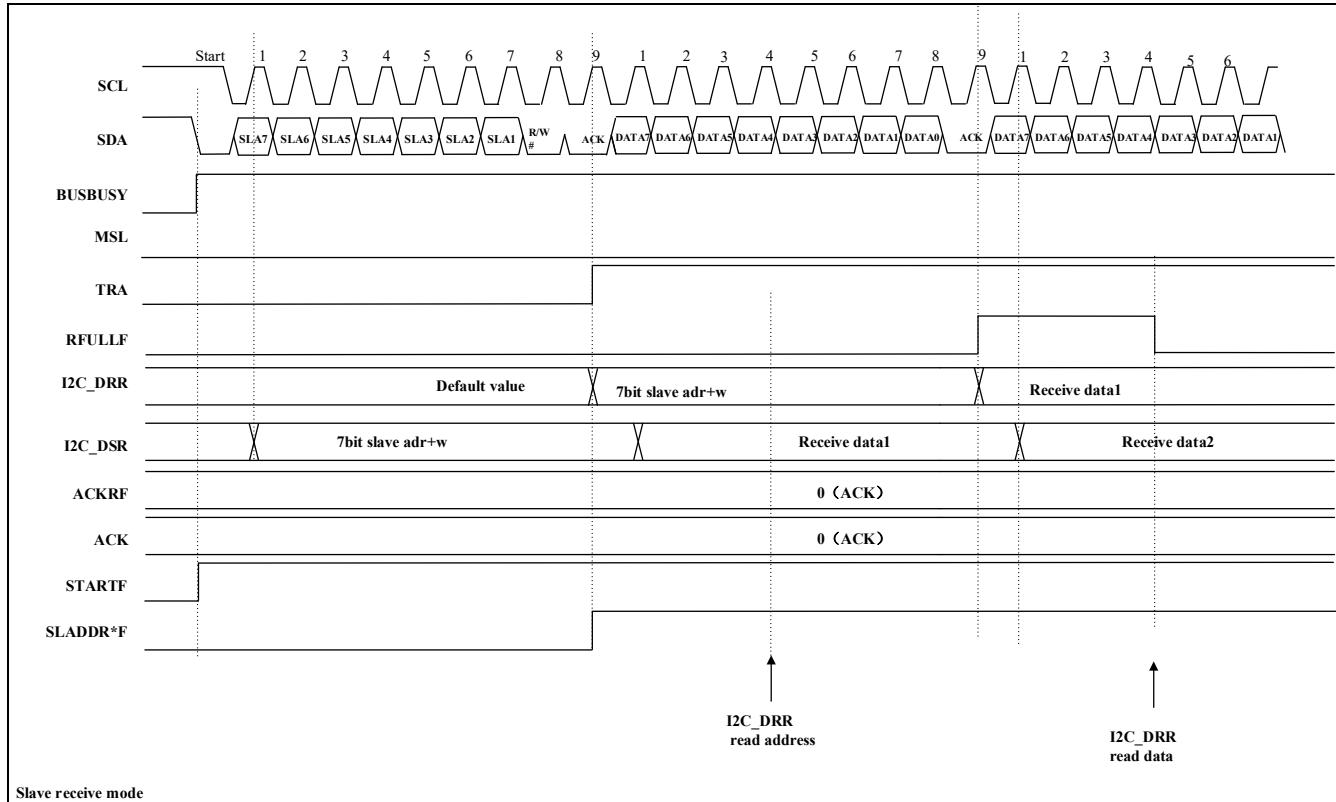


Figure 30-8 7-bit address format slave receiver mode timing diagram

30.3.1.4 Stop condition

Setting the I2C_CR1.STOP bit to "1" while the I2C_SR.BUSY flag and the I2C_SR.MSL bit are "1" will generate a STOP condition.

30.3.1.5 Restart condition

Setting the I2C_CR1.RESTART bit to "1" while the I2C_SR.BUSY flag and the I2C_SR.MSL bit are "1" will generate a Re-start condition.

By restarting the condition, the master can switch transmit/receive modes without releasing BUS authority. Communication can also be established with another slave without releasing BUS authority.

30.3.1.6 SCL clock synchronization

When using the I2C bus in the multi-master mode, the SCL clock may be conflicted due to competition with other masters. If the SCL clock conflicts, the master needs to synchronize with the SCL clock and synchronize the SCL clock bit by bit.

When the rising edge of the SCL is detected and the high level set by I2C_CCR.SHIGHW register is counting, If the SCL line is falling due to the SCL clock output from another master, the high-width increment count is aborted when the falling edge of the SCL line is detected, and the I2C_CCR.SLOWW is started while the SCL line is driven low. The set low-level width is counted up, and the low-level drive of the SCL line is ended when the count of the low-level width ends, and the SCL line is released.

Therefore, in case of SCL clock output conflicts, the high level width of SCL clock synchronizes with short clock, and the low level width synchronizes with long clock.

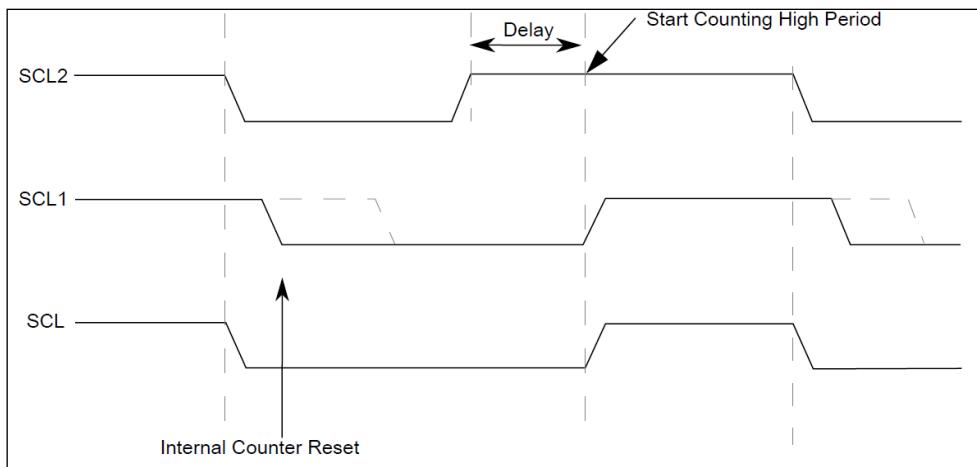


Figure 30-9 SCL synchronization timing

30.3.1.7 Arbitration

The I2C bus is a true multi-master bus, allowing multiple masters to connect

If two or more masters try to control the bus at the same time, the SCL clock synchronization process determines the bus clock. The low cycle of the bus clock depends on the longest low-level clock and the high cycle depends on the shortest high-level clock.

The results of arbitration are determined by the data collected while SCL is high.

If a high level is sent to SDA (the SDA pin is in a high impedance state) but a low level is detected at SDA, the arbitration fails. The I2C_SR.AROLF bit will be set to "1". If a master arbitration fails, it is immediately transferred to the slave receive mode. At this point, if the slave addresses, including general call addresses match, the slave mode continues to run.

30.3.1.8 Clock stretching

The handshake is realized by SCL clock synchronization mechanism during data transfer. After transmitting a frame of data (including ACK bit), the slave maintains the SCL clock line at low level. In this case, a low level of the SCL clock puts the master into a wait state until the slave releases the SCL.

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 (I2CDT register) is not written, the low level of the SCL line is automatically maintained during the 9th clock and the 1st clock interval of the next transfer, as shown in the following figure.

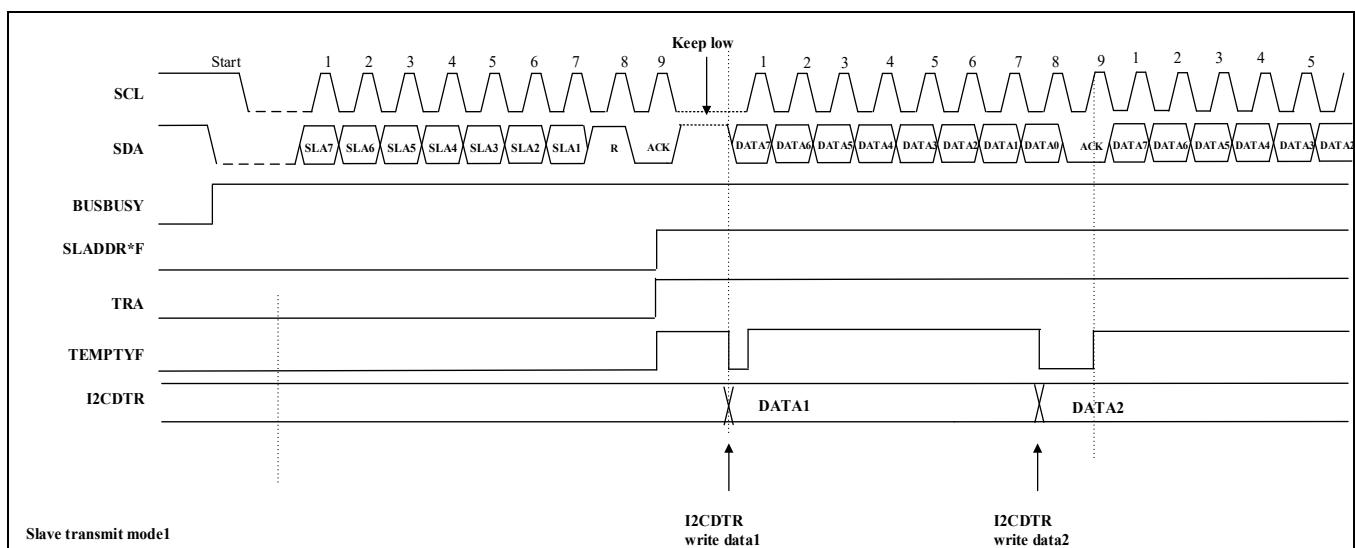


Figure 30-10 Slave sending timing diagram1

- 2) When the I2C_SR.NACKF or the I2C_SR.TENDF flag becomes "1" under the following condition1 the controller will keep the SCL line low after the 9th SCL falling edge. At this point, the communication must be terminated by reading the I2C_DRR register, thereby releasing the SCL line

Condition1:

After the I2C_SR.NACKF flag becomes "1" or the last transmit data is written to the I2C_DTR register, wait until the I2C_SR.TENDF flag becomes "1" while the I2C_SR.TEMPTYF flag is "1".

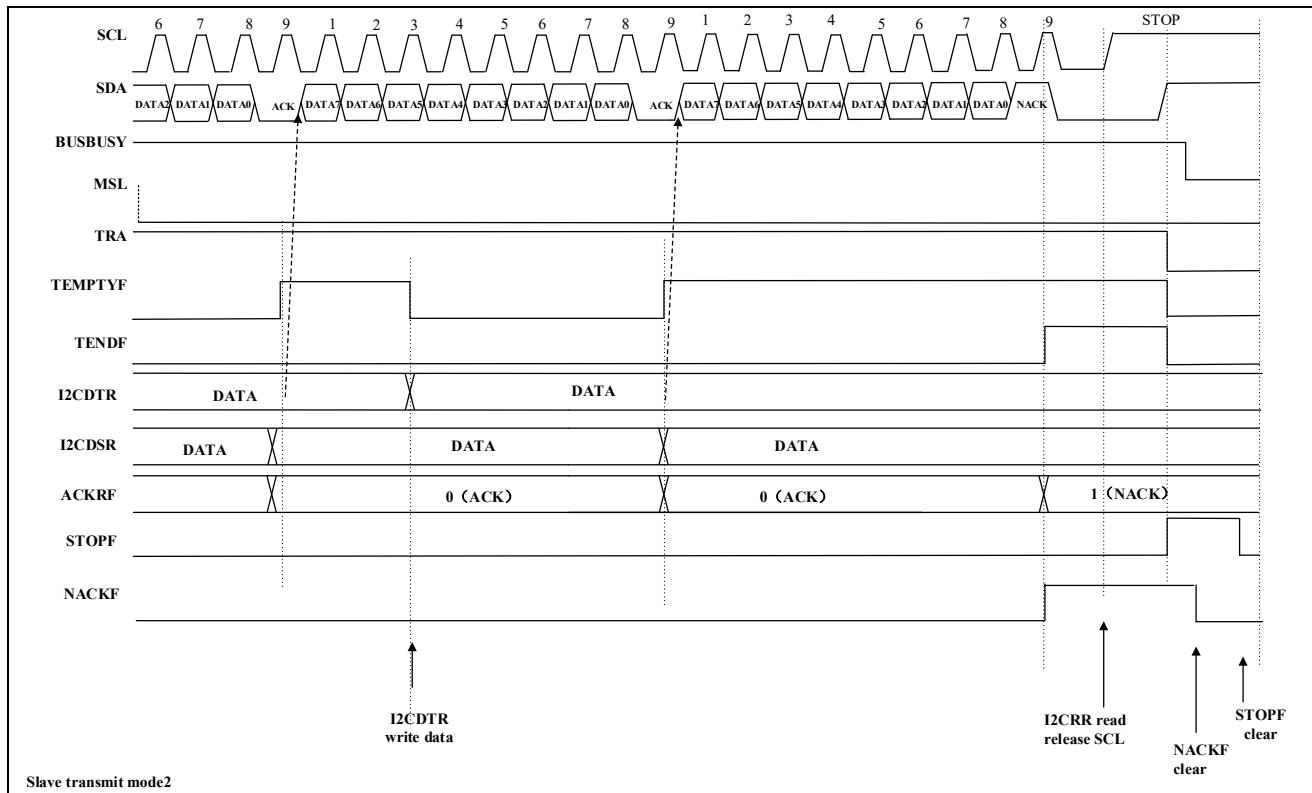


Figure 30-11 Slave sending timing diagram2

Slave Receive Mode

During the data reception process, if the current I2C_SR.RFULLF signal is '1', the I2C controller will automatically keep the low level of the SCL line between the 8th SCL and the 9th SCL of the current data reception process until the last received data in the I2C_DRR register is read out.

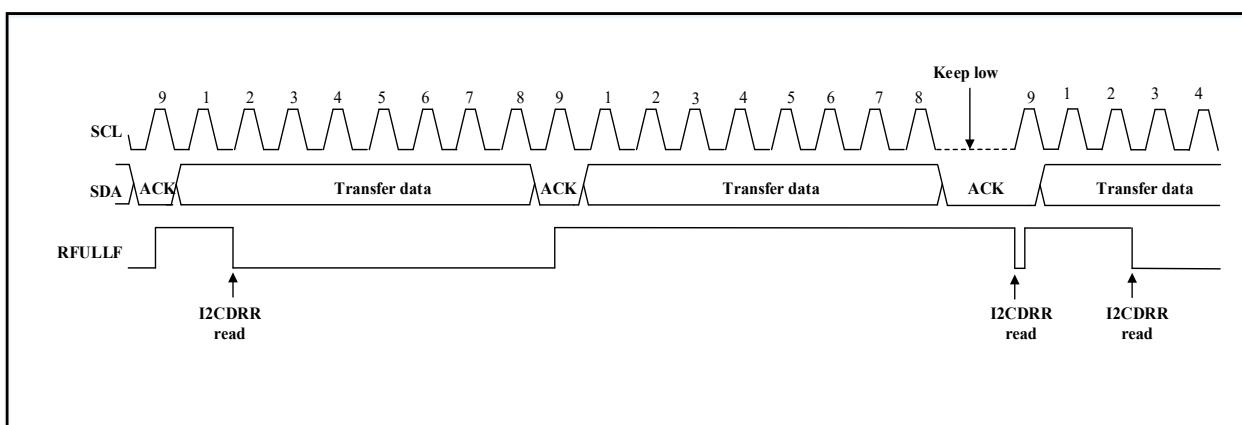


Figure 30-12 Slave Receive Timing Diagram

Fast ACK/NACK

In SMBus communication, the CPU first calculates the CRC then sends the packet error code (PEC) through the SMBus, the built-in CRC operator of the system can be used to calculate or verify the PEC code.

In the process of checking the PEC code, ACK or NACK is sent in the last byte according to whether it matches. This extra processing time requires keeping SCL low on the falling edge of the 8th clock of SCL when receiving the last byte to meet the software processing time. The software writes I2C_CR1.ACK bit to remove SCL low level. The function of fast ACK/NACK is controlled by the I2C_CR3.FACKEN bit, action sequence is shown in the following figure.

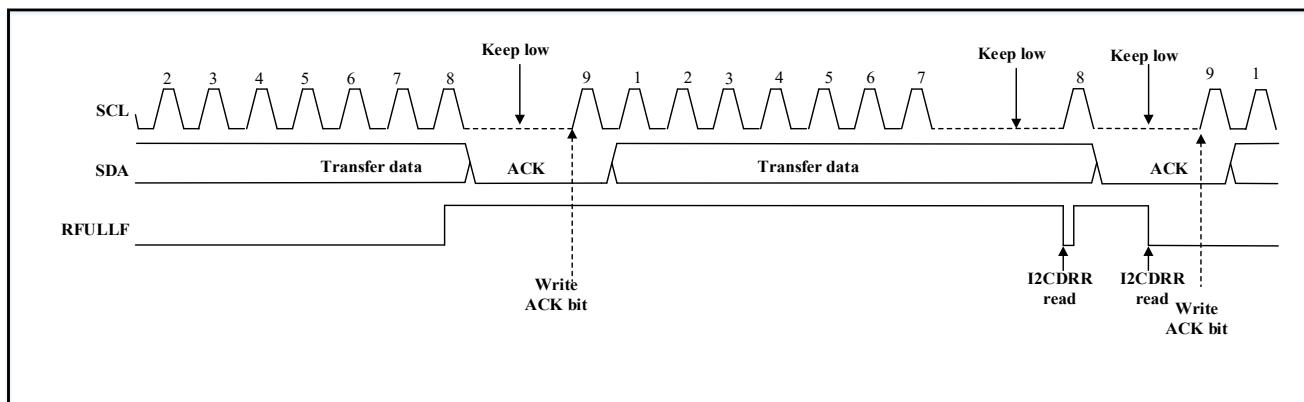


Figure 30-13 Fast ACK/NACK timing diagram

30.3.2 Address match

As a slave, two kinds of addresses other than general call address and master notification address can be set, and 7-bit address or 10-bit address format can be set for slave address.

30.3.2.1 Slave address match

This I2C bus can set 2 kinds of slave addresses, and there are corresponding slave address detection functions.

When I2C_SLR1.SLADDR1EN or I2C_SLR0.SLADDR0EN is set to "1", the I2C controller can detect whether the current received address is consistent with the slave address set by the I2C_SLR1 or I2C_SLR0 register. If the currently set slave address matches the received address, the I2C controller will set the corresponding I2C_SLR1.SLADDR1F or I2C_SLR0.SLADDR0F to "1" on the falling edge of the 9th clock of the SCL. The sequence of I2C_SLR1.SLADDR1F and I2C_SLR0.SLADDR0F flag changing to "1" is shown in the following figure.

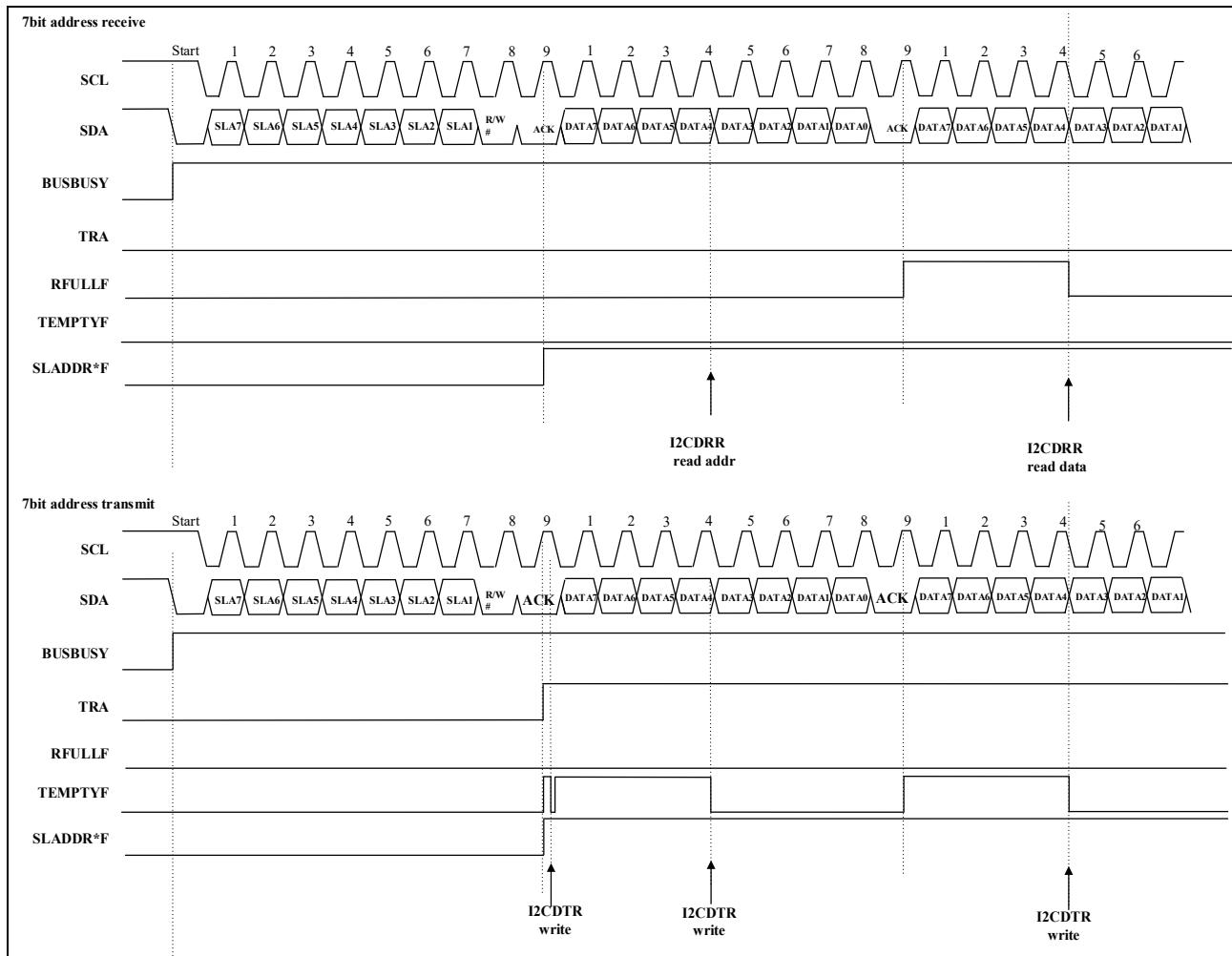


Figure 30-14 Timing for 7-bit address format

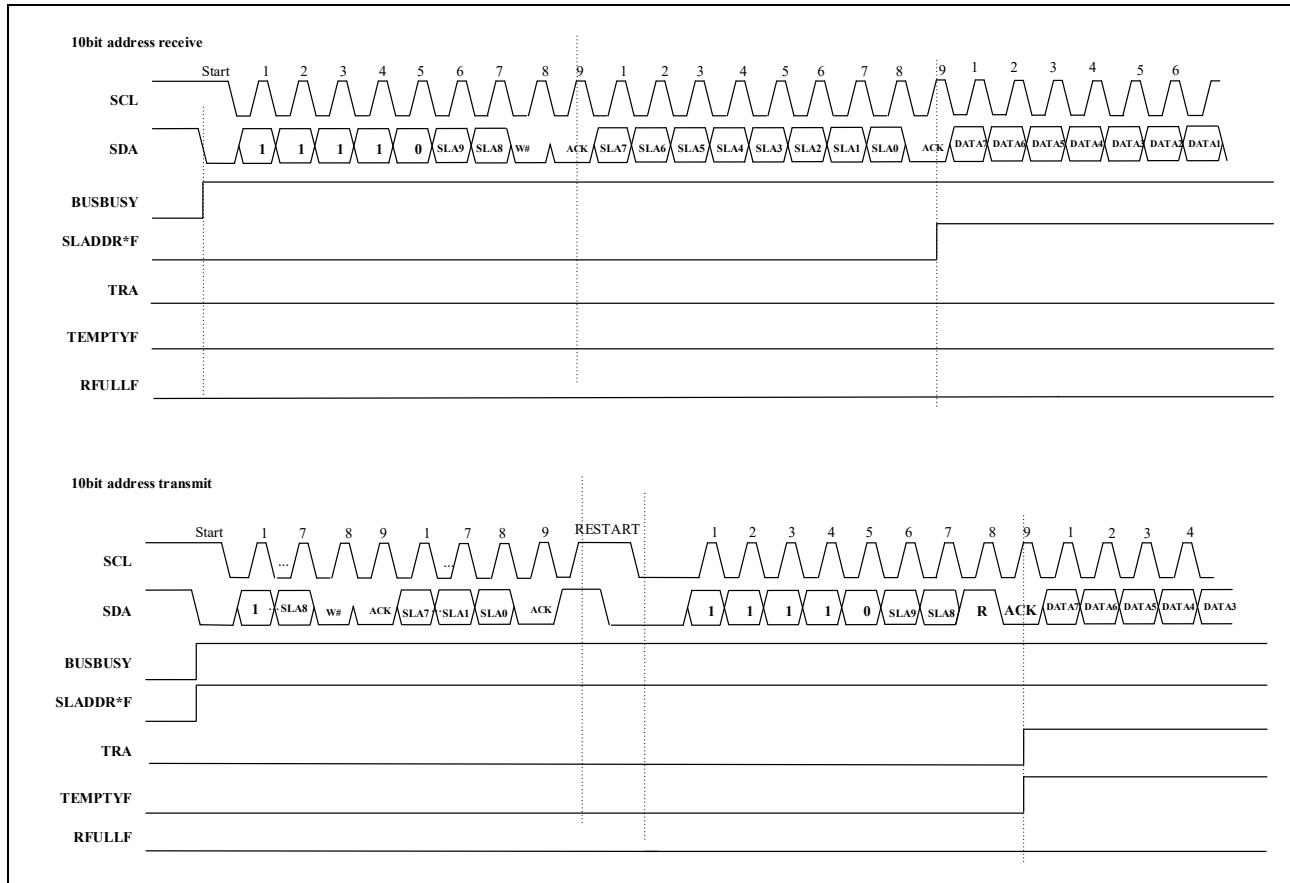


Figure 30-15 Timing for 10-bit address format

30.3.2.2 General call address match

When the I2C_CR1.GCEN bit is set to "1", the general call address (0000000b+0[W]) can be detected. But after the start condition or restart condition, the address (0000000b+1[R]) is considered as the slave address of All "0" and not the general call address.

If the general call address is matched, the I2C_SR.GENCALLF flag is set to "1" on the falling edge of the 9th clock of the SCL clock.

The operation after the general call address match is the same as the normal slave receiving operation.

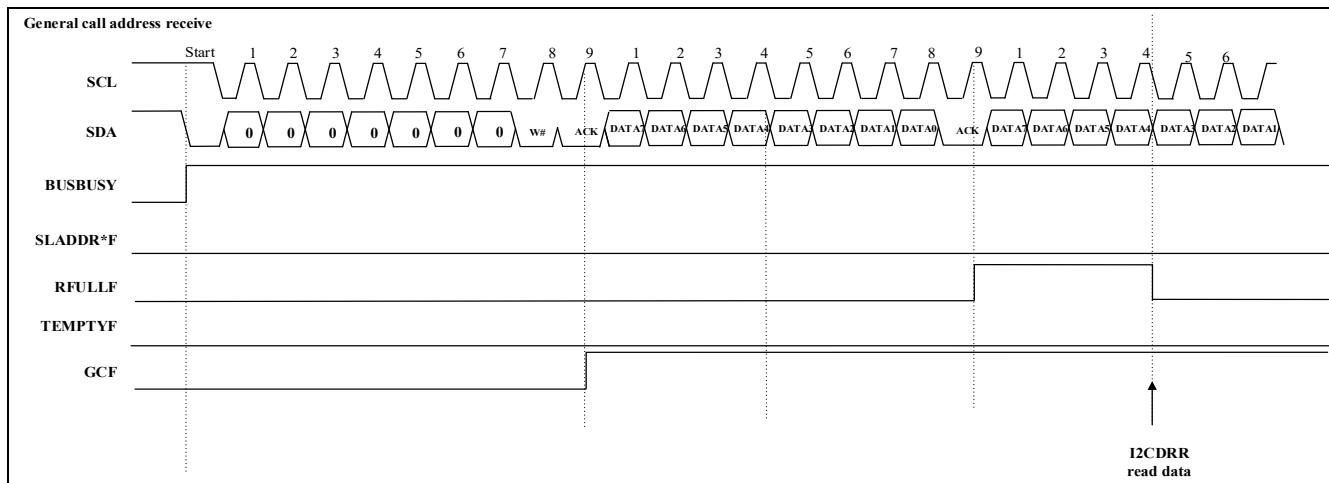


Figure 30-16 General call address receive timing diagram

30.3.2.3 SMBus Master Address Match

This I2C controller has the master address detection function when SMBus runs.

If the I2C_CR1.SMBHOSTEN bit is set to "1" when the I2C_CR1.SMBUS bit is "1", the master address (0001000b) can be detected in the slave receive mode (I2C_CR1.MSL bit TRA bit is "00b").

The I2C_SR.SMBHOSTF flag is set to '1' on the falling edge of the 9th clock of the SCL clock when the SMBus master address is detected.

Even if the bit following the SMBus master address (0001000b) is an Rd bit (R/W # bit receives "1"), the SMBus master address can also be detected. The running of SMBus master address detection is the same as that of normal slave mode.

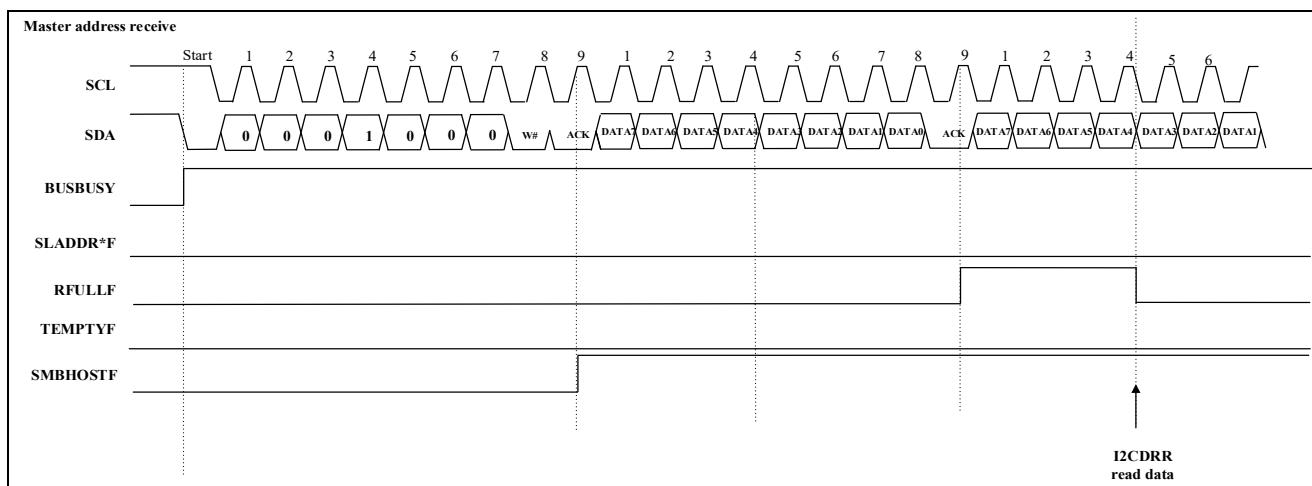


Figure 30-17 SMBus master address receive timing diagram

30.3.2.4 SMBus Alarm Response Address Matching

This I2C controller has the function of alarm response address detection when SMBus is running. If the I2C_CR1.SMBARLERTEN bit is set to "1" when the I2C_CR1.SMBUS bit is "1", the SMBus alarm response address (0001 100b) can be detected in the slave receive mode (I2C_CR1.MSL bit TRA bit is "00b").

If the SMBus alarm response address is detected, the I2C_SR.SMBARLERTF flag is set to "1" on the falling edge of the 9thclock of the SCL clock.

The operation after SMBus alarm response address detection is the same as that of normal slave mode.

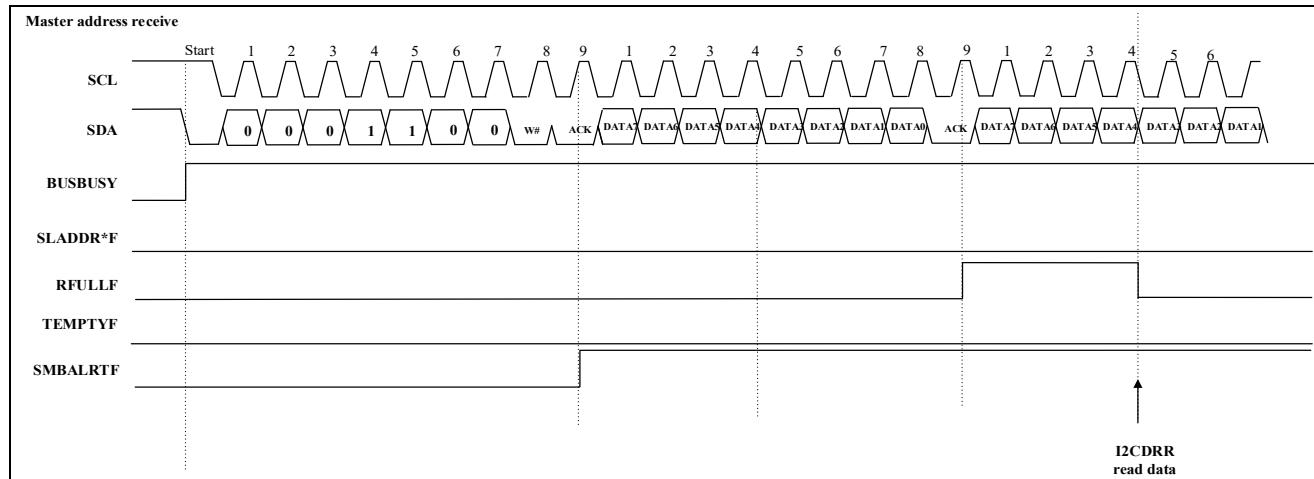


Figure 30-18 SMBus alarm response address reception timing diagram

30.3.2.5 SMBus default address match

This I2C controller has the default address detection function when SMBus runs. If the I2C_CR1.SMBDEFAULTEN bit is set to "1" when the I2C_CR1.SMBUS bit is "1", the SMBus default address (1100 001b) can be detected in the slave receive mode (I2C_CR1.MSL bit TRA bit is "00b").

If the SMBus default address is detected, set the I2C_SR. SMBDEFAULTF flag to "1" at the falling edge of the 9th clock of the SCL clock.

The operation after SMBus default address detection runs the same as the normal slave mode.

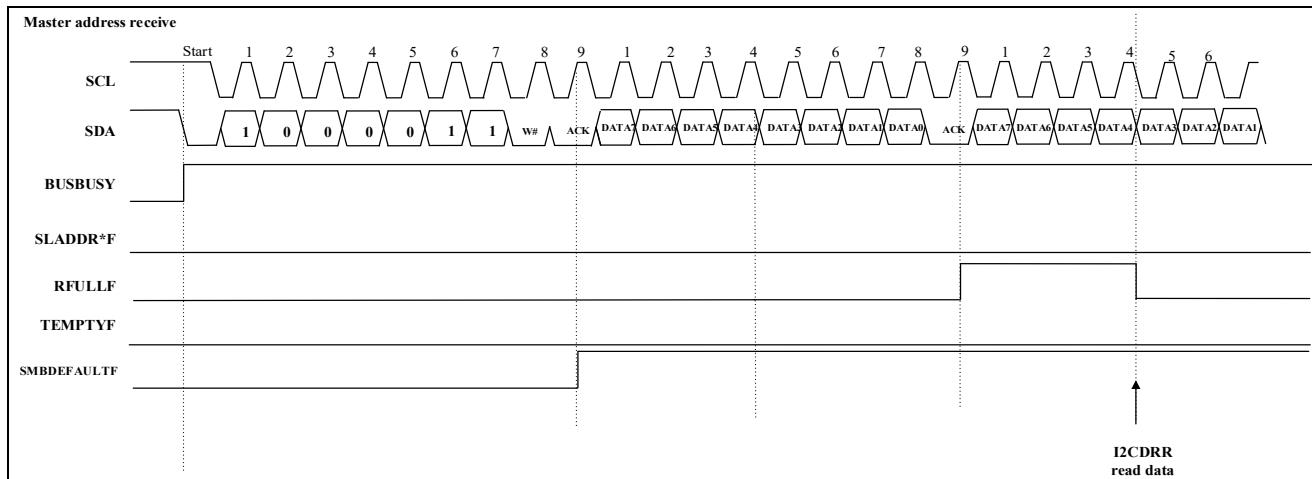


Figure 30-19 SMBus default address receive timing diagram

30.3.3 SMBus action

This I2C interface is compatible with SMBus (Ver.2.0).

Before using the SMBus interface you need to set the SMBus enable bit and the appropriate baudrate which varies from 10 kbps to 100 kbps and set the input level of SCL and SDA pin to TTL.

As for the SMBus enable bit you can set the I2C_CR1.SMBUS bit.

As for the SMBUS's baudrate you can set the I2C_CCR register.

As for the input level you can refer to PORT chapter for details.

30.3.3.1 SMBus timeout detection

SCL level timeout measurement

In the bus busy state, it is possible to detect that the SCL's low level or high level is fixed for a certain time or more, and the abnormal state of the bus can be detected.

The timeout detection function monitors the state of the SCL, and counts the high or low time through an internal counter. If the SCL changes (rising/falling), reset the internal counter, otherwise continue counting. If the internal counter counts to the TOUTHIGH/TOUTLOW setting value when the SCL has not changed, the timeout can be detected and the abnormal state of the bus can be notified.

For the counting of the internal counter, you can select whether to count in the low or high state of the SCL line by setting the HTMOUT and LTMOUT bits, or to count in both the low and high states. If both HTMOUT and LTMOUT bits are set to "0", no internal counting is performed.

Timeout measurement of slave machine

A slave device for SMBus communication needs to measure the interval shown below (timeout interval: TLOW:SEXT).

- Interval from start condition to stop condition

When timeout is measured by the slave device, the start condition interrupt and the stop condition interrupt are used as the measurement interval which is measured by the on-chip timer.

The measurement time of this timeout must be within the time specified by SMBus. If the low level time of the SCL measured by the on-chip timer exceeds the time specified by SMBus, the slave needs to release the bus.

Timeout Measurement of Master

The master device for SMBus communication needs to measure the interval shown below (timeout interval: TLOW:MEXT).

- Interval from start condition to the acknowledge bit
- Interval from the acknowledge bit to the next acknowledge bit
- Acknowledge bit to stop condition

When the master performs timeout measurement, the start condition interrupt, stop condition interrupt, transfer done interrupt and receive data full interrupt are used as the measurement interval which is measured by the on-chip timer.

The measurement time of this timeout must be within the time specified by SMBus. If the low level time of the SCL measured by the on-chip timer exceeds the time specified by SMBus, the master needs to issue a stop to abort the communication. When the master is under transmitting state then the master can issue a write operation to the I2C_DTR register to abort the communication.

30.3.3.2 Packet Error Code (PEC)

In SMBus communication, the CPU first calculates the CRC then sends the packet error code (PEC) through the SMBus, the built-in CRC operator of the system can be used to calculate or verify the PEC code.

30.3.4 Software Reset

There are two kinds of reset, one for All registers including the BUSY flag in SR register, the other is only for the slave address matching state and the internal counter in order to maintain the previous setting.

After reset the software must clear the SWRST bit in CR1 register to release the reset.

No matter what kind of reset, the SCL/SDA pin output is released to a high impedance state, it can also be used to release the bus from unexpected STOP state.

The reset in the slave mode is an asynchronous action to the I2C bus, so try to avoid using it.

Note: During reset (I2C_CR1.PE =0 & I2C_CR1.SWRST=1), the bus status such as the start condition cannot be monitored.

30.3.5 Interrupt and event signal output

I2C controller has four interrupts and event outputs for triggering the activation of other peripheral circuits for user selection.

The four interrupts or event outputs include: communication error, end of reception, tx buffer empty and end of transmission.

The communication error includes: arbitration lost, NACK detection, time-out detection, start condition detection, and stop condition detection

The following figure shows the interrupt list.

Table 30-2 Interrupt List

Name	Interrupt source	Interrupt logo	Interrupt condition
I2C_EEI	Communication error/communication event	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	Received data full	RFULLF	RFULLF=1&RFULLIE=1
I2C_TXI	Send data null	TEMPTYF	TEMPTYF=1&TEMPTYIE=1
I2C_TEI	End of transmission	TENDF	TENDF=1&TENDIE=1

The following table shows the output of the event signal.

Table 30-3 Event signal output list

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
I2C_RXI	Received data full	RFULLF=1
I2C_TXI	Send data null	TEMPTYF=1
I2C_TEI	End of transmission	TENDF=1

30.3.6 Programmable digital filtering

The SCL pin and SDA pin used internal are filtered by an analog filter circuit and a digital filter. The analog filter is set in I2C_FLTR.ANFEN and the typical value is 50ns. A block diagram of the digital filter circuit is shown below.

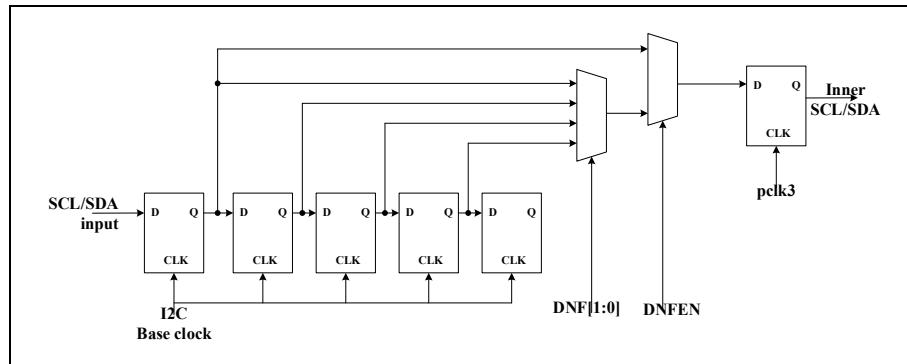


Figure 30-20 Digital filter circuit block diagram

The internal digital filter circuit is composed of 4 serial flipflops and matching detection circuit.

The effective number of segments of the digital filter is selected through the I2C_FLTR.DNF bits. According to the DNF bits setting, the capability of the filter can be varied from 1 to 4 I2C clock. 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.

If all the selected flipflops' output set by the I2C_FLTR.DNF bits are the same as the input then the internal signal updates, otherwise it keeps the original value.

30.4 Application software setting I2C initialization process

When starting to transmit or receive data, you must initialize the steps shown in the following figure.

1. Set PE to 0.
2. Set SWRST to 1, communication reset
3. Set PE to 1, internal state reset
4. Set slave address format and address
5. Set baud rate
6. Set control register function and interrupt as needed
7. SWRST bit is set to 0 to cancel internal state reset.
8. Initialization ends

30.5 Register description

I2C1 base address: 0x4004E000

I2C2 base address: 0x4004E400

I2C3 base address: 0x4004E800

I2C4 base address: 0x4004EC00

I2C5 base address: 0x4004F000

I2C6 base address: 0x4004F400

Table 30-4 I2C Registers

Register name	Symbol	Offset address	Bit width	Reset value
I2C control register1	I2C_CR1	0x00	32	0x0000 0040
I2C control register2	I2C_CR2	0x04	32	0x0000 0000
I2C control register3	I2C_CR3	0x08	32	0x0000 0006
I2C control register4	I2C_CR4	0x0C	32	0x0030 0307
I2C slave address register0	I2C_SLR0	0x10	32	0x0000 1000
I2C slave address register1	I2C_SLR1	0x14	32	0x0000 0000
I2C scl level time out control register	I2C_SLTR	0x18	32	0xFFFF FFFF
I2C status register	I2C_SR	0x1C	32	0x0000 0000
I2C status clear register	I2C_CLR	0x20	32	0x0000 0000
I2C data trasmit register	I2C_DTR	0x24	8	0xFF
I2C data receive register	I2C_DRD	0x28	8	0x00
I2C baud rate control register	I2C_CCR	0x2C	32	0x0000 1F1F
I2C filter control register	I2C_FLTR	0x30	32	0x0000 0010

30.5.1 I2C Controlled Register1 (I2C_CR1)

Reset value: 0x00000040

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	STAR T	REST ART	GCE N	-	SMB HOST EN	SMB DEFA ULTE N	SMB ALRT EN	SMB US	PE

Bit	Symbol	Bit name	Description	Read and write									
b31-16	Reserved	-	Read as "0", write as "0"	R/W									
b15	SWRST	Software reset	0: Software Reset disable 1: Software Reset enable Combination of native and PE bits, selecting internal state reset or communication reset <table border="1" style="margin-left: 20px;"> <tr> <td>SWRST</td><td>PE</td><td>Reset content</td></tr> <tr> <td>1</td><td>0</td><td>Communication reset: All the internal register and states within I2C are reset.</td></tr> <tr> <td>1</td><td>1</td><td>Internal state reset: I2C_SR, I2C_DSR register and internal state machine to reset</td></tr> </table>	SWRST	PE	Reset content	1	0	Communication reset: All the internal register and states within I2C are reset.	1	1	Internal state reset: I2C_SR, I2C_DSR register and internal state machine to reset	R/W
SWRST	PE	Reset content											
1	0	Communication reset: All the internal register and states within I2C are reset.											
1	1	Internal state reset: I2C_SR, I2C_DSR register and internal state machine to reset											
b14-11	Reserved	-	Read as "0", write as "0"	R/W									
b10	ACK	Acknowledge response bit	0: Generate ACK bit(response with bit '0' to the bus) 1: Generate NACK bit (response with bit'1' to the bus)	R/W									
b9	STOP	Stop condition generation bit	0: Do not generate stop conditions 1: Generation stop condition This bit can be set to 1 and cleared to 0. Hardware self-clearing conditions: Stopped condition detected Arbitration lost Start condition detected Communication reset	R/W									
b8	START	Start condition generating bit	0: No starting condition 1: Generation start condition This bit can be set to 1 and cleared to 0. Hardware self-clearing conditions: Start condition detected Failure of arbitration Communication reset	R/W									
B7	RESTART	Re-start condition generating bit	0: Do not generate re-start condition 1: Generate re-start condition This bit can be set to 1 and cleared to 0. Hardware self-clearing conditions: 1) Start condition detected 2) Arbitration lost 3) Communications resetting	R/W									
b6	GCEN	General call address match enable bit	0: Disable general call address detection 1: Enable general call address detection	R/W									
b5	Reserved	-	Read as "0", write as "0"	R/W									
b4	SMBHOSTEN	SMBus master address match enable bit	0: Disable SMBus master address match function 1: Ensable SMBus master address match function	R/W									
b3	SMBDEFAULTEN	SMBus default address match enable bit	0: Disable SMBus default address match function 1: Ensable SMBus default address match function	R/W									
b2	SMBALRTEN	SMBus alarm response address match enable bit	0: Disable SMBus alarm response address match function 1: Allow SMBus alarm response address match function	R/W									
b1	SMBUS	SMBus bus mode select bit	0: Select I2C bus mode 1: Select SMBus bus mode	R/W									
b0	PE	I2C function enable bit	0: Disable I2C function 1: Enable I2C function Combined with SWRST bit listed above to select the I2C reset mode	R/W									

30.5.2 I2C Control Register 1 (I2C_CR2)

Reset value: 0x00000000

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
-	TMO UTIE	-	NAC KIE	-	-	ARL OIE	-	TEM PTYIE	RFUL LIE	-	STOP IE	TEN DIE	SLA DDR1 IE	SLA DDR0 IE	STAR TIE

Bit	Symbol	Bit name	Description	Read and write
b31~b24	Reserved	-	Read as "0", write as "0"	R/W
b23	SMBALRTIE	SMBus alarm response address match interrupt enable	0: SMBus alarm response address Match interrupt disable 1: SMBus alarm response address Match interrupt enable	R/W
b22	SMBHOSTIE	SMBus master address match interrupt enable	0: SMBus master address natch interrupt disable 1: SMBus master address natch interrupt enable	R/W
b21	SMBDEFAULTIE	SMBus default address match interrupt enable	0: SMBus default address match interrupt dable 1: SMBus default address match interrupt enable	R/W
b20	GENCALLIE	General call address match interrupt enable	0: General call address match interrupt enable 1: General call address match interrupt disable	R/W
b19~b15	Reserved	-	Read as "0", write as "0"	R/W
b14	TMOUIE	Timeout interrupt enable	0: Timeout interrupt disabled 1: Timeout interrupt enabled	R/W
b13	Reserved	-	Read as "0", write as "0"	R/W
b12	NACKIE	NACK interrupt enable	0: NACK interrupt disable 1: NACK interrupt enable	R/W
b11~b10	Reserved	-	Read as "0", write as "0"	R/W
b9	ARLOIE	Arbitration lost interrupt enable	0: Arbitration lost interrupt disable 1: Arbitration lost interrupt enable	R/W
b8	Reserved	-	Read as "0", write as "0"	R/W
B7	TEMPTYIE	TX buffer empty interrupt enable	0: TX buffer empty interrupt disable 1: TX buffer empty interrupt enable	R/W
b6	RFULLIE	RX buffer full interrupt enable	0: RX buffer full interrupt disable 1: RX buffer full interrupt enable	R/W
b5	Reserved	-	Read as "0", write as "0"	R/W
b4	STOPIE	Stop condition interrupt enable	0: Stop condition interrupt disable 1: Stop condition interrupt enable	R/W
b3	TENDIE	TX packet transmission finish interrupt	0: TX transmition finish interrupt disable 1: TX transmition finish interrupt enable	R/W
b2	SLADDR1IE	Slave address1 match interrupt enable	0: Slave address1 match interrupt disable 1: Slave address1 match interrupt enable	R/W
b1	SLADDR0IE	Slave address0 match interrupt enable	0: Slave address0 match interrupt disable 1: Slave address0 match interrupt disable	R/W
b0	STARTIE	Start condition/restart condition interrupt enable	0: Start condition interrupt disable 1: Start condition interrupt enable	R/W

30.5.3 I2C Control Register 1 (I2C_CR3)

Reset value: 0x00000006

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	TMO UTEN

Bit	Symbol	Bit name	Description	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
B7	FACKEN	RFULLF flag timing selection	0: The RFULLF flag is set to '1' after the 9th rising edge of SCL, by the meantime the SCL will be forced to '0' after the 9th falling edge of the SCL 1: The RFULLF flag is set to '1' after the 8th rising edge of SCL, by the meantime the SCL will be forced to '0' after the 8th falling edge of the SCL Note: The forced SCL can be released by writing the ACK bit in CR1 register	R/W
b6~b3	Reserved	-	Read as "0", write as "0"	R/W
b2	HTMOUT	SCL high level timeout detection enable	0: Timeout detection is disabled when the SCL line is high. 1: Timeout detection is enabled when the SCL line is high.	R/W
b1	LTMOUT	SCL low level timeout detection enable	0: Timeout detection is disabled when the SCL line is low. 1: Timeout detection is enabled when the SCL line is low.	R/W
b0	TMOUTEN	SCL timeout function enable	0: SCL level timeout detection function is disabled 1: SCL level timeout detection function is enable	R/W

30.5.4 I2C Control Register 4 (I2C_CR4)

Reset value: 0x00300307

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 WAIT	-	-	-	-	-	-	-	-	-	-

Bit	Symbol	Bit name	Description	Read and write
b31~b22	Reserved	-	Read as "0", write as "0"	R/W
b21~b20	Reserved	-	Read as "1", write as "1"	R/W
b19~b11	Reserved	-	Read as "0", write as "0"	R/W
b10	BUSWAIT	bus wait bit	0: When I2C_DRR (data received register) is full and the I2C_DSR (data shift register) is empty the controller can continuously receive the next data 1: When I2C_DRR (data received register) is full and the I2C_DSR (data shift register) is empty the controller forces the SCL to '0' between the 9th SCL and the 1th SCL of the next bus operation. After reading the I2C_DRR register the forced SCL will be released and the new bus operation begins.	
b9~b8	Reserved	-	Read as "1", write as "1"	R/W
b7~b3	Reserved	-	Read as "0", write as "0"	R/W
b2~b0	Reserved	-	Read as "1", write as "1"	R/W

30.5.5 I2C slave address register0 (I2C_SLR0)

Reset value: 0x00001000

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 RMO D0	-	-	SLAD DR0E N	-	-							SLADDR0[9:0]			

Bit	Symbol	Bit name	Description	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	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 as "0", write as "0"	R/W
b12	SLADDR0EN	Slave address0 enable bit	0: Slave address0 for address match is disable 1: Slave address0 for address match is enable	R/W
b11~b10	Reserved	-	Read as "0", write as "0"	R/W
b9~b8	SLADDR0[9:8]	The high bits of a 10-bit slave address.	Bit9 and bit8 of the slave address in a 10-bit slave address format. When ADDRMODE0 bit is "1", SLADDR0 [9:8] is the highest two bits address of the 10-bit slave address	R/W
b7~b0	SLADDR0[7:0]	Lower bits of 7-bit address/10-bit address	Bit7 to bit0 of the slave address. When ADDRMODE0 bit is "0", SLADDR0 [7: 1] is the 7-bit slave address. SLADDR0 [0] bit is invalid. When ADDRMODE0 bit is "1", SLADDR0 [7: 0] is the lowest 8-bit address of the 10-bit slave address.	R/W

30.5.6 I2C slave address register1 (I2C_SLR1)

Reset value: 0x00000000

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 RMO D1	-	-	SLAD DR1E N	-	-	SLADDR1[9:0]									

Bit	Symbol	Bit name	Description	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	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 as "0", write as "0"	R/W
b12	SLADDR1EN	Slave address1 enable bit	0: Slave address1 for address match is disable 1: Slave address1 for address match is enable	R/W
b11~b10	Reserved	-	Read as "0", write as "0"	R/W
b9~b8	SLADDR1[9:8]	The high bits of a 10-bit slave address.	Bit9 and bit8 of the slave address in a 10-bit slave address format. When ADDRMOD1 bit is "1", SLADDR1 [9: 8] is the highest two bits address of the 10-bit slave address.	R/W
b7~b0	SLADDR1[7:0]	Lower bits of 7-bit address/10-bit address	Bit7 to bit0 of the slave address. When ADDRMOD1 bit is "0", SLADDR1 [7: 1] is the 7-bit slave address. SLADDR1 [0] bit is invalid. When ADDRMOD1 bit is "1", SLADDR1 [7: 0] is the lowest 8-bit address of the 10-bit slave address.	R/W

30.5.7 I2C SCL Level Timeout Control Register (I2C_SLTR)

Reset value: 0xFFFFFFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TOUTHIGH[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
TOUTLOW[15:0]															

Bit	Symbol	Bit name	Description	Read and write
b31~b16	TOUTHIGH	SCL high level timeout period	TOUTHIGH sets the SCL high timeout period. SCL high level timeout time = TOUTHIGH * I2C reference clock period	R/W
b15~b0	TOUTLOW	SCL low level timeout period	TOUTLOW sets the SCL low timeout period. SCL low level timeout = TOUTLOW*I2C reference clock period	R/W

30.5.8 I2C Status register (I2C_SR)

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	SMB ALRT F	SMB HOST F	SMB DEFA ULTF	GEN CALL F	-	TRA	BUS Y	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	RFUL LF	-	STOP F	TEN DF	SLA DDR1 F	SLA DDR0 F	STAR TF

Bit	Symbol	Bit name	Description	Read and write
b31~b21	Reserved	-	Read as "0", write as "0"	R
b23	SMBALRTF	SMBus alarm response address match flag	0: SMBus alarm response address not matched 1: SMBus alarm response address matched Note: When the received address matches 0001100b then it sets SMBALRTF flag When one of the following meets then it clears the SMBALRTF flag SMBALRTFCLR bit in CLR register is written to '1' 1) SMBALRTFCLR bit in CLR register is written to '1' 2) stop condition detected 3) communication reset	R
b22	SMBHOSTF	SMBus Master Address match flag	0: SMBus master address is not matched 1: SMBus master address is matched Note: When the received address matches 0001000b then it sets SMBHOSTF flag When one of the following meets then it clears the SMBHOSTF flag 1) SMBHOSTFCLR bit in CLR register is written to '1' 2) stop condition detected 3) communication reset	R
b21	SMBDEFAULTF	SMBus default address match flag	0: SMBus default address is not matched 1: SMBus default address is matched Note: When the received address matches 1100001b then it sets SMBDEFAULTF flag When one of the following meets then it clears the SMBDEFAULTF flag 1) SMBDEFAULTFCLR bit in CLR register is written to '1' 2) stop condition detected 3) communication reset	R
b20	GENCALLF	General call address match flag	0: General call address is not matched 1: General call address is matched Note: When the received address matches 0000000b then it sets GENCALLF flag When one of the following meets then it clears the GENCALLF flag 1) GENCALLFCLR bit in CLR register is written to '1' 2) stop condition detected 3) communication reset	R
b19	Reserved	-	Read as "0", write as "0"	R
b18	TRA	Transmit receive select bit	This bit indicates whether to choose to send or receive data. 0: Receive data 1: Send data When one of the following meets then it sets the TRA flag 1) Start condition detected 2) In master mode, the R/W bit of the send address is 0 3) In slave mode, the address matches and the received R/W bit is 1 When one of the following meets then it clears the TRA	R/W

			flag 1) Stop condition detected 2) In master mode, the R/W bit of the send address is 1 3) In slave mode, the address matches and the received R/W bit is 0 4) Communication reset																
b17	BUSY	Bus busy flag bit	<p>0: Idle state, no communication on bus 1: In possession, the bus is communicating When one of the following meets then it sets the BUSY flag</p> <p>Start condition detected on bus When one of the following meets then it clears the BUSY flag</p> <p>1) Bus stop condition detected 2) Communication reset</p>	R															
b16	MSL	Master-slave selection bit	<p>This bit indicates whether the host is still a slave. 0: Slave mode 1: Host mode The operating mode of I2C is represented by a combination of TRA bits.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <th>MSL</th><th>TRA</th><th>I2C operating mode</th></tr> <tr> <td>0</td><td>0</td><td>Slave receive mode</td></tr> <tr> <td>0</td><td>1</td><td>Slave mode</td></tr> <tr> <td>1</td><td>0</td><td>Host receive mode</td></tr> <tr> <td>1</td><td>1</td><td>Master transmit mode</td></tr> </table> <p>When one of the following meets then it sets the MSL flag</p> <p>Start condition detected when START bit is 1 When one of the following meets then it clears the MSL flag</p> <p>1) Stop condition detected 2) Arbitration lost 3) Communication reset</p>	MSL	TRA	I2C operating mode	0	0	Slave receive mode	0	1	Slave mode	1	0	Host receive mode	1	1	Master transmit mode	R/W
MSL	TRA	I2C operating mode																	
0	0	Slave receive mode																	
0	1	Slave mode																	
1	0	Host receive mode																	
1	1	Master transmit mode																	
b15	Reserved	-	Read as "0", write as "0"	R															
14	TMOUTF	time-out flag	<p>0: SCL level timeout not detected 1: SCL level timeout detected Bit set condition: In the period set by I2C_SLTR, SCL is not toggled Bit clear condition: 1) TMOUTFCLR writes "1" 2) Communication reset</p>	R															
b13	Reserved	-	Read as "0", write as "0"	R															
b12	NACKF	Nack flag bit	<p>0: No NACK Received 1: NACK received Bit set condition: Receive NACK in transmit mode Bit clear condition: 1) NACKFCLR writes "1" 2) Communication reset</p>	R															
b11	Reserved	-	Read as "0", write as "0"	R															
b10	ACKRF	Acknowledge bit	<p>0: Received ACK bit is "0" (ACK received) 1: Received reply bit is '1' (receive NACK) Bit set condition: Receive NACK in transmit mode Bit clear condition: 1) Receive ACK in transmit mode 2) Communication reset</p>	R															
b9	ARLOF	Arbitration lost flag bit	<p>0: No arbitration lost 1: Arbitration lost Bit set condition: Arbitration lost Bit clear condition: 1) ARLOFCLR Write "1" 2) Communication reset</p>	R															
b8	Reserved	-	Read as "0", write as "0"	R															
B7	TEMPTYF	TX buffer empty flag	<p>0: TX buffer(I2C_DTR) is not empty 1: TX buffer(I2C_DTR) is empty Bit set condition: 1) I2C_DTR Data Transfer to I2C_DSR</p>	R															

			2) TRA bit set to 1 Bit clear condition: 1) Write I2C_DTR in normal mode 2) TRA bit clear 0 3) Communication reset	
b6	RFULLF	RX buffer full flag bit	0: RX buffer(I2C_DRR) is not full 1: RX buffer(I2C_DRR) is full Bit set condition: Received data transfer from I2C DSR to I2C DRR Bit clear condition: 1) Read I2C DRR in normal mode 2) RFULLFCLR Write "1" 3) Communication reset	R
b5	Reserved	-	Read as "0", write as "0"	R
b4	STOPF	Stop condition flag bit	0: No stop condition detected on bus 1: Bus detected stop condition Bit set condition: Stop condition detected Bit clear condition: 1) STOPFCLR Write "1" 2) Communication reset	R
b3	TENDF	TX transmit end flag bit	0: One byte TX data is transmitting or there is no valid data in shift register 1: One byte TX data transmitted Set the "1" condition: Bit set condition: This bit sets after 9th rising edge of SCL (transmit clock) Bit clear condition: 1) Stopp condition detected 2) Write I2C_DTR in normal mode 3) TENDFCLR writes "1" 4) Communication reset	R
b2	SLADDR1F	Slave address1 match flag bit	0: The receiving slave address is not MATCH in slave address1 mode 1: The receiving slave address is MATCH in slave address1 mode Bit set condition: 7bit address mode (ADDRMOD1 ==0) The first received address is the same as SLADDR1 [7: 1] 10bit address mode (ADDRMOD1 ==1) The first received address is the same as {1110b, SLADDR1 [9: 8]} The second received address is the same as SLADDR1 [7: 0] Bit clear condition: 1) Stopp condition detected 2) SLADDR1FCLR write "1" 3) Communication reset	R
b1	SLADDR0F	Slave address0 match flag bit	0: The receiving slave address is not MATCH in slave address0 mode 1: The receiving slave address is MATCH in slave address0 mode Bit set condition: 7bit address mode (ADDRMOD0 ==0) The first received address is the same as SLADDR0 [7: 1] 10bit address mode (ADDRMOD0 ==1) The first received address is the same as {1110b, SLADDR0 [9: 8]} The second received address is the same as SLADDR0 [7: 0] Bit clear condition: 1) Stopped condition detected 2) SLADDR0FCLR write "1" 3) Communication reset	R
b0	STARTF	Start condition/restart condition flag bit	0: Start condition not detected 1: Start condition detected Bit set condition: Starting condition detected Bit clear condition: 1) Stop condition detected 2) STARTFCLR writes "1" 3) Communications resetting	R

30.5.9 I2C Status Clear Register (I2C_CLR)

Reset value: 0x00000000

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 UTFCL LR	-	NAC KFCL R	-	-	ARL OFCL R	-	TEM PTYF CLR	RFUL LFCL R	-	STOP FCLR	TEN DFCL R	SLA DDR1 FCLR	SLA DDRO FCLR	STAR TFCL R

Bit	Symbol	Bit name	Description	Read and write
b31~b24	Reserved	-	Read 0 when reading, write 0 when writing	W
b23	SMBALRTFCLR	SMVARLT clear bit	Write "1" to clear the SMBALRT flag bit	W
b22	SMBHOSTFCLR	SMBHOSTF clear bit	Write "1" to clear the SMBHOSTF flag bit	W
b21	SMBDEFAULTFCLR	SMBDEFAULTF clear bit	Write "1" to clear SMBDEFAULTF flag bit	W
b20	GENCALLFCLR	GENCALLF clear bit	Write "1" to clear GENCALLF flag bit	W
b19~b15	Reserved	clear bit	Read 0 when reading, write 0 when writing	W
b14	TMOUTFCLR	TMOUTF clear bit	Write "1" to clear the TMOUTF flag	W
b13	Reserved	clear bit	Read 0 when reading, write 0 when writing	W
b12	NACKFCLR	NACKF clear bit	Write "1" to clear the NACKF flag bit	W
b11~b10	Reserved	clear bit	Read 0 when reading, write 0 when writing	W
b9	ARLOFCLR	ARLOF clear bit	Write "1" to clear the ARLOF flag bit	W
b8	Reserved	clear bit	Read 0 when reading, write 0 when writing	W
B7	TEMPTYFCLR	TEMPTYF clear bit	Write "1" to clear TEMPTYF flag bit	W
b6	RFULLFCLR	RFULLF clear bit	Write "1" to clear RFULLF flag bit	W
b5	Reserved	clear bit	Read 0 when reading, write 0 when writing	W
b4	STOPFCLR	RFULLF clear bit	Write "1" to clear STOPF flag bit	W
b3	TENDFCLR	TENDF clear bit	Write "1" to clear the TENDF flag bit	W
b2	SLADDR1FCLR	SLADDR1F clear bit	Write "1" to clear the SLADDR1F flag bit	W
b1	SLADDR0FCLR	SLADDR0F clear bit	Write "1" to clear the SLADDR0F flag bit	W
b0	STARTFCLR	STARTF clear bit	Write '1' to clear the STARTF flag bit	W

30.5.10 I2C Data Transmitting Register (I2C_DTR)

Reset value: 0xFF

B7	b6	b5	b4	b3	b2	b1	b0
DT[7:0]							

If the I2C_DSR is empty, the transmit data written to the I2C_DTR register will be transformed to I2C_DSR register, and the data will be serially shifted to SDA in transmit mode.

The I2C_DSR register and the I2C_DTR register are double-buffered structures. During the data shifting process of the I2C_DSR register, if there is a pre-write operation to the I2C_DTR register, the data can be loaded into the DSR register at once after the previous data shifted over.

The I2C_DTR register is readable and writable. Write to the I2C_DTR register only once when the transmit data empty interrupt request occurs.

30.5.11 I2C Data Receiving Register (I2C_DRR)

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
DR[7:0]							

If one frame is received, the received data can be transferred from the shift register (I2C_DSR) to the I2C_DRR register, after that it can begin the next data reception.

The I2C_DSR register and the I2C_DRR register are double-buffered structures. During the data reception process of the I2C_DSR register, if the data of the I2C_DRR register is read, the data can be continuously received.

Writes to the I2C_DRR register are prohibited. Please read the I2C_DRR register only once when the received data full interrupt request occurs.

When the RFULLF flag in the I2C controller is valid, if the data is not read in time but continues to receive data, the I2C controller will force the SCK to low level one SCK before the next RFULLF flag changes to "1"

30.5.12 I2C Data Shift Register (I2C_DSR)

B7	b6	b5	b4	b3	b2	b1	b0
DSR							

The I2C_DSR register is a shift register used to transmit and receive data.

The I2C_DSR register is neither readable nor writable.

When data is transmitted, the transmit data is transferred from the I2C_DTR register to the I2C_DSR register, and the data is shifted to the SDA pin. During data reception, once a frame of data is received, the data is transferred from the I2C_DSR register to the I2C_DRR register.

30.5.13 I2C Clock Control Register (I2C_CCR)

Reset value: 0x00001F1F

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]				

Bit	Symbol	Bit name	Description	Read and write
b31~b23	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b22~b19	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b18~b16	FREQ[2:0]	reference clock select	000: I2C reference clock frequency = PCLK3/1 001: I2C reference clock frequency = PCLK3/2 010: I2C reference clock frequency = PCLK3/4 011: I2C reference clock frequency = PCLK3/8 100: I2C reference clock frequency = PCLK3/16 101: I2C reference clock frequency = PCLK3/32 110: I2C reference clock frequency = PCLK3/64 111: I2C reference clock frequency = PCLK3/128	R/W
b15~b13	Reserved	-	Read as "1", write as "1"	R/W
b12~b8	SHIGHW[4:0]	SCL high level width bit	Setting the High Level Width of SCL Clock	R/W
b7~b5	Reserved	-	Read as "1", write as "1"	R/W
b4~b0	SLOWW[4:0]	SCL low level width bit	Setting the Low Level Width of SCL Clock	R/W

SHIGHW bit (set SCL high level width bit)

In host mode, SHIGHW is used to set the high level width of the SCL clock. The setting is invalid in slave mode.

SLOWW bit (set SCL low level width bit)

SLOWW is used to set the low level width of the SCL clock. In slave mode, the set value is greater than the data preparation time. Data preparation time (tSU:DAT) 250ns (~100kbps: Standard mode) 100ns (~400kbps: fast mode)

Baud rate setting

DNFE=0, CKSDIV=000

Baud rate=1/{[(SHIGHW + 3) + (SLOWW + 3)]/ΦI2C + SCL rise time + SCL fall time}

SDNFE=1, CKSDIV=000

Baud rate=1/([(SHIGHW + 3 + filtering capability) + (SLOWW + 3 + filtering capability)]/ΦI2C + SCL rise time + SCL fall time}

DNFE=0, CKSDIV!=000

Baud rate=1/[(SHIGHW + 2) + (SLOWW + 2)]/ΦI2C + SCL rise time + SCL fall time}

DNFE=1, CKSDIV!=000

Baud rate=1/{[(SHIGHW + 2 + filter capability) + (SLOWW + 2 + filter capability)]/ΦI2C + SCL rise time + SCL fall time}

Note:

1. ΦI2C means I2C reference clk and can be set in I2C_CCR.FREO.
- 2.The rise time [tr] and fall time [tf] of the SCL line depend on the total capacity [Cb] and pull-up resistance [Rp] of the bus. Refer to NXP's I2C-bus specification and user manual for details.

30.5.14 I2C Filtering Control Register (I2C_FLTR)

Reset value: 0x00000010

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
-	-	-	-	-	-	-	-	-	-	ANFEN	DNFEN	-	-	DNF[1:0]	

Bit	Symbol	Bit name	Description	Read and write
b31~b6	Reserved	-	Read as "0", write as "0"	R/W
b5	ANFEN	Analog filtering enable bit	0: Analog filtering function disabled 1: Analog Filtering function enable	R/W
b4	DNFEN	Digital filtering enable bit	0: Digital filtering function disabled 1: Digital filtering function enable	R/W
b3~b2	Reserved	-	Read as "0", write as "0"	R/W
b1-b0	DNF[1:0]	Digital Filter Capability select	00: one I2C reference clock filter capability 01: two I2C reference clock filter capability 10: three I2C reference clock filter capability 11: four I2C reference clock filter capabilities	R/W

31 Serial Peripheral Interface (SPI)

31.1 Introduction

Serial peripheral interface SPI is equipped with six channels, which supports high-speed full-duplex serial synchronous communication and convenient data exchange with peripheral equipment. Users can set the range of 3/4-wire, master/slave and baud rate as needed.

Main features of SPI:

Table 31-1 Features of SPI

Essentials	Description
Number of channels	6 channels
Serial communication function	<ul style="list-style-type: none">Supporting 4-wire SPI mode and 3-wire clock synchronous operation modeSupport two communication modes: full duplex and transmit-onlySupport adjustable the polarity and phase of communication clockSupport adjustable transfer interval time
Data format	<ul style="list-style-type: none">Selectable data shift order: MSB first/LSB firstSelectable data width: 4/5/6/7/8/9/10/11/12/13/14/15/16/20/24/32 bitA maximum of 32 bits of data can be transmitted or received in four frames
Baud rate	<ul style="list-style-type: none">In master mode, the baud rate can be adjusted through internal baud rate generator. The baud rate range is the frequency division of PCLK1 by 2~256.In slave mode, baud rate is up to PCLK1 divided by 6
Data buffer	<ul style="list-style-type: none">With 16-byte data buffer areaSupport double buffering
Error monitoring	<ul style="list-style-type: none">Mode fault error detectionData overflow error detectionData underflow error detectionParity error detection
Slave select signal control	<ul style="list-style-type: none">Each channel is configured with four slave select signalsSupport adjustable time sequence between slave select signal and communication clockSupport adjustable invalid time of slave select signal between two consecutive communicationSupport adjustable polarity
Transmission Control in Master Mode	<ul style="list-style-type: none">Start transmission by writing data to the data registerCommunication auto-suspend function
Interrupt	<ul style="list-style-type: none">Receive buffer fullTransmit buffer emptySPI error (mode fault/overflow/underflow/parity)SPI idleTransfer complete (event source only)
Low power control	Support module stop function for low power mode
Other functions	<ul style="list-style-type: none">SPI initialization function

Note:

- When the communication auto-suspend function is used in master mode, overflow errors will not occur because the communication clock stops. For details, please refer to [31.8.3 overflow error].

31.2 SPI System Block Diagram

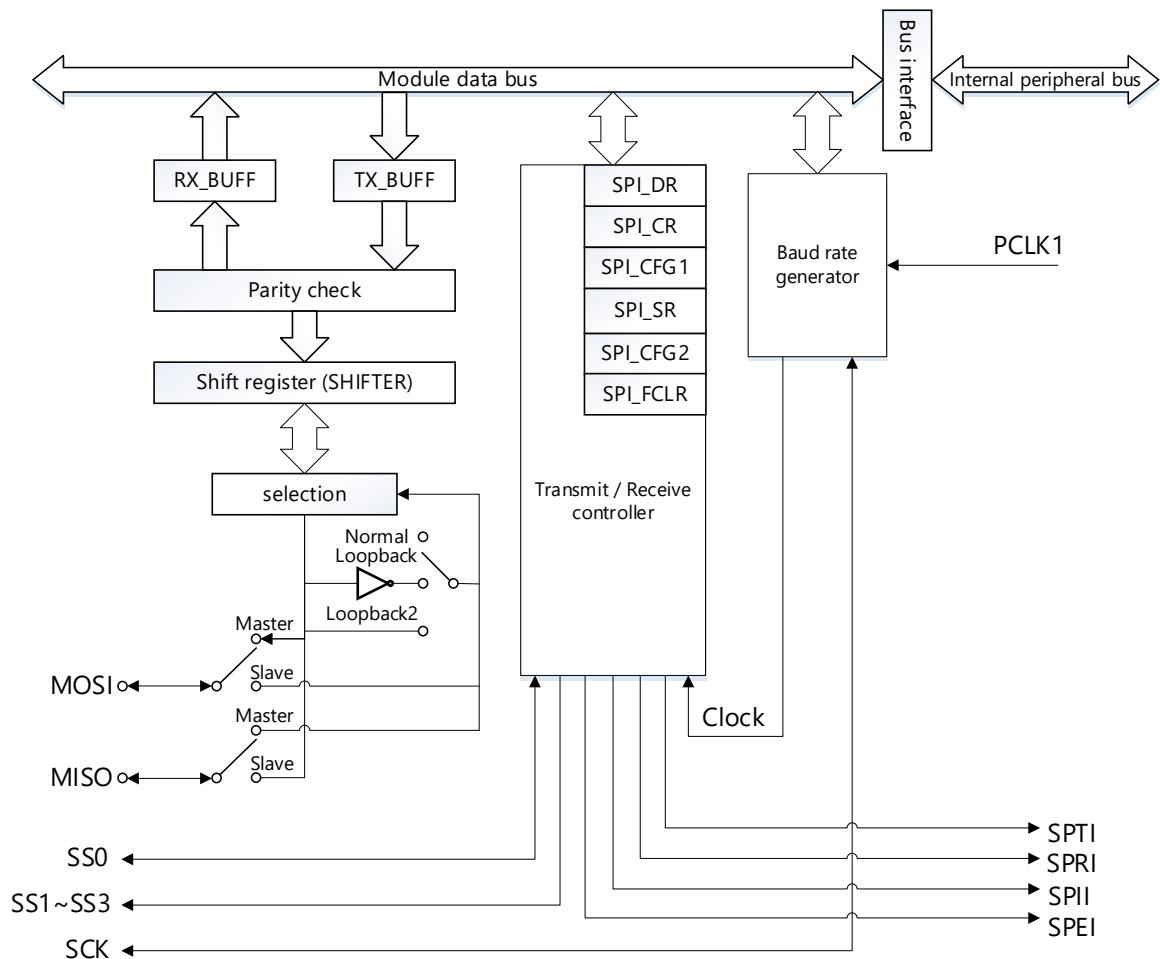


Figure 31-1 System Block Diagram

31.3 Pin description

Table 31-2 Pin Description

Pin name	Port direction	Function
SCK	Input/Output	Communication clock pin
MOSI	Input/Output	Master data transfer pin
MISO	Input/Output	Slave data transfer pin
SS0	Input/Output	Slave select input/output pin
SS1	Output	Slave select output pin
SS2	Output	Slave select output pin
SS3	Output	Slave select output pin

31.4 SPI System Description

31.4.1 Pin Status in Master Mode

When SPI works in master mode, the status of each pin is as followsTable 31-3 shown.

Table 31-3 SPI pin status description in master mode

Pattern		Pin name	Pin Status (PFS.ODS = 0)	Pin status (PFS.ODS = 1)
SPI action (SPIMDS=0)	Master mode (MSTR=1, MODFE=0)	SCK	CMOS Output	OD output
		SS0~SS3	CMOS Output	OD output
		MOSI	CMOS Output	OD output
		MISO	Input	Input
Clock synchronization operation (SPIMDS=1)	Master mode (MSTR=1)	SCK	CMOS Output	OD output
		SS0~SS3 (not used)	Hi-Z (available as GPIO)	Hi-Z (available as GPIO)
		MOSI	CMOS Output	OD output
		MISO	Input	Input

Note:

- Please set the pin input type to CMOS input and the output to high driving mode. Please refer to Chapter 11 GPIO Special Register PCR for settings.

31.4.2 Pin Status in slave mode

When SPI works in slave mode, the status of each pin is as followsTable 31-4 shown.

Table 31-4 SPI pin status description in slave mode

Pattern		Pin name	Pin Status (PFS.ODS = 0)	Pin status (PFS.ODS = 1)
SPI action (SPIMDS=0)	Slave mode (MSTR=0, MODFE=0)	SCK	Input	Input
		SS0	Input	Input
		SS1~SS3 (not used)	Hi-Z (available as GPIO)	Hi-Z (available as GPIO)
		MOSI	Input	Input
		MISO (*1)	CMOS Output/Hi-Z	OD Output/Hi-Z
Clock synchronization operation (SPIMDS=1)	Slave mode (MSTR=0)	SCK	Input	Input
		SS0~SS3 (not used)	Hi-Z (available as GPIO)	Hi-Z (available as GPIO)
		MOSI	Input	Input
		MISO	CMOS Output	OD output

Note:

- Please set the pin input type to CMOS input and the output to high driving mode. Please refer to Chapter 11 GPIO Special Register PCR for settings.

*1: MISO's status will be Hi-Z when SS0 level is in invalid status.

31.4.3 SPI System Connection Examples

SPI operation mode

In the single-master-multi-slave mode SPI system architecture, the master drives SCK, MOSI and SS0 ~ SS3. In SPI slave device 0~3, when the SS input of a certain slave device is active level, the corresponding slave device drives MISO.

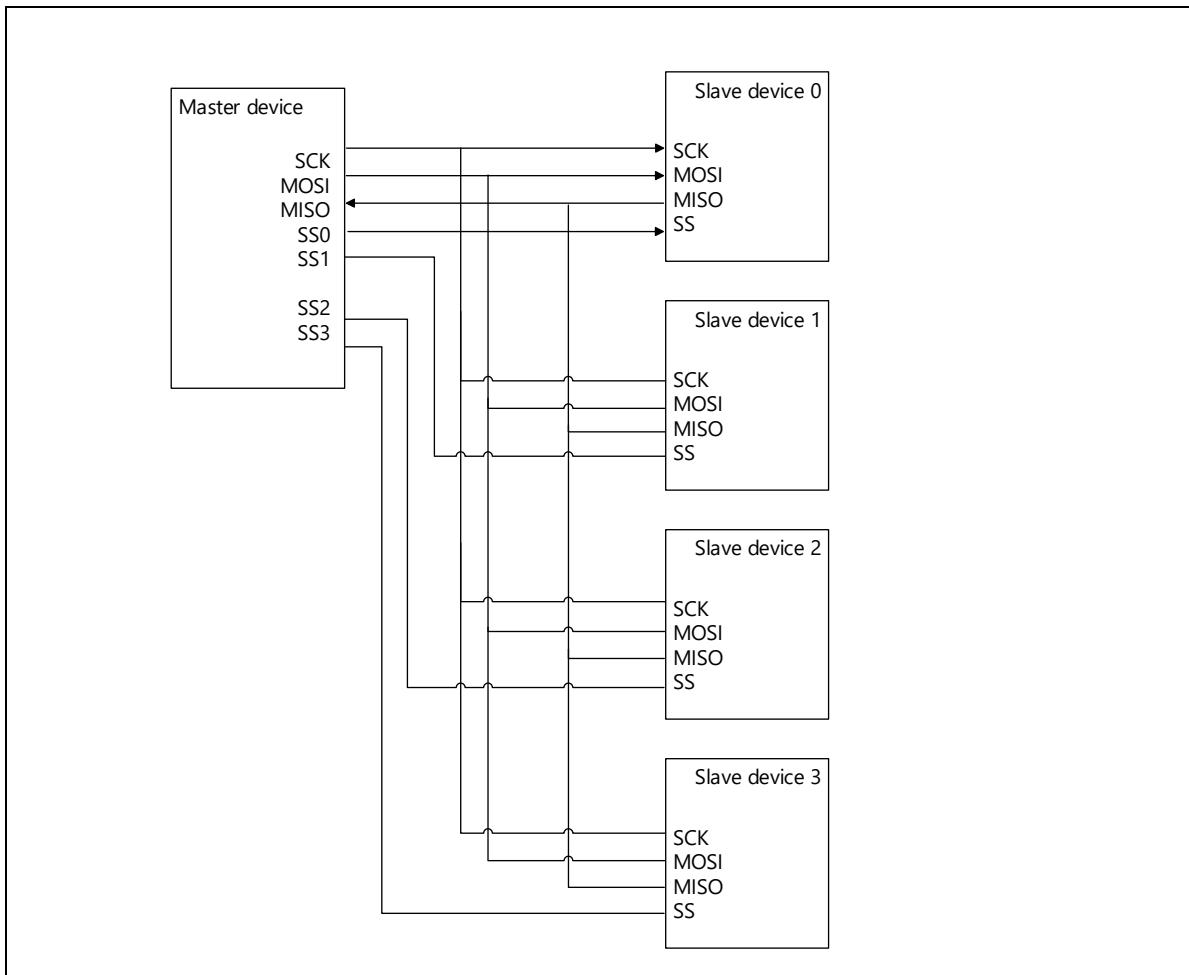


Figure 31-2 SPI operation mode structure

Clock synchronization operation mode

In the SPI system architecture used for clock synchronous operation, the master device drives SCK and MOSI, and the slave device drives MISO. SS pins are not used.

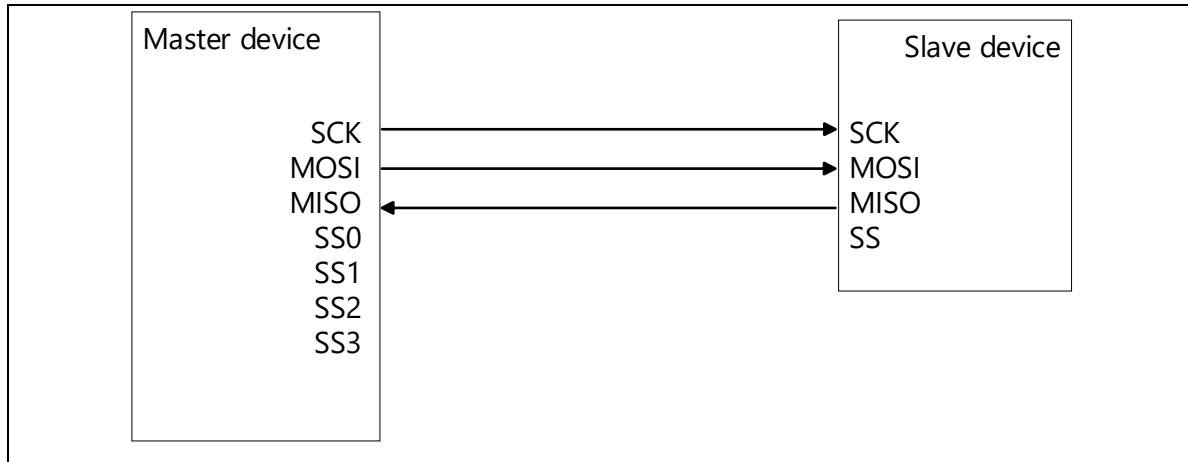


Figure 31-3 Clock synchronous operation mode (3 wire) structure

31.5 Data communication description

31.5.1 Baud rate

In master mode, SPI clock is provided by internal baud rate generator. In slave mode, the clock is inputted by the SCK pin.

Baud rate depends on SPI _ CFG2.MBR [2: 0] setting. The calculation method is as shown in the following formula, where N is the setting value of MBR [2: 0] bits, ranging from 0 to 7.

$$\text{Baud rate} = \frac{fpck}{2^{N+1}}$$

Table 31-5 baud rate calculated according to the set value

MBR [2: 0]	Frequency division ratio	Baud rate			
		PCLK1=5MHz	PCLK1=10MHz	PCLK1=20MHz	PCLK1=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

31.5.2 Data format

The SPI data format depends on register SPI_CFG2 and the value of SPI_CR.PAE. SPI processes a specific length data (set by SPI_CFG2.DSIZE[3:0]) from LSB in SPI_DR as the transfer data.

SPI_CFG2.DSIZE[3:0] determines the length of the data, and the range is from 4 to 32 bits. SPI_CR.PAE determines the last bit of the data. Such as Figure 31-4 shown.

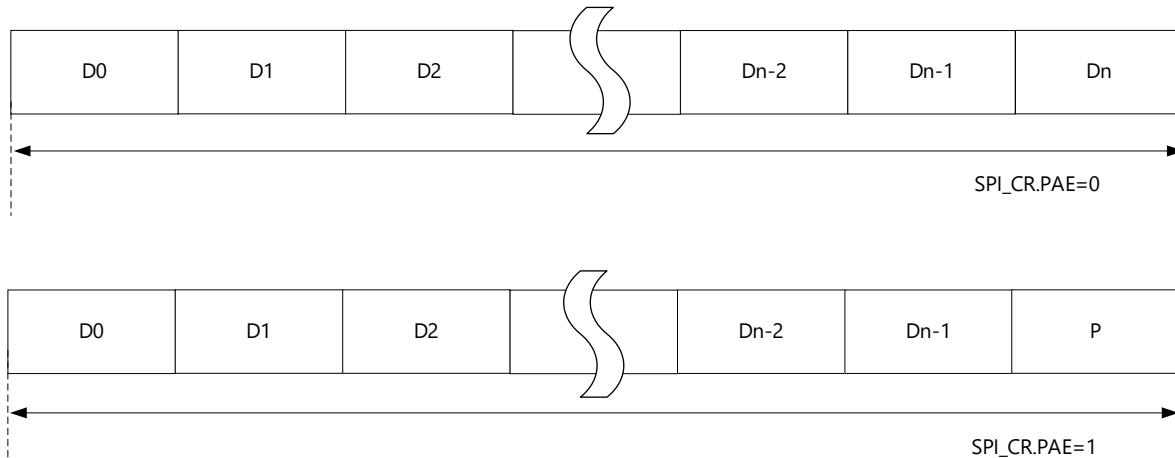


Figure 31-4 Data Format

At the beginning of transmission, SPI copies the data to the transmit buffer (TX_BUFF) first, then moves to the shift register (SHIFTER), and then sends it out in turn. At the beginning of reception, SPI captures the data to the shift register (SHIFTER) in turn first, and then copies to the receive buffer (RX_BUFF).

During data transfer, according to the setting of SPI_CFG2.LSBF and SPI_CR.PAE, there are four cases:

1. MSB first, parity check disabled

When transmitting, data (d31-d0) is copied from TX_BUFF to SHIFTER in order of d31-d0. Then SPI shifts data out from the highest position of SHIFTER in order of d31-d0;

When receiving, data (d31-d0) is shifted in the lowest position of SHIFTER, wait until all the data is moved in. Then SPI copies the data to RX_BUFF.

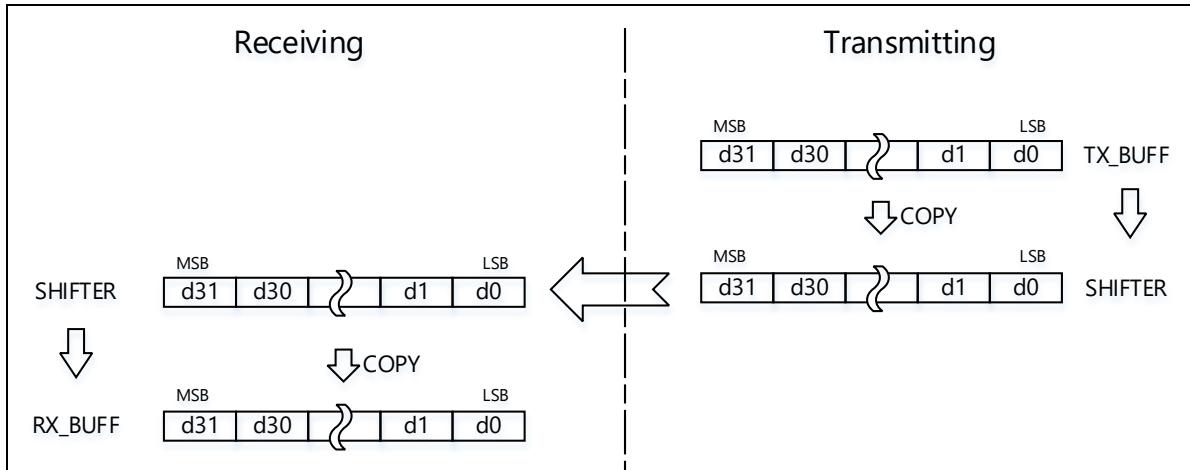


Figure 31-5 MSB first, parity check disabled

2. LSB first, parity check disabled

When transmitting, data (d31-d0) is copied from TX_BUFF to SHIFTER in order of d0-d31. Then SPI shifts out from the highest position of SHIFTER in order of d0-d31;

When receiving, data (d0-d31) is shifted in from the lowest position of SHIFTER. After the data is all shifted in, SPI copies from SHIFTER to RX_BUFF in order of d31-d0.

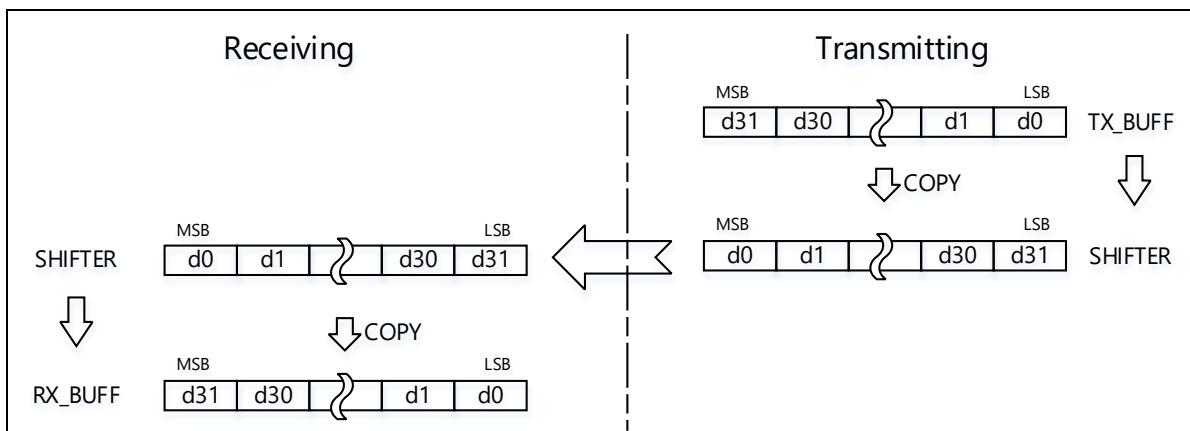


Figure 31-6 LSB first, parity check disabled

3. MSB first, parity check enabled

When transmitting, SPI calculates the value of the parity bit P according to the value of d31-d1 first, then replaces d0 with P, and copies from TX_BUFF to SHIFTER in order of d31-d1-P. SPI shifts out from the highest position of SHIFTER in order of d31-d1-P;

When receiving, data (d31-d1-P) is shifted in from the lowest position of SHIFTER. Then SPI performs to parity check, and copies the data to RX_BUFF. When parity check is enable, the parity bit replaces the last bit of the transmitted data.

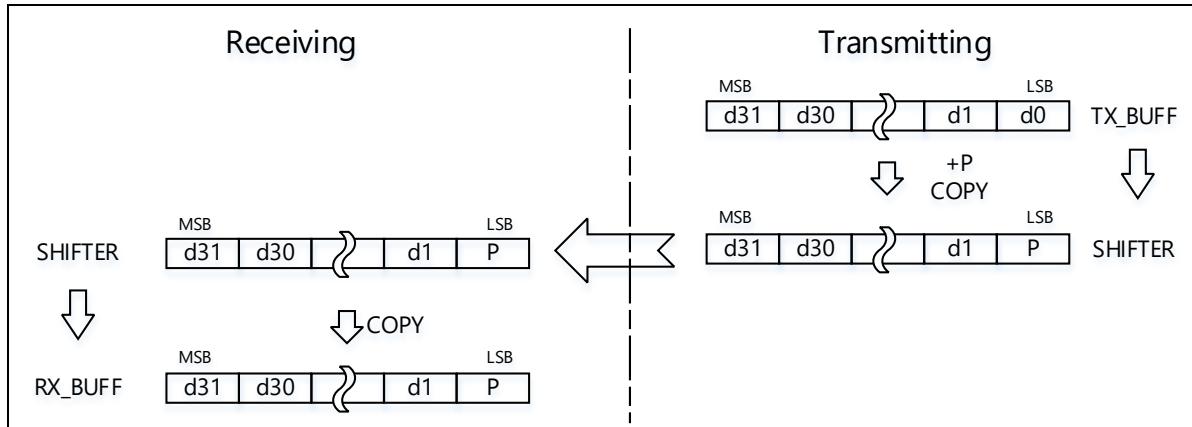


Figure 31-7 MSB first, parity check enabled

4. LSB first, parity check enabled

When transmitting, SPI calculates the value of the parity bit P according to the value of d30-d0 first, then replaces d31 with P, and copies from TX_BUFF to SHIFTER in order of d0-d30-P. SPI shifts out from the highest position of SHIFTER in order of d0-d30-P;

When receiving, the data (d0-d30-P) is shifted in from the lowest position of SHIFTER. Then SPI performs to parity check. The data (d0-d30-P) is rearranged during replication and copied to RX_BUFF in order of P-d30-d0. When parity check is enable, the parity bit replaces the last bit of the transmitted data.

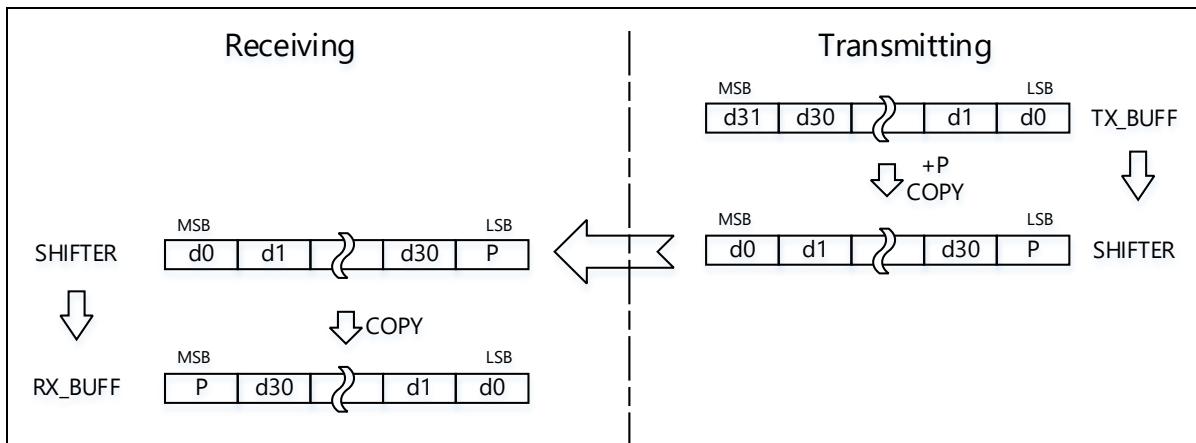


Figure 31-8 LSB first, parity check enabled

31.5.3 Transfer format

- CPHA = 0

When SPI_CFG2.CPHA = 0, SPI samples data at the odd edge of SCK and updates data at the even edge. Figure 31-9 is the transmission format diagram of SPI when CPHA=0. When the SS signal becomes valid, MOSI/MISO begins to transmit data. The first data sampling is performed on the first edge of SCK. After that, the data is sampled once per SCK. The data on MOSI/MISO is updated at 1/2 SCK after each sampling. The value of SPI_CFG2.CPOL does not affect the sequence of SCK, but only affects the signal polarity.

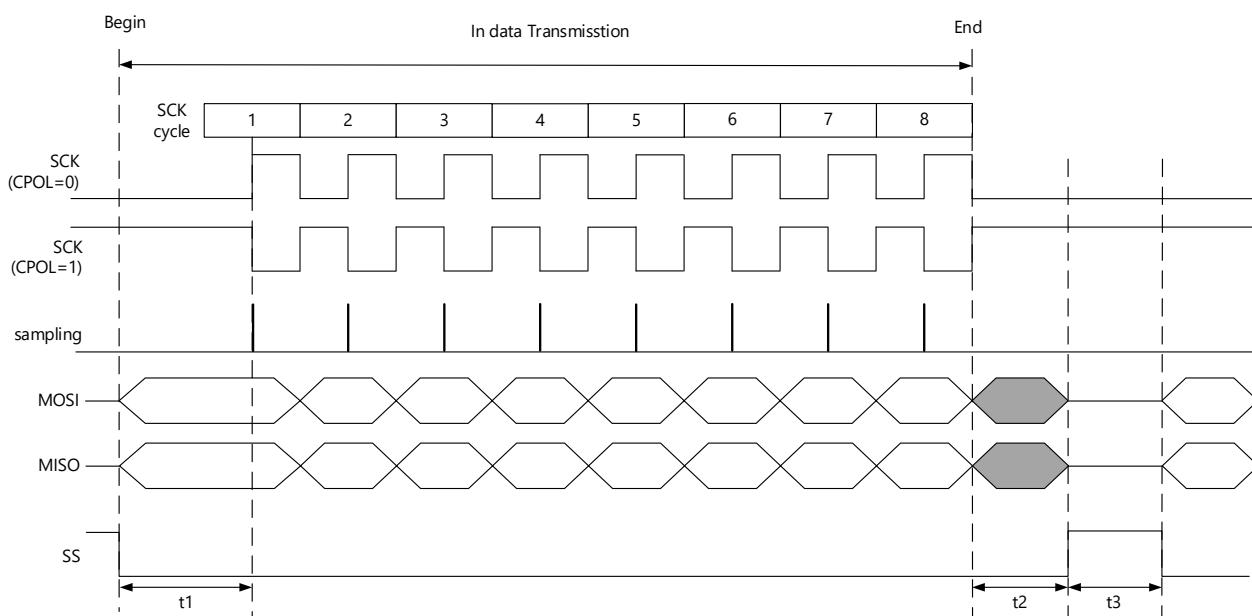


Figure 31-9 Data transfer format diagram (CPHA=0)

In the above figure, t1 represents the interval from the SS signal becoming valid to the first edge of SCK (SCK delay time, by setting SPI_CFG1.MSSI[2:0] and SPI_CFG2.MSSIE), t2 represents the interval from the last edge of SCK to SS signal becoming invalid (SS invalid delay time, by setting SPI_CFG1.MSSDL[2:0] and SPI_CFG2.MSSDLE), t3 represents the minimum waiting time from the end of this transmission to the beginning of the next transmission (next access delay, by setting SPI_CFG1.MIDI[2:0] and SPI_CFG2.MIDIE).

- CPHA = 1

When SPI_CFG2.CPHA = 1, SPI updates the data at the odd edge of SCK and samples at the even edge. Figure 31-10 is the transmission format diagram of SPI when CPHA=1. When the SS signal becomes valid, MOSI/MISO begins to transmit the data at the first SCK signal. After this, the data is updated once per SCK. The data is sampled at 1/2 SCK after each sampling. The value of SPI_CFG2.CPOL does not affect the sequence of SCK, but only affects the signal polarity.

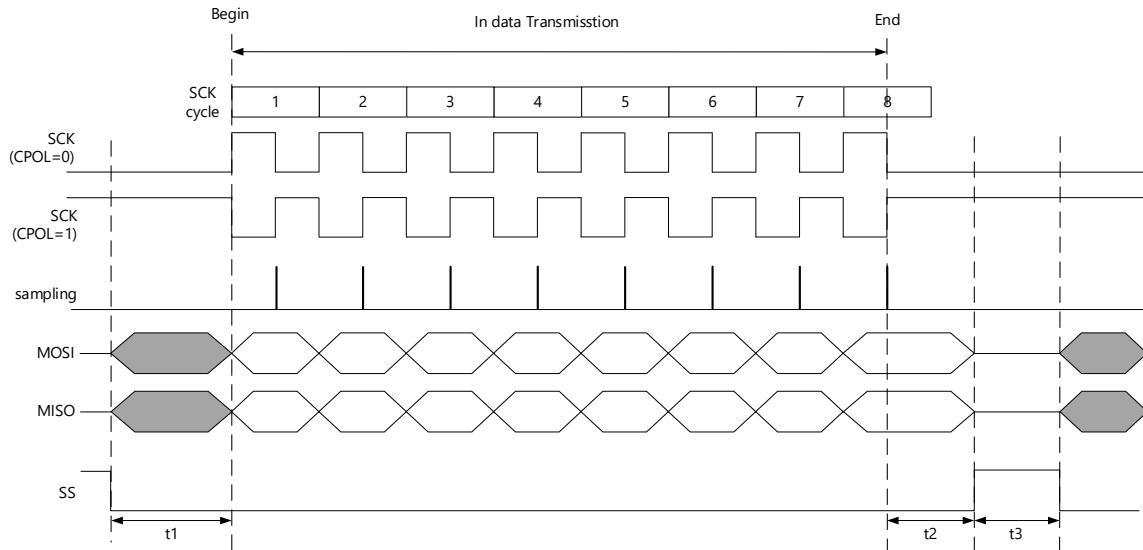


Figure 31-10 Data transfer format diagram (CPHA=1)

The t1, t2, t3 are the same sequence as CPHA=0..

31.5.4 Communication mode

SPI has two communication modes: Full-duplex synchronous serial communication and serial communications with transmit-only, which can be selected through the TXMDS bit of SPI control register (SPI_CR).

- Full-duplex synchronous serial communication mode

When SPI_CR.TXMDS=0, SPI runs in full-duplex synchronous serial communication. As shown in Figure 31-11, when SPI_CFG1.FTHLV[1:0] =00 and SPI_CFG2.CPHA=1 and SPI_CFG2.CPOL =0, then SPI runs as 8-bit serial transfer.

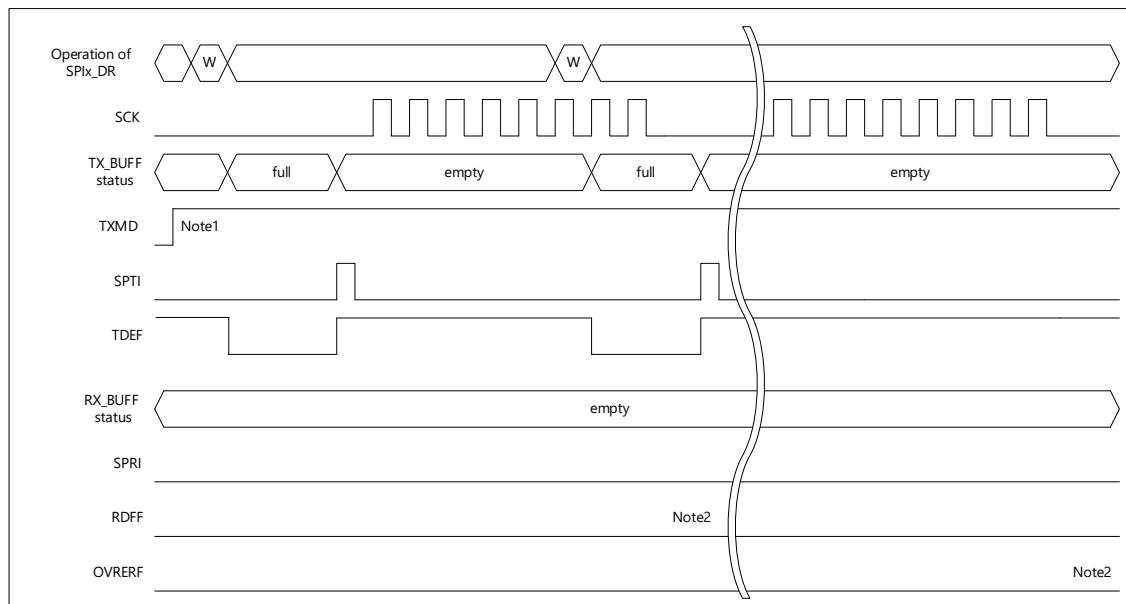


Figure 31-11 Full-duplex synchronous serial communication

Note:

- When this transfer end, if RX_BUFF is empty, SPI will copy the received data from SHIFTER to RX_BUFF, and then set the receive buffer full flag (SPI_SR.RDFF) to 1, generate an interrupt request (SPI_SPRI) with the received data full.
 - When this transfer end, if the last data is kept in the receive buffer but not read out, SPI will set the data overflow error flag (SPI_SR.OVRERF) to 1. The data in SHIFTER will be discarded.
- Serial communications with transmit-only

When SPI_CR.TXMDS=1, SPI runs in Serial communications with transmit-only. As shown in Figure 31-12 , SPI_CFG1.FTHLV[1:0] =00 and SPI_CFG2.CPHA=1 and SPI_CFG2.CPOL=0. SPI runs as 8-bit serial transfer.

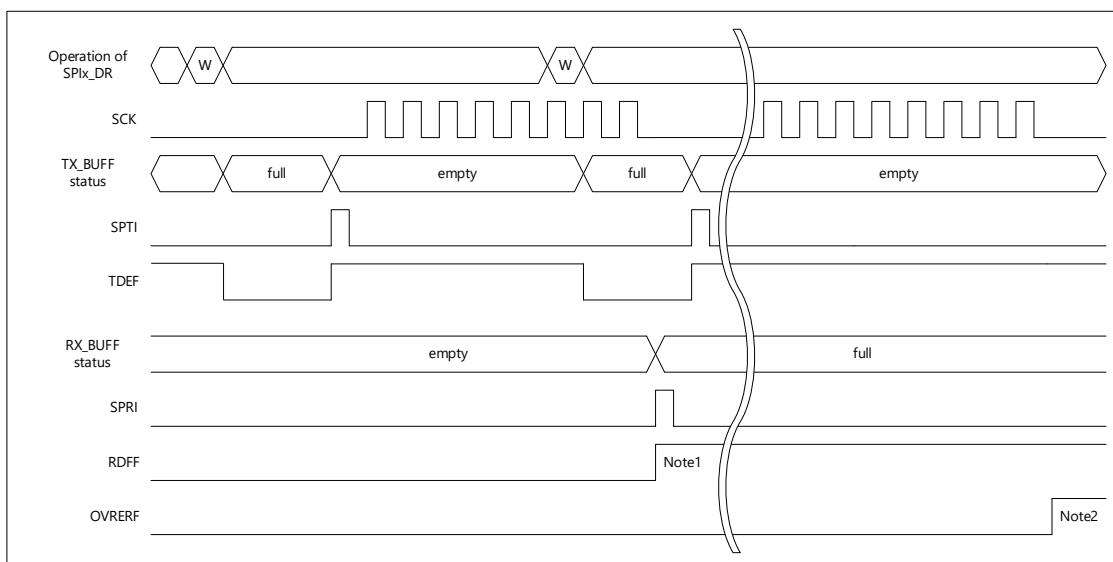


Figure 31-12 Serial communications with transmit-only

Note:

- Make sure that there is no unread data in RX_BUFF (SPI_SR.RDFF=0) and no data overflow error (SPI_SR.OVRERF=0) before setting into the transmit-only communication.
- In transmit-only communication mode, when this transmission ends, even if RX_BUFF is empty, no data will be received, SPI_SR.RDFF and SPI_SR.OVRERF flag will always “0”.

31.6 Operating instructions

31.6.1 Operational mode summary

SPI supports 4 wire type SPI operation mode and 3 wire clock synchronous operation mode. Serial communication can be performed as master or slave in each operation mode. Mode can be set according to SPI_CR.MSTR and SPI_CR.SPIMDS. Refer to Table 31-6 for setting.

Table 31-6 The relationship of SPI operation mode and register setting

Pattern	SPI operation mode		Clock synchronous operation mode	
	Master	Slave	Master	Slave
MSTR	1	0	1	0
SPIMDS	0	0	1	1
SCK(Note1)	Output	Input	Output	Input
MOSI(Note1)	Output	Input	Output	Input
MISO(Note1)	Input	Output/Hi-Z	Input	Output
SS0(Note1)	Output	Input	Hi-Z (not used)	Hi-Z (not used)
SS1-SS3(Note1)	Output	Hi-Z	Hi-Z (not used)	Hi-Z (not used)
SS Polarity selection Function	Supported	Supported	-	-
Maximum transfer rate	PCLK1/2	PCLK1/6	PCLK1/2	PCLK1/6
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 phase	2 kinds	2 kinds	2 kinds	1 kind (CHPA=1)
Data format	MSB/LSB	MSB/LSB	MSB/LSB	MSB/LSB
Data length	4-32 bits	4-32 bits	4-32 bits	4-32 bits
SCK delay control	Supported	Not supported	Supported	Not supported
SS invalid delay control	Supported	Not supported	Supported	Not supported
Next access delay control	Supported	Not supported	Supported	Not supported
Transfer start method	Request the transmit buffer empty interrupt and then write TX BUFF	SS Input Valid or SCK clock Edge input	Request the transmit buffer empty interrupt and then write TX_BUFF	SCK clock edge input
Mode fault error detection	Not supported	Supported	Not supported	Supported
Transmit buffer empty detection	Supported	Supported	Supported	Supported
Receive buffer full detection	Supported (Note 2)	Supported (Note 2)	Supported (Note 2)	Supported (Note 2)
Data overflow error detection	Supported (Note 2)	Supported (Note 2)	Supported (Note 2)	Supported (Note 2)
Parity error detection	Supported (Note 2, Note 3)	Supported (Note 2, Note 3)	Supported (Note 2, Note 3)	Supported (Note 2, Note 3)
Data underflow error detection	Not supported	Supported	Not supported	Supported

Note:

1. Output can be controlled as CMOS output or NMOS open-drain by PCRxy.NOD. For details

refer to chapter [General IO]..

2. When SPI_CR.TXMDS=1, receive buffer full detection, data overflow error detection and parity error detection are not supported..
3. When SPI_CR.PAE=0, parity error detection is not supported.

31.6.2 Action of master device in SPI Operation Mode

■ Operation description as master

When the transmit buffer is empty (SPI_SR.TDEF=1), after the data (length is set by SPI_CFG1.FTHLV[1:0]) is written to SPI_DR, SPI will copy the data to the transmit buffer (TX_BUFF). If the shift register (SHIFTER) is empty, SPI will copy the TX_BUFF data to SHIFTER, then start transmission.

When SPI sends the last sampling edge of SCK, the transmission ends. When the receive buffer (RX_BUFF) is empty, SPI will copy the data to RX_BUFF after this transmission. The data can be read by accessing SPI_DR..

The final sampling sequence depends on the length of the data transmitted. And the data length depends on the value of SPI_CFG2.DSIZE[3:0]. The polarity of the SS pin depends on the value of SPI_CFG1. For more information, , refer to Transfer format.32.5.3

■ Initialization of Master in SPI operation Mode

- 1) Set SPI_CFG1, including the baud rate, frames, and each delay times setting, etc. Then confirm the setting.
- 2) Set SPI_CFG2, including SS level, each delay allowable bit setting, data format, polarity and phase of clock setting, etc. Then confirm the setting.
- 3) Set the interrupt register if needed, refer to Chapter Interrupt
- 4) Set the relevant register if data is needed to transfer through DMA, refer to Chapter DMA.
- 5) Set the input/output pin.
- 6) Set SPI_CR, including the mode setting, self-diagnosis function setting, parity setting, etc. Then confirm the setting.
- 7) Clear all flags.
- 8) Set the interrupt permission bit.
- 9) Set SPI_CR.SPE=1 and then the communication starts.

31.6.3 Action of Slave device in SPI Operation Mode

■ Operation description as slave

When SPI_CFG2.CPHA=0, if SPI detects SS0 becoming valid, it will begin to output valid data to MISO. Therefore, the SS0 level from invalid to valid is considered as a trigger signal to start transmission.

When SPI_CFG2.CPHA=1, if SS0 becomes valid and then SPI detects the first edge of SCK, it will begin to output valid data to MISO. Therefore, the first edge of SCK is considered as the trigger signal to start transmission.

When SHIFTER is empty, if SPI detects this transmission has started, it will change the state of SHIFTER to full and cannot transfer data from TX_BUFF to SHIFTER during the transmission. If SHIFTER is full before starting transmission, SPI will keep SHIFTER full state.

If SPI detects the last sampling edge of the SCK, then this transmission will be end. When the receive buff (RX_BUFF) is empty, SPI will copy the received data of SHIFTER to RX_BUFF after this transmission. The data can be read by accessing SPI_DR. SPI will change SHIFTER to empty state after this transmission. This state is independent of RX_BUFF.

During this transmission, if SPI detects the SS0 is invalid, the mode fault error will be occurred.

The final sampling sequence depends on the length of the transmitted data. The data length in slave mode depends on the value of SPI_CFG2.DSIZE[3:0], and the polarity of SS0 depends on the value of SPI_CFG1.SS0PV. For more information, refer to Transfer format.32.5.3

Note:

- When SPI_CFG2.CPHA=0, SS0 will be changed from invalid to valid as a trigger signal to start transmission. The SS0 signal is fixed in valid level in slave mode, SPI cannot start to transmission at this time. In this case the SPI_CFG2.CPHA bit must be set to "1" in order to transmit and receive in slave mode. If SPI_CFG2.CPHA =0 in need, the SS0 input signal cannot be fixed.
- Initialization of Slave in SPI operation Mode
 - 1) Set SPI_CFG1, including the number of frames setting. Then confirm the setting.
 - 2) Set SPI_CFG2, including transmission rate, data format, polarity and phase of clock setting. Then confirm the setting.
 - 3) Set the interrupt register if needed, refer to Chapter Interrupt.
 - 4) Set the relevant register if data is needed to transfer through DMA, refer to Chapter DMA.
 - 5) Set the input/output pin.
 - 6) Set SPI_CR, including the mode setting, self-diagnosis function setting, parity setting, etc. Then confirm the setting.
 - 7) Clear all flags.
 - 8) Set the interrupt permission bit.
 - 9) Set SPI_CR.SPE=1 and the communication starts.

31.6.4 Action of Master device in Clock Synchronous Operation Mode

SPI runs in clock synchronous operation mode when SPI_CR.SPIMDS=1. When operating in this mode, SPI only uses the 3 pins of SCK, MOSI and MISO for communication. The SS pin is released and can be used for GPIO ports.

Although the SS pin are not used in clock synchronous operation mode, the internal operation of the module is the same as the SPI operation mode.

■ Operation description as Master

When the transmit buffer (TX_BUFF) is empty (SPI_SR.TDFF=0), SPI will write the data (length is set by SPI_CFG1.FTHLV[1:0]) to SPI_DR, then copy data to TX_BUFF,. If the shift register (SHIFTER) is empty, SPI will copy the TX_BUFF data to SHIFTER, then start transmission.

When SPI sends the last sampling edge of SCK, the transmission ends. When the receive buffer (RX_BUFF) is empty, SPI will copy the data to RX_BUFF after this transmission, the data can be read by accessing SPI_DR.

The final sampling sequence depends on the length of the data transmitted. And the data length in master mode depends on the value of SPI_CFG2.DSIZE[3:0]. For more information, refer to Transfer format.32.5.3

■ Initialization of Master in Clock Synchronous Operation Mode

- 1) Set SPI_CFG1, including the baud rate, frames, and each delay times, etc. Then confirm the setting.
- 2) Set SPI_CFG2, including each delay allowable bits setting, data format, polarity and phase of clock, etc. Then confirm the setting.
- 3) Set the interrupt register if needed, refer to Chapter Interrupt.
- 4) Set the relevant register if data is needed to transfer through DMA, refer to Chapter DMA.
- 5) Set the input/output pin.
- 6) Set SPI_CR, including the mode setting, self-diagnosis function setting, parity setting, etc. Then confirm the setting.
- 7) Clear all flags.
- 8) Set the interrupt permission bit.
- 9) Set SPI_CR.SPE=1 and then the communication starts.

31.6.5 Action of Slave device in Clock Synchronous Operation Mode

■ Operation description as slave

When SPI_CFG2.CPHA=0, SPI will detect the SS0 pin becoming valid as a trigger signal to start transmission. But in clock synchronous operation mode, the SS0 pin is not used. So transmission cannot be performed in this case.

When SPI_CFG2.CPHA=1, if SS0 becomes valid and then SPI detects the first edge of SCK, it will begin to output valid data to MISO. But in clock synchronous operation mode, the SS0 pin is not used, if SPI_CFG2.CPHA=1 is detected, the first SCK edge is regarded as the trigger signal to start transmission.

If SPI detects the last sampling edge of the SCK, then this transmission will be end. When the receive buff (RX_BUFF) is empty, SPI will copy the received data of SHIFTER to RX_BUFF after this transmission. The data can be read by accessing SPI_DR. SPI will change SHIFTER to empty state after this transmission. This state is independent of RX_BUFF.

The final sampling sequence depends on the length of the transmitted data. The data length in slave mode depends on the value of SPI_CFG2.DSIZE[3:0].

- Initialization of Slave in Clock Synchronous Operation Mode
 - 1) Set SPI_CFG1, including the number of frames setting. Then confirm the setting.
 - 2) Set SPI_CFG2, including transmission rate, data format, polarity and phase of clock setting. Then confirm the setting.
 - 3) Set the interrupt register if needed, refer to Chapter Interrupt.
 - 4) Set the relevant register if data is needed to transfer through DMA, refer to Chapter DMA.
 - 5) Set the input/output pin.
 - 6) Set SPI_CR, including the mode setting, self-diagnosis function setting, parity setting, etc. Then confirm the setting.
 - 7) Clear all flags.
 - 8) Set the interrupt permission bit.
 - 9) Set SPI_CR.SPE=1 and then the communication starts.

31.6.6 Examples of SPI communication processing flow

- SPI data transfer process as master
 - 1) Wait for the interrupt of the transmit buffer empty or confirm SPI_SR.TDEF=1 by polling.
 - 2) Write the transmission data to SPI_DR.
 - 3) Repeat step 1)-2), until the last data was transmitted.
 - 4) Clear the Transmit interrupt enable bit (SPI_CR.TXIE), and set the SPI idle interrupt enable (SPI_CR.IDIE) to 1.
 - 5) Hardware generates SPI idle state interrupt.
 - 6) Set SPI_CR.SPE to 0, stop SPI operation, and clear SPI_CR.IDIE.
- Data receiving and processing flow
 - 1) Wait for the interrupt of the receive buffer full or confirm SPI_SR.RDFF=1 by polling.
 - 2) Read data from the receive buffer by accessing SPI_DR.
 - 3) Repeat step 1)-2), until the last received data is read.
 - 4) Set SPI_CR.SPE to 0, stop SPI operation, and clear the receive interrupt enable

(SPI_CR.RXIE).

■ Communication error handling process

- 1) Wait for the SPI error interrupt or confirm the communication error flag bit (MODFERF/OVERRF/UDRERF/PERF) is set to 1 by polling.
- 2) Confirm the status of SSO to correct the mode fault errors.
- 3) Clear SPE and stop SPI operation.
- 4) Identify the types of SPI error by error flag and handle them.
- 5) Clear the error flag.
- 6) Start SPI and restart the communication.

31.7 Self-diagnosis of parity

Parity check function is composed of parity addition unit for transmitting data and error detection unit for receiving data. An example of parity check with self-diagnosis function is shown in the following figure.

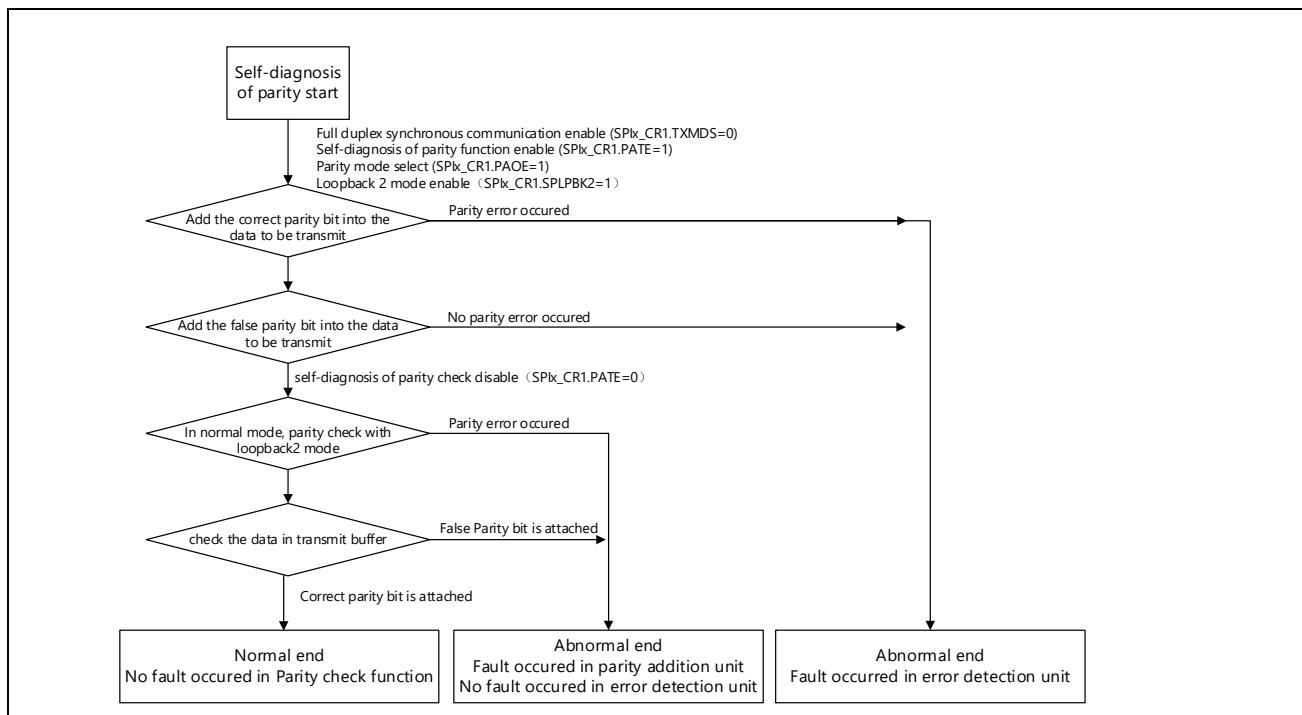


Figure 31-13 Parity check flow

31.8 Error detection

In the normal transfer, the data is sent serially by writing SPI_DR and the data received serially is obtained by reading SPI_DR. However, due to the state of the transmit buffer / receive buffer and the state of the SPI at the beginning or end of serial transfer, abnormal transfer may occur in some cases. When an abnormal transfer occurs, SPI detects this transfer as a data underflow error, a data overflow error, a parity error, or a mode fault error. The corresponding relationship between abnormal transfer and SPI error detection follows Table 31-7.

Table 31-7 Error detection correspondence table

No	condition	SPI operation	rrors
①	Write SPI_DR when transmit buffer is full	<ul style="list-style-type: none"> The data in transmit buffer is not changed The data write to SPI_DR is missing 	None
②	Read SPI_DR when Receive Buffer is empty	read last data in SPI_DR	
③	In slave mode, the transfer starts when the shift register is empty	<ul style="list-style-type: none"> Transfer is suspended Stop driving MISO output Stop SPI operation 	Data underflow error
④	In Slave mode, the effective level width of the SS0 pin has not reached the time required for data transfer.	<ul style="list-style-type: none"> Transfer is suspended Transmitting and receiving data loss Stop SPI operation 	Mode fault error
⑤	End transfer when receive buffer is full	<ul style="list-style-type: none"> The data in Receive Buffer is not changed receiving data loss 	Data overflow error
⑥	When the parity function is effective in Full-duplex synchronous serial communication, received error parity bit	Parity error flag is set to 1	Parity error

31.8.1 Data Underflow error

When SPI_CR.MSTR=0, SPI runs in slave mode. If SPI_CR.SPE=1, the transmit data is not ready before the SS0 pin is valid, then a data underflow error occurs. SPI_SR.MODFERF and SPI_SR.UDRERF will be set to 1. The SPI_SR.UDRERF flag can be cleared by writing it to 0 after reading its state as 1.

When a data underflow error is detected, SPI will stop driving signal output, and set SPI_CR.SPE to 0.

Data underflow error can be detected by directly accessing SPI_SR or read SPI_SR in error interrupt handle.

When SPI_SR.MODFERF = 1, SPI_CR.SPE cannot be written to 1. To enable SPI function by setting SPI_CR.SPE to 1, SPI_SR.MODFERF must be cleared first.

31.8.2 Mode fault error

In master mode, do not set SPI_CR.MODFE to 1.

In slave mode, the effective level width of SS0 pin does not reach the time required for data transfer, then a mode fault error occurs. SPI_SR.MODFERF will be set to 1 and SPI_CR.SPE will be set to 0.

When data underflow error is occurred, SPI_SR.MODFERF and SPI_SR.UDRERF will be set to 1.

The SPI_SR.MODFERF flag can be cleared by writing it to 0 after reading its state as 1.

When SPI_SR.MODFERF = 1, SPI_CR.SPE cannot be written to 1. To enable SPI function by setting SPI_CR.SPE to 1, SPI_SR.MODFERF must be cleared first.

31.8.3 Data overflow error

If the transfer ends when the receive buffer is full, a data overflow error occurs, and SPI_SR.OVRERF will be set to 1. When SPI_SR.OVRERF = 1, SPI does not copy the data from the shift register (SHIFTER) to the receive buffer (RX_BUFF). Therefore, the data received before the error is retained in RX_BUFF.

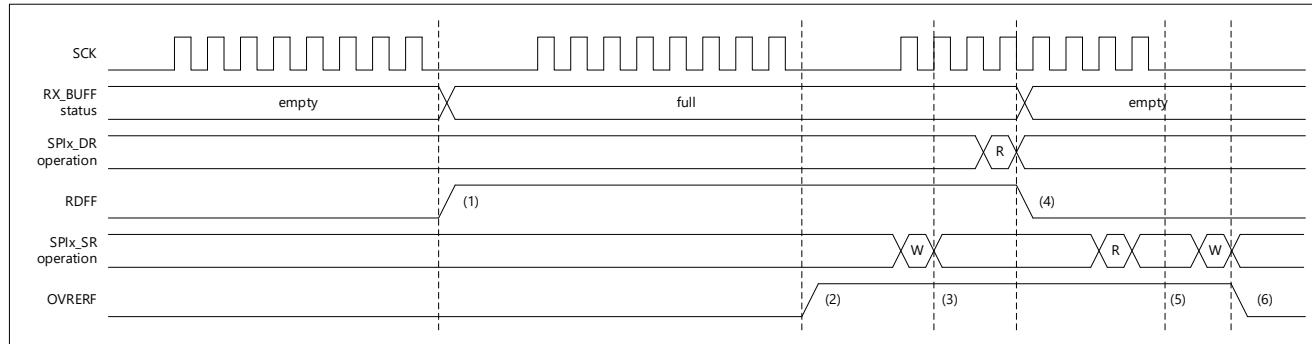


Figure 31-14 Data overflow error handling

The annotations shown in step ① to ⑥ in the figure are described below:

- ① If the transfer ends when the receive buffer is empty, SPI operates normally, copies the data from the shift register (SHIFTER) to the receive buffer (RX_BUFF), and sets SPI_SR.RDFF to 1.
- ② If the transfer ends when the receive buffer is full, SPI detects the Data overflow error and sets SPI_SR.OVRERF to 1. SPI will not copy the data from SHIFTER to RX_BUFF. Even when SPI_CR.PAE=1, no parity error is detected.
- ③ The SPI_SR.OVRERF flag can be cleared by writing it to 0 after reading its state as 1.
- ④ Read SPI_DR when SPI_SR.OVRERF =1 and SPI_SR.RDFF = 1, SPI_SR.RDFF will become 0. Even if the receive buffer is empty at this time, SPI_SR.OVRERF will not be cleared.
- ⑤ If the transfer ends when SPI_SR.OVRERF = 1, SPI will not copy the data from SHIFTER to RX_BUFF, and do not generate the interrupt of the receive buffer full. SPI_SR.RDFF keep 0. Even if SPI_CR.PAE=1, no parity error is detected.
Only after SPI_SR.OVRERF is cleared, SPI can receive normally.

In master mode, if the communication auto suspend function is enable (SPI_CR.CSUSPE=1), SPI will suspend the transfer clock in the last sampling period before a data overflow error occurs. At this time, SHIFTER has not completed the acceptance of the last bit. SPI remains in the normal communication state, and the data overflow error will not occur. The receive buffer can be read during the transfer clock pause. After reading, the receive buffer status becomes empty, and SPI restarts the transfer clock to complete data receiving of the last bit. For details, refer to the following Figure 31-15 and Figure 31-16 .

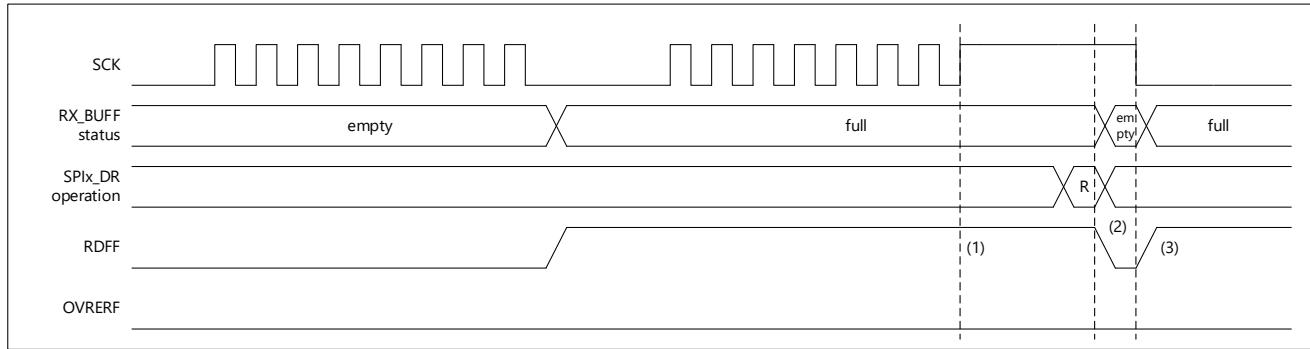


Figure 31-15 Schematic diagram of the operation when the automatic clock stop function is enabled (CPHA=1)

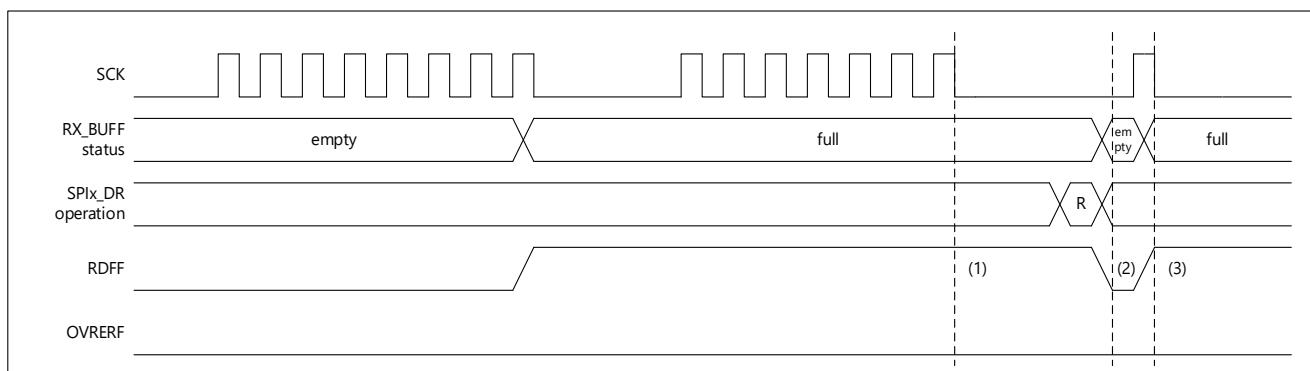


Figure 31-16 Schematic diagram of the operation when the automatic clock stop function is enabled (CPHA=0)

The annotations shown in step ① to ③ in the figure are described below:

- ① When the receive buffer (RX_BUFF) is full, SPI suspends the transfer clock before the last bit of data is received. No data overflow error will occur at this time.
- ② After reading the data in RX_BUFF by accessing SPI_DR, RX_BUFF becomes empty, SPI_SR.RDFF is cleared, and SPI restarts the transfer clock to complete the data transfer for the last bit.
- ③ The last bit of data transfer is completed, RX_BUFF becomes full again, SPI_SR.RDFF is set to 1, and the received data can be read by accessing SPI_DR.

31.8.4 Parity error

When SPI_CR.TXMDS=0 and SPI_CR.PAE=1, SPI will perform parity check at the end of full-duplex synchronous serial communication. When SPI detects a parity error in the received data, it sets SPI_SR.PERF to 1. An example of SPI_SR.OVRERF and SPI_SR.PERF shown in follows Figure 31-17. In the figure, SPI performs 8-bit serial transmission of full-duplex synchronous serial communication when SPI_CR.TXMDS=0 and SPI_CR.PAE=1.

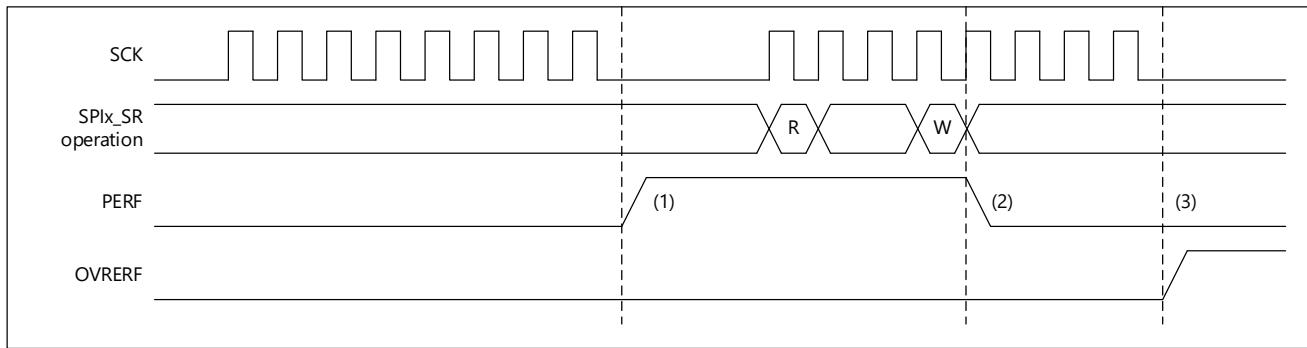


Figure 31-17 Parity error

The annotations shown in step ① to ③ in the figure are described below:

- ① SPI did not detect the data overflow error, transfer ended normally. SPI copies the data from the shift register (SHIFTER) to the receive buffer (RX_BUFF). At this time, SPI performs parity check on the received data. If a parity error is detected, SPI_SR.PERF is set to 1.
- ② The SPI_SR.PERF flag can be cleared by writing it to 0 after reading its state as 1.
- ③ When SPI detects a data overflow error, SPI does not copy the data from SHIFTER to RX_BUFF. SPI does not perform parity check, so no parity check error occurs.

Parity error can be detected by accessing SPI_SR or using error interrupt to access SPI_SR.

31.9 SPI Initialization

To clear SPI_CR.SPE by writing register or mode fault error occurs, SPI function is disabled and some SPI functions can be initialized.

31.9.1 Clear SPE for initialization

When the SPI_CR1.SPE = 0, SPI is initialized as follows:

- Suspend the transfer in progress.
- If operation in slave mode, stop driving the output signal (the state changes to Hi-Z).
- SPI internal state is initialized.
- Clear the transmit buffer (TX_BUFF), SPI_SR.TDEF set to 1.

When SPI_CR.SPE is initialized by clearing, the control register of SPI will not be initialized. Therefore, SPI can be started in the same transfer mode as before initialization, as long as SPI_CR.SPE is reset to 1.

Error flags does not initialized if clear SPI_CR.SPE. Therefore, even after SPI_CR.SPE is cleared, the error condition during SPI transfer can be confirmed by reading the data in RX_BUFF.

Clearing SPI_CR.SPE will clear TX_BUFF and set SPI_SR.TDEF to 1. Therefore, if SPI_CR.TXIE is set to 1 after initialization, an interrupt with the transmit buffer empty will be generated. To avoid this interrupt, SPI_CR.TXIE must be set to 0 before SPI_CR.SPE is cleared.

31.9.2 System reset initialization

All SPI registers are initialized by system reset.

31.10 Interrupt

Receive buffer full, transmit buffer empty, SPI error (mode fault, overflow, underflow, parity error) and SPI idle can be used as interrupt source or internal trigger source, and the transfer complete can only be used as internal trigger source.

The interrupt of mode fault, data overflow, data underflow and parity errors are integrated into the SPI error interrupt (SPI_SPEI), so the actual interrupt source needs to be judged by the flag. The specific description of SPI interrupt/internal trigger source follows Table 31-8. Once the condition is established, a corresponding interrupt/internal trigger request is generated. For receive buffer full and transmit buffer empty interrupt sources, the flag is cleared by buffer data read/write access.

Table 31-8 SPI Interrupt Source Description

Interrupt/Internal Trigger Source	Sketch	Conditions
Receiver buffer full	SPRI	When SPI_CR.RXIE=1, the receive buffer becomes full.
Transmit buffer empty	SPTI	When SPI_CR.TXIE=1, the transmit buffer becomes empty.
SPI error (mode fault, data overflow, data underflow, parity error)	SPEI	when SPI_CR.EIE=1, SPI_SR.OVRERF, SPI_SR.PERF, SPI_SR.MODFERF or SPI_SR.UDRERF is set to 1.
SPI idle	SPII	when SPI_CR.IDIE=1 Master mode: Data transfer is complete or SPI_CR.SPE is cleared. Slave mode: SPI_CR.SPE is cleared.

31.11 Available event trigger sources

SPI produces the following types of event trigger sources available:

- Transmit buffer empty
- Receive buffer full
- SPI error (including mode fault, data overflow, data underflow, parity error)
- SPI is idle
- The transmit end

Customers can write the vector corresponding to the event trigger source into different trigger object registers to implement various event trigger functions.

For the vector corresponding to the event trigger source, refer to [Interrupt controller (INTC)].

31.12 Register description

Register base address:

SPI1_BASE: 0x4001C000; SPI2_BASE: 0x4001C400

SPI3_BASE: 0x4001C800; SPI4_BASE: 0x40020000

SPI5_BASE: 0x40020400; SPI6_BASE: 0x40020800

Table 31-9 List of SPI registers

Symbol	Register name	Address offset (hex)
SPI_DR	SPI data register	0x00
SPI_CR	SPI control register	0x04
SPI_CFG1	SPI communication configuration register 1	0x0C
SPI_SR	SPI status register	0x14
SPI_CFG2	SPI communication configuration register 2	0x18

31.12.1 SPI data register (SPI_DR)

This register is used to store data.

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]															
<hr/>															
Bit	Symbol	Bit name	Description	Read and write											
b31~b0	SPD[31:0]	Serial data	SPI Data Storage	R/W											

31.12.2 SPI control register (SPI _ CR)

This register is used to control the SPI function.

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	MOD FE	IDIE	RXIE	TXIE	EIE	CSU SPE	SPE	SPLP BK2	SPLP BK	MST R	-	TXM DS	SPIM DS

Bit	Marking	Bit name	Description	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15	PAE	Parity permission	Parity check enable 0: Transmit data without parity bit, receive data without parity check 1: Transmit data with parity bit, receive data with parity check (When SPI_CR.TXMDS=0).Transmit data with parity bit, receive data without parity check (When SPI CR.TXMDS=1).	R/W
b14	PAOE	Parity mode selection	Parity odd/even select 0: Select even parity check for transmitting and receiving 1: Select odd parity check for transmitting and receiving	R/W
b13	PATE	Parity self-diagnosis	Parity self-diagnosis enable 0: Disable self-diagnosis of parity function 1: Enable self-diagnosis of parity function	R/W
b12	MODFE	Mode fault error detection permission	Mode fault error detection enable 0: Disable mode fault error detection 1: Enable mode fault error detection	R/W
b11	IDIE	SPI Idle Interrupt Allowed	SPI idle interrupt enable 0: Disable idle interrupt request generation 1: Enable idle interrupt request generation	R/W
b10	RXIE	SPI Receive Interrupt Permission	Receive interrupt enable 0: Disable receive interrupt request generation 1: Enable receive interrupt request generation	R/W
b9	TXIE	SPI transmit Interrupt permission	Transmit interrupt enable 0: Disable transmit interrupt request generation 1: Enable transmit interrupt request generation	R/W
b8	EIE	SPI error interrupt permission	SPI error interrupt enable 0: Disable SPI error interrupt request generation 1: Enable SPI error interrupt request generation	R/W
B7	CSUSPE	Communication auto-suspend feature allows	Communication auto suspend enable 0: Disable communication auto suspend function 1: Enable communication auto suspend function	R/W
b6	SPE	SPI function Allowed	SPI function enable 0: Disable SPI function 1: Enable SPI function	R/W
b5	SPLPBK2	SPI Loopback2 mode selection	Loopback 2 mode select 0: Normal mode 1: Loopback 2 mode (transmitted data=received data)	R/W
b4	SPLPBK	SPI loopback mode selection	Loopback mode select 0: Normal mode 1: Loopback mode (inverse of transmitted data=received data)	R/W
b3	MSTR	SPI Master-Slave Mode Selection	SPI mode select 0: Slave mode 1: Master mode	R/W
b2	Reserved	-	Read as "0", write as "0"	R/W
b1	TXMDS	Communication mode selection	Communication mode select 0: Full-duplex synchronous serial communication 1: Serial communication with transmit-only	R/W
b0	SPIMDS	SPI mode selection	Operation mode select 0: SPI operation mode (4 wire type) 1: Clock synchronous operation mode (3 wire type)	R/W

31.12.3 SPI Communication Configuration Register1 (SPI_CFG1)

This register is used to configure parameters for communication.

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
-	-	-	-	SS3P V	SS2P V	SS1P V	SS0P V	-	SPR DTD	-	-	-	-	-	FTHLV[1:0]

Bit	Symbol	Bit name	Description	Read and write
b31	Reserved	-	Read as "0", write as "0"	R/W
b30~b28	MIDI[2:0]	The master next access delay time setting bit	Next access delay setting: 0 0 0: 1 SCK+2 PCLK1 0 0 1: 2 SCK+2 PCLK1 0 1 0: 3 SCK+2 PCLK1 0 1 1: 4 SCK+2 PCLK1 1 0 0: 5 SCK+2 PCLK1 1 0 1: 6 SCK+2 PCLK1 1 1 0: 7 SCK+2 PCLK1 1 1 1: 8 SCK+2 PCLK1	R/W
b27	Reserved	-	Read as "0", write as "0"	R/W
b26~b24	MSSDL[2:0]	The master SS invalid delay time setting bit	SS invalid delay setting: 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	R/W
b23	Reserved	-	Read as "0", write as "0"	R/W
b22~b20	MSSI[2:0]	The master SCK delaytime setting bit	SCK delay setting: 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	R/W
b19~b12	Reserved	-	Read as "0", write as "0"	R/W
b11	SS3PV	SS3 Polarity setting	Polarity of SS3 setting 0: Low level of SS3 signal is valid 1: High level of SS3 signal is valid	R/W
b10	SS2PV	SS2 Polarity setting	Polarity of SS2 setting 0: Low level of SS2 signal is valid 1: High level of SS2 signal is valid	R/W
b9	SS1PV	SS1 Polarity setting	Polarity of SS1 setting 0: Low level of SS1 signal is valid 1: High level of SS1 signal is valid	R/W
b8	SS0PV	SS0 Polarity setting	Polarity of SS0 setting 0: Low level of SS0 signal is valid 1: High level of SS0 signal is valid	R/W
B7	Reserved	-	Read as "0", write as "0"	R/W
b6	SPRD TD	Data buffer read selection	read data buffer select 0: SPI_DR read receive buffer 1: SPI DR read transmit buffer (only when TDEF=1)	R/W
b5	Reserved	-	Read as "0", write as "0"	R/W
b4	Reserved	-	Read as "1", write as "1"	R/W
b3~b2	Reserved	-	Read as "0", write as "0"	R/W

b1~b0	FTHLV[1:0]	frame setting	frame setting: 0 0: 1 frame 0 1: 2 frames 1 0: 3 frames 1 1: 4 frames	R/W
-------	------------	---------------	---	-----

31.12.4 SPI status register (SPI_SR)

This register is used to detect the status of SPI.

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
-	-	-	-	-	-	-	-	RDFF	-	TDEF	UDRERF	PERF	MODFERF	IDLNRF	OVRERF

Bit	Symbol	Bit name	Description	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
B7	RDFF	Receive buffer full flag	0: Not full 1: Full Hardware set and clear, write "1" when writing	R
b6	Reserved	-	Read as "0", write as "0"	R/W
b5	TDEF	Transmit buffer empty flag	0: Not empty 1: Empty Hardware set, clear, write "1" when writing	R
b4	UDRERF	Data underflow error flag	0: Not occur 1: Occurred (when MODFERF=1) After the hardware is set, read 1 and write 0, the status bit is cleared	R/W
b3	PERF	Parity error flag	0: Not occur 1: Occurred After the hardware is set, read 1 and write 0, the status bit is cleared	R/W
b2	MODFERF	Mode fault error flag	0: Not occur 1: Occurred After the hardware is set, read 1 and write 0, the status bit is cleared	R/W
b1	IDLNRF	SPI Idle status flag	0: Idle status 1: Transfer status hardware set, cleared	R
b0	OVRERF	Overflow error flag	0: Not occur 1: Occurred After the hardware is set, read 1 and write 0, the status bit is cleared	R/W

31.12.5 SPI Communication Configuration Register2 (SPI_CFG2)

This register is used to configure parameters for communication.

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
MSSI E	MSS DLE	MIDI E	LSBF	DSIZE[3:0]		SSA[2:0]		MBR[2:0]		CPOL	CPHA				

Bit	Symbol	Bit name	Description	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15	MSSIE	SCK delay selection	SCK delay select 0: SCK delay is set to 1 SCK 1: SCK delay is set according to the value of MSSI[2:0]	R/W
b14	MSSDLE	SS invalid delay selection	SS invalid delay select 0: SS invalid delay is set to 1 SCK 1: SS invalid delay is set according to the value of MSSDL[2:0]	R/W
b13	MIDIE	SPI next access delay selection	SPI next access delay select 0: The next access delay is set to 1 SCK+2 Pclk 1: The next access delay is set according to the value of MIDI[2:0]	R/W
b12	LSBF	Shift order slection	0: MSB first 1: LSB first	R/W
b11~b8	DSIZE[3:0]	Data Length setting	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: 12 bits 1 0 0 1: 13 bits 1 0 1 0: 14 bits 1 0 1 1: 15 bits 1 1 0 0: 16 bits 1 1 0 1: 20 bits 1 1 1 0: 24 bits 1 1 1 1: 32 bits	R/W
b7~b5	SSA[2:0]	SS signal selection	SS signal select:0 0 0: SS0 0 0 1: SS1 0 1 0: SS2 0 1 1: SS3 1xx: Disable setting	R/W
b4~b2	MBR[2:0]	Bit Rate Frequency setting	0 0 0: Baud rate is frequency of PCLK1/2 0 0 0: Baud rate is frequency of PCLK1/4 0 0 0: Baud rate is frequency of PCLK1/8 0 0 0: Baud rate is frequency of PCLK1/16 0 0 0: Baud rate is frequency of PCLK1/32 0 0 0: Baud rate is frequency of PCLK1/64 0 0 0: Baud rate is frequency of PCLK1/128 0 0 0: Baud rate is frequency of PCLK1/256	R/W
b1	CPOL	SCK polarity set	0: SCK=Low, when SPI is idle 1: SCK=High, when SPI is idle	R/W
b0	CPHA	SCK phase setting	0: The data is sampled on the odd edge and updated on the even edge 1: The data is sampled on the even edge and updated on the odd edge	R/W

32 Quad Serial Peripheral Interface (QSPI)

32.1 Introduction

The Quad Serial Peripheral Interface (QSPI) is a memory control module mainly used to communicate with a serial ROM that has SPI-compatible interface, such as serial flash memory, serial EEPROM and serial FeRAM.

Table 32-1 QSPI main specifications

Parameter	Specifications
Number of channels	1 channel
SPI function	• Support Extended SPI, Dual SPI and Quad SPI and other protocols
	• Support SPI Mode 0 and SPI Mode 3
	• Address width selectable to 8-bit/16-bit/24-bit/32-bit
Timing adjustment	Timing adjustable to support various serial flash
Flash read function	• Support multiple reading methods a. Standard Read/Fast Read b. Fast Read Dual Output/Fast Read Dual I/O c. Fast Read Quad Output/Fast Read Quad I/O
	• Substitutable instructions
	• Adjustable number of dummy cycles
	• 16-byte prefetch function
	• Status query function
	• SPI bus cycle extension function
	• XIP control function
Direct communication function	Flexible and extensive support for a wide variety of serial flash software control instructions, including erase, write, ID read and power-down control, etc.
Interrupt source	Hardware error interrupt
Module stop function	Power consumption can be reduced by module stop

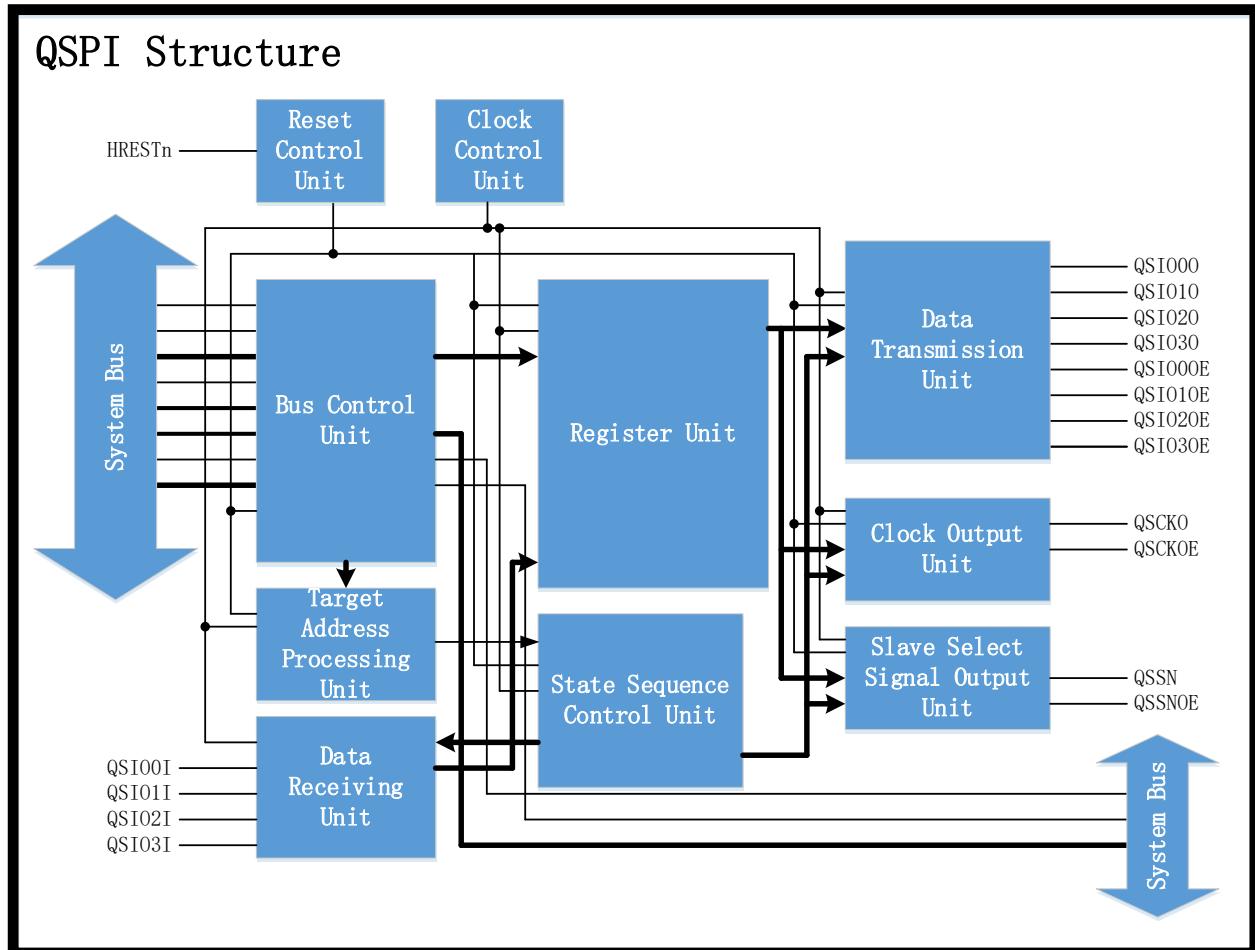


Figure 32-1 Block diagram of QSPI

32.2 Pin Description

Table 32-2 QSPI pins

Pin name	Input/Output	Functional description
QSCK	Output	QSPI clock output pin
QSSN	Output	QSPI slave select pin
QSI00	Input/Output	Data line 0
QSI01	Input/Output	Data line 1
QSI02	Input/Output	Data line 2
QSI03	Input/Output	Data line 3

32.3 Memory Map

32.3.1 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.

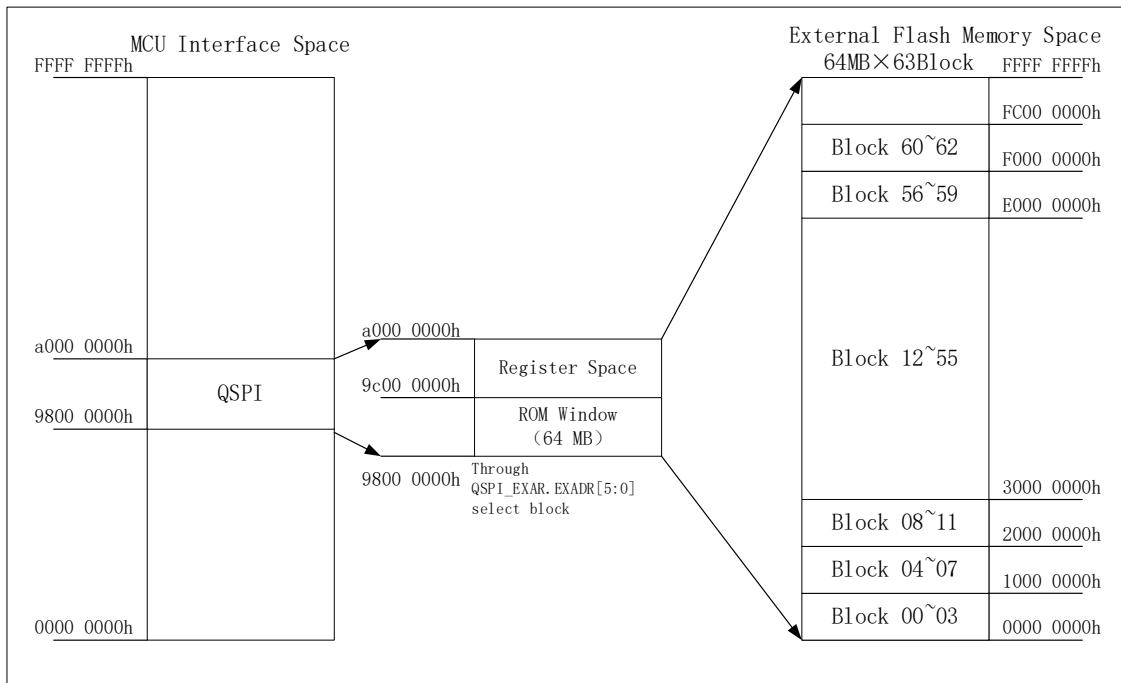


Figure 32-2 Default area setting and AHB bus space memory map

32.3.2 ROM Space and Bus Address Width

QSPI has a 32-bit address width to match the serial flash. Whenever the ROM space of QSPI is read accessed, the QSPI bus automatically starts to work and transfers the data read from the serial flash memory.

QSPI can not only use 32-bit address width, but also can choose to use 8-bit/16-bit/24-bit address width by setting AWSL[1:0] in QSPI_FCR register.

If the 8-bit/16-bit/24-bit address width is selected, only the lower space with matching address can be accessed normally, that is to say, accessing the higher serial flash memory mirror space in QSPI will repeatedly appear the lower space Content.

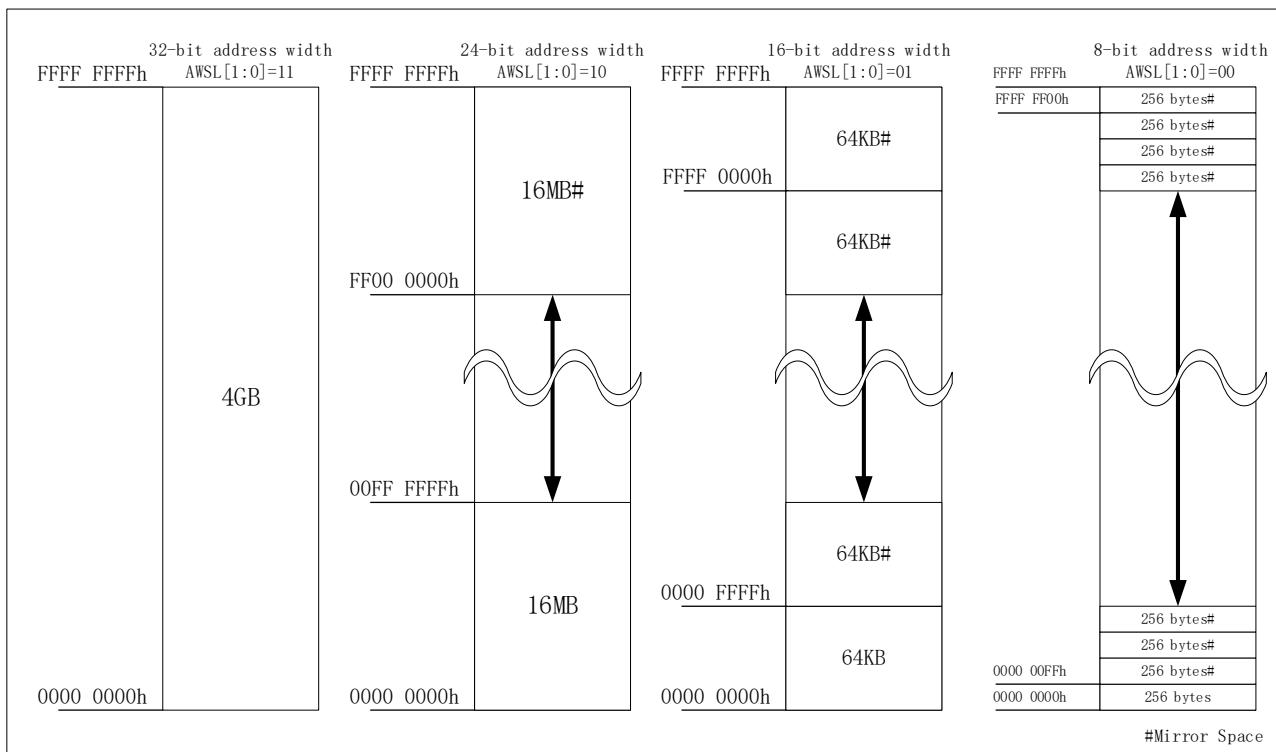


Figure 32-3 QSPI-ROM space memory map

Note:

- The address bus width can be selected to use 8-bit/16-bit/24-bit/32-bit by setting AWSL[1:0] in the QSPI_FCR register.

32.4 QSPI Bus

32.4.1 SPI Protocol

The QSPI supports three protocols: Extended SPI, Dual SPI and Quad SPI. The default protocol is the Extended SPI protocol. The protocol at each stage can be configured separately by setting the DPRSL[1:0]/APRSL[1:0]/IPRSL[1:0] bits in the QSPI_CR register. The Extended SPI protocol output instruction codes from a signal QSIO0 pin, and the subsequent addresses and data use single-pin, two-pin or four-pin output according to the specific read mode instructions.

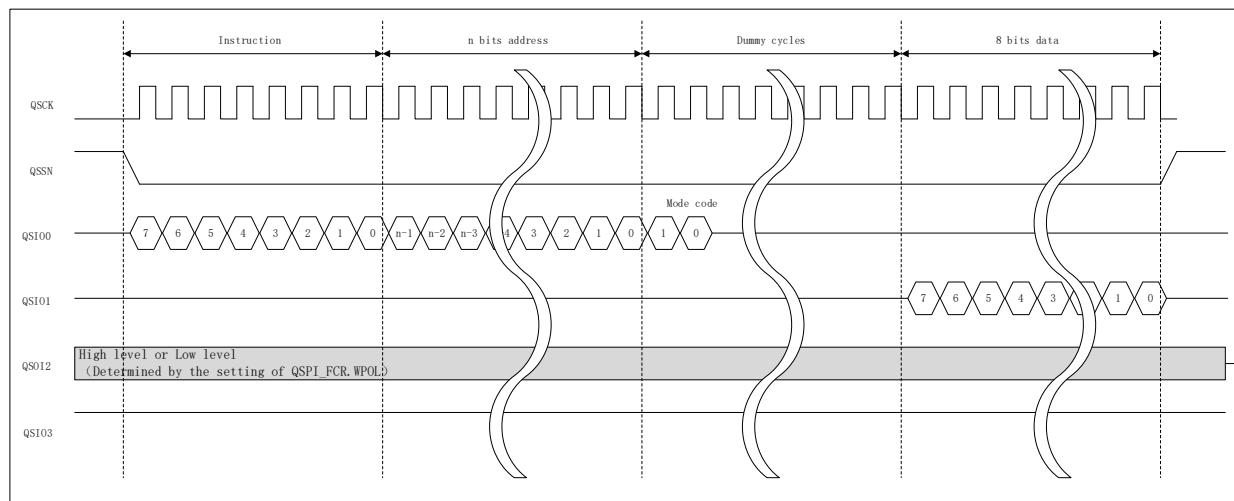


Figure 32-4 Extended SPI protocol operation diagram 1 (Fast Read mode)

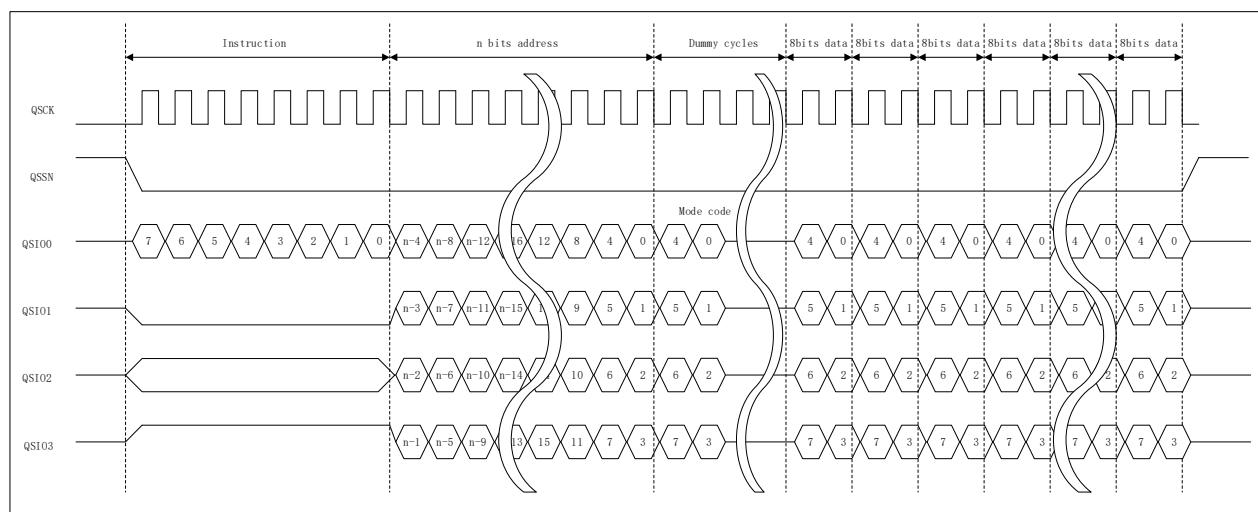


Figure 32-5 Extended SPI protocol operation diagram 2 (Fast Read Quad I/O mode)

The Dual SPI protocol uses two pins QSIO0 and QSIO1 to perform operations, such as sending instruction codes, addresses, and receiving data.

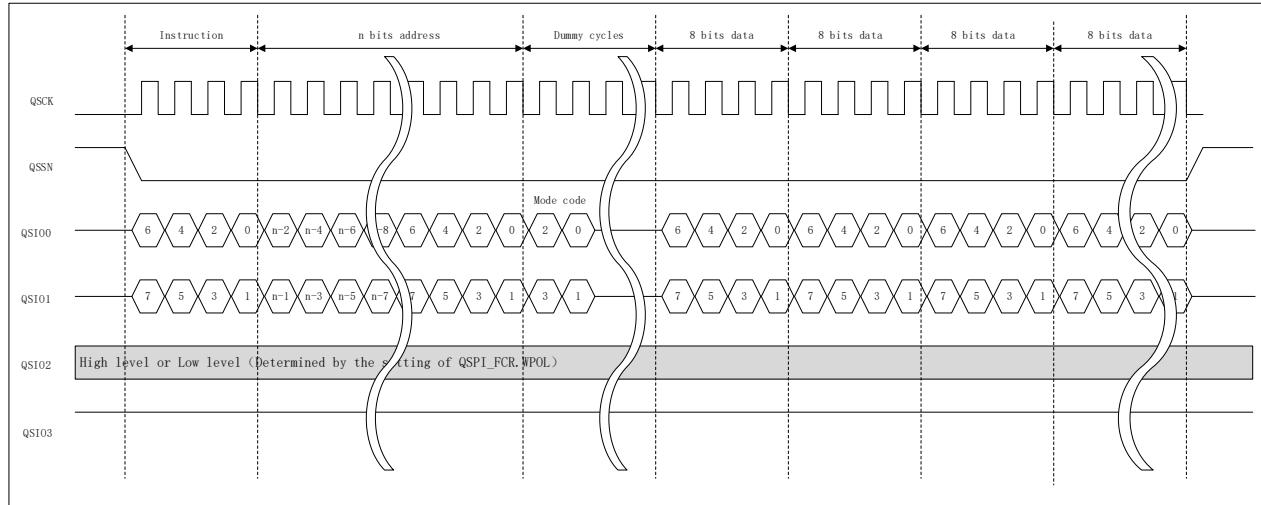


Figure 32-6 Dual SPI protocol operation diagram (Fast Read mode)

The Quad SPI protocol uses four pins QSIO0, QSIO1, QSIO2, and QSIO3 to perform operations, such as sending instruction codes, addresses, and receiving data.

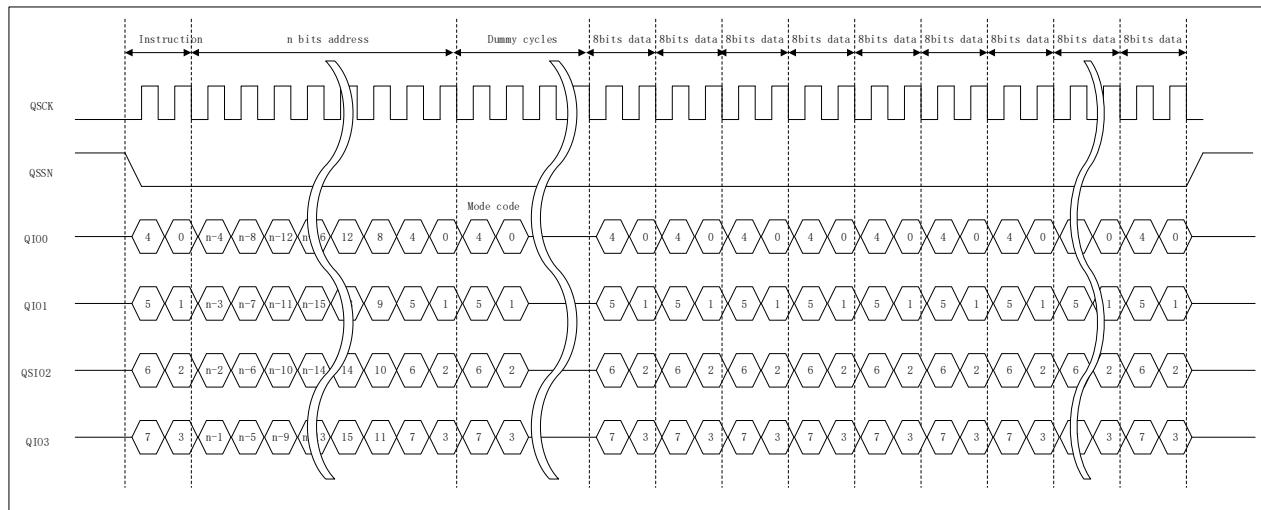


Figure 32-7 Quad SPI protocol operation diagram (Fast Read mode)

32.4.2 SPI Mode

There are two SPI modes: mode 0 and mode 3, which can be switched by setting the SPIMD3 bit in the QSPI_CR register. The difference between SPI mode 0 and mode 3 is the level of QSCK in the standby state. The standby level of QSCK is low in SPI mode 0, and high in SPI mode 3.

Serial data is output from the QSPI on the falling edge of the serial clock and is 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 into the QSPI on the falling edge of the next clock.

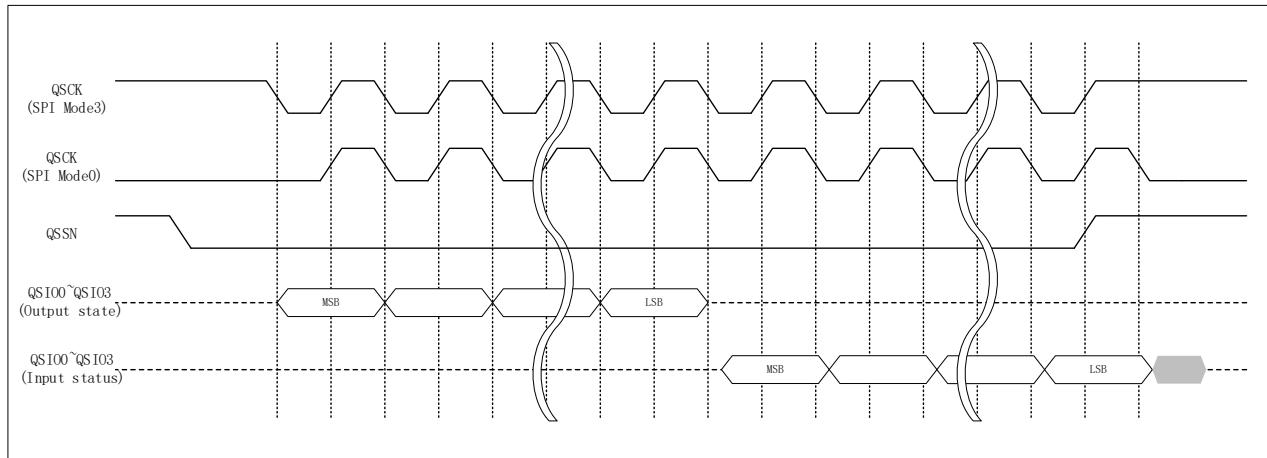


Figure 32-8 Basic timing of serial interface

32.5QSPI Bus Timing Adjustment

The timing of QSPI bus signals can be adjusted in the registers, and the adjusted timing can be better applied to various QSPI bus accesses, for both ROM access and direct communication.

32.5.1 QSPI Bus Reference Clock

The reference clock of the QSPI bus is obtained by dividing the frequency of HCLK. By setting the DIV[5:0] bits of the QSPI_CR register, various clock sources such as HCLK divided by 2 to 64 can be selected as the reference clock of the QSPI bus.

Table 32-3 QSPI bus reference clock selection list

DIV[5:0]	Frequency division ratio	Actual operating frequency	DIV[5:0]	Frequency division ratio	Actual operating 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]	Frequency division ratio	Actual operating frequency	DIV[5:0]	Frequency division ratio	Actual operating 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

32.5.2 SPI Bus Reference Clock

When the even-numbered frequency division of HCLK is selected as reference clock, the high- and low-level widths of the QSCK signal are consistent. When the odd-numbered multiple frequency division is selected, the high-level width of the QSCK signal will be one HCLK cycle longer than the low level.

If you want to select an odd frequency division and also output QSCK signal with a duty cycle of about 50%, you can set the DUTY bit in the QSPI_FCR register to 1. With this setting, the rising edge of the QSCK signal will be half a HCLK cycle later than the adjustment, and the falling edge will remain unchanged. So, a QSCK signal with 50% duty cycle can be obtained. When the reference clock selects an even-numbered frequency division of HCLK, please set the DUTY bit to 0. The DUTY bit defaults to 1 in the initial state.

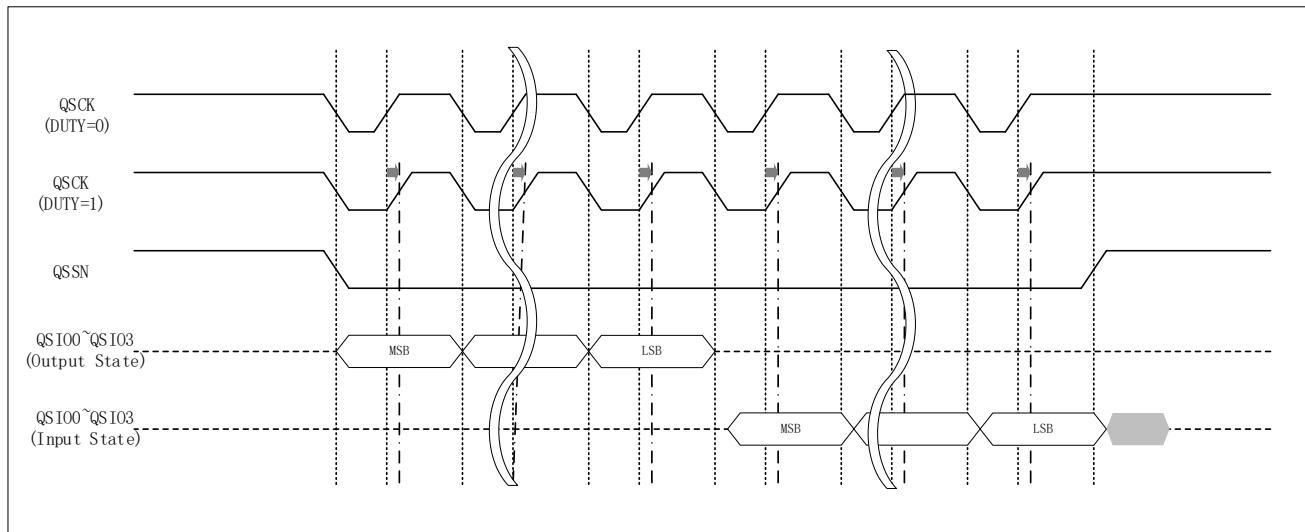


Figure 32-9 Correction diagram of output clock duty cycle when the reference clock selects HCLK divided by three

32.5.3 QSSN Signal Minimum High-Level Width

In order to meet the cancellation time required by serial flash, the QSSN signal must be held high (i.e., idle state) for a sufficient time between adjacent QSPI bus cycles. The minimum high-level width of QSSN can be selected by setting the SSHW[3:0] bits in the QSPI_CSCR register, and the selection range is from 1 to 16 QSPI reference clock cycles.

32.5.4 QSSN Signal Setup Time

The time from the QSSN signal starts to output low level (i.e., becomes active state) to the first rising edge of the QSCK signal is called the QSSN setup time. This time can be configured by register settings to satisfy the external serial flash memory requirements. Set the SSNLD bit in the QSPI_FCR register to select whether the QSSN setup time is 0.5 or 1.5 QSPI reference clock cycles. This setting can also be used to configure the data setup time from the data pins output permission to the first rising edge of the QSCK signal, which can be used reasonably as needed.

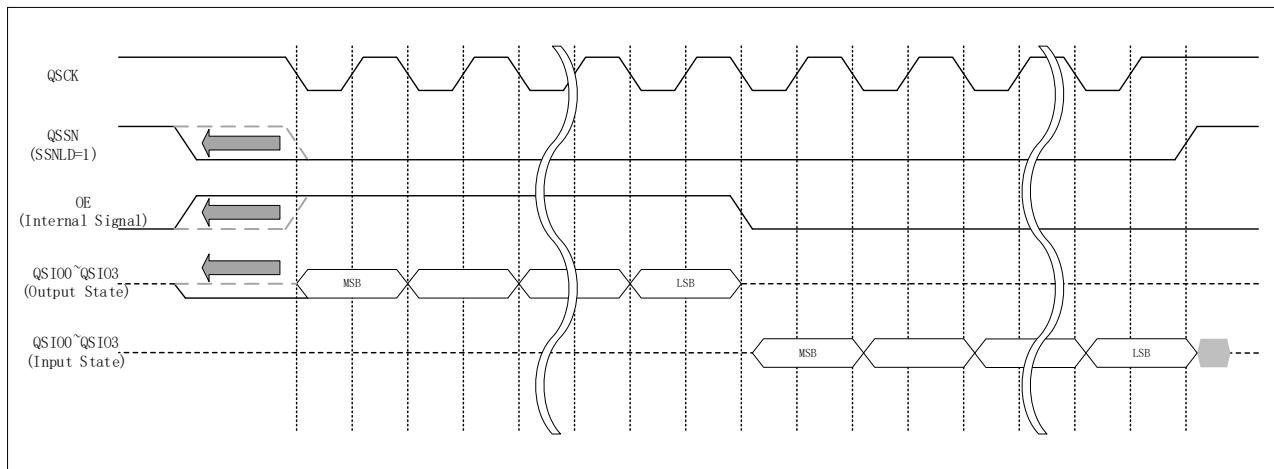


Figure 32-10 QSSN setup time configuration diagram

32.5.5 QSSN Signal Hold Time

The time from the last rising edge of the QSCK signal to the QSSN signal starts to output high level (i.e., becomes idle state) is called the QSSN hold time. This time can be configured by register settings to satisfy external device requirements. Set the SSNHD bit in the QSFCR register to select whether the hold time of the QSSN is 0.5 or 1.5 QSPI reference clock cycles.

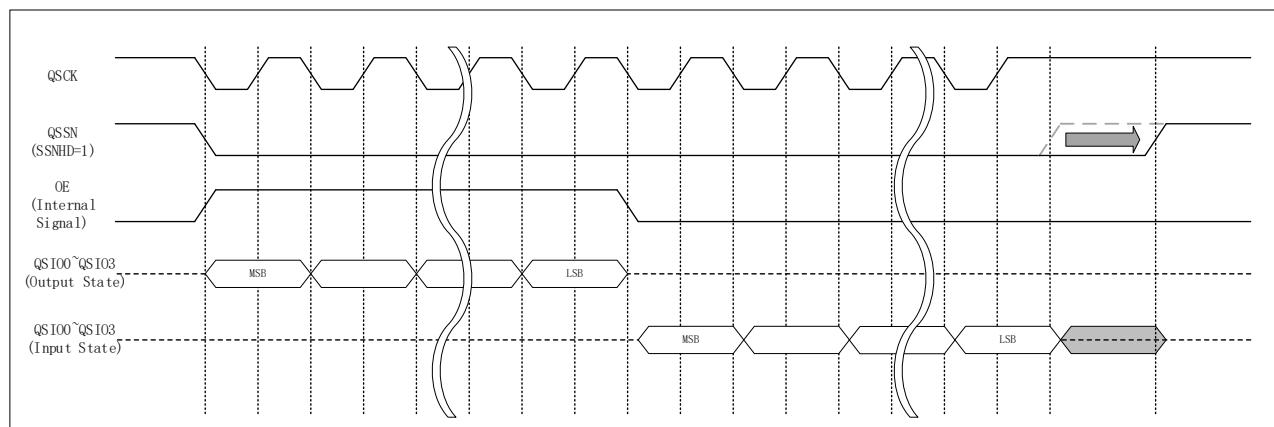


Figure 32-11 QSSN hold time configuration diagram

32.5.6 Serial Data Reception Delay

The data from the serial flash is output at the falling edge of QSCK, and QSPI receives the data at the next falling edge of QSCK. The time from the serial flash starts to output data to the data is received by QSPI is called the reception delay. QSPI adds a delay adjustment cycle before the first data receiving cycle. From the perspective of serial flash memory, this cycle can be regarded as an increase of the counterparty data receiving cycle. The delay adjustment period will only be generated during data reception.

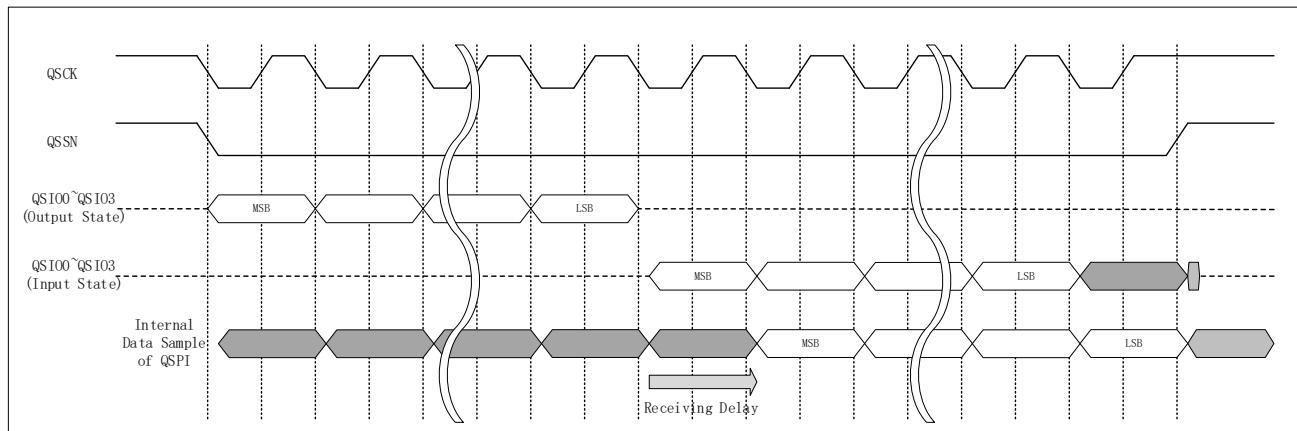


Figure 32-12 Data reception delay time diagram

32.6 Introduction to SPI Instructions for ROM Access

32.6.1 Existing QSPI-ROM Instruction Reference

Table 32-4 Reference instruction list

Schema name		Instruction code	Note
4-Byte Instruction Mode	Standard Read	8'h13	—
	Fast Read	8'h0c	—
	Fast Read Dual Output	8'h3c	—
	Fast Read Dual I/O	8'hbc	—
	Fast Read Quad Output	8'h6c	—
	Fast Read Quad I/O	8'hec	—
	Exit 4-Byte Mode	8'hb7	—
3-Byte Instruction Mode	Standard Read	8'h03	—
	Standard Read	8'h0b	When 8-bit address is selected and A8=1
	Fast Read	8'h0b	—
	Fast Read Dual Output	8'h3b	—
	Fast Read Dual I/O	8'hbb	—
	Fast Read Quad Output	8'h6b	—
	Fast Read Quad I/O	8'heb	—
	Enter 4-Byte Mode	8'he9	—
—	Write Mode	8'h06	—

When accessing the serial flash memory, the instruction needs to be set through the instruction register QSPI_CCMD.

32.6.2 Standard Read Instruction

The Standard Read instruction is a common read instruction supported by most serial flash memories. When the serial bus cycle starts, the serial flash memory select signal is asserted, and then the QSPI outputs the instruction code (03h/13h) and the target address. The width of the address can be specified by the AWSL[1:0] bits in the QSPI_FCR register. Then the data can be received. The instruction selected in the initial state of QSPI is the Standard Read instruction.

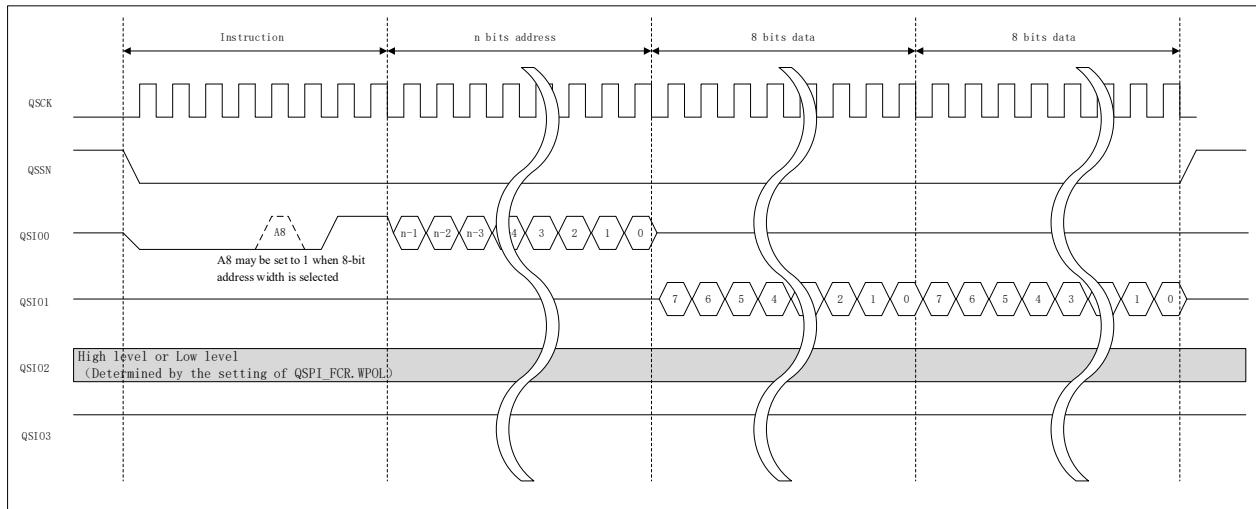


Figure 32-13 Standard Read bus cycle diagram

32.6.3 Fast Read Instruction

The Fast Read instruction is a read instruction that supports a faster communication clock. When the serial bus cycle starts, the serial flash select signal is asserted, and then the QSPI outputs the instruction code (0Bh/0Ch) and the target address. The address width can be specified by AWSL[1:0] in the QSPI_FCR register. After the address, there are a certain number of dummy cycles, the specific number can be determined by DMCYCN[3:0] in the QSPI_FCR register. Next is the data reception.

The first two dummy cycles are used to decide whether to select XIP mode. When XIP mode is selected, the instruction used in this transmission will be applied to the next SPI bus cycle, and the instruction transmission part will be omitted in the next SPI bus cycle. For details, please refer to 32.8 XIP Control.

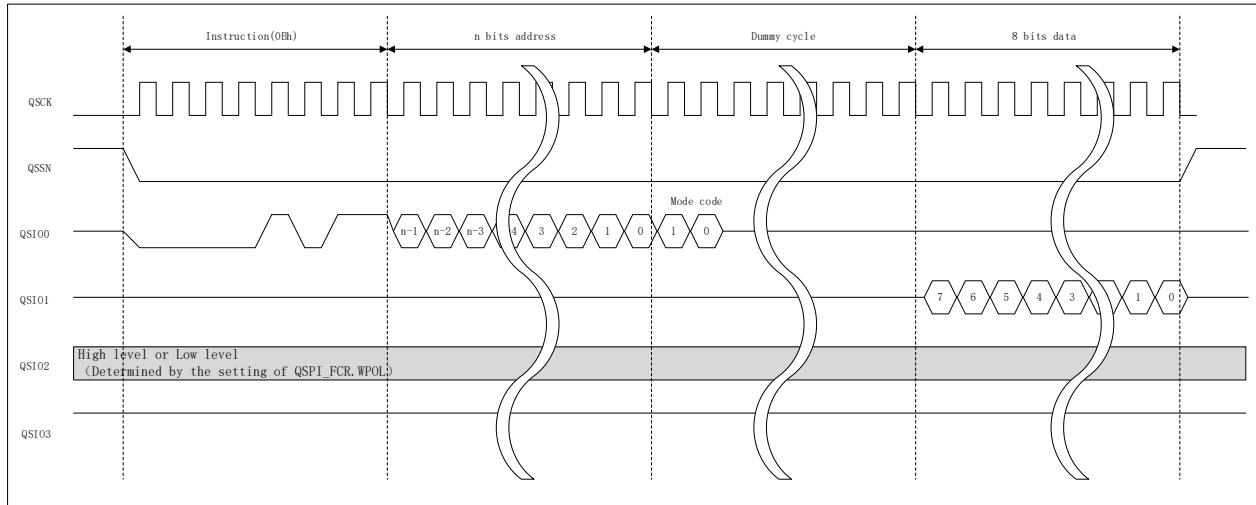


Figure 32-14 Schematic diagram of Fast Feed bus cycle

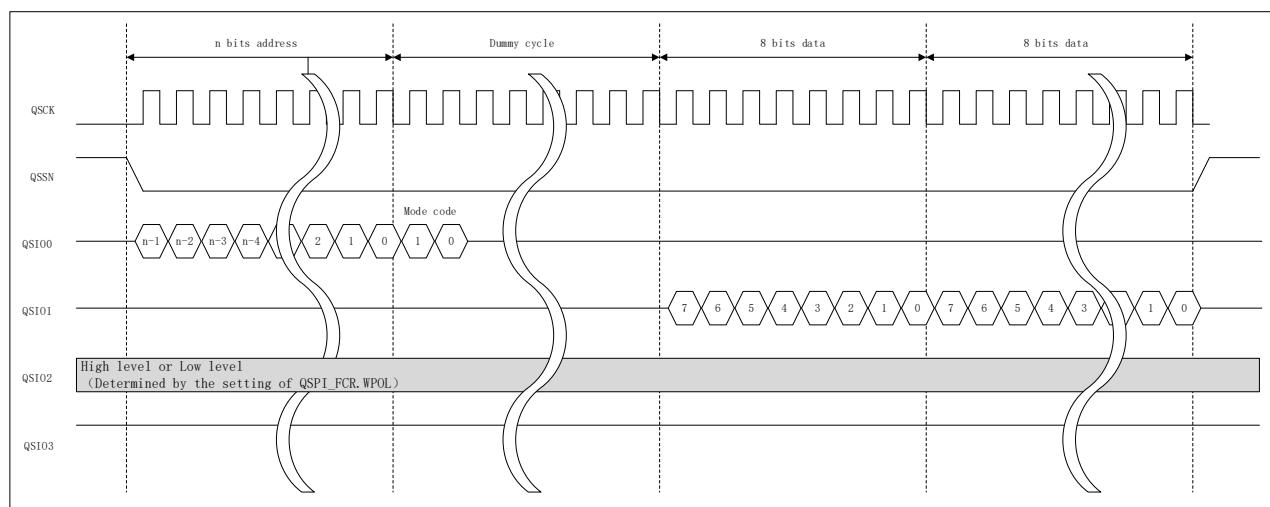


Figure 32-15 Schematic diagram of Fast Read bus cycle with XIP mode selected

Note:

- To use the Fast Read instruction, make sure to use a serial flash that supports Fast Read.

32.6.4 Fast Read Dual Output Instruction

The Fast Read Dual Output instruction is a read instruction that uses two signal lines for data reception. When the serial bus cycle starts, the serial flash memory select signal is asserted, and then QSPI outputs the instruction code (3Bh/3Ch) and target address to the QSIO0 pin. The address width can be specified by AWSL[1:0] in the QSPI_FCR register. After the address, there are a certain number of dummy cycles, the specific number can be determined by DMCYCN[3:0] in the QSPI_FCR register. Then data is received from the QSIO0 and QSIO1 pins. Even bit data is received from QSIO0, and odd bit data is received from QSIO1.

The first two dummy cycles are used to decide whether to select XIP mode. When XIP mode is selected, the instruction used in this transmission will be applied to the next QSPI bus cycle, and the instruction transmission part will be omitted in the next QSPI bus cycle. For details, please refer to 32.8 XIP Control.

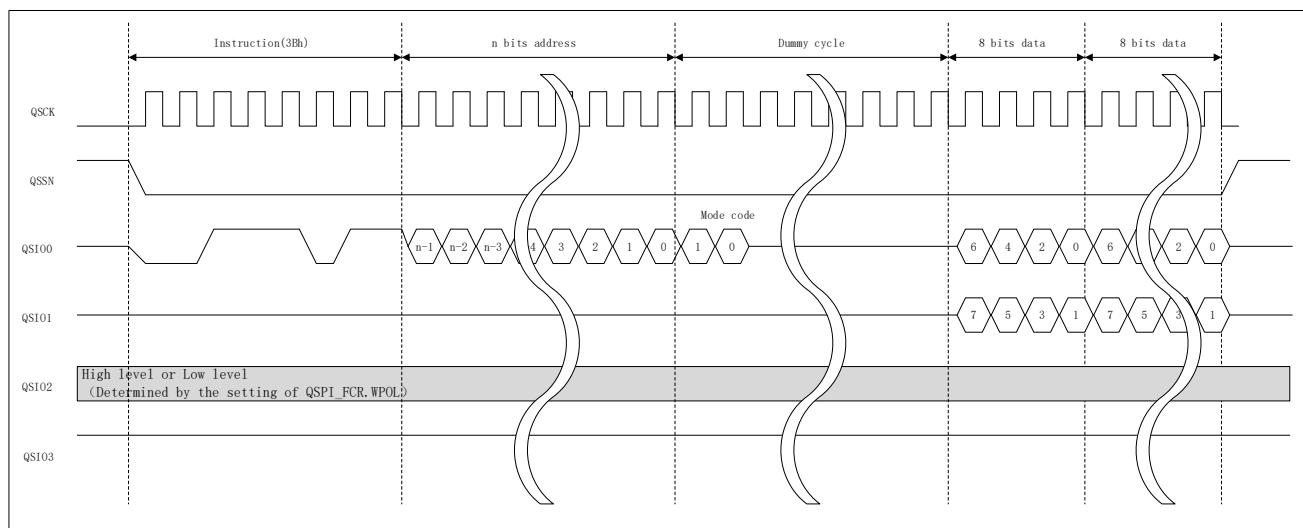


Figure 32-16 Schematic diagram of Fast Read Dual Output bus cycle

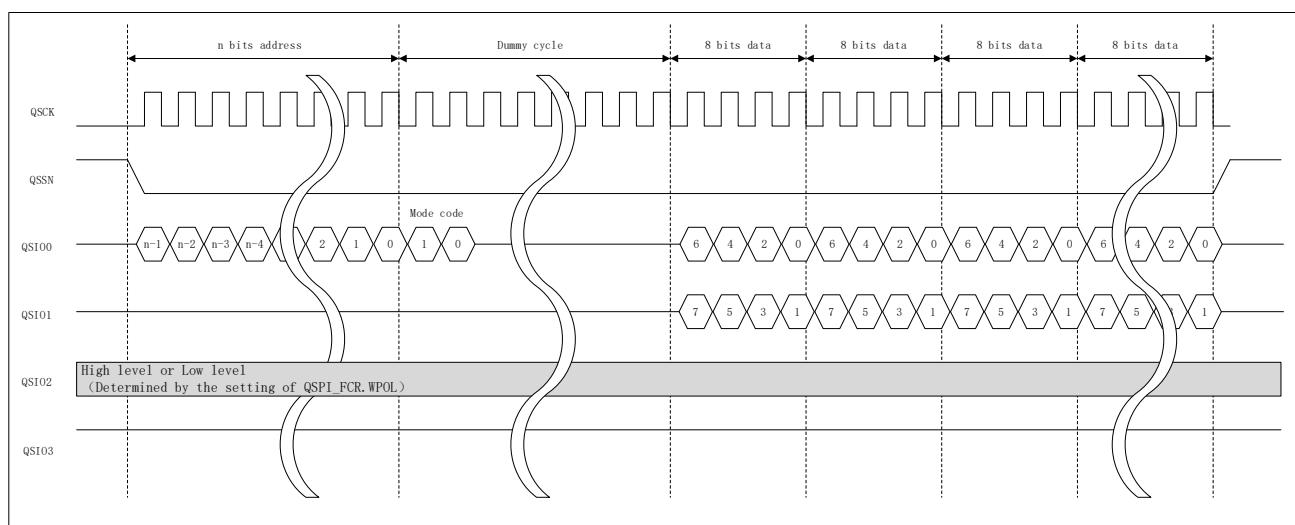


Figure 32-17 Schematic diagram of Fast Read Dual Output bus cycle with XIP mode selected

Note:

- To use the Fast Read Dual Output instruction, make sure to use a serial flash that supports this feature.

32.6.5 Fast Read Dual I/O Instruction

The Fast Read Dual I/O instruction is a read instruction that uses two signal lines for address transmission and data reception. When the serial bus cycle starts, the serial flash select signal is asserted, and the QSPI outputs the instruction code (BBh/BCh) to the QSIO0 pin and outputs the target address to the QSIO0 and QSIO1 pins. The address width can be specified by AWSL[1:0] in the QSPI_FCR register. After the address, there are a certain number of dummy cycles, the specific number can be determined by DMCYCN[3:0] in the QSPI_FCR register. Then data is received from the QSIO0 and QSIO1 pins. Address and dummy cycle transmission and data reception are performed through the QIO0 pin for even bits and through the QIO1 pin for odd bits. The first two dummy cycles are used to decide whether to select XIP mode. When XIP mode is selected, the instruction used in this transmission will be applied to the next QSPI bus cycle, and the instruction transmission part will be omitted in the next QSPI bus cycle. For details, please refer to [32.8 XIP Control].

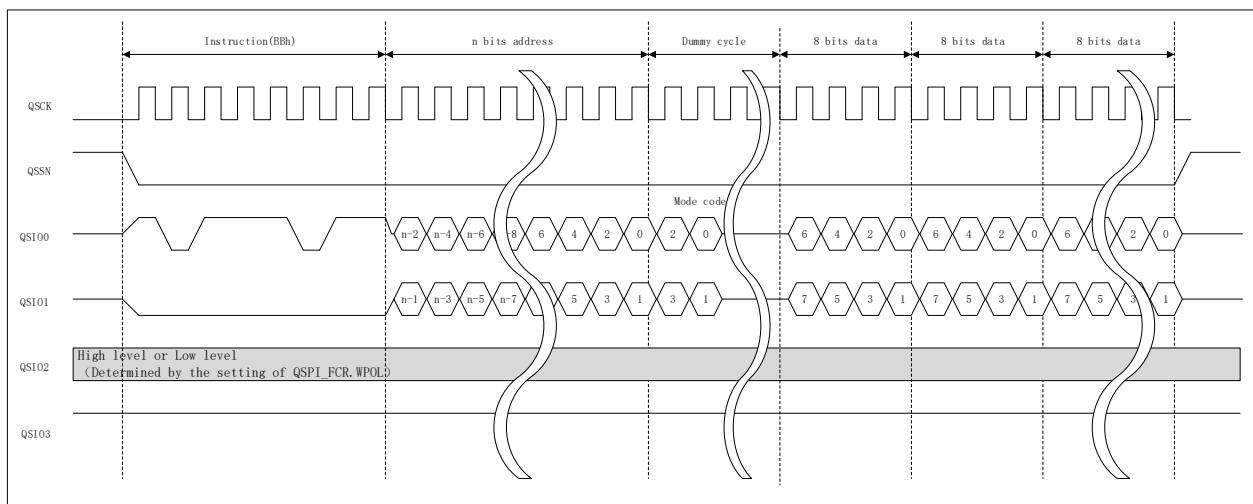


Figure 32-18 Schematic diagram of Fast Read Dual I/O bus cycle

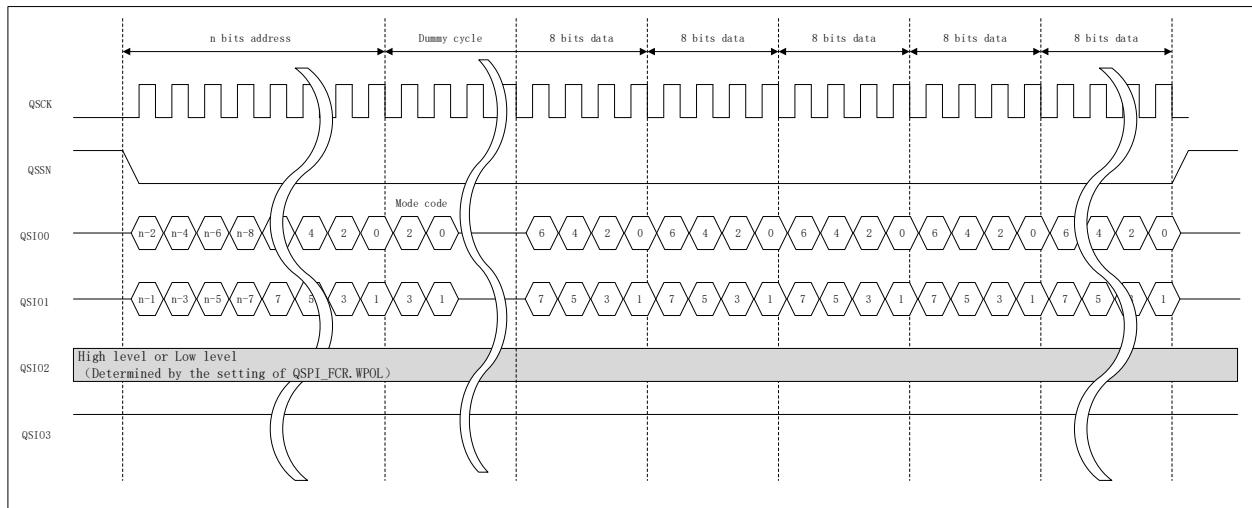


Figure 32-19 Schematic diagram of Fast Read Dual I/O with XIP mode selected

Note:

- To use the Fast Read Dual I/O instruction, make sure to use a serial flash that supports this feature.

32.6.6 Fast Read Quad Output Instruction

The Fast Read Quad Output instruction is a read instruction that uses four signal lines for data reception. When the serial bus cycle starts, the serial flash memory select signal is asserted, and the QSPI starts to output the instruction code (6Bh/6Ch) and the target address to the QSIO0 pin. The address width can be specified by AWSL[1:0] in the QSPI_FCR register. After the address there are a certain number of dummy cycles, the specific number can be determined by DMCYCN[3:0] in the QSPI_FCR register. Then data is received from the QSIO0, QSIO1, QSIO2 and QSIO3 pins.

The first two dummy cycles are used to decide whether to select XIP mode. When XIP mode is selected, the instruction used in this transmission will be applied to the next SPI bus cycle, and the instruction transmission part will be omitted in the next SPI bus cycle. For details, please refer to [32.8 XIP Control].

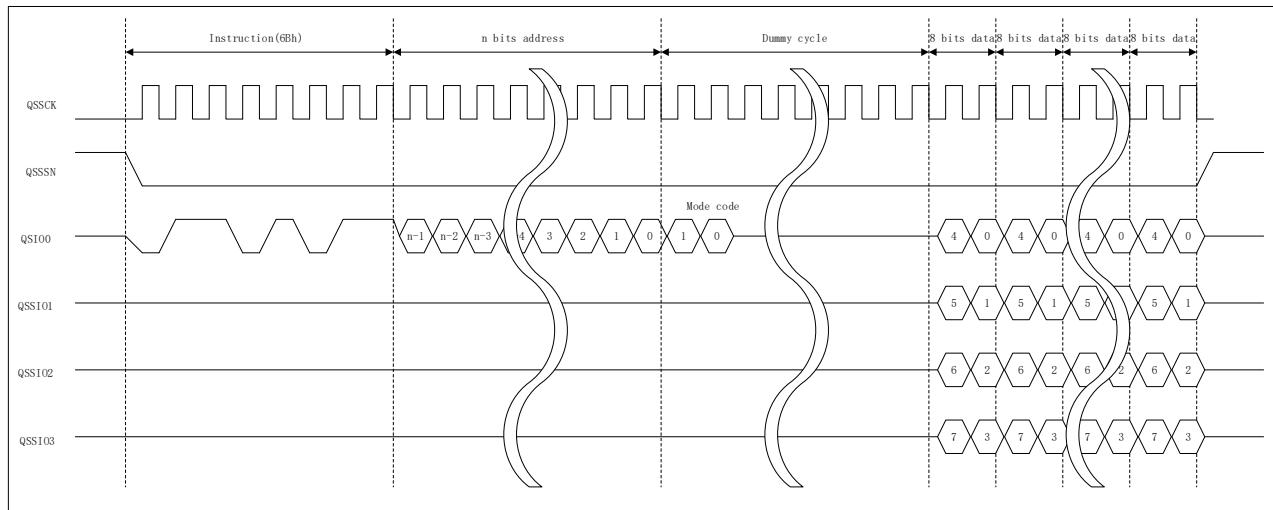


Figure 32-20 Schematic diagram of Fast Read Quad Output bus cycle

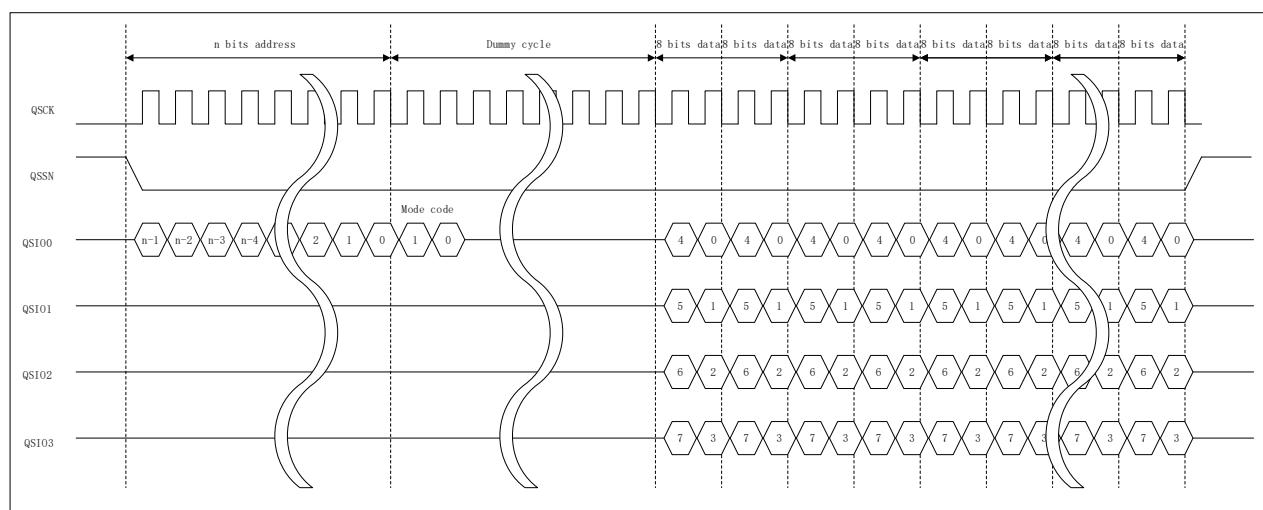


Figure 32-21 Schematic diagram of Fast Read Quad Output with XIP mode selected

Note:

- To use the Fast Read Quad Output instruction, make sure to use a serial flash that supports this feature.

32.6.7 Fast Read Quad I/O Instruction

The Fast Read Quad I/O instruction is a read instruction that uses four signal lines for address transmission and data reception. When the serial bus cycle starts, the serial flash select signal is asserted, and QSPI outputs the instruction code (EBh/ECh) to the QSIO0 pin and outputs the target address to the QSIO0, QSIO1, QSIO2 and QSIO3 pins. The address width can be specified by AWSL[1:0] in the QSPI_FCR register. After the address, there are a certain number of dummy cycles, the specific number can be determined by DMCYCN[3:0] in the QSPI_FCR register. Then data is received from the QSIO0, QSIO1, QSIO2 and QSIO3 pins.

The first two dummy cycles are used to decide whether to select XIP mode. When XIP mode is selected, the instruction used in this transmission will be applied to the next QSPI bus cycle, and the instruction transmission part will be omitted in the next QSPI bus cycle. For details, please refer to [32.8 XIP Control].

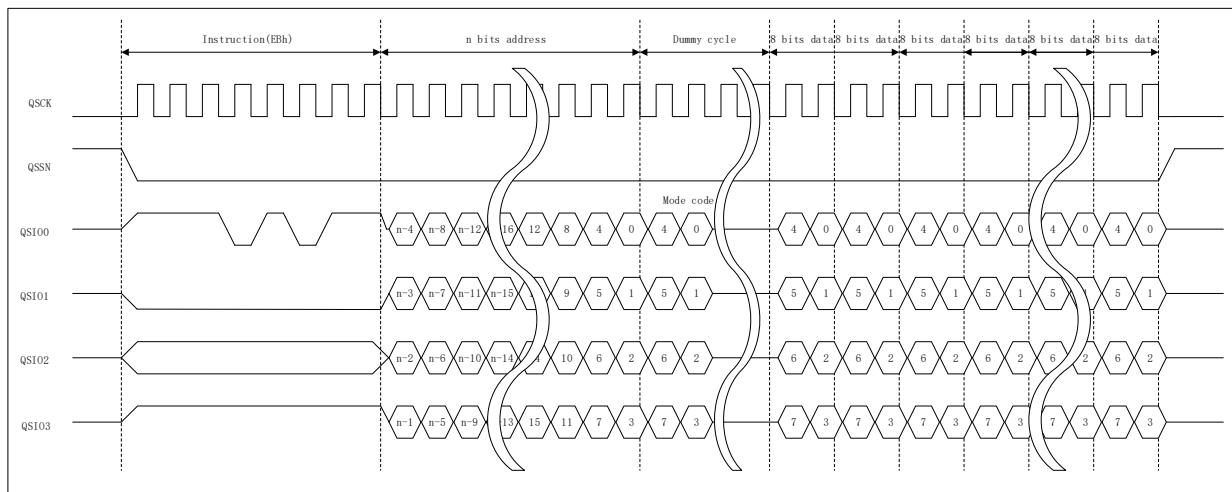


Figure 32-22 Fast Read Quad I/O bus cycle diagram

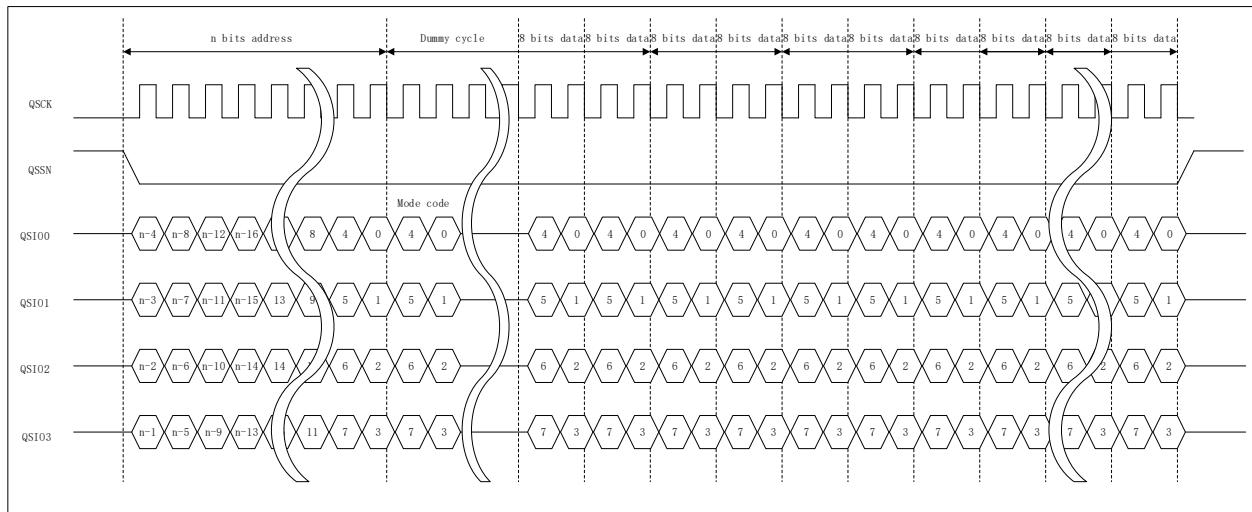


Figure 32-23 Schematic diagram of Fast Read Quad I/O with XIP mode selected

Note:

- To use Fast Read Quad I/O instruction, make sure to use a serial flash that supports this feature.

32.6.8 Enter 4-Byte Mode Instruction

The Enter 4-Byte Mode instruction sets the serial flash address width to 4 bytes. When the serial bus cycle starts, the serial flash select signal is asserted, and QSPI starts to output the instruction code (B7h) to the QSIO0 pin.

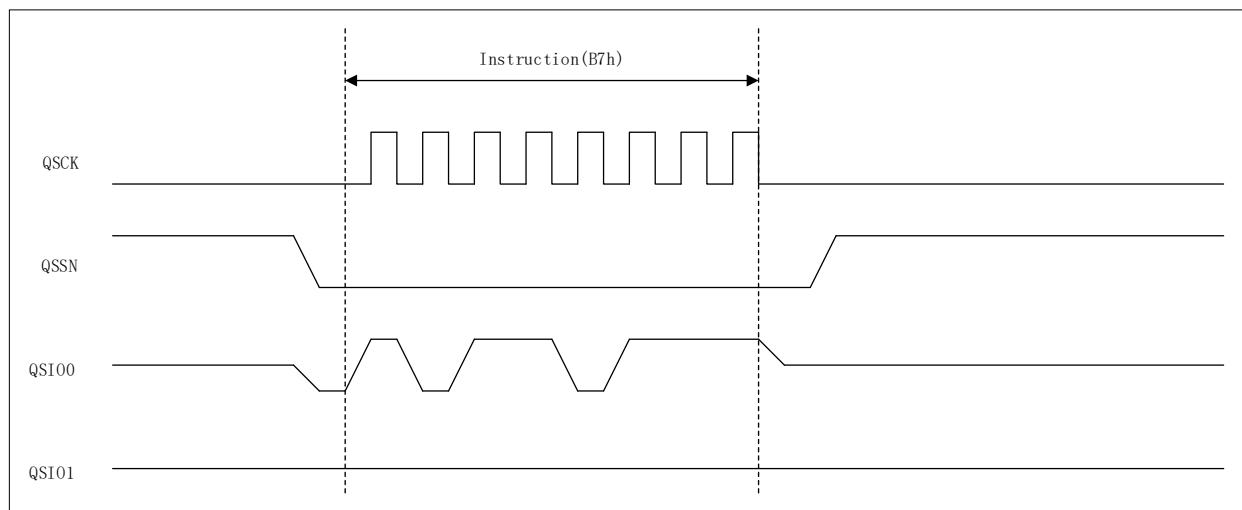


Figure 32-24 Enter 4-Byte Mode instruction bus cycle diagram

Note:

- This instruction can be issued regardless of whether the serial flash is in 4-Byte or 3-Byte mode.

32.6.9 Exit 4-Byte Mode Instruction

The Exit 4-Byte Mode instruction sets the serial flash address width to 3 bytes. When the serial bus cycle starts, the serial flash select signal is asserted, and QSPI starts to output the instruction code (E9h) to the QSIO0 pin.

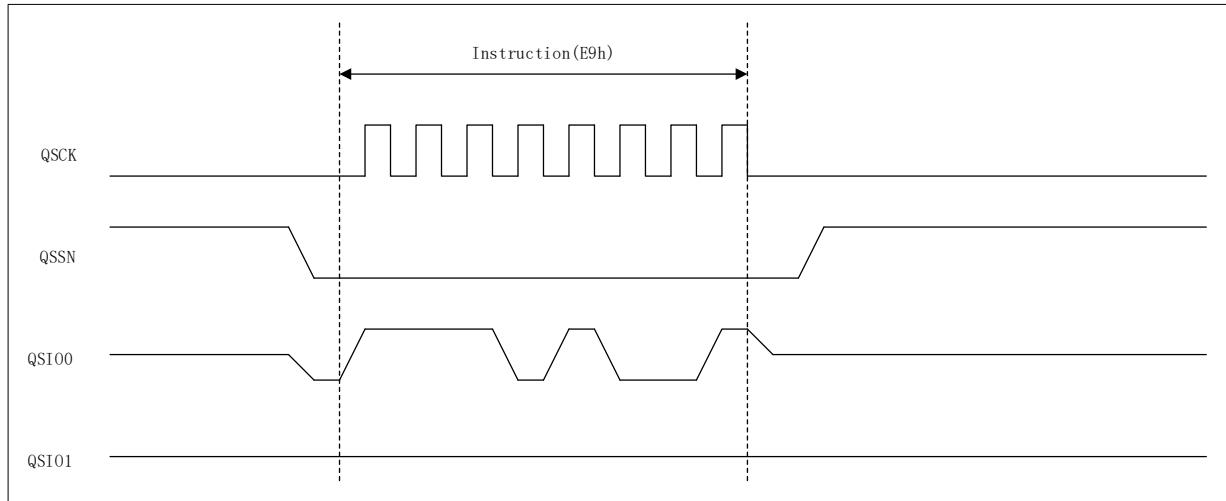


Figure 32-25 Exit 4-Byte Mode instruction bus cycle diagram

Note:

- This instruction can be issued regardless of whether the serial flash is in 4-Byte or 3-Byte mode.

32.6.10 Write Permission Instruction

The Write Permission instruction allows changing the address width of the serial flash. When a serial bus cycle starts, the serial flash select signal is asserted, and QSPI starts to output the instruction code (06h) to the QSIO0 pin.

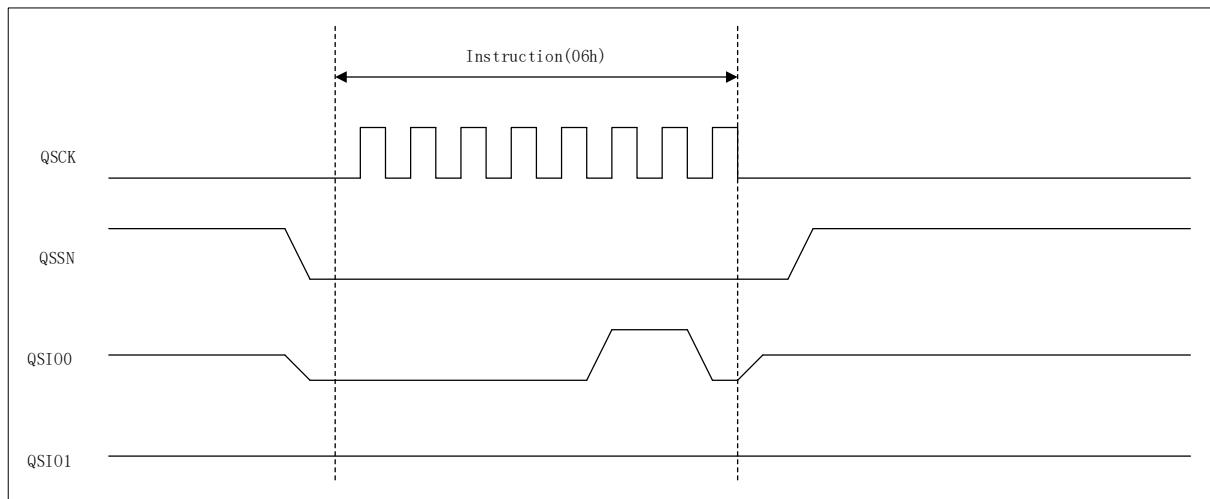


Figure 32-26 Schematic diagram of Write Permission instruction bus cycle

32.7 QSPI Bus Cycle Arrangement

32.7.1 Single Flash Read with Independent Conversion

The single ROM read instruction will be independently converted from the chip internal bus cycle to the QSPI bus cycle one-to-one. When a ROM read bus cycle is detected, the QSSN signal is asserted to initiate a QSPI bus cycle. After receiving the data from the serial flash, the QSSN signal becomes deasserted, and the QSPI bus cycle is declared complete.

When another ROM read bus cycle is detected, the QSSN signal will be asserted again after ensuring that the invalid hold time has exceeded the minimum invalid hold width, and a new QSPI bus cycle starts.

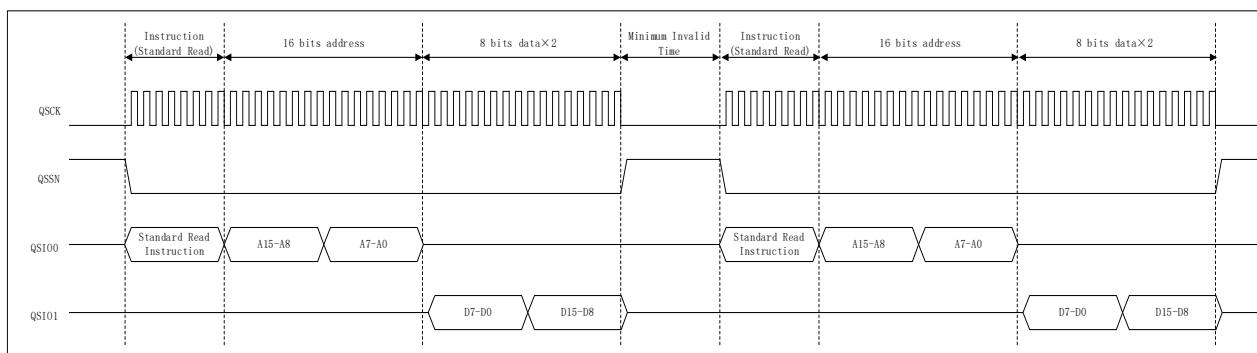


Figure 32-27 Schematic diagram of a single flash data read operation with independent conversion

32.7.2 Flash Read Using Prefetch Function

For transmissions such as CPU instructions and data blocks, the system usually reads data sequentially in ascending order from flash memory address. Serial flash has continuous data transfer capability without re-sending instruction codes and addresses. However, if the internal bus cycle issued by the MCU is an independent conversion, the QSPI bus cycle is also divided into independent individuals, resulting in the inability to effectively utilize the advantages of serial flash memory for continuous data transmission. In this regard, QSPI provides a prefetch function for continuous data reception.

The prefetch function is activated by setting the PFE bit in the QSPI_CR register to 1. When this function is enabled, data is continuously received and stored in the buffer without waiting for another flash read request. When the MCU issues a flash read operation, QSPI will match the access address. If the match is successful, the data in the buffer at the corresponding position will be transferred to the MCU. If the match fails, the data in the buffer will be discarded and a new QSPI bus cycle will be reissued.

The prefetch buffer can store up to 16 bytes of data. In addition, there is a 2-byte data receiving buffer that can also store prefetch data. When all the buffer is full, the QSPI bus cycle is ended. When the buffer space is released after the buffer data is read, QSPI will automatically start a new QSPI bus cycle to resume the prefetch operation.

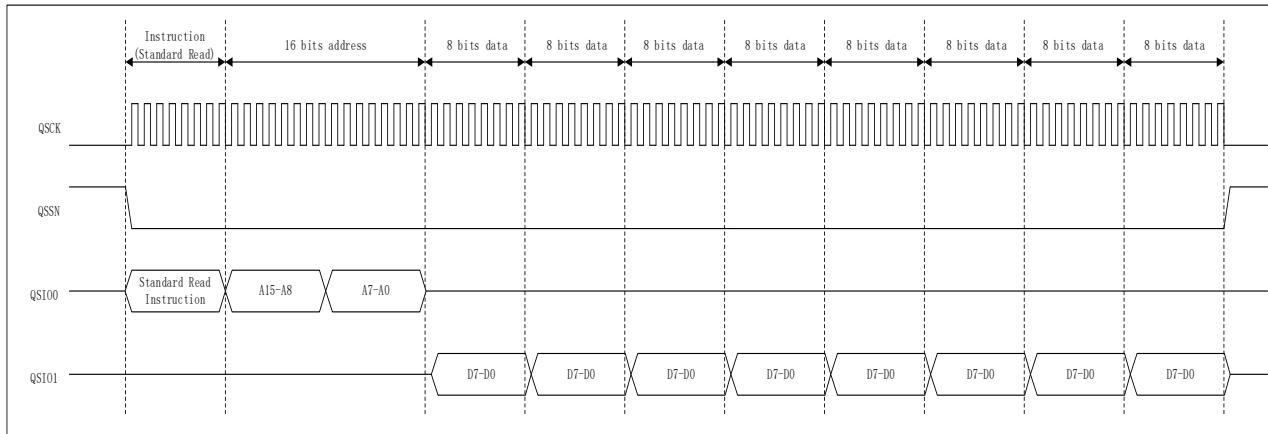


Figure 32-28 Schematic diagram of data read operation using prefetch function

32.7.3 Prefetch Termination

If a ROM read bus cycle to another address of the serial flash occurs during the prefetch transfer, the original prefetch operation will be terminated and a new QSPI bus cycle will start. Normally, the prefetch operation will be terminated after the current byte transfer is completed, but if the PFSAE bit in the QSPI_CR register is set to 1, QSPI will immediately stop the prefetch operation without waiting for the current byte transfer to complete. To use this function, the serial flash device must support the instant stop action feature.

32.7.4 Prefetch Status Monitoring

Reading data from a low-speed serial flash memory will increase the system load because the internal bus needs to be in a wait state until the received QSPI bus cycle is complete. QSPI provides a prefetch status monitoring to reduce this load.

In the prefetch status register QSPI_SR, the PFAN bit shows the current prefetch working state, the PFFUL bit indicates that the prefetch data buffer is full, and PFNUM[4:0] shows that the byte number of data that have been read in the buffer. Through these status bits, it is very convenient to determine the current prefetch status through a CPU instruction.

Note:

- When executing a prefetch status monitor program, place the program code outside the object serial flash area or enable the instruction cache. Otherwise, the prefetch object will frequently switch between the object data area and the instruction area, losing the meaning of prefetch, and the monitoring program will also enter an infinite loop because the prefetch can never be completed.

32.7.5 Flash Read Using QSPI Bus Cycle Extension Function

If SSNW[1:0] in the QSPI_CSCR register is set to a value other than 00, the QSPI bus cycle will be held in the hold state after receiving the data, waiting for the next data read. At this time, the QSSN signal will remain active low level, and the QSCK is stopped. When the next flash memory read instruction arrives, if the read object address is sequentially incremented next to the current address, QSPI will restart QSCK and directly accept data. If it is a non-consecutive address, the QSSN signal will be set to an inactive state of high level to end the previously maintained QSPI bus cycle, and then restart a new SPI bus cycle.

This function can prevent the system from repeatedly sending instruction codes and addresses when performing discontinuous reads of continuously increasing addresses, thereby improving read efficiency.

The extension time of the QSPI bus cycle can be set through SSNW[1:0]. When the next read does not occur after the set time, QSSN will automatically be set to a high level to end the QSPI bus cycle. If SSNW[1:0] is set to 11, the QSSN will be extended indefinitely, and the QSPI bus cycle will always be in the hold state, but it will increase the power consumption of the serial flash.

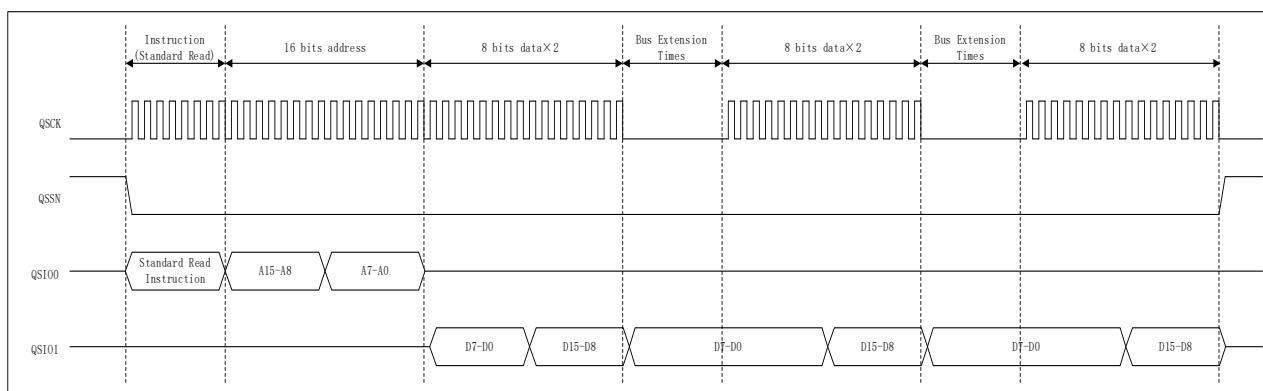


Figure 32-29 Schematic diagram of data read operation using the QSPI bus cycle extension function

32.8 XIP Control

Some serial flash devices can reduce latency by omitting receiving read instructions. This function can be selected by the mode code sent during the dummy cycle.

During the dummy cycle of Fast Read instructions, QSPI controls the XIP mode of the serial flash by sending the XIP mode code in the first two cycles. Different serial flash memory has different XIP mode codes, which can be set pertinently through the XIPMC[7:0] bits of the register QSPI_XCMD. Please refer to the following Figure 32-30 .

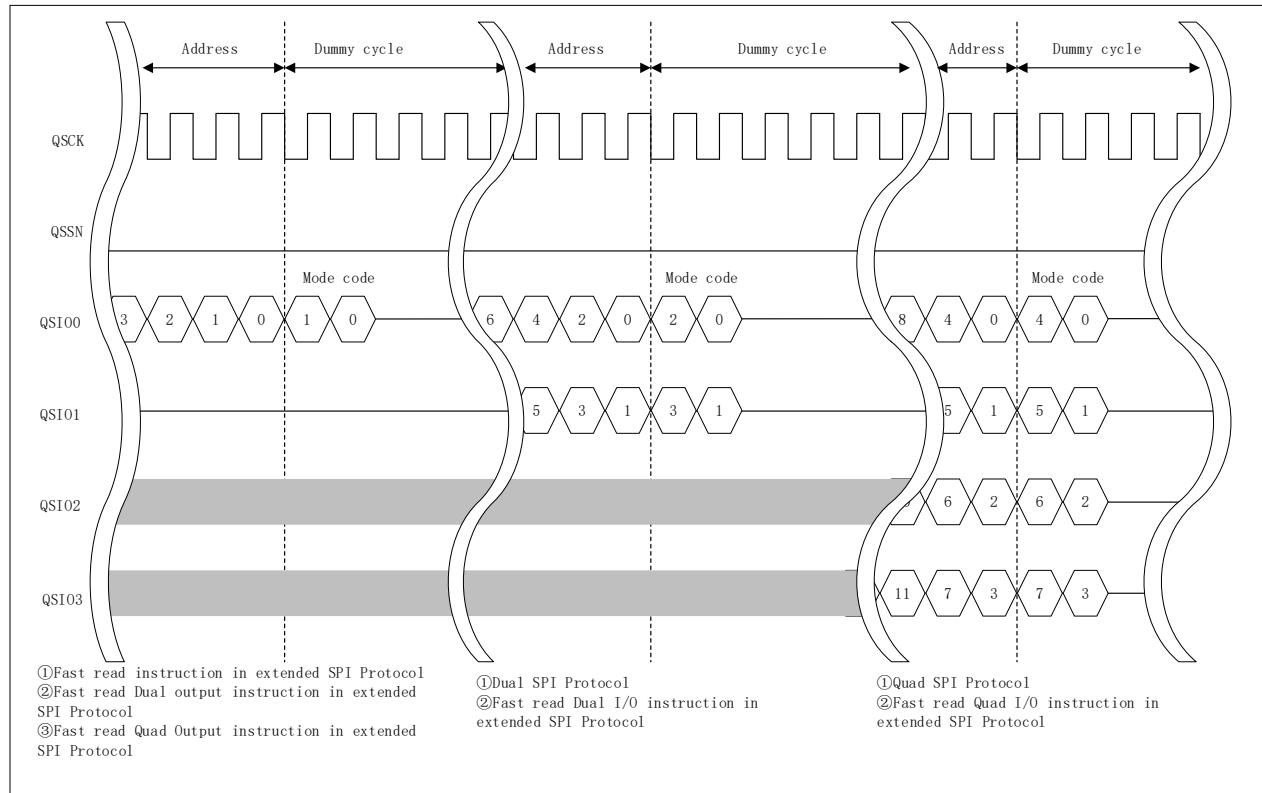


Figure 32-30 XIP mode control diagram

32.8.1 XIP Mode Settings

IF the XIP mode code corresponding to the serial flash is written into the XIPMC[7:0] bits of the QSPI_XCMD register and the XIPE bit of the QSPI_CR register is set to 1, when the next Fast Read instruction occurs, the set mode code will be transmitted to target serial flash memory during the first two dummy cycles. After the mode code is received, the serial flash memory and its control section start the XIP mode. It can be confirmed whether XIP mode has been entered by accessing the XIPF bit of the QSPI_SR register.

Note:

- To start the XIP mode of the serial flash, you need to set the corresponding mode code in QSPI_XCMD[7:0]. The XIP mode of the control part only needs to set the XIPE bit to 1, regardless of the value of QSPI_XCMD[7:0].

32.8.2 Exit from XIP Mode

IF the code for exiting XIP mode corresponding to the serial flash memory is written into the XIPMC[7:0] bits of QSPI_XCMD register and the XIPE bit of QSPI_CR register is set to 0, when the next Fast Read instruction occurs, the set exit mode code will be transmitted to the target serial flash during the first two dummy cycles. After the exit mode code is received, the XIP mode of the serial flash and its control section is terminated. It can be confirmed whether XIP mode has been exited by accessing the XIPF bit of the QSPI_SR register.

Note:

- To exit the XIP mode of the serial flash, you need to set the corresponding exit mode code in QSPI_XCMD[7:0]. The XIP mode of the control part only needs to clear the XIPE bit, regardless of the value of QSPI_XCMD[7:0].

32.9QSIO2 and QSIO3 Pin States

The QSIO2 and QSIO3 pin states depend on the serial read mode specified by the MDSEL[2:0] bits in the QSPI_CR register.

Table 32-5 Pin states of QIO2 and QIO3

QSPI_CR register MDSEL[2:0] bits	QSIO2 status	QSIO3 status	Remark
000			Standard read (initial state)
001			Fast Read
010	Output state, the output level is determined by the WPOL bit of the QSPI_FCR register, the initial output is low level	Output high level	Fast Read Dual Output
011			Fast Read Dual I/O
100	Input or output as the third data line, the standby state is Hi-Z	Input or output as the fourth data line, the standby state is Hi-Z	Fast Read Quad Output
101			Fast Read Quad I/O
110	Refer to the specific protocol settings at each stage	Refer to the specific protocol settings at each stage	Custom Protocol Standard Read
111			Custom Protocol Fast Read

Note:

- The QSIO2 pin can also be used as the WP# function of the serial flash.
- The QSIO3 pin can also be used as a serial flash HOLD# or RESET# function.

32.10 Direct Communication Mode

32.10.1 About Direct Communication Mode

QSPI can read serial flash memory by automatically converting the MCU's external ROM read bus cycle to QSPI bus cycle. However, serial flash has many different additional functions, such as ID information read, erase, write and status information read. There is no standardized instruction set for using these functions, and with the rapid increase of new functions in serial flash memory, the corresponding hardware level becomes more and more difficult.

In response to this situation, QSPI provides the direct communication mode, and users can directly control the serial flash memory through software. This mode can generate any desired QSPI bus cycle.

32.10.2 Direct Communication Mode Settings

Set the DCOME bit of the QSPI_CR register to 1 to enter direct communication mode. Once the direct communication mode is entered, the normal flash memory read operation cannot be performed. If the normal flash memory read operation is to be performed, the DCOME bit needs to be cleared to exit the direct communication mode.

Note:

- If you are in XIP mode, you need to exit XIP mode and then start direct communication mode.

32.10.3 Generation of QSPI Bus Cycles in Direct Communication Mode

A complete QSPI bus cycle in direct communication mode starts from the first operation to DCOM[7:0] of the QSPI_DCOM register and ends after a write operation to the QSPI_CR register. During this period, multiple operations can be performed on DCOM[7:0]. The write to DCOM[7:0] will be converted into a single-byte data transmission to the QSPI bus, and the read from DCOM[7:0] will be converted into a single-byte data reception from the QSPI bus.

From the first operation to DCOM[7:0] of the register QSCOM to the last write operation to the QSPI_CR register, the QSSN signal is held a valid state of low level during this period.

Direct communication mode does not support multi-line operation.

Note:

- It is impossible to write registers other than QSPI_CR and QSPI_DCOM in direct communication mode. Writes to other registers will exit direct communication mode. Exiting in this way may cause unpredictable situations and is not recommended.

32.11 Interrupt

When a read access operation to ROM is detected in the direct communication mode, the RAER bit of the QSPI_SR register is set to 1, and the QSPI will generate a bus hardware error interrupt. The interrupt request will be held until the RAER bit is cleared. Please refer to [Interrupt controller (INTC)] for details.

32.12 Precautions for Use

32.12.1 Setting Sequence of QSPI Registers

The QSPI control register can be set or changed dynamically during system operation. However, not paying attention to the setting sequence of the registers may cause the QSPI bus cycle to start before the registers are completely set. Therefore, please configure the setting sequence of the registers carefully to avoid such situations.

32.12.2 Module Stop Signal Setting

QSPI is in the module stop state after system reset, and the register can be set only after clearing the QSPI module stop signal in the module stop control register. For details, please refer to [Operation Mode and Low Power Mode].

32.13 Register Description

Register base address: 0x9c000000h

Table 32-6 List of QSPI registers

Register name	Offset address	Reset value
QSPI Control Register (QSPI_CR)	0x0000h	0x003f0000
QSPI Chip Select Control Register (QSPI_CSCR)	0x0004h	0x0000000f
QSPI Format Control Register (QSPI_FCR)	0x0008h	0x000080b3
QSPI Status Register (QSPI_SR)	0x000Ch	0x00008000
QSPI Direct Communication Register (QSPI_DCOM)	0x0010h	0xFFFFFFFFXX
QSPI Command Code Register (QSPI_CCMD)	0x0014h	0x00000000
QSPI XIP Mode Code Register (QSPI_XCMD)	0x0018h	0x000000ff
QSPI Status Flag Clear Register (QSPI_SR2)	0x0024h	- (write only)
QSPI External Address Register (QSPI_EXAR)	0x0804h	0x00000000

32.13.1 QSPI Control Register (QSPI_CR)

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
—	—	DPRSL[1:0]		APRSL[1:0]		IPRSL[1:0]	
B7	b6	b5	b4	b3	b2	b1	b0
SPIMD3	XIPE	DCOME	PFSAE	PFE		MDSEL[2:0]	

Bit	Symbol	Bit name	Description	Read and write
b31~b22	Reserved	—	Read as "0", write as "0"	R/W
b21~b16	DIV[5:0]	Reference clock selection bits	Serial Interface Reference Clock Selection b5 b4 b3 b2 b1 b0 0 0 0 0 0: 2 HCLK cycles 0 0 0 0 1: 2 HCLK cycles* 0 0 0 1 0: 3 HCLK cycles 0 0 0 1 1: 4 HCLK cycles* 0 0 0 1 0 0: 5 HCLK cycles 0 0 0 1 0 1: 6 HCLK cycles* 0 0 0 1 1 0: 7 HCLK cycles 0 0 0 1 1 1: 8 HCLK cycles* 0 0 1 0 0 0: 9 HCLK cycles 0 0 1 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: 17 HCLK cycles 0 1 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: 25 HCLK cycles 0 1 1 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 0 0: 29 HCLK cycles 0 1 1 1 0 1: 30 HCLK cycles 0 1 1 1 1 0: 31 HCLK cycles 0 1 1 1 1 1: 32 HCLK cycles 1 0 0 0 0 0: 33 HCLK cycles 1 0 0 0 0 1: 34 HCLK cycles* 1 0 0 0 1 0: 35 HCLK cycles 1 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: 44 HCLK cycles* 1 0 1 1 0 0: 45 HCLK cycles 1 0 1 1 0 1: 46 HCLK cycles* 1 0 1 1 1 0: 47 HCLK cycles 1 0 1 1 1 1: 48 HCLK cycles* 1 1 0 0 0 0: 49 HCLK cycles 1 1 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	R/W

			1 1 0 1 0 1: 54 HCLK cycles 1 1 0 1 1 0: 55 HCLK cycles 1 1 0 1 1 1: 56 HCLK cycles 1 1 1 0 0 0: 57 HCLK cycles 1 1 1 0 0 1: 58 HCLK cycles 1 1 1 0 1 0: 59 HCLK cycles 1 1 1 0 1 1: 60 HCLK cycles 1 1 1 1 0 0: 61 HCLK cycles 1 1 1 1 0 1: 62 HCLK cycles 1 1 1 1 1 0: 63 HCLK cycles 1 1 1 1 1 1: 64 HCLK cycles	
b15~b14	Reserved	—	Read as "0", write as "0"	R/W
b13~b12	DPRSL[1:0]	SPI protocol selection in data reception stage	SPI protocol selection in data reception stage. b1 b0 0 0: Extended SPI protocol 0 1: Dual SPI protocol 1 0: Quad SPI protocol 1 1: Prohibitions	R/W
b11~b10	APRSL[1:0]	SPI protocol selection in address sending stage	SPI protocol selection in address sending stage. b1 b0 0 0: Extended SPI protocol 0 1: Dual SPI protocol 1 0: Quad SPI protocol 1 1: Prohibitions	R/W
b9~b8	IPRSL[1:0]	SPI protocol selection in instruction sending stage	SPI protocol selection in instruction sending stage. b1 b0 0 0: Extended SPI protocol 0 1: Dual SPI protocol 1 0: Quad SPI protocol 1 1: Prohibitions	R/W
B7	SPIMD3	SPI mode selection	SPI mode selection 0: SPI mode 0 1: SPI mode 3	R/W
b6	XIPE	XIP mode enable	0: XIP mode disabled 1: XIP mode enable	R/W
b5	DCOME	Direct communication enable	QSPI bus communication mode selection 0: ROM access mode 1: Direct communication mode	R/W
b4	PFSAE	Prefetch immediately stop enable	Choose the location for stopping the prefetch 0: The current prefetch is stopped at the byte boundary 1: The current prefetch is stopped immediately	R/W
b3	PFE	Prefetch enable	Prefetch function valid/invalid selection 0: The prefetch function is disable 1: The prefetch function is enabled	R/W
b2~b0	MDSEL[2:0]	QSPI read mode selection	Serial interface read mode selection b2 b1 b0 0 0 0: Standard Read 0 0 1: Fast Rread 0 1 0: Fast Read Dual Output 0 1 1: Fast Read Dual I/O 1 0 0: Fast Read Quad Output 1 0 1: Fast Read Quad I/O 1 1 0: Custom protocol Standard Read 1 1 1: Custom protocol Fast Read	R/W

32.13.2 QSPI Chip Select Control Register (QSPI_CSCR)

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]			

Bit	Symbol	Bit name	Description	Read and write
b31~b6	Reserved	-	Read as "0", write as "0"	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: Do not extend QSSN valid time 0 1: Extend QSSN valid time by 32 QSCK cycles 1 0: Extend QSSN valid time by 128 QSCK cycles 1 1: Extend QSSN valid time indefinitely	R/W
b3~b0	SSHW[3:0]	QSSN minimum invalid time setting	QSSN signal minimum invalid time selection 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	R/W

32.13.3 QSPI Format Control Register (QSPI_FCR)

b31	b30	b29	b28	b27	b26	b25	b24
—	—	—	—	—	—	—	—
b23	b22	b21	b20	b19	b18	b17	b16
—	—	—	—	—	—	—	—
b15	b14	b13	b12	b11	b10	b9	b8
DUTY	—	—	—	DMCYCN[3:0]			
B7	b6	b5	b4	b3	b2	b1	b0
—	WPOL	SSNLD	SSNHD	—	4BIC	AWSL[1:0]	

Bit	Symbol	Bit name	Description	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15	DUTY	Duty cycle correction	QSCK output duty cycle correction 0: No duty cycle correction 1: Delay the rising edge of QSCK by 0.5 HCLK cycles (effective when the frequency selected by QSCK is an odd multiple of HCLK)	R/W
b14~b12	Reserved	-	Read as "0", write as "0"	R/W
b11~b8	DMCYCN[3:0]	Dummy cycle settings	Number of dummy cycles selection 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: 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 as "0", write as "0"	R/W
b6	WPOL	WP pin output level setting	WP pin (QIO2) level setting 0: Low level 1: High level	R/W
b5	SSNLD	QSSN signal output time delay setting	QSSN signal output timing selection 0: Output QSSN 0.5 QSCK cycles before the first rising edge of QSCK 1: Output QSSN 1.5 QSCK cycles before the first rising edge of QSCK	R/W
b4	SSNHD	QSSN signal release time delay setting	QSSN signal release timing selection 0: Release QSSN 0.5 QSCKcycles after the last rising edge of QSCK 1: Release QSSN 1.5 QSCK cycles after the last rising edge of QSCK	R/W
b3	Reserved	-	Read as "0", write as "0"	R/W
b2	4BIC	4-byte address read instruction code selection	Read instruction code selection when address width is 4 bytes 0: Do not use 4-byte address read instruction code 1: Use 4-byte address read instruction code	R/W
b1~b0	AWSL[1:0]	Address width selection	Serial interface address width selection b1 b0 0 0: 1 byte 0 1: 2 bytes 1 0: 3 bytes 1 1: 4 bytes	R/W

*1 In order to avoid conflict when the QIO0 pin switches the input and output states, please select a dummy cycle of more than 4 QSPICK cycles.

32.13.4 QSPI Status Register (QSPI_SR)

b31	b30	b29	b28	b27	b26	b25	b24
-	-	-	-	-	-	-	-
b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-
b15	b14	b13	b12	b11	b10	b9	b8
PFAN	PFFUL	-		PFNUM[4:0]			
B7	b6	b5	b4	b3	b2	b1	b0
RAER	XIPF	-	-	-	-	-	BUSY

Bit	Symbol	Bit name	Description	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15	PFAN	Prefetch action status	Prefetch operation status signal 0: Prefetch is active 1: Prefetch is stopped	R
b14	PFFUL	Prefetch buffer status	Prefetch buffer status signal 0: The prefetch buffer is not full 1: The prefetch buffer data is full	R
b13	Reserved	-	Read as "0", write as "0"	R/W
b12~b8	PFNUM[4:0]	The number of data bytes stored in the prefetch buffer	Display of the number of data bytes stored in the prefetch buffer b4 b3 b2 b1 b0 0 0 0 0 0: 0 bytes 0 0 0 0 1: 1 byte 0 0 0 1 0: 2 bytes 0 0 0 1 1: 3 bytes 0 0 1 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: 16 bytes 1 0 0 0 1: 17 bytes 1 0 0 1 0: 18 bytes The other value is invalid	R
B7	RAER*1	ROM access error flag	Error flag bit for ROM access in direct communication mode 0: No ROM access is detected 1: ROM access is detected	R/W
b6	XIPF	XIP mode flag	XIP mode status signal 0: non-XIP mode 1: XIP mode	R
b5~b1	Reserved	-	Read as "0", write as "0"	R/W
b0	BUSY	Bus busy flag	QSPI bus working status flag in direct communication mode 0: The bus is idle, there is no serial transmission process 1: The bus is busy and the serial transmission process is in progress	R

*1: RAER needs to be cleared by the RAERCLR bit of QSPI_SR2

32.13.5 QSPI Direct Communication Register (QSPI_DCOM)

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]							

Bit	Symbol	Bit name	Description	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
b7~b0	DCOM[7:0]	Direct communication mode instructions	The interface in direct communication mode communicates directly through QSPI bus. Read and write accesses to the interface will be converted into a corresponding QSPI bus cycle. This interface is only valid in direct communication mode, and access to this interface is prohibited in ROM access mode.	R/W

32.13.6 QSPI Command Code Register (QSPI_CCMD)

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]							

Bit	Symbol	Bit name	Description	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
b7~b0	RIC[7:0]	Replace instruction code	Serial flash instruction code to replace the default instruction	R/W

32.13.7 QSPI XIP Mode Code Register (QSPI_XCMD)

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]							

Bit	Symbol	Bit name	Description	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
b7~b0	XIPMC[7:0]	XIP mode code	Mode code for serial flash. (Set XIP mode)	R/W

32.13.8 QSPI Status Flag Clear Register (QSPI_SR2)

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	-	-	-	-	-	-	-

Bit	Symbol	Bit name	Description	Read and write
b31~b8	Reserved	-	Write "0" on write	W
B7	RAERCLR	RAER clear	Write 1 to clear the RAER bit in QSPI_SR	W
b6~b0	Reserved	-	Write "0" on write	W

32.13.9 QSPI External Address Register (QSPI_EXAR)

b31	b30	b29	b28	b27	b26	b25	b24
EXADR[5:0]						—	—
b23	b22	b21	b20	b19	b18	b17	b16
—	—	—	—	—	—	—	—
b15	b14	b13	b12	b11	b10	b9	b8
—	—	—	—	—	—	—	—
B7	b6	b5	b4	b3	b2	b1	b0
—	—	—	—	—	—	—	—

Bit	Symbol	Bit name	Description	Read and write
b31~b26	EXADR[5:0]	External extended address code	Set the upper 6 bits of the QSPI external address, and cooperate with the ROM access window address of QSPI, the maximum external ROM space of 64MB×63 blocks can be accessed.	R/W
b25~b0	Reserved	-	Read as "0", write as "0"	R/W

33 Integrated circuit built-in audio bus module (I2S)

33.1 Introduction

I2S (Inter_IC Sound Bus), an integrated circuit built-in audio bus, which is dedicated to data transmission between audio devices.

Table 33-1 I2S main features

Function	Main characteristics
Communication mode	<ul style="list-style-type: none">Supports full-duplex and half-duplex communicationSupport host mode or slave mode operation
Data format	<ul style="list-style-type: none">Optional channel lengths: 16/32 bitsOptional transmit data length: 16/24/32bitsData shift order: MSB starts
Baud rate	<ul style="list-style-type: none">8-bit programmable linear prescaler for precise audio sampling frequencySupport sampling frequency 192k, 96k, 48k, 44.1k, 32k, 22.05k, 16k, 8kCan output drive clock to drive external audio components, the ratio is fixed at 256*Fs (Fs is the audio sampling frequency)
Support I2S protocol	<ul style="list-style-type: none">I2S Philips StandardMSB alignment criteriaLSB alignment criteriaPCM standard
Data buffer	<ul style="list-style-type: none">With 4-word deep, 32-bit wide input and output FIFO buffer area
Timer	<ul style="list-style-type: none">Can use internal I2SCLK (PLLAR/PLLAQ/PLLAP/PLLHR/PLLHQ/PLLHP); can also be provided by external clock on I2S_EXCK pin
Interrupt	<ul style="list-style-type: none">An interrupt is generated when the valid space of the send buffer reaches the alarm thresholdAn interrupt is generated when the valid space of the receive buffer reaches the alarm thresholdThe receive data area is full, there is still a write data request, and the receive overflowsThe send data area is empty and there is still a send request, send underflowThe send data area is full and there is still a write data request, sending overflow

33.2 I2S system block diagram

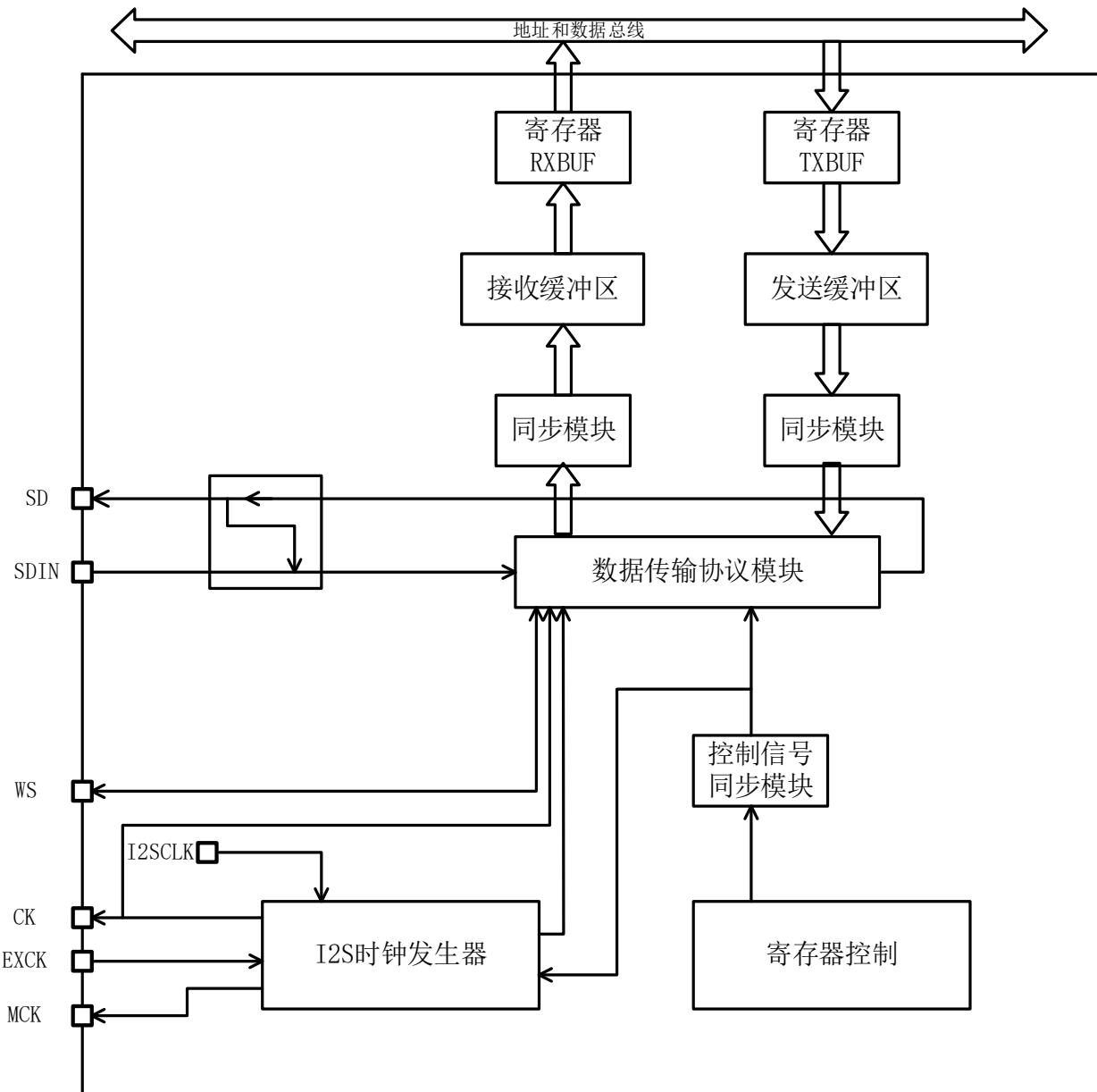


Figure 33-1 I2S system block diagram

33.3 Pin description

Table 33-2 I2S pin description

Pin name	Direction	Functional description
I2Sn_CK	Input output	communication clock
I2Sn_WS	Input output	word choice
I2Sn_SD	Input 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

33.4 Functional description

This chapter will describe in detail the functions of I2S.

33.4.1 I2S General Instructions

I2S pin function

- I2Sn_SD: Serial data for half-duplex mode data input, or half/full-duplex mode data output.
- I2Sn_WS: Word selection is the data control signal output in master mode and the data control signal input in slave mode.
- I2Sn_CK: The serial clock is the serial clock output in master mode and the serial clock input in slave mode.
- I2Sn_EXCK: The external clock source is that the clock generator in the host mode selects an external clock as the frequency-divided clock source.
- I2Sn_SDIN: Pin for serial data input in I2S full-duplex mode.
- I2Sn_MCK: When I2S is configured as host mode (and MCKOE bit is 1), use the drive clock (separate mapping) to output this additional clock. The clock output frequency is $256 \times F_s$, where F_s is the audio signal sampling frequency.

I2S uses its own clock generator to generate the communication clock in the host mode. This clock generator is also the source that drives the clock output.

33.4.2 Communication mode

Both half-duplex and full-duplex communication modes are supported, which can be selected by the DUPLEX bit of the I2S control register (I2S_CTRL).

When the I2S_CTRL.DUPLEX bit is 0, I2S operates in half-duplex communication mode, using the I2Sn_SD pin as the output data pin when only transmitting, and using the I2Sn_SD pin as the input data pin when only receiving.

When the I2S_CTRL.DUPLEX bit is 1, I2S operates in full-duplex communication mode. At this time, the I2Sn_SDIN pin is used as the input data pin, and the I2Sn_SD pin is used as the output data pin to realize full-duplex communication.

33.4.3 Supported Audio Protocols

There are four data and frame format combinations and data can be sent in the following formats:

- Pack 16-bit data in a 16-bit frame
- Pack 16-bit data in a 32-bit frame
- Pack 24-bit data into 32-bit frames
- Pack 32-bit data in a 32-bit frame

When using 16-bit data in a 32-bit packet, the first 16 bits (MSB) are valid bits, and the 16-bit LSB is forced to be cleared without any software operation (only one read/write operation).

When using 24-bit data in a 32-bit packet, the first 24 bits (MSB) are valid bits, and the 8-bit LSB is forcibly cleared without any software operation (only one read/write operation).

For all data formats and communication standards, the most significant bit is always sent first (MSB first).

The I2S interface supports four audio protocols, which can be configured using the I2SSTD[1:0] and PCMSYNC bits in the I2S_CFGR register.

33.4.3.1 I2S Philips Standard

The WS signal is used to indicate the channel to which the data currently being sent belongs. This signal is valid from one clock before the first bit (MSB) of the current channel data.

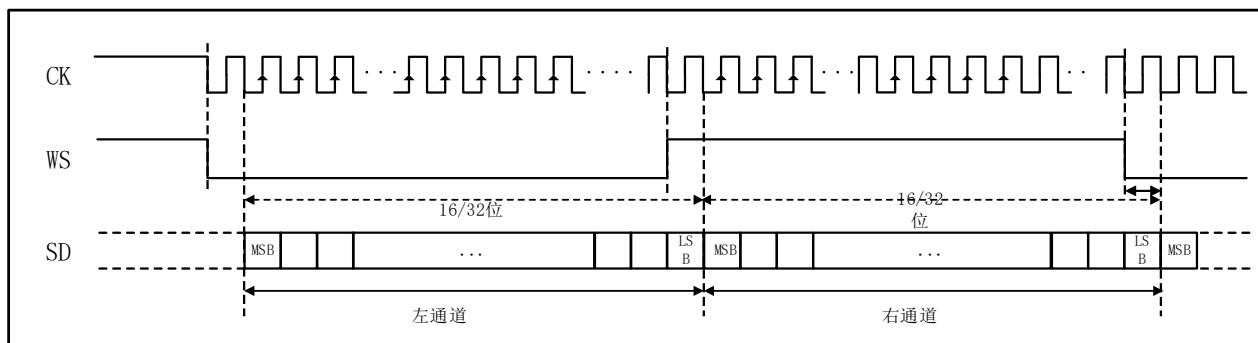


Figure 33-2 I2S Philips protocol waveform (16/32 bit full precision)

16 bits are loaded in a 16-bit frame. In transmit mode, write to I2S_TXBUF register 0xXXXX3344, SD outputs serial data 0x3344; in receive mode, SD inputs serial data 0xeedd, and I2S_RXBUF register reads data 0x0000eedd.

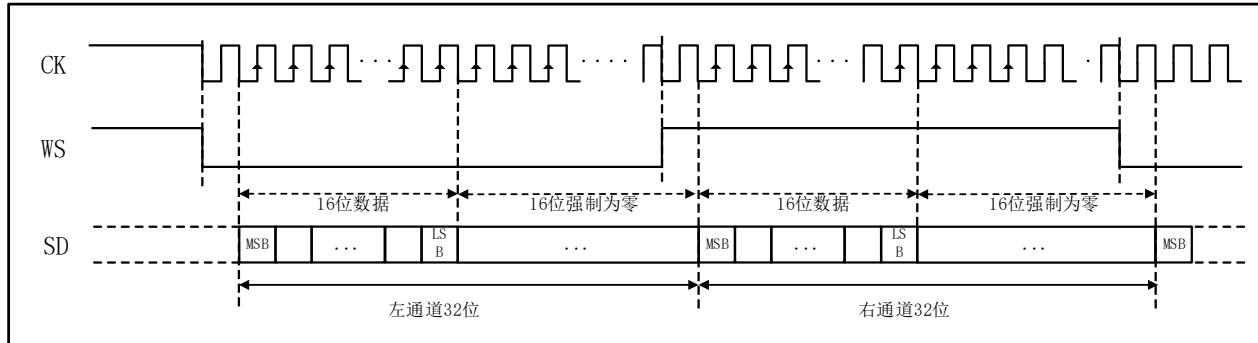


Figure 33-3 I2S Philips protocol waveform (16-bit data packed in 32-bit frames)

16 bits are loaded in a 32-bit frame, in transmit mode, write to I2S_TXBUF register 0xXXXX3344, SD output serial data 0x33440000; in receive mode, SD input serial data 0xeeddXXXX, I2S_RXBUF register read data 0x0000eedd.

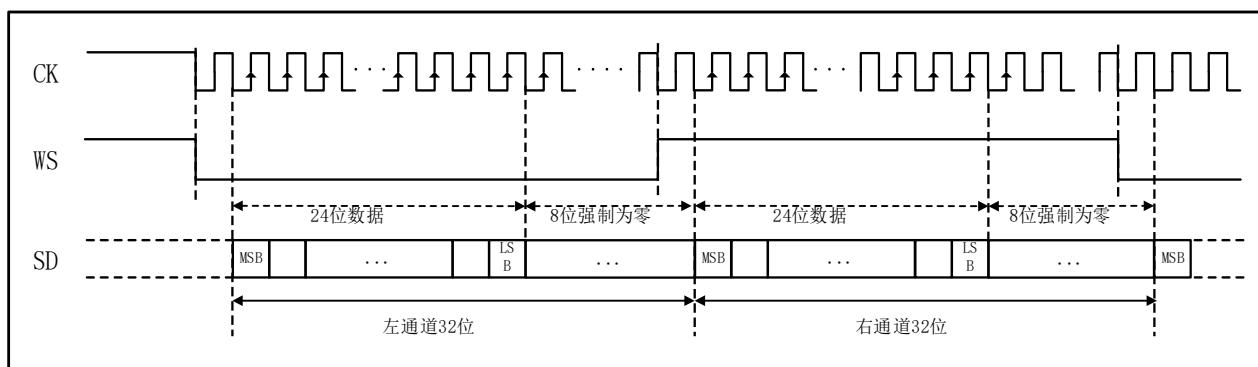


Figure 33-4 I2S Philips protocol waveform (24-bit data packed in 32-bit frame)

24-bit is loaded in a 32-bit frame, in transmit mode, write to I2S_TXBUF register 0xXX223344, SD output serial data 0x22334400; in receive mode, SD input serial data 0xeedd11XX, I2S_RXBUF register read data 0x00eedd11.

33.4.3.2 MSB alignment criteria

This standard generates both the WS signal and the first data bit (ie MSBit).

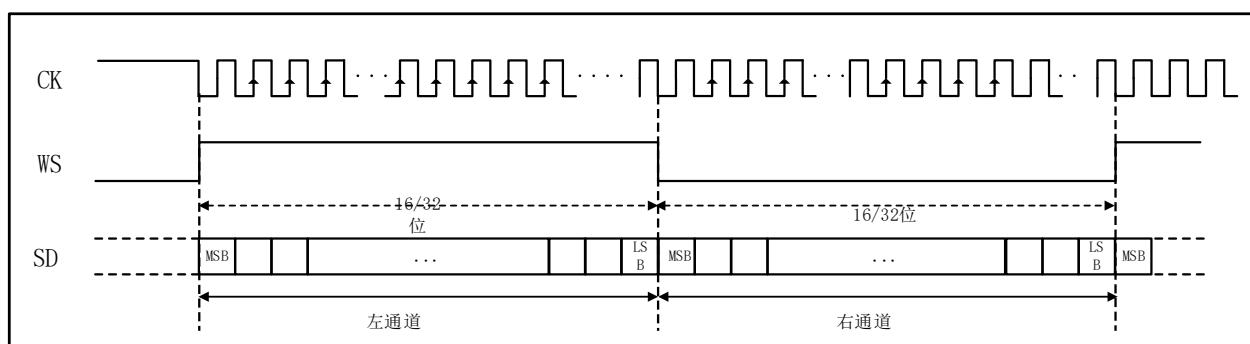


Figure 33-5 I2S MSB protocol waveform (16/32-bit full precision)

The sender changes data on the falling edge of the clock signal; the receiver reads data on the rising edge.

16 bits are loaded in a 16-bit frame. In transmit mode, write to I2S_TXBUF register 0xXXXX3344, SD outputs serial data 0x3344; in receive mode, SD inputs serial data 0xeedd, and I2S_RXBUF register reads data 0x0000eedd.

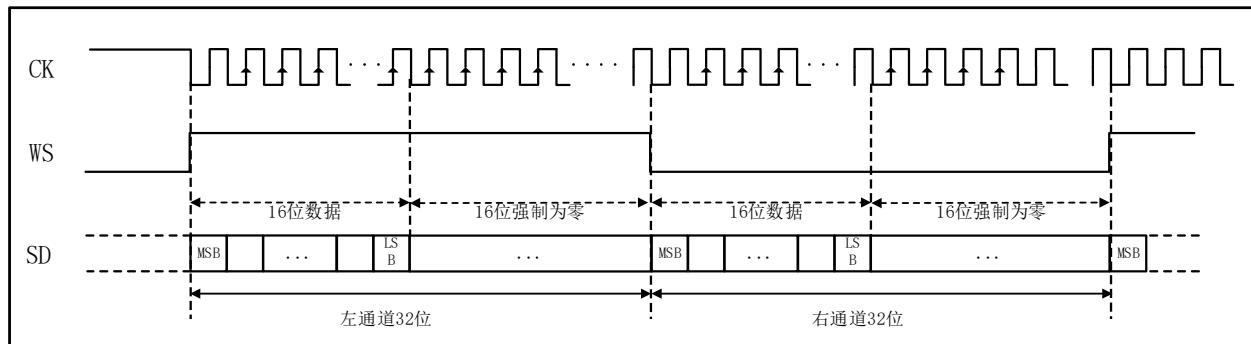


Figure 33-6 I2S MSB protocol waveform (16-bit data packed in 32-bit frame)

16 bits are loaded in a 32-bit frame, in transmit mode, write to I2S_TXBUF register 0xXXXX3344, SD output serial data 0x33440000; in receive mode, SD input serial data 0xeeddXXXX, I2S_RXBUF register read data 0x0000eedd.

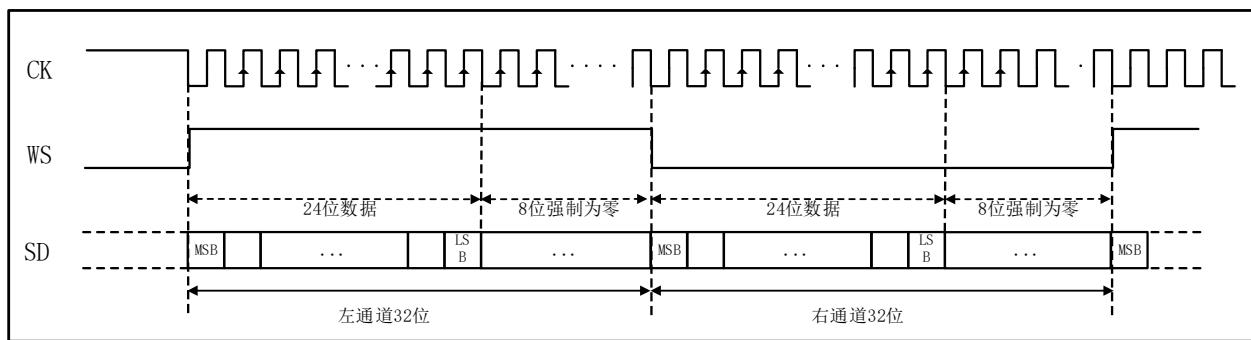


Figure 33-7 I2S MSB protocol waveform (24-bit data packed in 32-bit frame)

24-bit is loaded in a 32-bit frame, in transmit mode, write to I2S_TXBUF register 0xXX223344, SD output serial data 0x22334400; in receive mode, SD input serial data 0xeedd11XX, I2S_RXBUF register read data 0x00eedd11.

33.4.3.3 LSB alignment criteria

The standard is similar to the MSB alignment standard (for 16-bit and 32-bit full-precision frame formats, there is no difference).

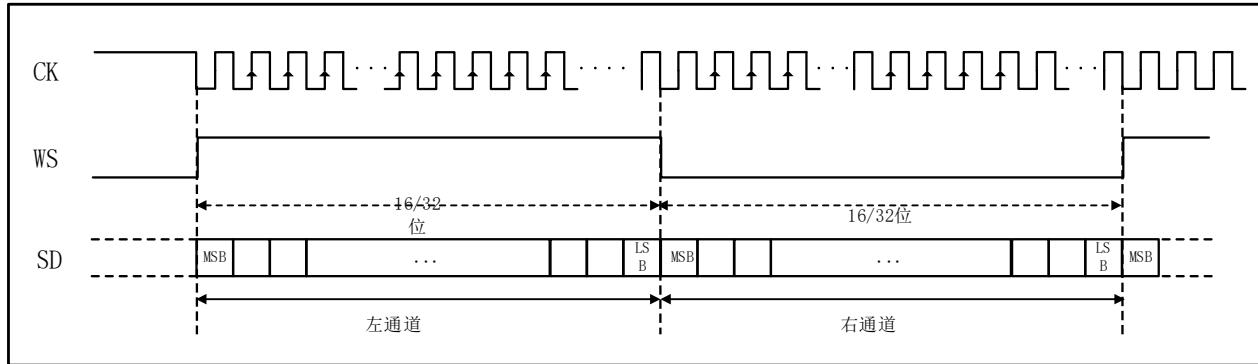


Figure 33-8 I2S LSB protocol waveform (16/32-bit full precision)

16 bits are loaded in a 16-bit frame. In transmit mode, write to I2S_TXBUF register 0xXXXX3344, SD outputs serial data 0x3344; in receive mode, SD inputs serial data 0xeedd, and I2S_RXBUF register reads data 0x0000eedd.

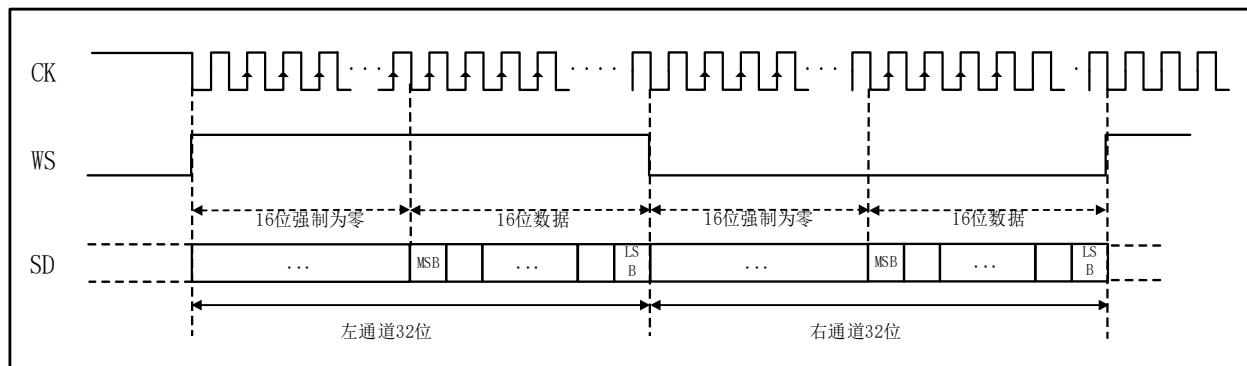


Figure 33-9 I2S LSB protocol waveform (16-bit data packed in 32-bit frame)

16 bits are loaded in a 32-bit frame, in transmit mode, write to I2S_TXBUF register 0xXXXX3344, SD output serial data 0x00003344; in receive mode, SD input serial data 0xXXXX1122, I2S_RXBUF register read data 0x00001122.

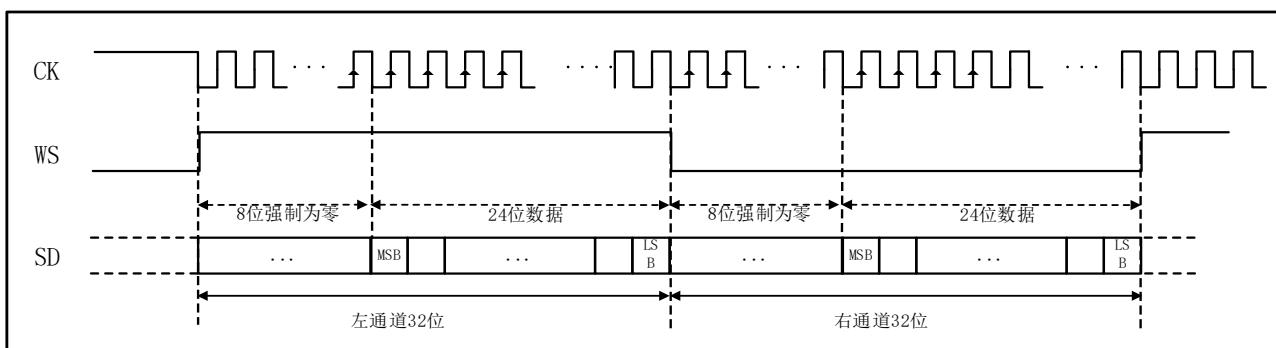


Figure 33-10 I2S LSB protocol waveform (24-bit data packed in 32-bit frame)

24-bit is loaded in a 32-bit frame, in transmit mode, write to I2S_TXBUF register 0xXX223344, SD output serial data 0x00223344; in receive mode, SD input serial data 0xXXdd1122, I2S_RXBUF register read data 0x00dd1122.

33.4.3.4 PCM standard

For the PCM standard, no channel information needs to be used. There are two PCM modes (short frame and long frame) and can be configured using the PCMSYNC bit in I2S_CFGR.

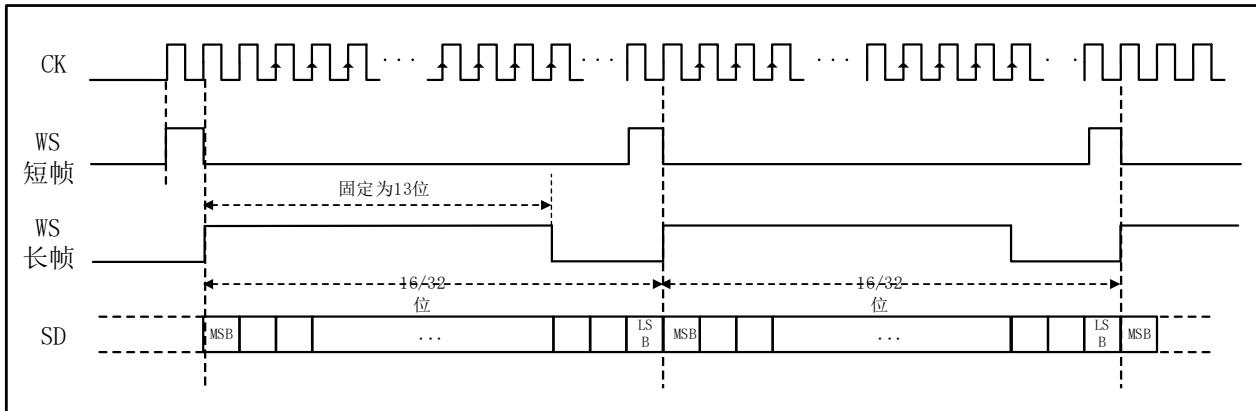


Figure 33-11 I2S PCM protocol waveform (16/32-bit full precision)

For long frame sync, the WS signal is held for 13 cycles in master mode.

For short frame sync, the duration of the WS sync signal is only one cycle.

16 bits are loaded in a 16-bit frame. In transmit mode, write to I2S_TXBUF register 0xXXXX3344, SD outputs serial data 0x3344; in receive mode, SD inputs serial data 0xeedd, and I2S_RXBUF register reads data 0x0000eedd.

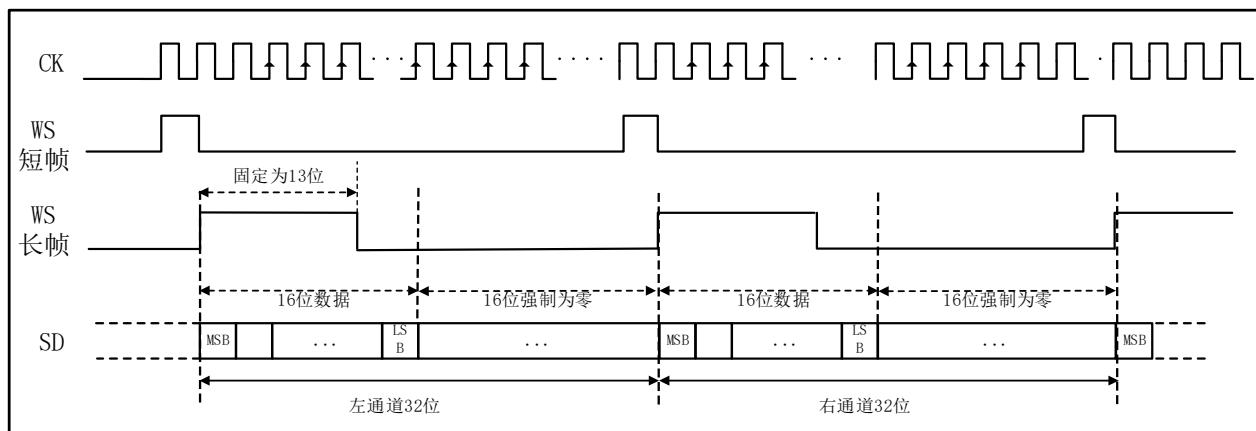


Figure 33-12 I2S PCM protocol waveform (16-bit data packed in 32-bit frame)

16 bits are loaded in a 32-bit frame. In transmit mode, write to I2S_TXBUF register 0xXXXX3344, SD outputs serial data 0x00003344; in receive mode, SD inputs serial data 0xeeddXXXX, and I2S_RXBUF register reads data 0x0000eedd.

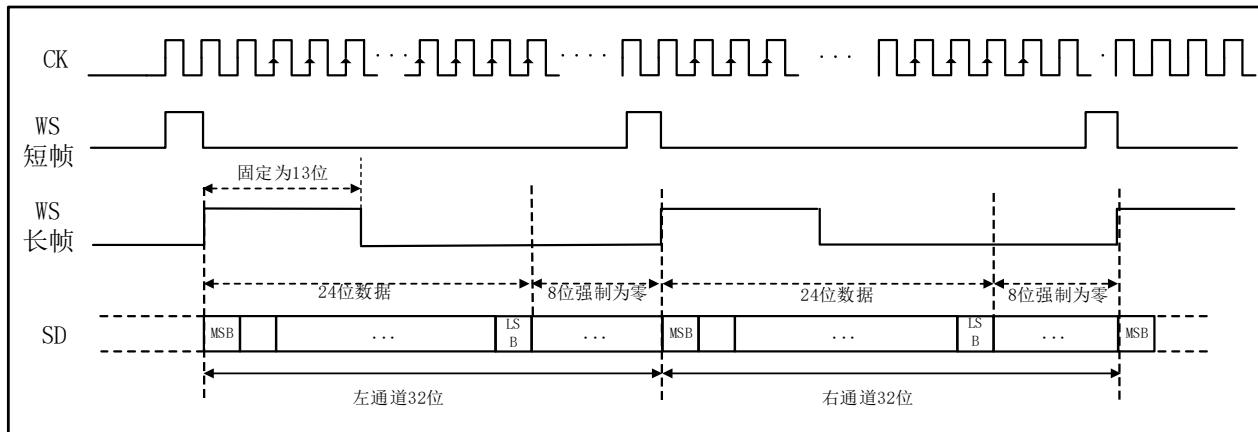


Figure 33-13 I2S PCM protocol waveform (24-bit data packed in 32-bit frame)

24-bit is loaded in a 32-bit frame, in transmit mode, write to I2S_TXBUF register 0xXX223344, SD output serial data 0x00223344; in receive mode, SD input serial data 0xeedd11XX, I2S_RXBUF register read data 0x00eedd11.

Note:

- For two modes (host/slave mode) and two synchronizations (short/long synchronization), even in slave mode, you need to specify the number of bits between two sets of continuous data (and two synchronization signals) (I2S_CFGR register) DATLEN bit and CHLEN bit in).

33.4.4 Clock generator

The I2S bit rate is used to determine the data flow on the I2S data line and the frequency of the I2S clock signal.

I2S bit rate = bits per channel × number of channels × audio sampling frequency

For 16-bit dual-channel audio, the formula for the I2S bit rate is as follows: I2S bit rate = $16 \times 2 \times F_s$. If the packet is 32 bits wide, then I2S bitrate = $32 \times 2 \times F_s$.

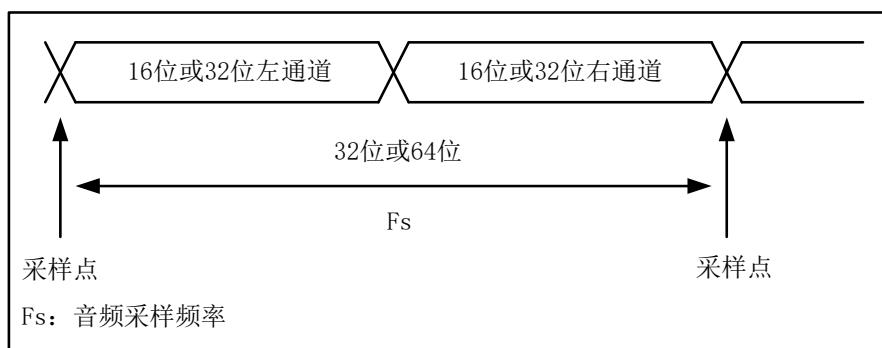


Figure 33-14 Audio sampling frequency definition

When configuring master mode, the linear divider needs to be set up correctly to communicate at the desired audio frequency.

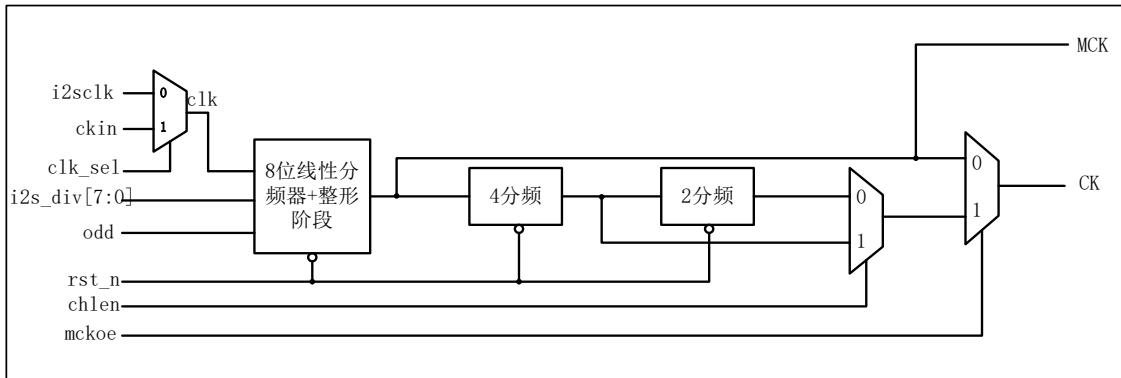


Figure 33-15 Clock Generator Architecture

The clk clock source can be the I2SCLK output or an external clock.

Audio sampling frequency may be 192kHz, 96kHz or 48kHz etc. To achieve the desired frequency, the current divider needs to be programmed according to the following formula:

When the output drives the clock (mckoe is set):

$$F_s = \text{clk} / [(16*2)*((2*I2SDIV+ODD)*8)] \text{ (when the channel frame width is 16 bits)}$$

$$F_s = \text{clk} / [(32*2)*((2*I2SDIV+ODD)*4)] \text{ (when the channel frame width is 32 bits)}$$

When the drive clock is turned off (mckoe is cleared):

$$F_s = \text{clk} / [(16*2)*(2*I2SDIV+ODD)] \text{ (when the channel frame width is 16 bits)}$$

$$F_s = \text{clk} / [(32*2)*(2*I2SDIV+ODD)] \text{ (when the channel frame width is 32 bits)}$$

The following table provides example precision values for different clock configurations. There are other configurations to achieve better clock accuracy.

Table 33-3 Audio frequency accuracy (for VCO input frequency = 1MHz)

Drive clock	Target fS(Hz)	Data format	Frequency multiplication factor	Output divider 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%

Drive clock	Target fS(Hz)	Data format	Frequency multiplication factor	Output divider ratio	I2SDIV	ODD	Real-time fS(Hz)	Error
Output enable	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%
Drive clock	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%

Note:

- The frequency multiplication factor can be PLLAN/PLLHN. When the frequency multiplication factor is PLLAN, the output frequency division ratio can be PLLAP/PLLAQ/PLLAR. When the frequency multiplication factor is PLLHN, the output frequency division ratio can be PLLHP/PLLHQ/PLLHR. For specific settings, refer to [Clock controller (CMU)] The CMU PLLA configuration register and the CMU PLLH configuration register.

33.4.5 I2S host mode

I2S can be configured as follows:

- Transmit master or receive master (half-duplex mode using I2S)
- A master device that transmits and receives simultaneously (using I2S in full duplex mode).

I2S works in the host mode, the serial clock is output by the pin CK, and the word selection signal is generated by the pin WS. The drive clock (MCK) can be selected to be output or not output by setting the MCKOE bit in register I2S_CTRL.

Step

1. Set the pins to be used for I2S.
2. The clock source is selected by the I2S_CTRL.CLKSEL, I2S_CTRL.I2SPLLSEL bits.
3. Set the I2S_PR.I2SDIV[7:0] bits and the 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 through the I2SSTD[1:0] and PCMSYNC bits, select the data length through the DATLEN[1:0] bits and select the number of bits per channel through the configuration CHLEN bits.
5. If you need to use interrupt, set the interrupt register of the system.
6. To use DMA, please set the relevant registers of DMA.

7. Set I2S control register (I2S_CTRL), including working mode setting, communication mode setting, clock output permission setting, data output permission setting, FIFO reset setting, sending/receiving buffer threshold setting, etc., working mode WMS Bit select I2S host mode.
8. Set the interrupt permission bit.
9. Set I2S_CTRL.TXE and I2S_CTRL.RXE and the action starts.

Note:

- When using TXIRQOUT interrupt to write communication data to TXBUF, if two data are written in each interrupt, first turn off the transmit interrupt enable flag TXIE after the interrupt starts, and then turn on TXIE after the data is written. Or write only one data per interrupt.

Send sequence

The TXE bit in the I2S_CTRL register is set to enable transmission. The transmit sequence begins as soon as 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 is no partial frame where only the left channel is sent. During the first bit transmission, the data is loaded into the shift register in parallel, then shifted serially and output to the SD pin (MSB first). For more details on write operations in the various I2S standard modes, see 33.4.3Supported Audio Protocols .

Receive sequence

This working mode is basically the same as the sending mode, only the sending and receiving settings of the I2S_CTRL register are different. Set the RXE bit to 1 to allow reception. For more details on read operations in the various I2S standard modes, see 33.4.3Supported Audio Protocols . If new data is received before previously received data has been read, an overrun error will be generated and the I2S_ER.RXERR flag will be set. If the I2S_CTRL.EIE bit is set, an interrupt will be generated to indicate the error.

33.4.6 I2S slave mode

I2S can be configured as follows:

- Transmit Slave or Receive Slave (Half-duplex mode using I2S)
- Slave devices that transmit and receive simultaneously (using I2S in full duplex mode).

The rules followed by this working mode are basically the same as the I2S master mode. In slave mode, the I2S interface does not generate a clock. Clock and WS signals are input from an external master connected to the I2S interface. In this way, the user does not need to configure the clock.

Step

1. Set the pins to be used for I2S.
2. Set the I2S configuration register (I2S_CFGR), select the I2S standard through the

I2SSTD[1:0] and PCMSYNC bits, select the data length through the DATLEN[1:0] bits and select the number of bits per channel through the configuration CHLEN bits.

3. If you need to use interrupt, set the interrupt register of the system.
4. To use DMA, please set the relevant registers of DMA.
5. To send data, you should pre-write 1~4 data to be sent into I2S_TXBUF.
6. Set I2S control register (I2S_CTRL), including working mode setting, communication mode setting, clock output permission setting, data output permission setting, FIFO reset setting, sending/receiving buffer threshold setting, etc., working mode WMS bit selects I2S slave mode.
7. Set the interrupt permission bit.
8. Set I2S_CTRL.TXE and I2S_CTRL.RXE and the action starts.

Send sequence

The transmit sequence begins when WMS is set to 1, TXE is set to 1, the external master sends the clock and requests data transfer via the WS signal. When communication begins, data is transferred from the transmit buffer to the shift register. During the first bit transmission, data is loaded from the internal bus into the shift register in parallel, then shifted serially and output to the SD pin (MSB first). An interrupt will be generated if the I2S_CTRL.TXIE bit is set every time the transmit buffer FIFO space is greater than the set threshold. For more details on write operations in the various I2S standard modes, see 33.4.3Supported Audio Protocols .

To ensure continuous audio data transmission, the next data to be transmitted must be written into the TX FIFO before the current data transmission ends. If the first clock edge of the next data communication arrives before the data has been written into the TX FIFO, a transmit underflow will occur, the I2S_ER.TXERR flag will be set to 1 and an interrupt may be generated. If the I2S_CTRL.EIE bit is set, an interrupt will be generated when the I2S_ER.TXERR flag becomes 1.

Receive sequence

This working mode is basically the same as the sending mode, only the sending and receiving settings of the I2S_CTRL register are different. Set the RXE bit to 1 to allow reception. For more details on read operations in the various I2S standard modes, see 33.4.3Supported Audio Protocols . If new data is received before previously received data has been read, a receive overflow error will be generated and the RXERR flag will be set. If the I2S_CTRL.EIE bit is set, an interrupt will be generated to indicate the error.

33.4.7 I2S interrupt

The interrupt sources of I2S are that the effective space of the sending buffer is greater than the alarm threshold, the effective space of the receiving buffer is less than the alarm threshold, the receiving overflow, the sending underflow and the sending overflow. The transmit underflow and transmit overflow are integrated as the I2S transmit error interrupt TXERR, so the actual interrupt source needs to be judged by the flag. The specific description of the I2S interrupt source is as followsTable 33-4 shown. Once the interrupt condition is established, the corresponding interrupt request is generated.

Users can write the vector corresponding to the event trigger source into different trigger object registers to implement various event trigger functions.

For the vector corresponding to the event trigger source, refer to [Interrupt controller (INTC)].

Table 33-4 is a list of interrupts for I2S.

Table 33-4 I2S interrupt request

Interrupt event	Event flag	Enable control bit
The effective space of the send buffer is greater than the alarm threshold	TXBA	TXIE
The valid space of the receive buffer is less than the alarm threshold	RXBA	RXIE
The receive data area is full, there is still a write data request, and the receive overflows	RXERR	EIE
The send data area is empty and there is still a send request, send underflow	TXERR	EIE
The send data area is full and there is still a write data request, sending overflow	TXERR	EIE

33.4.8 Precautions for use

33.4.8.1 Precautions when using as a host

1. When I2S acts as the host to only send data, if all data in I2S_TXBUF has been sent and no new data is written, I2S will suspend the action after the last data is sent, and I2S will no longer generate communication clock, the transmit error flag TXERR will be set to 1. At this time, the user can choose to write the I2S_CTRL.TXE bit to 0 to close the I2S, or write new transmit data to I2S_TXBUF to continue the transmit action. WS will restart from the left channel when continuing to send (Philips, MSB/LSB mode). If the I2S_CTRL.TXE bit is directly written to 0 during the sending action, the I2S will be closed immediately, and the current data transmission will be terminated. This approach will cause the slave state to be unpredictable, and the slave will receive data confusion when the communication is restarted without resetting the slave. Therefore, it is recommended that the user write the I2S_CTRL.TXE bit to 0 to close the I2S when the I2S is in a suspended state.
2. When I2S acts as the host for data receiving only, if you want to temporarily stop the receiving operation, you can write two frames of dummy data in advance. When you need to pause, count to the corresponding position and set I2S_CTRL.TXE to 1. When the baud rate is 8k~96k, it will be advanced in advance. 4 data TXEs are set to 1, and 5 data TXEs are set to 1 in advance when the baud rate is 192k. When the two frames of dummy data are sent, the I2S will suspend the action. At this time, the I2S will no longer generate the communication clock, and the transmission error flag TXERR will be set to 1. At this time, the user can choose to clear I2S_CTRL.TXE and I2S_CTRL.RXE to close I2S, or write another frame of dummy data to I2S_TXBUF to restart the communication action, and restart the TXE when it restarts to receive the first data after a pause, and restart the action WS will restart from the left channel (Philips, MSB/LSB mode). When the communication clock is regenerated, clear I2S_CTRL.TXE to return to the receiving-only state. Please refer to the figure below for details.

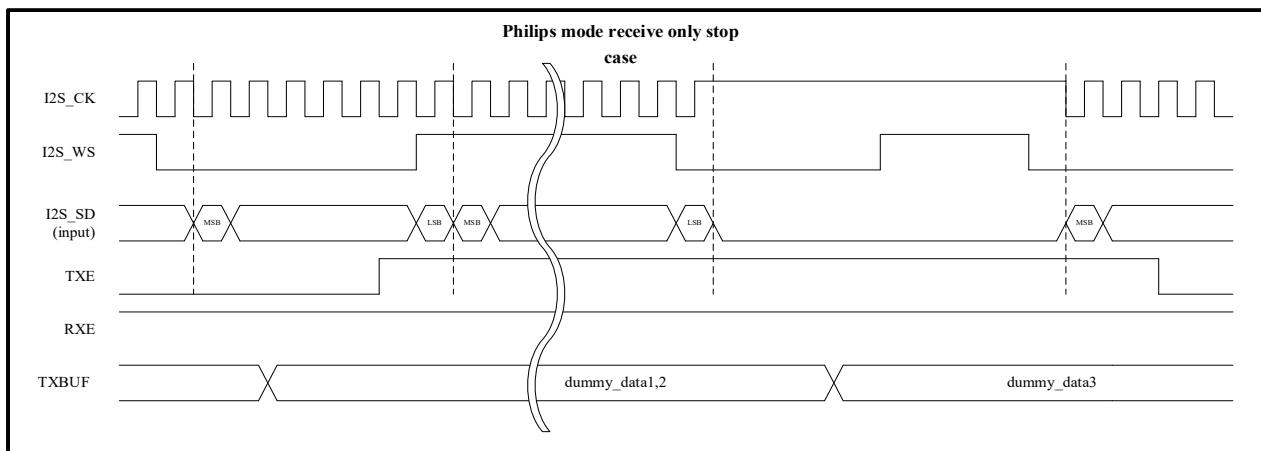


Figure 33-16 The host only receives and temporarily stops receiving

If the I2S_CTRL.RXE bit is directly written to 0 during the receiving operation, the I2S will be closed immediately, and the current data reception will be terminated. This approach will lead to the inability to grasp the state of the slave, and the data sent by the slave will be confused when the communication is restarted without resetting the slave. Therefore, it is recommended that the user clear the I2S_CTRL.TXE and I2S_CTRL.RXE bits to zero when the I2S is in a suspended state. to turn off I2S.

3. When I2S performs full-duplex operation as the host, if all data in I2S_TXBUF has been sent and no new data is written, I2S will suspend the action after the last data is sent, and I2S will no longer generate communication. clock, the transmit error flag TXERR will be set to 1. At this time, the user can choose to clear I2S_CTRL.TXE and I2S_CTRL.RXE to close I2S, or write new transmit data to I2S_TXBUF to continue the transmission and reception. WS will restart from the left channel when continuing to send (Philips, MSB/LSB mode). If the I2S_CTRL.TXE and I2S_CTRL.RXE bits are directly written to 0 during the full-duplex operation, the I2S will be closed immediately, and the current data transmission and reception will be terminated. This approach will cause the slave state to be unpredictable, and the communication data will be confused when the communication is restarted without resetting the slave. Therefore, it is recommended that the user clear the I2S_CTRL.TXE and I2S_CTRL.RXE bits to close when the I2S is in a suspended state. I2S.
4. When I2S is used as the host to send PCM short frame data, when I2S is suspended because I2S_TXBUF has no new data writing action, there are two setting methods to restart the transmission, which method to choose depends on the data reception specification of the slave to determine.

If the slave needs to detect the state of WS every time it receives data, after I2S is suspended, it needs to set I2S_CTRL.TXE to 0, then write new transmit data to I2S_TXBUF, and then set I2S_CTRL.TXE to 1 to restart transmission. The specific actions are shown in the figure below.

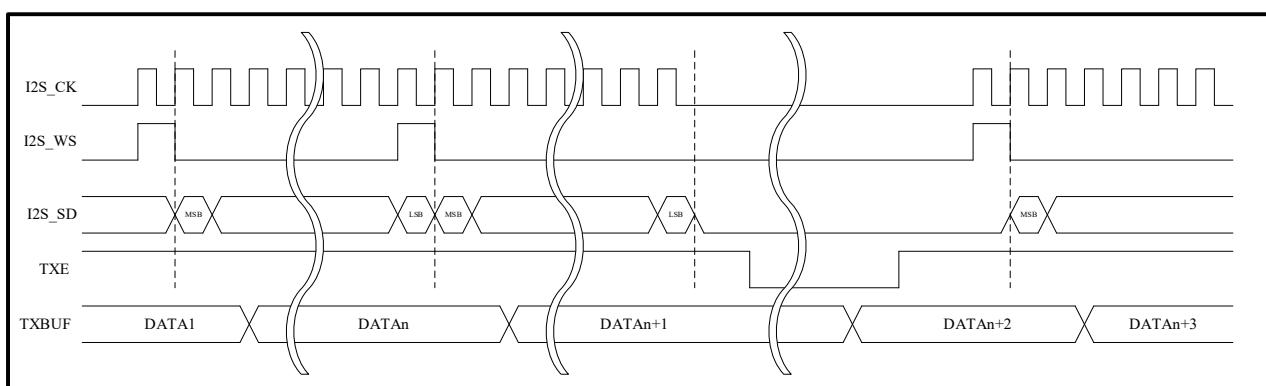


Figure 33-17 Mode 1 of PCM short frame retransmission after host transmission pauses

If the slave only detects the WS state when it receives the first frame of data, after I2S is suspended, it can directly write new transmit data to I2S_TXBUF to restart transmission. The specific actions are shown in the figure below.

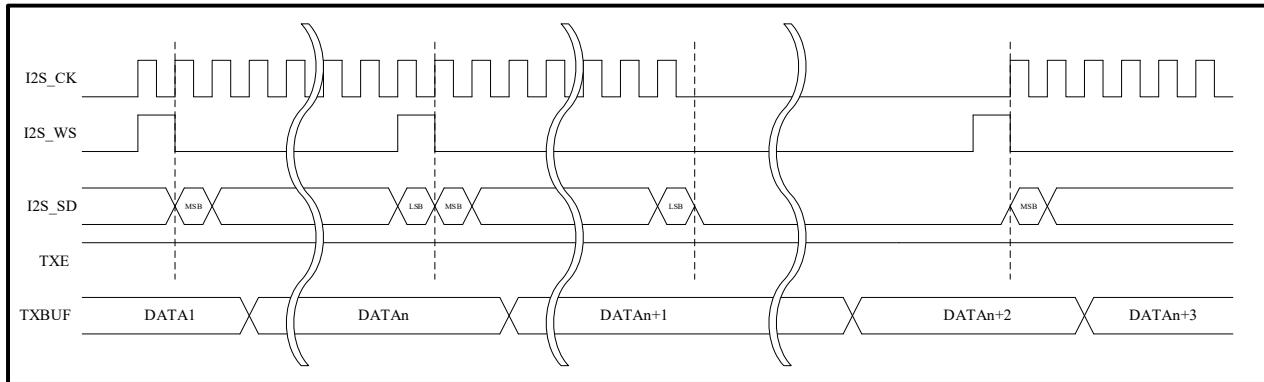


Figure 33-18 Mode 2 of PCM short frame retransmission after host sending pause

33.4.8.2 Precautions when using as a slave

When I2S acts as a slave, it is necessary to ensure that after all register configurations are completed, finally open I2S_CTRL.TXE or I2S_CTRL.RXE to start the slave action.

When I2S starts the slave action in Philips, MSB/LSB mode, it needs to ensure that the WS signal is in the right channel level state at the time of startup. When the I2S starts the slave action in PCM mode, it needs to ensure that the WS signal is low during startup. level status.

When the I2S is used as a slave to receive data, the data received each time will be read after the next frame of data is received, so when the communication is suspended or terminated, the last frame of data received by the I2S will be received next time. It can only be read when the communication starts.

When I2S is used as a slave in Philips, MSB/LSB mode to receive data, it will check whether WS is the left channel level when each frame of left channel data is received. When I2S is used as a slave to receive data in PCM mode, it will check whether WS has produced a valid level according to the standard communication protocol before each frame of data is received.

33.5 Register description

I2S1 base address: 0x4001E000

I2S2 base address: 0x4001E400

I2S3 base address: 0x40022000

I2S4 base address: 0x40022400

Table 33-5 List of I2S registers

Register name	Symbol	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_ADDR3 : 0x40054000

Register name	Symbol	Offset address	Bit width	Reset value
CMU I2S Clock Configuration Register	CMU_I2SCKSEL	0x12	16	0xbbbb

Note: When the I2S master mode clock source is selected I2SPLL, use this register to configure the clock source, which can be configured as PLLAR/PLLAQ/PLLAP/PLLHR/PLLHQ/PLLHP.

33.5.1 I2S Control Register (I2S_CTRL)

I2S Control Register

offset address: 0x000

Reset value: 0x00002200

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	SRST	CLKSEL	DUPLEX	CKOE	LRCKOE	SDOE	I2SPPLLSEL	CODECRC	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	

Bit	Marking	Place name	Function	Read and write
b31~b25	Reserved	-	Read as "0", write as "0"	R/W
b24	SRST	Software reset	0: Decommission 1: Software Reset	R/W
b23	CLKSEL	Timer selection	0: select I2SPPLL 1: select external clock	R/W
b22	DUPLEX	Communication method selection	0: half duplex 1: full duplex	R/W
b21	CKOE	Communication clock output permission	0: Output disable 1: Export enable	R/W
b20	LRCKOE	Channel clock output permission	0: Output disable 1: Export enable	R/W
b19	SDOE	data export license	0: Output disable 1: Export enable	R/W
b18	I2SPPLLSEL	I2SPPLL input selection	0: Input disabled 1: Enter permission	R/W
b17	CODECRC	Codec reset control	0: Decommission 1: Software Reset	R/W
b16	FIFOR	fifo reset	0: Decommission 1: Software Reset	R/W
b15	Reserved	-	Read as "0", write as "0"	R/W
b14~b12	RXBIRQWL[2:0]	Receive buffer interrupt request level	The interrupt request is triggered when the free space is less than the set value Note: Can only be set to 0/1/2/3/4, because the fifo space is 4	R/W
b11	Reserved	-	Read as "0", write as "0"	R/W
b10~b8	TXBIRQWL[2:0]	Transmit buffer interrupt request level	The interrupt request is triggered when the free space is more than the set value Note: Can only be set to 0/1/2/3/4, because the fifo space is 4	R/W
B7	MCKOE	Drive clock output enable	0: Disable drive clock output 1: Enable to drive the clock output Note: This bit is only used in I2S host mode	R/W
b6	ODD	prescaler odd factor	0: Actual frequency division value=I2SDIV*2 1: Actual frequency division value=I2SDIV*2+1 Note: This bit is only used in I2S host mode. If you want to set ODD to 1, you should first set the I2S_PR register, and then set the I2S_CTRL register.	R/W
b5	WMS	I2S working mode selection	0: I2S master mode 1: I2S slave mode	R/W
b4	EIE	Communication Error Interrupt Enable	0: Communication error interrupt is invalid 1: Communication error interrupt 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: Disable reception 1: Allow reception	R/W
b1	TXIE	Transmit interrupt enable	0: Send interrupt is invalid 1: Send interrupt is valid	R/W
b0	TXE	send enable	0: Disable sending 1: Allow sending	R/W

33.5.2 I2S Status Register (I2S_SR)

I2S Status Register

offset address: 0x004

Reset value: 0x00000014

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	BO		
-	-	-	-	-	-	-	-	-	-	-	-	RXBF	RXBE	TXBF	TXBE	RXBA	TXBA
<hr/>																	
Bit	Marking	Place name				Function								Read and write			
b31~b6	Reserved	-				Read as "0", write as "0"								R/W			
b5	RXBF	receive buffer full				0: The receive buffer is not full 1: The receive buffer is full								R			
b4	RXBE	Receive buffer empty				0: The receive buffer is not empty 1: The receive buffer is empty								R			
b3	TXBF	send buffer full				0: The send buffer is not full 1: Send buffer is full								R			
b2	TXBE	send buffer empty				0: Send buffer is not empty 1: The send buffer is empty								R			
b1	RXBA	Receive buffer alarm (related to water level)				0: The receive buffer is not alarmed 1: Receive buffer alarm								R			
b0	TXBA	Send buffer alarm (related to water level)				0: Send buffer is not alarmed 1: Send buffer alarm								R			

33.5.3 I2S Error Status Register (I2S_ER)

I2S Error Status Register

offset address: 0x008

Reset value: 0x00000000

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	BO		
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RXERR	TXERR
<hr/>																	
Bit	Marking	Place name				Function								Read and write			
b31~b2	Reserved	-				Read as "0", write as "0"								R/W			
b1	RXERR	receive error				0: Do not clear the flag bit 1: Clear the flag bit								R/W			
b0	TXERR	Send Error				0: Do not clear the flag bit 1: Clear the 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 when a transmit overflow/underflow occurs, and clears the flag by writing a 1 to the RXERR bit when a receive overflow occurs.

33.5.4 I2S Configuration Register (I2S_CFGR)

I2S Configuration Register

offset address: 0x00C

Reset value: 0x00000000

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]				
Bit	Marking	Place name	Function										Read and write				
b31~b6	Reserved	-	Read as "0", write as "0"										R/W				
b5	PCMSYNC	PCM frame sync	PCM frame sync 0: short frame sync 1: Long frame sync Note: This bit has meaning only when 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	Transmission 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 justified) 11: PCM protocol										R/W				

33.5.5 I2S transmit buffer FIFO data register (I2S_TXBUF)

I2S Transmit Buffer FIFO Data Register

offset address: 0x010

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TXBUF[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
TXBUF[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b0	TXBUF[31:0]	Send data	Store send data	W											

Note:

- In 16-bit frame, TXBUF[15:0] stores one frame of left channel or one frame of right channel transmit data.
- In 32-bit frame, TXBUF[31:0] stores one frame of left channel or one frame of right channel transmission data.

33.5.6 I2S receive buffer FIFO data register (I2S_RXBUF)

I2S Receive Buffer FIFO Data Register

offset address: 0x014

Reset value: 0x00000000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
RXBUF[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
RXBUF[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b0	RXBUF[31:0]	Receiving data	Store received data	R											

Note:

- In 16-bit frame, RXBUF[15:0] stores one frame of left channel or one frame of right channel receive data.
- In 32-bit frame, RXBUF[31:0] stores one frame of left channel or one frame of right channel receive data.

33.5.7 I2S divider register (I2S_PR)

I2S Prescaler Register

offset address: 0x018

Reset value: 0x00000002

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]]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b8	Reserved	-	Read as "0", write as "0"										R/W		
<hr/>															
b7~b0	I2SDIV[7:0]	Frequency division factor	I2SDIV[7:0]=0 or I2SDIV[7:0]=1 is disabled value See 4.4 Clock Generator Module Actual frequency division value=I2SDIV*2+I2S_CTRL.ODD 00000010: 2 frequency division 00000011: 3 frequency division 00000100: 4 frequency division 11111101: 253 frequency division 11111110: 254 frequency division 11111111: 255 frequency division Note: This bit is only used in I2S host mode										R/W		

33.5.8 CMU I2S Clock Configuration Register (CMU_I2SCKSEL)

CMU_BASE_ADDR3: 0x40054000

offset address: 0x12

Reset value: 0xbbbb

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0				
I2S4CKSEL				I2S3CKSEL				I2S2CKSEL				I2S1CKSEL							
<hr/>																			
Bit	Marking	Place name	Function	Read and write															
b15~b12	I2S4CKSEL	I2S clock source selection	0000: PCLK1 set by CMU_SCFGR 1000: PLLHQ 1001: PLLHR 1010: PLLAP 1011: PLLAQ 1100: PLLAR In addition, the setting is prohibited. Note: No clock for settings other than the above	R/W															
b11~b8	I2S3CKSEL	I2S clock source selection	0000: PCLK1 set by CMU_SCFGR 1000: PLLHQ 1001: PLLHR 1010: PLLAP 1011: PLLAQ 1100: PLLAR In addition, the setting is prohibited. Note: No clock for settings other than the above	R/W															
b7~b4	I2S2CKSEL	I2S clock source selection	0000: PCLK1 set by CMU_SCFGR 1000: PLLHQ 1001: PLLHR 1010: PLLAP 1011: PLLAQ 1100: PLLAR In addition, the setting is prohibited. Note: No clock for settings other than the above	R/W															
b3~b0	I2S1CKSEL	I2S clock source selection	0000: PCLK1 set by CMU_SCFGR 1000: PLLHQ 1001: PLLHR 1010: PLLAP 1011: PLLAQ 1100: PLLAR In addition, the setting is prohibited. Note: No clock for settings other than the above	R/W															

Note:

- When the target clock source for switching is PLLH/PLLA, ensure that the PLLH/PLLA oscillates in a stable state.
- When the target clock source selects PLLH/PLLA, please refer to the manual for the detailed method of configuring PLLH/PLLA [Clock controller (CMU)].

34 USB2.0 High Speed Module (USBHS)

34.1 Introduction to USBHS

The USB High Speed (USBHS) controller provides a USB communication solution for portable devices. The USBHS controller supports host mode and device mode. The chip integrates full-speed PHY and supports ULPI interface external high-speed PHY. In host mode, the USBHS controller supports high-speed (HS, 480Mb/s), full-speed (FS, 12Mb/s) and low-speed (LS, 1.5Mb/s) transceivers, while in device mode only high-speed (HS, 480Mb/s) and full-speed (FS, 12Mb/s) transceivers. The USBHS controller supports all four transfer modes (control transfer, bulk transfer, interrupt transfer and isochronous transfer) defined by the USB 2.0 protocol. The USBHS controller supports LPM (Link Power Management) function.

The integrated full-speed PHY inside the chip only supports full-speed (FS, 12Mb/s) and low-speed (LS, 1.5Mb/s) transmission, and an external high-speed PHY through the ULPI interface supports high-speed (HS, 480Mb/s), full-speed (FS, 12Mb/s) and low speed (LS, 1.5Mb/s) transmission.

Follow the protocol as follows:

- Universal Serial Bus Revision 2.0 Specification
- USB 2.0 Link Power Management Addendum Engineering Change Notice to the USB 2.0 specification, July 16, 2007
- Errata for USB 2.0 ECN: Link Power Management (LPM) - 7/2007

34.2 Main features of USBHS

Mainly divided into three categories: Common attributes, host mode attributes, and device mode attributes.

34.2.1 Common features

- Supports two PHY interfaces
 - Built-in on-chip USB2.0 full-speed PHY (need to set USB_SYCTLREG. USBHS_FSPHYE=1 enable)
 - ULPI interface to off-chip high-speed PHY
- Support host mode and device mode
- Supports HS/FS SOF as well as low-speed "Keep-alive" tokens and has the following features:
 - SOF pulse pin output function
 - SOF pulse can be used as the internal event source of the chip to trigger TIMER, DMA and other modules to work
 - Configurable frame period
 - Configurable end of frame interrupt
- The module has built-in DMA, and the AHB burst transfer type can be configured by software

- Power saving features such as USB suspend, stop RAM clock, stop PHY domain clock
- Features 8KB dedicated RAM with advanced FIFO control
 - The RAM space can be divided into different FIFOs for flexible and efficient use of RAM
 - Each FIFO can store multiple packets
 - Dynamically allocate memory
 - The size of the FIFO can be configured to a non-power-of-2 value for continuous use of memory cells
- No application intervention can be required within a frame to achieve maximum USB bandwidth
- The host mode or device mode can be automatically determined according to the level of the ID line

34.2.2 Host Mode Features

- Host mode supports USB2.0 high-speed (HS, 480Mb/s), full-speed (FS, 12Mb/s) and low-speed (LS, 1.5Mb/s) transfers
- The VBUS voltage needs to be generated by an external power chip
- Up to 16 host channels (pipes): Each channel can be dynamically reconfigured to support any type of USB transfer
- Built-in hardware scheduler to:
 - Stores up to 8 interrupts plus isochronous transfer requests in a periodic hardware queue
 - Stores up to 8 control plus bulk transfer requests in aperiodic hardware queues
- Manages one shared RX FIFO, one periodic TX FIFO, and one aperiodic TX FIFO for efficient use of USB data RAM

34.2.3 Device Mode Features

- The slave mode supports USB2.0 high-speed (HS, 480Mb/s) and full-speed (FS, 12Mb/s) transfers.
- 1 bidirectional control endpoint 0
- 15 OUT endpoints that can be configured to support bulk, interrupt, or isochronous transfers
- 15 IN endpoints that can be configured to support bulk, interrupt, or isochronous transfers
- Contains 16 transmit FIFOs (one for each 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.

34.3 USBHS System Block Diagram

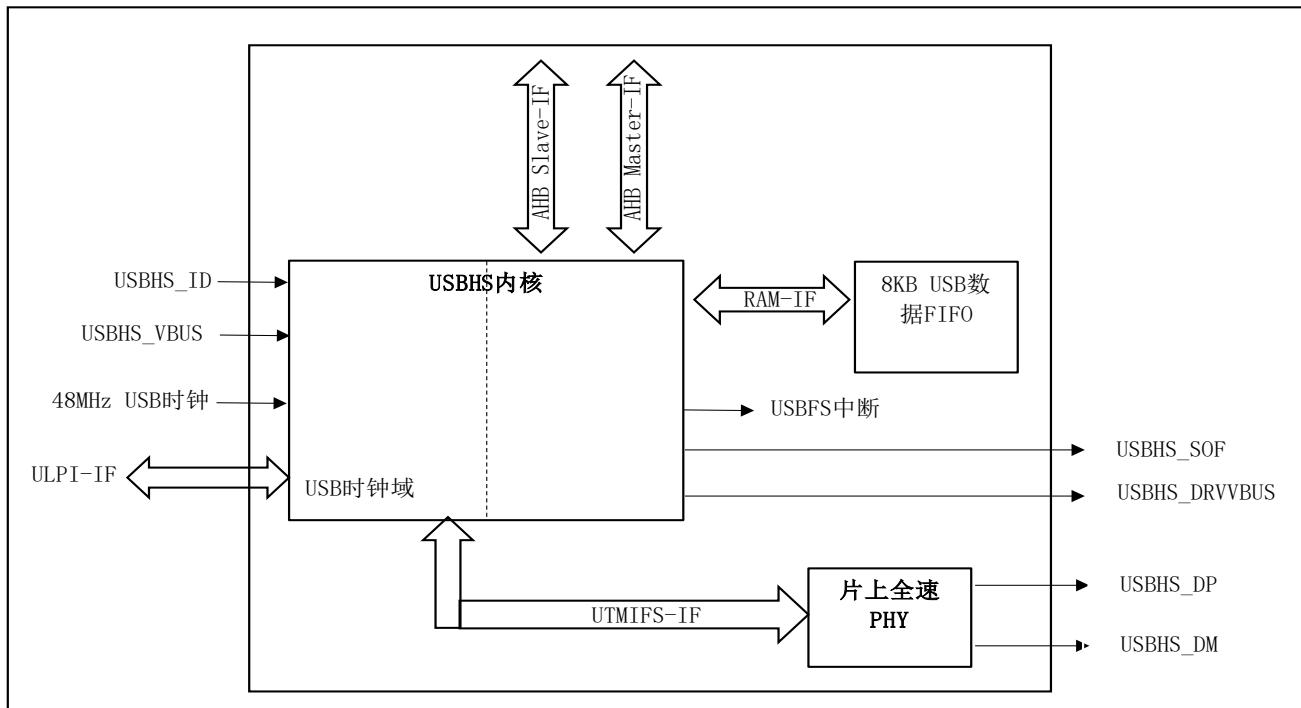


Figure 34-1 USBHS System Block Diagram

34.4 USBHS pin description

Table 34-1 USBHS pin description

Pin name	Direction	Applicable mode	corresponding pin	Functional description	Applicable PHY
USBHS_VBUS	Input	equipment	PA9	Power port, 5V withstand voltage	On-chip full-speed PHY
USBHS_DP	Input/Output	Host, device	PB15	Differential data D+	On-chip full-speed PHY
USBHS_DM	Input/Output	Host, device	PB14	Differential data D-	On-chip full-speed PHY
USBHS_DRVVBUS	Output	Host	PC9, PB8	External power chip enable	On-chip full-speed PHY
USBHS_ID	Input	Host	PA10	USB AB device identification	On-chip full-speed PHY
USBHS_SOF	Output	Host, device	PA8	SOF output pulse	On-chip full-speed PHY ULPI PHY
USBHS_ULPI_CLK	Input	Host, device	PE12, PA5	ULPI Clock	ULPI PHY
USBHS_ULPI_DIR	Input	Host, device	PC2, PI11	ULPI Direction	ULPI PHY
USBHS_ULPI_NXT	Input	Host, device	PC3, PH4	ULPI Next	ULPI PHY
USBHS_ULPI_STP	Output	Host, device	PC0	ULPI Stop	ULPI PHY
USBHS_ULPI_D0	Input/Output	Host, device	PE13, PA3	ULPI Data 0	ULPI PHY
USBHS_ULPI_D1	Input/Output	Host, device	PE14, PB0	ULPI Data 1	ULPI PHY
USBHS_ULPI_D2	Input/Output	Host, device	PE15, PB1	ULPI Data 2	ULPI PHY
USBHS_ULPI_D3	Input/Output	Host, device	PB10	ULPI Data 3	ULPI PHY
USBHS_ULPI_D4	Input/Output	Host, device	PB11, PD9	ULPI Data 4	ULPI PHY
USBHS_ULPI_D5	Input/Output	Host, device	PB12	ULPI Data 5	ULPI PHY
USBHS_ULPI_D6	Input/Output	Host, device	PB13	ULPI Data 6	ULPI PHY
USBHS_ULPI_D7	Input/Output	Host, device	PB5, PE11	ULPI Data 7	ULPI PHY

Since the USBHS_DP and USBHS_DM pins are multiplexed with general-purpose GPIOs, when using the USBHS on-chip full-speed PHY, it is recommended to turn off the digital functions of the corresponding pins. The USBHS_DP and USBHS_DM functions are analog functions and have nothing to do with the corresponding PFSR register settings. For details, please refer to 011. General IO (GPIO) chapter. In addition, when the USBHS function is not used, additional current consumption will be generated when the digital function pins corresponding to the USBHS_DP and USBHS_DM pins are flipped.

34.5 USBHS function description

34.5.1 USBHS clock and working mode

There are the following three clocks used by USBHS

- The 48MHz clock is generated by the internal PLL circuit. The PLL clock source needs to select an external high-speed oscillator. Before using the USBHS module, the USBHS clock needs to be configured in the CMU module.
- 60MHz clock, use the UPLI interface to connect the input clock of the high-speed PHY, only used when the PHY is connected.
- The module clock PCLK1 needs to be configured in the CMU module, and the clock needs to be greater than or equal to 60MHz.

The USBHS can be used as a host or a device, and includes an on-chip full-speed PHY and an ULPI interface for an external high-speed PHY.

Pull-up and pull-down resistors have been integrated inside the on-chip full-speed PHY, and USBHS can be automatically selected based on the current mode and connection status.

When USBHS uses the on-chip full-speed PHY, the VCC voltage range is 3.0~3.6V.

34.5.2 USBHS mode decision

There are two ways for USBHS to determine the current working mode:

Method 1: Automatically identify according to the state of the USBHS_ID line, when it is detected that the USBHS_ID line is at a high level, the module works in the device mode, and when it is detected that the USBHS_ID line is at a low level, the module works in the host state.

Method 2: Force the host/device mode, by setting the FDMOD or FHMOD bit of the register USBHS_GUSBCFG to 1, so that the module ignores the level of the USBHS_ID line and is forced to work in the device or host mode.

34.5.3 USBHS host function

34.5.3.1 Mainframe function introduction

When USBHS works in host mode, VBUS is the 5V power pin specified by the USB protocol. The internal PHY does not support 5V power supply, so an external USB power chip is required to power the device. USBHS_DRVVBUS is used to enable the external USB power chip, and the overcurrent detection of the external power chip can be realized through the external interrupt IRQ of this MCU. USBHS_VBUS can be used as GPIO in host mode.

A typical USB host mode system construction diagram is as follows:

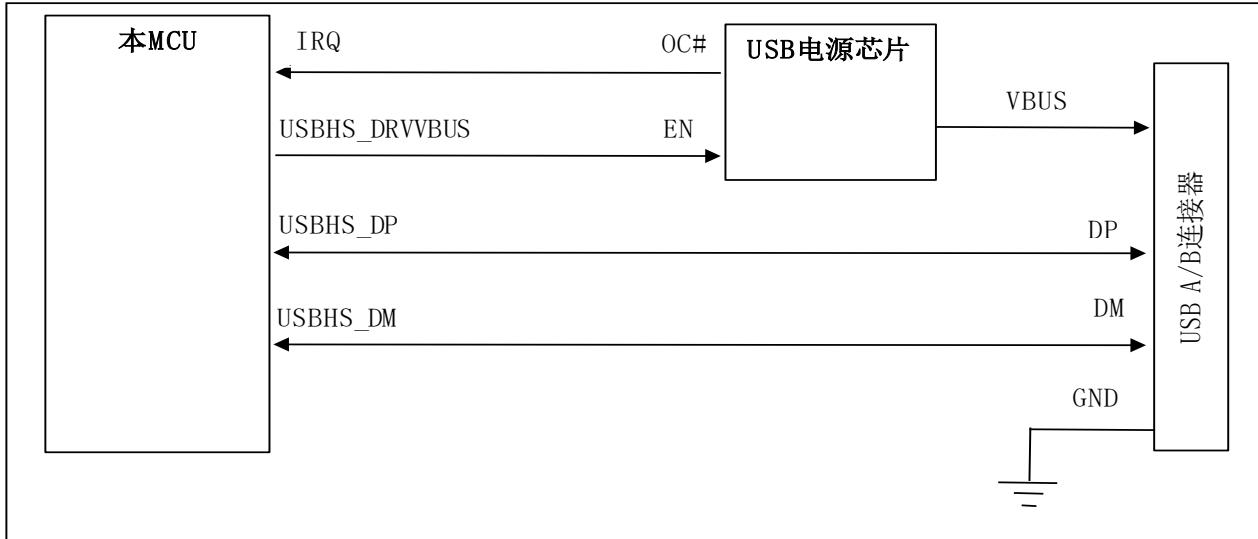


Figure 34-2 USBHS host mode system construction diagram

34.5.3.2 Host port power supply

This MCU cannot output 5V to provide VBUS. To do this, a USB power chip or basic power switch (such as a 5V supply from the application board) must be added outside the microcontroller to drive the 5V VBUS line. The external power chip can be driven by any GPIO output or USBHS_DRVVBUS. When the application determines to use the GPIO to control the external device to provide VBUS, it must still set the port power bit (PWPR bit in USBHS_HPRT) in the host port control and status register.

34.5.3.3 Host detects device connection and disconnection

USB devices will be detected as soon as they are connected. The USBHS module will signal a host port interrupt, which is triggered by the device connection bit in the host port control and status register (the PCDET bit in USBHS_HPRT).

A device disconnect event will trigger a disconnect detection interrupt (DISCINT bit in USBHS_GINTSTS).

34.5.3.4 Host enumeration

After a device connection is detected, if a new device is connected, the host must initiate the enumeration process by sending USB reset and configuration commands to the new device.

The application drives the USB reset signal (single-ended zero) through the USB by setting the port reset bit in the host port control and status register (PRST bit in USBHS_HPRT) for a minimum of 10ms and a maximum of 20ms. The application calculates the duration of this process and then clears the port reset bit.

Immediately after the USB reset sequence is complete, the port enable/disable change bit (PENCHNG bit in USBHS_HPRT) triggers a host port interrupt, which in turn notifies the application

that it can be accessed from the port speed field in the host port control and status register (in USBHS_HPRT PSPD) read the enumerated device speed, and the host has started driving SOF (HS/FS) or Keep-alive tokens (LS). At this point the host is ready to enumerate the device by sending commands to the device.

34.5.3.5 Host hangs

The application suspends the USB bus by setting the port suspend bit (PSUSP in USBHS_HPRT) in the host port control and status register. The USBHS module stops sending SOF and enters the suspend state.

The bus can be brought out of the suspend state by autonomous activity of the remote device (remote wakeup). In this case, the remote wakeup signal will trigger the remote wakeup interrupt (WKUPINT bit in USBHS_GINTSTS), the hardware will reset the port recovery bit in the host port control and status register (PRES bit in USBHS_HPRT) by itself, and automatically drive recovery through USB Signal. The application must time the resume window, then clear the port resume bit to exit the suspend state and restart the SOF.

If exit suspend is initiated by the host, the application must set the port resume bit to initiate the resume signal on the host port, time the resume window and eventually clear the port resume bit.

34.5.3.6 Host channel

The USBHS module implements 16 host channels. Each host channel can be used for USB host transfers (USB pipes). The host can handle up to 8 transfer requests at the same time. If the application has more than 8 pending transfer requests, after the channel is released from the previous task (that is, after receiving the transfer complete and channel stop interrupt), the host controller driver (HCD) must restart the Allocate channels.

Each host channel can be configured to support input/output and periodic/aperiodic transactions. Each host channel uses dedicated control (HCCHARx) registers, transfer configuration (HCTSIZx) registers/interrupt (HCINTx) registers, and their associated interrupt mask registers (HCINTMSKx).

Host channel control

The application program has the following control over the host channel through the host channel x characteristics register (HCCHARx):

- Channel enable/disable
- Set the speed of the target USB device: HS/FS/LS
- Set the address of the target USB device
- Sets 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 transfer on this channel: Control/Batch/Interrupt/Sync

- Sets the maximum packet length of device endpoints communicating with this channel
- Set the frame to be transmitted periodically: Odd frame/even frame

Host channel transmission

The host channel transfer size registers (HCTSIZx) allow the application to program the transfer size parameter and read the transfer status. This register must be set before the channel enable bit in the host channel characteristics register is set. After the endpoint is enabled, the packet count field becomes read-only immediately, and the USBHS module updates the field according to the current transfer status.

The following transfer parameters can be programmed:

- transfer size in bytes
- The number of packets that make up the entire transfer size
- Initial data PID

Host channel status/interrupt

The Host Channel x Interrupt Register (HCINTx) indicates the state of the endpoint when USB and AHB related events occur. When the host channel interrupt bit in the interrupt register (HCINT bit in USBHS_GINTSTS) is set, the application must read these registers for detailed information. Before reading these registers, the application must 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 appropriate bits in this register to clear the corresponding bits in the HAINT and GINTSTS registers. The USBHS_HCINTMSK x registers also provide mask bits for each interrupt source per channel.

The host module provides the following status checking and interrupt generation functions:

- Transfer complete interrupt, indicating that both the application (AHB) and the USB side have completed data transfer
- Channel stopped due to transfer complete, USB transaction error, or application issued a disable command
- The associated transmit FIFO is half empty or completely empty (IN endpoint)
- ACK response received
- NAK response received
- STALL response received
- USB transaction errors due to CRC check failures, timeouts, bit stuffing errors, and bad EOPs
- crosstalk error
- frame overflow
- Error in toggle bit for data synchronization

34.5.3.7 Host scheduler

The host module has a built-in hardware scheduler that can autonomously reorder and manage USB transaction requests issued by the application. At the beginning of each frame, the host performs periodic (sync and interrupt) transactions followed by aperiodic (control and bulk) transactions to comply with the USB specification's high priority guarantee for isochronous and interrupt transfers.

The host processes USB transactions through request queues (one periodic request queue and one aperiodic request queue). Each request queue can store up to 8 entries. Each entry represents a USB transaction request initiated by an application but has not yet received a response, and stores the number of the IN or OUT channel used to execute the USB transaction, as well as other relevant information. The order in which USB transaction requests are written in the queue determines the order in which transactions are executed on the USB interface.

At the beginning of each frame, the host processes the periodic request queue first, and then processes the aperiodic request queue. If a synchronous or interrupt-type USB transfer transaction request scheduled to be performed on the current frame is still pending at the end of the current frame, the host will issue an outstanding periodic transfer interrupt (IPXFR bit in USBHS_GINTSTS). The USBHS module is responsible for the management of periodic and aperiodic request queues. The Periodic Transmit FIFO and Queue Status Register (HPTXSTS) and the Aperiodic Transmit FIFO and Queue Status Register (HNPTXSTS) are read-only registers that the application can use to read the status of each request queue. These include:

- Number of free entries currently available in the periodic (aperiodic) request queue (up to 8)
- Free space currently available in periodic (aperiodic) TxFIFO (OUT transactions)
- IN/OUT tokens, host channel numbers, and other status information

Since each request queue can store up to 8 USB transaction requests, the application can send the host USB transaction request to the scheduler in advance; Appears on the USB bus after an acyclic transaction is complete.

To issue a transaction request to the host scheduler (queue), the application must read the PTXQSAV bit in the USBHS_HPTXSTS register or the NPTQXSAV bit in the USBHS_HNPTXSTS register, ensuring that there is at least one free space in the queue for periodic (aperiodic) requests to store the current ask.

34.5.4 USBHS Device Features

34.5.4.1 Device function introduction

When USBHS works in device mode, VBUS is the 5V power supply pin specified by the USB protocol and is a 5V withstand voltage pin. This module always detects the level state of the VBUS line to connect or disconnect the device.

A typical USB device mode system construction diagram is as follows:

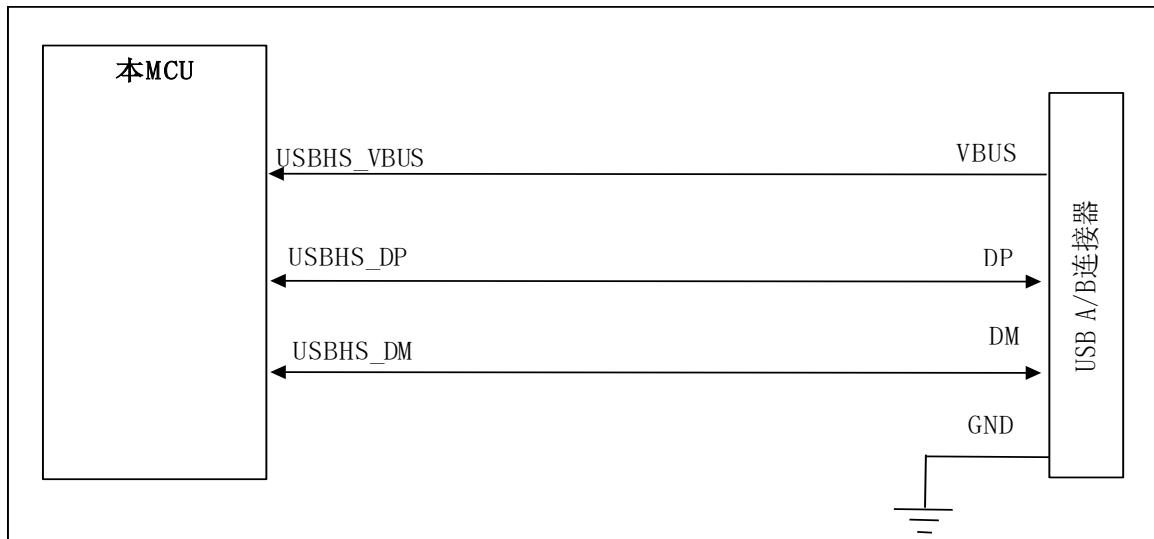


Figure 34-3 USBHS device mode system construction diagram

34.5.4.2 Device power status

When the module detects that USBHS_VBUS is high, it will make the USB device enter the power supply state. Then, USBHS automatically connects the DP pull-up resistor, sends a signal that the full-speed device is connected to the host and generates a session request interrupt (VBUSVINT bit in USBHS_GINTSTS), indicating that it enters the power supply state.

Additionally, the USBHS_VBUS input ensures that the host provides a valid VBUS level during USB operation. USBHS will automatically disconnect if it detects that the VBUS level is low (for example, due to power disturbance or host port shutdown).

In the powered state, the USBHS expects to receive a reset signal from the host. Other USB operations cannot be performed. A reset detected interrupt (USBRST in USBHS_GINTST) is generated as soon as the reset signal is received. After the reset signal ends, an enumeration complete interrupt (ENUMDNE bit in USBHS_GINTSTS) is generated, and USBHS enters the default state.

34.5.4.3 Device default state

By default, USBHS expects a SET_ADDRESS command from the host. Other USB operations cannot be performed. When a valid SET_ADDRESS command is decoded on the USB, the application will write the corresponding address value to the device address field in the device configuration register (DAD bit in USBHS_DCFG). USBHS then enters the address state and is ready to respond to host transactions with the configured USB address.

34.5.4.4 Device suspend state

USBHS devices continuously monitor USB activity. After the USB idle time reaches 3ms, the early suspend interrupt (bit ESUSP in USBHS_GINTSTS) will be issued, and 3ms later by the suspend interrupt (bit USBSUSP in USBHS_GINTSTS) to confirm that the device enters the suspended state. Then, the device suspend bit in the device status register (SUSPSTS bit in USBHS_DSTS) is automatically set to 1, and USBHS enters the suspend state immediately.

The suspend state can be exited through the device itself. In this case, the application will set the Remote Wakeup Signal bit in the Device Control Register (RWUSIG bit in USBHS_DCTL) and clear it within 1ms to 15ms.

But if the device detects the resume signal sent by the host, it will generate a resume interrupt (WKUPINT bit in USBHS_GINTSTS), and the device suspend bit is automatically cleared.

34.5.4.5 Device soft disconnect

The power state can be exited by software with the help of the soft disconnect function. The DP pull-up resistor can be removed by setting the soft disconnect bit in the device control register (SDIS bit in USBHS_DCTL), at which point a device disconnect occurs on the host side despite not actually unplugging the USB cable from the host port Detect interruption.

34.5.4.6 Device endpoint

Endpoint class

The USBHS module implements the following USB endpoints:

- Control endpoint 0:
 - Bidirectional and only process control messages
 - Use a separate set of registers to handle IN and OUT transactions
 - Dedicated control (USBHS_DIEPCTL0/USBHS_DOEPCTL0) registers, transfer configuration (USBHS_DIEPTSIZ0/USBHS_DIEPTSIZ0) registers, and status interrupt (USBHS_DIEPINTx/USBHS_DOEPINT0) registers. The set of bits available in the control and transfer size registers is slightly different than in other endpoints
- 15 IN endpoints
 - Each endpoint can be configured to support isochronous, bulk or interrupt transfer types

- Each endpoint has dedicated control (USBHS_DIEPCTLx) registers, transfer configuration (USBHS_DIEPTSIZx) registers, and status interrupt (USBHS_DIEPINTx) registers
- The device IN endpoint general interrupt mask register (USBHS_DIEPMSK) can be used to enable/disable the same type of endpoint interrupt sources on all IN endpoints (including EP0)
- Supports outstanding isochronous IN transfer interrupt (ISOIXFR bit in USBHS_GINTSTS), which will trigger when there is an outstanding transfer on at least one isochronous IN endpoint in the current frame. This interrupt is triggered together with the periodic frame interrupt (USBHS_GINTSTS/EOPF)
- 15 OUT endpoints
 - Each endpoint can be configured to support isochronous, bulk or interrupt transfer types
 - Each endpoint has dedicated control (USBHS_DOEPCTLx) registers, transfer configuration (USBHS_DOEPTSIZx) registers, and status interrupt (USBHS_DOEPINTx) registers
 - The device OUT endpoint general interrupt mask register (USBHS_DOEPMASK) can be used to enable/disable the same type of endpoint interrupt sources on all OUT endpoints (including EP0)
 - Supports outstanding isochronous OUT transfer interrupt (INCOMPISOOUT bit in USBHS_GINTSTS), which will trigger when there is an outstanding transfer on at least one isochronous OUT endpoint in the current frame. This interrupt is triggered together with the periodic frame interrupt (USBHS_GINTSTS/EOPF)

endpoint control

The application can take the following control of the endpoint through the device endpoint x IN/OUT control registers (DIEPCTLx/DOEPCTLx):

- Endpoint enable/disable
- Activate endpoint with current configuration
- Set USB transfer type (Sync, Bulk and Interrupt)
- Set the supported packet size
- Set the Tx-FIFO number associated with the IN endpoint
- Set the data0/data1 PID you want to receive or use when sending (only for bulk/interrupt transfers)
- Sets the odd/even frame to which a transaction is received or sent (isochronous transfers only)
- The NAK bit can be set so that no matter what the state of the FIFO is at this time, NAK is replied to the host's request
- The STALL bit can be set so that the host's token for that endpoint is STALL returned by hardware
- The OUT endpoint can be set to listen mode, i.e. no CRC check on the received data

endpoint transfer

The device endpoint x transfer size registers (DIEPTSIZx/DOEPTSIZx) allow applications to program transfer size parameters and read transfer status. This register must be set before the endpoint enable bit in the endpoint control register is set. After the endpoint is enabled, these fields become read-only immediately, and the USBHS module updates these fields according to the current transfer status.

The following transfer parameters can be programmed:

- transfer size in bytes
- The number of packets that make up the entire transmission

Endpoint Status/Status

The Device Endpoint x Interrupt Register (DIEPINTx/DOEPINTx) indicates the state of the endpoint upon USB and AHB related events. When the OUT endpoint interrupt bit or the IN endpoint interrupt bit in the module interrupt register (the OEPINT bit in USBHS_GINTSTS or the IEPINT bit in USBHS_GINTSTS, respectively) is set, the application must read these registers for detailed information. Before the application can read these registers, the device-wide endpoint interrupt (USBHS_DAINT) register must be read to obtain the endpoint number of the device endpoint x interrupt register. The application must clear the appropriate bits in this register to clear the corresponding bits in the DAINT and GINTSTS registers.

The module provides the following status checking and interrupt generation functions:

- Transfer complete interrupt, indicating that both the application AHB and the USB side have completed the data transfer
- Setup phase completed (only for OUT endpoints of control transfer type)
- The associated transmit FIFO is half empty or completely empty (IN endpoint)
- NAK reply sent to host (IN endpoint for isochronous transfers only)
- IN token received when TxFIFO is empty (only for IN endpoints of bulk and interrupt transfer types)
- OUT token received when endpoint is not yet enabled
- babble error detected
- Application shutdown endpoint takes effect
- The application sets NAK to the endpoint to take effect (only for IN endpoints of isochronous transmission type)
- Received more than 3 consecutive setup packets (only for control type OUT endpoints)
- Timeout condition detected (only for IN endpoints of control transfer type)

34.5.5 USBHS SOF pulse output function

USBHS can monitor, track and configure SOF frames in both host and device mode and also has SOF pulse output function. The SOF pulse is output through the USBHS_SOF pin, and the output width is 16 system clock cycles.

34.5.5.1 Host SOF

In host mode, the number of PHY clocks that occur during two consecutive SOF (HS/FS) or keep-alive (LS) tokens generated can be programmed in the Host Frame Interval Register (HFIR) so that the application can control the SOF frame period. Interrupts are generated at the start of a frame (SOF bit in USBHS_GINTSTS). The current frame number and the time remaining until the next SOF can be tracked by the application in the host frame number register (HFNUM).

Using the SOFEN bit in the USBHS system control register USBHS_SYCTLREG, the SOF pulse signal with a width of 16 system clock cycles that is generated when any SOF token is issued can be output from the USBHS_SOF pin.

In addition, the SOF pulse can also work as an internal event-triggered DMA transfer, TIMER count and other external modules.

34.5.5.2 Device SOF

In device mode, the start of frame interrupt (SOF bit in USBHS_GINTSTS) is triggered every time the USB receives a SOF token. The corresponding frame number can be read from the Device Status Register (FNSOF bit in USBHS_DSTS). Using the SOFEN bit in the USBHS system control register USBHS_SYCTLREG, it is also possible to generate a SOF pulse signal with a width of 16 system clock cycles and make the signal output on the USBHS_SOF pin for external availability.

In addition, the SOF pulse can also work as an internal event-triggered DMA transfer, TIMER count and other external modules.

Periodic end of frame interrupts (GINTSTS/EOPF) are 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 (USBHS_DCFG in the PFIVL bit). This function can be used to determine if all isochronous communications for the frame are complete.

34.5.6 USBHS power control

When the USBHS module is not used, the HCLK and PHY clocks of the USBHS module can be stopped through 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, the power reduction technique can be used in the USB suspend state.

- Stop PHY clock (STPPCLK bit in USBHS_GCCTL)

When the Stop PHY Clock bit in the Clock Gating Control register is set, most of the 48 MHz internal clock domains of the USBHS full-speed module are closed by clock gating. Even if the application still provides the clock input, it will save the dynamic power consumption of the module due to the inversion of the clock signal. It will also shut down most of the transceiver units. Only the part responsible for detecting asynchronous recovery events or remote wake-up events remains active. .

■ HCLK gate (GATEHCLK bit in USBHS_GCCTL)

When the GATEHCLK bit in the clock gating control register is set to 1, most of the system clock domains inside the USBHS module are turned off by clock gating. Only the register read and write interfaces remain active. Even if the application still provides the clock input, the dynamic power consumption of the module due to the inversion of the clock signal will be saved.

To save dynamic power, the USB data FIFO is only clocked when it is being accessed by the USBHS module.

34.5.7 USBHS dynamically updates the USBHS_HFIR register

In the host mode, the USB module has the function of dynamically fine-tuning the frame period, and can synchronize the external device with the SOF frame. If the USBHS_HFIR register is changed in the current SOF frame, the SOF cycle will be corrected accordingly in the next frame. For details, please refer to Figure 34-4 .

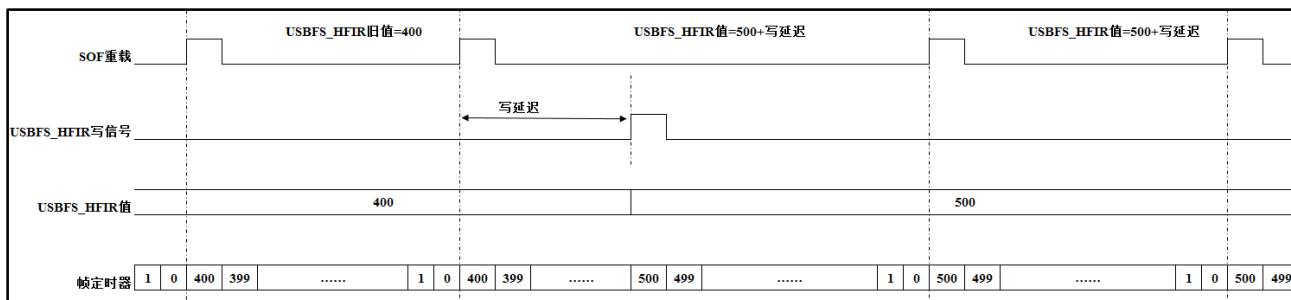


Figure 34-4 Schematic diagram of USBHS dynamically updating USBHS_HFIR register

34.5.8 USBHS data FIFO

The USBHS system has 8KB dedicated RAM and adopts an efficient FIFO control mechanism. The packet FIFO controller module in the USBHS module divides the RAM space into multiple TxFIFOs (where the application pushes data for short storage before USB transfers) and a single RxFIFO (before data received from USB is read by the application) , where it is temporarily stored).

The number and organization of FIFOs constructed 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 FIFO is configured by software to better meet the application requirements.

34.5.9 USBHS Host FIFO Architecture

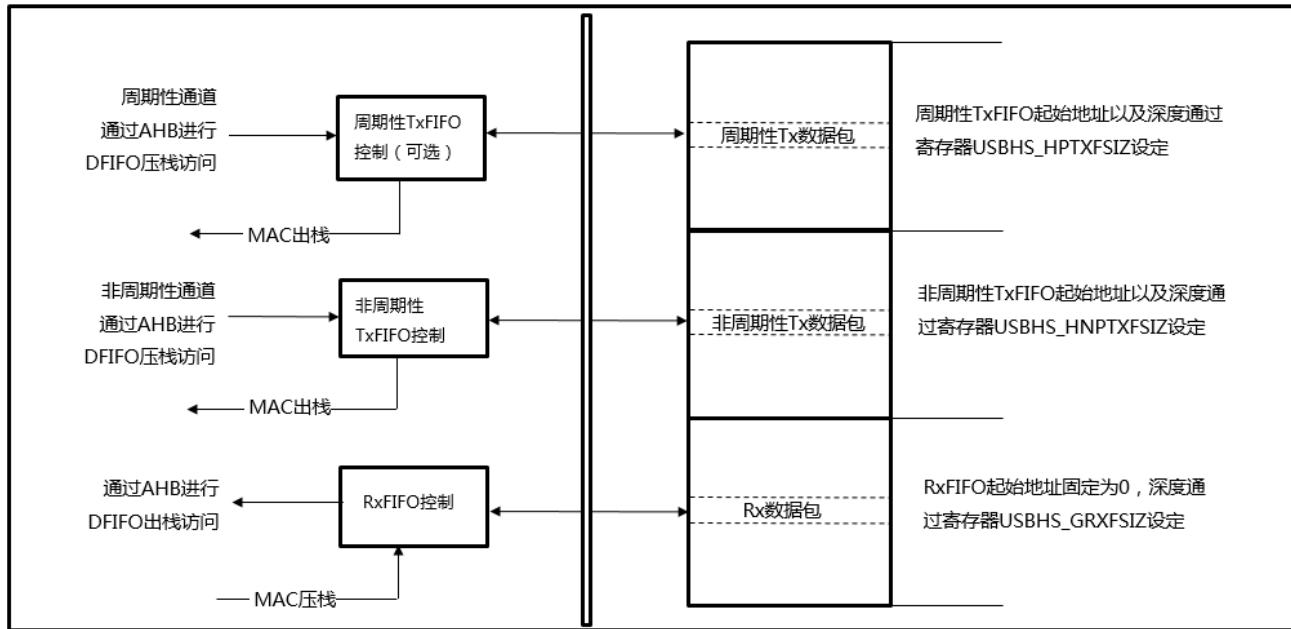


Figure 34-5 Schematic diagram of FIFO architecture in USBHS host mode

34.5.9.1 Host RxFIFO

The host uses one receive FIFO to handle all periodic and aperiodic 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 the system memory. As long as there is room in the FIFO, packets from the IN endpoint of the device are received and stored one by one. The status of each packet received (including host destination channel, number of bytes, data PID, and checksum of 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 enables 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 host software, the USBHS module can fill the receive FIFO to the limit

The application will receive the Rx FIFO not empty interrupt 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 pop registers, and finally reads the data from the RxFIFO.

34.5.9.2 Host TxFIFO

The host uses one transmit FIFO for all aperiodic (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 the USB (transmit packets). The size of the Periodic (Aperiodic) TxFIFO is configured in the Host Periodic (Aperiodic) Transmit FIFO Size (HPTXFSIZ/HNPTXFSIZ) register.

The two Tx FIFOs operate according to the priority, and the priority of the periodic communication is higher, so the periodic communication is performed first within the time of one USB frame. At the beginning of a frame, the built-in host scheduler processes the periodic request queue first, and then processes the aperiodic request queue.

The architecture of the two transmit FIFOs enables the USB host to optimally manage the periodic and aperiodic transmit data buffers separately:

- All host channels configured to support periodic (aperiodic) OUT transactions share the same RAM buffer (shared FIFO)
- For any sequence of OUT tokens driven by host software, the USBHS module can fill the periodic (aperiodic) transmit FIFO to the limit

The USBHS module issues a periodic TxFIFO empty interrupt (PTXFE bit in USBHS_GINTSTS) whenever the periodic TxFIFO is half empty or completely empty, depending on the periodic TxFIFO empty level bit in the AHB configuration register (PTXFELVL bit in USBHS_GAHBCFG) value. As long as there is free space in both the periodic TxFIFO and the periodic request queue, the application can write ahead of the transmit data. The available space for both can be known by reading the Host Periodic Transmit FIFO and Queue Status Register (HPTXSTS).

The USBHS module issues an aperiodic TxFIFO empty interrupt (NPTXFE bit in USBHS_GINTSTS) whenever the aperiodic TxFIFO is half empty or completely empty, depending on the aperiodic TxFIFO empty level bit in the AHB configuration register (TXFELVL in USBHS_GAHBCFG bits). The application can write transmit data as long as there is free space in both the aperiodic TxFIFO and the aperiodic request queue. The available space for both can be known by reading the Host Aperiodic Transmit FIFO and Queue Status Register (HNPTXSTS).

34.5.10 USBHS Device FIFO Architecture

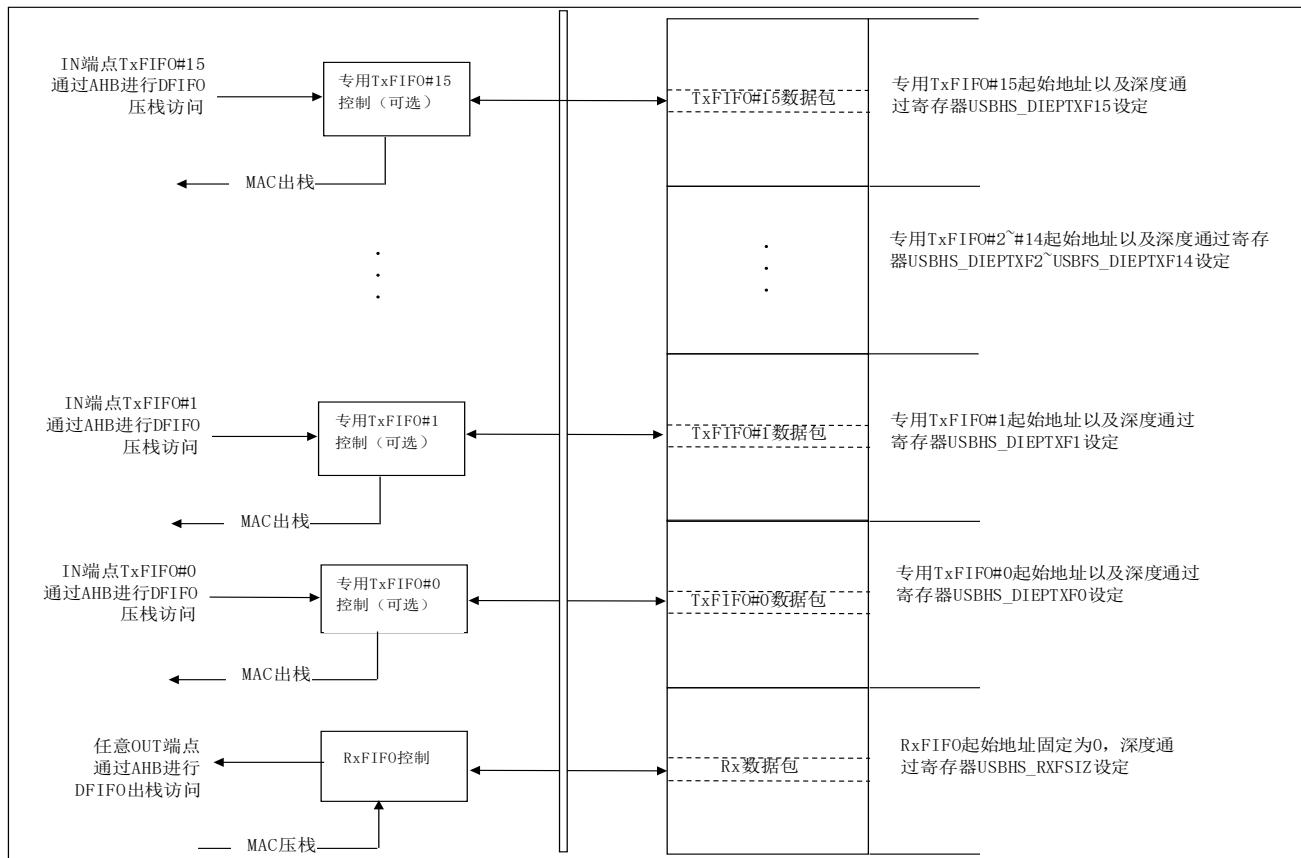


Figure 34-6 Schematic diagram of FIFO architecture in USBHS device mode

34.5.10.1 Device RxFIFO

USBHS devices use a single receive FIFO to receive data sent to all OUT endpoints. As long as there is free space in the Rx FIFO, the received packets are filled into the Rx FIFO one by one. In addition to valid data, the received packet status (including OUT endpoint destination number, byte count, data PID, and validation of received data) is also stored by the module. When no space is available, the device replies with a host transaction NAK acknowledgement and triggers an interrupt on the addressed endpoint. The size of the receive FIFO is configured in the receive FIFO size register (GRXFSIZ).

A single receive FIFO architecture allows USB devices to fill receive RAM buffers more efficiently:

- All OUT endpoints share the same RAM buffer (shared FIFO)
- For any host sequence OUT token, the USBHS module can fill the receive FIFO to the limit

The application will always receive the Rx FIFO not empty interrupt (RXFNE bit in USBHS_GINTSTS) as long as at least one packet is available for reading in the Rx FIFO. The application program reads the packet information from the receive status read and pop register (GRXSTSP), and finally reads the corresponding data from the receive FIFO by reading the pop address associated with the endpoint.

34.5.10.2 Device TxFIFO

The module provides dedicated FIFOs for each IN endpoint. The application configures the FIFO size for IN endpoint 0 through the aperiodic sending FIFO size register (USBHS_DIEPTSIZ0); configures the FIFO size for IN endpoint x through the device IN endpoint sending FIFOx register (DIEPTSIZx).

34.5.11 USBHS FIFO RAM allocation

34.5.11.1 Host mode

Receive FIFO RAM allocation

Status information is written to the FIFO with each received packet. Therefore, at least (maximum packet size / 4) + 2 space must be allocated for receive packets. If multiple isochronous channels are enabled, the space allocated for receiving consecutive packets must be at least twice (maximum packet size / 4) + 2. In general, the recommended space is twice (maximum packets / 4 + 2), so that while the previous packet is sent to the CPU, the USB can simultaneously receive subsequent packets.

The transfer completion status information is written to the FIFO along with the last packet received by the endpoint. So a place must be allocated for this.

When running in DMA mode, each host channel's DMA address register is stored in the FIFO, so a place in the FIFO needs to be reserved for each channel to store its address register.

Transmit FIFO RAM allocation

The minimum RAM required for the host aperiodic transmit FIFO is the size of the largest packet transmitted on all supported aperiodic 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 transmit FIFO with the next packet.

The minimum RAM required by the host periodic transmit FIFO is the size of the largest packet transmitted on all supported periodic OUT channels. If there is at least one isochronous OUT endpoint, the space must be at least twice the maximum packet size in that channel.

When running in DMA mode, each host channel's DMA address register will be stored in the FIFO, so it is necessary to reserve a location in the FIFO for each channel to store its address register.

34.5.11.2 Device mode

Receive FIFO RAM allocation

Applications should allocate RAM for SETUP packets: Thirteen locations must be reserved in the receive FIFO to receive SETUP packets on the control endpoint. The USBHS module will not write any other data to these locations reserved for SETUP packets. A location will be allocated for the global OUT NAK. Status information is written to the FIFO with each received packet. Therefore, at

least (maximum packet size / 4) + 1 space must be allocated for receive packets. If multiple isochronous endpoints are enabled, the space allocated for receiving consecutive packets must be at least twice (maximum packet size / 4) + 1. Typically, the recommended space is twice (maximum packets / 4 + 1) so that the USB can simultaneously receive subsequent packets while the previous packet is being sent to the CPU.

The transfer completion status information is pushed into the FIFO along with the last packet received by the endpoint. Typically, it is recommended to assign a location to each OUT endpoint.

Transmit FIFO RAM allocation

The minimum RAM space required for each IN endpoint transmit FIFO is the maximum packet size for that particular IN endpoint.

34.5.12 USBHS system performance

Optimum USB and system performance is achieved with large RAM buffers, highly configurable FIFO size, fast 32-bit FIFO access via AHB push/pop registers, and especially advanced FIFO control mechanisms. In fact, USBHS efficiently fills the available RAM space through this mechanism, regardless of the current USB sequence. Take advantage of these features:

- The application has enough headroom to calculate and correct the CPU load to optimize CPU bandwidth utilization:
 - The application can accumulate a large amount of sending data first, and then send it out through the USB
 - Provides sufficient time margin to read data from the receive FIFO
- The USB module can maintain full-speed operation, that is, provide the maximum full-speed bandwidth (as much hardware as possible runs automatically, and as little software as possible)
 - The USB module can accumulate a large amount of transmission data in advance for its disposal, so that it can self-manage USB data transmission
 - There is a lot of empty space in the receive buffer, which is automatically filled with data from the USB

Since the USBHS module can efficiently fill the 8KB RAM buffer and the 8KB transmit/receive data is sufficient for a full-speed frame, the USB system can reach the maximum USB bandwidth within a frame without application intervention.

34.5.13 USBHS interrupts and events

There are four types of USBHS interrupts: SOTP mode wake-up interrupt USBHS_WKUP, endpoint 1 OUT interrupt USBHS_EP1_OUT, endpoint 1 IN interrupt USBHS_EP1_IN, and USBHS global interrupt USBHS_GLB.

USBHS_WKUP interrupt

The USBHS_WKUP interrupt is used to wake up the system in STOP mode through USBHS_DP or USBHS_DM. The interrupt enable bit is INT_WUPEN.USH_WUEN.

Before using USBHS to wake up the system from STOP, it is necessary to ensure that the USBHS controller is in a suspended state, and set the corresponding filter range and enable filter function in the register USBHS_SYCTLREG.

USBHS_EP1_OUT interrupt

The USBHS_EP1_OUT interrupt is the OUT interrupt that the software needs to handle the device mode endpoint 1, and the interrupt flag can be queried in USBHS_DEACHINT.

USBHS_EP1_IN interrupt

The USBHS_EP1_IN interrupt is the IN interrupt that the software needs to handle the device mode endpoint 1. The interrupt flag can be queried in USBHS_DEACHINT

USBHS_GLB interrupt

The USBHS_GLB interrupt is the main interrupt that software needs to handle. The flag bit of the global interrupt can be read in the USBHS_GINTSTS register.

Table 34-2 USBHS_GLB interrupt event table

Interrupt logo	Description	Operating mode	internal event source
WKUPINT	Resume/Remote Wakeup Interrupt	host or device	-
VBUSVINT	VBUS active interrupt	equipment	-
DISCINT	disconnection interrupted	Host	-
CIDSCHG	Connector ID line state change interrupt	host or device	-
PTXFE	Periodic TxFIFO Empty Interrupt	Host	-
LPMINT	LPM interrupt	host or device	
HCINT	host channel interrupt	Host	-
HPRTINT	Host port interrupt	Host	-
DATAFSUSP	Data fetch pending	equipment	-
IPXFR/INCOMPISOOUT	Incomplete periodic transmission/Incomplete OUT synchronization transmission	equipment	-
IISOIXFR	IN SYNC TRANSFER NOT COMPLETED	equipment	-
OEPINT	OUT endpoint interrupt	equipment	-
IEPINT	IN endpoint interrupt	equipment	-
EOPF	Periodic end of frame interrupt	equipment	-
ISOODRP	Drop sync OUT packet interrupt	equipment	-
ENUMDNE	Enumeration complete	equipment	-
USBRST	USB reset interrupt	equipment	-

Interrupt logo	Description	Operating mode	internal event source
USBSUSP	USB suspend interrupt	equipment	-
ESUSP	early pending interrupt	equipment	-
GONAKEFF	Global OUT NAK valid interrupt	equipment	-
GINAKEFF	Global aperiodic IN NAK valid interrupt	equipment	-
NPTXFE	Aperiodic TxFIFO Empty Interrupt	Host	-
RXFNE	RxFIFO not empty interrupt	host or device	-
SOF	start of frame interrupt	host or device	Yes
MMIS	pattern mismatch interrupt	host or device	-

34.6 USBHS programming model

34.6.1 USBHS module initialization

The application must perform the module initialization sequence.

Please refer to the mode determination method 34.5.2 USBHS mode decision .

This section describes the initialization process after the USBHS controller is powered on. Whether working in host or device mode, an application must follow an initialization sequence. All module global registers are initialized according to the module configuration:

1. Program the following fields in the USBHS_GAHBCFG register:
 - Global interrupt mask bit GINTMSK = 1
 - Rx FIFO not empty (RXFNE bit in USBHS_GINTSTS)
 - Periodic Tx FIFO Empty Threshold
2. Program the following fields in the USBHS_GUSBCFG register:
 - FS Timeout Calibration Field
 - USB turnaround time field
3. Software must unmask the following bits in the USBHS_GINTMSK register:
 - pattern mismatch interrupt mask
4. By reading the CMOD bit in USBHS_GINTSTS, software can determine whether the USBHS controller is operating in host mode or device mode.

34.6.2 USBHS host initialization

To initialize the module as a host, the application must perform the following steps:

1. Program HPRTINT in the USBHS_GINTMSK register to unmask the pair.
2. Program the USBHS_HCFG register to select full-speed host.
3. Program the PWPR bit in USBHS_HPRT to 1 to provide VBUS to the USB bus.
4. Wait for the PCDET interrupt in USBHS_HPRT. This indicates that a device is connected to the host port.
5. Program the PRST bit in USBHS_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 USBHS_HPRT to 0.
8. Wait for the PENCHNG interrupt in USBHS_HPRT.
9. Read the PSPD bit in USBHS_HPRT for enumeration speed.
10. Using the selected PHY clock, set the HFIR register accordingly.
11. Program the FSLSPCS field in the USBHS_HCFG register according to the device speed detected in step 9. If the FSLSPCS is changed, a port reset must be performed.
12. Program the USBHS_GRXFSIZ register to select the size of the receive FIFO.
13. Program the USBHS_HNPTXFSIZ register to select the size and start address of the

aperiodic transmit FIFO used for aperiodic communication transactions.

14. Program the USBHS_HPTXFSIZ register to select the size and start address of the periodic communication transmit FIFO for periodic transactions.

To communicate with a device, system software must initialize and enable at least one channel.

34.6.3 USBHS 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 USBHS_DCFG register:
 - device speed
 - Non-zero length state OUT handshake signal
2. Program the USBHS_GINTMSK register to unmask the following interrupts:
 - USB reset
 - Enumeration complete
 - early hang
 - USB hang
 - SOF
3. Wait for the VBUSVINT interrupt in USBHS_GINTSTS to indicate entering the power supply state.
4. Wait for the USBRST interrupt in USBHS_GINTSTS. This indicates that a reset signal has been detected on the USB, and the reset process lasts approximately 10ms since this interrupt was received.
5. Wait for the ENUMDNE interrupt in USBHS_GINTSTS. This interrupt indicates the end of the reset process on the USB. When this interrupt is received, the application must read the USBHS_DSTS register to determine the enumeration speed and perform the steps listed in Endpoint initialization when enumeration completes.

At this point, the device is ready to accept SOF packets and perform control transfers on control endpoint 0.

34.6.4 USBHS DMA mode

USB uses the AHB master interface to get transmit packet data (AHB to USB) and receive data updates (USB to AHB). The AHB master interface uses programmed DMA addresses (HCDMAx registers in host mode and DIEPDMAx/DOEPDMAx registers in device mode) to access the data buffers.

34.6.5 USBHS Host Programming Model

34.6.5.1 Channel initialization

An application must initialize one or more channels before it can communicate with connected devices.

To initialize and enable a channel, the application must perform the following steps:

1. Program the `USBHS_GINTMSK` register to unmask the following bits:
 - The aperiodic transmit FIFO for OUT transactions is empty (applies when working at the pipelined transaction level and the packet count field is programmed to a value greater than 1).
 - The aperiodic transmit FIFO for OUT transactions is half-empty (applies when operating at the pipelined transaction level and the packet count field is programmed to a value greater than 1).
2. Program the `USBHS_HAINTMSK` register to enable interrupts for the selected channel.
3. Program the `USBHS_HCINTMSK` register to enable interrupts related to communication transactions reflected in the Host Channel Interrupt Register.
4. Program the `USBHS_HCTSIZx` registers of the selected channel, specifying the total transfer size in bytes and the expected number of packets including short packets. The application must program the PID field with the initial data PID (for the first OUT transaction or expected to be obtained from the first IN transaction).
5. Program the `USBHS_HCCHARx` registers for the selected channel, specifying the device's endpoint characteristics, such as type, speed, direction, etc. (The channel can be enabled by setting the channel enable bit to 1 only when the application is ready to send or receive packets).

34.6.5.2 Channel stop

The application can disable any channel by programming the `USBHS_HCCHARx` registers to set the CHDIS and CHENA bits. This causes the USBHS host to flush previous requests on that channel (if any) and generate a channel stop interrupt. The application must wait for the CHH interrupt in `USBHS_HCINTx` before reassigning the channel to other communication transactions. The USBHS host does not interrupt communication transactions already initiated on the USB.

Before disabling a channel, the application must ensure that there is at least one free space in the aperiodic request queue (when aperiodic channels are disabled) or the periodic request queue (when periodic channels are disabled). The application can clear the request queue when the request queue is full (before disabling the channel) by programming the `USBHS_HCCHARx` registers to set the CHDIS bit and clear the CHENA bit. The application blocks the channel when any of the following occurs:

1. STALL, TXERR, BBERR or DTERR interrupt received in USBHS_HCINTx of IN or OUT channel.
The application must be able to receive other interrupts (DTERR, Nak, Data, TXERR) for the same channel before receiving the channel stop signal.
2. Received DISCINT (disconnect device) interrupt in USBHS_GINTSTS. (The application will disable all enabled channels).
3. The application aborted the transfer before it completed normally.

In DMA mode, the application cannot stop the indivisible periodic transfer by overwriting the register.

34.6.5.3 Ping protocol

When the USBHS host is operating in high-speed mode, if the application wants to communicate with high-speed bulk or control (data and status phases) OUT endpoints, the ping protocol must be used.

When an application receives a NAK/NYET/TXER interrupt, the pin protocol must be used. When the USBHS host receives one of the above responses, it does not proceed with any communication transaction on that endpoint, but discards all issued or acquired OUT requests (from the request queue) and then flushes the corresponding data in the transmit FIFO .

This is only valid in slave mode. In slave mode, applications can send ping tokens in two ways: Set the DOPIN bit in HCTSIZx before enabling the channel, or set the DOPING bit on a write to the HCTSIZx register after the channel has been enabled. This will enable the USBHS host to write ping requests to the request queue. The application must wait for the device to respond to the ping token (NAK, ACK, or TXERR interrupt) before continuing with the communication transaction or sending another ping token. Only after the application receives an ACK response to the ping token from the device's OUT endpoint can the data communication transaction continue. When working in DMA mode, for bulk/control OUT transactions, the application does not need to set the DOPING bit in HCTSIZx when a NAK/NYET reply is received. The USBHS host automatically sets the DOPING bit in HCTSIZx and then issues a ping token on bulk/control OUT transfers. The USBHS host will continue to send ping tokens until an ACK is received, then automatically switch to a data communication transaction.

34.6.6 USBHS Device Programming Model

34.6.6.1 Endpoint endpoint initialization on USB reset

1. Set NAK bit to 1 for all OUT endpoints
 - In USBHS_DOEPCTLx, SNAK = 1 (for all OUT endpoints)
2. Unmask the following interrupt bits
 - In USBHS_DAINTMSK, INEP0=1 (control 0 IN endpoint)
 - In USBHS_DAINTMSK, OUTEP0=1 (control 0 OUT endpoint)

- In DOEPMISK, STUP=1
 - In DOEPMISK, XFRC=1
 - In DIEPMISK, XFRC=1
 - In DIEPMISK, TOC=1
3. Set data FIFO RAM for each FIFO
 - Program the USBHS_GRXFSIZ register to be able to receive OUT data and setup data for control transfers. This register must be at least equal to 1 maximum packet size for control endpoint 0 + 2 words (for controlling the status of OUT packets) + 10 words (for SETUP packets).
 - Program the USBHS_TX0FSIZ register (depending on the FIFO number selected) to be able to send control IN data. This register must be at least equal to control endpoint 0's 1 maximum packet size.
 4. Program the following fields in the endpoint related registers to control OUT endpoint 0 to receive the SETUP packet
 - STUPCNT=3 in USBHS_DOEPTSIZE0 (receives up to 3 consecutive SETUP packets)

At this point, all initialization required to receive the SETUP packet is complete.

34.6.6.2 Endpoint endpoint initialization on USB reset

1. In the enumeration complete interrupt (ENUMDNE in USBHS_GINTSTS), the USBHS_DSTS register is read to determine the enumeration speed of the device.
2. Program the MPSIZ field in USBHS_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 transfers at control endpoint 0.

34.6.6.3 Endpoint initialization when SetAddress command is received

This section describes what an application must do when it receives a SetAddress command in a SETUP packet.

1. Program the USBHS_DCFG register with the device address received in the SetAddress command
2. Program the module to emit IN packets for the status phase

34.6.6.4 Endpoint initialization when SetConfiguration/SetInterface command is received

This section describes what an application must do when it receives a SetConfiguration or SetInterface command in a SETUP package.

1. When the SetConfiguration command is received, the application must program the endpoint registers to configure these endpoint registers with the characteristics of the

- valid endpoint in the new configuration.
2. When a SetInterface command is received, the application must program the endpoint registers 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 disabled.
 4. Use the USBHS_DAINTMSK register to enable interrupts for valid endpoints and mask interrupts for invalid endpoints.
 5. Set up data FIFO RAM for each FIFO.
 6. After configuring all required endpoints, the application must program the module to send IN packets for the status phase.

At this point, the device module can already receive and send any type of data packet.

34.6.6.5 Endpoint activation

This section describes the steps required to activate a device endpoint or configure an existing device endpoint to a new type.

1. Program the characteristics of the desired endpoint in the following fields of the USBHS_DIEPCTLx register (for IN or bidirectional endpoint) or the USBHS_DOEPCTLx register (for OUT or bidirectional endpoint).
 - maximum packet size
 - USB Active Endpoint Position 1
 - Endpoint Initial Data Sync bit (for interrupt and bulk endpoints)
 - Endpoint type
 - TxFIFO number
2. Once an endpoint is activated, the module starts decoding the token sent to that endpoint and replies with a valid handshake if the received token is valid.

34.6.6.6 Endpoint deactivation

This section describes the steps required to decommission an existing endpoint.

1. Clear the USB Active Endpoint bit in the USBHS_DIEPCTLx register (for IN or bidirectional endpoint) or the USBHS_DOEPCTLx register (for OUT or bidirectional endpoint) in the endpoint to be disabled.
2. When an endpoint is deactivated, the module ignores tokens sent to that endpoint, causing the USB to time out.

34.6.7 USBHS operating model

34.6.7.1 SETUP and OUT data transfer

This section describes the internal data flow and application operation steps during data OUT transfers and SETUP transactions.

packet read

This section describes how to read packets (OUT data and SETUP packets) from the receive FIFO.

1. When the RXFNE interrupt (USBHS_GINTSTS register) is captured, the application must read the Receive Status Popup Register (USBHS_GRXSTSP).
2. The application can mask the RXFNE interrupt (in USBHS_GINTSTS) by writing RXFNE=0 (in USBHS_GINTMSK) 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 popped from the receive data FIFO and stored in memory. If the received packet byte count is 0, no data will be popped from the receive data FIFO.
4. The packet status read from the receive FIFO has the following status:

- Global OUT NAK:

PKTSTS=Global OUT NAK, BCNT=0x000, the values of EPNUM and DPID do not matter.

These data indicate that the global OUT NAK bits are 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 can now be read from the receive FIFO.

- The build phase is complete:

PKTSTS=Setup phase complete, BCNT=0x0, EPNUM=Control EP number, DPID value does not matter.

These data indicate that the setup phase for the specified endpoint is complete and the data phase has started. After this status entry is popped from the receive FIFO, the module will generate a setup interrupt on this control OUT endpoint.

- OUT packets:

PKTSTS=DataOUT, BCNT=the size of the received OUT data packet (BCNT:0~1024), EPNUM=the endpoint number of the received data packet, DPID=the actual data PID.

- Data transfer complete:

PKTSTS=OUT data transfer completed, BCNT=0x0, EPNUM=OUT EP number that completes data transfer, DPID value does not matter.

These data indicate that the OUT data transfer of the specified OUT endpoint is complete. After this status entry is popped from the receive FIFO, the module will raise a "transfer complete" interrupt on the specified OUT endpoint.

5. After popping data from the receive FIFO, the RXFNE interrupt must be unmasked (USBHS_GINTSTS).
6. Steps 1 to 5 are repeated each time the application detects an RXFNE interrupt in USBHS_GINTSTS. Reading an empty receive FIFO may result in undefined module behavior.

SETUP transaction

This section describes how the module handles SETUP packets and the order in which the application handles SETUP transactions.

Application Requirements:

1. To receive a SETUP packet, the Control OUT endpoint STUPCNT field (USBHS_DOEPTSIZx) must be programmed to a non-zero value. If the application programs the STUPCNT field to a non-zero value, the module receives and writes the SETUP packet to the receive FIFO, regardless of the NAK status and the EPENA bit setting in USBHS_DOEPCTLx. The STUPCNT field is decremented each time the control endpoint receives a SETUP packet. If the STUPCNT field is not programmed to an appropriate value before receiving the SETUP packet, the module can still receive the SETUP packet and decrement the STUPCNT field, but the application may not be able to determine the correct number of SETUP packets received during the setup phase of the control transfer.
 - In USBHS_DOEPTSIZx, 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.
 - Reserve space for 10 characters. The first SETUP packet requires 3 words, the "setup phase complete" status double word requires 1 word, and 6 words are required 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 to store SETUP packets and will never use this space for data packets.
3. The application must read 2 words of the SETUP packet from the receive FIFO.
4. The application must read and discard the "Setup Phase Complete" status word from the receive FIFO

Internal data flow:

1. When a SETUP packet is received, the module writes the received data into the receive FIFO without checking the available space in the receive FIFO and regardless of the endpoint's NAK and STALL bit settings.
 - The module will internally set the IN NAK and OUTNAK bits of the control IN/OUT endpoint that received the SETUP packet.
2. For each SETUP packet received on the USB, the module writes 3 words of data into the receive FIFO and decrements the STUPCNT field by 1.

- The first word contains 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 setup phase ends and the data IN/OUT phase begins, the module writes a status entry ("Setup Phase Complete" word) to the receive FIFO to indicate that the setup phase is complete.
 4. On the AHB side, the SETUP packet is read by the application.
 5. When the application pops the "Setup Phase Complete" word from the receive FIFO, the module will use the STUP interrupt (USBHS_DOEPINTx) to interrupt the application, indicating that it can process the received SETUP packet.
 - The module will clear the endpoint enable bit that controls the OUT endpoint.

Application programming sequence:

1. Program the USBHS_DOEPTSIZx registers.
 - STUPCNT=3
2. Wait for the RXFNE interrupt (USBHS_GINTSTS) and read the packet from the receive FIFO.
3. Triggering of STUP interrupt (USBHS_DOEPINTx) indicates the successful completion of SETUP data transfer.
 - When this interrupt occurs, the application must read the USBHS_DOEPTSIZx registers to determine the number of SETUP packets received and process the last SETUP packet received.

Process more than three consecutive SETUP packets:

According to the USB2.0 specification, in a SETUP packet error, the host usually does not send more than 3 consecutive SETUP packets to the same endpoint. However, the USB 2.0 specification does not limit the number of consecutive SETUP packets a host can send to the same endpoint. When this happens, the USBHS controller will generate an interrupt (B2BSTUP in USBHS_DOEPINTx).

Set global OUT NAK to 1

Internal data flow:

1. If the application sets the global OUT NAK (SGONAK bit in USBHS_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 with a NAK to the asynchronous OUT token sent by the host, and ignores the synchronous OUT packet.
2. The module writes the global OUT NAK to the receive FIFO. The application must allow enough space for this.
3. The module sets the GONAKEFF interrupt (USBHS_GINTSTS) when the application pops the global OUT NAK word from the receive FIFO.
4. When the application detects this interrupt, it considers the module to be in global OUT

NAK mode. The application can clear this interrupt by clearing the SGONAK bit in USBHS_DCTL.

Application programming sequence:

1. To stop receiving any type of data into the receive FIFO, the application must set the global OUT NAK bit by programming the following fields.
 - In USBHS_DCTL, SGONAK =1
2. Wait for GONAKEFF interrupt in USBHS_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 bit in USBHS_DCTL, the application can receive valid OUT packets before the module raises the GONAKEFF interrupt (USBHS_GINTSTS).
3. The application can temporarily mask this interrupt by writing to the GINAKEFFM bit in the USBHS_GINTMSK register.
 - In the USBHS_GINTMSK register, GINAKEFFM=0
4. When the application is ready to exit global OUT NAK mode, the SGONAK bit in USBHS_DCTL must be cleared. This action also clears the GONAKEFF interrupt (USBHS_GINTSTS).
 - In CGONAK, USBHS_DCTL=1
5. If the application has previously masked this interrupt, it must unmask the interrupt as follows:
 - In GINTMSK, GINAKEFFM=1

Set global OUT NAK to 1

The application must disable enabled OUT endpoints using the following sequence.

Application programming sequence:

1. The application must enable global OUT NAK mode in the module before disabling any OUT endpoints.
 - In USBHS_DCTL, SGONAK=1
2. Waiting for GONAKEFF interrupt (USBHS_GINTSTS)
3. Disable the OUT endpoint by programming the following fields:
 - In USBHS_DOEPCTLx, EPDIS=1
 - In USBHS_DOEPCTLx, SNAK=1
4. Wait for the EPDSD interrupt (USBHS_DOEPINTx), which indicates that the OUT endpoint is completely disabled. When an EPDSD interrupt is raised, the module also clears the following bits:
 - In USBHS_DOEPCTLx, EPDIS=0
 - In USBHS_DOEPCTLx, EPENA=0
5. The application must clear the global OUT NAK bit to start receiving data from other non-disabled OUT endpoints.

- In USBHS_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 data to be received as part of an OUT transfer.
2. For OUT transfers, the transfer size field in the endpoint's transfer size register must be a multiple (and word-aligned) of the endpoint's maximum packet size.
 - 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 endpoint's transfer size register to calculate the amount of data available in memory. The amount of valid data received may be less than the programmed transfer size.
 - Effective data volume in memory = initial transfer volume set by application - remaining transfer volume after module update
 - Number of received USB packets = initial number of packets set by the application - number of remaining packets after module update

Internal data flow:

1. The application must set the transfer size and packet count fields in the endpoint-related registers, clear the NAK bit, and enable the endpoint to receive data.
2. After the NAK bit is cleared, the module begins to receive data and write the data to the receive FIFO (as long as there is room in the receive FIFO). For every packet received on the USB, the packet and its status are written to the receive FIFO. Each packet written to the receive FIFO (either a packet with a maximum packet size or a short packet) decrements the packet count field for that endpoint by 1.
 - If the CRC of the received data packet is invalid, it is automatically cleared from the receive FIFO.
 - After an ACK is returned for the packet on the USB, the module will discard the asynchronous OUT packet that the host retransmitted because the ACK could not be detected. The application will not detect multiple consecutive OUT packets on the same endpoint with the same data PID. In this case, the packet count will not be decremented.
 - If there is no room in the receive FIFO, synchronous or asynchronous packets are ignored and not written to the receive FIFO. Additionally, asynchronous OUT tokens will receive a NAK handshake response.
 - In all three cases above, the packet count will not be decremented because no data is written to the receive FIFO.

3. When the packet count becomes 0 or a short packet is received on the endpoint, the NAK bit for that endpoint will be set to 1. When the NAK bit is set, synchronous or asynchronous packets are ignored and not written to the receive FIFO, and the non-synchronous OUT token receives a NAK handshake reply.
4. After the data is written to the receive FIFO, the application will read the data from the receive FIFO and write the data to the external memory, one packet at a time, endpoint by endpoint.
5. After each packet is written to external memory on the AHB, the transfer size of the endpoint is automatically subtracted by that packet size.
6. The OUT data transfer completion status of the OUT endpoint is written to the receive FIFO when:
 - The transfer size is 0 and the 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 completion interrupt for this endpoint, and clears the endpoint enable bit.

Application programming sequence:

1. Program the USBHS_DOEPTSIZx registers with the transfer size and the corresponding number of packets.
2. Program the USBHS_DOEPCTLx registers using the endpoint characteristics and set the EPENA and CNAK bits to 1.
 - In USBHS_DOEPCTLx, EPENA=1
 - In USBHS_DOEPCTLx, CNAK =1
3. Wait for the RXFNE interrupt (in USBHS_GINTSTS) and read the packet from the receive FIFO.
 - This step can be repeated multiple times, depending on the transfer size.
4. Trigger the XFRC interrupt (USBHS_DOEPINTx) to indicate the successful completion of the asynchronous OUT data transfer.
5. Read the USBHS_DOEPTSIZx registers to determine the amount of valid data.

Universal synchronous OUT data transfer

This section describes conventional synchronous OUT data transfers.

Application Requirements:

1. All application requirements for asynchronous OUT data transfers apply to same OUT data transfers.
2. For the transfer size and packet count fields in isochronous OUT data transfers, they must always be set to the number of packets of the maximum packet size that can be received

in a single frame. A synchronous type of OUT data transfer transaction must be completed within one frame.

3. Before the end of the periodic frame (EOPF interrupt in USBHS_GINTSTS), the application must read all isochronous OUT packets (data entry and status entry) from the receive FIFO.
4. To receive data in the next frame, a synchronous OUT endpoint must be enabled after EOPF (USBHS_GINTSTS) and before SOF (USBHS_GINTSTS).

Internal data flow:

1. The internal data flow of a synchronous OUT endpoint is basically the same as that of an asynchronous OUT endpoint, with a few differences.
2. When a synchronous OUT endpoint is enabled by setting the endpoint enable bit and clearing the NAK bit, the even/odd frame bits 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 USBHS_DOEPCTLx) = FNSOF[0] (in USBHS_DSTS)
3. When the application reads a complete isochronous OUT packet (data and status) from the receive FIFO, the module updates the RXDPID field in USBHS_DOEPSIZx according to the data PID of the last isochronous OUT packet read from the receive FIFO.

Application programming sequence:

1. Program the USBHS_DOEPSIZx registers with the transfer size and corresponding packet count
2. Program the USBHS_DOEPCTLx registers using the endpoint characteristics and set the endpoint enable bit, clear NAK bit, and odd/even frame bits.
 - EPENA1
 - CNAK=1
 - EONUM=(0: even/1: odd)
3. Wait for the RXFNE interrupt (in USBHS_GINTSTS) and read the packet from the receive FIFO.
 - This step can be repeated multiple times, depending on the transfer size.
4. The XFRC interrupt (in USBHS_DOEPINTx) indicates that the synchronous OUT data transfer is complete. The interrupt does not necessarily mean that the data in memory is valid.
5. For isochronous OUT transfers, the interrupt may not always be detected by the application. Instead, the application may detect the IISOOXFRM interrupt in USBHS_GINTSTS.
6. Read the USBHS_DOEPSIZx registers to determine the received transfer size and to determine the validity of the data received in the frame. An application must treat data received in memory as valid data only if one of the following conditions is met:

- RXDPID=D0 (in USBHS_DOEPTSIZx) and the number of USB packets receiving this valid data=1
- RXDPID=D1 (in USBHS_DOEPTSIZx) and number of USB packets receiving this valid data=2
- RXDPID=D2 (in USBHS_DOEPTSIZx) and the number of USB packets receiving this valid data=3

The number of USB data packets that receive the valid data = the number of initial data packets programmed by the application program - the number of remaining data packets after the module is updated.

Applications can drop invalid packets.

Incomplete sync OUT data transfer

This section describes the application programming sequence in the event of loss of synchronous OUT packets.

Internal data flow:

1. For synchronous OUT endpoints, the XFRC interrupt (in USBHS_DOEPINTx) may not always be raised. If the module drops isochronous OUT packets, the application may fail to detect the XFRC interrupt (USBHS_DOEPINTx) under the following conditions:
 - When the receive FIFO cannot hold a complete ISO OUT packet, the module will discard the received ISO OU data
 - Received sync OUT packet with CRC error
 - The SYNC OUT token received by the module is corrupted
 - The application is very slow to read data from the receive FIFO
2. If the module detects a periodic end of frame before the transfer of all isochronous OUT endpoints is complete, the outstanding isochronous OUT data interrupt (IISOXXFRM in USBHS_GINTSTS) is triggered, indicating that the XFRC interrupt (in USBHS_DOEPINTx) is not triggered on at least one isochronous OUT endpoint. At this point, the endpoint that has not completed the transfer remains enabled, but there is no active transfer in progress on that endpoint of the USB.

Application programming sequence:

1. A hardware-triggered IISOXXFRM interrupt (USBHS_GINTSTS) indicates that at least one isochronous OUT endpoint in the current frame has an outstanding transfer.
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 read from the receive FIFO before proceeding.
 - Once all data has been read from the receive FIFO, the application can detect the XFRC interrupt (USBHS_DOEPINTx). In this case, the application must re-enable the endpoint to

receive synchronous OUT data in the next frame.

3. When the application receives the IISO0XFRM interrupt (in USBHS_GINTSTS), the application must read the control registers (USBHS_DOEPCTLx) of all isochronous OUT endpoints to determine which endpoints have incomplete transfers in the current frame. When both of the following conditions are met at the same time, it indicates that the endpoint transfer is not complete:
 - EONUM bit (in USBHS_DOEPCTLx) = FNSOF[0] (in USBHS_DSTS)
 - EPENA=1 (in USBHS_DOEPCTLx)
4. Before the SOF interrupt is detected (in USBHS_GINTSTS), the previous step must be done to ensure that the current frame number has not changed.
5. For isochronous OUT endpoints with incomplete transfers, the application must discard the data in memory and disable the endpoint by setting the EPDIS bit in USBHS_DOEPCTLx.
6. Wait for the EPDIS interrupt (in USBHS_DOEPINTx) and enable the endpoint to receive new data in the next frame.
 - Since the module may take some time to disable the endpoint, the application may not receive data in the next frame after receiving invalid sync data.

Stop asynchronous OUT endpoints

This section describes how an application can stop an asynchronous endpoint.

1. Put the module in global OUT NAK mode.
2. Ban the required endpoints
 - When disabling an endpoint, set STALL=1 (in USBHS_DOEPCTL) instead of setting the SNAK bit in USBHS_DOEPCTL. The STALL bit always takes precedence over the NAK bit.
3. The STALL bit (in USBHS_DOEPCTLx) must be cleared when the application no longer needs the endpoint to reply to the STALL handshake signal.
4. If the application sets or clears the STALL state of an endpoint as a result of receiving a SetFeature.Endpoint Halt or ClearFeature.Endpoint Halt command from the host, it must set or clear the STALL bit before the state phase transfer on that control endpoint.

34.6.7.2 IN data transfer

packet write

This section describes how an application can write packets to the endpoint FIFO when the dedicated transmit FIFO is enabled.

1. The application can choose polling mode or interrupt mode.
 - In polling mode, the application monitors the status of the endpoint transmit data FIFO by reading the USBHS_DTXFSTSx registers to determine if there is enough space in the data FIFO.
 - In interrupt mode, the application waits for the TXFE interrupt (in USBHS_DIEPINTx) and

then reads the USBHS_DTXFSTSx registers 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 hold the entire packet.
 - To write a zero-length packet, the application cannot look into the FIFO space.
2. Using one of the above methods, when the application determines that there is enough space to write the transmit packet, the application must first write the endpoint control register accordingly, and then write the data to the data FIFO. Typically, applications must perform read-modify-write operations on the USBHS_DIEPCTLx registers to avoid modifying other contents of the registers while setting the endpoint enable bit.

An application can write multiple packets of 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. All packets to be sent in the next frame will not be written by the application until the transmission of the communication transaction of the previous frame is completed.

Set IN endpoint NAK to 1

Internal data flow:

1. When the application sets the IN NAK of a specific endpoint to 1, the module will stop data transmission on the endpoint regardless of whether the data in the endpoint transmit FIFO is available.
2. The asynchronous endpoint receives the IN token and replies with a NAK handshake response.
 - Sync endpoint receives IN token, returns zero-length packet
3. The module triggers the INEPNE (IN endpoint NAK valid) interrupt in USBHS_DIEPINTx in response to the SNAK bit in USBHS_DIEPCTLx.
4. When the application detects this interrupt, it considers the endpoint to be in IN NAK mode.
The application can clear this interrupt by setting the CNAK bit in USBHS_DIEPCTLx.

Application programming sequence:

1. To stop sending any data on a specific IN endpoint, the application must set the IN NAK bit. To set this bit, the following fields must be programmed.
 - SNAK=1 in USBHS_DIEPCTLx
2. Wait for the INEPNE interrupt in USBHS_DIEPINTx to fire. This interrupt indicates that the module has stopped sending data on the endpoint.
3. When the application sets the NAK bit but the "NAK Valid" interrupt has not yet triggered, the module can send valid IN data on the endpoint.
4. The application can temporarily mask this interrupt by writing to the INEPNEM bit in DIEPMSK.

- In DIEPMSK, INEPNEM = 0
5. To exit endpoint NAK mode, the application must clear the NAK status bit (NAKSTS) in USBHS_DIEPCTLx. This action also clears the INEPNE interrupt (in USBHS_DIEPINTx).
 - In USBHS_DIEPCTLx, CNAK=1 6. If the application has masked the interrupt, it must be unmasked as follows:
 - In DIEPMSK, INEPNEM=1

Disable IN endpoint

Use the following sequence to disable specific IN endpoints that were previously enabled.

Application programming sequence:

1. The application must stop writing data on the AHB before disabling the IN endpoint.
2. The application must set the endpoint to NAK mode.
 - SNAK=1 in USBHS_DIEPCTLx
3. Wait for INEPNE interrupt in USBHS_DIEPINTx.
4. Set the following bits in the USBHS_DIEPCTLx registers for endpoints that must be disabled.
 - EPDIS=1 in USBHS_DIEPCTLx
 - SNAK=1 in USBHS_DIEPCTLx
5. Triggering of the EPDISD interrupt in USBHS_DIEPINTx indicates that the module has completely disabled the specified endpoint. When the interrupt is triggered, the module also clears the following bits:
 - In USBHS_DIEPCTLx, EPENA=0
 - In USBHS_DIEPCTLx, EPDIS=0
6. The application must read the USBHS_DIEPTSIZx registers for periodic IN EPs to calculate how much data is being sent over the USB on the endpoint.
7. The application must clear the endpoint transmit FIFO by setting the following fields in the USBHS_GRSTCTL register:
 - TXFNUM (in USBHS_GRSTCTL) = endpoint transmit FIFO number
 - TXFFFLSH (in USBHS_GRSTCTL) = 1

The application must poll the USBHS_GRSTCTL register until the module clears the TXFFFLSH bit, which indicates the end of the FIFO flush operation. To send new data on this endpoint, the application can re-enable the endpoint at a later time.

Universal aperiodic IN data transmission

Application Requirements:

1. Before establishing an IN transfer, the application must ensure that each packet that makes up an IN transfer 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 maximum packet size packets plus a short packet at the end of the transfer:

$$\text{transfer size[EPNUM]} = x \times \text{MPSIZ[EPNUM]} + \text{sp}$$

If ($\text{sp} > 0$), packet count [EPNUM] = $x + 1$.

Otherwise, Packet Count [EPNUM] = x

- To send a single zero-length packet:

$$\text{transfer size [EPNUM]} = 0$$

Packet Count [EPNUM] = 1

- To send multiple maximum packet size packets with a zero-length packet at the end of the transfer, the application must split the transfer into two parts.

The first part sends packets of maximum packet size, and the second part only sends zero-length packets.

First transfer: transfer size[EPNUM] = $x \times \text{MPSIZ[epnum]}$; packet count = n ;

Second transfer: transfer size[EPNUM] = 0; packet count = 1;

3. After an endpoint is enabled for data transfer, the module updates the transfer size register. At the end of the 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 over the USB.
4. Amount of data fed into transmit FIFO = initial transfer size programmed by application - final transfer size after module update
 - Amount of data that has been sent over USB = (Initial packet count programmed by the application - Final packet count after module update) $\times \text{MPSIZ[EPNUM]}$
 - Amount of data remaining to be sent over USB = (initial transfer size programmed by the application - amount of data already sent over USB)

Internal data flow:

1. The application must set the transfer size and packet count fields in the registers of a specific endpoint and enable that endpoint to send data.
2. The application must also write the necessary data to that endpoint's transmit FIFO.
3. Each time the application writes a packet to the transmit FIFO, the packet size is automatically subtracted from the transfer size of the endpoint. The application continues to fetch data from memory to write to the transmit FIFO until the transfer size for that endpoint becomes 0. After writing data to the FIFO, the "Packets in FIFO" count is incremented (this is a 3-bit count maintained internally by the module, one for each IN endpoint transmit FIFO. In the IN endpoint FIFO, the maximum number of packets maintained by the module is always eight). For zero-length packets, each FIFO has an additional separate flag, and there is no data in the FIFO.
4. After the data is written into the transmit FIFO, the module will send the data out when it receives the IN token. After each packet is sent and the ACK handshake is received, the

packet count of the endpoint is decremented by 1 until the packet count becomes 0. When a timeout occurs, the packet count is not decremented.

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 receiving the IN token, and the packet count field of the endpoint is zero, the module will generate an "IN token received while TxFIFO is empty" (ITTXFE) interrupt for the endpoint (Provided that the endpoint's NAK bit is not set). The module replies with a NAK handshake signal on the asynchronous endpoint.
7. The module will internally bring the FIFO pointer back to the beginning and will not generate a timeout interrupt.
8. When the transfer size is 0 and the packet count is 0, a transfer complete (XFRC) interrupt for this endpoint is generated and the endpoint enable is cleared.

Application programming sequence:

1. Program the USBHS_DIEPTSIZx registers with the transfer size and corresponding packet count.
2. Program the USBHS_DIEPCTLx registers using the endpoint characteristics and set the CNAK and EPENA (endpoint enable) bits to 1.
3. When sending a non-zero length packet, the application must poll the USBHS_DTXFSTSx registers (where x is the FIFO number associated with this endpoint) to determine if there is enough space in the data FIFO. The application can also select the TXFE bit (in USBHS_DIEPINTx) before writing 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 aperiodic IN data transfer apply equally to cyclic IN data transfer (with a slight modification to requirement 2).
 - An application can only send a number of maximum packet size packets or a number of maximum packet size packets, plus a short packet at the end of the transfer.

To send multiple maximum packet size packets plus a short packet at the end of the transmission, the following conditions must be met:

transfer size[EPNUM] = $x \times \text{MPSIZ}[EPNUM] + sp$

(where x is an integer greater than 0, and sp ranges from 0 to $\text{MPSIZ}[EPNUM]-1$)

If ($sp > 0$), packet count [EPNUM] = $x + 1$

Otherwise, PacketCount[EPNUM] = x;

MCNT[EPNUM] = Packet Count [EPNUM]

- The application cannot send zero-length packets at the end of the transfer. An application

can send a zero-length packet by itself.

- To send a single zero-length packet:

Transfer Size[EPNUM]=0

Packet Count[EPNUM]=1

MCNT[EPNUM]=packet count[EPNUM]

2. An application can only schedule data transfers for one frame at a time.

- $(MCNT - 1) \times MPSIZ < XFERSIZ \leq MCNT \times MPSIZ$
- PKTCNT = MCNT (in USBHS_DIEPTSIZx)
- If XFERSIZ < MCNT × MPSIZ, the last packet transmitted is a short packet
- caution: MCNT is in USBHS_DIEPTSIZx, MPSIZ is in USBHS_DIEPCTLx, PKTCNT is in USBHS_DIEPTSIZx, XFERSIZ is in USBHS_DIEPTSIZx
- 3. Before receiving the IN token, the application must write the complete data to be sent in the frame into the transmit FIFO. When receiving the IN token, even if the data to be sent in the frame is only one double word missing in the transmit FIFO, the module will perform the operation when the FIFO is empty. When the transmit FIFO is empty:
 - A zero-length packet will be replied on the isochronous endpoint
 - The NAK handshake signal will be replied on the interrupt endpoint

Internal data flow:

1. The application must set the transfer size and packet count fields in the registers of a specific endpoint and enable that endpoint to send data.
2. The application must also write the necessary data to the transmit FIFO associated with the endpoint.
3. Each time an 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 for that endpoint becomes 0.
4. When the periodic endpoint receives the IN token, the module will start sending the data in the FIFO (if there is data in the FIFO). If the FIFO does not have a complete packet of the data to be sent for this frame, the module will generate an "IN token received while TxFIFO is empty" interrupt for that endpoint.
 - A zero-length packet will be replied on the isochronous endpoint
 - The NAK handshake signal will be replied on the interrupt endpoint
5. An endpoint's packet count is decremented by 1 under the following conditions:
 - For isochronous endpoints, when sending a zero-length or non-zero-length packet
 - For interrupt endpoints, decrement when sending ACK handshake
 - When the transfer size and packet count are both 0, a transfer complete interrupt for that endpoint will be generated and the endpoint enable bit will be cleared.
6. During the "periodic frame interval" (controlled by the PFIVL bit in USBHS_DCFG), when

the module finds that any data in the synchronous IN endpoint FIFO that should be empty in the current frame has not been sent, it will generate an IISOIXFR interrupt in USBHS_GINTSTS .

Application programming sequence:

1. Program the USBHS_DIEPCTLx registers using the endpoint characteristics and set the CNAK and EPENA bits.
2. Write the data that needs to be sent in the next frame into the transmit FIFO.
3. The hardware triggers the ITTXFE interrupt (in USBHS_DIEPINTx) to indicate that the application has not written all the data that needs to be sent into the transmit FIFO.
4. If the interrupt endpoint was enabled before the interrupt was detected, the interrupt will be ignored. If the interrupt endpoint is not enabled, enable this endpoint so that data can be sent out on the next IN token received.
5. When the hardware triggers the XFRC interrupt (in USBHS_DIEPINTx), if the ITTXFE interrupt is not generated in USBHS_DIEPINTx, it means that the synchronous IN transfer is successfully completed. Reading the USBHS_DIEPTSIz registers always results in transfer size = 0 and packet count = 0, which means that all data has been sent over the USB.
6. When the XFRC interrupt (in USBHS_DIEPINTx) is set, whether or not the ITTXFE interrupt (in USBHS_DIEPINTx) is generated, it indicates the successful completion of the interrupt IN transfer. Reading the USBHS_DIEPTSIz register always results in transfer size = 0 and packet count = 0, which means that all data has been sent over the USB.
7. If none of the aforementioned interrupts are generated when the outstanding synchronous IN transfer (IISOIXFR) interrupt is set in USBHS_GINTSTS, it means that the module has not received at least one periodic IN token in the current frame.

Incomplete sync IN data transfer

This section describes what an application must do for incomplete sync IN data transfers.

Internal data flow:

1. A synchronous IN transfer is considered incomplete when one of the following conditions is met:
 - a) The module received a corrupted sync IN token on at least one sync IN endpoint. At this point, the application detects an incomplete synchronous IN transfer interrupt (bit IISOIXFR in USBHS_GINTSTS).
 - b) The application was too slow to write data to the transmit FIFO and received an IN token before the complete data was written to the FIFO. At this point, the application detects an "IN token received while TxFIFO is empty" interrupt in USBHS_DIEPINTx. The application can ignore this interrupt, as this will eventually generate an

outstanding isochronous IN transfer interrupt (bit IISOIXFR in USBHS_GINTSTS) at the end of the periodic frame. The module responds to the received IN token by sending a zero-length packet over USB.

2. The application must stop writing data to the transmit FIFO as soon as possible.
3. The application must set the endpoint's NAK and inhibit bits to 1.
4. The module disables the endpoint, clears the disable bit and triggers the endpoint's Endpoint Disable interrupt.

Application programming sequence:

1. The application MAY ignore the "IN token received while TxFIFO is empty" interrupt in USBHS_DIEPINTx on any synchronous IN endpoint, as this will eventually generate an outstanding synchronous IN transfer interrupt (in USBHS_GINTSTS).
2. A hardware-triggered outstanding sync IN transfer interrupt (in USBHS_GINTSTS) indicates that there is an outstanding sync IN transfer on at least one sync IN endpoint.
3. The application must read the "Endpoint Control" registers of all isochronous IN endpoints to detect the presence of endpoints with outstanding IN data transfers.
4. The application must stop writing data to the "periodic transmit FIFO" associated with these endpoints.
5. Program the following fields in the USBHS_DIEPCTLx registers to disable endpoints:
 - SNAK=1 in USBHS_DIEPCTLx
 - EPDIS=1 in USBHS_DIEPCTLx
6. Hardware triggering of the "Endpoint Disabled" interrupt in USBHS_DIEPINTx indicates that the module has disabled the endpoint.
 - At this point, the application must either flush the data in the associated transmit FIFO, or overwrite the existing data in the FIFO by enabling the endpoint for a new transfer in the next frame. To refresh data, the application must use the USBHS_GRSTCTL register.

Stop asynchronous IN endpoint

This section describes how an application can stop an asynchronous endpoint.

Application programming sequence:

1. Disable the IN endpoint to be stopped. Also set the STALL bit to 1.
2. EPDIS=1 in USBHS_DIEPCTLx (when endpoint is enabled)
 - STALL=1 in USBHS_DIEPCTLx
 - STALL bits always have higher priority than NAK bits
3. Hardware triggering of the "Endpoint Disable" interrupt (in USBHS_DIEPINTx) lets the application know that the module has disabled the specified endpoint.
4. The application must empty the aperiodic or periodic transmit FIFO depending on the endpoint type. For aperiodic endpoints, the application must re-enable another aperiodic endpoint without stopping to send data.

5. When the application is ready to end the STALL handshake for this endpoint, it must clear the STALL bit in USBHS_DIEPCTLx.
6. If the application sets or clears the STALL state of an endpoint as a result of receiving a SetFeature.Endpoint Halt command or a ClearFeature.Endpoint Halt command from the host, the STALL bit must be set or cleared before the state phase transfer for that control endpoint.

special case: Stop controlling the OUT endpoint

If, during the data phase of the control transfer, the host sends more IN/OUT tokens than the value specified in the SETUP packet, the module must reply with STALL for these redundant IN/OUT tokens. In this case, the application must enable the ITTXFE interrupt of USBHS_DIEPINTx and the OTEPDIS interrupt of USBHS_DOEPINTx during the data phase of the control transfer (after the module has finished transferring the amount of data specified by the SETUP packet). Subsequently, when the application receives this interrupt, it must set the STALL bit in the corresponding endpoint control register and clear the interrupt.

34.7 Register description

The application program reads and writes the control and status registers through the AHB slave interface to control the USBHS module. All registers of the USBHS module are 32-bit registers, and their addresses are aligned with 32-bits, so they can only be accessed in 32-bit ways.

Control and status registers are divided into the following categories:

- USBHS System Control Register
- module global register
- Host Mode Register
- Device Mode Register
- Power and Clock Gating Control Registers
- Data FIFO (DFIFO) Access Register

Among them, the USBHS system control register is different from other register base addresses. This register is independent of the USBHS module to control the relevant settings of the USBHS module.

Only module global registers, power and clock gating control registers, and data FIFO access registers can be accessed in host and device modes. When the USBHS module is in one mode (host or device), the application must not access the registers in another role mode, such as the host mode, access the device mode registers. If an illegal access occurs, a pattern mismatch interrupt will be generated and reflected in the USBHS.GINTSTS.NMIS bit in 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.

Control Status Register Memory Map

The host and device mode registers occupy different addresses. All registers are implemented in the AHB clock domain.

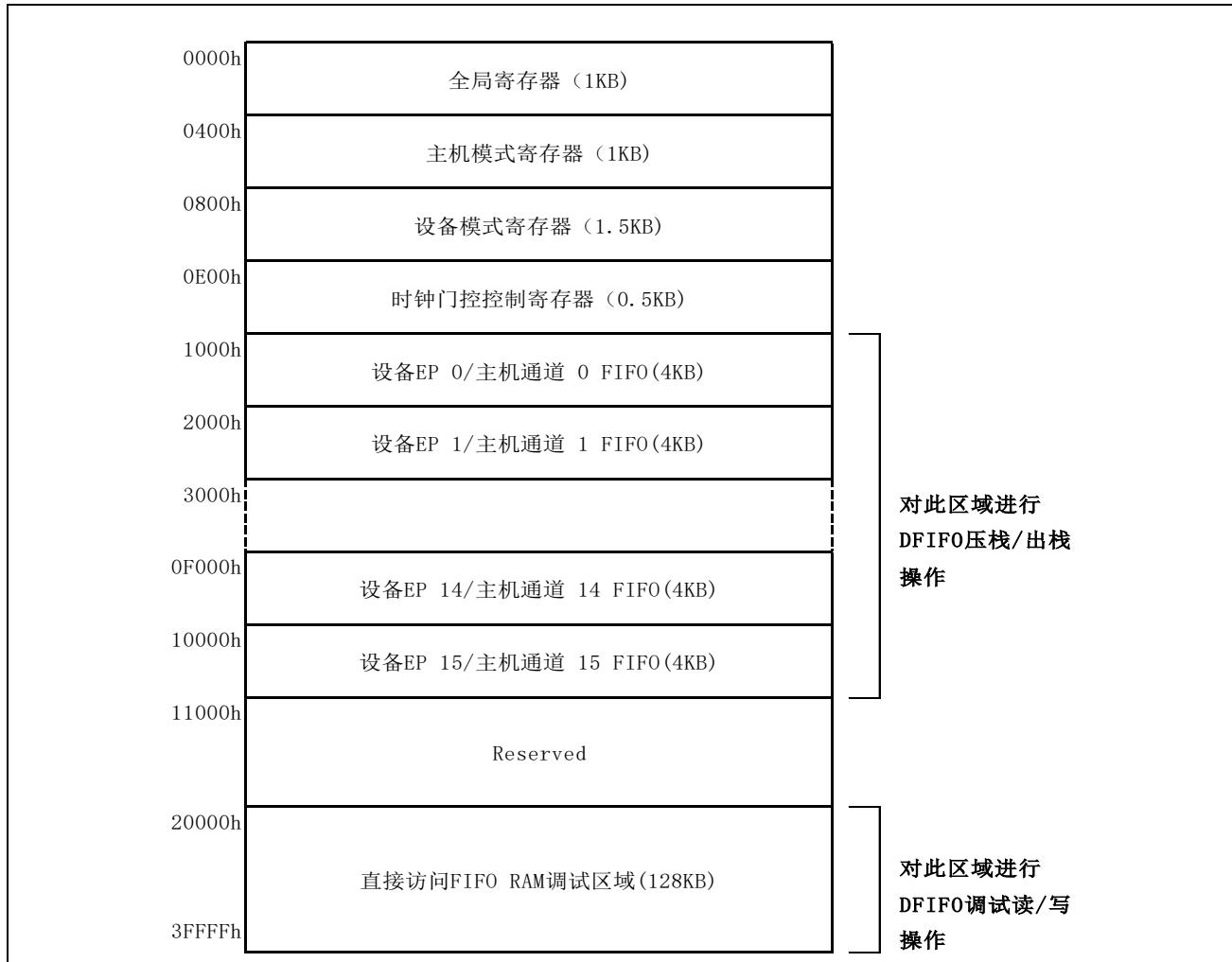


Figure 34-7 USBHS Control Status Register Memory Map

Please refer to USBHS module register list and base addressTable 34-3 ~Table 34-4 List of USBHS registers.

USB System Control Register Base Address: 0x40055400

Table 34-3 List of USBHS System Control Registers

(USB system control register) register name	Offset address	Reset value
USB System Control Register (USBHS SYCTLREG)	0x00	0x0000 0000

USBHS module register base address: 0x400C0000

Table 34-4 List of USBHS System Control Registers

(USBHS global register) register name	Offset address	Reset value
USBHS VBUS Control Register (USBHS_GVBUSCFG)	0x00	0x0000 0000
USBHS AHB Control Register (USBHS_GAHBCFG)	0x08	0x0000 0000
USBHS USB Configuration Register (USBHS_GUSBCFG)	0x0c	0x0000 1410
USBHS Reset Register (USBHS_GRSTCTL)	0x10	0x8000 0000
USBHS Module Interrupt Register (USBHS_GINTSTS)	0x14	0x1400 0020
USBHS Interrupt Mask Register (USBHS_GINTMSK)	0x18	0x0000 0000
USBHS Receive Status Debug Read Register (USBHS_GRXSTSR)	0x1c	0x0000 0000
USBHS Receive Status Read and Pop Register (USBHS_GRXSTSP)	0x20	0x0000 0000
USBHS Receive FIFO Size Register (USBHS_GRXFSIZ)	0x24	0x0000 0800
USBHS host aperiodic transmit FIFO size register (USBHS_HNPTXFSIZ)/ Device Endpoint 0 Transmit FIFO Size Register (USBHS_DIEPRXF0)	0x28	0x0800 0800
USBHS Aperiodic Transmit Status Register (USBHS_HNPTXSTS)	0x2c	0x0008 0800
USBHS Module ID Register (USBHS_CID)	0x3c	0x1234 5678
USBHS LPM Configuration Register (USBHS_GLPMCFG)	0x54	0x0000 0000
USBHS Periodic Transmit FIFO Size Register (USBHS_HPTXFSIZ)	0x100	0x0800 1000
USBHS device IN endpoint n transmit FIFO size register (USBHS_DIEPTXFx)	0x100+x*4(x=1~15)	0x0800 0100+(x-1)*0x800

(USBHS Host Control and Status Register) register name	Offset address	Reset value
USBHS Host Configuration Register (USBHS_HCFG)	0x400	0x0000 0200
USBHS Host Frame Interval Register (USBHS_HFIR)	0x404	0x0000 EA60
USBHS Host Frame Number/Frame Remaining Interval Register (USBHS_HFNUM)	0x408	0x0000 3FFF
USBHS Host Periodic Transmit FIFO/Queue Status Register (USBHS_HPTXSTS)	0x410	0x0008 0800
USBHS Host All Channel Interrupt Register (USBHS_HAINT)	0x414	0x0000 0000
USBHS Host All Channel Interrupt Mask Register (USBHS_HAINTMSK)	0x418	0x0000 0000
USBHS Host Port Control and Status Register (USBHS_HPRT)	0x440	0x0000 0000
USBHS Host Channel x Characteristics Register (USBHS_HCCHARx)	0x500+x*0x20(x=0~15)	0x0000 0000
USBHS Host Channel x Split Control Register (USBHS_HCSPLTx)	0x504+x*0x20(x=0~15)	0x0000 0000
USBHS Host Channel x Interrupt Register (USBHS_HCINTx)	0x508+x*0x20(x=0~15)	0x0000 0000
USBHS Host Channel x Interrupt Mask Register (USBHS_HCINTMSKx)	0x50c+x*0x20(x=0~15)	0x0000 0000
USBHS Host Channel x Transfer Size Register (USBHS_HCTSIZx)	0x510+x*0x20(x=0~15)	0x0000 0000
USBHS Host Channel x DMA Address Register (USBHS_HCDMAX)	0x514+x*0x20(x=0~15)	0xXXXX XXXX

(USBHS Device Control and Status Register) register name	Offset address	Reset value
USBHS Device Configuration Register (USBHS_DCFG)	0x800	0x0822 0000
USBHS Device Control Register (USBHS_DCTL)	0x804	0x0000 0002
USBHS Device Status Register (USBHS_DSTS)	0x808	0x0000 0002
USBHS device IN endpoint general interrupt mask register (USBHS_DIEPMSK)	0x810	0x0000 0000
USBHS device OUT endpoint general interrupt mask register (USBHS_DOEPMSK)	0x814	0x0000 0000
USBHS Device Entire Endpoint Interrupt Register (USBHS_DAINT)	0x818	0x0000 0000
USBHS Device All Endpoint Interrupt Mask Register (USBHS_DAINTMSK)	0x81c	0x0000 0000
USBHS Device Threshold Control Register (USBHS_DTHRCTL)	0x830	0x0c10 0020
USBHS device IN endpoint FIFO empty interrupt mask register (USBHS_DIEPEMPMSK)	0x834	0x0000 0000
USBHS Device Single Endpoint Interrupt Register (USBHS_DEACHINT)	0x838	0x0000 0000
USBHS Device Single Endpoint Interrupt Mask Register (USBHS_DEACHINTMSK)	0x83c	0x0000 0000
USBHS device IN endpoint 1 interrupt mask register (USBHS_DIEPEACHMSK1)	0x844	0x0000 0000
USBHS device OUT endpoint 1 interrupt mask register (USBHS_DOEPEACHMSK1)	0x884	0x0000 0000
USBHS Device IN Endpoint 0 Control Register (USBHS_DIEPCTL0)	0x900	0x0000 8000
USBHS Device IN Endpoint x Control Register (USBHS_DIEPCTLx)	0x900+x*0x20(x=1~15)	0x0000 0000
USBHS Device IN Endpoint x Interrupt Register (USBHS_DIEPINTx)	0x908+x*0x20(x=0~15)	0x0000 0080
USBHS Device IN Endpoint x Transfer Size Register (USBHS_DIEPTSIzX)	0x910+x*0x20(x=0~15)	0x0000 0000
USBHS Device IN Endpoint x DMA Address Register (USBHS_DIEPDMAx)	0x914+x*0x20(x=0~15)	0xXXXX XXXX
USBHS Device IN Endpoint x Transmit FIFO Status Register (USBHS_DTXFSTSx)	0x918+x*0x20(x=0~15)	0x0000 0800
USBHS Device OUT Endpoint 0 Control Register (USBHS_DOEPCTL0)	0xb00	0x0000 8000
USBHS Device OUT Endpoint x Control Register (USBHS_DOEPCTLx)	0xb00+x*0x20(x=1~15)	0x0000 0000
USBHS Device OUT Endpoint x Interrupt Register (USBHS_DOEPINTx)	0xb08+x*0x20(x=0~15)	0x0000 0000
USBHS device OUT endpoint x transfer size register (USBHS_DOEPSIZx)	0xb10+x*0x20(x=0~15)	0x0000 0000
USBHS Device OUT Endpoint x DMA Address Register (USBHS_DOEPDMAx)	0xb14+x*0x20(x=0~15)	0xFFFFFFFF

(USBHS power and clock empty control register) register name	Offset address	Reset value
USBHS Clock Gating Control Register (USBHS_GCCTL)	0xe00	0x0000 0000

34.7.1 USB System Control Register

34.7.1.1 USB System Control Register (USB_SYCTLREG)

USB System Control Register

offset address: 0x000

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	USBHS_S_NFE	USBHS_NFS[1:0]	-	-	-	-	-	USBFS_S_NFE	USBFS_NFS[1:0]		
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-	-	-	-	-	USBHS_S_FPHYE	USBHS_S_SOSEN	USBHS_S_DFB	-	-	-	-	-	USBFS_S_SOSEN	USBFS_S_DFB	

Bit	Marking	Place name	Function	Read and write
b31~b27	Reserved	-	The reset value must be maintained.	R/W
b26	USBHS_NFE	USBHS Filter Enable Register	USBHS Filter Enable Register This register is used to control the switch of the USBHS on-chip full-speed PHY DP/DM analog filter in STOP mode 0: Analog filter off 1: Analog filter is on, please refer to the setting of USBHS_NFS[1:0] for the filter range	R/W
b25~b24	USBHS_NFS	USBHS Filter Selection Register	USBHS Filter Selection Register This register is used to control the filter range of the USBHS on-chip full-speed PHY DP/DM analog filter in STOP mode 00b: Filter width gear 1 01b: Filter width gear 2 10b: Filter width gear 3 11b: Filter width gear 4 For the specific value of each gear, please refer to the chapter "Electrical Characteristics USB On-Chip Full-speed PHY STOP Mode Filter Characteristics".	R/W
b23~b19	Reserved	-	The reset value must be maintained.	R/W
b18	USBFS_NFE	USBFS Filter Enable Register	USBFS Filter Enable Register This register is used to control the switch of the USBHS on-chip full-speed PHY DP/DM analog filter in STOP mode 0: Analog filter off 1: Analog filter is on, please refer to the setting of USBFS_NFS[1:0] for the filter range	R/W
b17~b16	USBFS_NFS	USBFS filter selection register	USBFS filter selection register This register is used to control the filter range of the USBHS on-chip full-speed PHY DP/DM analog filter in STOP mode 00b: Filter width gear 1 01b: Filter width gear 2 10b: Filter width gear 3 11b: Filter width gear 4 For the specific values of each gear, please refer to the chapter "49. Electrical Characteristics 49.3.34 USB On-Chip Full-speed PHY Filtering Characteristics in STOP Mode".	R/W
b15~b11	Reserved	-	The reset value must be maintained.	R/W
b10	USBHS_FSPHYE	USBHS On-Chip Full-Speed PHY Enable Bit	USBHS On-Chip Full-Speed PHY Enable Bit 0: USBHS On-Chip Full-Speed PHY Suspended 1: USBHS on-chip full-speed PHY enable	R/W
b9	USBHS_SOSEN	USBHS SOF pulse output enable bit	When the USBHS host sends a SOF or the device successfully receives the SOF, a SOF pulse with a width of 16 system clock cycles is enabled from the PAD output 0: SOF pulse is not output 1: SOF pulse output	R/W
b8	USBHS_DFB	USBHS VBUS/ID pin internal debounce filter bypass enable bit	USBHS VBUS/ID pin module internal debounce filter bypass enable bit 0: The debounce filter inside the module is valid 1: Bypass module internal debounce filter Note: Accessible in both device mode and host mode.	R/W

b7~b2	Reserved	-	The reset value must be maintained.	R/W
b1	USBFS_SOFEN	USBFS SOF pulse output enable bit	<p>When the USBFS host sends a SOF or the device successfully receives the SOF, a 16 system clock cycle width SOF pulse is enabled from the PAD output</p> <p>0: SOF pulse is not output 1: SOF pulse output</p> <p>Note: Accessible in both device mode and host mode.</p>	R/W
b0	USBFS_DFB	USBFS VBUS/ID pin internal debounce filter bypass enable bit	<p>USBFS VBUS/ID pin module internal debounce filter bypass enable bit</p> <p>0: The debounce filter inside the module is valid 1: Bypass module internal debounce filter</p> <p>Note: Accessible in both device mode and host mode.</p>	R/W

34.7.2 USBHS Global Register

These registers are available in both host and device modes and do not need to be reprogrammed when switching between these two modes. Unless otherwise specified, bit values in register descriptions are represented in binary.

34.7.2.1 USBHS VBUS Control Register (USBHS_GVBUSCFG)

VBUS Configuration Register

offset address: 0x00

Reset value: 0x0000 0000

This register can be used to set the VBUS value to ignore the state 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								VBUS VAL	VBUS OVEN	Reserved							
<hr/>																	
Bit	Marking	Place name	Function								Read and write						
b31~b8	Reserved	-	The reset value must be maintained.								R/W						
B7	VBUSVAL	VBUS value	VBUS Value It is used to set the VBUS value of USBFS. When it is set to 1 and VBUSOVEN is set to 1, the USBFS is powered on. Note: Only accessible in device mode.								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 USBHS CORE. The value of VBUSVAL is valid only when this bit is set to 1. Note: Only accessible in device mode.								R/W						
b5~b0	Reserved	-	The reset value must be maintained.								R/W						

34.7.2.2 USBHS AHB Control Register (USBHS_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 role modes.
This register mainly contains configuration parameters related to the AHB system.

The application must program this register before starting any AHB or USB transaction. 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															
Bit	Marking	Place name	Function										Read and write		
b31~b9	Reserved	-	The reset value must be maintained.										R/W		
b8	PTXFELVL	Periodic TxFIFO Empty Threshold	Periodic TxFIFO empty level Indicates when the Periodic TxFIFO Empty Interrupt bit in the Module Interrupt Register (PTXFE bit in USBHS_GINTSTS) is triggered. 0: PTXFE (located in USBHS_GINTSTS) interrupt indicates that the periodic TxFIFO is half empty 1: PTXFE (located in USBHS_GINTSTS) interrupt indicates that the periodic TxFIFO is completely empty Note: Accessible only in host mode.										R/W		
B7	TXFELVL	Device TxFIFO Empty Threshold	Device TxFIFO empty threshold (Tx FIFO empty level) In device mode, this bit indicates when the IN endpoint transmit FIFO empty interrupt (TXFE in USBHS_DIEPINTx) is triggered. 0: TXFE (located in USBHS_DIEPINTx) interrupt indicates that the IN endpoint TxFIFO is half empty 1: TXFE (located in USBHS_DIEPINTx) interrupt indicates that the IN endpoint TxFIFO is completely empty Note: Only accessible in device mode.										R/W		
b6	Reserved	-	The reset value must be maintained.										R/W		
b5	DMAEN	DMA enable	DMA enable (DMA enable) 0: The module operates in slave mode 1: The 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: Keep Note: Accessible in both device mode and host mode.										R/W		
b0	GINTMSK	global interrupt mask	Global interrupt mask This bit is used to mask or unmask global interrupts. The Interrupt Status Register is updated by the module regardless of the setting of this bit. 0: Block interrupts triggered by the application 1: Unmask the interrupt triggered by the application Note: Accessible in both device mode and host mode.										R/W		

34.7.2.3 USBHS USB Configuration Register (USBHS_GUSBCFG)

USBHS USB configuration register

offset address: 0x00C

Reset value: 0x0000 1410

This register can be used to configure the module after power-up or changing role mode. It contains configuration parameters related to USB and USB-PHY.

The application must program this register before starting any AHB or USB transaction. 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	FDMOD	FHMOD	Reserved			ULPIPD	PTCI	PCCI	Reserved	ULPIEVBUSI	ULPIEVBUSD	ULPICSM	ULPIAR	ULFSLS	Reserved			
b15	b14	b13	b12	b11	b10	b9	8	B7	b6	b5	b4	b3	b2	b1	b0			
PHYLPCS	Reserved	TRDT[3:0]			Reserved			PHYSEL	Reserved			TOCAL[2:0]						
<hr/>																		
Bit	Marking	Place name			Function								Read and write					
b31	Reserved	-			The reset value must be maintained.								R/W					
b30	FDMOD	Force device mode			Force device mode Writing a 1 to this bit forces the module into device mode, ignoring the state of the USBHS_ID input pin. 0: Normal mode, depending on the input state of the USBHS_ID pin 1: Force device mode After setting the force bit, 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	Force host mode			Force host mode Writing a 1 to this bit forces the module to host mode, ignoring the state of the USBHS_ID input pin. 0: Normal mode, depending on the input state of the USBHS_ID pin 1: Force host mode After setting the force bit, 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~b26	Reserved	-			The reset value must be maintained.								R/W					
b25	ULPIPD	ULPI interface protection prohibited			ULPI Interface protect disable This bit controls the circuitry built into the PHY to protect the ULPI interface. Any pull-up or pull-down resistors used by this function can be disabled. Please refer to the ULPI specification for details. 0: Enable interface protection circuit 1: Disable the interface protection circuit								R/W					
b24	PTCI	by the indicator -			Indicator pass through This bit controls whether the complementary output needs to pass through the internal V BUS Active level comparator function before it can be used in RX CMD V BUS state. Please refer to the ULPI specification for details. 0: Complementary output signal verified by internal VBUS active level comparator 1: Complementary output signal does not pass the verification of the internal VBUS active level comparator								R/W					
b23	PCCI	Complementary indicator -			Indicator complement This bit controls the PHY to invert the ExternalVbusIndicator input signal and generate a complementary output. See the ULPI specification for details. 0: PHY does not invert the ExternalVbusIndicator signal 1: PHY inverts the ExternalVbusIndicator signal								R/W					
b22	Reserved	-			The reset value must be maintained.								R/W					

b21	ULPIEVBUS I	ULPI External VBUS Indicator -	ULPI external VBUS indicator This bit instructs the ULPI PHY to use an external VBUS overcurrent indicator. 0: PHY uses internal VBUS valid comparator 1: PHY uses external VBUS valid comparator	R/W
b20	ULPIEVBUS D	ULPI external VBUS driver	ULPI External VBUS Drive (ULPI External VBUS Drive) This bit selects the ULPI PHY to use the internal or external power supply to drive 5 V on VBUS. 0: PHY drives VBUS with internal charge pump (default) 1: The PHY uses an external power supply to drive VBUS.	R/W
b19	ULPICSM	ULPI clock hang	ULPI Clock SuspendM (ULPI Clock SuspendM) This bit sets the ClockSuspendM bit in the ULPI PHY Interface Control Register. This bit is only available in serial mode. 0: PHY powers down internal clock during suspend 1: PHY does not power down the internal clock	R/W
b18	ULPIAR	ULPI clock automatic recovery	ULPI Auto-resume This bit sets the AutoResume bit in the ULPI PHY Interface Control Register. 0: PHY does not support auto-recovery function 1: The PHY uses the auto-recovery function	R/W
b17	ULFSLS	ULPI FS/LS option	ULPI FS/LS select (ULPI FS/LS select) The application uses this bit to select the FS/LS serial interface for the ULPI PHY. This bit is only valid if the FS serial transceiver is selected on the ULPI PHY. 0: ULPI interface 1: ULPI FS/LS serial interface	R/W
b16	Reserved	-	The reset value must be maintained.	R/W
b15	PHYLPCS	ULPI FS/LS option	PHY Low-power clock select This bit is used to select 480 MHz or 48 MHz (low power) PHY mode. In FS and LS modes, the PHY can typically run with a 48 MHz clock, saving power. 0: 480 MHz internal PLL clock 1: 48 MHz external clock In 480 MHz mode, the UTMI interface runs at 60 MHz or 30 MHz, depending on whether 8-bit or 16-bit data width is selected. In 48 MHz mode, the UTMI interface operates at 48 MHz in FS and LS modes.	R/W
b14	Reserved	-	The reset value must be maintained.	R/W
b13~b10	TRDT	USB turnaround time	USB turnaround time Set the turnaround time in units of PHY clocks. To calculate the value of TRDT, use the following formula: $\text{TRDT} = 4 \times \text{AHB clock} + 1 \text{ PHY clock}$ E.g: 1. If AHB clock frequency = 84 MHz (PHY clock frequency = 48 MHz), then TRDT is set to 9. 2. If AHB clock frequency = 48 MHz (PHY clock frequency = 48 MHz), then TRDT is set to 5. Note: Only accessible in device mode.	R/W
b9~b7	Reserved	-	The reset value must be maintained.	R/W
b6	PHYSSEL	USB 2.0 high-speed ULPI PHY or USB 1.1 full-speed serial transceiver selection	USB 2.0 high-speed ULPI PHY or USB 1.1 full-speed serial transceiver select 0: USB 2.0 high-speed ULPI PHY 1: USB on-chip full-speed serial transceiver	R/W
b5~b3	Reserved	-	The reset value must be maintained.	R/W
b2~b0	TOCAL	FS Timeout Calibration	FS timeout calibration Additional latency introduced by the PHY includes the number of PHY clocks set by the application in this field, and the module's full-speed inter-packet timeout interval. The delay introduced by different PHYs affects the state of the data line differently. The USB standard timeout value for full-speed operation is 16 to 18 (inclusive) bit times. The application must program this field according to the enumeration speed. The number of bit times added per PHY clock is 0.25 bit times. Note: Accessible in both device mode and host mode.	R/W

34.7.2.4 USBHS Reset Register (USBHS_GRSTCTL)

USBHS reset register

offset address: 0x10

Reset value: 0x8000 0000

The application program resets various hardware features in the module through this register.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16					
AHBI DL	DMA REQ	Reserved																		
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0					
Reserved					TXFNUM[4:0]					TXFF LSH	RXFF FLSH	Reser ved	FCRS T	HSRS T	CSRS T					
<hr/>																				
Bit	Marking	Place name	Function												Read and write					
b31	AHBIDL	AHB master is idle	AHB master idle Indicates that the AHB master state machine is in an idle condition. Note: Accessible in both device mode and host mode.												R					
b30	DMAREQ	AHB master is idle	DMA request signal This bit indicates that a DMA request is in progress. for debugging. Note: Accessible in both device mode and host mode.												R					
b30~b11	Reserved	-	Read as "0", write as "0"												R/W					
b10~b6	TXFNUM	TxFIFO number	TxFIFO number FIFO number for FIFO flushing using the TxFIFO flush bits. This field can only be changed after the module clears the TxFIFO flush bit. <ul style="list-style-type: none"> ● 00000: — Flush aperiodic TxFIFO in master mode — Flush Tx FIFO 0 in device mode ● 00001: — Flush periodic TxFIFO in master mode — Flush TXFIFO 1 in device mode ● 00010: refresh TXFIFO 2 in device mode ... ● 00101: refresh TXFIFO 15 in device mode ● 10000: flush all transmit FIFOs in device mode or host mode Note: Accessible in both device mode and host mode.												R/W					
b5	TXFFLSH	TxFIFO refresh	TxFIFO flush (TxFIFO flush) This bit selectively flushes one or all transmit FIFOs, but cannot do so while the module is processing a communication transaction. The application can only write to this bit after verifying that the module is not currently reading or writing to the TxFIFO. Use the following registers to confirm: <ul style="list-style-type: none"> - read: NAK valid interrupt ensures that the module is not currently performing a read of the FIFO - Write: The AHBIDL bit in USBHS_GRSTCTL ensures that the module is not currently doing any writes to the FIFO Note: Accessible in both device mode and host mode.												R/W					
b4	RXFFFLSH	RxFIFO flush	RxFIFO flush (RxFIFO flush) An application can use this bit to flush the entire RxFIFO, but must first ensure that the module is not currently processing a communication transaction. The application can write to this bit only after confirming that the module is not currently reading or writing to the RxFIFO. The application must wait until this bit is cleared before performing other operations. It usually takes 8 clock cycles to wait (whichever is the slowest of the PHY or AHB clock). Note: Accessible in both device mode and host mode.												R/W					
b3	Reserved	-	The reset value must be maintained.												R/W					
b2	FCRST	Host frame counter reset	Host frame counter reset When the application program writes to this bit, the frame counter in the module is reset. 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					

			HCLK domain logic soft reset (HCLK soft reset) Applications use this bit to refresh control logic in the AHB clock domain. Only the AHB clock domain pipeline is reset. The FIFO is not refreshed by this bit. All state machines in the AHB clock domain are reset to the idle state after the transaction on the AHB is terminated according to the protocol. The CSR control bit used by the AHB clock domain state machine is cleared. To clear this interrupt, the status mask bit generated by the AHB clock domain state machine and used to control the state of the interrupt needs to be cleared. Since the interrupt status bit is not cleared, the application can get the state. This bit is self-clearing and will be cleared by the module after all necessary logic in it is 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
b1	HSRST	HCLK domain logic soft reset	Module soft reset (Core soft reset) Reset the HCLK and PCLK domains as follows: Clears individual interrupts and all CSR register bits except: — GATEHCLK bit in USBHS_GCCTL — STPPCLK bit in USBHS_GCCTL — FLSSPCS bit in USBHS_HCFG — DSPD bit in USBHS_DCFG Resets all module state machines (except AHB slaves) to idle state and clears all transmit and receive FIFOs. Terminate all transactions on the AHB master as soon as possible after the last data phase of an AHB transfer. Immediately terminate all transactions on the USB. The application can write to this bit at any time when the module needs to be reset. This bit is self-clearing and the module will clear this bit after all necessary logic in it is reset, a process that takes several clocks depending on the current state of the module. Once this bit is cleared, software must wait at least 3 PHY clocks before accessing the PHY domain (synchronization delay). In addition, software It must also be determined that Bit 31 in this register is set (AHB master is idle) before running. Software reset is usually used in two situations, one is during software development, and the other is after the user has dynamically changed the PHY select bits in the USB configuration registers listed above. When the user changes the PHY, the corresponding clock will be selected for the PHY and used in the PHY domain. Once a new clock is selected, the PHY domain must be reset for proper operation. Note: Accessible in both device mode and host mode.	R/W
b0	CSRST	Module soft reset		

34.7.2.5 USBHS Global Interrupt Status Register (USBHS_GINTSTS)

USBHS interrupt status register

offset address: 0x14

Reset value: 0x14000020

This register is used to interrupt the application with system level events in the current mode (device mode or host mode).

Some bits in this register are valid only in host mode, while other bits are valid only in device mode. In addition, this register can also indicate the current mode.

FIFO status interrupts are read-only; if software reads or writes to the FIFO while these interrupts are being handled, the FIFO interrupt flag is automatically cleared.

Before enabling the interrupt bit, the application must clear the USBHS_GINTSTS register during initialization to avoid any interrupts before initialization.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
WKUINT	VBUSVINT	DISCINT	CIDSCHG	LPMI NT	PTXF E	HCINT	HPRTINT	Reser ved	DATAFSUSP	IPXFR / INCO MPIS OOUT	IISOIXFR	OEPI NT	IEPIN T	Reser ved	Reser ved
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
EOPF	ISOODRP	ENUMDNE	USBRSST	USBSSUSP	ESUSP	Reser ved	Reser ved	GONAKEFF	GINAKEFF	NPTXFE	RXFN E	SOF	Reser ved	MMIS	CMOD

Bit	Marking	Place name	Function	Read and write
b31	WKUINT	Resume/remote wakeup interrupt detected	Resume/remote wakeup detected interrupt In device mode, this interrupt is triggered when a resume signal is detected on the USB bus. In host mode, this interrupt is triggered when a remote wakeup is detected on the USB. This bit is cleared by software by writing a 1 to it. Note: Accessible in both device mode and host mode.	R/W
b30	VBUSVINT	VBUS active interrupt	VBUS valid interrupt In device mode, this interrupt will be triggered when USBHS_VBUS pin is detected from low to high. This bit is cleared by software by writing a 1 to it. Note: Only accessible in device mode.	R/W
b29	DISCINT	Disconnect interruption detected	Disconnect detected interrupt This interrupt is triggered when a device disconnection is detected. This bit is cleared by software by writing a 1 to it. Note: Accessible only in host mode.	R/W
b28	CIDSCHG	Connector ID line state change interrupt	Connector ID status change The module sets this bit when the connector ID line state changes. This bit is cleared by software by writing a 1 to it. Note: Accessible in both device mode and host mode.	R/W
b27	LPMINT	LPM interrupt	LPM interrupt device mode This bit is set when the device has received a correct LPM transmission and a valid reply. Host mode This bit is set to 1 when the host receives a valid response after sending LPM, or exceeds the set number of retries.	R/W

			Periodic TxFIFO empty interrupt This interrupt is triggered when the periodic transmit FIFO is half empty or completely empty and there is room in the periodic request queue to write at least one entry. Whether the FIFO is half empty or fully empty is determined by the Periodic TxFIFO Empty Level bit in the USBHS_GAHBCFG register (PTXFELVL bit in USBHS_GAHBCFG). Note: Accessible only in host mode.	R
b26	PTXFE	Periodic TxFIFO Empty Interrupt	Host channels interrupt When the module sets this bit, it indicates that there is a pending interrupt (in host mode) on a channel in the module. The application must read the host USBHS_HAINT register to determine the exact channel number on which the interrupt occurred, and then read the corresponding USBHS_HCINTx registers to determine the exact cause of the interrupt. The application must clear the appropriate status bit in the USBHS_HCINTx register before clearing this bit. Note: Accessible only in host mode.	R
b24	HPRTINT	Host port interrupt	Host port interrupt When the module sets this bit to 1, it indicates a change in the state of the USBHS controller port in host mode. The application must read the USBHS_HPRT register to determine the exact event that caused this interrupt. The application must clear the appropriate status bit in the USBHS_HPRT register before clearing this bit. Note: Accessible only in host mode.	R
b23	Reserved	-	The reset value must be maintained.	R/W
b22	DATAFSUSP	Data fetch pending	Data fetch suspended This interrupt is only valid 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 detecting an endpoint mismatch, the application would do the following: — Set the global aperiodic IN NAK handshake signal to 1 — Disable IN endpoints — Empty FIFO — Determine the token sequence from the IN token sequence learning queue — re-enable endpoint — If the global aperiodic IN NAK is cleared but the module has not yet acquired data for the IN endpoint and an IN token has been received, clear the global aperiodic IN NAK handshake signal: The module will generate an "IN token received while FIFO is empty" interrupt. USBHS then sends a NAK response to the host. To avoid this from happening, the application can check for the DATAFSUSP interrupt in USBHS_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 clearing the global IN NAK handshake signal. This bit is cleared by software by writing a 1 to it. Note: Only accessible in device mode.	R/W
b21	IPXFR/ INCOMPISOOUT	Incomplete periodic transmission/ OUT sync transfer not completed	IPXFR: Incomplete periodic transfer In master mode, the module sets this interrupt bit if there are outstanding periodic transactions still pending that are scheduled to complete during the current frame. This bit is cleared by software by writing a 1 to it. Note: Accessible only in host mode. INCOMPISOOUT: Incomplete isochronous OUT transfer In device mode, when the module sets this interrupt, it indicates that the transfer on at least one isochronous OUT endpoint in the current frame is not complete. This interrupt is triggered along with the Periodic End of Frame Interrupt (EOPF) bit in this register. This bit is cleared by software by writing a 1 to it. Note: Only accessible in device mode.	R/W
b20	IISOIXFR	IN SYNC TRANSFER NOT COMPLETED	Incomplete isochronous IN transfer When the module sets this interrupt to 1, it indicates that there is an incomplete transfer on at least one synchronous IN endpoint in the current frame. This interrupt is triggered along with the Periodic End of Frame Interrupt (EOPF) bit in this register. This bit is cleared by software by writing a 1 to it. Note: Only accessible in device mode.	R/W

b19	OEPINT	OUT endpoint interrupt	<p>OUT endpoint interrupt When the module sets this bit, it indicates that there is a pending interrupt (in device mode) on one of the OUT endpoints in the module. The application must read the host USBHS_DAINT register to determine the exact number of the OUT endpoint on which the interrupt occurred, and then read the corresponding USBHS_DOEPIXTx registers to determine the exact cause of the interrupt. The application must clear the corresponding status bit in the corresponding USBHS_DOEPIXTx register before clearing this bit.</p> <p>Note: Only accessible in device mode.</p>	R
b18	IEPINT	IN endpoint interrupt	<p>IN endpoint interrupt When the module sets this bit, it indicates that there is a pending interrupt (in device mode) on one of the IN endpoints in the module. The application must read the host USBHS_DAINT register to determine the exact number of the IN endpoint where the interrupt occurred, and then read the corresponding USBHS_DIEPIXTx registers to determine the exact cause of the interrupt. The application must first clear the corresponding status bit in the USBHS_DIEPIXTx register before this bit can be cleared.</p> <p>Note: Only accessible in device mode.</p>	R
b17~b16	Reserved	-	The reset value must be maintained.	R/W
b15	EOPF	Periodic end of frame interrupt	<p>End of periodic frame interrupt Indicates that the current frame has reached the period specified by the Periodic Frame Interval field in the USBHS_DCFG register (the PFIVL bit in USBHS_DCFG). This bit is cleared by software by writing a 1 to it.</p> <p>Note: Only accessible in device mode.</p>	R/W
b14	ISOODRP	Drop sync OUT packet interrupt	<p>Isochronous OUT packet dropped interrupt If the module cannot write the synchronous OUT data packet to the RxFIFO due to insufficient space in the RxFIFO to accommodate the maximum data packet of the synchronous OUT endpoint, the module will set this bit to 1. This bit is cleared by software by writing a 1 to it.</p> <p>Note: Only accessible in device mode.</p>	R/W
b13	ENUMDNE	Enumeration complete interrupt	<p>Enumeration done interrupt When the module sets this bit, it indicates that the speed enumeration is complete. The application must read the USBHS_DSTS register to get the enumeration speed. This bit is cleared by software by writing a 1 to it.</p> <p>Note: Only accessible in device mode.</p>	R/W
b12	USBRST	USB reset interrupt	<p>USB reset interrupt When the module sets this bit to 1, it indicates that a reset signal was detected on the USB. This bit is cleared by software by writing a 1 to it.</p> <p>Note: Only accessible in device mode.</p>	R/W
b11	USBSUSP	USB suspend interrupt	<p>USB suspend interrupt When the module sets this bit, it indicates that a suspend state was detected on the USB. When the idle state on the USB bus remains for 3ms, the module will enter the suspend state. This bit is cleared by software by writing a 1 to it.</p> <p>Note: Only accessible in device mode.</p>	R/W
b10	ESUSP	early pending interrupt	<p>Early suspend interrupt When the module sets this bit to 1, it indicates that the USB has been detected to be in an idle state for 3ms. Note: Only accessible in device mode.</p>	R/W
b9~b8	Reserved	-	The reset value must be maintained.	R/W
B7	GONAKEFF	Global OUT NAK valid interrupt	<p>Global OUT NAK effective interrupt Indicates that the "set global OUT NAK" bit in the USBHS_DCTL register set by the application (SGONAK bit in USBHS_DCTL) has taken effect in the module. This bit can be cleared by writing to the "Clear Global OUT NAK" bit in the USBHS_DCTL register (CGONAK bit in USBHS_DCTL).</p> <p>Note: Only accessible in device mode.</p>	R
b6	GINAKEFF	Global aperiodic IN NAK valid interrupt	<p>Global IN nonperiodic NAK effective interrupt Indicates that the "set global aperiodic IN NAK" bit in the USBHS_DCTL register set by the application (SGINAK bit in USBHS_DCTL) has taken effect in the module. That is, the module has sampled the global IN NAK bit set by the application and the result has taken effect. This bit can be cleared by clearing the "Clear Global Aperiodic IN NAK" bit in the USBHS_DCTL register (CGINAK bit in UBSFS_DCTL). 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.</p> <p>Note: Only accessible in device mode.</p>	R

b5	NPTXFE	Aperiodic TxFIFO Empty Interrupt	<p>Non-periodic TxFIFO empty interrupt This interrupt is triggered when the aperiodic TxFIFO is half-empty or completely empty, and there is at least one entry available in the aperiodic send request queue. Whether the FIFO is half empty or fully empty is determined by the Aperiodic TxFIFO Empty Level bit in the USBHS_GAHBCFG register (TXFELVL bit in USBHS_GAHBCFG).</p> <p>Note: Accessible only in host mode.</p>	R
b4	RXFNE	RxFIFO not empty interrupt	<p>RxFIFO non-empty interrupt Indicates that there is at least one packet in the Rx FIFO waiting to be read.</p> <p>Note: Accessible in both host mode and device mode.</p>	R
b3	SOF	start of frame interrupt	<p>Start of frame interrupt In host mode, when the module sets this bit, it indicates that a SOF (FS) or Keep-Alive (LS) signal has been sent on the USB. The application must set this bit to clear this interrupt.</p> <p>In device mode, when the module sets this bit, it indicates that a SOF token has been received on the USB. The application can obtain the current frame number by reading the device status register. This interrupt only occurs when the module is running in FS mode.</p> <p>This bit is cleared by software by writing a 1 to it.</p> <p>Note: Accessible in both host mode and device mode.</p>	R/W
b2	Reserved	-	The reset value must be maintained.	R/W
b1	MMIS	pattern mismatch interrupt	<p>Mode mismatch interrupt The module sets this bit when the application attempts to access the following registers:</p> <ul style="list-style-type: none"> — The module operates in device mode to access the host mode registers — Module running in host mode to access device mode registers <p>A 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>This bit is cleared by software by writing a 1 to it.</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

34.7.2.6 USBHS Global Interrupt Mask Register (USBHS_GINTMSK)

USBHS interrupt mask register

offset address: 0x18

Reset value: 0x00000000

This register is used in conjunction with the module interrupt register to interrupt the application program. If an interrupt bit is masked, the interrupt associated with that bit will not be generated.

However, the module interrupt (USBHS_GINTSTS) register bit corresponding to this interrupt is still set.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
WKUIM	VBUSVIM	DISCM	CIDSCHGM	LPMIEM	PTXFEM	HCIM	HPRTIM	Reserved	DATAFSUSPM	IPXFRM/INCO	IISOIXFRM	OEPI	IEPIM	Reserved	Reserved
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
EOPFM	ISODRPM	ENUDNEM	USBRSTM	USBSUSPM	ESUSPM	Reserved	Reserved	GONAKEFFFM	GINAKEFFM	NPTXFEM	RXFNEM	SOFM	Reserved	MMISM	Reserved

Bit	Marking	Place name	Function	Read and write
b31	WKUIM	Resume/Remote Wakeup Interrupt Masking Detected	Resume/remote wakeup detected interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in both host mode and device mode.	R/W
b30	VBUCSVIM	VBUS active interrupt mask	VBUS valid interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b29	DISCM	Disconnect detected interrupt mask	Disconnect detected interrupt interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible only in host mode.	R/W
b28	CIDSCHGM	Interrupt Connector ID Line State 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	LPMINTM	LPM interrupt masking	LPM interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in both device mode and host mode.	R/W
b26	PTXFEM	Periodic TxFIFO Empty Interrupt Mask	Periodic TxFIFO empty interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible only in host mode.	R/W
b25	HCIM	Host channel interrupt mask	Host channels interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible only in host mode.	R/W
b24	HPRTIM	Host Port Interrupt Mask	Host port interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible only in host mode.	R/W
b23	Reserved	-	The reset value must be maintained.	R/W
b22	DATAFSUSPM	Data fetch pending interrupt mask	Data fetch suspended interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W

			IPXFR: Incomplete periodic transfer interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible only in host mode.	
b21	IPXFRM/ INCOMPISOOUTM	Incomplete periodic transmission interrupt mask/ Incomplete OUT synchronous transfer interrupt mask	INCOMPISOOUT: Incomplete isochronous OUT transfer interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b20	IISOIXFRM	Incomplete IN isochronous transfer interrupt mask	Incomplete isochronous IN transfer interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b19	OEPIM	OUT endpoint interrupt mask	OUT endpoint interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b18	IEPIM	IN endpoint interrupt mask	IN endpoint interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b17~b16	Reserved	-	The reset value must be maintained.	R/W
b15	EOPFM	Periodic end of frame interrupt mask	End of periodic frame interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b14	ISOODRPM	Drop sync OUT packet interrupt mask	Isochronous OUT packet dropped interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b13	ENUMDNEM	Enumeration complete interrupt mask	Enumeration done interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b12	USBRSTM	USB reset interrupt mask	USB reset interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b11	USBSUSPM	USB suspend interrupt mask	USB suspend interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b10	ESUSPM	early pending interrupt mask	Early suspend interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b9~b8	Reserved	-	The reset value must be maintained.	R/W
B7	GONAKEFFM	Global OUT NAK active interrupt mask	Global OUT NAK effective interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b6	GINAKEFFM	Global Aperiodic IN NAK Active Interrupt Mask	Global IN nonperiodic NAK effective interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b5	NPTXFEM	Aperiodic TxFIFO Empty Interrupt Mask	Non-periodic TxFIFO empty interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible only in host mode.	R/W
b4	RXFNEM	RxFIFO non-empty interrupt mask	RxFIFO non-empty interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in both host mode and device mode.	R/W
b3	SOFM	start of frame 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/W
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/W

34.7.2.7 USBHS Receive Status Debug Read/USBHS Status Read and Pop Registers (USBHS_GRXSTS/USBHS_GRXSTSP)

USBHS Receive status debug read/USBHS status read and pop registers

Read offset address: 0x01C

Offset address to pop from the stack: 0x020

Reset value: 0x0000 0000

Reading the receive status debug read register will return the contents of the top of the receive FIFO. Reading the receive status read and pop registers will additionally pop the data entry at the top of the RxFIFO. Receive status content is interpreted differently in host mode and device mode.

When the receive FIFO is empty, the module ignores the read or pop operation of this register and returns the value 0x0000 0000. The application must only pop the Receive Status FIFO when the Receive FIFO Not Empty bit of the Module Interrupt Register (RXFNE bit in USBHS_GINTSTS) is set.

Host 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
DPID[0]	BCNT[10:0]										CHNUM[3:0]				

Bit	Marking	Place name	Function	Read and write
b31~b21	Reserved	-	The reset value must be maintained.	R/W
b20~b17	PKTSTS	packet status	Packet status Indicates the status of received packets 0010: IN packet received 0011: IN transfer completed (trigger interrupt) 0101: Data synchronization error (trigger interrupt) 0111: Pause the channel (trigger interrupt) Other values: Keep	R
b16~b15	DPID	data PID	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
DPID[0]	BCNT[11:0]										EPNUM[3:0]				
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b21	Reserved	-	The reset value must be maintained.										R/W		
b20~b17	PKTSTS	packet status	Packet status Indicates the status of received packets 0001: Global OUT NAK (triggered interrupt) 0010: OUT packet received 0011: OUT transfer completed (triggered interrupt) 0100: SETUP transaction completed (interrupt triggered) 0110: SETUP packet received Other values: Keep										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		

34.7.2.8 USBHS Receive FIFO Size Register (USBHS_GRXFSIZ)

USBHS Receive FIFO size register

offset address: 0x024

Reset value: 0x0000 0800

This application can program the amount 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[11:0]											

Bit	Marking	Place name	Function	Read and write
b31~b12	Reserved	-	The reset value must be maintained.	R/W
b11~b0	RXFD	RxFIFO depth	RXFD: RxFIFO depth In 32-bit words. The minimum value is 16 The maximum value is 2048 The power-on reset value is the maximum Rx data FIFO depth.	R/W

34.7.2.9 USBHS Host Aperiodic Transmit FIFO Size Register (USBHS_HNPTXFSIZ)/Endpoint 0 Transmit FIFO Size (USBHS_DIEPTXF0)

USBHS Host non-periodic transmit FIFO size register/Device endpoint0 transmit FIFO size register
offset address: 0x028

Reset value: 0x08000800

This application can program the amount 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
NPTXFSA[15:0]/TX0FSA[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	NPTXFD/ TX0FD	Aperiodic TxFIFO Depth/Endpoint 0 TxFIFO Depth	Host mode: NPTXFD Non-periodic TxFIFO depth In 32-bit words. The minimum value is 16 The maximum value is 2048 Device Mode: TX0FD Endpoint 0 TxFIFO depth In 32-bit words. The minimum value is 16 The maximum value is 2048	R/W											
b15~b0	NPTXFSA/ TX0FSA	Aperiodic transmit RAM start address / Endpoint 0 transmit RAM start address	Host mode: NPTXFSA Non-periodic transmit RAM start address This field contains the memory start address of the aperiodic transmit FIFO RAM. Device Mode: TX0FSA Endpoint 0 transmit RAM start address This field contains the memory start address of the endpoint 0 transmit FIFO RAM.	R/W											

34.7.2.10 USBHS Aperiodic Transmit FIFO/Queue Status Register (USBHS_HNPTXSTS)

USBHS Host non-periodic transmit FIFO size register/Device endpoint0 transmit FIFO size register
offset address: 0x02C

Reset value: 0x00080800

This read-only register contains free space information for the aperiodic TxFIFO and the aperiodic transmit request queue.

This register is only valid in host mode, not in device mode.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reser ved	NPTXQTOP[6:0]										NPTQXSAR[7:0]				
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
NPTXFSAV[15:0]															

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	The reset value must be maintained.	R/W
b30~b24	NPTXQTOP	Aperiodic send request queue top	Top of the non-periodic transmit request queue The entry currently being processed by the MAC in the aperiodic send request queue. Bits 30:27: Channel/endpoint number Bit 26:25: — 00: IN/OUT token — 01: Transmit packet of length zero — 11: Channel stop command Bit 24: Terminate (last entry for selected channel)	R
b23~b16	NPTQXSAR	Aperiodic send request queue free space	Aperiodic send request queue free space (Non-periodic transmit request queue space available) Indicates the amount of free space available in the aperiodic send request queue. In host mode, this queue holds IN and OUT requests. 0: The aperiodic send request queue is full 1:1 position available 2: 2 positions available n: n positions available (where n ranges: 0~8) Other values: Keep	R
b15~b0	NPTXFSAV	Aperiodic TxFIFO free space	Non-periodic TxFIFO space available Indicates the amount of free space available in the aperiodic TxFIFO. In 32-bit words. 0: Aperiodic TxFIFO is full 1:1 word available 2: 2 words available n: n words available (where n ranges: 0~2048) Other values: Keep	R

34.7.2.11 USBHS Module ID Register (USBHS_CID)

USBHS core ID register

offset address: 0x03C

Reset value: 0x12345678

This register is the programmable user configuration ID 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]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b0	PRODUCT_ID	Product ID field	Product ID field Application-programmable ID field.										R/W		

34.7.2.12 USBHS LPM Configuration Register (USBHS_GLPMCFG)

USBHS LPM configuration register

offset address: 0x054

Reset value: 0x00000000

This register is a programmable LPM configuration register.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16																								
Reserved		ENBS EL	LPMRCNTSTS[2:0]			SNDL PM	LPMRCNT[2:0]			LPMCHIDX[3:0]				L1RS MOK																									
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0																								
SLPS TS	LPMRSP[1:0]		L1DS EN	BSELTHRS[3:0]				L1SS EN	REM WAKE	BSEL[3:0]			LPMA CK	LPME N																									
<hr/>																																							
Bit	Marking		Place name		Function									Read and write																									
b31~b29	Reserved		-		The reset value must be maintained.									R/W																									
b28	ENBSEL	BSEL enable bit		BSEL enable bit 0: Follow USB 2.0 Link Power Management Addendum Engineering Change Notice to the USB 2.0 specification, July 16, 2007 1: Follow Errata for USB 2.0 ECN: Link Power Management (LPM) - 7/2007											R/W																								
b27~b25	LPMRCNTSTS [2:0]	LPM Retry Status Register	LPM Retry Status Register Record the number of retries remaining for LPM, only valid in host mode													R																							
b24	SENDLPM	Send LPM transport	Send LPM transport 0: Do not send LPM transfers 1: Send LPM transfers (EXT and LPM tokens) This bit is automatically cleared when an acknowledgement (ACK, STALL or NYET) is received from the device. Only supported in host mode.													R/W																							
b23~b21	LPMRCNT[2:0]	LPM retry count register	LPM retry count register When an ERROR response is received, the host sends the number of LPM transmission retries until a valid response (ACK, STALL or NYET) is received Only supported in host mode.													R/W																							
b20~b17	LPMCHIDX[2: 0]	LPM transmit channel index register	LPM transmit channel index register The host channel index used for LPM transmission, the hardware automatically inserts the device address and endpoint information set by the channel. Only supported in host mode.													R/W																							
b16	L1RSMOK	L1 Status Bit	L1 Status Bit 0: The current state cannot be recovered from L1 1: The current state can be restored from L1													R																							
b15	SLPSTS	sleep status bit	sleep status bit device mode When the device sends ACK in response to LPM transmission and the time specified by the protocol has passed, this bit is set to 1, indicating that it enters the SLEEP mode. When the bus state changes, or disconnects, or sends a remote wake-up signal, this bit is automatically cleared to 0 and exits SLEEP mode. Host mode When the host sends an LPM transmission and receives an ACK response from the device, this bit is set to 1, indicating that it enters the SLEEP mode. When the host receives a remote wake-up signal, or the host initiates a wake-up, or the host initiates a reset, this bit is automatically cleared to 0 to exit the SLEEP mode. 0: Not in L1 1: In L1													R																							
b14~b13	LPMRSP[1:0]	LPM response	LPM response The LPM response received by the master or sent by the slave 00b: ERROR (no response) 01b: STALL 00b: NYET 00b: ACK													R																							

b12	L1DSEN	L1 deep sleep enable bit	L1 deep sleep enable bit To minimize power consumption, the application needs to set this bit to 1.	R/W
b11~b8	BSELTHRS[3:0]	BSEL Threshold Setting Register	BSEL Threshold Setting Register device mode When the device receives that the BSEL value is greater than or equal to the setting of this register, it enters L1. Host mode The time when the host sends the resume signal.	R/W
B7	L1SEN	L1 shallow sleep enable bit	L1 shallow sleep enable bit To minimize power consumption, the application needs to set this bit to 1.	R/W
b6	REMWAKE	bRemoteWake value	bRemoteWake value bRemoteWake value sent by the host or received by the device BSEL register device mode The BSEL bmAttribute value is automatically updated after a valid response to LPM transmission. Host mode The host sends the BSEL value of the LPM transfer, or the length of the resume signal.	R/W
b5~b2	BSEL[3:0]	BSEL register	0000b: 125us 0001b: 150us 0010b: 200us 0100b: 400us 0101b: 500us 0110b: 1000us 0111b: 2000us 1000b: 3000us 1001b: 4000us 1010b: 5000us 1011b: 6000us 1100b: 7000us 1101b: 8000us 1110b: 9000us 1111b: 10000us	R/W
b1	LPMACK	LPM Reply Register	LPM Reply Register 1: ACK Even if this bit is written to 1, an ACK will not be sent when there is an error in the LPM transmission. PID/CRC error exists, ERROR response bLinkState is not equal to 0001b, STALL responds There is a pending data transfer, NYET acknowledges 0: NYET Only device mode works	R/W
b0	LPMEN	LPM enable bit	LPM enable bit 0: LPM disabled 1: LPM enable	R/W

34.7.2.13 USBHS host periodic transmit FIFO size register (USBHS_HPTXFSIZ)

USBHS Host periodic transmit FIFO size register

offset address: 0x100

Reset value: 0x08001000

This application can program the RAM size that must be allocated to the periodic TxFIFO.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PTXFD[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
PTXSA[15:0]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b16	PTXFD	Host periodic TxFIFO depth	Host periodic TxFIFO depth In 32-bit words. The minimum value is 16										R/W		
b15~b0	PTXSA	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 aperiodic TxFIFO depth.										R/W		

34.7.2.14 USBHS Device IN Endpoint Transmit FIFO Size Register (USBHS_DIEPTXF_x) (_x = 1..15)

USBHS device IN endpoint transmit FIFO size register

offset address: 0x104+(x-1)*0x4

Reset value: 0x08001000+(x-1)*0x800

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
INEPTXFD[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
INEPTXSA[15:0]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b16	INEPTXFD	IN endpoint TxFIFO depth	Device IN endpoint TxFIFO depth In 32-bit words. The minimum value is 16										R/W		
b15~b0	INEPTXSA	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.	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		

34.7.3 USBHS Host Mode Register

The host mode register affects the operation of the module in host mode. Host mode registers must not be accessed in device mode because the result is ambiguous.

Unless otherwise specified, bit values in register descriptions are represented in binary.

34.7.3.1 USBHS Host Configuration Register (USBHS_HCFG)

USBHS Host configuration register

offset address: 0x400

Reset value: 0x00000200

This register will configure 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															
FSLSS FSLSPCS[1:0]															
Bit	Marking	Place name	Function										Read and write		
b31~b3	Reserved	-	The reset value must be maintained.										R/W		
b2	FSLSS	Only FS and LS are supported	The application uses this bit to control the enumeration speed of the module. Using this bit, an 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. 0: HS/FS/LS, depending on the maximum speed supported by the connected device 1: FS/LS only, even if the connected device supports 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: Keep When the module is in LS host mode 00: reserved 01: Select 48MHz PHY clock frequency 10: Select 6MHz PHY clock frequency 11: Reserved Note: When the device is connected to the host, the FSLSPCS must be set according to the speed of the connected device (after changing this bit, a software reset must be performed).										R/W		

34.7.3.2 USBHS Host Frame Interval Register (USBHS_HFIR)

USBHS Host frame interval register

offset address: 0x404

Reset value: 0x0000EA60

This register is used to store the frame interval information set by the USBHS 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]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b26	Reserved	-	Reset value must be maintained										R/W		
b15~b0	FRIVL	frame interval	Frame interval 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. An 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 USBHS_HPRT). If the value is not programmed, the module will calculate it based on the PHY clock specified in the FS/LS PHY Clock Select field of the Host Configuration Register (FSLSPCS in USBHS_HCFG). Do not change the value of this field after initial configuration. Setting value = frame interval (ms) × (PHY clock frequency) -1 Note: The FRIVL bit can be modified whenever the application needs to change the frame interval time.										R/W		

34.7.3.3 USBHS Host Frame Number/Frame Remaining Time Register (USBHS_HFNUM)

USBHS 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 time remaining in the current frame (in PHY clocks).

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
FTREM[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
FRNUM[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	FTREM	Frame remaining time	Frame time remaining Indicates the time remaining in the current frame (in PHY clocks). This field is decremented by 1 every 1 PHY clock that elapses. When the value reaches zero, this field will be reloaded with the value in the Interframe Interval register and a new SOF will be sent by the module over USB.	R											
b15~b0	FRNUM	frame number	Frame number The value of this field is incremented by 1 when a new SOF is sent over USB and cleared when 0x3FFF is reached.	R											

34.7.3.4 USBHS Host Periodic Transmit FIFO/Queue Status Register (USBHS_HPTXSTS)

USBHS Host periodic transmit FIFO/queue status register

offset address: 0x410

Reset value: 0x00080800

This read-only register contains free space information for the periodic TxFIFO and periodic transmit 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]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b24	PTXQTOP	Periodically send requests to the top of the queue	Periodically send requests to the top of the queue (Top of the periodic transmit request queue) Indicates the item currently being processed by the MAC in the periodic Tx request queue. This register is used for debugging. Bit 31: Odd/Even frame — 0: Send in even frames — 1: Send in odd-numbered frames Bits 30:27: Channel/Endpoint Number (Channel number) Bit 26:25: Type (Type) — 00: Input/Output — 01: zero-length packet — 11: Disable channel command Bit 24: Terminate (last entry for the selected channel)	R											
b23~b16	PTXQSAV	Periodically send request queue free space	Periodically send request queue free space (Periodic transmit request queue space available) Indicates the number of free slots in the periodic send request queue available for writing. The queue contains both IN and OUT requests. 0: The periodic send request queue is full 1:1 position available 2: 2 positions available n: n positions available (where, n ranges: 0~8) Other values: Keep	R											
b15~b0	PTXFSAVL	Periodically transmit data FIFO free space	Periodically transmit data FIFO free space (Periodic transmit data FIFO space available) Indicates the number of free locations available for writing to the periodic TxFIFO. in 32-bit words 0: Periodic TxFIFO is full 1:1 word available 2: 2 words available n: n words available (where, n ranges: 0~PTXFD) Other values: Keep	R											

34.7.3.5 USBHS Host All Channel Interrupt Register (USBHS_HINT)

USBHS Host all channels interrupt register

offset address: 0x414

Reset value: 0x0000 0000

When an event occurs on the channel, the host-wide channel interrupt register uses the host channel interrupt bit in the module interrupt register (HCINT bit in USBHS_GINTSTS) to interrupt the application. Each channel corresponds to 1 interrupt bit, with a maximum of 16 bits.

When the application clears an interrupt via 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
HAINT[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	The reset value must be maintained.	R/W											
b15~b0	HAINT	channel interrupt	Channel interrupt One bit for each channel: Channel 0 corresponds to bit 0, and channel 15 corresponds to bit 15.	R											

34.7.3.6 USBHS Host All Channel Interrupt Mask Register (USBHS_HAINTMSK)

USBHS 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 to interrupt the application when an event occurs on the channel.

Each channel corresponds to one interrupt mask bit, with a maximum of 16 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
HAINTM[15:0]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b16	Reserved	-	Read as "0", write as "0"										R/W		
b15~b0	HAINTM	channel interrupt mask	Channel interrupt mask 0: Mask interrupt 1: Enable interrupt One bit for each channel: Channel 0 corresponds to bit 0, and channel 15 corresponds to bit 15.										R/W		

34.7.3.7 USBHS Host Port Control and Status Register (USBHS_HPRT)

USBHS Host port control and status register

offset address: 0x440

Reset value: 0x0000 0000

This register is only available in master mode. Currently, the USBHS host only supports one port.

This register contains USB port related information such as USB reset, enable, suspend, resume, 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 USBHS_GINTST). When a port interrupt occurs, the application must read this register and clear the bit that caused the interrupt. 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]	PTCTL[3]
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
PTCTL[2:0]	PWPR	PLSTS[1:0]	Reser ved	PRST	PSUS P	PRES	POCC HNG	POCA	PENC HNG	PENA	PCDE T	PCST S			
<hr/>															
Bit	Marking	Place name	Function												Read and write
b31~b19	Reserved	-	The reset value must be maintained.												R/W
b18~b17	PSPD	port speed	Port speed Indicates the speed of the device connected to this port. 00: high speed 01: Full Speed 10: low speed 11: Reserved												R
b16~b13	PTCTL[3:0]	Port Test Control	Port test control The application writes a non-zero value to this field to put the port in test mode, and the port generates a signal for the corresponding mode. 0000: Test mode disabled 0001: Test_J mode 0010: Test_K mode 0011: Test_SE0_NAK mode 0100: Test_Packet mode 0101: Test_Force_Enable Other values: Keep												R/W
b12	PWPR	port power	Port power Applications use this field to control power to this port. Since the built-in PHY of the USBHS does not have the power supply capability, when this is set to 1, the external USB power chip is enabled to supply power through USBHS_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 USBHS_DM Bit 10: Logic level of USBHS_DP												R
b9	Reserved	-	The reset value must be maintained.												R/W
b8	PRST	port reset	Port reset When an application sets this bit, it initiates a reset sequence on this port. The application program must time the reset cycle and clear this bit after the reset sequence is complete. 0: Port not in reset state 1: Port is in reset state The application must set this bit for a minimum of 10 ms to initiate a reset on the port.												R/W

			Port suspend Applications set this bit to put this port in suspend mode. Only when this bit is set will the module stop sending SOF. To stop the PHY clock, the application must set the port clock stop bit, which enables the PHY's suspend input pin. The read value of this bit reflects the current pending state of the port. When a remote wake-up signal is detected, or the application program sets the port reset bit or port recovery bit in this register to 1, the module can clear this bit; or the application program sets the recovery/remote wake-up detection interrupt bit or interrupt in the module interrupt register. The open connection detection interrupt bit (respectively WKINT or DISCINT in USBHS_GINTSTS) is set to 1, and the module can also clear this bit. 0: The port is not in suspend mode 1: The port is in suspend mode	
B7	PSUSP	port hang	Port resume The application sets this bit to drive the resume signal on this port. The module will continue to drive the resume signal until the application program clears this bit. As indicated by the port resume/remote wake-up detection interrupt bit in the module's interrupt register (WKINT bit in USBHS_GINTSTS), if the module detects a USB remote wake-up sequence, it starts driving the resume signal without application intervention; if the module detects a disconnect If connected, clear this bit. The read value of this bit indicates whether the current module The recovery signal is being driven. 0: Do not drive the recovery signal 1: Drive recovery signal	R/W
b6	PRES	port recovery	Port enable/disable change The module sets this bit to 1 when the state of port enable bit 2 in this register changes. This bit is cleared by software by writing a 1 to it.	R/W
b5~b4	Reserved	-	The reset value must be maintained.	R/W
b3	PENCHNG	Port enable/disable change	Port enable After the port has performed a reset sequence, it can only be enabled by the module and can be disabled by an overcurrent condition, a disconnect condition, or by clearing this bit by the application program. The application cannot set this bit by writing to the register. The port can only be disabled by clearing this bit. Operations on this bit will not trigger any interrupts to the application. 0: disable port 1: enable port	R/W
b2	PENA	port enable	Port connect detected When a device connection is detected, the module sets this bit to trigger an interrupt to the application using the host port interrupt bit in the module's interrupt register (HPRTINT bit in USBHS_GINTSTS). The application must set this bit to clear this interrupt.	R/W
b1	PCDET	port connection detected	Port connect status 0: No device is connected to the port 1: The port is connected to a device	R/W
b0	PCSTS	port connection status	Port connect status 0: No device is connected to the port 1: The port is connected to a device	R

34.7.3.8 USBHS Host Channel x Characteristic Register (USBHS_HCCHARx) ($x = 0..15$)

USBHS Host channel-x characteristics register

offset address: 0x500 + (channel number × 0x20)

Reset value: 0x0000 0000

This register is used to set the host channel characteristics.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CHENA	CHDIS	ODDFRM	DAD[6:0]				MC[1:0]		EPTYP[1:0]		LSDEV	Reserved			
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
EPDIR	EPNUM[3:0]				MPSIZ[10:0]										
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31	CHENA	channel enable	Channel enable This field is set by application software and cleared by USBHS host hardware. 0: Channel disabled 1: enable channel										R/W		
b30	CHDIS	Channel forbidden	Channel disable The application sets this bit to stop sending/receiving data over the channel, even if the transmission over the channel has not completed, the stop operation still takes effect. The application must wait for the interrupt of the disabled channel to confirm that the channel has been disabled.										R/W		
b29	ODDFRM	odd frame	Odd frame This field is set or reset by the application to indicate that the USBHS host must transmit odd or even frames, respectively. This field applies only to periodic (synchronized and interrupted) transactions. 0: even (micro) frames 1: Odd (micro) frames										R/W		
b28~b22	DAD	Device address	Device address This field is used to specify a specific device to communicate with this host.										R/W		
b21~b20	MC	Multiple Count (MC)/Error Count (EC)	Multi Count (MC)/Error Count (EC) — When the Split Enable bit (SPLITEN) in the Host Channel x Split Control Register (USBHS_HCSPLTx) is reset (0), this field indicates to the host the number of transactions this periodic endpoint must perform per microframe. For aperiodic transfers, this field specifies the number of packets to acquire for this channel before the internal DMA engine changes arbitration. 00: reserved, will produce ambiguous results of operations on this field 01: 1 transaction 10: This endpoint needs to issue 2 transactions per microframe 11: This endpoint needs to issue 3 transactions per microframe. - When the SPLITEN bit (1) in USBHS_HCSPLTx, this field indicates the number of immediate retries to be performed on a periodically detached transaction in the event of an error. This field must be set to at least 01.										R/W		
b19~b18	EPTYP	Endpoint type	Endpoint type Indicates the selected transfer type. 00: Control 01: Sync 10: Batch 11: Interrupt										R/W		
b17	LSDEV	low speed equipment	Low-speed device This field is set 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/W		

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 device endpoints communicating with this host channel.	R/W

34.7.3.9 USBHS Host Channel x Split Control Register (USBHS_HCSPLTx) (x = 0..15)

USBHS host channel-x split control register

offset address: 0x504 + (channel number × 0x20)

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
SPLIT EN	Reserved													COMP LSPLT	
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
XACTPOS[1:0]	HUBADDR[6:0]										PRTADDR[6:0]				

Bit	Marking	Place name	Function	Read and write
b31	SPLITEN	Separate enable	Split enable When an application sets this bit, it indicates that the channel is allowed to perform split communication transactions.	R/W
b30~b17	Reserved	-	The reset value must be maintained.	R/W
b16	COMPLSPLT	Perform full separation	Do complete split When an application sets this bit, it can request the host to perform a fully detached communication transaction.	R/W
b15~b14	XACTPOS	Transaction location	Transaction position This field is used to decide whether to send the full, first, middle or last payload with each OUT transaction. 00: Middle. Refers to the intermediate payload of this transaction (greater than 188 bytes) 01: The end. Refers to the last payload (greater than 188 bytes) of this transaction. 10: Start. Refers to the first data payload of this transaction (greater than 188 bytes) 11: All. Refers to the entire data payload of this transaction (less than or equal to 188 bytes)	R/W
b13~b7	HUBADDR	hub address	Hub address This field stores the hub device address of the transaction forwarder.	R/W
b6~b0	PRTADDR	port address	Port address This field is the port number of the recipient transaction forwarder.	R/W

34.7.3.10 USBHS Host Channel x Interrupt Register (USBHS_HCINTx) (x = 0..15)

USBHS Host channel-x interrupt register

offset address: 0x508 + (channel number × 0x20)

Reset value: 0x0000 0000

This register indicates the state of the channel when USB and AHB related events occur. The application must read this register when the host channel interrupt bit in the module interrupt register (the HCINT bit in USBHS_GINTSTS) is set. Before performing a read operation on the register, the application must read the Host All Channels Interrupt (USBHS_HAIN) register to obtain the exact channel number of the Host Channel x Interrupt Register. The application must clear the corresponding bits in this register to clear the corresponding bits in the USBHS_HAIN and USBHS_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															
Bit	Marking	Place name	Function										Read and write		
b31~b11	Reserved	-	The reset value must be maintained.										R/W		
b10	DTERR	data switching error	Data toggle error The application needs to clear this bit by writing a 1.										R/W		
b9	FRMOR	frame overflow error	Frame overrun error (Frame overrun) The application needs to clear this bit by writing a 1.										R/W		
b8	BBERR	crosstalk error	Babble error A typical cause of a crosstalk event is that an endpoint sends a packet, but the packet length exceeds the endpoint's maximum packet length. The application needs to clear this bit by writing a 1.										R/W		
B7	TXERR	communication transaction error	Transaction error Indicates that one of the following errors has occurred on the USB: CRC check failed time out Bit stuffing error wrong EOP The application needs to clear this bit by writing a 1.										R/W		
b6	NYET	NYET response received	Response received interrupt										R/W		
b5	ACK	Receive/send ACK response	ACK response received/transmitted interrupt The application needs to clear this bit by writing a 1.										R/W		
b4	NAK	Received NAK response	NAK response received interrupt The application needs to clear this bit by writing a 1.										R/W		
b3	STALL	Receive STALL response	Received STALL response (STALL response received interrupt										R/W		
b2	AHBERR	AHB error	AHB error This error is only generated when in internal DMA mode and an AHB error occurs during an AHB read/write operation. The application can obtain the error address by reading the corresponding DMA channel address register. The application needs to clear this bit by writing a 1										R/W		
b1	CHH	channel stop	Channel halted Abnormal end of transfer due to any USB transaction error or in response to an application inhibit request. The application needs to clear this bit by writing a 1.										R/W		
b0	XFRC	Transmission completion	Transfer completed No errors occurred and the transfer completed normally. The application needs to clear this bit by writing a 1.										R/W		

34.7.3.11 USBHS Host Channel x Interrupt Mask Register (USBHS_HCINTMSKx) (x = 0..15)

USBHS Host channel-x interrupt mask register

offset address: 0x50C + (channel number × 0x20)

Reset value: 0x0000 0000

This register is used to select masking of host channel interrupts.

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																
Bit	Marking	Place name	Function													Read and write
b31~b11	Reserved	-	The reset value must be maintained.													R/W
b10	DTERRM	Data Switch 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 Mask	Babble error mask 0: Mask interrupt 1: Enable interrupt													R/W
B7	TXERRM	Communication Transaction Error Interrupt Mask	Transaction error mask 0: Mask interrupt 1: Enable interrupt													R/W
b6	NYETM	NYET responds to receive interrupt mask	NYET response received interrupt mask 0: Mask interrupt 1: Enable interrupt													R/W
b5	ACKM	Receive/Send ACK Response Interrupt Mask	ACK response receive/transmit interrupt mask (ACK response received/transmitted interrupt mask) 0: Mask 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	Receive STALL response interrupt mask	STALL response received interrupt mask 0: Mask interrupt 1: Enable interrupt													R/W
b2	AHBERRM	AHBERR interrupt mask	AHBERR interrupt mask (AHB error mask) 0: Mask interrupt 1: Enable interrupt													R/W
b1	CHHM	Channel stop interrupt mask	Channel halted mask 0: Mask interrupt 1: Enable interrupt													R/W
b0	XFRCM	Transport complete interrupt mask	Transfer completed mask 0: Mask interrupt 1: Enable interrupt													R/W

34.7.3.12 USBHS Host Channel x Transfer Size Register (USBHS_HCTSIZx) (x = 0..15)

USBHS 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 data PID.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DOPING	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]															

Bit	Marking	Place name	Function	Read and write
b31	DOPING	Perform a PING operation	Perform PING operation (Do ping), this bit is only used for OUT transmission. Setting this bit to 1 instructs the host to implement the PING protocol. Note: Do not set this bit for IN transfers. If this bit is set for IN transfers, the channel will be disabled.	R/W
b30~b29	DPID	data PID	Data PID (Data PID) The application sets the initial sync PID for data communication in this field. The host retains the setting of this field during this transfer transaction. 00: DATA0 01: DATA2 10: DATA1 11: MDATA/SETUP	R/W
b28~b19	PKTCNT	packet count	Packet count The application sets the number of packets to be sent or received in this field. The host decrements the count each time a packet is successfully sent or received. After this value reaches 0, the application is interrupted to indicate that the operation completed normally.	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 the transfer. For IN operations, this field is the buffer size reserved by the application for transmission. For IN transactions (periodic and aperiodic), the application programs this field to an integer multiple of the maximum packet size.	R/W

34.7.3.13 USBHS Host Channel xDMA Address Register (USBHS_HCDMAx) (x = 0..15)

USBHS Host channel-x DMA address register

offset address: 0x514 + (channel number × 0x20)

Reset value: 0xFFFF XXXX

This register is used to set the DMA address in 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]															

Bit	Marking	Place name	Function	Read and write
b31~b0	DMAADDR	DMA address	DMA address This field stores the address of the memory for the DMA transfer used by the host to get data from or send data to the device endpoint. This register is incremented each time an AHB transfer ends.	R/W

34.7.4 USBHS Device Mode Register

The device mode registers affect the operation of the module in device mode. Device mode registers must not be accessed in host mode because the result is ambiguous.

Unless otherwise specified, bit values in register descriptions are represented in binary.

34.7.4.1 USBHS Device Configuration Register (USBHS_DCFG)

USBHS Device configuration register

offset address: 0x800

Reset value: 0x0822 0000

This register configures the module in device mode after power-up, some control command 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]							Reser ved	NZLS OHSK		DSPD[1:0]
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b13	Reserved	-	The reset value must be maintained.	R/W											
b12~b11	PFIVL	Periodic frame interval	Periodic frame interval Indicates the point in time within a frame at which the application must be notified using periodic frame interrupts. This function can be used to determine if all isochronous communications for the frame are complete. 00: 80% (micro) frame interval 01: 85% (micro) frame interval 10: 90% (micro) frame interval 11: 95% (micro) frame interval	R/W											
b10~b4	DAD	Device address	Device address The application must set this field according to the command parameter after executing each SetAddress control command.	R/W											
b3	Reserved	-	The reset value must be maintained.	R/W											
b2	NZLSOHSK	Non-zero length state OUT handshake signal	Non-zero length state OUT handshake signal (Non-zero-length status OUT handshake) During an OUT transaction in the control transfer status phase, an application can use this field to select the handshake signal to send when a non-zero length packet is received by the module. 1: When receiving a non-zero-length state OUT transaction, reply to the STALL handshake signal, and the received OUT data packet is not sent to the application. 0: Send the received OUT packet (zero length or non-zero length) to the application and reply to the handshake based on the NAK and STALL bits of the endpoint in the device endpoint control register.	R/W											
b1~b0	DSPD	device speed	Device speed Indicates the speed at which the application requires the module to enumerate, or the maximum speed supported by the application. However, the actual bus speed can only be determined after completing the chirp sequence, and this speed is based on the speed of the USB host connected to the module. 00: high speed 01: reserved 10: Reserved 11: Full speed (USB 1.1 transceiver clock is 48 MHz)	R/W											

34.7.4.2 USBHS Device Control Register (USBHS_DCTL)

USBHS Device control register

offset address: 0x804

Reset value: 0x0000 0002

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				POPR GDN E	CGO NAK	SGO NAK	CGIN AK	SGIN AK	TCTL[2:0]			GON STS	GINS TS	SDIS	RWAS IG			
<hr/>																		
Bit	Marking	Place name	Function	Read and write														
b31~b12	Reserved	-	The reset value must be maintained.	R/W														
b11	POPRGDNE	Power-on programming completed	Power-on programming done The application uses this bit to indicate that the registers have been programmed after a wake-up from Power-down mode.	R/W														
b10	CGONAK	Clear global OUT NAK	Clear global OUT NAK(Clear global OUT NAK) A write to this bit clears the global OUT NAK.	R/W														
b9	SGONAK	Set global OUT NAK	Set global OUT NAK A write to this bit will set the global OUT NAK. Applications use this bit to send NAK handshake signals on all OUT endpoints. The application should only set this bit if it is sure that the global OUT NAK valid bit in the module interrupt register (GONAKEFF bit in USBHS_GINTSTS) is cleared.	R/W														
b8	CGINAK	Clear global IN NAK	Clear global IN NAK (Clear global IN NAK) A write to this bit clears the global IN NAK.	R/W														
B7	SGINAK	Set global IN NAK	Set global IN NAK A write to this field sets the global aperiodic IN NAK. The application uses this bit to cause all aperiodic IN endpoints to send the NAK handshake signal. The application should only set this bit if it is sure that the global IN NAK valid bit in the module interrupt register (GINAKEFF bit in USBHS_GINTSTS) is cleared.	R/W														
b6~b4	TCTL	test control	Test control 000: Test mode disabled 001: Test_J mode 010: Test_K mode 011: Test_SE0_NAK mode 100: Test_Packet mode 101: Test_Force_Enable Other values: Keep	R/W														
b3	GONSTS	Global OUT NAK status	Global OUT NAK status 0: Handshaking will be sent based on FIFO status and NAK and STALL bit settings. 1: No data is received regardless of whether there is free space in the RxFIFO. Except for SETUP transactions, the NAK handshake signal is replied to all received packets. All isochronous type OUT packets will be discarded.	R														
b2	GINSTS	Global IN NAK status	Global IN NAK status 0: The handshake signal will be replied based on the data availability in the transmit FIFO. 1: Causes all aperiodic IN endpoints to reply to the NAK handshake signal, regardless of data availability in the transmit FIFO.	R														

			Soft disconnect The application uses this bit to signal the USBHS module to perform a soft disconnect. When this bit is set, the host will not see that the device is connected, and the device will not receive signals on the USB. The module remains disconnected until the application clears this bit. 0: Normal operation. Clearing this bit after a soft disconnect will cause the host to receive a device connected event. After reconnecting the device, the USB host restarts the device enumeration. 1: Causes the host to receive a device disconnect event.	R/W
b1	SDIS	soft disconnect	At full speed, the minimum soft disconnect time is specified as follows: Suspended state: The minimum time is 1ms+2.5us idle state: 2.5us Non-idle or suspended state: 2.5us	

			Send Remote wakeup signaling When the application sets this bit to 1, the module will initiate a remote wake-up signal to wake up the USB host. Applications must set this bit to bring the module out of the suspended state. According to the USB 2.0 specification, the application must clear this bit within 1 ms to 15 ms after setting it.	R/W
b0	RWASIG	Send remote wakeup signal		

34.7.4.3 USBHS Device Status Register (USBHS_DSTS)

USBHS Device status register

offset address: 0x808

Reset value: 0x0000 0002

This register indicates the state of the module when a USB related event occurs. When an interrupt occurs, the endpoint information on which the interrupt occurred must be read from the device-wide interrupt (USBHS_DAINT) register.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
Reserved								LNSTS[1:0]	FNSOF[13:8]							
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0	
FNSOF[7:0]								Reserved				EERR	ENUMSPD[1:0]	SUSP STS		

Bit	Marking	Place name	Function	Read and write
b31~b24	Reserved	-	The reset value must be maintained.	R/W
b23~b22	LNSTS	USB bus status	USB bus status LNSTS[1]: Logic level of D+ LNSTS[0]: The logic level of D-	R
b21~b8	FNSOF	Frame number of received SOF	Frame number of the received SOF	R
b7~b4	Reserved	-	The reset value must be maintained.	R/W
b3	EERR	indefinite error	Erratic error The module sets this bit to report any indeterminate errors. Due to an indeterminate error, the USBHS controller goes into a suspend state and generates an interrupt on the early suspend bit in the USBHS_GINTSTS register (ESUSP bit in USBHS_GINTSTS). If the early pending interrupt was triggered by an indeterminate error, the application can only perform a soft disconnect to resume communication.	R
b2~b1	ENUMSPD	Enumeration speed	Enumerated speed Indicates the speed to be enumerated by the USBHS controller after detecting the speed through the chirp sequence. 00: high speed 01: reserved 10: Reserved 11: Full speed (PHY clock running at 48 MHz) Other values: Keep	R
b0	SUSPSTS	Suspended state	Suspend status In device mode, this bit is set whenever a suspend state is detected on the USB. When the idle state on the USB bus remains for 3ms, the module will enter the suspend state. The module exits the suspended state when: — There is activity on the USB cable — The application performs a write operation to the Remote Wakeup Signal bit (RWUSIG bit in USBHS_DCTL) of the USBHS_DCTL register.	R

34.7.4.4 USBHS device IN endpoint general interrupt mask register (USBHS_DIEPMSK)

USBHS Device IN endpoint common interrupt mask register

offset address: 0x810

Reset value: 0x0000 0000

This register is used in conjunction with the individual USBHS_DIEPINTx registers of all endpoints to generate interrupts on each IN endpoint. The IN endpoint interrupt in the USBHS_DIEPINTx registers can be masked by writing to the appropriate bits 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	NAK M	Reserved				TXFURM	Reser ved	INEPNEM	INEPNMM	TTXFEMSK	TOM	Reser ved	EPDM	XFRCM	
<hr/>															
Bit	Marking	Place name		Function										Read and write	
b31~b14	Reserved	-		The reset value must be maintained.										R/W	
b13	NAKM	NAK interrupt mask		NAK interrupt mask (NAK mask) 0: Mask interrupt 1: Enable interrupt										R/W	
b12~b9	Reserved	-		The reset value must be maintained.										R/W	
b8	TXFURM	Transmit FIFO underrun interrupt mask		Transmit FIFO underrun interrupt mask (FIFO underrun mask) 0: Mask interrupt 1: Enable interrupt										R/W	
B7	Reserved	-		The reset value must be maintained.										R/W	
b6	INEPNEM	IN endpoint NAK valid interrupt mask		IN endpoint NAK effective mask 0: Mask interrupt 1: Enable interrupt										R/W	
b5	INEPNMM	Received 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	TTXFEMSK	IN token received interrupt mask when TxFIFO is empty		(IN token received when TxFIFO empty mask) 0: Mask interrupt 1: Enable interrupt										R/W	
b3	TOM	Timeout interrupt mask (asynchronous endpoints)		(Timeout condition mask (Non-isochronous endpoints)) 0: Mask interrupt 1: Enable interrupt										R/W	
b2	Reserved	-		The reset value must be maintained.										R/W	
b1	EPDM	Endpoint Disable Interrupt Mask		Endpoint disabled interrupt mask 0: Mask interrupt 1: Enable interrupt										R/W	
b0	XFRCM	Transport complete interrupt mask		Transfer completed interrupt mask 0: Mask interrupt 1: Enable interrupt										R/W	

34.7.4.5 USBHS device OUT endpoint general interrupt mask register (USBHS_DOEPMSK)

USBHS Device OUT endpoint common interrupt mask register

offset address: 0x814

Reset value: 0x0000 0000

This register is used in conjunction with the individual USBHS_DOEPINTx registers of all endpoints to generate interrupts on each OUT endpoint. The OUT endpoint interrupt in the USBHS_DOEPINTx registers can be masked by writing to the appropriate bits 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	NYETM	Reserved				OPEM	Reserved	B2BS TUP	Reserved	OTEP DM	STUP M	Reserved	EPDM	XFRCM	
<hr/>															
Bit	Marking	Place name		Function										Read and write	
b31~b15	Reserved	-		The reset value must be maintained.										R/W	
b14	NYETM	NYET interrupt mask		NYET interrupt mask (NYET mask) 0: Mask interrupt 1: Enable interrupt										R/W	
b13~b9	Reserved	-		The reset value must be maintained.										R/W	
b8	OPEM	OUT packet error interrupt mask		OUT packet error mask 0: Mask interrupt 1: Enable interrupt										R/W	
B7	Reserved	-		The reset value must be maintained.										R/W	
b6	B2BSTUP	Received consecutive SETUP packets interrupt mask		Back-to-back SETUP packets received mask Applies only to control OUT endpoints. 0: Mask interrupt 1: Enable interrupt										R/W	
b5	Reserved	-		The reset value must be maintained.										R/W	
b4	OTEPDM	OUT token received interrupt mask when endpoint is disabled		(OUT token received when endpoint disabled mask) Applies only to control OUT endpoints. 0: Mask interrupt 1: Enable interrupt										R/W	
b3	STUPM	SETUP phase completes interrupt masking		SETUP phase done mask Applies to control endpoints only. 0: Mask interrupt 1: Enable interrupt										R/W	
b2	Reserved	-		The reset value must be maintained.										R/W	
b1	EPDM	Endpoint Disable Interrupt Mask		Endpoint disabled interrupt mask 0: Mask interrupt 1: Enable interrupt										R/W	
b0	XFRCM	Transport complete interrupt mask		Transfer completed interrupt mask 0: Mask interrupt 1: Enable interrupt										R/W	

34.7.4.6 USBHS Device Entire Endpoint Interrupt Register (USBHS_DAINT)

USBHS Device OUT endpoint common interrupt mask register

offset address: 0x818

Reset value: 0x0000 0000

When a valid event occurs on the endpoint, the USBHS_DAINT register will interrupt the application through the Device OUT endpoint interrupt bit or the Device IN endpoint interrupt bit in the USBHS_GINTSTS register (OEPINT or IEPINT bits in USBHS_GINTSTS, respectively). Each endpoint corresponds to an interrupt bit, and both the OUT endpoint and the IN endpoint have a maximum of 16 interrupt bits. Bidirectional endpoints will use the corresponding IN and OUT interrupt bits. When the application sets and clears the bits in the corresponding device endpoint x interrupt registers (USBHS_DIEPINTx/USBHS_DOEPINTx), the corresponding bits in this register are also set and cleared.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
OEPINT[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
IEPINT[15:0]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b16	OEPINT	OUT endpoint interrupt bit	OUT endpoint interrupt bits One bit for each OUT endpoint: OUT endpoint 0 corresponds to bit 16 and OUT endpoint 15 corresponds to bit 31.										R/W		
b15~b0	IEPINT	IN endpoint interrupt bit	IN endpoint interrupt bits One bit for each IN endpoint: IN endpoint 0 corresponds to bit 0 and IN endpoint 15 corresponds to bit 15.										R/W		

34.7.4.7 USBHS Device Entire Endpoint Interrupt Mask Register (USBHS_DAINTMSK)

USBHS Device all endpoints interrupt mask register

offset address: 0x81C

Reset value: 0x0000 0000

The USBHS_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 USBHS_DAINT register bit corresponding to the interrupt will still be set.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
OEPINTM[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
IEPINTM[15:0]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b16	OEPINTM	OUT endpoint interrupt mask bit	OUT endpoint interrupt mask bits One bit for each OUT endpoint: OUT endpoint 0 corresponds to bit 16 and OUT endpoint 15 corresponds to bit 31. 0: Mask interrupt 1: Enable interrupt										R/W		
b15~b0	IEPINTM	IN endpoint interrupt mask bit	IN endpoint interrupt mask bits One bit for each IN endpoint: IN endpoint 0 corresponds to bit 0 and IN endpoint 15 corresponds to bit 15. 0: Mask interrupt 1: Enable interrupt										R/W		

34.7.4.8 USBHS Device Threshold Control Register (USBHS_DTHRCTL)

USBHS Device threshold control register

offset address: 0x0830

Reset value: 0x0C10 0020

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved				ARPE N	Reser ved	RXTHRLEN[8:0]								RXTH REN	
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
Reserved				TXTHRLEN[7:0]								ISOT HREN	NON OISO THRE N		
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b28	Reserved	-	The reset value must be maintained.	R/W											
b27	ARPEN	Arbiter Dwell Enable	Arbiter parking enable This bit is used to control the residency of the internal DMA arbitration on the IN endpoint. When the threshold is enabled and this bit is set, the DMA will hold its arbitration on the IN endpoint that received the token. In this way, underflow conditions can be avoided. Persistence is enabled by default.	R/W											
b26	Reserved	-	The reset value must be maintained.	R/W											
b25~b17	RXTHRLEN	receive threshold length	Receive threshold length This field specifies the size of the receive threshold in double words. This field specifies the amount of data received on USB before the module initiates a transfer on AHB. The minimum threshold length is eight double words. It is recommended that RXTHRLEN be the same as the set AHB burst transmission length.	R/W											
b16	RXTHREN	Receive Threshold Enable	Receive threshold enable When this bit is set to 1, the module enables the threshold for the receive direction.	R/W											
b15~b11	Reserved	-	The reset value must be maintained.	R/W											
b10~b2	TXTTHRLEN	Send Threshold Length	Transmit threshold length This field specifies the size of the transmit threshold in double words. This field specifies the amount of data in the corresponding endpoint transmit FIFO before the module initiates a transfer on USB. The minimum threshold length is eight double words. This field controls the synchronous and asynchronous IN endpoint thresholds. It is recommended that TXTTHRLEN is the same as the set AHB burst transmission length.	R/W											
b1	ISOTHREN	ISO IN endpoint threshold enable	ISO IN endpoint threshold enable When this bit is set to 1, the module enables synchronization of the IN endpoint threshold.	R/W											
b0	NONOISOTHREN	Asynchronous IN endpoint threshold enable	Nonisochronous IN endpoints enable (Nonisochronous IN endpoints enable) When this bit is set to 1, the module will enable the threshold for non-synchronous IN endpoints.	R/W											

34.7.4.9 USBHS device IN endpoint FIFO empty interrupt mask register (USBHS_DIEPEMPMSK)

USBHS 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 empty 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
INEPTXFEM[15:0]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b16	Reserved	-	The reset value must be maintained.										R/W		
b15~b0	INEPTXFEM	IN EP TxFIFO empty interrupt mask bit	IN EP Tx FIFO empty interrupt mask bit (IN EP Tx FIFO empty interrupt mask bits) These bits are used as mask bits for USBHS_DIEPINTx. Each bit corresponds to a TXFE interrupt for an IN endpoint: IN endpoint 0 corresponds to bit 0 and IN endpoint 15 corresponds to bit 15 0: Mask interrupt 1: Enable interrupt										R/W		

34.7.4.10 USBHS Device Single Endpoint Interrupt Register (USBHS_DEACHINT)

USBHS Device each endpoint interrupt register

offset address: 0x0838

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved														OEP1 INT	Reser ved
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
Reserved														IEP1I NT	Reser ved
<hr/>															
Bit	Marking	Place name	Function												Read and write
b31~18	Reserved	-	The reset value must be maintained.												R/W
b17	OEP1INT	OUT endpoint 1 interrupt bit	OUT endpoint 1 interrupt bit												R/W
b16~b2	Reserved	-	The reset value must be maintained.												R/W
b1	IEP1INT	IN endpoint 1 interrupt bit	IN endpoint 1 interrupt bit (IN endpoint 1interrupt bit)												R/W
b0	Reserved	-	The reset value must be maintained.												R/W

34.7.4.11 USBHS Device Single Endpoint Interrupt Mask Register (USBHS_DEACHINTMASK)

USBHS Device each endpoint interrupt mask register

offset address: 0x083C

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved														OEP1INTM	Reserved
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
Reserved														IEP1INTM	Reserved
<hr/>															
Bit	Marking	Place name	Function											Read and write	
b31~18	Reserved	-	The reset value must be maintained.											R/W	
b17	OEP1INTM	OUT endpoint 1 interrupt mask bit	OUT endpoint 1 interrupt mask bit 0: Mask interrupt 1: Enable interrupt											R/W	
b16~b2	Reserved	-	The reset value must be maintained.											R/W	
b1	IEP1INTM	IN endpoint 1 interrupt mask bit	IN endpoint 1 interrupt mask bit (IN endpoint 1interrupt mask bit) 0: Mask interrupt 1: Enable interrupt											R/W	
b0	Reserved	-	The reset value must be maintained.											R/W	

34.7.4.12 USBHS device IN terminal 1 interrupt mask register (USBHS_DIEPEACHMSK1)

USBHS device each in endpoint-1 interrupt register

offset address: 0x844

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	NAK M	Reserved					TXFURM	Reser ved	INEP NEM	INEP NMM	TTXF EMSK	TOM	Reser ved	EPDM	XFR M
Bit	Marking	Place name	Function										Read and write		
b31~14	Reserved	-	The reset value must be maintained.										R/W		
b13	NAKM	NAK interrupt mask	NAK interrupt mask 0: Mask interrupt 1: Enable interrupt										R/W		
b12~b9	Reserved	-	The reset value must be maintained.										R/W		
b8	TXFURM	Transmit FIFO underflow interrupt mask	Transmit FIFO underrun interrupt mask (FIFO underrun mask) 0: Mask interrupt 1: Enable interrupt										R/W		
B7	Reserved	-	The reset value must be maintained.										R/W		
b6	INEPNEM	IN endpoint NAK valid interrupt mask	IN endpoint NAK effective mask 0: Mask interrupt 1: Enable interrupt										R/W		
b5	INEPNMM	Received 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	TTXFEMSK	IN token received interrupt mask when TxFIFO is empty	IN token received when TxFIFO empty mask 0: Mask interrupt 1: Enable interrupt										R/W		
b3	TOM	Timeout interrupt mask	Timeout condition mask (nonisochronous endpoints) 0: Mask interrupt 1: Enable interrupt										R/W		
b2	Reserved	-	The reset value must be maintained.										R/W		
b1	EPDM	Endpoint Disable Interrupt Mask	Endpoint disabled interrupt mask 0: Mask interrupt 1: Enable interrupt										R/W		
b0	XFRM	Transport complete interrupt mask	Transfer completed interrupt mask 0: Mask interrupt 1: Enable interrupt										R/W		

34.7.4.13 USBHS device OUT endpoint 1 interrupt mask register (USBHS_DOEPEACHMSK1)

USBHS device each out endpoint-1 interrupt register

offset address: 0x884

Reset value: 0x0000 0000

31	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
Reser ved	NYET M						OPEM	Reser ved	B2BS TUP	Reser ved	OTEP DM	STUP M	Reser ved	EPDM	XFRC M
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b15	Reserved	-	The reset value must be maintained.										R/W		
b14	NYETM	NYET interrupt mask	NYET interrupt mask (NYET mask) 0: Mask interrupt 1: Enable interrupt										R/W		
b13~b9	Reserved	-	The reset value must be maintained.										R/W		
b8	OPEM	OUT packet error interrupt mask	OUT packet error mask 0: Mask interrupt 1: Enable interrupt										R/W		
B7	Reserved	-	The reset value must be maintained.										R/W		
b6	B2BSTUP	Received consecutive SETUP packets interrupt mask	Back-to-back SETUP packets received mask Applies only to control OUT endpoints. 0: Mask interrupt 1: Enable interrupt										R/W		
b5	Reserved	-	The reset value must be maintained.										R/W		
b4	OTEPDM	OUT token received interrupt mask when endpoint is disabled	OUT token received interrupt mask when endpoint is disabled (OUT token received when endpoint disabled mask) Applies only to control OUT endpoints. 0: Mask interrupt 1: Enable interrupt										R/W		
b3	STUPM	SETUP phase completes interrupt masking	SETUP phase done mask Applies to control endpoints only. 0: Mask interrupt 1: Enable interrupt										R/W		
b2	Reserved	-	The reset value must be maintained.										R/W		
b1	EPDM	Endpoint Disable Interrupt Mask	Endpoint disabled interrupt mask 0: Mask interrupt 1: Enable interrupt										R/W		
b0	XFRCM	Transport complete interrupt mask	Transfer completed interrupt mask 0: Mask interrupt 1: Enable interrupt										R/W		

34.7.4.14 USBHS Device Control IN Endpoint 0 Control Register (USBHS_DIEPCTL0)

USBHS Device control IN endpoint 0 control register

offset address: 0x900

Reset value: 0x0000 8000

This register is used to control the control transfer endpoint 0.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPEN A	EPDIS	Reserved	SNAK	CNAK		TXFNUM[3:0]		STAL L	Reser ved	EPTYP[1:0]	NAKS TS	Reser ved			
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
USBA EP								Reserved							MPSIZ[1:0]

Bit	Marking	Place name	Function	Read and write
b31	EPENA	Endpoint enable	Endpoint enable The application sets this bit to start sending data on endpoint 0. The module clears this bit before triggering any of the following interrupts on this endpoint: — SETUP phase completed - Endpoint barring —Transmission completion	R/W
b30	EPDIS	Endpoint forbidden	Endpoint disable Applications can set this bit to stop sending data on an endpoint even before the transfer on that endpoint is complete. An application must wait until an endpoint inhibit interrupt occurs before an endpoint can be considered inhibited. The module will clear this bit before the endpoint disable interrupt bit is set. An application can set this bit only after the endpoint's endpoint enable bit is set.	R/W
b29~b28	Reserved	-	The reset value must be maintained.	R/W
b27	SNAK	Set the NAK bit	Set NAK bit (Set NAK) A write to this bit sets the endpoint's NAK bit. With this bit, the application can control the sending of the NAK handshake signal on the endpoint. The module can also set this bit of the endpoint after the endpoint receives the SETUP packet.	R/W
b26	CNAK	Clear the NAK bit	Clear NAK bit (Clear NAK) A write to this bit clears the endpoint's NAK bit.	R/W
b25~b22	TXFNUM	TxFIFO number	TxFIFO number The 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, when the endpoint receives the SETUP token, the module will clear this bit. If the NAK bit, the global IN NAK, or the global OUT NAK and this bit are all set, the STALL bit takes precedence.	R/W
b20	Reserved	-	The reset value must be maintained.	R/W
b19~b18	EPTYP	Endpoint type	Endpoint type Hardware is set to '00', indicating a control-type endpoint.	R
b17	NAKSTS	NAK status	NAK status Indicates the following results: 0: The module replies to a non-NAK handshake according to the FIFO status. 1: The module replies with a NAK handshake on this endpoint. When this bit is set (either by the application or by the module), the module will stop sending data even if there is still data available in the TxFIFO. Regardless of how this bit is set, the module always responds to the SETUP packet with an ACK handshake.	R
b16	Reserved	-	The reset value must be maintained.	R/W
b15	USBAEP	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/W

b1~b0	MPSIZ	maximum packet size	Maximum packet size Applications must program this field to the maximum packet size of the current logical endpoint. 00: 64 bytes 01: 32 bytes 10: 16 bytes 11: 8 bytes	R/W
-------	-------	---------------------	--	-----

34.7.4.15 USBHS device IN endpoint x control register (USBHS_DIEPCTLx) (x=1..15)

USBHS Device IN endpoint x control register

offset address: 0x900 + (endpoint number × 0x20)

Reset value: 0x0000 0000

Applications use this register to control the behavior of individual logical endpoints (except endpoint 0).

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPEN A	EPDI S	SOD DFR M	SDOP ID/ SEVN FRM	SNAK	CNAK		TXFNUM[3:0]		STAL L	Reser ved		EPTYP[1:0]	NAKS TS	ENOUM/ DPID	
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
USBA EP	Reserved			MPSIZ[10:0]											

Bit	Marking	Place name	Function	Read and write
b31	EPENA	Endpoint enable	Endpoint enable The application sets this bit to start sending data on the endpoint. The module clears this bit before triggering any of the following interrupts on this endpoint: - Endpoint barring —Transmission completion	R/W
b30	EPDIS	Endpoint forbidden	Endpoint disable Applications can set this bit to stop sending data on an endpoint even before the transfer on that endpoint is complete. An application must wait until an endpoint inhibit interrupt occurs before an endpoint can be considered inhibited. The module will clear this bit before the endpoint disable interrupt bit is set. An application can set this bit only after the endpoint's endpoint enable bit is set.	R/W
b29	SODDFRM	set odd frames	Set odd frame Applies only to synchronous IN and OUT endpoints. A write to this field sets the Even/Odd Frame (EONUM) field to odd frame.	R/W
b28	SD0PID/ SEVNFRM	set DATA0 PID/ SEVNFRM	Set DATA0 PID (Set DATA0 PID) Applies to interrupt/bulk IN endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. SEVNFRM: Set even frame Applies only to synchronous IN endpoints. A write to this field will set the Even/Odd Frame (EONUM) field to an even frame.	R/W
b27	SNAK	Set the NAK bit	Set NAK bit (Set NAK) A write to this bit sets the endpoint's NAK bit. With this bit, the application can control the sending of the NAK handshake signal on the endpoint. The module can also set this bit of the OUT endpoint to 1 when a transfer completion interrupt occurs or after a SETUP is received on the endpoint	R/W
b26	CNAK	Clear the NAK bit	Clear NAK bit (Clear NAK) A write to this bit clears the endpoint's NAK bit.	R/W
b25~b22	TXFNUM	TxFIFO number	TxFIFO number These bits specify the FIFO number associated with this endpoint. A separate FIFO number must be set for each valid IN endpoint. This field is only valid for IN endpoints.	R/W
b21	STALL	STALL handshake	STALL handshake The application sets this bit to cause the device to reply STALL for all tokens from the USB host. If the NAK bit, global IN NAK, or global OUT NAK is set 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/W
b19~b18	EPTYP	Endpoint type	<p>Endpoint type The following are the transport types supported by this logical endpoint. 00: Control 01: Sync 10: Batch 11: Interrupt</p>	R
b17	NAKSTS	NAK status	<p>NAK status Indicates the following results: 0: The module replies to a non-NAK handshake according to the FIFO status. 1: The module replies with a NAK handshake on this endpoint. When an application or module sets this bit: For asynchronous IN endpoints: The module stops sending any data through the IN endpoint even if there is data available in the TxFIFO. For synchronous IN endpoints: The module sends a zero-length packet even if there is data available in the TxFIFO. Regardless of how this bit is set, the module always responds to the SETUP packet with an ACK handshake.</p>	R
b16	EONUM/ DPID	even/odd frames/ Endpoint data PID	<p>Even/odd frame Applies only to 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 numbers through the SEVNFRM and SODDFRM fields in this register for this endpoint to send/receive isochronous data. 0: even frame 1: Odd frame</p> <p>DPID: Endpoint data PID Applies to interrupt/bulk IN endpoints only. Contains the PID of the packet that will be received or sent on this endpoint. After an endpoint is activated, the application must program the PID of the first packet to be received or sent on this endpoint. The application program uses the SDOPID register field to program the DATA0 or DATA1 PID. 0: DATA0 1: DATA1</p>	R
b15	USBAEP	USB Active Endpoint	<p>USB active endpoint Indicates whether this endpoint is active in the current configuration and interface. The module clears this bit for all endpoints except endpoint 0 when a USB reset is detected. After receiving the SetConfiguration and SetInterface commands, the application must program the endpoint registers accordingly and set this bit.</p>	R/W
b14~b11	Reserved	-	The reset value must be maintained.	R/W
b10~b0	MPSIZ	maximum packet size	<p>Maximum packet size Applications must program this field to the maximum packet size of the current logical endpoint. This value is in bytes.</p>	R/W

34.7.4.16 USBHS Device IN Endpoint x Interrupt Register (USBHS_DIEPINTx) (x=0..15)

USBHS Device IN endpoint x interrupt register

offset address: 0x908 + (endpoint number × 0x20)

Reset value: 0x0000 0080

This register indicates the state of the endpoint in the presence of USB and AHB related events. The application must read this register when the IN endpoint interrupt bit in the module interrupt register (the IEPINT bit in USBHS_GINTSTS) is set. Before an application can read this register, the device-wide endpoint interrupt (USBHS_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 to clear the corresponding bits in the USBHS_DAINT and USBHS_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	NAK	BERR	PKTD RPST S	Reser ved	Reser ved	TXFIF OUD RN	TXFE	INEP NE	Reser ved	TTXF E	TOC	Reser ved	EPDI SD	XFRC	
Bit	Marking	Place name	Function											Read and write	
b31~b14	Reserved	-	The reset value must be maintained.											R/W	
b13	NAK	NAK interrupt	NAK interrupt The module will generate this interrupt when the device sends or receives a NAK. In the case of a synchronous IN endpoint, this interrupt is also generated when a zero-length packet is sent because there is no data in the Tx FIFO to send. Cleared by software writing 1.											R/W	
b12	BERR	Babble error interrupt	Babble error interrupt Cleared by software writing 1.											R/W	
b11	PKTDRPSTS	packet drop status	Packet dropped status This bit is used to indicate to the application that an ISOC OUT packet was dropped. There is no corresponding interrupt mask bit for this bit and no interrupt will be generated. Cleared by software writing 1.											R/W	
b10~b9	Reserved	-	The reset value must be maintained.											R/W	
b8	TXFIFOUDRN	Transmit FIFO underflow	Transmit FIFO underflow (Tx fifo Undr n) This interrupt is generated when the module detects an underflow of the transmit FIFO for this endpoint. Correlation: This interrupt is only valid when the threshold is enabled. Cleared by software writing 1.											R/W	
B7	TXFE	Transmit FIFO is empty	Transmit FIFO empty This interrupt is asserted when the Tx FIFO of this endpoint is half empty or completely empty. Whether the Tx FIFO is half empty or completely empty is determined by the Tx FIFO empty bit in the USBHS_GAHBCFG register (TXFELVL bit in USBHS_GAHBCFG).											R	

			INEPNE: IN endpoint NAK effective This bit can be cleared when the application clears the IN endpoint NAK by writing to the CNAK bit in USBHS_DIEPCTLx. This interrupt indicates that the module has sampled a NAK set (by the application or the module) and the result has taken effect. This interrupt indicates that the IN endpoint NAK bit, set by the application, is active in the module. This interrupt does not guarantee that the NAK handshake signal was sent on the USB. The STALL bit has higher priority than the NAK bit. Software can also write 1 to clear this bit.	R/W
b6	INEPNE	IN endpoint NAK valid		
b5	Reserved	-	The reset value must be maintained.	R/W
b4	TTXFE	IN token received when Tx FIFO is empty	IN token received when Tx FIFO is empty (IN token received when Tx FIFO is empty) Applies to aperiodic IN endpoints only. When the Tx FIFO (periodic/aperiodic) corresponding to this endpoint is empty, an IN token is received and an interrupt is generated. <u>Cleared by software writing 1.</u>	R/W
b3	TOC	time out	Timeout condition Applies only to control IN endpoints. Indicates that this endpoint timed out the response to the most recently received IN token. <u>Cleared by software writing 1.</u>	R/W
b2	Reserved	-	The reset value must be maintained.	R/W
b1	EPDISD	Endpoint Disable Interrupts	Endpoint disabled interrupt This bit indicates that the endpoint has been disabled by the application. <u>Cleared by software writing 1.</u>	R/W
b0	XFRC	transfer complete interrupt	Transfer completed interrupt This field indicates that the transfer set on this endpoint has completed on USB and AHB. <u>Cleared by software writing 1.</u>	R/W

34.7.4.17 USBHS device IN endpoint 0 transfer size register (USBHS_DIEPTSIZE0)

USBHS Device IN endpoint 0 transfer size register

offset address: 0x910

Reset value: 0x0000 0000

The application must modify this register before enabling endpoint 0. After enabling endpoint 0 through the endpoint enable bit (EPENA in USBHS_DIEPCTL0) in the device control endpoint 0 control register, the module modifies this register. The application can read this register only after the module clears the endpoint enable bit.

Non-zero endpoints use the registers of endpoints 1~15.

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]							
<hr/>															
Bit	Marking	Place name	Function								Read and write				
b31~b21	Reserved	-	The reset value must be maintained.								R/W				
b20~b19	PKTCNT	packet count	Packet count Indicates the number of data packets contained in a data transfer of endpoint 0. This field is decremented each time a packet (maximum size or short packet) is read from the TxFIFO.								R/W				
b18~b7	Reserved	-	The reset value must be maintained.								R/W				
b6~b0	XFRSIZ	transfer size	Transfer size Indicates the amount of data contained in a data transfer for endpoint 0, in bytes. The module only interrupts the application when it has finished transferring this data. The transfer size can be set to the endpoint's maximum packet size to break 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				

34.7.4.18 USBHS device IN endpoint x transfer size register (USBHS_DIEPTSIx) (x=1..15)

USBHS 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. The module modifies this register after enabling the endpoint through the endpoint enable bit in the USBHS_DIEPCTLx register (EPENA bit in USBHS_DIEPCTLx). The application can read this register only after the module clears the endpoint enable bit.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Res	MCNT[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]															

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	The reset value must be maintained.	R/W
b30~b29	MCNT	multiple count	Multi count For periodic IN endpoints, this field indicates the number of packets that must be sent per frame on USB. The module uses this field to calculate the data PID of the sync IN endpoint. 01: 1 packet 10: 2 packets 11: 3 packets	R/W
b28~b19	PKTCNT	packet count	Packet count Indicates the number of packets contained in a data transfer on this endpoint. This field is 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 only interrupts the application when it has finished transferring this data. The transfer size can be set to the endpoint's maximum packet size to break 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

34.7.4.19 USBHS device IN endpoint x DMA address register (USBHS_DIEPDMAx) (x=0..15)

USBHS 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 of the device endpoint 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]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b0	DMAADDR	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 storage area pointed to by this field is also used to store control OUT packets and SETUP transaction packets. When more than three SETUP packets are received consecutively, the SETUP packets in the memory will be overwritten. This register is incremented with each AHB transfer. The application must set a doubleword-aligned address.	R/W											

34.7.4.20 USBHS device IN endpoint transmit FIFO status register (USBHS_DTXFSTSx) (x=0..15)

USBHS Device IN endpoint transmit FIFO status register

offset address: 0x918 + (endpoint number × 0x20)

Reset value: 0x0000 0800

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]															
Bit	Marking	Place name	Function										Read and write		
b31~b16	Reserved	-	The reset value must be maintained.										R/W		
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		

34.7.4.21 USBHS Device Control OUT Endpoint 0 Control Register (USBHS_DOEPCTL0)

USBHS Device control OUT endpoint 0 control register

offset address: 0xB00

Reset value: 0x0000 8000

This register is used to control the control transfer endpoint 0.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPEN A	EPDIS	Reserved		SNAK	CNAK		Reserved		STAL L	SNPM	EPTYP[1:0]	NAKS TS	Reser ved		
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
USBA EP							Reserved							MPSIZ[1:0]	

Bit	Marking	Place name	Function	Read and write
b31	EPENA	Endpoint enable	Endpoint enable The application sets this bit to start data reception on endpoint 0. The module clears this bit before triggering any of the following interrupts on this endpoint: — SETUP phase completed - Endpoint barring —Transmission completion	R/W
b30	EPDIS	Endpoint forbidden	Endpoint disable The application cannot disable control of OUT endpoint 0.	R/W
b29~b28	Reserved	-	The reset value must be maintained.	R/W
b27	SNAK	Set the NAK bit	Set NAK bit (Set NAK) A write to this bit sets the endpoint's NAK bit. With this bit, the application can control the sending of the NAK handshake signal on the endpoint. The module can also set this bit of the endpoint after the endpoint receives the SETUP packet.	R/W
b26	CNAK	Clear the NAK bit	Clear NAK bit (Clear NAK) A write to this bit clears the endpoint's NAK bit.	R/W
b25~b22	Reserved	-	The reset value must be maintained.	R/W
b21	STALL	STALL handshake	STALL handshake When this endpoint receives a SETUP token, the application can only set this bit and the module will clear it. If the NAK bit, global OUT NAK and this bit are set at the same time, the STALL bit takes precedence. Regardless of how this bit is set, the module always responds to the SETUP packet with an ACK handshake.	R/W
b20	SNPM	monitor mode	Snoop mode This bit is used to configure the endpoint to listen mode. In listen mode, the module does not check for correct OUT packets before transferring them to the application store.	R/W
b19~b18	EPTYP	Endpoint type	Endpoint type Hardware is set to '00', indicating a control-type endpoint.	R/W
b17	NAKSTS	NAK status	NAK status Indicates the following results: 0: The module replies to a non-NAK handshake according to the FIFO status. 1: The module replies with a NAK handshake on this endpoint. When an application or module sets this bit to 1, the module stops receiving data even if there is room in the RxFIFO to continue to accommodate received packets. Regardless of how this bit is set, the module always responds to the SETUP packet with an ACK handshake.	R
b16	Reserved	-	The reset value must be maintained.	R/W
b15	USBAEP	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/W

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
-------	-------	---------------------	--	-----

34.7.4.22 USBHS device OUT endpoint x control register (USBHS_DOEPCTLx) (x=1..15)

USBHS Device OUT endpoint x control register

offset address: 0xB00 + (endpoint number × 0x20)

Reset value: 0x0000 0000

Applications use this register to control the behavior of individual logical endpoints (except endpoint 0).

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPEN A	EPDI S	SOD DFRM / SD1P ID	SD0P ID/ SEVN FRM	SNAK	CNAK		Reserved		STAL L	SNPM	EPTYP[1:0]	NAKS TS	EONU M/ DPID		
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
USBA EP		Reserved									MPSIZ[10:0]				

Bit	Marking	Place name	Function	Read and write
b31	EPENA	Endpoint enable	Endpoint enable Set by software, USBHS is cleared 0: Endpoint disable 1: Endpoint enable	R/W
b30	EPDIS	Endpoint forbidden	Endpoint disable Applications can set this bit to stop sending/receiving data on an endpoint even before the transfer on that endpoint is complete. An application must wait until an endpoint inhibit interrupt occurs before an endpoint can be considered inhibited. The module will clear this bit before the endpoint disable interrupt bit is set. An application can set this bit only after the endpoint's endpoint enable bit is set.	R/W
b29	SD1PID/ SODDFRM	set DATA1 PID/set odd frame	Set DATA1 PID (Set DATA1 PID) Applies to interrupt/bulk OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1.	R/W
b28	SD0PID/ SEVNFRM	set DATA0 PID/ SEVNFRM	SODDFRM: Set odd frame Applies only to synchronous OUT endpoints. A write to this field sets the Even/Odd Frame (EONUM) field to odd frame. Set DATA0 PID (Set DATA0 PID) Applies to interrupt/bulk OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0.	R/W
b27	SNAK	Set the NAK bit	SEVNFRM: Set even frame Applies only to synchronous OUT endpoints. A write to this field will set the Even/Odd Frame (EONUM) field to an even frame. Set NAK bit (Set NAK) A write to this bit sets the endpoint's NAK bit. With this bit, the application can control the sending of the NAK handshake signal on the endpoint. The module can also set this bit on the OUT endpoint when a transfer complete interrupt occurs or after a SETUP is received on the endpoint.	R/W
b26	CNAK	Clear the NAK bit	Clear NAK bit (Clear NAK) A write to this bit clears the endpoint's NAK bit.	R/W
b25~b22	Reserved	-	The reset value must be maintained.	R/W
b21	STALL	STALL handshake	STALL handshake When this endpoint receives a SETUP token, the application can only set this bit and the module will clear it. If the NAK bit, global OUT NAK and this bit are set at the same time, the STALL bit takes precedence. Only the application can clear this bit, not the module. Snoop mode	R/W
b20	SNPM	monitor mode	This bit is used to configure the endpoint to listen mode. In monitor mode, the module will no longer check the correctness of the received data.	R/W

			Endpoint type The following are the transport types supported by this logical endpoint. 00: Control 01: Sync 10: Batch 11: Interrupt	
b19~b18	EPTYP	Endpoint type	00: Control 01: Sync 10: Batch 11: Interrupt	R/W
b17	NAKSTS	NAK status	NAK status Indicates the following results: 0: The module replies to a non-NAK handshake according to the FIFO status. 1: The module replies with a NAK handshake on this endpoint. When an application or module sets this bit: The module stops receiving any data on the OUT endpoint even if there is room in the RxFIFO for incoming packets. Regardless of how this bit is set, the module always responds to the SETUP packet with an ACK handshake.	R
b16	EONUM/ DPID	even/odd frames/ Endpoint data PID	Even/odd frame Applies only to 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 numbers through the SEVNFRM and SODDFRM fields in this register for this endpoint to transmit/receive synchronously data. 0: even frame 1: Odd frame DPID: Endpoint data PID Applies to interrupt/bulk OUT endpoints only. Contains the PID of the packet that will be received or sent on this endpoint. After an endpoint is activated, the application must program the PID of the first packet to be received or sent on this endpoint. The application program uses the SD0PID register field to program the DATA0 or DATA1 PID. 0: DATA0 1: DATA1	R
b15	USBAEP	USB Active Endpoint	USB active endpoint Indicates whether this endpoint is active in the current configuration and interface. The module clears this bit for all endpoints except endpoint 0 when a USB reset is detected. After receiving the SetConfiguration and SetInterface commands, the application must program the endpoint registers accordingly and set this bit.	R/W
b14~b11	Reserved	-	The reset value must be maintained.	R/W
b10~b0	MPSIZ	maximum packet size	Maximum packet size Applications must program this field to the maximum packet size of the current logical endpoint. This value is in bytes.	R/W

34.7.4.23 USBHS device OUT endpoint x interrupt register (USBHS_DOEPINTx) (x=0..15)

USBHS Device OUT endpoint x interrupt register

offset address: 0xb08 + (endpoint number × 0x20)

Reset value: 0x0000 0080

This register indicates the state of the endpoint in the presence of USB and AHB related events. The application must read this register when the OUT endpoint interrupt bit in the USBHS_GINTSTS register (OEPINT bit in USBHS_GINTSTS) is set. Before an application can read this register, the USBHS_DAINT register must be read to obtain the exact endpoint number of the USBHS_DOEPINTx registers. The application must clear the corresponding bits in this register to clear the corresponding bits in the USBHS_DAINT and USBHS_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	NYET	Reserved						B2BS TUP	Reserved	OTEP DIS	STUP	Reserved	EPDI SD	XFRC	
<hr/>															
Bit	Marking	Place name		Function								Read and write			
b31~b15	Reserved	-		The reset value must be maintained.								R/W			
b14	NYET	NYET outage		The module will generate this interrupt when the asynchronous OUT endpoint replies to the NYET handshake signal. Software can also write 1 to clear this bit.								R/W			
b13~b7	Reserved	-		The reset value must be maintained.								R/W			
b6	B2BSTUP	Received consecutive SETUP packets		Back-to-back SETUP packets received Applies only to control OUT endpoints. This bit indicates that the endpoint has received more than three consecutive SETUP packets. Software can also write 1 to clear this bit.								R/W			
b5	Reserved	-		The reset value must be maintained.								R/W			
b4	OTEPDIS	OUT token received when endpoint is disabled		OUT token received when endpoint disabled Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled, thereby generating an interrupt. Cleared by software writing 1.								R/W			
b3	STUP	SETUP phase completed		SETUP phase done Applies only to control OUT endpoints. Indicates that the SETUP phase of the control endpoint is complete and that no consecutive SETUP packets are being received in the current control transfer. On this interrupt, the application can decode the received SETUP packet. Cleared by software writing 1.								R/W			
b2	Reserved	-		The reset value must be maintained.								R/W			
b1	EPDISD	Endpoint Disable Interrupts		Endpoint disabled interrupt This bit indicates that the endpoint has been disabled by the application. Cleared by software writing 1.								R/W			
b0	XFRC	transfer complete interrupt		Transfer completed interrupt This field indicates that the transfer set on this endpoint has completed on USB and AHB. Cleared by software writing 1.								R/W			

34.7.4.24 USBHS device OUT endpoint 0 transfer size register (USBHS_DOEPTSIZ0)

USBHS Device OUT endpoint 0 transfer size register

offset address: 0xB10

Reset value: 0x0000 0000

The application must modify this register before enabling endpoint 0. After enabling endpoint 0 through the endpoint enable bit (EPENA in USBHS_DIEPCTL0) in the device control endpoint 0 control register, the module modifies this register. The application can read this register only after the module clears the endpoint enable bit.

Non-zero endpoints use the registers of endpoints 1~15.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved	STUPCNT[1:0]		Reserved								PKTCNT	Reserved			
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
Reserved								XFRSIZ[6:0]							
<hr/>															
Bit	Marking	Place name	Function								Read and write				
b31	Reserved	-	The reset value must be maintained.								R/W				
b30~b29	STUPCNT	SETUP packet count	SETUP packet count This field specifies the number of SETUP packets that the endpoint can continuously receive. 01: 1 packet 10: 2 packets 11: 3 packets								R/W				
b28~b20	Reserved	-	The reset value must be maintained.								R/W				
b19	PKTCNT	packet count	Packet count The number of packets that should be received in one transmission. Before the endpoint is enabled, the software sets this bit. After the transmission starts, the value of this field is automatically decremented each time a data packet is received.								R/W				
b18~b7	Reserved	-	The reset value must be maintained.								R/W				
b6~b0	XFRSIZ	transfer size	Transfer size Indicates the amount of data contained in a data transfer for endpoint 0, in bytes. The module only interrupts the application when it has finished transferring this data. The transfer size can be set to the endpoint's maximum packet size to break at the end of each packet. The module decrements this field each time a packet is read from the RxFIFO and written to external memory.								R/W				

34.7.4.25 USBHS device OUT endpoint x transfer size register (USBHS_DOEPTSIZx) (x=1..15)

USBHS 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. This register is modified by the module after the endpoint is enabled through the endpoint enable bit in the USBHS_DOEPCTLx register (EPENA bit in USBHS_DOEPCTLx). The application can read this register only after the module clears the endpoint enable bit.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Res	RXDPID[1:0]/S TUPCNT[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]															

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	The reset value must be maintained.	R/W
b30~b29	RXDPID/ STUPCNT	Received data PID/SETUP packet count	Received data PID (Received data PID) Applies only to synchronous OUT endpoints. This is the PID of the last packet received by this endpoint. 00: DATA0 01: DATA2 10: DATA1 11: MDATA STUPCNT: SETUP packet count Applies only to control OUT endpoints. This field specifies the number of SETUP packets that the endpoint can continuously receive. 01: 1 packet 10: 2 packets 11: 3 packets	R/W
b28~b19	PKTCNT	packet count	Packet count Indicates the number of packets contained in a data transfer on this endpoint. This field is decremented after each write packet (maximum size or short packet) to the 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. The module only interrupts the application when it has finished transferring this data. The transfer size can be set to the endpoint's maximum packet size to break at the end of each packet. The module decrements this field each time a packet is read from the RxFIFO and written to external memory.	R/W

34.7.4.26 USBHS device OUT endpoint x DMA address register (USBHS_DOEPDMAx) (x=0..15)

USBHS 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 of the device endpoint 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]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b0	DMAADDR	DMA address	DMA address This bit contains the starting address of the external memory area when using DMA for data transfers on the endpoint. Note: For control endpoints, the storage area pointed to by this field is also used to store control OUT packets and SETUP transaction packets. When more than three SETUP packets are received consecutively, the SETUP packets in the memory will be overwritten. This register is incremented with each AHB transfer. The application must set a doubleword-aligned address.	R/W											

34.7.5 USBHS Clock Gating Control Register

The HCLK and PHY clocks are controlled by gated clock control registers to reduce power consumption. Unless otherwise specified, bit values in register descriptions are represented in binary.

34.7.5.1 USBHS Clock Gating Control Register (USBHS_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															
Bit	Marking	Place name	Function												
b31~b8	Reserved	-	The reset value must be maintained.												
B7	SUSP	Deep Sleep status bits	Deep Sleep (L1 suspended) status bit												
b6	PHYSLEEP	PHY SLEEP Mode Status Bits	PHY SLEEP Mode Status Bit (PHY in sleep) This bit is used to indicate that the PHY is in the SLEEP mode state.												
b5	ENL1GTG	L1 clock gating	L1 Clock Gating (Enable Sleep Clock Gating) This bit is used for clock gating control in L1 mode.												
b4~b2	Reserved	-	The reset value must be maintained.												
b1	GATEHCLK	Gated HCLK	Gate HCLK (Gate HCLK) The application sets this bit to 1 to stop clocking modules other than the AHB bus slave interface, master interface, and wake-up logic when USB communication is suspended or the session is invalid. The application clears this bit when the USB resumes communication or a new session starts.												
b0	STPPCLK	stop PHY clock	Stop PHY clock (Stop PHY clock) The application sets this bit to stop the PH clock when USB communication is suspended, the session is invalid, or the device is disconnected. The application clears this bit when USB resumes communication.												

35 USB2.0 Full Speed Module (USBFS)

35.1 Introduction to USBFS

The USB Full Speed (USBFS) controller provides a set of USB communication solutions for portable devices. The USBFS controller supports host mode and device mode, and the chip integrates a full-speed PHY. In host mode, the USBFS controller supports full-speed (FS, 12Mb/s) and low-speed (LS, 1.5Mb/s) transceivers, while in device mode only full-speed (FS, 12Mb/s) transceivers are supported. The USBFS controller supports all four transfer modes defined by the USB 2.0 protocol (control transfer, bulk transfer, interrupt transfer, and isochronous transfer). The USBFS controller supports LPM (Link Power Management) function.

Follow the protocol as follows:

- Universal Serial Bus Revision 2.0 Specification
- USB 2.0 Link Power Management Addendum Engineering Change Notice to the USB 2.0 specification, July 16, 2007
- Errata for USB 2.0 ECN: Link Power Management (LPM) - 7/2007

35.2 Main Features of USBFS

Mainly divided into three categories: Common attributes, host mode attributes, and device mode attributes.

35.2.1 Common features

- Built-in on-chip USB2.0 full-speed PHY
- Support host mode and device mode
- Supports FS SOF as well as low-speed "Keep-alive" tokens and has the following features:
 - SOF pulse pin output function
 - SOF pulse can be used as the internal event source of the chip to trigger TIMER, DMA and other modules to work
 - Configurable frame period
 - Configurable end of frame interrupt
- The module has built-in DMA, and the AHB burst transfer type can be configured by software
- Power saving features such as USB suspend, stop RAM clock, stop PHY domain clock
- Features 2.5KB dedicated RAM with advanced FIFO control
 - The RAM space can be divided into different FIFOs for flexible and efficient use of RAM
 - Each FIFO can store multiple packets
 - Dynamically allocate memory
 - The size of the FIFO can be configured to a non-power-of-2 value for continuous use of memory cells

- No application intervention can be required within a frame to achieve maximum USB bandwidth
- The host mode or device mode can be automatically determined according to the level of the ID line

35.2.2 Host Mode Features

- Host mode supports USB2.0 full-speed (FS, 12Mb/s) and low-speed (LS, 1.5Mb/s) transfers
- The VBUS voltage needs to be generated by an external power chip
- Up to 16 host channels (pipes): Each channel can be dynamically reconfigured to support any type of USB transfer
- Built-in hardware scheduler to:
 - Stores up to 8 interrupts plus isochronous transfer requests in a periodic hardware queue
 - Stores up to 8 control plus bulk transfer requests in aperiodic hardware queues
- Manages one shared RX FIFO, one periodic TX FIFO, and one aperiodic TX FIFO for efficient use of USB data RAM

35.2.3 Device Mode Features

- Slave mode supports USB2.0 full speed (FS, 12Mb/s) transfer.
- 1 bidirectional control endpoint 0
- 15 OUT endpoints that can be configured to support bulk, interrupt, or isochronous transfers
- 15 IN endpoints that can be configured to support bulk, interrupt, or isochronous transfers
- Contains 16 transmit FIFOs (one for each 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.

35.3 USBFS System Block Diagram

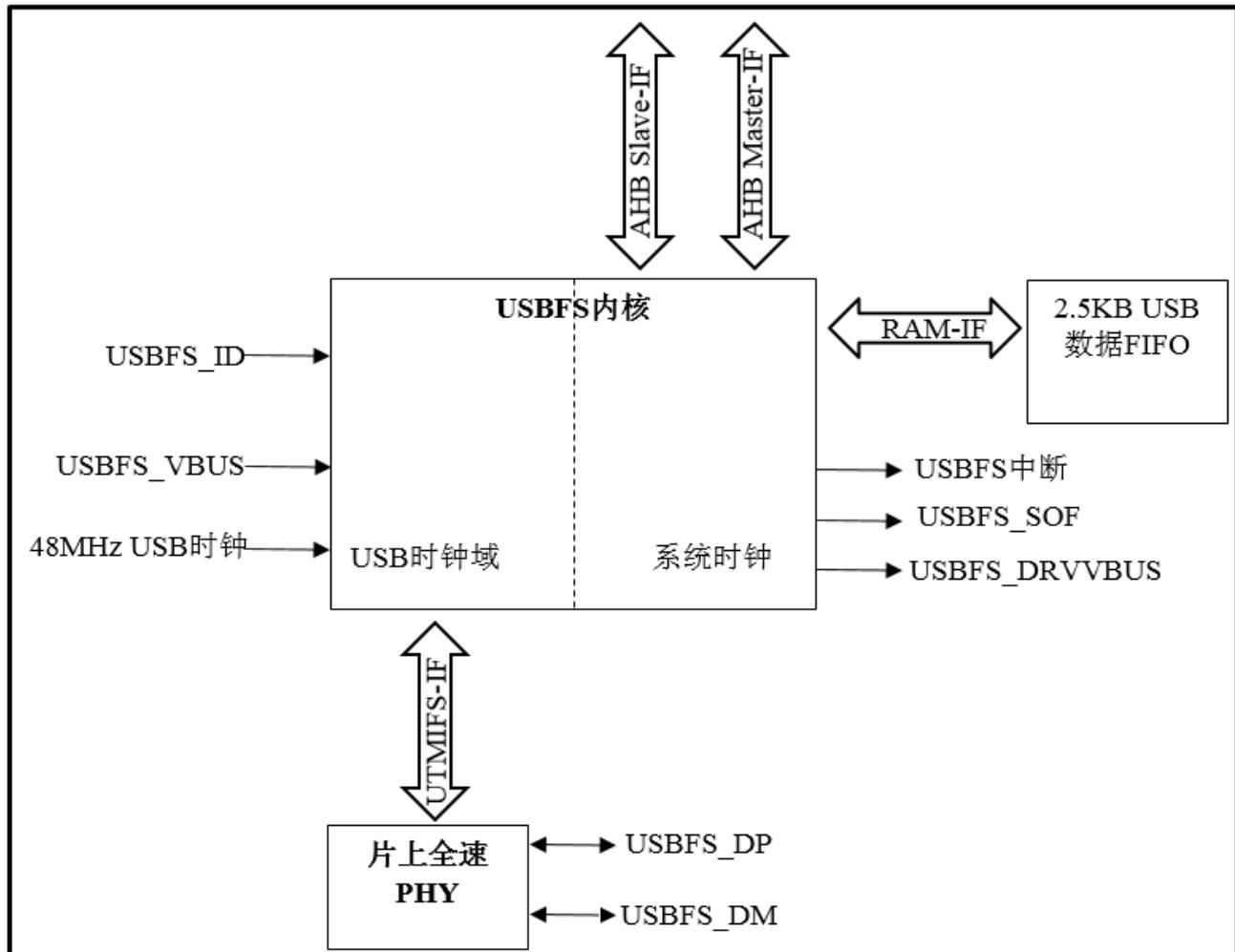


Figure 35-1 USBFS system block diagram

35.4 USBFS pin description

Table 35-1 USBFS pin description

Pin name	Direction	Applicable mode	corresponding pin	Functional description
USBFS_VBUS	Input	equipment	PA9	Power port, 5V withstand voltage
USBFS_DP	Input/Output	Host, device	PA12	Differential data D+ signal
USBFS_DM	Input/Output	Host, device	PA11	Differential Data D-Signal
USBFS_DRVVBUS	Output	Host	PC9, PB8	External power chip enable signal
USBFS_ID	Input	Host	PA10	USB AB device identification signal
USBFS_SOF	Output	Host, device	PA8	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 functions of their corresponding pins when using USBFS. The USBFS_DP and USBFS_DM functions are analog functions and have nothing to do with the corresponding PFSR register settings. For details, please refer to 011. General IO (GPIO) chapter. In addition, when the USBFS function is not used, additional current consumption will be generated when the digital function pins corresponding to the USBFS_DP and USBFS_DM pins are flipped.

35.5 USBFS function description

35.5.1 USBFS clock and working mode

The clock used by USBFS needs to be configured as 48MHz. The 48MHz clock is generated by the internal PLL circuit. The PLL clock source needs to select an external high-speed oscillator. Before using the USBFS module, the USBFS clock needs to be configured in the CMU module.

USBFS can be used as a host or device and includes an on-chip full-speed PHY.

Pull-up and pull-down resistors have been integrated inside the on-chip full-speed PHY, and the USBFS can be automatically selected based on the current mode and connection status.

When USBFS is working, the VCC voltage range is 3.0~3.6V.

35.5.2 USBFS mode decision

There are two ways for USBFS to determine the current working mode:

Method 1: Automatically identify according to the state of the USBFS_ID line, when it is detected that the USBFS_ID line is at a high level, the module works in the device mode, and when it is detected that the USBFS_ID line is at a low level, the module works in the host state.

Method 2: Force the host/device mode, by setting the FDMOD or FHMOD bit of the register USBFS_GUSBCFG to 1, so that the module ignores the level of the USBFS_ID line and is forced to work in the device or host mode.

35.5.3 USBFS host function

35.5.3.1 Mainframe function introduction

When USBFS works in host mode, VBUS is the 5V power pin specified by the USB protocol. The internal PHY does not support 5V power supply, so an external USB power chip is required to power the device. USBFS_DRVVBUS is used to enable the external USB power chip, and the overcurrent detection of the external power chip can be realized through the external interrupt IRQ of this MCU. USBFS_VBUS can be used as GPIO in host mode.

A typical USB host mode system construction diagram is as follows:

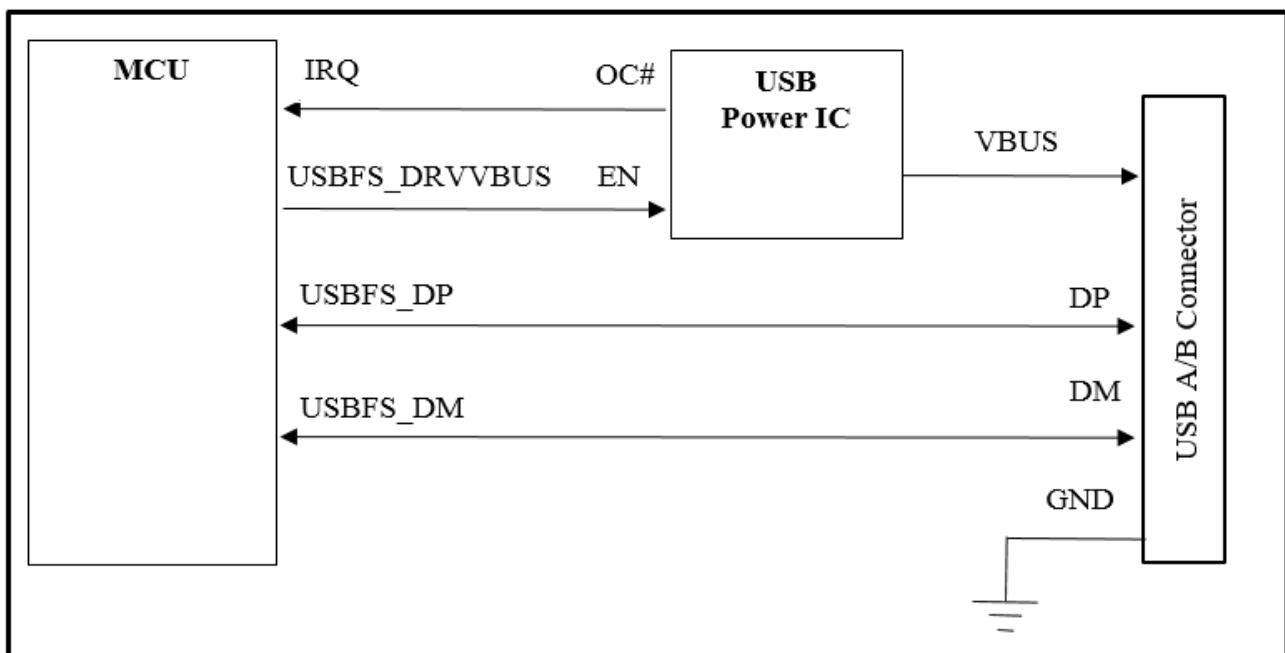


Figure 35-2 USBFS host mode system construction diagram

35.5.3.2 Host port power supply

This MCU cannot output 5V to provide VBUS. To do this, a USB power chip or basic power switch (such as a 5V supply from the application board) must be added outside the microcontroller to drive the 5V VBUS line. The external power chip can be driven by any GPIO output or USBFS_DRVVBUS. When the application determines to use the GPIO to control the external device to provide VBUS, the port power bit in the host port control and status register (PWPR bit in USBFS_HPRT) must still be set.

35.5.3.3 Host detects device connection and disconnection

USB devices will be detected as soon as they are connected. 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 (the PCDET bit in USBFS_HPRT).

A device disconnect event will trigger a disconnect detection interrupt (DISCINT bit in USBFS_GINTSTS).

35.5.3.4 Host enumeration

After a device connection is detected, if a new device is connected, the host must initiate the enumeration process by sending USB reset and configuration commands to the new device.

The application drives the USB reset signal (single-ended zero) through the USB by setting the port reset bit in the host port control and status register (PRST bit in USBFS_HPRT) for a minimum of 10ms and a maximum of 20ms . The application calculates the duration of this process and then clears the port reset bit.

Immediately after the USB reset sequence is complete, the port enable/disable change bit (PENCHNG bit in USBFS_HPRT) triggers a host port interrupt, which in turn notifies the application that it can be accessed from the port speed field in the host port control and status register (in USBFS_HPRT PSPD) read the enumerated device speed, and the host has started driving SOF (FS) or Keep-alive tokens (LS). At this point the host is ready to enumerate the device by sending commands to the device.

35.5.3.5 Host hangs

The application suspends the USB bus by setting the port suspend bit (PSUSP in USBFS_HPRT) in the host port control and status register. The USBFS module stops sending SOF and enters the suspend state.

The bus can be brought out of the suspend state by autonomous activity of the remote device (remote wakeup). In this case, the remote wakeup signal will trigger the remote wakeup interrupt (WKUPINT bit in USBFS_GINTSTS), the hardware will reset the port recovery bit in the host port control and status register (PRES bit in USBFS_HPRT) by itself, and automatically drive recovery through USB Signal. The application must time the resume window, then clear the port resume bit to exit the suspend state and restart the SOF.

If exit suspend is initiated by the host, the application must set the port resume bit to initiate the resume signal on the host port, time the resume window and eventually clear the port resume bit.

35.5.3.6 Host channel

The USBFS module implements 16 host channels. Each host channel can be used for USB host transfers (USB pipes). The host can handle up to 8 transfer requests at the same time. If the application has more than 8 pending transfer requests, after the channel is released from the previous task (that is, after receiving the transfer complete and channel stop interrupt), the host controller driver (HCD) must restart the Allocate channels.

Each host channel can be configured to support input/output and periodic/aperiodic transactions. Each host channel uses dedicated control (HCCHARx) registers, transfer configuration (HCTSIZx) registers/interrupt (HCINTx) registers, and their associated interrupt mask registers (HCINTMSKx).

Host channel control

The application program has the following control over the host channel through the host channel x characteristics register (HCCHARx):

- Channel enable/disable
- Set the speed of the target USB device: FS/LS
- Set the address of the target USB device
- Sets 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 transfer on this channel: Control/Batch/Interrupt/Sync
- Sets the maximum packet length of device endpoints communicating with this channel
- Set the frame to be transmitted periodically: Odd frame/even frame

Host channel transmission

The host channel transfer size registers (HCTSIZx) allow the application to program the transfer size parameter and read the transfer status. This register must be set before the channel enable bit in the host channel characteristics register is set. After the endpoint is enabled, the packet count field becomes read-only immediately, and the USBFS module updates the field according to the current transfer status.

The following transfer parameters can be programmed:

- transfer size in bytes
- The number of packets that make up the entire transfer size
- Initial data PID

Host channel status/interrupt

The Host Channel x Interrupt Register (HCINTx) indicates the state of the endpoint when USB and AHB related events occur. When the host channel interrupt bit in the interrupt register (HCINT bit in USBFS_GINTSTS) is set, the application must read these registers for detailed information. Before reading these registers, the application must 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 appropriate bits in this register to clear the corresponding bits in the HAINT and GINTSTS registers. The USBFS_HCINTMSK x registers also provide mask bits for each interrupt source per channel.

The host module provides the following status checking and interrupt generation functions:

- Transfer complete interrupt, indicating that both the application (AHB) and the USB side have completed data transfer
- Channel stopped due to transfer complete, USB transaction error, or application issued a disable command
- The associated transmit FIFO is half empty or completely empty (IN endpoint)
- ACK response received
- NAK response received
- STALL response received
- USB transaction errors due to CRC check failures, timeouts, bit stuffing errors, and bad EOPs
- crosstalk error
- frame overflow
- Error in toggle bit for data synchronization

35.5.3.7 Host scheduler

The host module has a built-in hardware scheduler that can autonomously reorder and manage USB transaction requests issued by the application. At the beginning of each frame, the host performs periodic (sync and interrupt) transactions followed by aperiodic (control and bulk) transactions to comply with the USB specification's high priority guarantee for isochronous and interrupt transfers.

The host processes USB transactions through request queues (one periodic request queue and one aperiodic request queue). Each request queue can store up to 8 entries. Each entry represents a USB transaction request initiated by an application but has not yet received a response, and stores the number of the IN or OUT channel used to execute the USB transaction, as well as other relevant information. The order in which USB transaction requests are written in the queue determines the order in which transactions are executed on the USB interface.

At the beginning of each frame, the host processes the periodic request queue first, and then processes the aperiodic request queue. If, at the end of the current frame, a synchronous or interrupt-type USB transfer transaction request scheduled to be performed on the current frame is still pending, the host will issue an outstanding periodic transfer interrupt (IPXFR bit in USBFS_GINTSTS). The USBFS module is responsible for the management of periodic and aperiodic request queues. The periodic transmit FIFO and queue status register (HPTXSTS) and the aperiodic transmit FIFO and queue status register (HNPTXSTS) are read-only registers that the application can use to read the status of each request queue. These include:

- Number of free entries currently available in the periodic (aperiodic) request queue (up to 8)
- Free space currently available in periodic (aperiodic) TxFIFO (OUT transactions)

- IN/OUT tokens, host channel numbers, and other status information

Since each request queue can store up to 8 USB transaction requests, the application can send the host USB transaction request to the scheduler in advance; Appears on the USB bus after an acyclic transaction is complete.

To issue a transaction request to the host scheduler (queue), the application must read the PTXQSAV bit in the USBFS_HPTXSTS register or the NPTQXSAV bit in the USBFS_HNPTXSTS register, ensuring that there is at least one free space in the queue for periodic (aperiodic) requests to store the current ask.

35.5.4 USBFS Device Features

35.5.4.1 Device function introduction

When USBFS works in device mode, VBUS is the 5V power supply pin specified by the USB protocol and is a 5V withstand voltage pin. This module always detects the level state of the VBUS line to connect or disconnect the device.

A typical USB device mode system construction diagram is as follows:

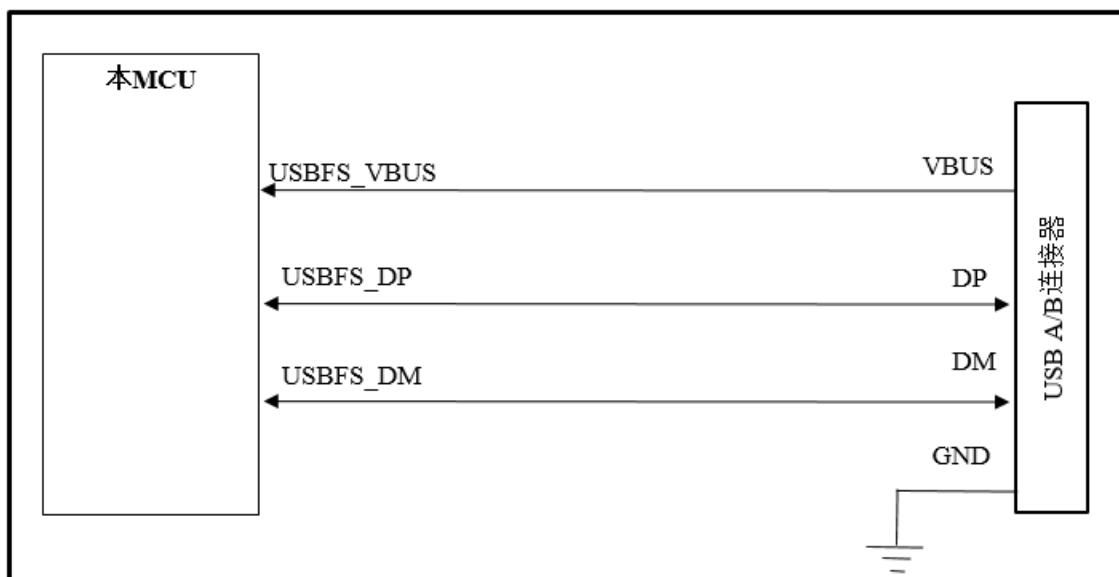


Figure 35-3 USBFS device mode system construction diagram

35.5.4.2 Device power status

When the module detects that USBFS_VBUS is high, it will make the USB device enter the power supply state. Then, USBFS automatically connects the DP pull-up resistor, sends a signal that the full-speed device is connected to the host, and generates a session request interrupt (VBUSVINT bit in USBFS_GINTSTS), indicating that it enters the power supply state.

Additionally, the USBFS_VBUS input ensures that the host provides a valid VBUS level during USB operation. USBFS will automatically disconnect if it detects that the VBUS level is low (for example, due to power disturbance or host port shutdown).

In the powered state, the USBFS expects to receive a reset signal from the host. Other USB operations cannot be performed. A reset detected interrupt (USBRST in USBFS_GINTST) is generated as soon as the reset signal is received. After the reset signal ends, an enumeration complete interrupt (ENUMDNE bit in USBFS_GINTSTS) is generated, and USBFS enters the default state.

35.5.4.3 Device default state

By default, USBFS expects a SET_ADDRESS command from the host. Other USB operations cannot be performed. When a valid SET_ADDRESS command is decoded on the USB, the application will write the corresponding address value to the device address field in the device configuration register (DAD bit in USBFS_DCFG). The USBF then enters the address state and is ready to respond to host transactions with the configured USB address.

35.5.4.4 Device suspend state

USBFS devices continuously monitor USB activity. After the USB idle time reaches 3ms, the early suspend interrupt (bit ESUSP in USBFS_GINTSTS) will be issued, and 3ms later by the suspend interrupt (bit USBSUSP in USBFS_GINTSTS) to confirm that the device enters the suspended state. Then, the device suspend bit in the device status register (SUSPSTS bit in USBFS_DSTS) is automatically set to 1, and USBFS enters the suspend state immediately.

The suspend state can be exited through the device itself. In this case, the application will set the Remote Wakeup Signal bit in the Device Control Register (RWUSIG bit in USBFS_DCTL) and clear it within 1ms to 15ms.

But if the device detects the resume signal sent by the host, it will generate a resume interrupt (WKUPINT bit in USBFS_GINTSTS), and the device suspend bit is automatically cleared.

35.5.4.5 Device soft disconnect

The power state can be exited by software with the help of the soft disconnect function. The DP pull-up resistor can be removed by setting the soft disconnect bit in the device control register (SDIS bit in USBFS_DCTL) to 1, at which point a device disconnect occurs on the host side even though the USB cable is not actually unplugged from the host port Detect interruption.

35.5.4.6 Device endpoint

Endpoint class

The USBFS module implements the following USB endpoints:

- Control endpoint 0:
 - Bidirectional and only process control messages
 - Use a separate set of registers to handle IN and OUT transactions

- Dedicated control (USBFS_DIEPCTL0/USBFS_DOEPCTL0) registers, transfer configuration (USBFS_DIEPTSIZ0/USBFS_DIEPTSIZ0) registers, and status interrupt (USBFS_DIEPINTx/USBFS_DOEPINT0) registers. The set of bits available in the control and transfer size registers is slightly different than in other endpoints
- 15 IN endpoints
 - Each endpoint can be configured to support isochronous, bulk or interrupt transfer types
 - Each endpoint has dedicated control (USBFS_DIEPCTLx) registers, transfer configuration (USBFS_DIEPTSIZx) registers, and status interrupt (USBFS_DIEPINTx) registers
 - Device IN Endpoint General Interrupt Mask Register (USBFS_DIEPMSK) can be used to enable/disable the same type of endpoint interrupt sources on all IN endpoints (including EP0)
 - Supports outstanding isochronous IN transfer interrupt (IISOIXFR bit in USBFS_GINTSTS), which will trigger when there is an outstanding transfer on at least one isochronous IN endpoint in the current frame. This interrupt is triggered together with the periodic frame interrupt (USBFS_GINTSTS/EOPF)
- 15 OUT endpoints
 - Each endpoint can be configured to support isochronous, bulk or interrupt transfer types
 - Each endpoint has dedicated control (USBFS_DOEPCTLx) registers, transfer configuration (USBFS_DOEPTSIZx) registers, and status interrupt (USBFS_DOEPINTx) registers
 - The device OUT endpoint general interrupt mask register (USBFS_DOEPMSK) can be used to enable/disable the same type of endpoint interrupt sources on all OUT endpoints (including EP0)
 - Supports outstanding isochronous OUT transfer interrupt (INCOMPISOOUT bit in USBFS_GINTSTS), which will trigger when there is an outstanding transfer on at least one isochronous 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 of the endpoint through the device endpoint x IN/OUT control registers (DIEPCTLx/DOEPCTLx):

- Endpoint enable/disable
- Activate endpoint with current configuration
- Set USB transfer type (Sync, Bulk and Interrupt)
- Set the supported packet size
- Set the Tx-FIFO number associated with the IN endpoint
- Set the data0/data1 PID you want to receive or use when sending (only for bulk/interrupt transfers)
- Sets the odd/even frame to which a transaction is received or sent (isochronous transfers only)

- The NAK bit can be set so that no matter what the state of the FIFO is at this time, NAK is replied to the host's request
- The STALL bit can be set so that the host's token for that endpoint is STALL returned by hardware
- The OUT endpoint can be set to listen mode, i.e. no CRC check on the received data

Endpoint transfer

The device endpoint x transfer size registers (DIEPTSIZx/DOEPTSIZx) allow applications to program transfer size parameters and read transfer status. This register must be set before the endpoint enable bit in the endpoint control register is set. After enabling the endpoint, these fields become read-only immediately, and the USBFS module updates these fields according to the current transfer status.

The following transfer parameters can be programmed:

- transfer size in bytes
- The number of packets that make up the entire transmission

Endpoint Status/Status

The Device Endpoint x Interrupt Register (DIEPINTx/DOEPINTx) indicates the state of the endpoint upon USB and AHB related events. When the OUT endpoint interrupt bit or the IN endpoint interrupt bit in the module interrupt register (the OEPINT bit in USBFS_GINTSTS or the IEPINT bit in USBFS_GINTSTS, respectively) is set, the application must read these registers for detailed information. Before the application can read these registers, the device-wide 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 appropriate bits in this register to clear the corresponding bits in the DAINT and GINTSTS registers.

The module provides the following status checking and interrupt generation functions:

- Transfer complete interrupt, indicating that both the application AHB and the USB side have completed the data transfer
- Setup phase completed (only for OUT endpoints of control transfer type)
- The associated transmit FIFO is half empty or completely empty (IN endpoint)
- NAK reply sent to host (IN endpoint for isochronous transfers only)
- IN token received when TxFIFO is empty (only for IN endpoints of bulk and interrupt transfer types)
- OUT token received when endpoint is not yet enabled
- babble error detected
- Application shutdown endpoint takes effect
- The application sets NAK to the endpoint to take effect (only for IN endpoints of isochronous transmission type)

- Received more than 3 consecutive setup packets (only for control type OUT endpoints)
- Timeout condition detected (only for IN endpoints of control transfer type)

35.5.5 USBFS SOF pulse output function

USBFS can monitor, track and configure SOF frames in both host and device mode and also has SOF pulse output function. The SOF pulse is output through the USBFS_SOF pin, and the output width is 16 system clock cycles.

35.5.5.1 Host SOF

In host mode, the number of PHY clocks that occur during two consecutive SOF (FS) or keep-alive (LS) tokens generated can be programmed in the host frame interval register (HFIR), allowing the application to program the SOF frame period is controlled. Interrupts are generated at the start of a frame (SOF bit in USBFS_GINTSTS). The current frame number and the time remaining until the next SOF can be tracked by the application in the host frame number register (HFNUM).

Using the SOFEN bit in the USBFS system control register USBFS_SYCTLREG, the SOF pulse signal with a width of 16 system clock cycles that is generated when any SOF token is issued can be output from the USBFS_SOF pin.

In addition, the SOF pulse can also work as an internal event-triggered DMA transfer, TIMER count and other external modules.

35.5.5.2 Device SOF

In device mode, the start of frame interrupt (SOF bit in USBFS_GINTSTS) is triggered every time the USB receives a SOF token. 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, it is also possible to generate a SOF pulse signal with a width of 16 system clock cycles and make this signal output on the USBFS_SOF pin for external availability.

In addition, the SOF pulse can also work as an internal event-triggered DMA transfer, TIMER count and other external modules.

Periodic end-of-frame interrupts (GINTSTS/EOPF) are 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 (USBFS_DCFG in the PFIVL bit). This function can be used to determine if all isochronous communications for the frame are complete.

35.5.6 USBFS power control

When the USBFS module is not used, the HCLK and PHY clocks of the USBFS module can be stopped through the CMU module to reduce power consumption.

When using the USB module, but the device USB session is not started or the device is not connected, the power reduction technique can be used in the USB suspend state.

- Stop PHY clock (STPPCLK bit in USBFS_GCCTL)

When the Stop PHY Clock bit in the Clock Gating Control register is set, most of the 48 MHz internal clock domains of the USBFS full-speed block are closed by clock gating. Even if the application still provides the clock input, it will save the dynamic power consumption of the module due to the inversion of the clock signal. It will also shut down most of the transceiver units. Only the part responsible for detecting asynchronous recovery events or remote wake-up events remains active. .

- HCLK gate (GATEHCLK bit in USBFS_GCCTL)

When the GATEHCLK bit in the clock gating control register is set to 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. Even if the application still provides the clock input, the dynamic power consumption of the module due to the inversion of the clock signal will be saved.

To save dynamic power, the USB data FIFO is only clocked when it is being accessed by the USBFS module.

35.5.7 USBFS dynamically updates the USBFS_HFIR register

In the host mode, the USB module has the function of dynamically fine-tuning the frame period, and can synchronize the external device with the SOF frame. If the USBFS_HFIR register is changed in the current SOF frame, the SOF cycle will be corrected accordingly in the next frame. For details, please refer to Figure 35-4 .

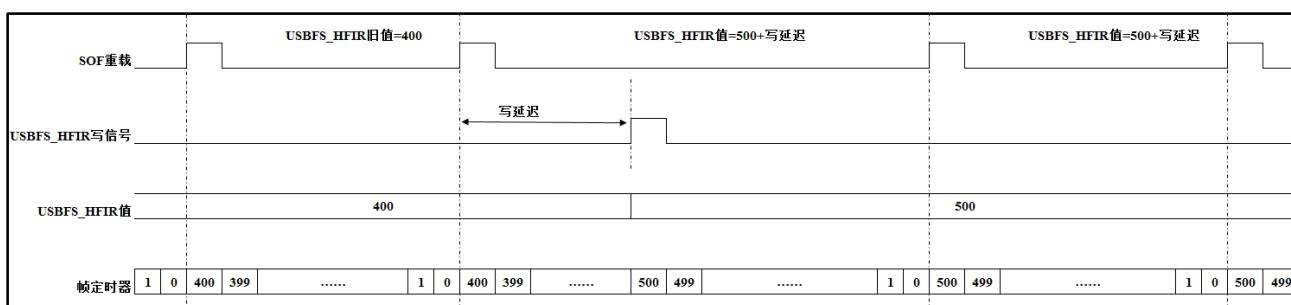


Figure 35-4 Schematic diagram of USBFS dynamically updating USBFS_HFIR register

35.5.8 USBFS data FIFO

The USBFS system has 2.5KB 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 (before USB transfers, into which the application pushes data for short storage) and a single RxFIFO (before data received from USB is read by the application) , where it is temporarily stored).

The number and organization of FIFOs constructed 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 FIFO is configured by software to better meet the application requirements.

35.5.9 USBFS Host FIFO Architecture

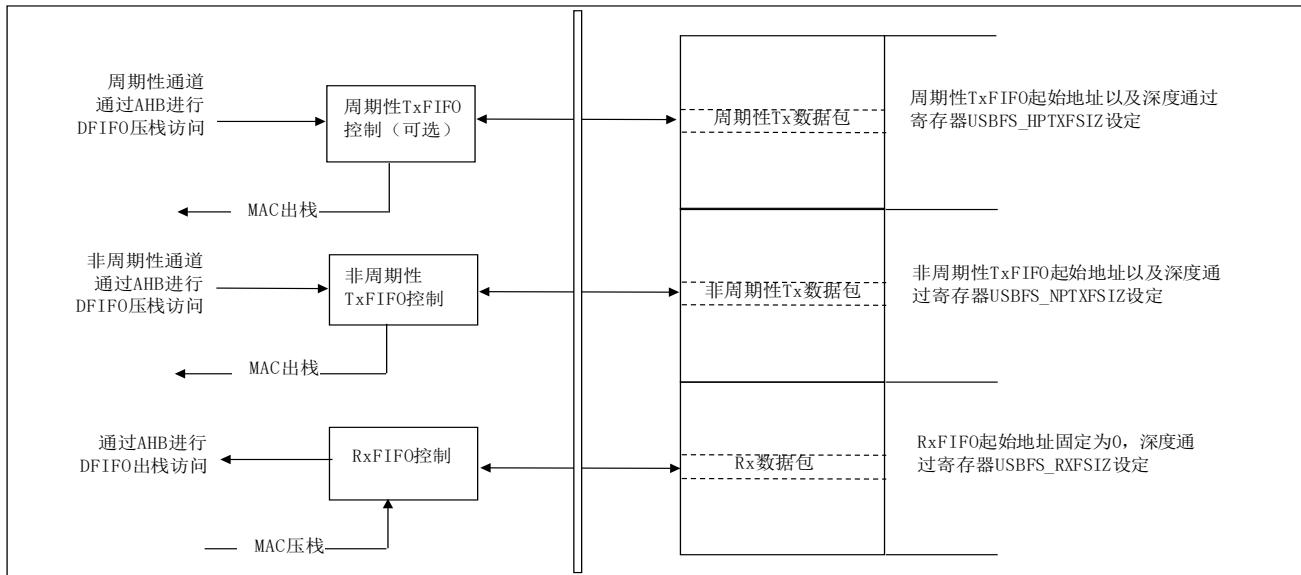


Figure 35-5 Schematic diagram of FIFO architecture in USBFS host mode

35.5.9.1 Host RxFIFO

The host uses one receive FIFO to handle all periodic and aperiodic 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 the system memory. As long as there is room in the FIFO, packets from the IN endpoint of the device are received and stored one by one. The status of each packet received (including host destination channel, number of bytes, data PID, and checksum of 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 enables 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 host software, the USBFS module can fill the receive FIFO to the limit

The application will receive the Rx FIFO not empty interrupt 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 pop registers, and finally reads the data from the RxFIFO.

35.5.9.2 Host TxFIFO

The host uses one transmit FIFO for all aperiodic (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 the USB (transmit packets). The size of the Periodic (Aperiodic) TxFIFO is configured in the Host Periodic (Aperiodic) Transmit FIFO Size (HPTXFSIZ/HNPTXFSIZ) register.

The two Tx FIFOs operate according to the priority, and the priority of the periodic communication is higher, so the periodic communication is performed first within the time of one USB frame. At the beginning of a frame, the built-in host scheduler processes the periodic request queue first, and then processes the aperiodic request queue.

The architecture of the two transmit FIFOs enables the USB host to optimally manage the periodic and aperiodic transmit data buffers separately:

- All host channels configured to support periodic (aperiodic) OUT transactions share the same RAM buffer (shared FIFO)
- For any sequence of OUT tokens driven by host software, the USBFS module can fill the periodic (aperiodic) transmit FIFO to the limit

The USBFS module issues a periodic TxFIFO empty interrupt (PTXFE bit in USBFS_GINTSTS) whenever the periodic TxFIFO is half empty or completely empty, depending on the periodic Tx-FIFO empty level bit in the AHB configuration register (PTXFELVL bit in USBFS_GAHBCFG) value. As long as there is free space in both the periodic TxFIFO and the periodic request queue, the application can write transmit data ahead of time. The available space for both can be known by reading the Host Periodic Transmit FIFO and Queue Status Register (HPTXSTS).

The USBFS module issues an aperiodic TxFIFO empty interrupt (NPTXFE bit in USBFS_GINTSTS) whenever the aperiodic TxFIFO is half empty or completely empty, depending on the aperiodic TxFIFO empty level bit in the AHB configuration register (TXFELVL in USBFS_GAHBCFG bits). The application can write transmit data as long as there is free space in both the aperiodic TxFIFO and the aperiodic request queue. The available space for both can be known by reading the Host Aperiodic Transmit FIFO and Queue Status Register (HNPTXSTS).

35.5.10 USBFS Device FIFO Architecture

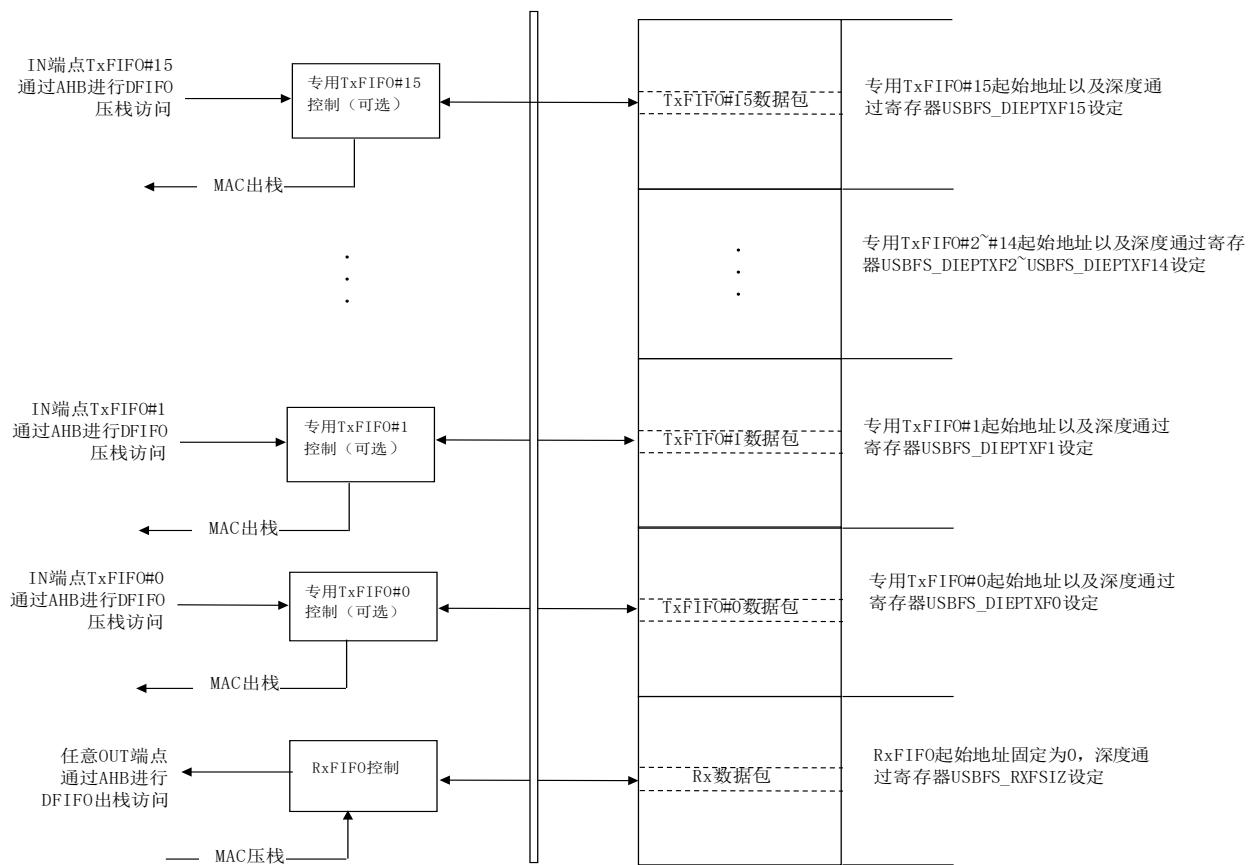


Figure 35-6 Schematic diagram of FIFO architecture in USBFS device mode

35.5.10.1 Device RxFIFO

USBFS devices use a single receive FIFO to receive data sent to all OUT endpoints. As long as there is free space in the Rx FIFO, the received packets are filled into the Rx FIFO one by one. In addition to valid data, the received packet status (including OUT endpoint destination number, byte count, data PID, and validation of received data) is also stored by the module. When no space is available, the device replies with a host transaction NAK acknowledgement and triggers an interrupt on the addressed endpoint. The size of the receive FIFO is configured in the receive FIFO size register (GRXFSIZ).

A single receive FIFO architecture allows USB devices to fill receive RAM buffers 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 the Rx FIFO not empty interrupt (RXFNE bit in USBFS_GINTSTS) as long as at least one packet is available for reading in the Rx FIFO. The application program reads the packet information from the receive status read and pop register (GRXSTSP), and finally reads the corresponding data from the receive FIFO by reading the pop address associated with the endpoint.

35.5.10.2 Device TxFIFO

The module provides dedicated FIFOs for each IN endpoint. The application configures the FIFO size for IN endpoint 0 through the aperiodic sending FIFO size register (USBFS_DIEPTSIZ0); configures the FIFO size for IN endpoint x through the device IN endpoint sending FIFOx register (DIEPTSIZx).

35.5.11 USBFS FIFO RAM allocation

35.5.11.1 Host mode

Receive FIFO RAM allocation

Status information is written to the FIFO with each received packet. Therefore, at least (maximum packet size / 4) + 1 space must be allocated for receive packets. If multiple isochronous channels are enabled, the space allocated for receiving consecutive packets must be at least twice (maximum packet size / 4) + 1. In general, the recommended space is twice (maximum packets / 4 + 1), so that while the previous packet is sent to the CPU, the USB can simultaneously receive subsequent packets.

The transfer completion status information is written to the FIFO along with the last packet received by the endpoint. So a place must be allocated for this.

When running in DMA mode, each host channel's DMA address register is stored in the FIFO, so a place in the FIFO needs to be reserved for each channel to store its address register.

Transmit FIFO RAM allocation

The minimum RAM required for the host aperiodic transmit FIFO is the size of the largest packet transmitted on all supported aperiodic 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 transmit FIFO with the next packet.

The minimum RAM required by the host periodic transmit FIFO is the size of the largest packet transmitted on all supported periodic OUT channels. If there is at least one isochronous OUT endpoint, the space must be at least twice the maximum packet size in that channel.

When running in DMA mode, each host channel's DMA address register will be stored in the FIFO, so it is necessary to reserve a location in the FIFO for each channel to store its address register.

35.5.11.2 Device mode

Receive FIFO RAM allocation

Applications should allocate RAM for SETUP packets: Eleven 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 allocated for the global OUT NAK. Status information is written to the FIFO with each received packet. Therefore, at

least (maximum packet size / 4) + 1 space must be allocated for receive packets. If multiple isochronous endpoints are enabled, the space allocated for receiving consecutive packets must be at least twice (maximum packet size / 4) + 1. In general, the recommended space is twice (maximum packets / 4 + 1), so that while the previous packet is sent to the CPU, the USB can simultaneously receive subsequent packets.

The transfer completion status information is pushed into the FIFO along with the last packet received by the endpoint. Typically, it is recommended to assign a location to each OUT endpoint.

Transmit FIFO RAM allocation

The minimum RAM space required for each IN endpoint transmit FIFO is the maximum packet size for that particular IN endpoint.

35.5.12 USBFS system performance

Optimum USB and system performance is achieved with large RAM buffers, highly configurable FIFO size, fast 32-bit FIFO access via AHB push/pop registers, and especially advanced FIFO control mechanisms. In fact, USBFS efficiently fills the available RAM space through this mechanism, regardless of the current USB sequence. Take advantage of these features:

- The application has enough headroom to calculate and correct the CPU load to optimize CPU bandwidth utilization:
 - The application can accumulate a large amount of sending data first, and then send it out through the USB
 - Provides sufficient time margin to read data from the receive FIFO
- The USB module can maintain full-speed operation, that is, provide the maximum full-speed bandwidth (as much hardware as possible runs automatically, and as little software as possible)
 - The USB module can accumulate a large amount of transmission data in advance for its disposal, so that it can self-manage USB data transmission
 - There is a lot of empty space in the receive buffer, which is automatically filled with data from the USB

Since the USBFS module can efficiently fill the 2.5KB RAM buffer and the 2.5KB transmit/receive data is enough to accommodate a full-speed frame, the USB system can reach the maximum USB bandwidth without application intervention within one frame.

35.5.13 USBFS interrupts and events

There are two types of USBFS interrupts: SOTP mode wake-up interrupt USBFS_WKUP and USBFS global interrupt USBFS_GLB.

USBFS_WKUP interrupt

The USBFS_WKUP interrupt is used to wake up the system in STOP mode through USBFS_DP or USBFS_DM. The interrupt enable bit is INT_WUPEN.USF_WUEN.

Before using USBFS to wake up the system from STOP, it is necessary to ensure that the USBFS controller is in a suspended state, and set the corresponding filter range and enable filter function in the register USBFS_SYCTLREG.

USBFS_GLB interrupt

The USBFS_GLB interrupt is the main interrupt that software needs to handle. The flag bit of the global interrupt can be read in the USBFS_GINTSTS register.

Table 35-2 USBFS_GLB interrupt event table

Interrupt logo	Description	Operating mode	internal event source
WKUPINT	Resume/remote wakeup interrupt	Host or device	-
VBUSVINT	VBUS active interrupt	Equipment	-
DISCINT	Disconnection interrupted	Host	-
CIDSCHG	Connector ID line state change interrupt	Host or device	-
PTXFE	Periodic txfifo Empty Interrupt	Host	-
LPMINT	LPM interrupt	Host or device	
HCINT	Host channel interrupt	Host	-
HPRTINT	Host port interrupt	Host	-
DATAFSUSP	Data fetch pending	Equipment	-
IPXFR/INCOMPISOOUT	Incomplete periodic transmission/Incomplete OUT synchronization transmission	Equipment	-
IISOIXFR	In sync transfer not completed	Equipment	-
OEPINT	OUT endpoint interrupt	Equipment	-
IEPINT	IN endpoint interrupt	Equipment	-
EOPF	Periodic end of frame interrupt	Equipment	-
ISOODRP	Drop sync OUT packet interrupt	Equipment	-
ENUMDNE	Enumeration complete	Equipment	-
USBRST	USB reset interrupt	Equipment	-
USBSUSP	USB suspend interrupt	Equipment	-
ESUSP	Early pending interrupt	Equipment	-
GONAKEFF	Global OUT NAK valid interrupt	Equipment	-
GINAKEFF	Global aperiodic IN NAK valid interrupt	Equipment	-

Interrupt logo	Description	Operating mode	internal event source
NPTXFE	Aperiodic txfifo Empty Interrupt	Host	-
RXFNE	Rxfifo not empty interrupt	Host or device	-
SOF	Start of frame interrupt	Host or device	Yes
MMIS	Pattern mismatch interrupt	Host or device	-

35.6 USBFS programming model

35.6.1 USBFS module initialization

The application must perform the module initialization sequence.

Please refer to the mode determination method 35.5.2 USBFS mode decision .

This section describes the initialization process after the USBFS controller is powered on. Whether working in host or device mode, an application must follow an initialization sequence. All module global registers are initialized according to the module configuration:

1. Program the following fields in the USBFS_GAHBCFG register:
 - Global interrupt mask bit GINTMSK = 1
 - RxFIFO not empty (RXFNE bit in USBFS_GINTSTS)
 - Periodic TxFIFO Empty Threshold
2. Program the following fields in the USBFS_GUSBCFG register:
 - FS Timeout Calibration Field
 - USB turnaround time field
3. Software must unmask the following bits in the USBFS_GINTMSK register:
 - pattern mismatch interrupt mask
4. By reading the CMOD bit in USBFS_GINTSTS, software can determine whether the USBFS controller is operating in host mode or device mode.

35.6.2 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 the pair.
2. Program the USBFS_HCFG register to select the full-speed host.
3. Program the PWPR bit in USBFS_HPRT to 1 to provide VBUS to the USB bus.
4. Wait for 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 PENCHNG interrupt in USBFS_HPRT.

9. Read the PSPD bit in USBFS_HPRT for enumeration speed.
10. Using the selected PHY clock, set the HFIR register accordingly.
11. Program the FSLSPCS field in the USBFS_HCFG register according to 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 start address of the aperiodic transmit FIFO used for aperiodic communication transactions.
14. Program the USBFS_HPTXFSIZ register to select the size and start address of the periodic communication transmit FIFO for periodic transactions.

To communicate with a device, system software must initialize and enable at least one channel.

35.6.3 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:
 - device speed
 - Non-zero length state OUT handshake signal
2. Program the USBFS_GINTMSK register to unmask the following interrupts:
 - USB reset
 - Enumeration complete
 - early hang
 - USB hang
 - SOF
3. Wait for the VBUSVINT interrupt in USBFS_GINTSTS to enter 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 since this interrupt was 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 speed and perform the steps listed in Endpoint initialization when enumeration completes.

At this point, the device is ready to accept SOF packets and perform control transfers on control endpoint 0.

35.6.4 USBFS DMA mode

USB uses the AHB master interface to get transmit packet data (AHB to USB) and receive data updates (USB to AHB). The AHB master interface uses programmed DMA addresses (HCDMAX registers in host mode and DIEPDMAx/DOEPDMAx registers in device mode) to access the data buffers.

35.6.5 USBFS Host Programming Model

35.6.5.1 Channel initialization

An application must initialize one or more channels before it can communicate with connected devices.

To initialize and enable a channel, the application must perform the following steps:

1. Program the USBFS_GINTMSK register to unmask the following bits:
 - The aperiodic transmit FIFO for OUT transactions is empty (applies when working at the pipelined transaction level and the packet count field is programmed to a value greater than 1).
 - The aperiodic transmit FIFO for OUT transactions is half-empty (applies when working at the pipelined transaction level and the packet count field is programmed to a value greater than 1).
2. Program the USBFS_HAINTMSK register to enable interrupts for the selected channel.
3. Program the USBFS_HCINTMSK register to enable interrupts related to communication transactions reflected in the Host Channel Interrupt Register.
4. Program the USBFS_HCTSIZx registers of the selected channel, specifying the total transfer size in bytes and the expected number of packets including short packets. The application must program the PID field with the initial data PID (for the first OUT transaction or expected to be obtained from the first IN transaction).
5. Program the selected channel's USBFS_HCCHARx registers to specify the device's endpoint characteristics such as type, speed, orientation, etc. (The channel can be enabled by setting the channel enable bit to 1 only when the application is ready to send or receive packets).

35.6.5.2 Channel stop

The application can disable any channel by programming the USBFS_HCCHARx registers to set the CHDIS and CHENA bits. This causes the USBFS host to flush previous requests on that channel (if any) and generate a channel stop interrupt. The application must wait for the CHH interrupt in USBFS_HCINTx before reassigning the channel to other communication transactions. The USBFS host does not interrupt communication transactions already initiated on the USB.

Before disabling a channel, the application must ensure that there is at least one free space in the aperiodic request queue (when aperiodic channels are disabled) or the periodic request queue (when periodic channels are disabled). The application can clear the request queue when the request queue is full (before disabling the channel) by programming the USBFS_HCCHARx registers to set the CHDIS bit and clear the CHENA bit. The application blocks the channel when any of the following occurs:

1. STALL, TXERR, BBERR or DTERR interrupt received in USBFS_HCINTx of IN or OUT channel.
The application must be able to receive other interrupts (DTERR, Nak, Data, TXERR) for the same channel before receiving the channel stop signal.
2. Received DISCINT (disconnect device) interrupt in USBFS_GINTSTS. (The application will disable all enabled channels).
3. The application aborted the transfer before it completed normally.

In DMA mode, the application cannot stop the indivisible periodic transfer by overwriting the register.

35.6.6 USBFS Device Programming Model

35.6.6.1 Endpoint endpoint initialization on USB reset

1. Set NAK bit to 1 for all OUT endpoints
 - In USBFS_DOEPCRTLx, SNAK = 1 (for all OUT endpoints)
2. Unmask the following interrupt bits
 - In USBFS_DAINTMSK, INEP0=1 (control 0 IN endpoint)
 - In USBFS_DAINTMSK, OUTEP0=1 (control 0 OUT endpoint)
 - In DOEPMASK, STUP=1
 - In DOEPMASK, XFRC=1
 - In DIEPMASK, XFRC=1
 - In DIEPMASK, 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 transfers. This register must be at least equal to 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 at least equal to control endpoint 0's 1 maximum packet size.
4. Program the following fields in the endpoint related registers to control OUT endpoint 0 to receive the SETUP packet
 - STUPCNT=3 in USBFS_DOEPTSIZ0 (receives up to 3 consecutive SETUP packets)

At this point, all initialization required to receive the SETUP packet is complete.

35.6.6.2 Endpoint endpoint initialization on USB reset

1. In the enumeration complete 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 transfers at control endpoint 0.

35.6.6.3 Endpoint initialization when SetAddress command is received

This section describes what an application must do when it receives a SetAddress command in a SETUP packet.

1. Program the USBFS_DCFG register with the device address received in the SetAddress command
2. Program the module to emit IN packets for the status phase

35.6.6.4 Endpoint initialization when SetConfiguration/SetInterface command is received

This section describes what an application must do when it receives a SetConfiguration or SetInterface command in a SETUP package.

1. When the SetConfiguration command is received, the application must program the endpoint registers to configure these endpoint registers with the characteristics of the valid endpoint in the new configuration.
2. When a SetInterface command is received, the application must program the endpoint registers 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 disabled.
4. Use the USBFS_DAINTMSK register to enable interrupts for valid endpoints and mask interrupts for invalid endpoints.
5. Set up data FIFO RAM for each FIFO.
6. After configuring all required endpoints, the application must program the module to send IN packets for the status phase.

At this point, the device module can already receive and send any type of data packet

35.6.6.5 Endpoint activation

This section describes the steps required to activate a device endpoint or 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 endpoint) or the USBFS_DOEPCTLx register (for OUT or bidirectional endpoint).
 - maximum packet size
 - USB Active Endpoint Position 1
 - Endpoint Initial Data Sync bit (for interrupt and bulk endpoints)
 - Endpoint type
 - TxFIFO number
2. Once an endpoint is activated, the module starts decoding the token sent to that endpoint and replies with a valid handshake if the received token is valid.

35.6.6.6 Endpoint deactivation

This section describes the steps required to decommission an existing endpoint.

1. Clear the USB Active Endpoint bit in the USBFS_DIEPCTLx register (for IN or bidirectional endpoint) or the USBFS_DOEPCTLx register (for OUT or bidirectional endpoint) in the endpoint to be disabled.
2. When an endpoint is deactivated, the module ignores tokens sent to that endpoint, causing the USB to time out.

35.6.7 USBFS operating model

35.6.7.1 SETUP and OUT data transfer

This section describes the internal data flow and application operation steps during data OUT transfers and SETUP transactions.

Packet read

This section describes how to read packets (OUT data and SETUP packets) from the receive FIFO.

1. When the RXFNE interrupt is captured (USBFS_GINTSTS register), the application must read the Receive Status Popup Register (USBFS_GRXSTSP).
2. The application can mask the RXFNE interrupt (in USBFS_GINTSTS) by writing RXFNE=0 (in USBFS_GINTMSK) until it reads the packet from the receive FIFO.
3. If the byte count of the received packet is not 0, the data is popped from the receive data FIFO and stored in memory. If the received packet byte count is 0, no data will be popped from the receive data FIFO.
4. The packet status read from the receive FIFO has the following status:

- Global OUT NAK:

PKTSTS=Global OUT NAK, BCNT=0x000, the values of EPNUM and DPID do not matter.

These data indicate that the global OUT NAK bits are 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 can now be read from the receive FIFO.

- The build phase is complete:

PKTSTS=Setup phase complete, BCNT=0x0, EPNUM=Control EP number, DPID value does not matter.

These data indicate that the setup phase for the specified endpoint is complete and the data phase has started. After this status entry is popped from the receive FIFO, the module will generate a setup interrupt on this control OUT endpoint.

- OUT packets:

PKTSTS=DataOUT, BCNT=the size of the received OUT data packet (BCNT:0~1024), EPNUM=the endpoint number of the received data packet, DPID=the actual data PID.

- Data transfer complete:

PKTSTS=OUT data transfer completed, BCNT=0x0, EPNUM=OUT EP number that completes data transfer, DPID value does not matter.

These data indicate that the OUT data transfer of the specified OUT endpoint is complete. After this status entry is popped from the receive FIFO, the module will raise a "transfer complete" interrupt on the specified OUT endpoint.

5. After popping data from the receive FIFO, the RXFNE interrupt must be unmasked (USBFS_GINTSTS).
6. Steps 1 to 5 are 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 how the module handles SETUP packets and the order in which the application handles SETUP transactions.

Application Requirements:

1. To receive a SETUP packet, the Control OUT endpoint STUPCNT field (USBFS_DOEPTSIZx) must be programmed to 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_DOEPCTLx. The STUPCNT field is decremented each time the control endpoint receives a SETUP packet. If the STUPCNT field is not programmed to the appropriate value before receiving the SETUP packet, the module can still receive the SETUP packet and decrement the STUPCNT field,

but the application may not be able to determine the correct number of SETUP packets received during the setup phase of the control transfer .

- In USBFS_DOEPTSIZx, 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.
- Reserve space for 10 characters. The first SETUP packet requires 3 words, the "setup phase complete" status double word requires 1 word, and 6 words are required 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 to store SETUP packets and will never use this space for data packets.
- 3. The application must read 2 words of the SETUP packet from the receive FIFO.
- 4. The application must read and discard the "Setup Phase Complete" status word from the receive FIFO

Internal data flow:

- 1. When a SETUP packet is received, the module will write the received data into the receive FIFO without checking the available space in the receive FIFO and regardless of the endpoint's NAK and STALL bit settings.
 - The module will internally set the IN NAK and OUTNAK bits of the control IN/OUT endpoint that received the SETUP packet.
- 2. For each SETUP packet received on the USB, the module writes 3 words of data into the receive FIFO and decrements the STUPCNT field by 1.
 - The first word contains 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 setup phase ends and the data IN/OUT phase begins, the module writes a status entry ("Setup Phase Complete" word) to the receive FIFO to indicate that the setup phase is complete.
- 4. On the AHB side, the SETUP packet is read by the application.
- 5. When the application pops the "Setup Phase Complete" word from the receive FIFO, the module will use the STUP interrupt (USBFS_DOEPINTx) to interrupt the application, indicating that it can process the received SETUP packet.
 - The module will clear the endpoint enable bit that controls the OUT endpoint.

Application programming sequence:

- 1. Program the USBFS_DOEPTSIZx registers.
 - STUPCNT=3
- 2. Wait for the RXFNE interrupt (USBFS_GINTSTS) and read the packet from the receive FIFO.

3. Triggering of STUP interrupt (USBFS_DOEPINTx) indicates the successful completion of SETUP data transfer.
 - When this interrupt occurs, the application must read the USBFS_DOEPTSIZx registers to determine the number of SETUP packets received and process the last SETUP packet received.

Process more than three consecutive SETUP packets:

According to the USB2.0 specification, in a SETUP packet error, the host usually does not send more than 3 consecutive SETUP packets to the same endpoint. However, the USB 2.0 specification does not limit the number of consecutive SETUP packets 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 with a NAK to asynchronous OUT tokens sent by the host, and ignores synchronous OUT packets.
2. The module writes the global OUT NAK to the receive FIFO. The application must allow enough space for this.
3. The module sets the GONAKEFF interrupt (USBFS_GINTSTS) when the application pops the global OUT NAK word from the receive FIFO.
4. When the application detects this interrupt, it considers the module to be in global OUT NAK mode. The application can clear this interrupt by clearing the SGONAK bit in USBFS_DCTL.

Application programming sequence:

1. To stop receiving any type of data into the receive FIFO, the application must set the global OUT NAK bit by programming the following fields.
 - In USBFS_DCTL, SGONAK =1
2. Wait for 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 bit in USBFS_DCTL, the application can receive valid OUT packets before the module raises the GONAKEFF interrupt (USBFS_GINTSTS).
3. The application can temporarily mask this interrupt by writing to the GINAKEFFM bit in the USBFS_GINTMSK register.
 - In the USBFS_GINTMSK register, GINAKEFFM=0
4. When the application is ready to exit global OUT NAK mode, the SGONAK bit in USBFS_DCTL must be cleared. This action also clears the GONAKEFF interrupt

(USBFS_GINTSTS).

- In CGONAK, USBFS_DCTL=1
- 5. If the application has previously masked this interrupt, it must unmask the interrupt as follows:
 - In GINTMSK, GINAKEFFM=1

Set global OUT NAK to 1

The application must disable enabled OUT endpoints using the following sequence.

Application programming sequence:

1. The application must enable global OUT NAK mode in the module before disabling any OUT endpoints.
 - In USBFS_DCTL, SGONAK=1
2. Waiting for GONAKEY 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 EPDSD interrupt (USBFS_DOEPINTx), which indicates that the OUT endpoint is completely disabled. When an EPDSD interrupt is raised, the module also clears the following bits:
 - 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 non-disabled OUT endpoints.
 - 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 data to be received as part of an OUT transfer.
2. For OUT transfers, the transfer size field in the endpoint's transfer size register must be a multiple (and word-aligned) of the endpoint's maximum packet size.
 - 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 endpoint's transfer size register to calculate the amount of data available in memory. The amount of valid data received may be less than the programmed transfer size.

- Effective data volume in memory = initial transfer volume set by application - remaining transfer volume after module update
- Number of received USB packets = initial number of packets set by the application - number of remaining packets after module update

Internal data flow:

1. The application must set the transfer size and packet count fields in the endpoint-related registers, clear the NAK bit, and enable the endpoint to receive data.
2. After the NAK bit is cleared, the module begins to receive data and write the data to the receive FIFO (as long as there is room in the receive FIFO). For every packet received on the USB, the packet and its status are written to the receive FIFO. Each packet written to the receive FIFO (either a packet with a maximum packet size or a short packet) decrements the packet count field for that endpoint by 1.
 - If the CRC of the received data packet is invalid, it is automatically cleared from the receive FIFO.
 - After an ACK is returned for the packet on the USB, the module will discard the asynchronous OUT packet that the host retransmitted because the ACK could not be detected. The application will not detect multiple consecutive OUT packets on the same endpoint with the same data PID. In this case, the packet count will not be decremented.
 - If there is no room in the receive FIFO, synchronous or asynchronous packets are ignored and not written to the receive FIFO. Additionally, asynchronous OUT tokens will receive a NAK handshake response.
 - In all three cases above, the packet count will not be decremented because no data is written to the receive FIFO.
3. When the packet count becomes 0 or a short packet is received on the endpoint, the NAK bit for that endpoint will be set to 1. When the NAK bit is set, synchronous or asynchronous packets are ignored and not written to the receive FIFO, and the non-synchronous OUT token receives a NAK handshake reply.
4. After the data is written to the receive FIFO, the application will read the data from the receive FIFO and write the data to the external memory, one packet at a time, endpoint by endpoint.
5. After each packet is written to external memory on the AHB, the transfer size of the endpoint is automatically subtracted by that packet size.
6. The OUT data transfer completion status of the OUT endpoint is written to the receive FIFO when:
 - The transfer size is 0 and the 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 completion interrupt for this endpoint, and clears the endpoint enable bit.

Application programming sequence:

1. Program the USBFS_DOEPTSI x registers with the transfer size and the corresponding number of packets.
2. Program the USBFS_DOEPCTL x registers using the endpoint characteristics and set the EPENA and CNAK bits to 1.
 - In USBFS_DOEPCTL x , EPENA=1
 - In USBFS_DOEPCTL x , CNAK =1
3. Wait for the RXFNE interrupt (in USBFS_GINTSTS) and read the packet from the receive FIFO.
 - This step can be repeated multiple times, depending on the transfer size.
4. Trigger the XFRC interrupt (USBFS_DOEPINT x) to indicate the successful completion of the asynchronous OUT data transfer.
5. Read the USBFS_DOEPTSI x registers to determine the amount of valid data.

Universal synchronous OUT data transfer

This section describes conventional synchronous OUT data transfers.

Application Requirements:

1. All application requirements for asynchronous OUT data transfers apply to same OUT data transfers.
2. For the transfer size and packet count fields in isochronous OUT data transfers, they must always be set to the number of packets of the maximum packet size that can be received in a single frame. A synchronous type of OUT data transfer transaction must be completed within one frame.
3. Before the end of the periodic frame (EOPF interrupt in USBFS_GINTSTS), the application must read all isochronous OUT packets (data entry and status entry) from the receive FIFO.
4. To receive data in the next frame, a synchronous OUT endpoint must be enabled after EOPF (USBFS_GINTSTS) and before SOF (USBFS_GINTSTS).

Internal data flow:

1. The internal data flow of a synchronous OUT endpoint is basically the same as that of an asynchronous OUT endpoint, with a few differences.
2. When a synchronous OUT endpoint is enabled by setting the endpoint enable bit and clearing the NAK bit, the even/odd frame bits 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_DOEPCTL x) = FNSOF[0] (in USBFS_DSTS)

3. When the application reads a complete isochronous OUT packet (data and status) from the receive FIFO, the module updates the RXDPID field in USBFS_DOEPTSIZx according to the data PID of the last isochronous OUT packet read from the receive FIFO.

Application programming sequence:

1. Program the USBFS_DOEPTSIZx registers with the transfer size and corresponding packet count
2. Program the USBFS_DOEPCTLx registers using the endpoint characteristics and set the endpoint enable bit, clear NAK bit, and odd/even frame bits.
 - EPENA1
 - 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 multiple times, depending on the transfer size.
4. The XFRC interrupt (in USBFS_DOEPINTx) indicates that the synchronous OUT data transfer is complete. The interrupt does not necessarily mean that the data in memory is valid.
5. For isochronous OUT transfers, the interrupt may not always be detected by the application. Instead, the application may detect the IISOXFMR interrupt in USBFS_GINTSTS.
6. Read the USBFS_DOEPTSIZx registers to determine the received transfer size and to determine the validity of the data received in the frame. An application must treat data received in memory as valid data 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 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 data packets that receive the valid data = the number of initial data packets programmed by the application program - the number of remaining data packets after the module is updated.

Applications can drop invalid packets.

Incomplete sync OUT data transfer

This section describes the application programming sequence in the event of loss of synchronous OUT packets.

Internal data flow:

1. For synchronous OUT endpoints, the XFRC interrupt (in USBFS_DOEPINTx) may not always be raised. If the module drops isochronous OUT packets, the application may fail to detect the XFRC interrupt (USBFS_DOEPINTx) under the following conditions:
 - When the receive FIFO cannot hold a complete ISO OUT packet, the module will discard the received ISO OU data
 - Received sync OUT packet with CRC error
 - The SYNC OUT token received by the module is corrupted
 - The application is very slow to read data from the receive FIFO
2. If the module detects a periodic end of frame before the transfer of all isochronous OU endpoints is complete, the outstanding isochronous OUT data interrupt (IISOOXFRM in USBFS_GINTSTS) is triggered, indicating that the XFRC interrupt (in USBFS_DOEPINTx) is not triggered on at least one isochronous OUT endpoint. At this point, the endpoint that has not completed the transfer remains enabled, but there is no active transfer in progress on that endpoint of the USB.

Application programming sequence:

1. A hardware-triggered IISOOXFRM interrupt (USBFS_GINTSTS) indicates that at least one isochronous OUT endpoint in the current frame has an outstanding transfer.
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 read from the receive FIFO before proceeding.
 - Once all data has been read from the receive FIFO, the application can detect the XFRC interrupt (USBFS_DOEPINTx). In this case, the application must re-enable the endpoint to receive synchronous OUT data in the next frame.
3. When the application receives the IISOOXFRM interrupt (in USBFS_GINTSTS), the application must read the control registers (USBFS_DOEPCTLx) of all isochronous OUT endpoints to determine which endpoints have incomplete transfers in the current frame. When both of the following conditions are met at the same time, it indicates that the endpoint transfer is not complete:
 - EONUM bit (in USBFS_DOEPCTLx) = FNSOF[0] (in USBFS_DSTS)
 - EPENA=1 (in USBFS_DOEPCTLx)
4. Before the SOF interrupt is detected (in USBFS_GINTSTS), the previous step must be done to ensure that the current frame number has not changed.
5. For isochronous OUT endpoints with incomplete transfers, the application must discard the

- data in memory and disable the endpoint by setting the EPDIS bit 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 the module may take some time to disable the endpoint, the application may not receive data in the next frame after receiving invalid sync data.

Stop asynchronous OUT endpoints

This section describes how an application can stop an asynchronous endpoint.

1. Put the module in global OUT NAK mode.
2. Ban the required endpoints
 - When disabling an endpoint, set STALL=1 (in USBFS_DOEPCTL) instead of setting the SNAK bit in USBFS_DOEPCTL. The STALL bit always takes precedence over the NAK bit.
3. When the application no longer needs the endpoint to reply to the STALL handshake, the STALL bit (in USBFS_DOEPCTLx) must be cleared.
4. If the application sets or clears the STALL state of an endpoint as a result of receiving a SetFeature.Endpoint Halt or ClearFeature.Endpoint Halt command from the host, it must set or clear the STALL bit before the state phase transfer on that control endpoint.

35.6.7.2 IN data transfer

Packet write

This section describes how an application can write packets to the endpoint FIFO when the dedicated transmit FIFO is enabled.

1. The application can choose polling mode or interrupt mode.
 - In polling mode, the application monitors the status of the endpoint transmit data FIFO by reading the USBFS_DTXFSTSx registers to determine if there is enough space in the data FIFO.
 - In interrupt mode, the application waits for the TXFE interrupt (in USBFS_DIEPINTx) and then reads the USBFS_DTXFSTSx registers 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 hold the entire packet.
 - To write a zero-length packet, the application cannot look into the FIFO space.
 2. Using one of the above methods, when the application determines that there is enough space to write the transmit packet, the application must first write the endpoint control register accordingly, and then write the data to the data FIFO. Typically, the application must perform read-modify-write operations on the USBFS_DIEPCTLx registers to avoid modifying other contents of the register while the endpoint enable bit is set.
- An application can write multiple packets of 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. All packets to be sent in the next frame will not be written by the application until the transmission of the communication transaction of the previous frame is completed.

Set IN endpoint NAK to 1

Internal data flow:

1. When the application sets the IN NAK of a specific endpoint to 1, the module will stop data transmission on the endpoint regardless of whether the data in the endpoint transmit FIFO is available.
2. The asynchronous endpoint receives the IN token and replies with a NAK handshake response.
 - Sync endpoint receives IN token, returns zero-length packet
3. The module triggers the INEPNE (IN endpoint NAK valid) interrupt in USBFS_DIEPINTx in response to the SNAK bit in USBFS_DIEPCTLx.
4. When the application detects this interrupt, it considers the endpoint to be in IN NAK mode. The application can clear this interrupt by setting the CNAK bit in USBFS_DIEPCTLx.

Application programming sequence:

1. To stop sending any data on a specific IN endpoint, the application must set the IN NAK bit. To set this bit, the following fields must be programmed.
 - SNAK=1 in USBFS_DIEPCTLx
2. Wait for the INEPNE interrupt in USBFS_DIEPINTx to fire. This interrupt indicates that the module has stopped sending data on the endpoint.
3. When the application sets the NAK bit but the "NAK Valid" interrupt has not yet triggered, the module can send valid IN data on the endpoint.
4. The application can temporarily mask this interrupt by writing to the INEPNEM bit in DIEPMSK.
 - In DIEPMSK, INEPNEM = 0
5. To exit endpoint NAK mode, the application must clear the NAK status bit (NAKSTS) in USBFS_DIEPCTLx. 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

Disable IN endpoint

Use the following sequence to disable specific IN endpoints that were previously enabled.

Application programming sequence:

1. The application must stop writing data on the AHB before disabling the IN endpoint.
2. The application must set the endpoint to NAK mode.
 - SNAK=1 in USBFS_DIEPCTLx
3. Wait for INEPNE interrupt in USBFS_DIEPINTx.
4. Set the following bits in the USBFS_DIEPCTLx registers for endpoints that must be disabled.
 - EPDIS=1 in USBFS_DIEPCTLx
 - SNAK=1 in USBFS_DIEPCTLx
5. Triggering of the EPDISD interrupt in USBFS_DIEPINTx indicates that the module has completely disabled the specified endpoint. When the interrupt is triggered, the module also clears the following bits:
 - In USBFS_DIEPCTLx, EPENA=0
 - In USBFS_DIEPCTLx, EPDIS=0
6. The application must read the USBFS_DIEPTSIZx registers for periodic IN EPs to calculate how much data is being sent over the USB on the endpoint.
7. The application must clear the endpoint transmit FIFO by setting the following fields in the USBFS_GRSTCTL register:
 - 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, which indicates the end of the FIFO flush operation. To send new data on this endpoint, the application can re-enable the endpoint at a later time.

Universal aperiodic IN data transmission

Application Requirements:

1. Before establishing an IN transfer, the application must ensure that each packet that makes up an IN transfer 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 maximum packet size packets plus a short packet at the end of the transfer:

$$\text{transfer size[EPNUM]} = x \times \text{MPSIZ[EPNUM]} + \text{sp}$$

If (sp > 0), packet count [EPNUM] = x + 1.

Otherwise, Packet Count [EPNUM] = x

- To send a single zero-length packet:

transfer size [EPNUM] = 0

Packet Count [EPNUM] = 1

- To send multiple maximum packet size packets with a zero-length packet at the end of the transfer, the application must split the transfer into two parts.

The first part sends packets of maximum packet size, and the second part only sends zero-length packets.

First transfer: transfer size[EPNUM] = $x \times \text{MPSIZ}[\text{epnum}]$; packet count = n;

Second transfer: transfer size[EPNUM] = 0; packet count = 1;

3. After an endpoint is enabled for data transfer, the module updates the transfer size register. At the end of the 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 over the USB.
4. Amount of data fed into transmit FIFO = initial transfer size programmed by application - final transfer size after module update
 - Amount of data that has been sent over USB = (Initial packet count programmed by the application - Final packet count after module update) $\times \text{MPSIZ}[\text{EPNUM}]$
 - Amount of data remaining to be sent over USB = (initial transfer size programmed by the application - amount of data already sent over USB)

Internal data flow:

1. The application must set the transfer size and packet count fields in the registers of a specific endpoint and enable that endpoint to send data.
2. The application must also write the necessary data to that endpoint's transmit FIFO.
3. Each time the application writes a packet to the transmit FIFO, the packet size is automatically subtracted from the transfer size of the endpoint. The application continues to fetch data from memory to write to the transmit FIFO until the transfer size for that endpoint becomes 0. After writing data to the FIFO, the "Packets in FIFO" count is incremented (this is a 3-bit count maintained internally by the module, one for each IN endpoint transmit FIFO. In the IN endpoint FIFO, the maximum number of packets maintained by the module is always eight). For zero-length packets, each FIFO has an additional separate flag, and there is no data in the FIFO.
4. After the data is written into the transmit FIFO, the module will send the data out when it receives the IN token. After each packet is sent and the ACK handshake is received in response, the packet count of the endpoint is decremented by 1 until the packet count becomes 0. When a timeout occurs, the packet count is not decremented.
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 receiving the IN token, and the packet count field of the endpoint is zero, the module will generate an "IN token

received when TxFIFO is empty" (ITTXFE) interrupt for the endpoint (Provided that the endpoint's NAK bit is not set). The module replies with a NAK handshake signal on the asynchronous endpoint.

7. The module will internally bring the FIFO pointer back to the beginning and will not generate a timeout interrupt.
8. When the transfer size is 0 and the packet count is 0, a transfer complete (XFRC) interrupt for this endpoint is generated and the endpoint enable is cleared.

Application programming sequence:

1. Program the USBFS_DIEPTSIZx registers with the transfer size and corresponding packet count.
2. Program the USBFS_DIEPCTLx registers using the endpoint characteristics and set the CNAK and EPENA (endpoint enable) bits to 1.
3. When sending a non-zero length packet, the application must poll the USBFS_DTXFSTSx registers (where x is the FIFO number associated with this 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 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 aperiodic IN data transfer apply equally to cyclic IN data transfer (with a slight modification to requirement 2).
 - An application can only send a number of maximum packet size packets or a number of maximum packet size packets, plus a short packet at the end of the transfer.

To send multiple maximum packet size packets plus a short packet at the end of the transmission, the following conditions must be met:

$$\text{transfer size[EPNUM]} = x \times \text{MPSIZ[EPNUM]} + \text{sp}$$

(where x is an integer greater than 0, and sp ranges from 0 to MPSIZ[EPNUM]-1)

If ($\text{sp} > 0$), packet count [EPNUM] = $x + 1$

Otherwise, PacketCount[EPNUM] = x;

MCNT[EPNUM] = Packet Count [EPNUM]

- The application cannot send zero-length packets at the end of the transfer. An application can send a zero-length packet by itself.
- To send a single zero-length packet:

Transfer Size[EPNUM]=0

Packet Count[EPNUM]=1

MCNT[EPNUM]=packet count[EPNUM]

2. An application can only schedule data transfers for one frame at a time.

- $(MCNT - 1) \times MPSIZ < XFERSIZ \leq MCNT \times MPSIZ$
 - $PKTCNT = MCNT$ (in `USBFS_DIEPTSIZx`)
 - If $XFERSIZ < MCNT \times MPSIZ$, the last packet transmitted is a short packet
 - caution: `MCNT` is in `USBFS_DIEPTSIZx`, `MPSIZ` is in `USBFS_DIEPCTLx`, `PKTCNT` is in `USBFS_DIEPTSIZx`, is in `USBFS_DIEPTSIZx`
3. Before receiving the IN token, the application must write the complete data to be sent in the frame into the transmit FIFO. When receiving the IN token, even if the data to be sent in the frame is only one double word missing in the transmit FIFO, the module will perform the operation when the FIFO is empty. When the transmit FIFO is empty:
 - A zero-length packet will be replied on the isochronous endpoint
 - The NAK handshake signal will be replied on the interrupt endpoint

Internal data flow:

1. The application must set the transfer size and packet count fields in the registers of a specific endpoint and enable that endpoint to send data.
2. The application must also write the necessary 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 the endpoint. The application continues to fetch data from memory to write to the transmit FIFO until the transfer size for that endpoint becomes 0.
4. When the periodic endpoint receives the IN token, the module will start sending the data in the FIFO (if there is data in the FIFO). If the FIFO does not have a complete packet of the data to be sent for this frame, the module will generate an "IN token received while TxFIFO is empty" interrupt for that endpoint.
 - A zero-length packet will be replied on the isochronous endpoint
 - The NAK handshake signal will be replied on the interrupt endpoint
5. An endpoint's packet count is decremented by 1 under the following conditions:
 - For isochronous endpoints, when sending a zero-length or non-zero-length packet
 - For interrupt endpoints, decrement when sending ACK handshake
 - When the transfer size and packet count are both 0, a transfer complete interrupt for that endpoint will be generated and the endpoint enable bit will be cleared.
6. During the "periodic frame interval" (controlled by the PFIVL bit in `USBFS_DCFG`), when the module finds that any data in the synchronous IN endpoint FIFO that should be empty in the current frame has not been sent, it will generate an `IISOIXFR` interrupt in `USBFS_GINTSTS`.

Application programming sequence:

1. Program the `USBFS_DIEPCTLx` registers using the endpoint characteristics and set the

- CNAK and EPENA bits to 1.
2. Write the data that needs to be sent in the next frame into the transmit FIFO.
 3. The hardware triggers the ITTXFE interrupt (in USBFS_DIEPINTx) to indicate that the application has not written all the data that needs to be sent into the transmit FIFO.
 4. If the interrupt endpoint was enabled before the interrupt was detected, the interrupt will be ignored. If the interrupt endpoint is not enabled, enable this endpoint so that data can be sent out on the next IN token received.
 5. When the hardware triggers the XFRC interrupt (in USBFS_DIEPINTx), if the ITTXFE interrupt is not generated in USBFS_DIEPINTx, it means that the synchronous IN transfer is successfully completed. Reading the USBFS_DIEPTSIZx register always gets transfer size = 0 and packet count = 0, which means all data has been sent over USB.
 6. Whether or not the ITTXFE interrupt (in USBFS_DIEPINTx) is generated when the XFRC interrupt (in USBFS_DIEPINTx) is set, indicates a successful completion of the interrupt IN transfer. Reading the USBFS_DIEPTSIZx register always gets transfer size = 0 and packet count = 0, which means that all data has been sent over USB.
 7. If none of the aforementioned interrupts are generated when the outstanding synchronous IN transfer (IISOIXFR) interrupt is set in USBFS_GINTSTS, it means that the module has not received at least one periodic IN token in the current frame.

Incomplete sync IN data transfer

This section describes what an application must do for incomplete sync IN data transfers.

Internal data flow:

1. A synchronous IN transfer is considered incomplete when one of the following conditions is met:
 - c) The module received a corrupted sync IN token on at least one sync IN endpoint. At this point, the application detects an incomplete isochronous IN transfer interrupt (bit IISOIXFR in USBFS_GINTSTS).
 - d) The application was too slow to write data to the transmit FIFO and received an IN token before the complete data was written to the FIFO. At this point, the application detects an "IN token received while TxFIFO is empty" interrupt in USBFS_DIEPINTx. The application can ignore this interrupt, as this will eventually generate an outstanding isochronous IN transfer interrupt (bit IISOIXFR in USBFS_GINTSTS) at the end of the periodic frame. The module responds to the received IN token by sending a zero-length packet over USB.
2. The application must stop writing data to the transmit FIFO as soon as possible.
3. The application must set the endpoint's NAK and inhibit bits to 1.
4. The module disables the endpoint, clears the disable bit and triggers the endpoint's Endpoint Disable interrupt.

Application programming sequence:

1. The application MAY ignore the "IN token received while TxFIFO is empty" interrupt in USBFS_DIEPINTx on any isochronous IN endpoint, as this will eventually generate an outstanding isochronous IN transfer interrupt (in USBFS_GINTSTS).
2. A hardware-triggered outstanding sync IN transfer interrupt (in USBFS_GINTSTS) indicates that there is an outstanding sync IN transfer on at least one sync IN endpoint.
3. The application must read the "Endpoint Control" registers of all isochronous IN endpoints to detect the presence of endpoints with outstanding IN data transfers.
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 registers to disable endpoints:
 - SNAK=1 in USBFS_DIEPCTLx
 - EPDIS=1 in USBFS_DIEPCTLx
6. Hardware triggering of the "Endpoint Disabled" interrupt in USBFS_DIEPINTx indicates that the module has disabled the endpoint.
 - At this point, the application must either flush the data in the associated transmit FIFO, or overwrite the existing data in the FIFO by enabling the endpoint for a new transfer in the next frame. To refresh data, the application must use the USBFS_GRSTCTL register.

Stop asynchronous IN endpoint

This section describes how an application can stop an asynchronous endpoint.

Application programming sequence:

1. Disable the IN endpoint to be stopped. Also set the STALL bit to 1.
2. EPDIS=1 in USBFS_DIEPCTLx (when endpoint is enabled)
 - STALL=1 in USBFS_DIEPCTLx
 - STALL bits always have higher priority than NAK bits
3. Hardware triggering of the "Endpoint Disable" interrupt (in USBFS_DIEPINTx) lets the application know that the module has disabled the specified endpoint.
4. The application must empty the aperiodic or periodic transmit FIFO depending on the endpoint type. For aperiodic endpoints, the application must re-enable another aperiodic endpoint without stopping to send data.
5. When the application is ready to end the STALL handshake for this endpoint, it must clear the STALL bit in USBFS_DIEPCTLx.
6. If the application sets or clears the STALL state of an endpoint as a result of receiving a SetFeature.Endpoint Halt command or a ClearFeature.Endpoint Halt command from the host, the STALL bit must be set or cleared before the state phase transfer for that control endpoint.

Special case: Stop controlling the OUT endpoint

If, during the data phase of the control transfer, the host sends more IN/OUT tokens than the value specified in the SETUP packet, the module must reply with STALL for these redundant IN/OUT tokens. 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 by the SETUP packet). Subsequently, when the application receives this interrupt, it must set the STALL bit in the corresponding endpoint control register and clear the interrupt.

35.7 Register description

The application program reads and writes the control and status registers through the AHB slave interface to control the USBFS module. All registers of the USBFS module are 32-bit registers, and their addresses are aligned with 32-bits, so they can only be accessed in 32-bit ways.

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
- Data FIFO (DFIFO) Access Register

The USBFS system control register is different from other register base addresses, and this register is independent of the USBFS module to control the relevant settings of the USBFS module.

Only module global registers, power and clock gating control registers, and data FIFO access 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 the host mode, access the device mode registers. If an illegal access occurs, a pattern mismatch interrupt will be generated and reflected in the USBFS.GINTSTS.NMIS bit in 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.

Control Status Register Memory Map

The host and device mode registers occupy different addresses. All registers are implemented in the AHB clock domain.

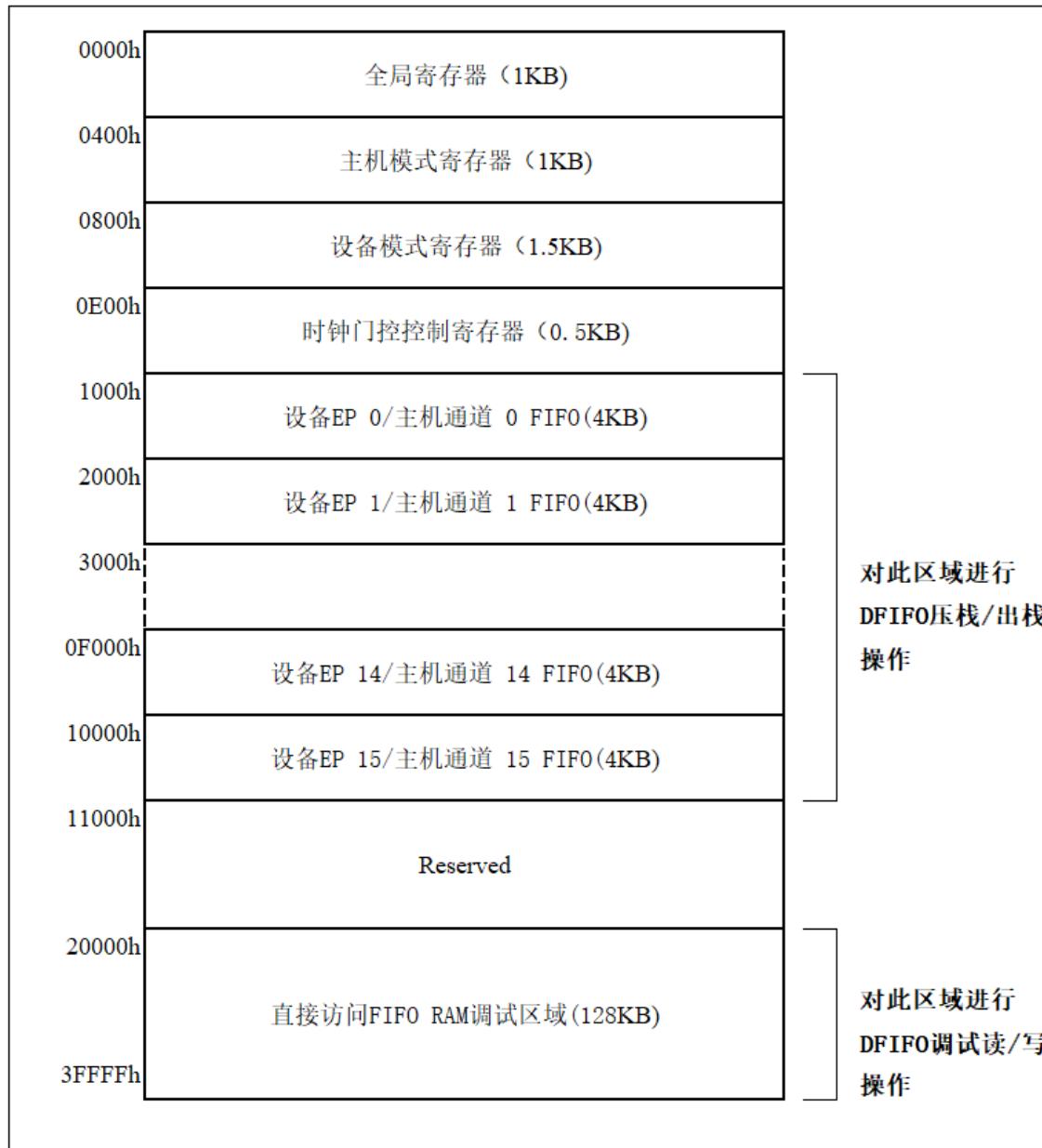


Figure 35-7 USBFS Control Status Register Memory Map

Please refer to the USBFS module register list and base addressTable 35-3 ~Table 35-4 List of USBFS registers.

USB System Control Register Base Address: 0x40055400

Table 35-3 List of USBFS System Control Registers

(USBFS system control register) register name	Offset address	Reset value
USB System Control Register (USB_SYCTLREG)	0x00	0x0000 0000

USBFS module register base address: 0x40080000

Table 35-4 List of USBFS System Control Registers

(USBFS global register) register name	Offset address	Reset value
USBFS VBUS Control Register (USBFS_GVBUSCFG)	0x00	0x00XX 0000
USBFS AHB Control Register (USBFS_GAHBCFG)	0x08	0x0000 0000
USBFS USB Configuration Register (USBFS_GUSBCFG)	0x0c	0x0000 1440
USBFS Reset Register (USBFS_GRSTCTL)	0x10	0x8000 0000
USBFS Module Interrupt Register (USBFS_GINTSTS)	0x14	0x1400 0020
USBFS Interrupt Mask Register (USBFS_GINTMSK)	0x18	0x0000 0000
USBFS Receive Status Debug Read Register (USBFS_GRXSTSR)	0x1c	0x0000 0000
USBFS Receive Status Read and Pop Register (USBFS_GRXSTSP)	0x20	0x0000 0000
USBFS Receive FIFO Size Register (USBFS_GRXFSIZ)	0x24	0x0000 0280
USBFS host aperiodic transmit FIFO size register (USBFS_HNPTXFSIZ)/ Device Endpoint 0 Transmit FIFO Size Register (USBFS_DIEPRXF0)	0x28	0x0280 0280
USBFS Aperiodic Transmission Status Register (USBFS_HNPTXSTS)	0x2c	0x0008 0280
USBFS Module ID Register (USBFS_CID)	0x3c	0x1234 5678
USBFS LPM Configuration Register (USBHS_GLPMCFG)	0x54	0x0000 0000
USBFS periodic transmit FIFO size register (USBFS_HPTXFSIZ)	0x100	0x0280 0500
USBFS device IN endpoint n transmit FIFO size register (USBFS_DIEPTXFx)	0x100+x*4(x=1~15)	0x0280 0500+(x-1)*0x280

(USBFS host control and status register) register name	Offset address	Reset value
USBFS Host Configuration Register (USBFS_HCFG)	0x400	0x0000 0200
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 Periodically Transmit FIFO/Queue Status Register (USBFS_HPTXSTS)	0x410	0x0008 0280
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 Characteristics Register (USBFS_HCCHARx)	0x500+x*0x20(x=0~15)	0x0000 0000
USBFS Host Channel x Interrupt Register (USBFS_HCINTx)	0x508+x*0x20(x=0~15)	0x0000 0000
USBFS Host Channel x Interrupt Mask Register (USBFS_HCINTx)	0x50c+x*0x20(x=0~15)	0x0000 0000
USBFS Host Channel x Transfer Size Register (USBFS_HCTSIZx)	0x510+x*0x20(x=0~15)	0x0000 0000
USBFS Host Channel x DMA Address Register (USBFS_HCDMAx)	0x514+x*0x20(x=0~15)	0xFFFF XXXX

(USBFS device control and status register) register name	Offset address	Reset value
USBFS Device Configuration Register (USBFS_DCFG)	0x800	0x0820 0000
USBFS Device Control Register (USBFS_DCTL)	0x804	0x0000 0002
USBFS Device Status Register (USBFS_DSTS)	0x808	0x0000 0002
USBFS device IN endpoint general interrupt mask register (USBFS_DIEPMSK)	0x810	0x0000 0000
USBFS device OUT endpoint general interrupt mask register (USBFS_DOEPMASK)	0x814	0x0000 0000
USBFS Device Entire Endpoint Interrupt Register (USBFS_DAINT)	0x818	0x0000 0000
USBFS device all endpoint interrupt mask register (USBFS_DAINTMSK)	0x81c	0x0000 0000
USBFS device IN endpoint FIFO empty interrupt mask register (USBFS_DIEPEMPMSK)	0x834	0x0000 0000
USBFS Device IN Endpoint 0 Control Register (USBFS_DIEPCTL0)	0x900	0x0000 8000
USBFS Device IN Endpoint n Control Register (USBFS_DIEPCTLx)	0x900+x*0x20(n=1~15)	0x0000 0000
USBFS Device IN Endpoint n Interrupt Register (USBFS_DIEPINTx)	0x908+x*0x20(n=0~15)	0x0000 0080
USBFS device IN endpoint n transfer size register (USBFS_DIEPTSIz)	0x910+x*0x20(n=0~15)	0x0000 0000
USBFS device IN endpoint n DMA address register (USBFS_DIEPDMAx)	0x914+x*0x20(n=0~15)	0x0000 0000
USBFS device IN endpoint n transmit FIFO status register (USBFS_DTXFSTSx)	0x918+x*0x20(n=0~15)	0x0000 0280
USBFS device OUT endpoint 0 control register (USBFS_DOEPCTL0)	0xb00	0x0000 8000
USBFS device OUT endpoint n control register (USBFS_DOEPCTLx)	0xb00+x*0x20(n=1~15)	0x0000 0000
USBFS device OUT endpoint n interrupt register (USBFS_DOEPINTx)	0xb08+x*0x20(n=0~15)	0x0000 0080
USBFS device OUT endpoint n transfer size register (USBFS_DOEPSIZx)	0xb10+x*0x20(n=0~15)	0x0000 0000
USBFS device OUT endpoint n DMA address register (USBFS_DOEPDMAx)	0xb14+x*0x20(n=0~15)	0XXXXXXXX

(USBFS power and clock gate empty control register) register name	Offset address	Reset value
USBFS Clock Gating Control Register (USBFS_GCCTL)	0xe00	0x0000 0000

35.7.1 USB System Control Register

35.7.1.1 USB System Control Register (USB_SYCTLREG)

USB System Control Register

offset address: 0x000

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	USBHS_S_NF_E	USBHS_NFS[1:0]	-	-	-	-	-	USBF_S_NF_E	USBFS_NFS[1:0]		
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	USBHS_S_FPHYE	USBHS_S_SO_FEN	USBHS_S_DF_B	-	-	-	-	-	-	USBF_S_SO_FEN	USBF_S_DF_B
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b27	Reserved	-	The reset value must be maintained.										R/W		
b26	USBHS_NFE	USBHS Filter Enable Register	USBHS Filter Enable Register This register is used to control the switch of the USBHS on-chip full-speed PHY DP/DM analog filter in STOP mode 0: Analog filter off 1: Analog filter is on, please refer to the setting of USBHS_NFS[1:0] for the filter range										R/W		
b25~b24	USBHS_NFS	USBHS Filter Selection Register	USBHS Filter Selection Register This register is used to control the filter range of the USBHS on-chip full-speed PHY DP/DM analog filter in STOP mode 00b: Filter width gear 1 01b: Filter width gear 2 10b: Filter width gear 3 11b: Filter width gear 4 For the specific value of each gear, please refer to the chapter "Electrical Characteristics USB On-Chip Full-speed PHY STOP Mode Filter Characteristics".										R/W		
b23~b19	Reserved	-	The reset value must be maintained.										R/W		
b18	USBFS_NFE	USBFS Filter Enable Register	USBFS Filter Enable Register This register is used to control the switch of the USBFS on-chip full-speed PHY DP/DM analog filter in STOP mode 0: Analog filter off 1: Analog filter is on, please refer to the setting of USBFS_NFS[1:0] for the filter range										R/W		
b17~b16	USBFS_NFS	USBFS filter selection register	USBFS filter selection register This register is used to control the filter range of the USBFS on-chip full-speed PHY DP/DM analog filter in STOP mode 00b: Filter width gear 1 01b: Filter width gear 2 10b: Filter width gear 3 11b: Filter width gear 4 For the specific values of each gear, please refer to the chapter "49. Electrical Characteristics 49.3.34 USB On-Chip Full-speed PHY Filtering Characteristics in STOP Mode".										R/W		
b15~b11	Reserved	-	The reset value must be maintained.										R/W		
b10	USBHS_FSPHYE	USBHS On-Chip Full-Speed PHY Enable Bit	USBHS On-Chip Full-Speed PHY Enable Bit 0: USBHS On-Chip Full-Speed PHY Suspended 1: USBHS on-chip full-speed PHY enable										R/W		
b9	USBHS_SOFEN	USBHS SOF pulse output enable bit	When the USBHS host sends a SOF or the device successfully receives the SOF, a SOF pulse with a width of 16 system clock cycles 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		

b8	USBHS_DFB	USBHS VBUS/ID pin internal debounce filter bypass enable bit	USBHS VBUS/ID pin module internal debounce filter bypass enable bit 0: The debounce filter inside the module is valid 1: Bypass module internal debounce filter Note: Accessible in both device mode and host mode.	R/W
b7~b2	Reserved	-	The reset value must be maintained.	R/W
b1	USBFS_SOFEN	USBFS SOF pulse output enable bit	When the USBFS host sends a SOF or the device successfully receives the SOF, a 16 system clock cycle width SOF pulse 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	USBFS_DFB	USBFS VBUS/ID pin internal debounce filter bypass enable bit	USBFS VBUS/ID pin module internal debounce filter bypass enable bit 0: The debounce filter inside the module is valid 1: Bypass module internal debounce filter Note: Accessible in both device mode and host mode.	R/W

35.7.2 USBFS global registers

These registers are available in both host and device modes and do not need to be reprogrammed when switching between these two modes. Unless otherwise specified, bit values in register descriptions are represented in binary.

35.7.2.1 USBFS VBUS Control Register (USBFS_GVBUSCFG)

VBUS Configuration Register

offset address: 0x00

Reset value: 0x00XX 0000

This register can be used to set the VBUS value to ignore the state 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										VBUS VAL	VBUS OVEN	Reserved			
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b8	Reserved	-	The reset value must be maintained.										R/W		
B7	VBUSVAL	VBUS value	VBUS Value It is used to set the VBUS value of USBFS. When it is set to 1 and VBUSOVEN is set to 1, the USBFS is powered on. Note: Only accessible in device mode.										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 the USBFS CORE. The value of VBUSVAL is valid only when this bit is set to 1. Note: Only accessible in device mode.										R/W		
b5~b0	Reserved	-	The reset value must be maintained.										R/W		

35.7.2.2 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 role modes.
This register mainly contains configuration parameters related to the AHB system.

The application must program this register before starting any AHB or USB transaction. 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															
Bit	Marking	Place name	Function										Read and write		
b31~b9	Reserved	-	The reset value must be maintained.										R/W		
b8	PTXFELVL	Periodic TxFIFO Empty Threshold	Periodic TxFIFO empty level Indicates when the Periodic TxFIFO Empty Interrupt bit in the Module Interrupt Register (PTXFE bit in USBFS_GINTSTS) is triggered. 0: PTXFE (located in USBFS_GINTSTS) interrupt indicates that the periodic TxFIFO is half empty 1: PTXFE (located in USBFS_GINTSTS) interrupt indicates that the periodic TxFIFO is completely empty Note: Accessible only in host mode.										R/W		
B7	TXFELVL	Device TxFIFO Empty Threshold	Device TxFIFO empty threshold (Tx FIFO empty level) In device mode, this bit indicates when the IN endpoint transmit FIFO empty interrupt (TXFE in USBFS_DIEPINTx) is triggered. 0: TXFE (located in USBFS_DIEPINTx) interrupt indicates that the IN endpoint TxFIFO is half empty 1: TXFE (located in USBFS_DIEPINTx) interrupt indicates that the IN endpoint TxFIFO is completely empty Note: Only accessible in device mode.										R/W		
b6	Reserved	-	The reset value must be maintained.										R/W		
b5	DMAEN	DMA enable	DMA enable (DMA enable) 0: The module operates in slave mode 1: The 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: Keep Note: Accessible in both device mode and host mode.										R/W		
b0	GINTMSK	global interrupt mask	Global interrupt mask This bit is used to mask or unmask global interrupts. The Interrupt Status Register is updated by the module regardless of the setting of this bit. 0: Block interrupts triggered by the application 1: Unmask the interrupt triggered by the application Note: Accessible in both device mode and host mode.										R/W		

35.7.2.3 USBFS USB Configuration Register (USBFS_GUSBCFG)

USBFS USB configuration register

offset address: 0x00C

Reset value: 0x0000 01440

This register can be used to configure the module after power-up or changing role mode. It contains configuration parameters related to USB and USB-PHY.

The application must program this register before starting any AHB or USB transaction. 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	FDMOD	FHMOD	Reserved																		
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0						
Reserved		TRDT[3:0]				Reserved			PHYSEL	Reserved			TOCAL[2:0]								
<hr/>																					
Bit	Marking	Place name	Function										Read and write								
b31	Reserved	-	The reset value must be maintained.										R/W								
b30	FDMOD	Force 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 pin. 0: Normal mode, depending on the input state of the USBFS_ID pin 1: Force device mode After setting the force bit, 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	Force host mode	Force host mode Writing a 1 to this bit forces the module to 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: Force host mode After setting the force bit, 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/W								
b13~b10	TRDT	USB turnaround time	USB turnaround time Set the turnaround time in units of PHY clocks. To calculate the value of TRDT, use the following formula: $TRDT = 4 \times AHB\ clock + 1\ PHY\ clock$ E.g: 1. If AHB clock frequency = 84 MHz (PHY clock frequency = 48 MHz), then TRDT is set to 9. 2. If AHB clock frequency = 48 MHz (PHY clock frequency = 48 MHz), then TRDT is set to 5. Note: Only accessible in device mode.										R/W								
b9~b7	Reserved	-	The reset value must be maintained.										R/W								
b6	PHYSEL	Full-speed serial transceiver selection	Full Speed serial transceiver select										R/W								
b5~b3	Reserved	-	The reset value must be maintained.										R/W								
b2~b0	TOCAL	FS Timeout Calibration	FS timeout calibration Additional latency introduced by the PHY includes the number of PHY clocks set by the application in this field, and the module's full-speed inter-packet timeout interval. The delay introduced by different PHYs affects the state of the data line differently. The USB standard timeout value for full-speed operation is 16 to 18 (inclusive) bit times. The application must program this field according to the enumeration speed. The number of bit times added per PHY clock is 0.25 bit times. Note: Accessible in both device mode and host mode.										R/W								

35.7.2.4 USBFS Reset Register (USBFS_GRSTCTL)

USBFS reset register

offset address: 0x10

Reset value: 0x8000 0000

The application program resets various hardware features in the module through this register.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
AHBI DL	DMA REQ														
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
		Reserved						TXFNUM[4:0]		TXFF LSH	RXFF FLSH	Reser ved	FCRS T	HSRS T	CSRS T
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31	AHBIDL	AHB master is idle	AHB master idle Indicates that the AHB master state machine is in an idle condition. Note: Accessible in both device mode and host mode.										R		
b30	DMAREQ	AHB master is idle	DMA request signal This bit indicates that a DMA request is in progress. for debugging. Note: Accessible in both device mode and host mode.										R		
b29~b11	Reserved	-	Read as "0", write as "0"										R/W		
b10~b6	TXFNUM	TxFIFO number	TxFIFO number FIFO number for FIFO flushing using the TxFIFO flush bits. This field can only be changed after the module clears the TxFIFO flush bit. <ul style="list-style-type: none"> ● 00000: — Flush aperiodic TxFIFO in master mode — Flush Tx FIFO 0 in device mode <ul style="list-style-type: none"> ● 00001: — Flush periodic TxFIFO in master mode — Flush TXFIFO 1 in device mode <ul style="list-style-type: none"> ● 00010: refresh TXFIFO 2 in device mode <ul style="list-style-type: none"> ... ● 00101: refresh TXFIFO 15 in device mode ● 10000: flush all transmit FIFOs in device mode or host mode Note: Accessible in both device mode and host mode.										R/W		
b5	TXFFLSH	TxFIFO refresh	TxFIFO flush (TxFIFO flush) This bit selectively flushes one or all transmit FIFOs, but cannot do so while the module is processing a communication transaction. The application can only write to this bit after verifying that the module is not currently reading or writing to the TxFIFO. Use the following registers to confirm: <ul style="list-style-type: none"> - read: NAK valid interrupt ensures that the module is not currently performing a read of the FIFO - Write: The AHBIDL bit in USBFS_GRSTCTL ensures that the module is not currently doing any writes to the FIFO Note: Accessible in both device mode and host mode.										R/W		
b4	RXFFFLSH	RxFIFO flush	RxFIFO flush (RxFIFO flush) An application can use this bit to flush the entire RxFIFO, but must first ensure that the module is not currently processing a communication transaction. The application can write to this bit only after confirming that the module is not currently reading or writing to the RxFIFO. The application must wait until this bit is cleared before performing other operations. It usually takes 8 clock cycles to wait (whichever is the slowest of the PHY or AHB clock). Note: Accessible in both device mode and host mode.										R/W		
b3	Reserved	-	The reset value must be maintained.										R/W		
b2	FCRST	Host frame counter reset	Host frame counter reset When the application program writes to this bit, the frame counter in the module is reset. 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		

			HCLK domain logic soft reset (HCLK soft reset) Applications use this bit to refresh control logic in the AHB clock domain. Only the AHB clock domain pipeline is reset. The FIFO is not refreshed by this bit. All state machines in the AHB clock domain are reset to the idle state after the transaction on the AHB is terminated according to the protocol. The CSR control bit used by the AHB clock domain state machine is cleared. To clear this interrupt, the status mask bit generated by the AHB clock domain state machine and used to control the state of the interrupt needs to be cleared. Since the interrupt status bit is not cleared, the application can get the state. This bit is self-clearing and will be cleared by the module after all necessary logic in it is 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
b1	HSRST	HCLK domain logic soft reset	Module soft reset (Core soft reset) Reset the HCLK and PCLK domains as follows: Clears individual interrupts and all CSR register bits except: — GATEHCLK bit in USBFS_GCCTL — STPPCLK bit in USBFS_GCCTL — FLSSPCS bit in USBFS_HCFG — DSPD bit in USBFS_DCFG Resets all module state machines (except AHB slaves) to idle state and clears all transmit and receive FIFOs. Terminate all transactions on the AHB master as soon as possible after the last data phase of an AHB transfer. Immediately terminate all transactions on the USB. The application can write to this bit at any time when the module needs to be reset. This bit is self-clearing and the module will clear this bit after all necessary logic in it is reset, a process that takes several clocks depending on the current state of the module. Once this bit is cleared, software must wait at least 3 PHY clocks before accessing the PHY domain (synchronization delay). In addition, software It must also be determined that Bit 31 in this register is set (AHB master is idle) before running. Software reset is usually used in two situations, one is during software development, and the other is after the user has dynamically changed the PHY select bits in the USB configuration registers listed above. When the user changes the PHY, the corresponding clock will be selected for the PHY and used in the PHY domain. Once a new clock is selected, the PHY domain must be reset for proper operation. Note: Accessible in both device mode and host mode.	R/W
b0	CSRST	Module soft reset		

35.7.2.5 USBFS Global Interrupt Status Register (USBFS_GINTSTS)

USBFS interrupt status register

offset address: 0x14

Reset value: 0x14000020

This register is used to interrupt the application with system level events in the current mode (device mode or host mode).

Some bits in this register are valid only in host mode, while other bits are valid only in device mode. In addition, this register can also indicate the current mode.

FIFO status interrupts are read-only; if software reads or writes to the FIFO while these interrupts are being handled, the FIFO interrupt flag is automatically cleared.

Before enabling the interrupt bit, the application must clear the USBFS_GINTSTS register during initialization to avoid any interrupts before initialization.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
WKUINT	VBUSVINT	DISCINT	CIDSCHG	LPMI NT	PTXF E	HCINT	HPRTINT	Reser ved	DATA FSUSP	IPXFR / INCO MPIS OOUT	IISOIXFR	OEPI NT	IEPIN T	Reser ved	Reser ved
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
EOPF	ISOODRP	ENUMDNE	USBRSST	USBSSUSP	ESUSP	Reser ved	Reser ved	GONAKEFF	GINAKEFF	NPTXFE	RXFN E	SOF	Reser ved	MMIS	CMOD

Bit	Marking	Place name	Function	Read and write
b31	WKUINT	Resume/remote wakeup interrupt detected	Resume/remote wakeup detected interrupt In device mode, this interrupt is triggered when a resume signal is detected on the USB bus. In host mode, this interrupt is triggered when a remote wakeup is detected on the USB. This bit is cleared by software by writing a 1 to it. Note: Accessible in both device mode and host mode.	R/W
b30	VBUSVINT	VBUS active interrupt	VBUS valid interrupt In device mode, this interrupt will be triggered when the USBFS_VBUS pin is detected from low to high. This bit is cleared by software by writing a 1 to it. Note: Only accessible in device mode.	R/W
b29	DISCINT	Disconnect interruption detected	Disconnect detected interrupt This interrupt is triggered when a device disconnection is detected. This bit is cleared by software by writing a 1 to it. Note: Accessible only in host mode.	R/W
b28	CIDSCHG	Connector ID line state change interrupt	Connector ID status change The module sets this bit when the connector ID line state changes. This bit is cleared by software by writing a 1 to it. Note: Accessible in both device mode and host mode.	R/W
b27	LPMINT	LPM interrupt	LPM interrupt device mode This bit is set when the device has received a correct LPM transmission and a valid reply. Host mode This bit is set to 1 when the host receives a valid response after sending LPM, or exceeds the set number of retries.	R/W

b26	PTXFE	Periodic TxFIFO Empty Interrupt	<p>Periodic TxFIFO empty interrupt This interrupt is triggered when the periodic transmit FIFO is half empty or completely empty and there is room in the periodic request queue to write at least one entry. Whether the FIFO is half empty or fully empty is determined by the Periodic TxFIFO Empty Level bit in the USBFS_GAHBCFG register (PTXFELVL bit in USBFS_GAHBCFG).</p> <p>Note: Accessible only in host mode.</p>	R
b25	HCINT	host channel interrupt	<p>Host channels interrupt When the module sets this bit, it indicates that there is a pending interrupt (in host mode) on a channel in the module. The application must read the host USBFS_HAINT register to determine the exact channel number on which the interrupt occurred, and then read the corresponding USBFS_HCINTx registers to determine the exact cause of the interrupt. The application must clear the appropriate status bit in the USBFS_HCINTx register before clearing this bit.</p> <p>Note: Accessible only in host mode.</p>	R
b24	HPRTINT	Host port interrupt	<p>Host port interrupt When the module sets this bit to 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 caused this interrupt. The application must clear the appropriate status bit in the USBFS_HPRT register before clearing this bit.</p> <p>Note: Accessible only in host mode.</p>	R
b23	Reserved	-	The reset value must be maintained.	R/W
b22	DATAFSUSP	Data fetch pending	<p>Data fetch suspended This interrupt is only valid 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 detecting an endpoint mismatch, the application would do the following:</p> <ul style="list-style-type: none"> — Set the global aperiodic IN NAK handshake signal to 1 — Disable IN endpoints — Empty FIFO — Determine the token sequence from the IN token sequence learning queue — re-enable endpoint — If the global aperiodic IN NAK is cleared but the module has not yet acquired data for the IN endpoint and an IN token has been received, clear the global aperiodic IN NAK handshake signal: The module will generate an "IN token received while FIFO is empty" interrupt. Then, USBFS sends a NAK response to the host. To avoid this from happening, the application can check for 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 clearing the global IN NAK handshake signal. <p>This bit is cleared by software by writing a 1 to it.</p> <p>Note: Only accessible in device mode.</p>	R/W
b21	IPXFR/ INCOMPISOOUT	Incomplete periodic transmission/ OUT sync transfer not completed	<p>IPXFR: Incomplete periodic transfer In master mode, the module sets this interrupt bit if there are outstanding periodic transactions still pending that are scheduled to complete during the current frame.</p> <p>This bit is cleared by software by writing a 1 to it.</p> <p>Note: Accessible only in host mode.</p> <p>INCOMPISOOUT: Incomplete isochronous OUT transfer In device mode, when the module sets this interrupt, it indicates that the transfer on at least one isochronous OUT endpoint in the current frame is not complete. This interrupt is triggered along with the Periodic End of Frame Interrupt (EOPF) bit in this register.</p> <p>This bit is cleared by software by writing a 1 to it.</p> <p>Note: Only accessible in device mode.</p>	R/W
b20	IISOIXFR	IN SYNC TRANSFER NOT COMPLETED	<p>Incomplete isochronous IN transfer When the module sets this interrupt to 1, it indicates that there is an incomplete transfer on at least one synchronous IN endpoint in the current frame. This interrupt is triggered along with the Periodic End of Frame Interrupt (EOPF) bit in this register.</p> <p>This bit is cleared by software by writing a 1 to it.</p> <p>Note: Only accessible in device mode.</p>	R/W

b19	OEPINT	OUT endpoint interrupt	<p>OUT endpoint interrupt When the module sets this bit, it indicates that there is a pending interrupt (in device mode) on one of the OUT endpoints in the module. The application must read the host USBFS_DAINT register to determine the exact number of the OUT endpoint on which the interrupt occurred, and then read the corresponding USBFS_DOEPINTx registers to determine the exact cause of the interrupt. The application must clear the corresponding status bit in the corresponding USBFS_DOEPINTx register before clearing this bit.</p> <p>Note: Only accessible in device mode.</p>	R
b18	IEPINT	IN endpoint interrupt	<p>IN endpoint interrupt When the module sets this bit, it indicates that there is a pending interrupt (in device mode) on one of the IN endpoints in the module. The application must read the host USBFS_DAINT 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 interrupt. The application must first clear the corresponding status bit in the USBFS_DIEPINTx register before this bit can be cleared.</p> <p>Note: Only accessible in device mode.</p>	R
b17~b16	Reserved	-	The reset value must be maintained.	R/W
b15	EOPF	Periodic end of frame interrupt	<p>End of periodic frame interrupt Indicates that the current frame has reached the period specified by the Periodic Frame Interval field in the USBFS_DCFG register (the PFIVL bit in USBFS_DCFG). This bit is cleared by software by writing a 1 to it.</p> <p>Note: Only accessible in device mode.</p>	R/W
b14	ISOODRP	Drop sync OUT packet interrupt	<p>Isochronous OUT packet dropped interrupt If the module cannot write the synchronous OUT data packet to the RxFIFO due to insufficient space in the RxFIFO to accommodate the maximum data packet of the synchronous OUT endpoint, the module will set this bit to 1. This bit is cleared by software by writing a 1 to it.</p> <p>Note: Only accessible in device mode.</p>	R/W
b13	ENUMDNE	Enumeration complete interrupt	<p>Enumeration done interrupt When the module sets this bit, it indicates that the speed enumeration is complete. The application must read the USBFS_DSTS register to get the enumeration speed. This bit is cleared by software by writing a 1 to it.</p> <p>Note: Only accessible in device mode.</p>	R/W
b12	USBRST	USB reset interrupt	<p>USB reset interrupt When the module sets this bit to 1, it indicates that a reset signal was detected on the USB. This bit is cleared by software by writing a 1 to it.</p> <p>Note: Only accessible in device mode.</p>	R/W
b11	USBSUSP	USB suspend interrupt	<p>USB suspend interrupt When the module sets this bit, it indicates that a suspend state was detected on the USB. When the idle state on the USB bus remains for 3ms, the module will enter the suspend state. This bit is cleared by software by writing a 1 to it.</p> <p>Note: Only accessible in device mode.</p>	R/W
b10	ESUSP	early pending interrupt	<p>Early suspend interrupt When the module sets this bit to 1, it indicates that the USB has been detected to be in an idle state for 3ms. Note: Only accessible in device mode.</p>	R/W
b9~b8	Reserved	-	The reset value must be maintained.	R/W
B7	GONAKEFF	Global OUT NAK valid interrupt	<p>Global OUT NAK effective interrupt Indicates that the "set global OUT NAK" bit in the USBFS_DCTL register set by the application (SGONAK bit in USBFS_DCTL) has taken effect in the module. This bit can be cleared by writing to the "Clear Global OUT NAK" bit in the USBFS_DCTL register (CGONAK bit in USBFS_DCTL).</p> <p>Note: Only accessible in device mode.</p>	R
b6	GINAKEFF	Global aperiodic IN NAK valid interrupt	<p>Global IN nonperiodic NAK effective interrupt Indicates that the "set global aperiodic IN NAK" bit in the USBFS_DCTL register set by the application (SGINAK bit in USBFS_DCTL) has taken effect in the module. That is, the module has sampled the global IN NAK bit set by the application and the result has taken effect. This bit can be cleared by clearing the "Clear Global Aperiodic IN NAK" bit in the USBFS_DCTL register (CGINAK bit in USBFS_DCTL). 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.</p> <p>Note: Only accessible in device mode.</p>	R

b5	NPTXFE	Aperiodic TxFIFO Empty Interrupt	<p>Non-periodic TxFIFO empty interrupt This interrupt is triggered when the aperiodic TxFIFO is half-empty or completely empty, and there is at least one entry available in the aperiodic send request queue. Whether the FIFO is half empty or fully empty is determined by the Aperiodic TxFIFO Empty Level bit in the USBFS_GAHBCFG register (TXFELVL bit in USBFS_GAHBCFG).</p> <p>Note: Accessible only in host mode.</p>	R
b4	RXFNE	RxFIFO not empty interrupt	<p>RxFIFO non-empty interrupt Indicates that there is at least one packet in the RxFIFO waiting to be read.</p> <p>Note: Accessible in both host mode and device mode.</p>	R
b3	SOF	start of frame interrupt	<p>Start of frame interrupt In host mode, when the module sets this bit, it indicates that a SOF (FS) or Keep-Alive (LS) signal has been sent on the USB. The application must set this bit to clear this interrupt.</p> <p>In device mode, when the module sets this bit, it indicates that a SOF token has been received on the USB. The application can obtain the current frame number by reading the device status register. This interrupt only occurs when the module is running in FS mode.</p> <p>This bit is cleared by software by writing a 1 to it.</p> <p>Note: Accessible in both host mode and device mode.</p>	R/W
b2	Reserved	-	The reset value must be maintained.	R/W
b1	MMIS	pattern mismatch interrupt	<p>Mode mismatch interrupt The module sets this bit when the application attempts to access the following registers: — The module operates in device mode to access the host mode registers — Module running in host mode to access device mode registers A 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>This bit is cleared by software by writing a 1 to it.</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. 0: device mode 1: Host mode</p> <p>Note: Accessible in both host mode and device mode.</p>	R

35.7.2.6 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 program. If an interrupt bit is masked, the interrupt associated with that bit will not be generated.

However, the module interrupt (USBFS_GINTSTS) register bit corresponding to this interrupt is still set.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
WKUIM	VBUSVIM	DISCM	CIDSCHGM	LPMINTM	PTXFEM	HCIM	HPRTIM	Reserved	DATAFSUSPM	IPXFRM/INCO	IISOIXFRM	OEPI	IEPIM	Reserved	Reserved
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
EOPFM	ISODRPM	ENUMDNE	USBRSTM	USBSUSPM	ESUSPM	Reserved	Reserved	GONAKEFF	GINAKEFF	NPTXFEM	RXFNEM	SOFM	Reserved	MMISM	Reserved

Bit	Marking	Place name	Function	Read and write
b31	WKUIM	Resume/Remote Wakeup Interrupt Masking Detected	Resume/remote wakeup detected interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in both host mode and device mode.	R/W
b30	VBUCSVIM	VBUS active interrupt mask	VBUS valid interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b29	DISCM	Disconnect detected interrupt mask	Disconnect detected interrupt interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible only in host mode.	R/W
b28	CIDSCHGM	Interrupt Connector ID Line State 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	LPMINTM	LPM interrupt masking	LPM interrupt mask 0: Mask interrupt 1: Enable interrupt	R/W
b26	PTXFEM	Periodic TxFIFO Empty Interrupt Mask	Periodic TxFIFO empty interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible only in host mode.	R/W
b25	HCIM	Host channel interrupt mask	Host channels interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible only in host mode.	R/W
b24	HPRTIM	Host Port Interrupt Mask	Host port interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible only in host mode.	R/W
b23	Reserved	-	The reset value must be maintained.	R/W
b22	DATAFSUSPM	Data fetch pending interrupt mask	Data fetch suspended interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W

			IPXFR: Incomplete periodic transfer interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible only in host mode.	
b21	IPXFRM/ INCOMPISOOUTM	Incomplete periodic transmission interrupt mask/ Incomplete OUT synchronous transfer interrupt mask	INCOMPISOOUT: Incomplete isochronous OUT transfer interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b20	IISOIXFRM	Incomplete IN isochronous transfer interrupt mask	Incomplete isochronous IN transfer interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b19	OEPIM	OUT endpoint interrupt mask	OUT endpoint interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b18	IEPIM	IN endpoint interrupt mask	IN endpoint interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b17~b16	Reserved	-	The reset value must be maintained.	R/W
b15	EOPFM	Periodic end of frame interrupt mask	End of periodic frame interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b14	ISOODRPM	Drop sync OUT packet interrupt mask	Isochronous OUT packet dropped interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b13	ENUMDNEM	Enumeration complete interrupt mask	Enumeration done interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b12	USBRSTM	USB reset interrupt mask	USB reset interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b11	USBSUSPM	USB suspend interrupt mask	USB suspend interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b10	ESUSPM	early pending interrupt mask	Early suspend interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b9~b8	Reserved	-	The reset value must be maintained.	R/W
B7	GONAKEFFM	Global OUT NAK active interrupt mask	Global OUT NAK effective interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b6	GINAKEFFM	Global Aperiodic IN NAK Active Interrupt Mask	Global IN nonperiodic NAK effective interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Only accessible in device mode.	R/W
b5	NPTXFEM	Aperiodic TxFIFO Empty Interrupt Mask	Non-periodic TxFIFO empty interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible only in host mode.	R/W
b4	RXFNEM	RxFIFO non-empty interrupt mask	RxFIFO non-empty interrupt mask 0: Mask interrupt 1: Enable interrupt Note: Accessible in both host mode and device mode.	R/W
b3	SOFM	start of frame 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/W
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/W

35.7.2.7 USBFS Receive Status Debug Read/USBFS Status Read and Pop Register (USBFS_GRXSTSR/USBFS_GRXSTSP)

USBFS Receive status debug read/USBFS status read and pop registers

Read offset address: 0x01C

Offset address to pop from the stack: 0x020

Reset value: 0x0000 0000

Reading the receive status debug read register will return the contents of the top of the receive FIFO. Reading the receive status read and pop registers will additionally pop the data entry at the top of the RxFIFO. Receive status content is interpreted differently in host mode and device mode.

When the receive FIFO is empty, the module ignores the read or pop operation of this register and returns the value 0x0000 0000. The application must only pop the Receive Status FIFO when the Receive FIFO Not Empty bit of the Module Interrupt Register (RXFNE bit in USBFS_GINTSTS) is set.

Host 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
DPID[0]	BCNT[10:0]										CHNUM_EPNUM[3:0]				

Bit	Marking	Place name	Function	Read and write
b31~b21	Reserved	-	The reset value must be maintained.	R/W
b20~b17	PKTSTS	packet status	Packet status Indicates the status of received packets 0010: IN packet received 0011: IN transfer completed (trigger interrupt) 0101: Data synchronization error (trigger interrupt) 0111: Pause the channel (trigger interrupt) Other values: Keep	R
b16~b15	DPID	data PID	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_EPNUM	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
DPID[0]	BCNT[11:0]										CHNUM_EPNUM[3:0]				
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b21	Reserved	-	The reset value must be maintained.										R/W		
b20~b17	PKTSTS	packet status	Packet status Indicates the status of received packets 0001: Global OUT NAK (triggered interrupt) 0010: OUT packet received 0011: OUT transfer completed (triggered interrupt) 0100: SETUP transaction completed (interrupt triggered) 0110: SETUP packet received Other values: Keep										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	CHNUM_EPNUM	endpoint number	Endpoint number Indicates the endpoint number to which the currently received packet belongs.										R		

35.7.2.8 USBFS Receive FIFO Size Register (USBFS_GRXFSIZ)

USBFS Receive FIFO size register

offset address: 0x024

Reset value: 0x0000 0280

This application can program the amount 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]										

Bit	Marking	Place name	Function	Read and write
b31~b11	Reserved	-	The reset value must be maintained.	R/W
b10~b0	RXFD	RxFIFO depth	RXFD: RxFIFO depth In 32-bit words. The minimum value is 16 The maximum value is 256 The power-on reset value is the maximum Rx data FIFO depth.	R/W

35.7.2.9 USBFS Host Aperiodic Transmit FIFO Size Register (USBFS_HNPTXFISZ)/Endpoint 0 Transmit FIFO Size (USBFS_DIEPTXF0)

USBFS Host non-periodic transmit FIFO size register/Device endpoint0 transmit FIFO size register
offset address: 0x028

Reset value: 0x02800280

This application can program the amount 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
NPTXFDF[15:0]/TX0FD[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
NPTXFSA[15:0]/TX0FSA[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	NPTXFDF/ TX0FD	Aperiodic TxFIFO Depth/Endpoint 0 TxFIFO Depth	Host mode: NPTXFDF Non-periodic TxFIFO depth In 32-bit words. The minimum value is 16 The maximum value is 256 Device Mode: TX0FD Endpoint 0 TxFIFO depth In 32-bit words. The minimum value is 16 The maximum value is 256	R/W											
b15~b0	NPTXFSA/ TX0FSA	Aperiodic transmit RAM start address / Endpoint 0 transmit RAM start address	Host mode: NPTXFSA Non-periodic transmit RAM start address This field contains the memory start address of the aperiodic transmit FIFO RAM. Device Mode: TX0FSA Endpoint 0 transmit RAM start address This field contains the memory start address of the endpoint 0 transmit FIFO RAM.	R/W											

35.7.2.10 USBFS Aperiodic Transmit 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: 0x00080280

This read-only register contains free space information for the aperiodic TxFIFO and the aperiodic transmit request queue.

This register is only valid in host mode, not in device mode.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reser ved	NPTXQTOP[6:0]														NPTQXSAR[7:0]
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
NPTXFSAV[15:0]															

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	The reset value must be maintained.	R/W
b30~b24	NPTXQTOP	Aperiodic send request queue top	Top of the non-periodic transmit request queue The entry currently being processed by the MAC in the aperiodic send request queue. Bits 30:27: Channel/endpoint number Bit 26:25: — 00: IN/OUT token — 01: Transmit packet of length zero — 11: Channel stop command Bit 24: Terminate (last entry for selected channel)	R
b23~b16	NPTQXSAR	Aperiodic send request queue free space	Aperiodic send request queue free space (Non-periodic transmit request queue space available) Indicates the amount of free space available in the aperiodic send request queue. In host mode, this queue holds IN and OUT requests. 0: The aperiodic send request queue is full 1: 1 position available 2: 2 positions available n: n positions available (where n ranges: 0~8) Other values: Keep	R
b15~b0	NPTXFSAV	Aperiodic TxFIFO free space	Non-periodic TxFIFO space available Indicates the amount of free space available in the aperiodic TxFIFO. In 32-bit words. 00: Aperiodic TxFIFO is full 01: 1 word available 10: 2 words available 0xn: n words available (where n ranges: 0~256) Other values: Keep	R

35.7.2.11 USBFS Module ID Register (USBFS_CID)

USBFS core ID register

offset address: 0x03C

Reset value: 0x12345678

This register is the programmable user configuration ID 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]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b0	PRODUCT_ID	Product ID field	Product ID field Application-programmable ID field.										R/W		

35.7.2.12 USBFS LPM Configuration Register (USBFS_GLPMCFG)

USBFS LPM configuration register

offset address: 0x054

Reset value: 0x00000000

This register is a programmable LPM configuration register.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16													
Reserved		ENBS EL	LPMRCNTSTS[2:0]			SNDL PM	LPMRCNT[2:0]			LPMCHIDX[3:0]				L1RS MOK														
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0													
SLPS TS	LPMRSP[1:0]		L1DS EN	BSELTHRS[3:0]				L1SS EN	REM WAKE	BSEL[3:0]			LPMA CK	LPME N														
<hr/>																												
Bit	Marking		Place name		Function									Read and write														
b31~b29	Reserved		-		The reset value must be maintained.									R/W														
b28	ENBSEL	BSEL enable bit		BSEL enable bit 0: Follow USB 2.0 Link Power Management Addendum Engineering Change Notice to the USB 2.0 specification, July 16, 2007 1: Follow Errata for USB 2.0 ECN: Link Power Management (LPM) - 7/2007												R/W												
b27~b25	LPMRCNTSTS [2:0]	LPM Retry Status Register	LPM Retry Status Register Record the number of retries remaining for LPM, only valid in host mode												R													
b24	SENDLPM	Send LPM transport	Send LPM transport 0: Do not send LPM transfers 1: Send LPM transfers (EXT and LPM tokens) This bit is automatically cleared when an acknowledgement (ACK, STALL or NYET) is received from the device. Only supported in host mode.												R/W													
b23~b21	LPMRCNT[2:0]	LPM retry count register	LPM retry count register When an ERROR response is received, the host sends the number of LPM transmission retries until a valid response (ACK, STALL or NYET) is received Only supported in host mode.												R/W													
b20~b17	LPMCHIDX[2: 0]	LPM transmit channel index register	LPM transmit channel index register The host channel index used for LPM transmission, the hardware automatically inserts the device address and endpoint information set by the channel. Only supported in host mode.												R/W													
b16	L1RSMOK	L1 Status Bit	L1 Status Bit 0: The current state cannot be recovered from L1 1: The current state can be restored from L1												R													
b15	SLPSTS	sleep status bit	sleep status bit device mode When the device sends ACK in response to LPM transmission and the time specified by the protocol has passed, this bit is set to 1, indicating that it enters the SLEEP mode. When the bus state changes, or disconnects, or sends a remote wake-up signal, this bit is automatically cleared to 0 and exits SLEEP mode. Host mode When the host sends an LPM transmission and receives an ACK response from the device, this bit is set to 1, indicating that it enters the SLEEP mode. When the host receives a remote wake-up signal, or the host initiates a wake-up, or the host initiates a reset, this bit is automatically cleared to 0 to exit the SLEEP mode. 0: Not in L1 1: In L1												R													
b14~b13	LPMRSP[1:0]	LPM response	LPM response The LPM response received by the master or sent by the slave 00b: ERROR (no response) 01b: STALL 00b: NYET 00b: ACK												R													

b12	L1DSEN	L1 deep sleep enable bit	L1 deep sleep enable bit To minimize power consumption, the application needs to set this bit to 1.	R
b11~b8	BSELTHRS[3:0]	BSEL Threshold Setting Register	BSEL Threshold Setting Register device mode When the device receives that the BSEL value is greater than or equal to the setting of this register, it enters L1. Host mode The time when the host sends the resume signal.	R/W
B7	L1SEN	L1 shallow sleep enable bit	L1 shallow sleep enable bit To minimize power consumption, the application needs to set this bit to 1.	R/W
b6	REMWAKE	bRemoteWake value	bRemoteWake value bRemoteWake value sent by the host or received by the device BSEL register device mode The BSEL bmAttribute value is automatically updated after a valid response to LPM transmission. Host mode The host sends the BSEL value of the LPM transfer, or the length of the resume signal.	R/W
b5~b2	BSEL[3:0]	BSEL register	0000b: 125us 0001b: 150us 0010b: 200us 0100b: 400us 0101b: 500us 0110b: 1000us 0111b: 2000us 1000b: 3000us 1001b: 4000us 1010b: 5000us 1011b: 6000us 1100b: 7000us 1101b: 8000us 1110b: 9000us 1111b: 10000us	R/W
b1	LPMACK	LPM Reply Register	LPM Reply Register 1: ACK Even if this bit is written to 1, an ACK will not be sent when there is an error in the LPM transmission. PID/CRC error exists, ERROR response bLinkState is not equal to 0001b, STALL responds There is a pending data transfer, NYET acknowledges 0: NYET Only device mode works	R/W
b0	LPMEN	LPM enable bit	LPM enable bit 0: LPM disabled 1: LPM enable	R/W

35.7.2.13 USBFS host periodic transmit FIFO size register (USBFS_HPTXFSIZ)

USBFS Host periodic transmit FIFO size register

offset address: 0x100

Reset value: 0x02800500

This application can program the RAM size that must be allocated to the periodic TxFIFO.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PTXFD[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
PTXSA[15:0]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b16	PTXFD	Host periodic TxFIFO depth	Host periodic TxFIFO depth In 32-bit words. The minimum value is 16										R/W		
b15~b0	PTXSA	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 aperiodic TxFIFO depth.										R/W		

35.7.2.14 USBFS Device IN Endpoint Transmit FIFO Size Register (USBFS_DIEPTXF_x) (_x = 1..15)

USBFS device IN endpoint transmit FIFO size register

offset address: 0x104+(x-1)*0x4

Reset value: 0x02800500+(x-1)*0x280

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
INEPTXFD[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
INEPTXSA[15:0]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b16	INEPTXFD	IN endpoint TxFIFO depth	Device IN endpoint TxFIFO depth In 32-bit words. The minimum value is 16										R/W		
b15~b0	INEPTXSA	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.	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		

35.7.3 USBFS Host Mode Register

The host mode register affects the operation of the module in host mode. Host mode registers must not be accessed in device mode because the result is ambiguous.

Unless otherwise specified, bit values in register descriptions are represented in binary.

35.7.3.1 USBFS Host Configuration Register (USBFS_HCFG)

USBFS Host configuration register

offset address: 0x400

Reset value: 0x00000200

This register will configure 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														FSLS S	FSLSPCS[1:0]
Bit	Marking	Place name	Function										Read and write		
b31~b3	Reserved	-	The reset value must be maintained.										R/W		
b2	FSLSS	Only FS and LS are supported	The application uses this bit to control the enumeration speed of the module. Using this bit, an 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 supports 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: Keep When the module is in LS host mode 00: reserved 01: Select 48MHz PHY clock frequency 10: Select 6MHz PHY clock frequency 11: Reserved Note: When the device is connected to the host, the FSLSPCS must be set according to the speed of the connected device (after changing this bit, a software reset must be performed).										R/W		

35.7.3.2 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]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b26	Reserved	-	Reset value must be maintained										R/W		
b15~b0	FRIVL	frame interval	Frame interval 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. An 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). If the value is not programmed, the module will calculate it 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 initial configuration. Setting value = frame interval (ms) × (PHY clock frequency) -1 Note: The FRIVL bit can be modified whenever the application needs to change the frame interval time.										R/W		

35.7.3.3 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 time remaining in the current frame (in PHY clocks).

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
FTREM[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
FRNUM[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	FTREM	Frame remaining time	Frame time remaining Indicates the time remaining in the current frame (in PHY clocks). This field is decremented by 1 every 1 PHY clock that elapses. When the value reaches zero, this field will be reloaded with the value in the Interframe Interval register and a new SOF will be sent by the module over USB.	R											
b15~b0	FRNUM	frame number	Frame number The value of this field is incremented by 1 when a new SOF is sent over USB and cleared when 0x3FFF is reached.	R											

35.7.3.4 USBFS Host Periodic Transmit FIFO/Queue Status Register (USBFS_HPTXSTS)

USBFS Host periodic transmit FIFO/queue status register

offset address: 0x410

Reset value: 0x00080280

This read-only register contains free space information for the periodic TxFIFO and periodic transmit 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]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b24	PTXQTOP	Periodically send requests to the top of the queue	Periodically send requests to the top of the queue (Top of the periodic transmit request queue) Indicates the item currently being processed by the MAC in the periodic Tx request queue. This register is used for debugging. Bit 31: Odd/Even frame — 0: Send in even frames — 1: Send in odd-numbered frames Bits 30:27: Channel/Endpoint Number (Channel number) Bit 26:25: Type (Type) — 00: Input/Output — 01: zero-length packet — 11: Disable channel command Bit 24: Terminate (last entry for the selected channel)	R											
b23~b16	PTXQSAV	Periodically send request queue free space	Periodically send request queue free space (Periodic transmit request queue space available) Indicates the number of free slots in the periodic send request queue available for writing. The queue contains both IN and OUT requests. 0: The periodic send request queue is full 1:1 position available 2: 2 positions available n: n positions available (where, n ranges: 0~8) Other values: Keep	R											
b15~b0	PTXFSAVL	Periodically transmit data FIFO free space	Periodically transmit data FIFO free space (Periodic transmit data FIFO space available) Indicates the number of free locations available for writing to the periodic TxFIFO. in 32-bit words 0: Periodic TxFIFO is full 1:1 word available 2: 2 words available n: n words available (where, n ranges: 0~PTXFD) Other values: Keep	R											

35.7.3.5 USBFS Host All Channel Interrupt Register (USBFS_HINT)

USBFS Host all channels interrupt register

offset address: 0x414

Reset value: 0x0000 0000

When an event occurs on the channel, the host-wide channel interrupt register uses the host channel interrupt bit in the module interrupt register (HCINT bit in USBFS_GINTSTS) to interrupt the application. Each channel corresponds to 1 interrupt bit, with a maximum of 16 bits.

When the application clears an interrupt via 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
HAINT[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	The reset value must be maintained.	R/W											
b15~b0	HAINT	channel interrupt	Channel interrupt One bit for each channel: Channel 0 corresponds to bit 0, and channel 15 corresponds to bit 15.	R											

35.7.3.6 USBFS Host All Channel Interrupt Mask Register (USBFS_HINTMSK)

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 to interrupt the application when an event occurs on the channel.

Each channel corresponds to one interrupt mask bit, with a maximum of 16 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
HINTM[11:0]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b16	Reserved	-	Read as "0", write as "0"										R/W		
b15~b0	HINTM	channel interrupt mask	Channel interrupt mask 0: Mask interrupt 1: Enable interrupt One bit for each channel: Channel 0 corresponds to bit 0, and channel 15 corresponds to bit 15.										R/W		

35.7.3.7 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 master mode. Currently, the USBFS host only supports one port.

This register contains USB port related information such as USB reset, enable, suspend, resume, 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. 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]	Reserved	
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
Reserved	PWPR	PLSTS[1:0]	Reserved	PRST	PSUSP	PRES	Reserved	PENCHNG	PENA	PCDET	PCSTS				
<hr/>															
Bit	Marking	Place name	Function											Read and write	
b31~b19	Reserved	-	The reset value must be maintained.											R/W	
b18~b17	PSPD	port speed	Port speed Indicates the speed of the device connected to this port. 00/11: Reserved 01: Full Speed 10: low speed											R	
b16~b13	Reserved	-	The reset value must be maintained.											R/W	
b12	PWPR	port power	Port power Applications use this field to control power to this port. Since the built-in PHY of the USBFS does not have the power supply capability, when this is set to 1, the external USB power chip is enabled to supply power through 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/W	
b8	PRST	port reset	Port reset When an application sets this bit, it initiates a reset sequence on this port. The application program must time the reset cycle and clear this bit after the reset sequence is complete. 0: Port not in reset state 1: Port is in reset state The application must set this bit for a minimum of 10 ms to initiate a reset on the port.											R/W	

			Port suspend Applications set this bit to put this port in suspend mode. Only when this bit is set will the module stop sending SOF. To stop the PHY clock, the application must set the port clock stop bit, which enables the PHY's suspend input pin. The read value of this bit reflects the current pending state of the port. When a remote wake-up signal is detected, or the application program sets the port reset bit or port recovery bit in this register to 1, the module can clear this bit; or the application program sets the recovery/remote wake-up detection interrupt bit or interrupt in the module interrupt register. The open connection detection interrupt bit (respectively WKINT or DISCINT in USBFS_GINTSTS) is set to 1, and the module can also clear this bit. 0: The port is not in suspend mode 1: The port is in suspend mode	
B7	PSUSP	port hang		R/W
b6	PRES	port recovery	Port resume The application sets this bit to drive the resume signal on this port. The module will continue to drive the resume signal until the application program clears this bit. As indicated by the port resume/remote wake-up detection interrupt bit in the module's interrupt register (WKINT bit in USBFS_GINTSTS), if the module detects a USB remote wake-up sequence, it starts driving the resume signal without application intervention; if the module detects a disconnect If connected, clear this bit. The read value of this bit indicates whether the current module The recovery signal is being driven. 0: Do not drive the recovery signal 1: Drive recovery signal	R/W
b5~b4	Reserved	-	The reset value must be maintained.	R/W
b3	PENCHNG	Port enable/disable change	Port enable/disable change The module sets this bit to 1 when the state of port enable bit 2 in this register changes. This bit is cleared by software by writing a 1 to it.	R/W
b2	PENA	port enable	Port enable After the port has performed a reset sequence, it can only be enabled by the module and can be disabled by an overcurrent condition, a disconnect condition, or by clearing this bit by the application program. The application cannot set this bit by writing to the register. The port can only be disabled by clearing this bit. Operations on this bit will not trigger any interrupts to the application. 0: disable port 1: enable port	R/W
b1	PCDET	port connection detected	Port connect detected When a device connection is detected, the module sets this bit to trigger an interrupt to the application using the host port interrupt bit in the module's interrupt register (HPRTINT bit in USBFS_GINTSTS). The application must set this bit to clear this interrupt.	R/W
b0	PCSTS	port connection status	Port connect status 0: No device is connected to the port 1: The port is connected to a device	R

35.7.3.8 USBFS Host Channel x Characteristic Register (USBFS_HCCHARx) (x = 0..15)

USBFS Host channel-x characteristics register

offset address: 0x500 + (channel number × 0x20)

Reset value: 0x0000 0000

This register is used to set the host channel characteristics.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CHEN A	CHDI S	ODD FRM	DAD[6:0]				Reserved			EPTYP[1:0]		LSDE V	Reser ved		
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
EPDI R	EPNUM[3:0]				MPSIZ[10:0]										

Bit	Marking	Place name	Function	Read and write
b31	CHENA	channel enable	Channel enable This field is set by application software and cleared by USBFS host hardware. 0: Channel disabled 1: enable channel	R/W
b30	CHDIS	Channel forbidden	Channel disable The application sets this bit to stop sending/receiving data over the channel, even if the transmission over the channel has not completed, the stop operation still takes effect. The application must wait for the interrupt of the disabled channel to confirm that the channel has been disabled.	R/W
b29	ODDFRM	odd frame	Odd frame This field is set or reset by the application to indicate that the USBFS host must transmit odd or even frames, respectively. This field applies only to periodic (synchronized and interrupted) transactions. 0: even frame 1: Odd frame	R/W
b28~b22	DAD	Device address	Device address This field is used to specify a specific device to communicate with this host.	R/W
b21~b20	Reserved	-	The reset value must be maintained.	R/W
b19~b18	EPTYP	Endpoint type	Endpoint type Indicates the selected transfer type. 00: Control 01: Sync 10: Batch 11: Interrupt	R/W
b17	LSDEV	low speed equipment	Low-speed device This field is set 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/W
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 device endpoints communicating with this host channel.	R/W

35.7.3.9 USBFS Host Channel x Interrupt Register (USBFS_HCINTx) ($x = 0..15$)

USBFS Host channel-x interrupt register

offset address: 0x508 + (channel number × 0x20)

Reset value: 0x0000 0000

This register indicates the state of the channel when USB and AHB related events occur. When the host channel interrupt bit in the module interrupt register (HCINT bit in USBFS_GINTSTS) is set to 1, the application must read this register. Before performing a read operation on the register, the application must read the Host All Channels Interrupt (USBFS_HAIN) register to obtain the exact channel number of the Host Channel x Interrupt Register. The application must clear the corresponding bits in this register to 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				DTER R	FRM OR	BBER R	TXER R	Reser ved	ACK	NAK	STAL L	AHBE RR	CHH	XFRC	
<hr/>															
Bit	Marking	Place name	Function												Read and write
b31~b11	Reserved	-	The reset value must be maintained.												R/W
b10	DTERR	data switching error	Data toggle error The application needs to clear this bit by writing a 1.												R/W
b9	FRMOR	frame overflow error	Frame overrun error (Frame overrun) The application needs to clear this bit by writing a 1.												R/W
b8	BBERR	crosstalk error	Babble error A typical cause of a crosstalk event is that an endpoint sends a packet, but the packet length exceeds the endpoint's maximum packet length. The application needs to clear this bit by writing a 1.												R/W
B7	TXERR	communication transaction error	Transaction error Indicates that one of the following errors has occurred on the USB: CRC check failed time out Bit stuffing error wrong EOP The application needs to clear this bit by writing a 1.												R/W
b6	Reserved	-	The reset value must be maintained.												R/W
b5	ACK	Receive/send ACK response	ACK response received/transmitted interrupt The application needs to clear this bit by writing a 1.												R/W
b4	NAK	Received NAK response	NAK response received interrupt The application needs to clear this bit by writing a 1.												R/W
b3	STALL	Receive STALL response	Received STALL response (STALL response received interrupt												R/W
b2	AHBERR	AHB error	AHB error This error is only generated when in internal DMA mode and an AHB error occurs during an AHB read/write operation. The application can obtain the error address by reading the corresponding DMA channel address register. The application needs to clear this bit by writing a 1												R/W
b1	CHH	channel stop	Channel halted Abnormal end of transfer due to any USB transaction error or in response to an application inhibit request. The application needs to clear this bit by writing a 1.												R/W
b0	XFRC	Transmission completion	Transfer completed No errors occurred and the transfer completed normally. The application needs to clear this bit by writing a 1.												R/W

35.7.3.10 USBFS Host Channel x Interrupt Mask Register (USBFS_HCINTMSKx) (x = 0..15)

USBFS Host channel-x interrupt mask register

offset address: 0x50C + (channel number × 0x20)

Reset value: 0x0000 0000

This register is used to select masking of host channel interrupts.

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															
Bit	Marking	Place name	Function												
b31~b11	Reserved	-	The reset value must be maintained.												
b10	DTERRM	Data Switch Error Interrupt Mask	Data toggle error mask 0: Mask interrupt 1: Enable interrupt												
b9	FRMORM	Frame overflow error interrupt mask	Frame overrun mask 0: Mask interrupt 1: Enable interrupt												
b8	BBERRM	Crosstalk Error Interrupt Mask	Babble error mask 0: Mask interrupt 1: Enable interrupt												
B7	TXERRM	Communication Transaction Error Interrupt Mask	Transaction error mask 0: Mask interrupt 1: Enable interrupt												
b6	Reserved	-	The reset value must be maintained.												
b5	ACKM	Receive/Send ACK Response Interrupt Mask	ACK response receive/transmit interrupt mask (ACK response received/transmitted interrupt mask) 0: Mask interrupt 1: Enable interrupt												
b4	NAKM	Receive NAK response interrupt mask	NAK response received interrupt mask 0: Mask interrupt 1: Enable interrupt												
b3	STALLM	Receive STALL response interrupt mask	STALL response received interrupt mask 0: Mask interrupt 1: Enable interrupt												
b2	AHBERRM	AHBERR interrupt mask	AHBERR interrupt mask (AHB error mask) 0: Mask interrupt 1: Enable interrupt												
b1	CHHM	Channel stop interrupt mask	Channel halted mask 0: Mask interrupt 1: Enable interrupt												
b0	XFRCM	Transport complete interrupt mask	Transfer completed mask 0: Mask interrupt 1: Enable interrupt												

35.7.3.11 USBFS Host Channel x Transfer Size Register (USBFS_HCTSIZx) (x = 0..15)

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 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]															

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	The reset value must be maintained.	R/W
b30~b29	DPID	data PID	Data PID (Data PID) The application sets the initial sync PID for data communication in this field. The host retains the setting of this field during this transfer transaction. 00: DATA0 01: reserved 10: DATA1 11: SETUP	R/W
b28~b19	PKTCNT	packet count	Packet count The application sets the number of packets to be sent or received in this field. The host decrements the count each time a packet is successfully sent or received. After this value reaches 0, the application is interrupted to indicate that the operation completed normally.	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 the transfer. For IN operations, this field is the buffer size reserved by the application for transmission. For IN transactions (periodic and aperiodic), the application programs this field to an integer multiple of the maximum packet size.	R/W

35.7.3.12 USBFS Host Channel xDMA Address Register (USBFS_HCDMAx) ($x = 0..15$)

USBFS Host channel-x DMA address register

offset address: 0x514 + (channel number × 0x20)

Reset value: 0xFFFF XXXX

This register is used to set the DMA address in 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]															

Bit	Marking	Place name	Function	Read and write
b31~b0	DMAADDR	DMA address	DMA address This field stores the address of the memory for the DMA transfer used by the host to get data from or send data to the device endpoint. This register is incremented each time an AHB transfer ends.	R/W

35.7.4 USBFS Device Mode Register

The device mode registers affect the operation of the module in device mode. Device mode registers must not be accessed in host mode because the result is ambiguous.

Unless otherwise specified, bit values in register descriptions are represented in binary.

35.7.4.1 USBFS Device Configuration Register (USBFS_DCFG)

USBFS Device configuration register

offset address: 0x800

Reset value: 0x0820 0000

This register configures the module in device mode after power-up, some control command 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															
Reserved	PFIVL[1:0]							DAD[6:0]				Reser ved	NZLS OHSK	DSPD[1:0]	
Bit	Marking	Place name	Function	Read and write											
b31~b13	Reserved	-	The reset value must be maintained.	R/W											
b12~b11	PFIVL	Periodic frame interval	Periodic frame interval Indicates the point in time within a frame at which the application must be notified using periodic frame interrupts. This function can be used to determine if all isochronous communications for the frame are 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 parameter after executing each SetAddress control command.	R/W											
b3	Reserved	-	The reset value must be maintained.	R/W											
b2	NZLSOHSK	Non-zero length state OUT handshake signal	Non-zero length state OUT handshake signal (Non-zero-length status OUT handshake) During an OUT transaction in the control transfer status phase, an application can use this field to select the handshake signal to send when a non-zero length packet is received by the module. 1: When receiving a non-zero-length state OUT transaction, reply to the STALL handshake signal, and the received OUT data packet is not sent to the application. 0: Send the received OUT packet (zero length or non-zero length) to the application and reply to the handshake signal based on the NAK and STALL bits of the endpoint in the device endpoint control register.	R/W											
b1~b0	DSPD	device speed	Device speed Indicates the speed at which the application requires the module to enumerate, or the maximum speed supported by the application. However, the actual bus speed can only be determined after completing the chirp sequence, and this speed is based on the speed of the USB host connected to the module. 00: reserved 01: reserved 10: Reserved 11: Full speed (USB 1.1 transceiver clock is 48 MHz)	R/W											

35.7.4.2 USBFS Device Control Register (USBFS_DCTL)

USBFS Device control register

offset address: 0x804

Reset value: 0x0000 0002

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				POPR GDN E	CGO NAK	SGO NAK	CGIN AK	SGIN AK	Reserved			GON STS	GINS TS	SDIS	RWAS IG																
<hr/>																															
Bit	Marking	Place name	Function										Read and write																		
b31~b12	Reserved	-	The reset value must be maintained.										R/W																		
b11	POPRGDNE	Power-on programming completed	Power-on programming done The application uses this bit to indicate that the registers have been programmed after a wake-up from Power-down mode.										R/W																		
b10	CGONAK	Clear global OUT NAK	Clear global OUT NAK(Clear global OUT NAK) A write to this bit clears the global OUT NAK.										R/W																		
b9	SGONAK	Set global OUT NAK	Set global OUT NAK (Set global OUT NAK) A write to this bit will set the global OUT NAK. Applications use this bit to send NAK handshake signals on all OUT endpoints. The application should only set this bit if it is sure that the global OUT NAK valid bit in the module interrupt register (GONAKEFF bit in USBFS_GINTSTS) is cleared.										R/W																		
b8	CGINAK	Clear global IN NAK	Clear global IN NAK (Clear global IN NAK) A write to this bit clears the global IN NAK.										R/W																		
B7	SGINAK	Set global IN NAK	Set global IN NAK A write to this field sets the global aperiodic IN NAK. The application uses this bit to cause all aperiodic IN endpoints to send the NAK handshake signal. The application should only set this bit if it is sure that the global IN NAK valid bit in the module interrupt register (GINAKEFF bit in USBFS_GINTSTS) is cleared.										R/W																		
b6~b4	Reserved	-	The reset value must be maintained.										R/W																		
b3	GONSTS	Global OUT NAK status	Global OUT NAK status 0: Handshaking will be sent based on FIFO status and NAK and STALL bit settings. 1: No data is received regardless of whether there is free space in the RxFIFO. Except for SETUP transactions, the NAK handshake signal is replied to all received packets. All isochronous type OUT packets will be discarded.										R																		
b2	GINSTS	Global IN NAK status	Global IN NAK status 0: The handshake signal will be replied based on the data availability in the transmit FIFO. 1: Causes all aperiodic IN endpoints to reply to the NAK handshake signal, regardless of data availability in the transmit FIFO.										R																		
b1	SDIS	soft disconnect	Soft disconnect The application uses this bit to signal the USBFS module to perform a soft disconnect. When this bit is set, the host will not see that the device is connected, and the device will not receive signals on the USB. The module remains disconnected until the application clears this bit. 0: Normal operation. Clearing this bit after a soft disconnect will cause the host to receive a device connected event. After reconnecting the device, the USB host restarts the device enumeration. 1: Causes the host to receive a device disconnect event.										R/W																		
<hr/>																															
At full speed, the minimum soft disconnect time is specified as follows: Suspended state: The minimum time is 1ms+2.5us idle state: 2.5us Non-idle or suspended state: 2.5us																															

b0	RWASIG	Send remote wakeup signal	Send Remote wakeup signaling When the application sets this bit to 1, the module will initiate a remote wake-up signal to wake up the USB host. Applications must set this bit to bring the module out of the suspended state. According to the USB 2.0 specification, the application must clear this bit within 1 ms to 15 ms after setting it.	R/W
----	--------	---------------------------	---	-----

35.7.4.3 USBFS Device Status Register (USBFS_DSTS)

USBFS Device status register

offset address: 0x808

Reset value: 0x0000 0002

This register indicates the state of the module when a USB related event occurs. When an interrupt occurs, the endpoint information on which the interrupt occurred must be read from the device-wide interrupt (USBFS_DAINT) register.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
Reserved								LNSTS[1:0]	FNSOF[13:8]							
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0	
FNSOF[7:0]								Reserved				EERR	ENUMSPD[1:0]	SUSP STS		

Bit	Marking	Place name	Function	Read and write
b31~b24	Reserved	-	The reset value must be maintained.	R/W
b23~b22	LNSTS	USB bus status	USB bus status LNSTS[1]: Logic level of D+ LNSTS[0]: The logic level of D-	R
b21~b8	FNSOF	Frame number of received SOF	Frame number of the received SOF	R
b7~b4	Reserved	-	The reset value must be maintained.	R/W
b3	EERR	indefinite error	Erratic error The module sets this bit to report any indeterminate errors. Due to an indeterminate error, the USBFS controller goes into a suspend state and generates an interrupt on the early suspend bit in the USBFS_GINTSTS register (ESUSP bit in USBFS_GINTSTS). If the early pending interrupt was triggered by an indeterminate error, the application can only perform a soft disconnect to resume communication.	R
b2~b1	ENUMSPD	Enumeration speed	Enumerated speed Indicates the speed to be enumerated by the USBFS controller after detecting the speed through the chirp sequence. 01: reserved 10: Reserved 11: Full speed (PHY clock running at 48 MHz) Other values: Keep	R
b0	SUSPSTS	Suspended state	Suspend status In device mode, this bit is set whenever a suspend state is detected on the USB. When the idle state on the USB bus remains for 3ms, the module will enter the suspend state. The module exits the suspended state when: — There is activity on the USB cable — The application performs a write operation to the Remote Wakeup Signal bit (RWUSIG bit in USBFS_DCTL) of the USBFS_DCTL register.	R

35.7.4.4 USBFS device IN endpoint general 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 of all endpoints to generate interrupts on each IN endpoint. The IN endpoint interrupt in the USBFS_DIEPINTx registers can be masked by writing to the appropriate bits 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	NAK M	Reserved					INEP NEM	INEP NMM	TTXF EMSK	TOM	Reser ved	EPDM	XFRC M		
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b14	Reserved	-	The reset value must be maintained.										R/W		
b13	NAKM	NAK interrupt mask	NAK interrupt mask (NAK mask) 0: Mask interrupt 1: Enable interrupt										R/W		
b12~b7	Reserved	-	The reset value must be maintained.										R/W		
b6	INEPNEM	IN endpoint NAK valid interrupt mask	IN endpoint NAK effective mask 0: Mask interrupt 1: Enable interrupt										R/W		
b5	INEPNMM	Received IN token interrupt mask when EP does not match	Received 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	TTXFEMSK	IN token received interrupt mask when Tx FIFO is empty	IN token received interrupt mask when Tx FIFO is empty (IN token received when Tx FIFO empty mask) 0: Mask interrupt 1: Enable interrupt										R/W		
b3	TOM	Timeout interrupt mask (asynchronous endpoints)	(Timeout condition mask (Non-isochronous endpoints)) 0: Mask interrupt 1: Enable interrupt										R/W		
b2	Reserved	-	The reset value must be maintained.										R/W		
b1	EPDM	Endpoint Disable Interrupt Mask	Endpoint disabled interrupt mask 0: Mask interrupt 1: Enable interrupt										R/W		
b0	XFRM	Transport complete interrupt mask	Transfer completed interrupt mask 0: Mask interrupt 1: Enable interrupt										R/W		

35.7.4.5 USBFS device OUT endpoint general 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 of all endpoints to generate interrupts on each OUT endpoint. OUT endpoint interrupts in the USBFS_DOEPINTx registers can be masked by writing to the appropriate bits 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															
Bit	Marking	Place name	Function										Read and write		
b31~b5	Reserved	-	The reset value must be maintained.										R/W		
b4	OTEPDM	OUT token received interrupt mask when endpoint is disabled (OUT token received when endpoint disabled mask) Applies only to control OUT endpoints. 0: Mask interrupt 1: Enable interrupt	OUT token received interrupt mask when endpoint is disabled (OUT token received when endpoint disabled mask) Applies only to control OUT endpoints. 0: Mask interrupt 1: Enable interrupt										R/W		
b3	STUPM	SETUP phase completes interrupt masking	SETUP phase done mask Applies to control endpoints only. 0: Mask interrupt 1: Enable interrupt										R/W		
b2	Reserved	-	The reset value must be maintained.										R/W		
b1	EPDM	Endpoint Disable Interrupt Mask	Endpoint disabled interrupt mask 0: Mask interrupt 1: Enable interrupt										R/W		
b0	XFRCM	Transport complete interrupt mask	Transfer completed interrupt mask 0: Mask interrupt 1: Enable interrupt										R/W		

35.7.4.6 USBFS Device Entire 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 the endpoint, the USBFS_DAINT register will interrupt the application through the Device OUT endpoint interrupt bit or the Device IN endpoint interrupt bit in the USBFS_GINTSTS register (OEPINT or IEPINT bits in USBFS_GINTSTS, respectively). Each endpoint corresponds to an interrupt bit, and both the OUT endpoint and the IN endpoint have a maximum of 16 interrupt bits. Bidirectional endpoints will use the corresponding IN and OUT interrupt bits. When the application sets and clears bits in the corresponding device endpoint x interrupt registers (USBFS_DIEPINTx/USBFS_DOEPINTx), the corresponding bits in this register are also set and cleared.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
OEPINT[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
IEPINT[15:0]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b16	OEPINT	OUT endpoint interrupt bit	OUT endpoint interrupt bits One bit for each OUT endpoint: OUT endpoint 0 corresponds to bit 16 and OUT endpoint 15 corresponds to bit 31.										R/W		
b15~b0	IEPINT	IN endpoint interrupt bit	IN endpoint interrupt bits One bit for each IN endpoint: IN endpoint 0 corresponds to bit 0 and IN endpoint 15 corresponds to bit 15.										R/W		

35.7.4.7 USBFS device-wide endpoint interrupt mask register (USBFS_DAIINTMSK)

USBFS Device all endpoints interrupt mask register

offset address: 0x81C

Reset value: 0x0000 0000

The USBFS_DAIINTMSK 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_DAIINT register bit corresponding to this interrupt is still set.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
OEPINTM[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
IEPINTM[15:0]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b16	OEPINTM	OUT endpoint interrupt mask bit	OUT endpoint interrupt mask bits One bit for each OUT endpoint: OUT endpoint 0 corresponds to bit 16 and OUT endpoint 15 corresponds to bit 31. 0: Mask interrupt 1: Enable interrupt										R/W		
b15~b0	IEPINTM	IN endpoint interrupt mask bit	IN endpoint interrupt mask bits One bit for each IN endpoint: IN endpoint 0 corresponds to bit 0 and IN endpoint 15 corresponds to bit 15. 0: Mask interrupt 1: Enable interrupt										R/W		

35.7.4.8 USBFS device IN endpoint FIFO empty interrupt 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 empty 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
INEPTXFEM[5:0]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b16	Reserved	-	The reset value must be maintained.										R/W		
b15~b0	INEPTXFEM	IN EP TxFIFO empty interrupt mask bit	IN EP Tx FIFO empty interrupt mask bit (IN EP Tx FIFO empty interrupt mask bits) These bits are used as mask bits for USBFS_DIEPINTx. Each bit corresponds to a TXFE interrupt for an IN endpoint: IN endpoint 0 corresponds to bit 0 and IN endpoint 15 corresponds to bit 15 0: Mask interrupt 1: Enable interrupt										R/W		

35.7.4.9 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 transfer endpoint 0.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPEN A	EPDIS	Reserved	SNAK	CNAK		TXFNUM[3:0]		STAL L	Reser ved	EPTYP[1:0]	NAKS TS	Reser ved			
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
USBA EP								Reserved							MPSIZ[1:0]

Bit	Marking	Place name	Function	Read and write
b31	EPENA	Endpoint enable	Endpoint enable The application sets this bit to start sending data on endpoint 0. The module clears this bit before triggering any of the following interrupts on this endpoint: — SETUP phase completed - Endpoint barring —Transmission completion	R/W
b30	EPDIS	Endpoint forbidden	Endpoint disable Applications can set this bit to stop sending data on an endpoint even before the transfer on that endpoint is complete. An application must wait until an endpoint inhibit interrupt occurs before an endpoint can be considered inhibited. The module will clear this bit before the endpoint disable interrupt bit is set. An application can set this bit only after the endpoint's endpoint enable bit is set.	R/W
b29~b28	Reserved	-	The reset value must be maintained.	R/W
b27	SNAK	Set the NAK bit	Set NAK bit (Set NAK) A write to this bit sets the endpoint's NAK bit. With this bit, the application can control the sending of the NAK handshake signal on the endpoint. The module can also set this bit of the endpoint after the endpoint receives the SETUP packet.	R/W
b26	CNAK	Clear the NAK bit	Clear NAK bit (Clear NAK) A write to this bit clears the endpoint's NAK bit.	R/W
b25~b22	TXFNUM	TxFIFO number	TxFIFO number The 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, when the endpoint receives the SETUP token, the module will clear this bit. If the NAK bit, the global IN NAK, or the global OUT NAK and this bit are all set, the STALL bit takes precedence.	R/W
b20	Reserved	-	The reset value must be maintained.	R/W
b19~b18	EPTYP	Endpoint type	Endpoint type Hardware is set to '00', indicating a control-type endpoint.	R
b17	NAKSTS	NAK status	NAK status Indicates the following results: 0: The module replies to a non-NAK handshake according to the FIFO status. 1: The module replies with a NAK handshake on this endpoint. When this bit is set (either by the application or by the module), the module will stop sending data even if there is still data available in the TxFIFO. Regardless of how this bit is set, the module always responds to the SETUP packet with an ACK handshake.	R
b16	Reserved	-	The reset value must be maintained.	R/W
b15	USBAEP	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/W

b1~b0	MPSIZ	maximum packet size	Maximum packet size Applications must program this field to the maximum packet size of the current logical endpoint. 00: 64 bytes 01: 32 bytes 10: 16 bytes 11: 8 bytes	R/W
-------	-------	---------------------	--	-----

35.7.4.10 USBFS device IN endpoint x control register (USBFS_DIEPCTLx) (x=1..15)

USBFS Device IN endpoint x control register

offset address: 0x900 + (endpoint number × 0x20)

Reset value: 0x0000 0000

Applications use this register to control the behavior of individual logical endpoints (except endpoint 0).

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPEN A	EPDI S	SOD DFR M	SD0P ID/ SEVN FRM	SNAK	CNAK	TXFNUM[3:0]				STAL L	Reser ved	EPTYP[1:0]	NAKS TS	ENOUM/ DPID	
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
USBA EP	Reserved			MPSIZ[10:0]											

Bit	Marking	Place name	Function	Read and write
b31	EPENA	Endpoint enable	Endpoint enable The application sets this bit to start sending data on the endpoint. The module clears this bit before triggering any of the following interrupts on this endpoint: - Endpoint barring —Transmission completion	R/W
b30	EPDIS	Endpoint forbidden	Endpoint disable Applications can set this bit to stop sending data on an endpoint even before the transfer on that endpoint is complete. An application must wait until an endpoint inhibit interrupt occurs before an endpoint can be considered inhibited. The module will clear this bit before the endpoint disable interrupt bit is set. An application can set this bit only after the endpoint's endpoint enable bit is set.	R/W
b29	SODDFRM	set odd frames	Set odd frame Applies only to synchronous IN and OUT endpoints. A write to this field sets the Even/Odd Frame (EONUM) field to odd frame.	R/W
b28	SD0PID/ SEVNFRM	set DATA0 PID/ SEVNFRM	Set DATA0 PID (Set DATA0 PID) Applies to interrupt/bulk IN endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. SEVNFRM: Set even frame Applies only to synchronous IN endpoints. A write to this field will set the Even/Odd Frame (EONUM) field to an even frame.	R/W
b27	SNAK	Set the NAK bit	Set NAK bit (Set NAK) A write to this bit sets the endpoint's NAK bit. With this bit, the application can control the sending of the NAK handshake signal on the endpoint. The module can also set this bit of the OUT endpoint to 1 when a transfer completion interrupt occurs or after a SETUP is received on the endpoint	R/W
b26	CNAK	Clear the NAK bit	Clear NAK bit (Clear NAK) A write to this bit clears the endpoint's NAK bit.	R/W
b25~b22	TXFNUM	TxFIFO number	TxFIFO number These bits specify the FIFO number associated with this endpoint. A separate FIFO number must be set for each valid IN endpoint. This field is only valid for IN endpoints.	R/W
b21	STALL	STALL handshake	STALL handshake The application sets this bit to cause the device to reply STALL for all tokens from the USB host. If the NAK bit, global IN NAK, or global OUT NAK is set 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/W

			Endpoint type The following are the transport types supported by this logical endpoint. 00: Control 01: Sync 10: Batch 11: Interrupt	
b19~b18	EPTYP	Endpoint type	NAK status Indicates the following results: 0: The module replies to a non-NAK handshake according to the FIFO status. 1: The module replies with a NAK handshake on this endpoint. When an application or module sets this bit: For asynchronous IN endpoints: The module stops sending any data through the IN endpoint even if there is data available in the TxFIFO. For synchronous IN endpoints: The module sends a zero-length packet even if there is data available in the TxFIFO. Regardless of how this bit is set, the module always responds to the SETUP packet with an ACK handshake.	R
b17	NAKSTS	NAK status	Even/odd frame Applies only to 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 numbers through the SEVNFRM and SODDFRM fields in this register for this endpoint to send/receive isochronous data. 0: even frame 1: Odd frame	R
b16	EONUM/ DPID	even/odd frames/ Endpoint data PID	DPID: Endpoint data PID Applies to interrupt/bulk IN endpoints only. Contains the PID of the packet that will be received or sent on this endpoint. After an endpoint is activated, the application must program the PID of the first packet to be received or sent on this endpoint. The application program uses the SD0PID register field to program the DATA0 or DATA1 PID. 0: DATA0 1: DATA1	R
b15	USBAEP	USB Active Endpoint	USB active endpoint Indicates whether this endpoint is active in the current configuration and interface. The module clears this bit for all endpoints except endpoint 0 when a USB reset is detected. After receiving the SetConfiguration and SetInterface commands, the application must program the endpoint registers accordingly and set this bit.	R/W
b14~b11	Reserved	-	The reset value must be maintained.	R/W
b10~b0	MPSIZ	maximum packet size	Maximum packet size Applications must program this field to the maximum packet size of the current logical endpoint. This value is in bytes.	R/W

35.7.4.11 USBFS Device IN Endpoint x Interrupt Register (USBFS_DIEPINTx) (x=0..15)

USBFS Device IN endpoint x interrupt register

offset address: 0x908 + (endpoint number × 0x20)

Reset value: 0x0000 0080

This register indicates the state of the endpoint in the presence of USB and AHB related events. The application must read this register when the IN endpoint interrupt bit in the module interrupt register (the IEPINT bit in USBFS_GINTSTS) is set. Before an application can read this register, the device-wide 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 to 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	Reser ved	TTXF E	TOC	Reser ved	EPDI SD	XFRC
Bit	Marking	Place name	Function										Read and write		
b31~b8	Reserved	-	The reset value must be maintained.										R/W		
B7	TXFE	Transmit FIFO is empty	Transmit FIFO empty This interrupt is asserted when the TxFIFO of this endpoint is half empty or completely empty. Whether the TxFIFO is half empty or completely empty is determined by the TxFIFO empty bit in the USBFS_GAHBCFG register (TXFELVL bit in USBFS_GAHBCFG).										R		
b6	INEPNE	IN endpoint NAK valid	INEPNE: IN endpoint NAK effective This bit can be cleared when the application clears the IN endpoint NAK by writing to the CNAK bit in USBFS_DIEPCTLx. This interrupt indicates that the module has sampled a NAK set (by the application or the module) and the result has taken effect. This interrupt indicates that the IN endpoint NAK bit, set by the application, is active in the module. This interrupt does not guarantee that the NAK handshake signal was sent on the USB. The STALL bit has higher priority than the NAK bit. Software can also write 1 to clear this bit.										R/W		
b5	Reserved	-	The reset value must be maintained.										R/W		
b4	TTXFE	IN token received when TxFIFO is empty	IN token received when TxFIFO is empty (IN token received when TxFIFO is empty) Applies to aperiodic IN endpoints only. When the TxFIFO (periodic/aperiodic) corresponding to this endpoint is empty, an IN token is received and an interrupt is generated. Cleared by software writing 1.										R/W		
b3	TOC	time out	Timeout condition Applies only to control IN endpoints. Indicates that this endpoint timed out the response to the most recently received IN token. Cleared by software writing 1.										R/W		
b2	Reserved	-	The reset value must be maintained.										R/W		
b1	EPDISD	Endpoint Disable Interrupts	Endpoint disabled interrupt This bit indicates that the endpoint has been disabled by the application. Cleared by software writing 1.										R/W		

b0	XFRC	transfer complete interrupt	Transfer completed interrupt This field indicates that the transfer set on this endpoint has completed on USB and AHB. Cleared by software writing 1.	R/W
----	------	-----------------------------	---	-----

35.7.4.12 USBFS device IN endpoint 0 transfer size register (USBFS_DIEPTSIZ0)

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. After enabling endpoint 0 through the endpoint enable bit (EPENA in USBFS_DIEPCTL0) in the device control endpoint 0 control register, the module modifies this register. The application can read this register only after the module clears the endpoint enable bit.

Non-zero endpoints use the registers of endpoints 1~15.

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]							
<hr/>															
Bit	Marking	Place name	Function								Read and write				
b31~b21	Reserved	-	The reset value must be maintained.								R/W				
b20~b19	PKTCNT	packet count	Packet count Indicates the number of data packets contained in a data transfer of endpoint 0. This field is decremented each time a packet (maximum size or short packet) is read from the TxFIFO.								R/W				
b18~b7	Reserved	-	The reset value must be maintained.								R/W				
b6~b0	XFRSIZ	transfer size	Transfer size Indicates the amount of data contained in a data transfer for endpoint 0, in bytes. The module only interrupts the application when it has finished transferring this data. The transfer size can be set to the endpoint's maximum packet size to break 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				

35.7.4.13 USBFS device IN endpoint x transfer size register (USBFS_DIEPTSIzX) (x=1..15)

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. This register is modified by the module after the endpoint is enabled through the endpoint enable bit in the USBFS_DIEPCTLx register (EPENA bit in USBFS_DIEPCTLx). The application can read this register only after the module clears the endpoint enable bit.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Res	MCNT[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]															

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	The reset value must be maintained.	R/W
b30~b29	MCNT	multiple count	Multi count For periodic IN endpoints, this field indicates the number of packets that must be sent per frame on USB. The module uses this field to calculate the data PID of the sync IN endpoint. 01: 1 packet 10: 2 packets 11: 3 packets	R/W
b28~b19	PKTCNT	packet count	Packet count Indicates the number of packets contained in a data transfer on this endpoint. This field is 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 only interrupts the application when it has finished transferring this data. The transfer size can be set to the endpoint's maximum packet size to break 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

35.7.4.14 USBFS device IN endpoint x DMA address register (USBFS_DIEPDMAx) (x=0..15)

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 of the device endpoint 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]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b0	DMAADDR	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 storage area pointed to by this field is also used to store control OUT packets and SETUP transaction packets. When more than three SETUP packets are received consecutively, the SETUP packets in the memory will be overwritten. This register is incremented with each AHB transfer. The application must set a doubleword-aligned address.	R/W											

35.7.4.15 USBFS device IN endpoint transmit FIFO status register (USBFS_DTXFSTSx) (x=0..15)

USBFS Device IN endpoint transmit FIFO status register

offset address: 0x918 + (endpoint number × 0x20)

Reset value: 0x0000 0280

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]															
Bit	Marking	Place name	Function										Read and write		
b31~b16	Reserved	-	The reset value must be maintained.										R/W		
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		

35.7.4.16 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 transfer endpoint 0.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPEN A	EPDI S	Reserved		SNAK	CNAK		Reserved		STAL L	SNPM	EPTYP[1:0]	NAKS TS	Reser ved		
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
USBA EP								Reserved							MPSIZ[1:0]

Bit	Marking	Place name	Function	Read and write
b31	EPENA	Endpoint enable	Endpoint enable The application sets this bit to start data reception on endpoint 0. The module clears this bit before triggering any of the following interrupts on this endpoint: — SETUP phase completed - Endpoint barring —Transmission completion	R/W
b30	EPDIS	Endpoint forbidden	Endpoint disable The application cannot disable control of OUT endpoint 0.	R
b29~b28	Reserved	-	The reset value must be maintained.	R/W
b27	SNAK	Set the NAK bit	Set NAK bit (Set NAK) A write to this bit sets the endpoint's NAK bit. With this bit, the application can control the sending of the NAK handshake signal on the endpoint. The module can also set this bit of the endpoint after the endpoint receives the SETUP packet.	R/W
b26	CNAK	Clear the NAK bit	Clear NAK bit (Clear NAK) A write to this bit clears the endpoint's NAK bit.	R/W
b25~b22	Reserved	-	The reset value must be maintained.	R/W
b21	STALL	STALL handshake	STALL handshake When this endpoint receives a SETUP token, the application can only set this bit and the module will clear it. If the NAK bit, global OUT NAK and this bit are set at the same time, the STALL bit takes precedence. Regardless of how this bit is set, the module always responds to the SETUP packet with an ACK handshake.	R/W
b20	SNPM	monitor mode	Snoop mode This bit is used to configure the endpoint to listen mode. In listen mode, the module does not check for correct OUT packets before transferring them to the application store.	R/W
b19~b18	EPTYP	Endpoint type	Endpoint type Hardware is set to '00', indicating a control-type endpoint.	R/W
b17	NAKSTS	NAK status	NAK status Indicates the following results: 0: The module replies to a non-NAK handshake according to the FIFO status. 1: The module replies with a NAK handshake on this endpoint. When an application or module sets this bit to 1, the module stops receiving data even if there is room in the RxFIFO to continue to accommodate received packets. Regardless of how this bit is set, the module always responds to the SETUP packet with an ACK handshake.	R
b16	Reserved	-	The reset value must be maintained.	R/W
b15	USBAEP	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/W

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
-------	-------	---------------------	--	-----

35.7.4.17 USBFS device OUT endpoint x control register (USBFS_DOEPCTLx) (x=1..15)

USBFS Device OUT endpoint x control register

offset address: 0xB00 + (endpoint number × 0x20)

Reset value: 0x0000 0000

Applications use this register to control the behavior of individual logical endpoints (except endpoint 0).

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
EPEN A	EPDI S	SOD DFRM / SD1P ID	SD0P ID/ SEVN FRM	SNAK	CNAK		Reserved		STAL L	SNPM	EPTYP[1:0]	NAKS TS	EONU M/ DPID		
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
USBA EP		Reserved									MPSIZ[10:0]				

Bit	Marking	Place name	Function	Read and write
b31	EPENA	Endpoint enable	Endpoint enable Set by software, USBFS is cleared 0: Endpoint disable 1: Endpoint enable	R/W
b30	EPDIS	Endpoint forbidden	Endpoint disable Applications can set this bit to stop sending/receiving data on an endpoint even before the transfer on that endpoint is complete. An application must wait until an endpoint inhibit interrupt occurs before an endpoint can be considered inhibited. The module will clear this bit before the endpoint disable interrupt bit is set. An application can set this bit only after the endpoint's endpoint enable bit is set.	R/W
b29	SD1PID/ SODDFRM	set DATA1 PID/set odd frame	Set DATA1 PID (Set DATA1 PID) Applies to interrupt/bulk OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. SODDFRM: Set odd frame Applies only to synchronous OUT endpoints. A write to this field sets the Even/Odd Frame (EONUM) field to odd frame.	R/W
b28	SD0PID/ SEVNFRM	set DATA0 PID/ SEVNFRM	Set DATA0 PID (Set DATA0 PID) Applies to interrupt/bulk OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. SEVNFRM: Set even frame Applies only to synchronous OUT endpoints. A write to this field will set the Even/Odd Frame (EONUM) field to an even frame.	R/W
b27	SNAK	Set the NAK bit	Set NAK bit (Set NAK) A write to this bit sets the endpoint's NAK bit. With this bit, the application can control the sending of the NAK handshake signal on the endpoint. The module can also set this bit on the OUT endpoint when a transfer complete interrupt occurs or after a SETUP is received on the endpoint.	R/W
b26	CNAK	Clear the NAK bit	Clear NAK bit (Clear NAK) A write to this bit clears the endpoint's NAK bit.	R/W
b25~b22	Reserved	-	The reset value must be maintained.	R/W
b21	STALL	STALL handshake	STALL handshake When this endpoint receives a SETUP token, the application can only set this bit and the module will clear it. If the NAK bit, global OUT NAK and this bit are set at the same time, the STALL bit takes precedence. Only the application can clear this bit, not the module. Snoop mode	R/W
b20	SNPM	monitor mode	This bit is used to configure the endpoint to listen mode. In monitor mode, the module will no longer check the correctness of the received data.	R/W

			Endpoint type The following are the transport types supported by this logical endpoint. 00: Control 01: Sync 10: Batch 11: Interrupt	
b19~b18	EPTYP	Endpoint type	NAK status Indicates the following results: 0: The module replies to a non-NAK handshake according to the FIFO status. 1: The module replies with a NAK handshake on this endpoint. When an application or module sets this bit: The module stops receiving any data on the OUT endpoint even if there is room in the RxFIFO for incoming packets. Regardless of how this bit is set, the module always responds to the SETUP packet with an ACK handshake.	R/W
b17	NAKSTS	NAK status	Even/odd frame Applies only to 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 numbers through the SEVNFRM and SODDFRM fields in this register for this endpoint to transmit/receive synchronously data. 0: even frame 1: Odd frame	R
b16	EONUM/ DPID	even/odd frames/ Endpoint data PID	DPID: Endpoint data PID Applies to interrupt/bulk OUT endpoints only. Contains the PID of the packet that will be received or sent on this endpoint. After an endpoint is activated, the application must program the PID of the first packet to be received or sent on this endpoint. The application program uses the SD0PID register field to program the DATA0 or DATA1 PID. 0: DATA0 1: DATA1	R
b15	USBAEP	USB Active Endpoint	USB active endpoint Indicates whether this endpoint is active in the current configuration and interface. The module clears this bit for all endpoints except endpoint 0 when a USB reset is detected. After receiving the SetConfiguration and SetInterface commands, the application must program the endpoint registers accordingly and set this bit.	R/W
b14~b11	Reserved	-	The reset value must be maintained.	R/W
b10~b0	MPSIZ	maximum packet size	Maximum packet size Applications must program this field to the maximum packet size of the current logical endpoint. This value is in bytes.	R/W

35.7.4.18 USBFS device OUT endpoint x interrupt register (USBFS_DOEPINTx) (x=0..15)

USBFS Device OUT endpoint x interrupt register

offset address: 0xb08 + (endpoint number × 0x20)

Reset value: 0x0000 0080

This register indicates the state of the endpoint in the presence of USB and AHB related events. The application must read this register when the OUT endpoint interrupt bit in the USBFS_GINTSTS register (OEPINT bit in USBFS_GINTSTS) is set. Before an application can read this register, the USBFS_DAINT register must be read to obtain the exact endpoint number of the USBFS_DOEPINTx registers. The application must clear the corresponding bits in this register to 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															
Bit	Marking	Place name	Function	Read and write											
b31~b7	Reserved	-	The reset value must be maintained.	R/W											
b6	B2BSTUP	Received consecutive SETUP packets	Back-to-back SETUP packets received Applies only to control OUT endpoints. This bit indicates that the endpoint has received more than three consecutive SETUP packets. Software can also write 1 to clear this bit.	R/W											
b5	Reserved	-	The reset value must be maintained.	R/W											
b4	OTEPDIS	OUT token received when endpoint is disabled	OUT token received when endpoint disabled Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled, thereby generating an interrupt. Cleared by software writing 1.	R/W											
b3	STUP	SETUP phase completed	SETUP phase done Applies only to control OUT endpoints. Indicates that the SETUP phase of the control endpoint is complete and that no consecutive SETUP packets are being received in the current control transfer. On this interrupt, the application can decode the received SETUP packet. Cleared by software writing 1.	R/W											
b2	Reserved	-	The reset value must be maintained.	R/W											
b1	EPDISD	Endpoint Disable Interrupts	Endpoint disabled interrupt This bit indicates that the endpoint has been disabled by the application. Cleared by software writing 1.	R/W											
b0	XFRC	transfer complete interrupt	Transfer completed interrupt This field indicates that the transfer set on this endpoint has completed on USB and AHB. Cleared by software writing 1.	R/W											

35.7.4.19 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. After enabling endpoint 0 through the endpoint enable bit (EPENA in USBFS_DIEPCTL0) in the device control endpoint 0 control register, the module modifies this register. The application can read this register only after the module clears the endpoint enable bit.

Non-zero endpoints use the registers of endpoints 1~15.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved	STUPCNT[1:0]		Reserved								PKTCNT	Reserved			
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
Reserved								XFRSIZ[6:0]							
<hr/>															
Bit	Marking	Place name	Function								Read and write				
b31	Reserved	-	The reset value must be maintained.								R/W				
b30~b29	STUPCNT	SETUP packet count	SETUP packet count This field specifies the number of SETUP packets that the endpoint can continuously receive. 01: 1 packet 10: 2 packets 11: 3 packets								R/W				
b28~b20	Reserved	-	The reset value must be maintained.								R/W				
b19	PKTCNT	packet count	Packet count The number of packets that should be received in one transmission. Before the endpoint is enabled, the software sets this bit, and after the transmission starts, the value of this field is automatically decremented every time a data packet is received.								R/W				
b18~b7	Reserved	-	The reset value must be maintained.								R/W				
b6~b0	XFRSIZ	transfer size	Transfer size Indicates the amount of data contained in a data transfer for endpoint 0, in bytes. The module only interrupts the application when it has finished transferring this data. The transfer size can be set to the endpoint's maximum packet size to break at the end of each packet. The module decrements this field each time a packet is read from the RxFIFO and written to external memory.								R/W				

35.7.4.20 USBFS device OUT endpoint x transfer size register (USBFS_DOEPTSIZx) (x=1..15)

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. The module modifies this register after enabling the endpoint through the endpoint enable bit in the USBFS_DOEPCTLx register (EPENA bit in USBFS_DOEPCTLx). The application can read this register only after the module clears the endpoint enable bit.

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Res	RXDPID[1:0]/S TUPCNT[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]															
Bit	Marking	Place name		Function										Read and write	
b31	Reserved	-		The reset value must be maintained.										R/W	
b30~b29	RXDPID/ STUPCNT	Received data PID/SETUP packet count		Received data PID (Received data PID) Applies only to synchronous OUT endpoints. This is the PID of the last packet received by this endpoint. 00: DATA0 01: DATA2 10: DATA1 11: MDATA STUPCNT: SETUP packet count Applies only to control OUT endpoints. This field specifies the number of SETUP packets that the endpoint can continuously receive. 01: 1 packet 10: 2 packets 11: 3 packets										R/W	
b28~b19	PKTCNT	packet count		Packet count Indicates the number of packets contained in a data transfer on this endpoint. This field is decremented after each write packet (maximum size or short packet) to the 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. The module only interrupts the application when it has finished transferring this data. The transfer size can be set to the endpoint's maximum packet size to break at the end of each packet. The module decrements this field each time a packet is read from the RxFIFO and written to external memory.										R/W	

35.7.4.21 USBFS device OUT endpoint x DMA address register (USBFS_DOEPDMAx) (x=0..15)

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 of the device endpoint 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]															
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31~b0	DMAADDR	DMA address	DMA address This bit contains the starting address of the external memory area when using DMA for data transfers on the endpoint. Note: For control endpoints, the storage area pointed to by this field is also used to store control OUT packets and SETUP transaction packets. When more than three SETUP packets are received consecutively, the SETUP packets in the memory will be overwritten. This register is incremented with each AHB transfer. The application must set a doubleword-aligned address.										R/W		

35.7.5 USBFS Clock Gating Control Register

The HCLK and PHY clocks are controlled by gated clock control registers to reduce power consumption. Unless otherwise specified, bit values in register descriptions are represented in binary.

35.7.5.1 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															
Bit	Marking	Place name	Function										Read and write		
b31~b8	Reserved	-	The reset value must be maintained.										R/W		
B7	SUSP	Deep Sleep status bits	Deep Sleep (L1 suspended) status bit										R		
b6	PHYSLEEP	PHY SLEEP Mode Status Bits	PHY SLEEP Mode Status Bit (PHY in sleep) This bit is used to indicate that the PHY is in the SLEEP mode state.										R		
b5	ENL1GTG	L1 clock gating	L1 Clock Gating (Enable Sleep Clock Gating) This bit is used for clock gating control in L1 mode.										R/W		
b4~b2	Reserved	-	The reset value must be maintained.										R/W		
b1	GATEHCLK	Gated HCLK	Gate HCLK (Gate HCLK) The application sets this bit to 1 to stop clocking modules other than the AHB bus slave interface, master interface, and wake-up logic when USB communication is suspended or the session is invalid. The application clears this bit when the USB resumes communication or a new session starts.										R/W		
b0	STPPCLK	stop PHY clock	Stop PHY clock (Stop PHY clock) The application sets this bit to stop the PH clock when USB communication is suspended, the session is invalid, or the device is disconnected. The application clears this bit when USB resumes communication.										R/W		

36 CAN FD Controller (CAN FD)

36.1 Introduction

In products with CAN FD function, the CAN FD controller corresponds to CAN controller channel 2 (CAN_2). For details, please refer to the model function comparison table.

CAN (Controller Area Network) bus is a bus standard that can realize mutual communication between microprocessors or devices without a host.

CAN FD controller follows CAN bus CAN2.0 (CAN2.0A, CAN2.0B) and CAN FD protocol.

The CAN bus controller can handle data transmission and reception on the bus. In this product, the CAN FD controller has 16 acceptance filters. Filters are used to select messages for an application to receive.

The application program can transmit data through one high-priority primary transmit buffer (PTB) and 3 secondary transmit buffers (STB), the transmission order of transmit buffers is determined by the transmission scheduler. There are 8 receive buffers (RB) for data reception. 3 STBs and 8 RBs can be understood as a 3-level FIFO and an 8-level FIFO, and the FIFO is completely controlled by hardware.

CAN FD controller can also support time-triggered CAN communication.

Main features of CAN FD:

- Fully supports CAN2.0A/CAN2.0B/CAN FD protocol.
- CAN FD supports the highest communication baud rate of 8Mbit/s
- Supports 1~1/256 baud rate prescaler, flexible baud rate configuration.
- 8 receive buffers
 - FIFO mode
 - Incorrect or unreceived data does not overwrite stored messages
- 1 high priority primary transmit buffer PTB
- 3 secondary transmit buffers STB
 - FIFO mode
 - Priority decision mode
- 16 independent filters
 - Supports 11-bit standard ID and 29-bit extended ID
 - Programmable ID CODE bit and MASK bit
- Both PTB/STB support single shot transmission mode
- Supports listen only mode
- Supports loopback mode
- Supports capturing of last occurred kind of error and arbitration lost position

- Programmable error warning limit
- Supports ISO11898-4 Time-Triggered CAN with partial hardware

36.2CAN FD system block diagram

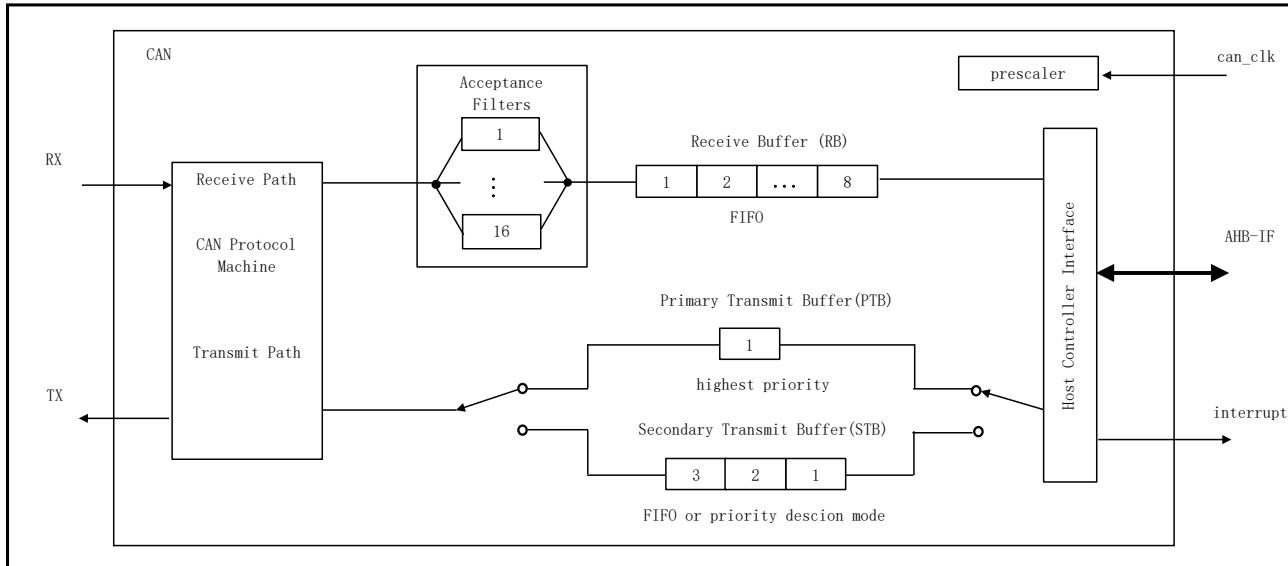


Figure 36-1 CANFD system block

36.3Pin description

Table 36-1 CAN pin description

Pin name	Direction	Functional description
CAN2_RX	Input	CAN receive data signal
CAN2_TX	Output	CAN transmit data signal
CAN2_TST_SAMPLE	Output	For observation only, shows the used sample points (activated one period after the sample point)
CAN2_TST_CLOCK	Output	For observation only, shows the used bit time periods (activated one period before the bit time starts)

36.4 Function description

36.4.1 Operation mode

CAN FD controller has two operation modes, reset mode (CAN_CFG_STAT.RESET=1) and action mode (CAN_CFG_STAT.RESET=0). When the module is initialized, the registers that can only be operated in the reset mode should first be set (see the register description chapter for details), and then exit the reset mode, operate the remaining registers in the action mode.

36.4.2 Baud rate setting

CAN communication clock uses can_clk. Before using the CAN module, first set the CAN communication clock according to CMU chapter. The clock selection must satisfy the setting condition that CAN control logic clock(PCLK1) is 1.5 times or more than CAN communication clock(can_clk).

The following figure shows the definition of CAN bit time. The part on the dotted line is the bit time specified by the CAN protocol, and the part below the dotted line is the bit time defined by CAN_CTRL. Among them, segment1 and segment2 can be set through registers SBT and FBT. The SBT register and the FBT register can only be set when CAN_CFG_STAT.RESET=1. The SBT register is used for the arbitration field of CAN2.0 and CAN FD, and the FBT register is used for the CAN FD data field.

For FD communication, it is recommended to select 20MHz/40MHz/80MHz for the communication clock.

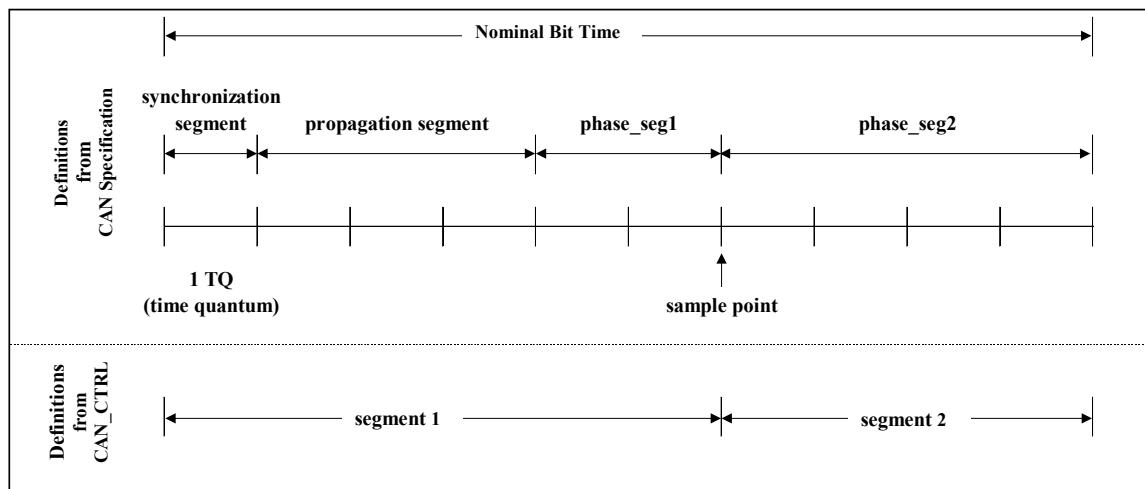


Figure 36-2 CAN Bit Time Definition

Refer to the following formula for the TQ calculation, PRESC is set by SBT.S_PRESC or FBT.F_PRESC. f_{can_clk} is the CAN communication clock frequency.

$$S_TQ = \frac{S_PRESC+1}{f_{can_clk}}$$

$$F_{TQ} = \frac{F_{PRESC+1}}{f_{can_clk}}$$

Refer to the following formula for the bit time calculation, S_segment1 and S_segment2 are set by SBT.S_SEG_1 and SBT.S_SEG_2, and F_segment1 and F_segment2 are set by the FBT.F_SEG_1 and FBT.F_SEG_2.

$$\text{Slow_BT} = t_{S_{\text{segment1}}} + t_{S_{\text{segment2}}} = ((S_{\text{SEG_1}}+2) + (S_{\text{SEG_2}}+1)) \times S_{\text{TQ}}$$

$$\text{Fast_BT} = t_{F_{\text{segment1}}} + t_{F_{\text{segment2}}} = ((F_{\text{SEG_1}}+2) + (F_{\text{SEG_2}}+1)) \times F_{\text{TQ}}$$

Table 36-2 CAN bit time setting rules

Register	Parameter range	Rule
SBT.S_SEG_1	[0..63] CAN2.0 bits (slow) [0..63] CAN FD nominal bits (slow)	
SBT.S_SEG_2	[0..7] CAN2.0 bits (slow) [0..31] CAN FD nominal bits (slow)	SEG_1 ≥ SEG_2 + 1 SEG_2 ≥ SJW
SBT.S_SJW	[0..15] CAN2.0 bits (slow) [0..15] CAN FD nominal bits (slow)	
FBT.F_SEG_1	[0..15] CAN FD data bits (fast)	
FBT.F_SEG_2	[0..7] CAN FD data bits (fast)	SEG_1 ≥ SEG_2 SEG_2 ≥ SJW
FBT.F_SJW	[0..7] CAN FD data bits (fast)	

The baud rate setting recommendation of CANFD is given below for reference only.

PSP: Primary Sample Point

SSP: Secondary Sample Point

Seg 1: Segment 1

Seg 2: Segment 2

TDC: Transmitter Delay Compensation

Table 36-3 Recommendations for 20MHz can_clk

Bit Rate [Mbit/s]	PSP [%]	SSP [%]	Prescaler	Bit Time [TQ]	Seg 1 [TQ]	Seg 2 [TQ]	SJW [TQ]	TDC [can_clk]
0.25 Arbitration	80	-	1	80	64	16	16	-
0.5 Arbitration	80	-	1	40	32	8	8	-
0.5	80	(disable)	1	40	32	8	8	-
1	80	80	1	20	16	4	4	16
2	80	80	1	10	8	2	2	8
4	80	80	1	5	4	1	1	4
5	75	75	1	4	3	1	1	3

Table 36-4 Recommendations for 40MHz can_clk

Bit Rate [Mbit/s]	PSP [%]	SSP [%]	Prescaler	Bit Time [TQ]	Seg 1 [TQ]	Seg 2 [TQ]	SJW [TQ]	TDC [can_clk]
0.25 Arbitration	80	-	2	80	64	16	16	-
0.5 Arbitration	80	-	1	80	64	16	16	-
0.5	80	(disable)	2	40	32	8	8	-
1	80	80	1	40	32	8	8	32
2	80	80	1	20	16	4	4	16
4	80	80	1	10	8	2	2	8
5	75	75	1	8	6	2	2	6
8	80	80	1	5	4	1	1	4

Table 36-5 Recommendations for 80MHz can_clk

Bit Rate [Mbit/s]	PSP [%]	SSP [%]	Prescaler	Bit Time [TQ]	Seg 1 [TQ]	Seg 2 [TQ]	SJW [TQ]	TDC [can_clk]
0.25 Arbitration	80	-	4	80	64	16	16	-
0.5 Arbitration	80	-	2	80	64	16	16	-
0.5	80	(disable)	4	40	32	8	8	-
1	80	80	2	40	32	8	8	64
2	80	80	2	20	16	4	4	32
4	80	80	1	20	16	4	4	16
5	75	75	1	16	12	4	4	12
8	80	80	1	10	8	2	2	8

36.4.3 Transmit buffer (TB)

CAN_CTRL provides two transmit buffers for transmitting data, the primary transmit buffer PTB and the secondary transmit buffer STB. PTB has the highest priority but can only buffer one frame. The priority of STB is lower than PTB, but it can buffer 3 frames, and the 3 frames in STB can act in FIFO mode or priority decision mode.

The 3 frames in the STB can be all transmitted by setting TCMD.TSALL to 1. In FIFO mode, the frame filled first is transmitted first, and in priority decision mode, the frame with the smallest ID is transmitted first.

The frame in the PTB has the highest priority, so the PTB transmission can postpone the STB transmission, but the STB that has won arbitration and started transmission cannot be postponed by the PTB transmission.

PTB and STB can be accessed through the TBUF register. Select PTB or STB through TCMD.TBSEL, TBSEL=0, select PTB, TBSEL=1, select STB. The next SLOT in the STB is selected by TCTRL.TSNEXT. The corresponding relationship is shown in the following figure:

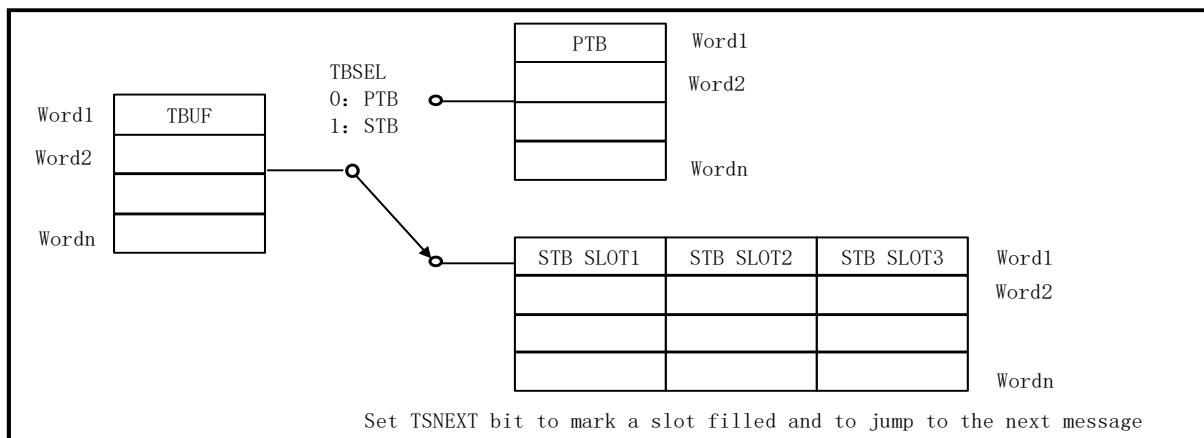


Figure 36-3 Schematic of CANFD TBUF register write transmit buffer

36.4.4 Receive buffer (RB)

CAN_CTRL provides 8 receive buffer slots which act in FIFO mode for storing received data. RB slot reads the received data through the RBUF register, always reads the oldest received data first, and releases the RB slot that has been read by setting RCTRL.RREL to 1, and points to the next RB slot.

The schematic diagram of reading RB slot by RBUF is as follows.

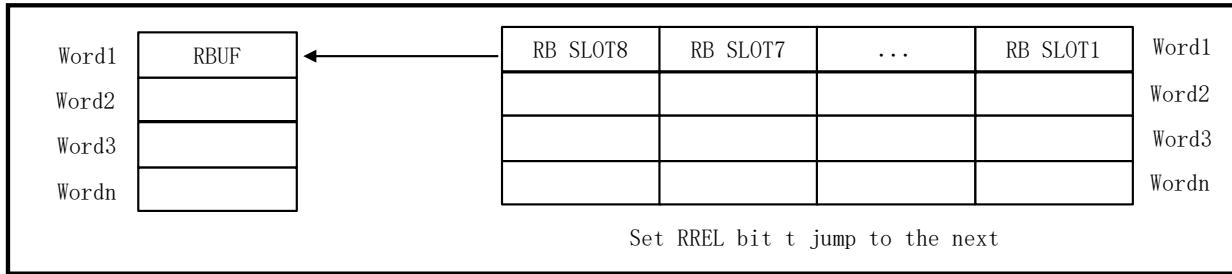


Figure 36-4 Schematic of CANFD RBUF register read receive buffer

36.4.5 Acceptance filters

CAN_CTRL provides 16 independent 32-bit filters for filtering received data to reduce CPU load. The filters can support standard format 11-bit ID or extended format 29-bit ID. Each filter has a 32-bit ID CODE register and a 32-bit ID MASK register, the ID CODE register is used to compare the received CAN ID, and the ID MASK register is used to select the CAN ID bits for comparison. When the corresponding ID MASK bit is 1, the ID CODE of this bit is not compared.

The received data will be received as long as it passes any one of the 16 filters, and the received data will be stored in the RB, otherwise the data will not be received or stored.

Each filter is enabled or disabled through the ACFEN register. ID CODE and ID MASK are set by ACFCTRL.SELMASK. When SELMASK=0, it points to ID CODE, and when SELMASK=1, it points to ID MASK. The filter is selected by ACFCTRL.ACFFADR. ID CODE and ID MASK are accessed through ACF register and can only be set when CFG_STAT.RESET=1. Refer to the following figure for the way of ACF register access to filter register group.

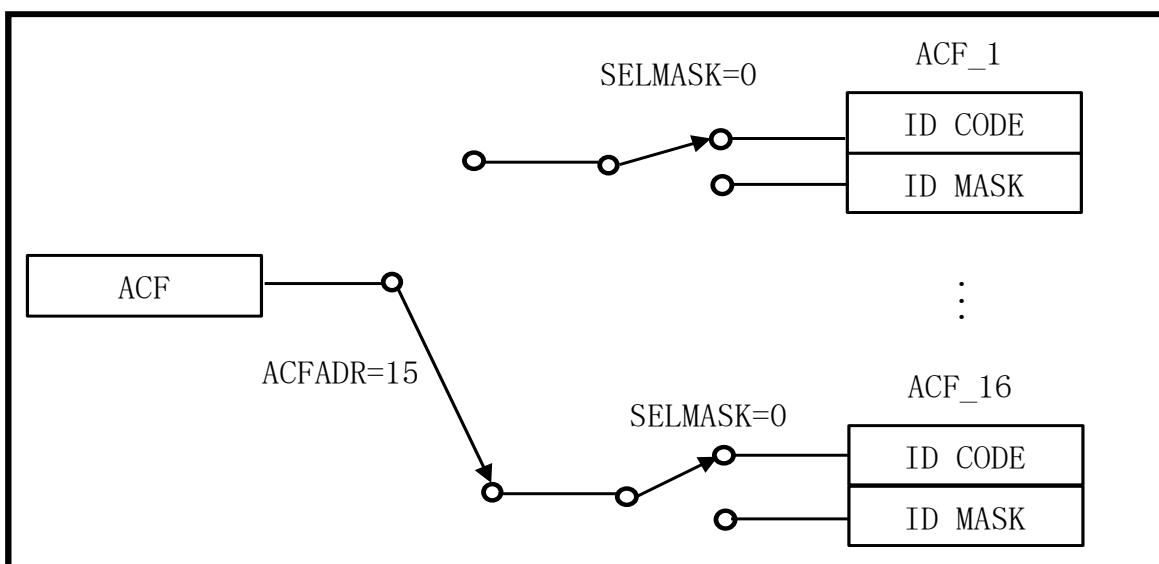


Figure 36-5 Access to the CANFD Acceptance Filters

36.4.6 Data transmission

Before transmitting, ensure that the data to be transmitted by the PTB or STB has been filled, and then start the PTB or STB transmission. Refill data is not allowed during transmitting.

The steps of transmitting data setting are as follows:

1. Set TBSEL to select TB from PTB and STB
2. Write the data to be transmitted through the TBUF register.
3. If STB is selected, set TSNEXT=1 to complete the loading of all STB SLOTS.
4. Transmission enable
 - PTB transmission using TPE
 - STB transmission using TSALL or TSONE
5. Transmission completion status confirmation
 - For the PTB, TPIF is set if TPIE is enabled
 - For the STB using TSONE, TSIF is set if one message has been completed and TSIE is enabled
 - For the STB using TSALL, TSIF is set if all messages have been completed and if TSIE is enabled

36.4.7 Single shot transmission

When the automatic retransmission function is not required, it can be set to the single shot transmission mode through the register. PTB single shot transmission is set by CFG_STAT.TPSS and STB is set by CFG_STAT.TSSS. When the data is successfully transmitted, the operation of single shot transmission is the same as the normal transmission. But the following results appear when the data is not transmitted successfully:

- When TPIF is set (if TPIE=1), the corresponding transmit buffer slot will be cleared.
- When there is an error occurred during the transmission, KOER is updated and BEIF is set (if BEIE=1).
- Arbitration is lost and ALIF is set (if ALIE=1).

In single shot transmission mode, TPIF cannot be relied on alone to judge whether the transmission is completed. It is necessary to judge whether the transmission is completed together with BEIF and ALIF.

36.4.8 Transmission abort

Data transmissions that have been requested but not yet executed can be aborted by TPA or TSA. Transmission abort will be executed in the following situations:

- There is no abort during bus arbitration
 - If the node loses arbitration, the abort will be executed afterwards.
 - If the node wins arbitration, the frame will be transmitted.

- There is no abort while a frame is transmitted
 - If a frame is transmitted successfully and ACK is received, the corresponding flag and status are normally set. In this case no abort is signaled.
 - After an unsuccessful transmission where the CAN node does not receive an ACK, the error counter is incremented and the abort will be executed.
 - For the transmission requested by TSALL=1, the STB slot that is being transmitted is transmitted normally, and the STB slot that has not started to transmit will be aborted.

If an abort is executed, this results in the following actions.

- TPA releases the PTB which results in TPE=0.
- TSA releases one single slot or all slots of the STB, depends on whether TSONE or TSALL was used to start the transmission.

36.4.9 Data reception

Use the acceptance filters can filter out unnecessary receive data, reduce the occurrence of interrupts and the reading of RBs, thereby reducing the CPU load. The steps for receiving data setting are as follows:

1. Configures acceptance filters.
2. Set RFIE, RAFIE and AFWL.
3. Wait for RFIF or RAFIF.
4. Read the oldest received data from the RB FIFO via RBUF.
5. Set RREL=1 to select the next RB slot.
6. Repeat 4, 5 until RSTAT signals an empty RB.

36.4.10 Error handling

On one hand CAN_CTRL does automatic error handling. This includes automatic retransmission and automatic deletion of received messages with errors. On the other hand, CAN_CTRL may report errors to the CPU through interrupts.

CAN nodes have the following three error states:

- Error active: An active error flag is automatically transmitted when a node detects an error.
- Error passive: A passive error flag is automatically transmitted when the node detects an error.
- Bus Off: In Bus Off state, this node no longer affects the entire CAN network.

CAN_CTRL provides two counters, TECNT and RECNT, for counting errors. The TECNT and RECNT counters are incremented and decremented according to the rules specified by the CAN protocol. In addition, a programmable CAN error warning LIMIT register is provided for generating an error interrupt to notify the CPU.

There are the following 5 kinds of error in the CAN communication process, and the errors can be identified by EALCAP.KOER.

- Bit error

- Form error
- Stuff error
- ACK error
- CRC error

36.4.11 Bus Off

When the transmit error counter becomes > 255, the CAN node automatically enters the Bus Off state and does not participate in CAN communication until it returns to the error-active state. The Bus Off status can be confirmed by CFG_STAT.BUSOFF. The EIF interrupt is generated while BUSOFF is set.

There are two ways to recover from the Bus Off state to the error-active state:

- Power-on reset
- Received 128 sequences of 11-bit recessive bit(recovery sequence)

When the node isBus Off, the TECNT value remains unchanged and RECNT is used to count the recovery sequence. TECNT and RECNT are reset to 0 after recovery from the Bus Off state.

36.4.12 Arbitration lost capture

CAN_CTRL can accurately capture the position of the arbitration lost bit and reflect it in the ALC register. The ALC register holds the position of the latest arbitration lost bit. If the node wins the arbitration, the ALC bit is not updated.

The ALC value is defined as follows:

After the SOF bit, the first ID bit ALC is 0, the second ID bit ALC is 1, and so on. The maximum value of ALC is 31 because arbitration only occurs within the arbitration field. For example, a standard remote frame arbitrated with an extended frame, the extended frame lost arbitration at the IDE bit, then ALC=12.

36.4.13 Loopback mode

CAN_CTRL supports the following two loopback modes:

- Internal loopback
- External loopback

Both loopback modes can receive data frames sent by themselves, which are mainly used for self-tests.

In internal loopback mode, the module internally connects rxd to txd, and the txd output is set to recessive. In internal loopback mode, the node generates a self-ACK to avoid ACK error.

The external loopback mode keeps the connection with the transceiver, so the transmitted frame still appear on the CAN bus. With the help of the transceiver, the CAN-CTRL can receive its own

frame. In external loopback mode, whether to generate a self-ACK can be determined by RCTRL.SACK. When SACK=0, no self-ACK is generated, and when SACK=1, a self-ACK is generated.

In external loopback mode, when SACK=0, the following two results will be occurred:

- Another node receives the frame too and generates an ACK. This will result in a successful transmission and reception.
- If no other node generates an ACK, ACK error will be occurred, the frame will be retransmitted and the error counter will be incremented. In this case, it is recommended to use the single shot transmission mode.

When returning from loopback mode to normal mode, in addition to switching back to normal mode, software is also required to reset CAN_CTRL.

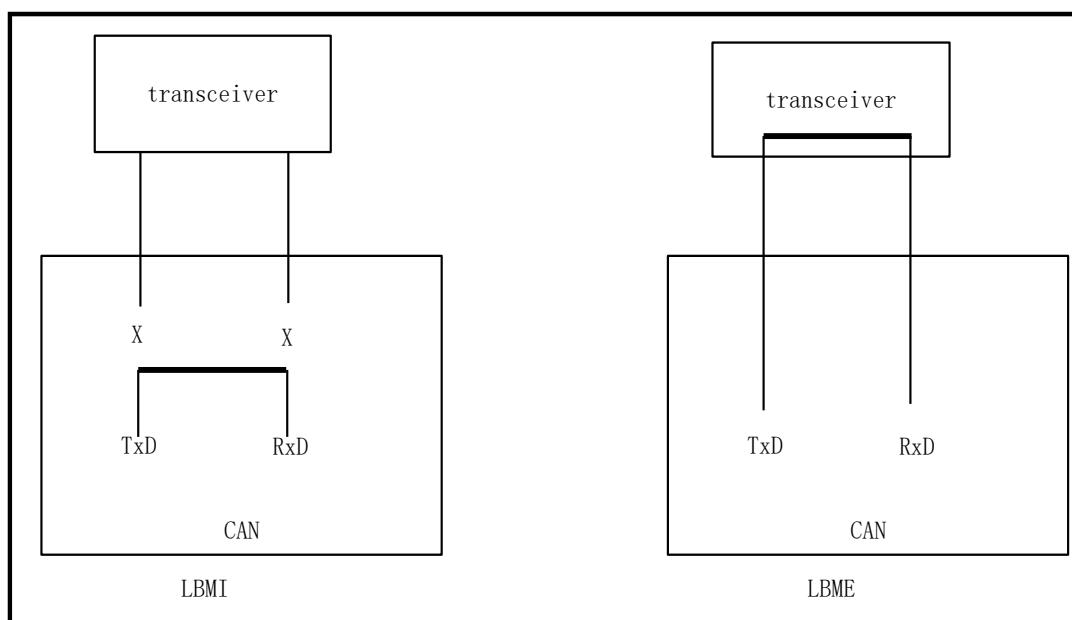


Figure 36-6 Schematic of CANFD LBMI and LBME

36.4.14 Listen only mode (LOM)

Listen only mode can be used to monitor CAN network. In this mode, data can be received from the CAN bus without transmitting any data to the bus. Set TCMD.LOM to 1 to put the CAN-CTRL into listen only mode, and clear it to 0 to leave listen only mode.

The external loopback mode can be combined with the listen only mode to form the external loopback listen only mode, at which time the CAN can be considered a silent receiver but can transmit when necessary. In external loopback listen only mode, frames containing self-ACK signals are allowed to be transmitted, but the node will not generate error or overload frames.

36.4.15 Software reset

By setting CFG_STAT.RESET to 1, the software reset function is realized. The range of the software reset function is shown in the following table.

Table 36-6 Software Reset

Register	Reset	Comment	Register	Reset	Comment
ACFADR	no	-	F_SEG_1	Yes	Writeable if RESET=1
ACODE	no	Writeable if RESET=1	F_SEG_2	Yes	Writeable if RESET=1
AE_x	no	-	F_SJW	Yes	Writeable if RESET=1
AFWL	no	-	KOER	Yes	-
AIF	Yes	-	LBME	Yes	-
ALC	Yes	-	LBMI	Yes	-
ALIE	no	-	RACTIVE	Yes	Reception is immediately cancelled even if a reception is active. No ACK will be generated
ALIF	Yes	-	RAFIE	no	-
AMASK	no	Writeable if RESET=1	RAFIF	Yes	-
BEIE	no	-	RBALL	Yes	-
BEIF	Yes	-	RBUF	Yes	All RB slots are marked as empty. RBUF contains unknown data
BUSOFF	no	Clear by writing 1	RECNT	no	An error counter reset is possible by setting BUSOFF=1
EIE_F	no	-	REF_ID	no	-
EIF	no	-	REF_IDE	no	-
EPASS	no	-	RFIE	no	-
EPIE	no	-	RFIF	Yes	-
EPIF	Yes	-	RIE	no	-
EWARN	no	-	RIF	Yes	-
EWL	Yes	-	ROIE	no	-
FD_ISO	no	Writeable if RESET=1	ROIF	Yes	-
F_FRESC	no	Writeable if RESET=1	ROM	no	
ROV	Yes	-	TSMODE	no	
RREL	Yes	-	TSNEXT	Yes	-
RSTAT	Yes		TSONE	Yes	-
SACK	Yes	-	TPIE	no	-
SELMASK	no	-	TPIF	Yes	-
S_PRESC	no	Writeable if RESET=1	TPSS	Yes	-
S_SEG_1	no	Writeable if RESET=1	TSFF	Yes	All STB slots are marked as empty
S_SEG_2	no	Writeable if RESET=1	TSIE	no	-
S_SJW	no	Writeable if RESET=1	TSIF	Yes	-
SSPOFF	Yes	-	TSSS	Yes	-

Register	Reset	Comment	Register	Reset	Comment
TACTIVE	Yes	All transmissions are immediately terminated with RESET	TSSTAT	Yes	All STB slots are marked as empty
TBE	Yes	-	TTEN	Yes Yes	-
TBF	no	-	TTIF	Yes	-
TBPTR	no	-	TTIE	no	-
TBSEL	Yes	-	TTPTR	no	-
TBUF	Yes	All STB slots are marked as empty. Because of TBSEL TBUF points to the PTB	TTTBM	no	-
TDCEN	Yes	-	TTYPE	no	-
TECNT	no	An error counter reset is possible by setting BUSOFF=1	TT_TRIG	no	-
TEIF	Yes	-	TT_WTRIG	no	-
TPA	Yes	-	T_PRESC	no	-
TPE	Yes	-	WTIE	no	-
TSA	Yes	-	WTIF	Yes	
TSALL	Yes	-			

36.4.16 Upward compatible with CAN-FD

When CAN-CTRL is disabled in CAN FD function, even if CAN FD frames are received in the network including CAN FD, the receiver will automatically ignore these frames, do not generate an ACK, wait for bus idle and may transmit or receive the next frame.

36.4.17 Time-triggered CAN(TTCAN)

CAN-CTRL provides partial (lever 1) hardware support for the time-triggered communication method specified by ISO11898-4. This chapter introduces the TTCAN function from the following 5 parts.

36.4.17.1 TBUF in TTCAN mode

TTTBM=1

When TTTBM=1, PTB and STB slot form a TB slot, and each slot can be addressed by TBPTR. When TBPTR=0, it points to PTB, and TBPTR=1 points to STB slot 1, and so on. The host can mark a slot as filled by setting TBF or as empty by setting TBE. Filled slots are write-locked. TBSEL and TSNEXT have no meaning in TTCAN mode and can be ignored.

When TTTBM=1, PTB has no special properties in this mode and is associated with the STB. This furthermore results in the fact that a successful transmission is always signaled using TSIF.

There is no FIFO operation and no priority decision for the TBUF in TTCAN mode. Furthermore, only one frame can be selected for transmission.

In TTCAN mode, all transmissions can only be started using a trigger, TPE, TSONE, TSALL, TPSS and TPA are fixed to 0 and ignored.

TTTBM=0

TTTBM=0 offers the combination of event-driven CAN communication and time-stamping for received messages. In this mode the PTB and the STB provide the same behavior as if TTEN=0. Then the PTB always has higher priority than the STB and the STB can operate in FIFO mode (TSMODE=0) or in priority mode (TSMODE=1).

36.4.17.2 TTCAN description

After power-up a time master needs to do the initialization as defined in ISO 11898-4. There may be up to 8 potential time masters in a CAN network. Each one has its own reference message ID (the last 3 bits of the ID). Potential time masters may try to transmit their reference message according to their priority. Lower-prioritized time masters shall try to transmit their reference messages later.

If TTEN is set, then the 16-bit timer is running. If a reference message is detected or if a time master successfully transmits its reference message, then CAN-CTRL copies the Sync_Mark of this message to the Ref_Mark which sets the cycle time to 0. The reception of the reference message will set RIF respectively the successful transmission will set TPIF or TSIF. Then the host needs to prepare the trigger for the next action.

A trigger for an action can be a reception trigger. This just triggers the interrupt, and it can be used to detect if an expected message is not received. Such a trigger can also be used for other actions, but this depends on the host application.

A different trigger is a transmission trigger. This starts the frame in the TBUF slot, where the trigger points to with TTPTR. If the TBUF slot is marked as empty, then no transmission is started, but the trigger interrupt is set.

36.4.17.3 TTCAN timing

CAN_CTRL supports ISO 11898-4 level 1. This includes a 16 bit timer running at the speed of a CAN bit time (defined by S_PRESC, S_SEG_1, S_SEG_2). An additional prescaler is defined by T_PRESC. If TTEN=1 then the timer is counting continuously.

At the SOF of the message, the timer value is the Sync_Mark. If the message was a reference message, then this value is copied to the Ref_Mark. The cycle time is the timer value minus the Ref_Mark. It is the time that is used as time-stamp for received messages or as trigger time for messages, that need to be transmitted.

36.4.17.4 TTCAN trigger types

The trigger type is defined by TTYPE. TTPTR is a pointer to a TB slot and TT_TRIG defines the cycle time of the trigger.

It includes the following five trigger types:

- immediate trigger
- time trigger
- single shot transmit trigger
- transmit start trigger
- transmit stop trigger

Except for the immediate trigger, all triggers set TTIF if TTIE is enabled. When TTTBM=1, only time triggers are supported.

Immediate Trigger

An immediate trigger starts the immediate transmission of a frame that is pointed to by TTPTR. TTIF is not set. To start a trigger, TT_TRIG_1 needs to be written. The value that is written, does not care for an immediate trigger.

Time Trigger

A time trigger just generates an event by setting TTIF and therefore generates an interrupt. No other actions are done. If a node expects to receive the expected data within a time window, the time trigger method can be used. If the TT_TRIG value is less than the actual cycle time, TEIF is set and no action is done.

Single Short Transmit Trigger

A single shot transmit trigger is intended to be used for exclusive time windows, where a message needs to be transmitted in single shot mode. In this case, the TSSS bit is ignored.

For this ISO 11898-4 defines a transmit enable window of up to 16 ticks of the cycle time. The register bits TEW(3:0)+1 define the number of ticks. If the data does not start to transmit within the specified transmit enable time window, the frame is discarded. As a result the TB slot of the frame will be marked as empty and AIF will be set if AIE is set. The data in the corresponding TB will not be rewritten, because it can be sent again by setting TPF.

If the TT_TRIG value is less than the actual cycle time, TEIF is set and no action is done.

Transmit Start Trigger

A transmit start trigger is intended to be used for merged arbitrating time windows, where several nodes may transmit messages and CAN arbitration takes place. The selected message is defined by TTPTR. TSSS defines if retransmission or single shot mode is used. If the selected frame cannot

be transmitted (arbitration loss, several transmissions after an error), then the transmission can be stopped using the transmit stop trigger.

If the TT_TRIG value is less than the actual cycle time, TEIF is set and no action is done.

Transmit Stop Trigger

A transmit stop trigger is intended to be used to stop a transmission, that is started with the transmit start trigger. If a transmission is stopped, then the frame is aborted. Therefore, AIF is set if AIE is set and the TB slot of the frame is marked as empty. The data in the corresponding TB will not be rewritten, because it can be sent again by setting TPF.

If TT_TRIG value is less than the actual cycle time, then TEIF is set but the stop is executed.

36.4.17.5 TTCAN Watch Trigger

The watch trigger can be used if TTTBM=1. The watch trigger is intended to be used if the time since the last valid reference message has been too long. Reference messages can be received in a periodic cycle or after an event. The host application needs to take care of this and has to adjust the watch trigger accordingly.

If the cycle count equal to the value defined by TT_WTRIG, then WTIF is set is WTIE is set.

If TT_WTRIG is less than the actual cycle time, then TEIF is set.

36.4.18 TDC and RDC

For communication with CAN FD data bit rate it may be that the delay of the transmitting transceiver or the delay of the bus is bigger than a bit time, so compensation is required. TDC (Transmitter Delay Compensation) and RDC (Receiver Delay Compensation) are used for data delay compensation. Among them, TDC requires software controlled, while RDC does not, and it is automatically effective. In TDC, the auxiliary sampling point SSP (Secondary Sample Point) is used to compensate the data delay. As shown below:

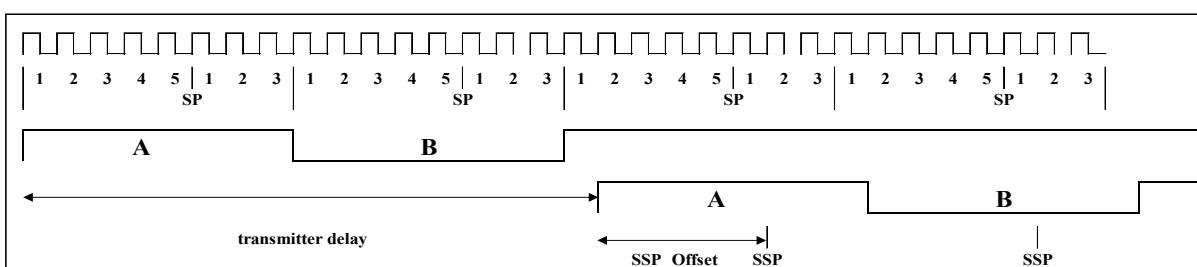


Figure 36-7 Transmitter Delay

When the software enables the TDC function, the controller can automatically determine the transmitter delay, and set the SSP offset by setting the register TDC.SSPOFF. It is suggested to set SSPOFF equal to $t_{F_segment1}$.

36.4.19 Interrupt

Table 36-7 CAN Interrupt flag

Channel	Interrupt flag	Description
CAN_2	RIF	Receive interrupt flag
	ROIF	Receive buffer overrun interrupt flag
	RFIF	Receive buffer full interrupt flag
	RAFIF	Receive buffer almost full interrupt flag
	TPIF	PTB transmission interrupt flag
	TSIF	STB transmission interrupt flag
	EIF	Error interrupt flag
	AIF	Abort interrupt flag
	EPIF	Error passive interrupt flag
	ALIF	Arbitration lost interrupt flag
	BEIF	Bus error interrupt flag
	WTIF	Watch trigger interrupt flag
	TEIF	Trigger error interrupt flag
	TTIF	Time trigger interrupt flag

36.5 Register description

CAN_2_BASE_ADDR: 0x40078000

Table 36-8 CAN register list

Register name	Symbol	Offset address	Bit width	Reset value
CAN Receive Buffer Register	CAN_RBUF	0x00~0x4F	-	0xFFFF XXXX
CAN Transmit Buffer Register	CAN_TBUF	0x50~0x97	-	0xFFFF XXXX
CAN Configuration and Status Register	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 Register	CAN_RTIE	0xA4	8	0xFE
CAN Receive and Transmit Interrupt Flag Register	CAN_RTIF	0xA5	8	0x00
CAN Error Interrupt Enable and Flag Register	CAN_ERRINT	0xA6	8	0x00
CAN Warning Limits Register	CAN_LIMIT	0xA7	8	0x1B
CAN Slow Bit Timing Register	CAN_SBT	0xA8	32	0x0102 0203
CAN Fast Bit Timing Register	CAN_FBT	0xAC	32	0x0102 0203
CAN Error and Arbitration Lost Capture Register	CAN_EALCAP	0xB0	8	0x00
CAN Transmitter Delay Compensation Register	CAN_TDC	0xB1	8	0x00
CAN Receive Error Counter Register	CAN_RECNT	0xB2	8	0x00
CAN Transmit Error Counter Register	CAN_TENCNT	0xB3	8	0x00
CAN Acceptance Filter Control Register	CAN_ACFCTRL	0xB4	8	0x00
CAN Acceptance Filter Enable Register	CAN_ACFEN	0xB6	8	0x01
CAN Acceptance CODE and MASK Register	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 Watch Trigger Time Register	CAN_TT_WTRIG	0xC8	16	0xFFFF

36.5.1 CAN Receive Buffer Register (CAN_RBUF)

Offset address: 0x00

Reset value: 0XXXX XXXX

The RBUF registers point the message slot with the oldest received message in the receive buffer.

All RBUF registers can be read in any order.

KOER in RBUF has the same meaning as the bits KOER in register EALCAP. KOER in RBUF becomes meaningful if RBALL=1.

The TX bit indicates that the message sent by itself is received in loopback mode.

The CYCLE_TIME bit is only valid in TTCAN mode and represents the cycle time at the SOF of this frame.

CAN_RBUF only supports WORD access.

The data format of the CAN receive buffer is as follows:

Table 36-9 Standard format of CAN receive buffer

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	FDF	BRS	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
...	...								Data
...	...								Data
...	...								Data
RBUF+70	DATA63								Data
RBUF+71	DATA64								Data

Table 36-10 Extended format of CAN receive buffer

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	FDF	BRS	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
...	...								Data
...	...								Data
...	...								Data
RBUF+70	DATA63								Data
RBUF+71	DATA64								Data

The meanings of the control bits are as follows:

IDE (IDentifier Extension):

0: Standard format

1: Extended format

RTR (Remote Transmission Request)

0: Data frame

1: Remote frame (only applicable to CAN2.0, fixed to 0 in CAN FD)

FDF (CAN FD Frame)

0: CAN2.0 frame

1: CAN FD frame

BRS (Bit Rate Switch)

0: The whole frame is a low-speed baud rate

1: Data and CRC are fast baud rate (only for CAN FD, fixed to 0 when FDF=0)

DLC (Data Length Code):

Data length code, CAN2.0 setting range is 0~8, corresponding data length is 0Byte~8Byte, CAN FD setting range is 0~15, corresponding data length is 0Byte~64Byte

Table 36-11 DLC control bits

DLC (binary)	Frame Type	Playload in Bytes
0000~1000	CAN2.0 and CAN FD	0~8
1001~1111	CAN2.0	8
1001	CAN FD	12
1010	CAN FD	16
1011	CAN FD	20
1100	CAN FD	24
1101	CAN FD	32
1110	CAN FD	48
1111	CAN FD	64

The meanings of the status bits are as follows:

KOER: Same as EALCAP.KOER

TX: This bit is set to 1 when receiving data sent by itself in loopback mode

36.5.2 CAN Transmit Buffer Register (CAN_TBUF)

Offset address: 0x50

Reset value: 0xFFFF XXXX

The TBUF registers point the next empty message slot in the STB if TBSEL=1 or to the PTB otherwise. All TBUF registers can be written in any order. For the STB it is necessary to set TSNEXT to mark a slot filled and to jump to the next message slot.

TBUF can only be accessed by WORD.

The data format of CAN transmit buffer is as follows:

Table 36-12 Standard format of CAN transmit buffer

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	FDF	BRS	DLC[3:0]				Control
TBUF+5	-								-
TBUF+6	-								-
TBUF+7	-								-
TBUF+8	DATA1								Data
TBUF+9	DATA2								Data
TBUF+10	DATA3								Data
...	...								Data
...	...								Data
...	...								Data
TBUF+70	DATA63								Data
TBUF+71	DATA64								Data

Table 36-13 Extended format of CAN transmit buffer

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	FDF	BRS	DLC[3:0]				Control
TBUF+5	-								-
TBUF+6	-								-
TBUF+7	-								-
TBUF+8	DATA1								Data
TBUF+9	DATA2								Data
TBUF+10	DATA3								Data
...	...								Data
...	...								Data
...	...								Data
TBUF+70	DATA63								Data
TBUF+71	DATA64								Data

For the meaning of the control bits, please refer to the description of the CAN Receive Buffer Register chapter.

36.5.3 CAN Configuration and Status Register (CAN_CFG_STAT)

Offset address: 0xA0

Reset value: 0x80

B7	b6	b5	b4	b3	b2	b1	b0
RESET	LBME	LBMI	TPSS	TSSS	RACTIVE	TACTIVE	BUSOFF

Bit	Symbol	Bit name	Description	Read and write
B7	RESET	RESET request	Reset request bit 0: Do not request a partial reset 1: Request a partial reset Some registers can only be written when RESET=1. For details, please refer to the software reset function. When the node enters the BUS OFF state, the hardware will automatically set the RESET bit to 1. Please note that it takes 11 CAN bit times for the node to participate in communication after RESET=0.	R/W
b6	LBME	eExternal Loopback Mode	External loopback mode enable bit 0: Disable external loopback mode 1: Enable external loopback mode Note: LBME should not be enabled while a transmission is active.	R/W
b5	LBMI	Internal Loopback Mode	Internal loopback mode enable bit 0: Disable internal loopback mode 1: Enable internal loopback mode Note: LBMI should not be enabled while a transmission is active.	R/W
b4	TPSS	Transmission Primary Single Shot	PTB single shot transmission mode 0: Disable PTB single shot transmission mode 1: Enable PTB single shot transmission mode	R/W
b3	TSSS	Transmission Secondary Single Shot	STB single shot transmission mode 0: Disable STB single shot transmission mode 1: Enable STB single shot transmission mode	R/W
b2	RACTIVE	Reception Active	Reception status 0: No receive activity 1: Receiving	R
b1	TACTIVE	Transmission Active	Transmission status 0: No transmission activity 1: transmitting	R
b0	BUSOFF	Bus Off state	Bus off state 0: Error active state 1: Bus off state Note: Writing a 1 clears the TENCNT and RECNT registers, but only for debugging purposes.	R/W

36.5.4 CAN Command Register (CAN_TCMD)

Offset address: 0xA1

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
TBSEL	LOM	-	TPE	TPA	TSONE	TSALL	TSA

Bit	Symbol	Bit name	Description	Read and write
B7	TBSEL	Transmit Buffer Select	<p>Transmit buffer select 0: PTB 1: STB When TTEN=1&TTTBM=1, TBSEL is reset to the reset value. Note: TBSEL needs to be stable all the time the TBUF registers are written and when TSNEXT is set.</p>	R/W
b6	LOM	Listen Only Mode	<p>Silent mode enable bit 0: Disable silent mode 1: Enable silent mode Transmission is prohibited when LOM=1&LBME=0. When LOM=1&LBME=1, it is forbidden to respond to the corresponding received frames and error frames, but the own frames can be transmitted. Note: LOM should not be enabled while a transmission is active.</p>	R/W
b5	Reserved	-	The reset value must be maintained.	R/W
b4	TPE	Transmit Primary Enable	<p>PTB transmission enable bit 0: No transmission for the PTB 1: Enable PTB transmission When this bit is enabled, the message in the PTB will be transmitted at the next available position. An STB transmission that has already started will be completed before, but the next pending STB transmission will be delayed until the PTB transmission has been completed. After this bit is written to 1, it will remain at 1 until the PTB transmission is completed or the transmission is aborted by TPA. Software cannot reset this bit to 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&TTTBM=1</p>	R/W

Bit	Marking	Place name	Function	Read and write
b3	TPA	Transmit Primary Abort	<p>Abort PTB transmission 0: No abort 1: Aborts the PTB transmission that has been requested by the TPE set but has not yet started</p> <p>This bit is written to 1 by software but cleared by hardware. Setting TPA automatically de-asserts TPE, it should not be set simultaneously with TPE. The TPE is reset to the reset value by hardware in the following cases:</p> <ul style="list-style-type: none"> - RESET=1 - BUOFF=1 - TTEN=1&TTBM=1 	R/W
b2	TSONE	Transmit Secondary One frame	<p>Transmit one frame of STB 0: No transmission for STB 1: Transmit one frame of STB</p> <p>In FIFO mode this is the oldest message and in priority mode this is the one with the highest priority.</p> <p>After this bit is written to 1, it will remain at 1 until the STB transmission is completed or the transmission is aborted by TSA. Software cannot reset this bit to 0.</p> <p>TSONE is reset to the reset value by hardware in the following cases:</p> <ul style="list-style-type: none"> - RESET=1 - BUOFF=1 - LOM=1&LBME=0 - TTEN=1&TTBM=1 	R/W
b1	TSALL	Transmit Secondary All frames	<p>Transmit all frames of STB 0: No transmission for STB 1: Transmit all frames of STB</p> <p>After this bit is written to 1, it will remain at 1 until the STB transmission is completed or the transmission is aborted by TSA. Software cannot reset this bit to 0.</p> <p>TSALL is reset to the reset value by hardware in the following cases:</p> <ul style="list-style-type: none"> - RESET=1 - BUOFF=1 - LOM=1&LBME=0 - TTEN=1&TTBM=1 	R/W
b0	TSA	Transmit Secondary Abort	<p>Abort STB transmission 0: No abort 1: Abort the STB transmission that has been requested by TSONE or TSALL set but has not yet started</p> <p>This bit is written to 1 by software but cleared by hardware. Setting TSA automatically de-asserts TSONE or TSALL respectively, it should not be set simultaneously with TSONE or TSALL.</p> <p>The TSA is reset by hardware to its reset value when:</p> <ul style="list-style-type: none"> - RESET=1 - BUOFF=1 	R/W

36.5.5 CAN Transmit Control Register (CAN_TCTRL)

Offset address: 0xA2

Reset value: 0x90

B7	b6	b5	b4	b3	b2	b1	b0
FD_ISO	TSNEXT	TSMODE	TTTBM	-	-	TSSTAT[1:0]	

Bit	Symbol	Bit name	Description	Read and write
B7	FD_ISO	CAN FD ISO mode	CAN FD ISO mode (can only be written when CAN_CFG_STAT.RESET=1) 0: Bosch CAN FD(non-ISO) mode 1:ISO 11898-1:2015 CAN FD Mode	R/W
b6	TSNEXT	Transmit buffer Secondary NEXT	Transmit next STB 0: No action 1: The current STB slot is filled and pointing to the next slot After all frame bytes are written to the TBUF registers, the host controller has to set TSNEXT to signal that this slot has been filled. Then the TBUF registers connects to the next slot. The slot is marked as filled a transmission can be started using TSONE or TSALL. TSNEXT has to be set by the host controller and is automatically reset by hardware immediately after it was set. After all STB slots are filled, TSNEXT remains at 1 until a STB slot is released. Note: This bit is fixed to 0 in TTCAN mode.	R/W
b5	TSMODE	Transmit buffer Secondary operation Mode	STB transmission mode 0: FIFO mode 1: Priority mode The FIFO mode is sent according to the order in which the data frames are written. The priority mode is automatically judged according to the ID. The lower 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 Transmit Buffer Mode	TTCAN transmit buffer mode When TTEN=0, TTTBM is ignored. 0: Separate PTB and STB, behavior defined by TSMODE 1: Buffer slots selectable by TBPTR and TTPTR In TTCAN mode, with only support of reception timestamps, TTTBM=0 can be chosen. Then the transmit buffer acts as in event-driven mode and the behavior can be selected by TSMODE. Note: The TTTBM bit can only be set when the TBUF is empty.	R/W
b3~b2	Reserved	-	The reset value must be maintained.	R/W
b1~b0	TSSTAT	Transmission Secondary Status	STB status TTEN=0 or TTEN=1 & TTTBM=0 00: STB empty 01: STB is less than or equal to half full 10: STB is more than half full 11: STB full TTEN=1 and TTTBM=1 00: PTB and STB empty 01: PTB and STB are not empty and not full 10: Reserved 11: PTB and STB are full	R

36.5.6 CAN Receive Control Register (CAN_RCTRL)

Offset address: 0xA3

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
SACK	ROM	ROV	RREL	RBALL	-		RSTAT[1:0]

Bit	Symbol	Bit name	Description	Read and write
B7	SACK	Self-Acknowledge	Self-Acknowledge 0: No self-ACK 1: self-ACK when LBME=1	R/W
b6	ROM	Receive buffer Overflow Mode	Receive buffer overflow mode 0: The oldest message will be overwritten 1: The new message will not be stored	R/W
b5	ROV	Receive buffer Overflow	Receive buffer overflow flag 0: No overflow 1: Overflow, at least one message is lost Cleared by writing RREL to 1.	R
b4	RREL	Receive buffer Release	Release the receive buffer 0: No release 1: Indicates that the receiving buffer has been read, and the RBUF register points to the next RB slot.	R/W
b3	RBALL	Receive Buffer stores All data frames	Receive buffer stores all data frames 0: Normal operation 1: Store all data including frames with errors.	R/W
b2	Reserved	-	The reset value must be maintained.	R/W
b1~b0	RSTAT	Receive buffer Status	Receive buffer status 00: RBUF empty 01: RBUF is not empty but less than AFWL programmed value 10: RBUF is greater than or equal to the AFWL programmed value but not full 11: full (hold this value on overflow)	R

36.5.7 CAN Receive and Transmit Interrupt Enable Register (CAN_RTIE)

Offset address: 0xA4

Reset value: 0xFE

B7	b6	b5	b4	b3	b2	b1	b0
RIE	ROIE	RFIE	RAFIE	TPIE	TSIE	EIE	TSFF

Bit	Symbol	Bit name	Description	Read and write
B7	RIE	Receive Interrupt Enable	Receive interrupt enable 0: Disable 1: Enable	R/W
b6	ROIE	RB Overrun Interrupt Enable	Receive overrun interrupt enable 0: Disable 1: Enable	R/W
b5	RFIE	RB Full Interrupt Enable	Receive buffer full interrupt enable 0: Disable 1: Enable	R/W
b4	RAFIE	RB Almost Full Interrupt Enable	Receive buffer almost full interrupt enable 0: Disable 1: Enable	R/W
b3	TPIE	Transmission Primary Interrupt Enable	PTB transmission interrupt enable 0: Disable 1: Enable	R/W
b2	TSIE	Transmission Secondary Interrupt Enable	STB transmission interrupt enable 0: Disable 1: Enable	R/W
b1	EIE	Error Interrupt Enable	Error interrupt enable 0: Disable 1: Enable	R/W
b0	TSFF	Transmit Buffer Full Flag	TTEN=0 or TTTBM=0: STB full flag (Transmit Secondary buffer Full Flag) 0: STB slots are not fully filled 1: STB slots are fully filled TTEN=1 and TTTBM=1: TB full flag (Transmit buffer Full Flag) 0: The transmit buffer selected by TBPTR is empty 1: The transmit buffer selected by TBPTR is filled	R

36.5.8 CAN Receive and Transmit Interrupt Flag Register (CAN_RTIF)

Offset address: 0xA5

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
RIF	ROIF	RFIF	RAFIF	TPIF	TSIF	EIF	AIF

Bit	Symbol	Bit name	Description	Read and write
B7	RIF	Receive Interrupt Flag	Receive interrupt flag 0: No frame has been received 1: Data or a remote frame has been received and is available in the receive buffer Write 1 to clear by application program.	R/W
b6	ROIF	RB Overrun Interrupt Flag	Receive buffer overrun flag 0: No RB is overwritten 1: At least one received message has been overwritten in the receive buffer Both ROIF and RFIF are set to 1 on overflow. Write 1 to clear by application program.	R/W
b5	RFIF	RB Full Interrupt Flag	Receive buffer full interrupt flag 0: Receive buffers are not full 1: Receive buffers are full Write 1 to clear by application program.	R/W
b4	RAFIF	RB Almost Full Interrupt Flag	Receive buffer almost full interrupt flag 0: The number of filled RB slots is less than the AFWL setting value 1: The number of filled RB slots is greater than or equal to the AFWL setting value Write 1 to clear by application program.	R/W
b3	TPIF	Transmission Primary Interrupt Flag	PTB transmission interrupt flag 0: No PTB transmission completed 1: The requested PTB transmission is completed successfully Write 1 to clear by application program. Note: In TTCAN mode, TPIF is invalid, only the TSIF flag is valid	R/W
b2	TSIF	Transmission Secondary Interrupt Flag	STB transmission interrupt flag 0: No STB transmission completed 1: The requested STB transmission is completed successfully Write 1 to clear by application program. Note: In TTCAN mode, TPIF is invalid, only the TSIF flag is valid	R/W
b1	EIF	Error Interrupt Flag	Error interrupt flag 0: The BUSOFF bit has not changed, or the relative relationship between the value of the error counter and the set value of the ERROR warning limit has not changed. 1: The BUSOFF bit changes, or the relative relationship between the value of the error counter and the set value of the ERROR warning limit changes. For example, the value of the error counter changes from less than the set value to greater than the set value, or from greater than the set value to less than the set value. Write 1 to clear by application program.	R/W
b0	AIF	Abort Interrupt Flag	Abort interrupt flag 0: No abort has been executed 1: The message(s) requested by TPA or TSA to transmit have been aborted. Write 1 to clear by application program.	R/W

36.5.9 CAN Error Interrupt Enable and Flag Register (CAN_ERRINT)

Offset address: 0xA6

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
EWARN	EPASS	EPIE	EPIF	ALIE	ALIF	BEIE	BEIF

Bit	Symbol	Bit name	Description	Read and write
B7	EWARN	Error Warning	Reached the EWL setting value 0: Both RECNT and TENCNT are less than the EWL setting value 1: RECNT or TENCNT is greater than or equal to the EWL setting value	R
b6	EPASS	Error Passive	Error passive mode active 0: The node is error active 1: The node is error passive	R
b5	EPIE	Error Passive Interrupt Enable	Error passive interrupt enable 0: Disable 1: Enable	R/W
b4	EPIF	Error Passive Interrupt Flag	Error passive interrupt flag 0: No change from error active to error passive or error passive to error active 1: A change from error active to error passive or error passive to error active Write 1 to clear by application program.	R/W
b3	ALIE	Arbitration Lost Interrupt Enable	Arbitration lost interrupt enable 0: Disable 1: Enable	R/W
b2	ALIF	Arbitration Lost Interrupt Flag	Arbitration lost interrupt flag 0: The node won the arbitration 1: The node lost the arbitration Write 1 to clear by application program.	R/W
b1	BEIE	Bus Error Interrupt Enable	Bus error interrupt enable 0: Disable 1: Enable	R/W
b0	BEIF	Bus Error Interrupt Flag	Bus error interrupt flag 0: No bus error 1: Bus error Write 1 to clear by application program.	R/W

36.5.10 CAN Slow Bit Timing Register (CAN_SBT)

Offset address: 0xA8

Reset value: 0x0102 0203

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
S_PRESC[7:0]								-	S_SJW[6:0]						
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-	S_SEG_2[6:0]								S_SEG_1[7:0]						

Bit	Symbol	Bit name	Description	Read and write
b31~b24	S_PRESC	Slow bit time Prescaler	Prescaler of time quanta(TQ) of slow bit time TQ=CAN clock period*(S_PRESC+1)	R/W
b23	Reserved	-	The reset value must be maintained.	R
b22~b16	S_SJW	Slow bit time SJW	Synchronization Jump Width(SJW) of slow bit time SJW=(S_SJW+1)*TQ	R/W
b15	Reserved	-	The reset value must be maintained.	R
b14~b8	S_SEG_2	Slow bit time Segment 2	Bit Timing Segment 2 Bit segment 2 time = (S_SEG_2+1)*TQ	R/W
b7~b0	S_SEG_1	Slow bit time Segment 2	Bit Timing Segment 1 Bit segment 1 time = (S_SEG_1+2)*TQ	R/W

36.5.11 CAN Fast Bit Timing Register (CAN_FBT)

Offset address: 0xAC

Reset value: 0x0102 0203

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
F_PRESC[7:0]								-	-	-	-	F_SJW[3:0]			
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-	-	-	-	F_SEG_2[3:0]				-	-	-	F_SEG_1[4:0]				

Bit	Symbol	Bit name	Description	Read and write
b31~b24	F_PRESC	Fast bit time Prescaler	Prescaler of time quanta(TQ) of fast bit time TQ=CAN clock period*(F_PRESC+1)	R/W
b23~b20	Reserved	-	The reset value must be maintained.	R
b29~b16	F_SJW	Fast bit time SJW	Synchronization Jump Width of fast bit time The Synchronization Jump Width=(F_SJW+1)*TQ	R/W
b15~12	Reserved	-	The reset value must be maintained.	R
b11~b8	F_SEG_2	Fast bit time Segment 2	Bit Timing Segment 2 Bit segment 2 time = (F_SEG_2+1)*TQ	R/W
b7~b5	Reserved	-	The reset value must be maintained.	R
b4~b0	F_SEG_1	Fast bit time Segment 1	Bit Timing Segment 1 Bit segment 1 time = (F_SEG_1+2)*TQ	R/W

36.5.12 CAN Transmitter Delay Compensation Register (CAN_TDC)

Offset address: 0xb1

Reset value: 0x00

b7	b6	b5	b4	b3	b2	b1	b0
TDCEN	SSPOFF[6:0]						

Bit	Symbol	Bit name	Description	Read and write
B7	TDCEN	Transmitter Delay Compensation Enable	Transmitter Delay Compensation Enable For CAN FD and BRS=1 0: Disable 1: Enable	R/W
b6~b0	SSPOFF	Secondary Sample Point Offset	The transmitter delay plus SSPOFF defines the time of the secondary sample point for TDC. SSPOFF is given as a number of TQ	R/W

36.5.13 CAN Error and Arbitration Lost Capture Register (CAN_EALCAP)

Offset address: 0xB0

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0			
				KOER[2:0]						ALC[4:0]

Bit	Symbol	Bit name	Description	Read and write
b7~b5	KOER	Kind Of Error	Error code 000: no error 001: Bit error 010: Form error 011: Stuff error 100: ACK error 101: CRC error 110: Other errors 111: reserved KOER is updated with each new error. Therefore it stays untouched when frames are successfully transmitted or received.	R
b4~b0	ALC	Arbitration Lost Capture	Arbitration lost position ALC records the bit position in the frame where the arbitration has been lost.	R

36.5.14 CAN Warning Limits Register (CAN_LIMIT)

Offset address: 0xA7

Reset value: 0x1B

B7	b6	b5	b4	b3	b2	b1	b0			
				AFWL[3:0]						EWL[3:0]

Bit	Symbol	Bit name	Description	Read and write
b7~b4	AFWL	Almost Full Warning Limit	Receive buffer almost full warning limit The set value range is 1~8. AFWL=0 is meaningless and is treated as AFWL=1.	R/W
b3~b0	EWL	Error Waring Limit	Programmable error warning limit Error Waring Limit=(EWL+1)*8. The value of EWL affects the EIF flag.	R/W

36.5.15 CAN Receive Error Counter Register (CAN_RECNT)

Offset address: 0xB2

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
RECNT[7:0]							

Bit	Symbol	Bit name	Description	Read and write
b7~b0	RECNT	Receive Error Count	Receive error countRECNT is incremented and decremented as defined in the CAN specification. RECNT does not overflow.	R

36.5.16 CAN Transmit Error Counter Register (CAN_TECNT)

Offset address: 0xB3

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
TEMCNT[7:0]							

Bit	Symbol	Bit name	Description	Read and write
b7~b0	TECNT	Transmit Error Count	Transmit error count TECNT is incremented and decremented as defined in the CAN specification. In case of the "bus off state" TECNT may overflow.	R

36.5.17 CAN Acceptance Filter Control Register (CAN_ACFCTRL)

Offset address: 0xB4

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	B0
-		SELMASK	-				ACFADR

Bit	Symbol	Bit name	Description	Read and write
b7~b6	Reserved	-	The reset value must be maintained.	R
b5	SELMASK	Select acceptance MASK	Select filter mask register 0: ACF points to the filter ID register 1: ACF points to the filter MASK register Select a specific filter register set via ACFADR	R/W
b4	Reserved	-	The reset value must be maintained.	R
b3~b0	ACFADR	Acceptance Filter Address	Acceptance filter address ACFADR points to a specific filter, and uses SELMASK to distinguish ID and MASK. 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: Points to ACF_6 0110: Points to ACF_7 0111: Points to ACF_8 1000: points to ACF_9 1001: Points to ACF_10 1010: points to ACF_11 1011: Points to ACF_12 1100: points to ACF_13 1101: Points to ACF_14 1110: Points to ACF_15 1111: Points to ACF_16	R/W

36.5.18 CAN Acceptance Filter Enable Register (CAN_ACFEN)

Offset address: 0xB6

Reset value: 0x0001

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
AE_16	AE_15	AE_14	AE_13	AE_12	AE_11	AE_10	AE_9	AE_8	AE_7	AE_6	AE_5	AE_4	AE_3	AE_2	AE_1

Bit	Symbol	Bit name	Description	Read and write
b15	AE_16	ACF_16 Enable	Acceptance filter 16 enable 0: Disable 1: Enable	R/W
b14	AE_15	ACF_15 Enable	Acceptance filter 15 enable 0: Disable 1: Enable	R/W
b13	AE_14	ACF_14 Enable	Acceptance filter 14 enable 0: Disable 1: Enable	R/W
b12	AE_13	ACF_13 Enable	Acceptance filter 13 enable 0: Disable 1: Enable	R/W
b11	AE_12	ACF_12 Enable	Acceptance filter 12 enable 0: Disable 1: Enable	R/W
b10	AE_11	ACF_11 Enable	Acceptance filter 11 enable 0: Disable 1: Enable	R/W
b9	AE_10	ACF_10 Enable	Acceptance filter 10 enable 0: Disable 1: Enable	R/W
b8	AE_9	ACF_9 Enable	Acceptance filter 9 enable 0: Disable 1: Enable	R/W
B7	AE_8	ACF_8 Enable	Acceptance filter 8 enable 0: Disable 1: Enable	R/W
b6	AE_7	ACF_7 Enable	Acceptance filter 7 enable 0: Disable 1: Enable	R/W
b5	AE_6	ACF_6 Enable	Acceptance filter 6 enable 0: Disable 1: Enable	R/W
b4	AE_5	ACF_5 Enable	Acceptance filter 5 enable 0: Disable 1: Enable	R/W
b3	AE_4	ACF_4 Enable	Acceptance filter 4 enable 0: Disable 1: Enable	R/W
b2	AE_3	ACF_3 Enable	Acceptance filter 3 enable 0: Disable 1: Enable	R/W
b1	AE_2	ACF_2 Enable	Acceptance filter 2 enable 0: Disable 1: Enable	R/W
b0	AE_1	ACF_1 Enable	Acceptance filter 1 enable 0: Disable 1: Enable	R/W

36.5.19 CAN Acceptance CODE and MASK Register(CAN_ACF)

Offset address: 0xB8

Reset value: 0xFFFF XXXX

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	AIDEE	AIDE	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]															

Bit	Symbol	Bit name	Description	Read and write
b31	Reserved	-	The read value is indeterminate.	R
b30	AIDEE	Acceptance mask IDE bit check Enable	IDE bit check enable Only valid when SELMASK=1 0: Acceptance filter accepts both standard or extended frames. 1: Acceptance filter accepts either standard or extended as defined by AIDE.	R/W
b29	AIDE	Acceptance mask IDE bit value	IDE bit value 0: Acceptance filter accepts only standard frames. 1: Acceptance filter accepts only extended frames.	R/W
b28~b0	ACODE/AMASK	Acceptance CODE/ Acceptance MASK	Acceptance filter code Point to specific filters via ACFADR. When SELMASK=0, it means the CODE of the filter. Bit 10~bit 0 are used in the standard format, and bit 29~bit 0 are used in the extended format. Acceptance filter mask Point to specific filters via ACFADR. When SELMASK=1, it indicates the MASK of the filter. Bit 10~bit 0 are used in the standard format, and bit 29~bit 0 are used in the extended format.	R/W

36.5.20 TTCAN TB Slot Pointer Register(CAN_TBSLOT)

Offset address: 0xBE

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
TBE	TBF	-	-	-			TBPTR[2:0]

Bit	Symbol	Bit name	Description	Read and write
B7	TBE	set TB to Empty	Marks the transmit buffer as empty 0: No action 1: The slot selected by TBPTR is marked as empty TBE is automatically reset to 0 as soon as the slot is marked as empty and TSFF=0. If a transmission from this slot is active, then TBE stays set as long as either the transmission completes or after a transmission error or arbitration loss the transmission is not active any more. If both TBF and TBE are set, then TBE wins.	R/W
b6	TBF	set TB slot to Filled	Marks the transmit buffer as filled 0: No action 1: The slot selected by TBPTR is marked as filled TBE is automatically reset to 0 when SLOT is marked as filled and TSFF=1.	R/W
b5~b3	Reserved	-	The reset value must be maintained.	R/W
b2~b0	TBPTR	TB slot Pointer	Pointer to a transmit buffer slot 000: Pointer to PTB 001: Pointer to STB slot 1 010: Pointer to STB slot 2 011: Pointer to STB slot 3 other: reserved The message slot pointed to by TBPTR is readable / writable using the TBUF registers. Write access is only possible if TSFF=0. Setting TBF to 1 marks the selected slot as filled and setting TBE to 1 marks the selected slot as empty. In TTCAN mode, the TBSEL and TSNEXT registers have no effect. Note: This bit can only be written when TSFF=0.	R/W

36.5.21 TTCAN Time Trigger Configuration Register (CAN_TTCFG)

Offset address: 0xBF

Reset value: 0x90

B7	b6	b5	b4	b3	b2	b1	b0
WTIE	WTIF	TEIF	TTIE	TTIF	T_PRESC[1:0]		TTEN

Bit	Symbol	Bit name	Description	Read and write
b7	WTIE	Watch Trigger Interrupt Enable	Watch trigger interrupt enable 0: Disable 1: Enable	R/W
b6	WTIF	Watch Trigger Interrupt Flag	Watch trigger interrupt flag WTIF will be set if the cycle count reaches the limited defined by TT_WTRIG and WTIE is set. Write 1 to clear by application program.	R/W
b5	TEIF	Trigger Error Interrupt Flag	Trigger error interrupt flag TEIF is set when the set value of TT_TTIG is less than the actual CYCLE_TIME. Write 1 to clear by application program.	R/W
b4	TTIE	Time Trigger Interrupt Enable	Time trigger interrupt enable 0: Disable 1: Enable	R/W
b3	TTIF	Time Trigger Interrupt Flag	Time trigger interrupt flag TTIF will be set if TTIE is set and the cycle time is equal to the trigger time TT_TRIG. TTIF will be set only once. If TT_TRIG gets not updated, then TTIF will be not set again in the next basic cycle. Write 1 to clear by application program.	R/W
b2~b1	T_PRESC	TTCAN Timer Prescaler	TTCAN time prescaler 00: Divide by 1 of the bit time set by the SBT register 01: Divide by 2 of the bit time set by the SBT register 10: Divide by 4 of the bit time set by the SBT register 11: Divide by 8 of the bit time set by the SBT register Note: T_PRESC can only be modified if TTEN=0, but it is possible to modify T_PRESC and set TTEN simultaneously with one write access.	R/W
b0	TTEN	Time Trigger Enable	TTCAN enable 0: Disable 1: Enable TTCAN, the timer is running.	R/W

36.5.22 TTCAN Reference Message Register (CAN_REF_MSG)

Offset address: 0xC0

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
REF_IDE	-	REF_ID[28:16]													
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
REF_ID[15:0]															

Bit	Symbol	Bit name	Description	Read and write
b31	REF_IDE	Reference message IDE bit	Reference message IDE bit 0: Standard format 1: Extended format	R/W
b30~b29	Reserved	-	The read value is indeterminate.	R/W
b28~b0	REF_ID	Reference message Identifier	Reference message ID 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 applies to both transmission and reception. After the reference message is detected, the Sync_Mark of the current frame becomes Ref_Mark. REF_ID[2:0] is fixed at 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, it needs to wait for 6 CAN clock cycles to complete the transfer of REF_MSG to the CAN clock domain.	R/W

36.5.23 TTCAN Trigger Configuration Register (CAN_TRG_CFG)

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]			

Bit	Symbol	Bit name	Description	Read and write
b15~b12	TEW	Transmit Enable Window	Transmit enable window For a single shot transmit trigger, a window of TEW+1 cycle time can be set, and the transmission can only be started in this window.	R/W
b11	Reserved	-	The reset value must be maintained.	R
b10~b8	TTYPE	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 other: No action The trigger time is set by the TT_TRIG register, and the TB Slot is selected by TTPTR.	R/W
b7~b3	Reserved	-	The reset value must be maintained.	R
b2~b0	TTPTR	Transmit Trigger TB slot Pointer	Transmit trigger TB slot pointer 000: Points to PTB 001: Points to STB slot 1 010: Points to STB slot 2 011: Points to STB slot 3 other: reserved If TTPTR points to an empty slot, then TEIF will be set at the moment, when the trigger time is reached.	R/W

36.5.24 TTCAN Trigger Time Register (CAN_TT_TRIG)

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]															

Bit	Symbol	Bit name	Description	Read and write
b15~b0	TT_TRIG	Trigger Time	Trigger Time Used to specify the cycle time of the trigger. For a transmission trigger the earliest point of transmission of the SOF of the appropriate frame will be TT_TRIG+1. A write access to the high byte starts a data transfer of the trigger definition to the CAN clock domain and activates the trigger. Therefore, if the BYTE operation is performed, the low byte needs to be written first and then the high byte.	R/W

36.5.25 TTCAN Watch Trigger Time Register (CAN_TT_WTRIG)

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]															

Bit	Symbol	Bit name	Description	Read and write
b15~b0	TT_WTRIG	Watch Trigger Time	Watch trigger time Cycle time for a watch trigger. A write access to the high byte starts a data transfer of the trigger definition to the CAN clock domain. Therefore, if the BYTE operation is performed, the low byte needs to be written first and then the high byte.	R/W

36.6 Precautions for use

36.6.1 CAN bus anti-interference measures

CAN bus is widely used in automobile, industrial control, and other industries. If the electromagnetic environment of the CAN application site is relatively harsh, there are factors such as circuit imbalance, space electromagnetic field, power grid incoming lines, etc., which will cause the CAN bus to generate a lot of communication noise due to radiation and conduction interference, resulting in the increase of bus error frames, frequent retransmissions, and the failure of correct data to arrive in time, which seriously affects the quality of data communication. Therefore, in practical applications, we should focus on eliminating noise interference and ensuring the stable operation of the CAN bus network.

The following are several types of commonly used CAN bus anti-interference measures (including but not limited to)

1. Increase the electrical isolation of CAN bus interface
2. The GND of the transceiver needs to be connected to a common GND
3. Use shielded twisted pair cables and ground them properly
4. Improve the twisted pair degree of CAN transmission line
5. Add a signal protector
6. Improve network topology
7. Application layer software anti-interference mechanism

36.6.2 CAN Controller Noise Constraints

In the CAN bus network, it should be ensured that the bit time of communication meets the requirements of the standard protocol. If the noise that does not meet the bit time width is introduced, it may cause abnormal actions of the CAN controller.

37 CAN2.0B Controller (CAN2.0B)

37.1 Introduction

The CAN 2.0B controller corresponds to CAN controller channel 1 (CAN_1) or CAN controller channels 1 and 2 (CAN_1/CAN_2) according to different product models. For details, please refer to the model function comparison table.

CAN (Controller Area Network) bus is a bus standard that can realize mutual communication between microprocessors or devices without a host.

The CAN2.0B controller follows the CAN bus CAN2.0 (CAN2.0A, CAN2.0B) protocol.

The CAN bus controller can handle data transmission and reception on the bus. In this product, the CAN2.0B controller has 16 acceptance filters. Filters are used to select messages for an application to receive.

The application program can transmit data through one high-priority primary transmit buffer (PTB) and 3 secondary transmit buffers (STB), the transmission order of transmit buffers is determined by the transmission scheduler. There are 8 receive buffers (RB) for data reception. 3 STBs and 8 RBs can be understood as a 3-level FIFO and an 8-level FIFO, and the FIFO is completely controlled by hardware.

CAN2.0B bus controller can also support time-triggered CAN communication (Time-triggered communication).

CAN main features:

- Fully supports CAN2.0A/CAN2.0B protocol.
- CAN2.0B is upward compatible with CAN FD.
- CAN2.0B supports the highest communication baud rate of 1Mbit/s
- Supports 1~1/256 baud rate prescaler, flexible baud rate configuration.
- 8 receive buffers
 - FIFO mode
 - Incorrect or unreceived data does not overwrite stored messages
- 1 high priority primary transmit buffer PTB
- 3 secondary transmit buffers STB
 - FIFO mode
 - Priority decision mode
- 16 independent filters
 - Supports 11-bit standard ID and 29-bit extended ID
 - Programmable ID CODE bit and MASK bit
- Both PTB/STB support single shot transmission mode

- Supports listen only mode
- Supports loopback mode
- Supports capturing of last occurred kind of error and arbitration lost position
- Programmable error warning limit
- Supports ISO11898-4 Time-Triggered CAN with partial hardware

37.2 CAN system block diagram

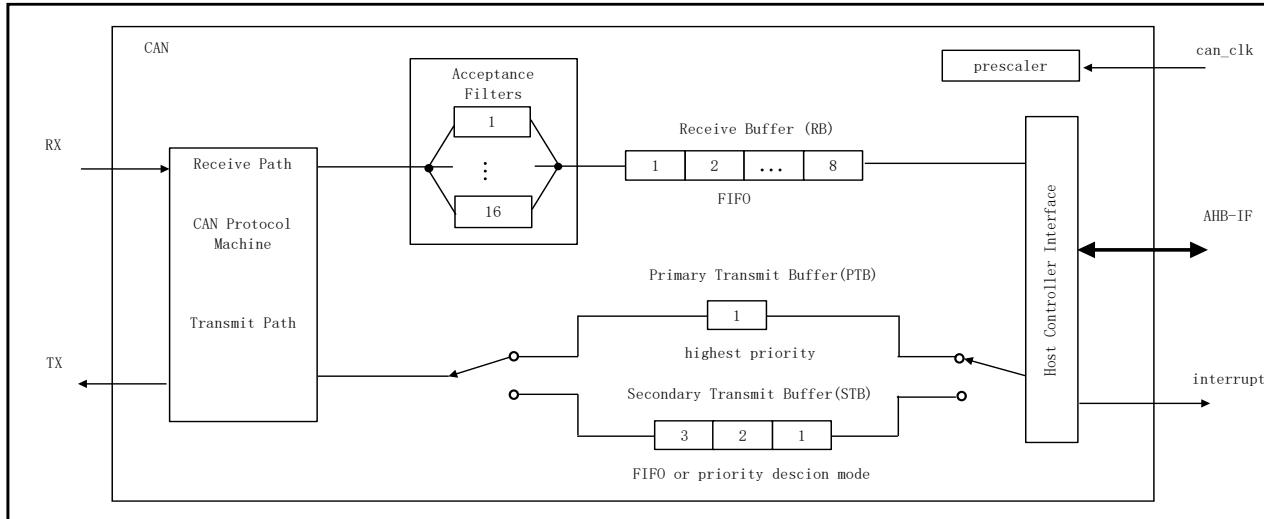


Figure 37-1 CAN system block

37.3 Pin description

Table 37-1 CAN pin description

Pin name	Direction	Functional description
CANn_RX	Input	CAN receive data signal
CANn_TX	Output	CAN transmit data signal
CANn_TST_SAMPLE	Output	For observation only, shows the used sample points (activated one period after the sample point)
CANn_TST_CLOCK	Output	For observation only, shows the used bit time periods (activated one period before the bit time starts)

n: 1~2

37.4 Function description

37.4.1 Operation mode

CAN2.0 controller has two operation modes, reset mode (CAN_CFG_STAT.RESET=1) and action mode (CAN_CFG_STAT.RESET=0). When the module is initialized, the registers that can only be operated in the reset mode should first be set (see the register description chapter for details), and then exit the reset mode, operate the remaining registers in the action mode.

37.4.2 Baud rate setting

CAN communication clock uses can_clk. Before using the CAN module, first set the CAN communication clock according to CMU chapter. The clock selection must satisfy the setting condition that CAN control logic clock(PCLK1) is 1.5 times or more than CAN communication clock(can_clk).

The following figure shows the definition of CAN bit time. The part on the dotted line is the bit time specified by the CAN protocol, and the part below the dotted line is the bit time defined by CAN-CTRL. Among them, segment1 and segment2 can be set by register SBT. The SBT register can only be set when CAN_CFG_STAT.RESET=1.

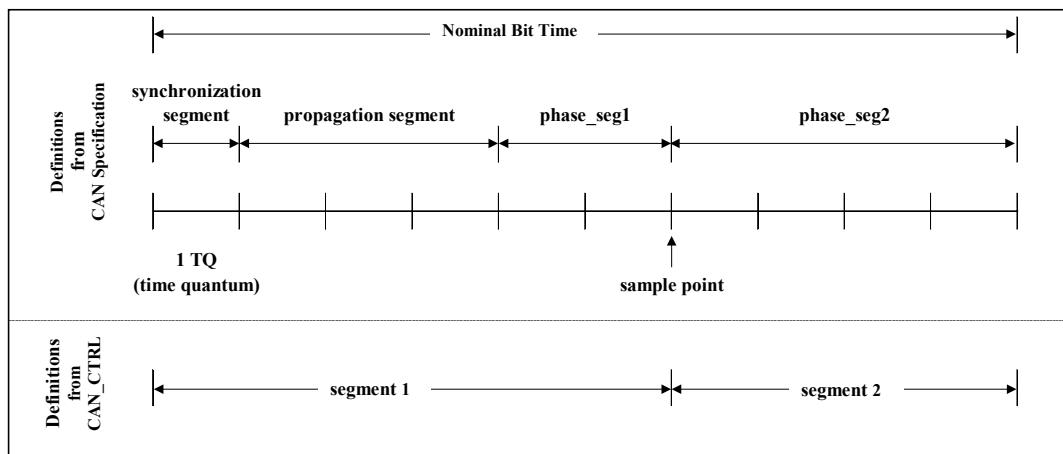


Figure 37-2 CAN Bit Time Definition

Refer to the following formula for the TQ calculation , PRESC is set by SBT.S_PRESC. f_{can_clk} is the CAN communication clock frequency.

$$S_TQ = \frac{S_PRESC+1}{f_{can_clk}}$$

Refer to the following formula for the bit time calculation, S_segment1 and S_segment2 are set by SBT.S_SEG_1 and SBT.S_SEG_2.

$$\text{Slow_BT} = t_{S_segment1} + t_{S_segment2} = ((S_SEG_1+2)+(S_SEG_2+1)) \times S_TQ$$

Table 37-2 CAN bit time setting rules

Register	Parameter range	Rule
SBT.S_SEG_1	[0..63] CAN2.0 bits (slow)	SEG_1 ≥ SEG_2 + 1 SEG_2 ≥ SJW
SBT.S_SEG_2	[0..7] CAN2.0 bits (slow))	
SBT.S_SJW	[0..15] CAN2.0 bits (slow)	

37.4.3 Transmit buffer (TB)

CAN_CTRL provides two transmit buffers for transmitting data, the primary transmit data buffer PTB and the secondary transmit buffer STB. PTB has the highest priority but can only buffer one frame. The priority of STB is lower than PTB, but it can buffer 3 frames, and the 3 frames in STB can act in FIFO mode or priority decision mode.

The 3 frames in the STB can be all transmitted by setting TCMD.TSALL to 1. In FIFO mode, the frame filled first is transmitted first, and in priority decision mode, the frame with the smallest ID is transmitted first.

The frame in the PTB has the highest priority, so the PTB transmission can postpone the STB transmission, but the STB that has won arbitration and started transmission cannot be postponed by the PTB transmission.

PTB and STB can be accessed through the TBUF register. Select PTB or STB through TCMD.TBSEL, TBSEL=0, select PTB, TBSEL=1, select STB. The next SLOT in the STB is selected by TCTRL.TSNEXT. The corresponding relationship is shown in the following figure:

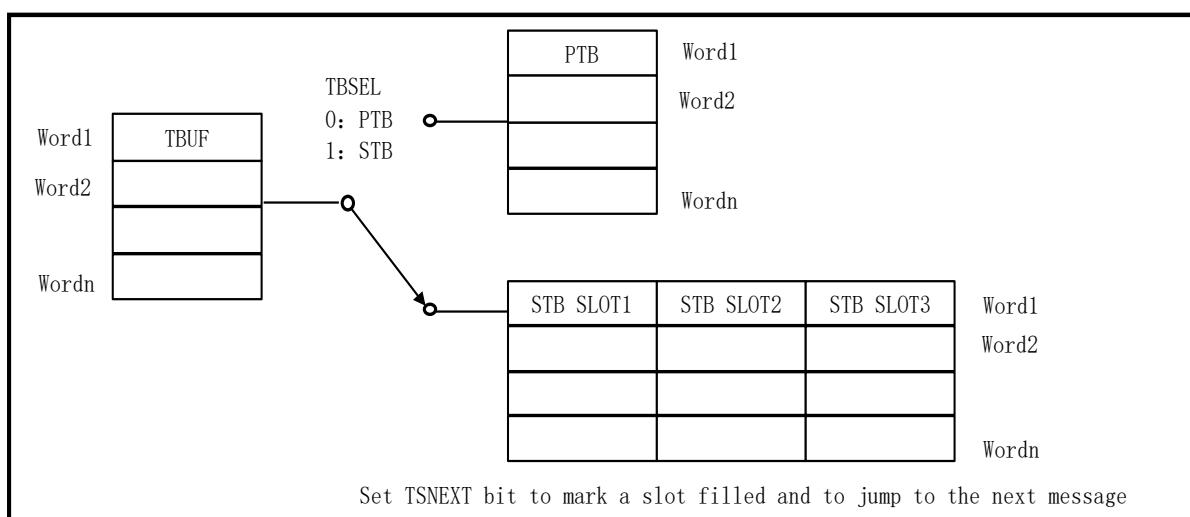


Figure 37-3 Schematic of CAN TBUF register write transmit buffer

37.4.4 Receive buffer (RB)

CAN_CTRL provides 8 receive buffer slots which act in FIFO mode for storing received data. RB slot reads the received data through the RBUF register, always reads the oldest received data first, and releases the RB slot that has been read by setting the RCTRL.RREL to 1, and points to the next RB slot.

The schematic diagram of reading RB slot by RBUF is as follows.

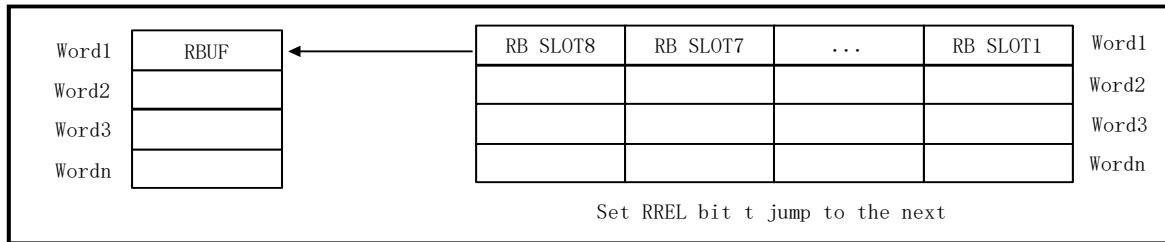


Figure 37-4 Schematic of CAN RBUF register read receive buffer

37.4.5 Acceptance filters

CAN_CTRL provides 16 independent 32-bit filters for filtering received data to reduce CPU load. The filters can support standard format 11-bit ID or extended format 29-bit ID. Each filter has a 32-bit ID CODE register and a 32-bit ID MASK register, the ID CODE register is used to compare the received CAN ID, and the ID MASK register is used to select the CAN ID bits for comparison. When the corresponding ID MASK bit is 1, the ID CODE of this bit is not compared.

The received data will be received as long as it passes any one of the 16 filters, and the received data will be stored in the RB, otherwise the data will not be received or stored.

Each filter is enabled or disabled through the ACFEN register. ID CODE and ID MASK are set by ACFCTRL.SELMASK. When SELMASK=0, it points to ID CODE, and when SELMASK=1, it points to ID MASK. The filter is selected by ACFCTRL.ACFFADR. ID CODE and ID MASK are accessed through ACF register and can only be set when CFG_STAT.RESET=1. Refer to the following figure for the way of ACF register access to filter register group.

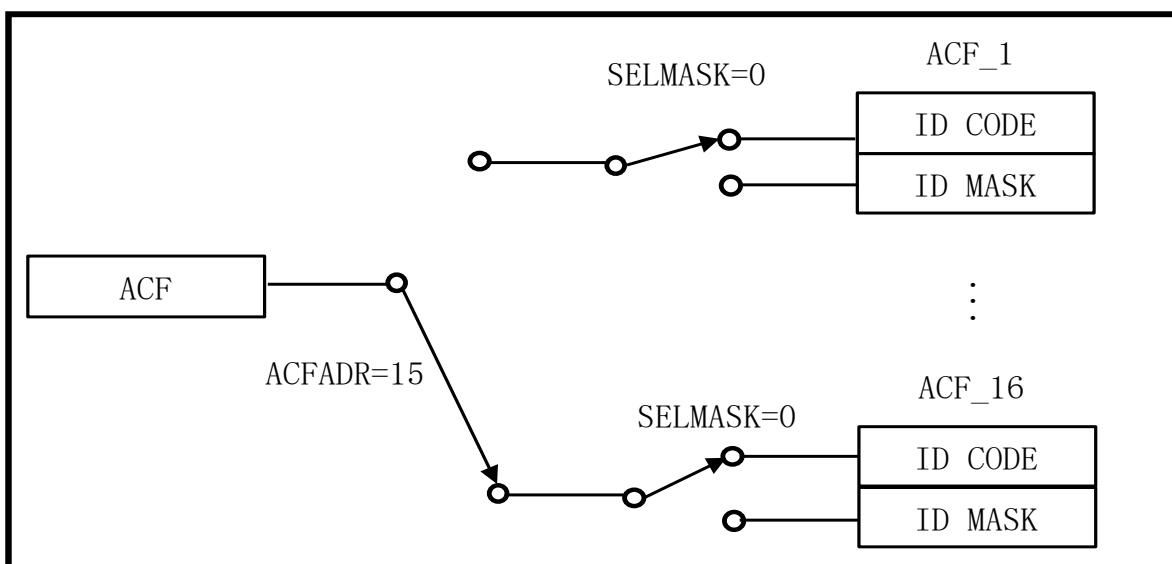


Figure 37-5 Access to the CAN Acceptance Filters

37.4.6 Data transmission

Before transmitting, ensure that the data to be transmitted by the PTB or STB has been filled, and then start the PTB or STB transmission. Refill data is not allowed during transmitting.

The steps of transmitting data setting are as follows:

1. Set TBSEL to select TB from PTB and STB
2. Write the data to be transmitted through the TBUF register.
3. If STB is selected, set TSNEXT=1 to complete the loading of all STB SLOTS.
4. Transmission enable
 - PTB transmission using TPE
 - STB transmission using TSALL or TSONE
5. Transmission completion status confirmation
 - For the PTB, TPIF is set if TPIE is enabled
 - For the STB using TSONE, TSIF is set if one message has been completed and TSIE is enabled
 - For the STB using TSALL, TSIF is set if all messages have been completed and if TSIE is enabled

37.4.7 Single shot transmission

When the automatic retransmission function is not required, it can be set to the single shot transmission mode through the register. PTB single shot transmission is set by CFG_STAT.TPSS and STB is set by CFG_STAT.TSSS. When the data is successfully transmitted, the operation of single shot transmission is the same as the normal transmission. But the following results appear when the data is not transmitted successfully:

- When TPIF is set (if TPIE=1), the corresponding transmit buffer slot will be cleared.
- When there is an error occurred during the transmission, KOER is updated and BEIF is set (if BEIE=1).
- Arbitration is lost and ALIF is set (ALIE=1).

In single shot transmission mode, TPIF cannot be relied on alone to judge whether the transmission is completed. It is necessary to judge whether the transmission is completed together with BEIF and ALIF.

37.4.8 Transmission abort

Data transmissions that have been requested but not yet executed can be aborted by TPA or TSA. Transmission abort will be executed in the following situations:

- There is no abort during bus arbitration
 - If the node loses arbitration, the abort will be executed afterwards.
 - If the node wins arbitration, the frame will be transmitted.

- There is no abort while a frame is transmitted
 - If a frame is transmitted successfully and ACK is received, the corresponding flag and status are normally set. In this case no abort is signaled.
 - After an unsuccessful transmission where the CAN node does not receive an ACK, the error counter is incremented and the abort will be executed.
 - For the transmission requested by TSALL=1, the STB slot that is being transmitted is transmitted normally, and the STB slot that has not started to transmit will be aborted.

If an abort is executed, this results in the following actions.

- TPA releases the PTB which results in TPE=0.
- TSA releases one single slot or all slots of the STB, depends on whether TSONE or TSALL was used to start the transmission.

37.4.9 Data reception

Use the acceptance filters can filter out unnecessary receive data, reduce the occurrence of interrupts and the reading of RBs, thereby reducing the CPU load. The steps for receiving data setting are as follows:

1. Configures acceptance filters.
2. Set RFIE, RAFIE and AFWL.
3. Wait for RFIF or RAFIF.
4. Read the oldest received data from the RB FIFO via RBUF.
5. Set RREL=1 to select the next RB slot.
6. Repeat 4, 5 until RSTAT signals an empty RB.

37.4.10 Error handling

On one hand CAN_CTRL does automatic error handling. This includes automatic retransmission and automatic deletion of received messages with errors. On the other hand, CAN_CTRL may report errors to the CPU through interrupts.

CAN nodes have the following three error states:

- Error active: An active error flag is automatically transmitted when a node detects an error.
- Error passive: A passive error flag is automatically transmitted when the node detects an error.
- Node Bus Off: In Bus Off state, this node no longer affects the entire CAN network.

CAN_CTRL provides two counters, TECNT and RECNT, for counting errors. The TECNT and RECNT counters are incremented and decremented according to the rules specified by the CAN protocol.

In addition, a programmable CAN error warning LIMIT register is provided for generating an error interrupt to notify the CPU.

There are the following 5 kinds of error in the CAN communication process, and the errors can be identified by EALCAP.KOER.

- Bit error
- Form error
- Stuff error
- ACK error
- CRC error

37.4.11 Bus Off

When the transmit error counter becomes > 255, the CAN node automatically enters the Bus Off state and does not participate in CAN communication until it returns to the error-active state. The Bus Off status can be confirmed by CFG_STAT.BUSOFF. The EIF interrupt is generated while BUSOFF is set.

There are two ways to recover from the Bus Off state to the error-active state:

- Power-on reset
- Received 128 sequences of 11-bit recessive bit (recovery sequence)

When the node is Bus Off, the TECNT value remains unchanged and RECNT is used to count the recovery sequence. TECNT and RECNT are reset to 0 after recovery from the Bus Off state.

37.4.12 Arbitration lost capture

CAN_CTRL can accurately capture the position of the arbitration lost bit and reflect it in the ALC register. The ALC register holds the position of the latest arbitration lost bit. If the node wins the arbitration, the ALC bit is not updated.

The ALC value is defined as follows:

After the SOF bit, the first ID bit ALC is 0, the second ID bit ALC is 1, and so on. The maximum value of ALC is 31 because arbitration only occurs within the arbitration field. For example, a standard remote frame arbitrated with an extended frame, the extended frame lost arbitration at the IDE bit, then ALC=12.

37.4.13 Loopback mode

CAN_CTRL supports the following two loopback modes:

- Internal loopback
- External loopback

Both loopback modes can receive data frames sent by themselves, which are mainly used for self-tests.

In internal loopback mode, the module internally connects rxd to txd, and the txd output is set to recessive. In internal loopback mode, the node generates a self-ACK to avoid ACK error.

The external loopback mode keeps the connection with the transceiver, so the transmitted frame still appear on the CAN bus. With the help of the transceiver, the CAN-CTRL can receive its own frame. In external loopback mode, whether to generate a self-ACK can be determined by RCTRL.SACK. When SACK=0, no self-ACK is generated, and when SACK=1, a self-ACK is generated.

In external loopback mode, when SACK=0, the following two results will be occurred:

- Another node receives the frame too and generates an ACK. This will result in a successful transmission and reception.
- If no other node generates an ACK, ACK error will be occurred, the frame will be retransmitted and the error counter will be incremented. In this case, it is recommended to use the single shot transmission mode.

When returning from loopback mode to normal mode, in addition to switching back to normal mode, software is also required to reset CAN_CTRL.

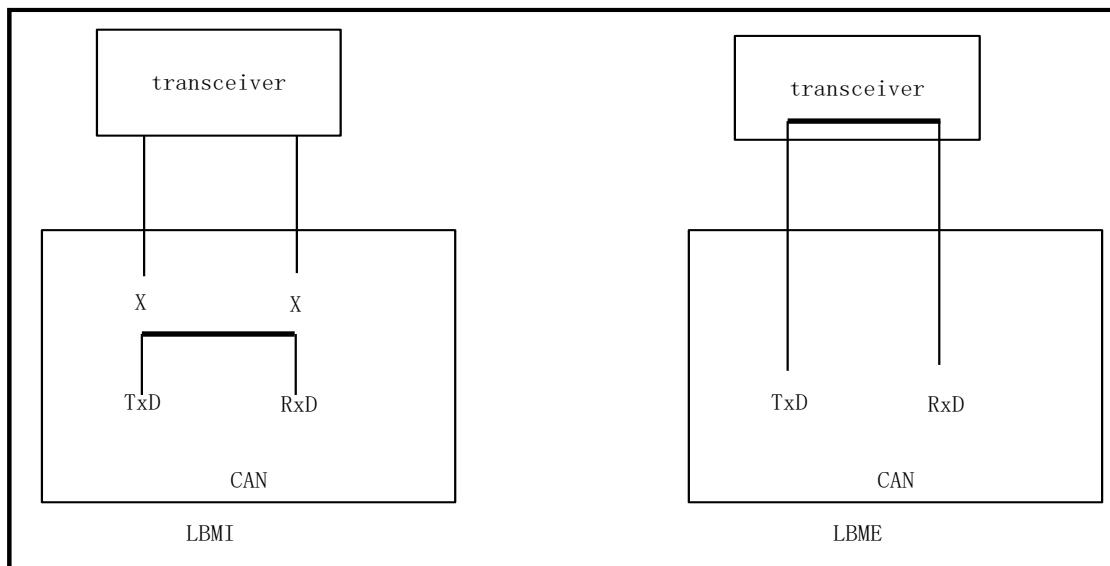


Figure 37-6 Schematic of CAN LBMI and LBME

37.4.14 Listen only mode (LOM)

Listen only mode can be used to monitor CAN network. In this mode, data can be received from the CAN bus without transmitting any data to the bus. Set TCMD.LOM to 1 to put the CAN-CTRL into listen only mode, and clear it to 0 to leave silent mode.

The external loopback mode can be combined with the listen only mode to form the external loopback listen only mode, at which time the CAN can be considered a silent receiver but can transmit data when necessary. In external loopback listen only mode, frames containing self-ACK signals are allowed to be transmitted, but the node will not generate error or overload frames.

37.4.15 Software reset

By setting CFG_STAT.RESET to 1, the software reset function is realized. The range of the software reset function is shown in the following table.

Table 37-3 Software Reset Range Table

Register	Reset	Comment	Register	Reset	Comment
ACFADR	no	-	KOER	Yes	-
ACODE	no	Writeable if RESET=1	LBME	Yes	-
AE_x	no	-	LBMI	Yes	-
AFWL	no	-	RACTIVE	Yes	Reception is immediately cancelled even if a reception is active. No ACK will be generated
AIF	Yes	-	RAFIE	No	-
ALC	Yes	-	RAFIF	Yes	-
ALIE	no	-	RBALL	Yes	-
ALIF	Yes	-	RBUF	Yes	All RB slots are marked as empty. RBUF contains unknown data
AMASK	no	Writeable if RESET=1	RECNT	No	An error counter reset is possible by setting BUSOFF=1
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	
EWL	Yes	-	ROV	Yes	

Register	Reset	Comment	Register	Reset	Comment
RREL	Yes	-	TSMODE	No	
RSTAT	Yes		TSNEXT	Yes	-
SACK	Yes	-	TSONE	Yes	-
SELMASK	no	-	TPIE	No	-
S_PRESC	no	Writeable if RESET=1	TPIF	Yes	-
S_SEG_1	no	Writeable if RESET=1	TPSS	Yes	-
S_SEG_2	no	Writeable if RESET=1	TSFF	Yes	All STB slots are marked as empty

Register	Reset	Comment	Register	Reset	Comment
S_SJW	no	Writeable if RESET=1	TSIE	No	-
TACTIVE	Yes	All transmissions are immediately terminated with RESET	TSIF	Yes	-
TBE	Yes	-	TSSS	Yes	-
TBF	no	-	TSSTAT	Yes	All STB slots are marked as empty
TBPTR	no	-	TTEN	Yes Yes	-
TBSEL	Yes	-	TTIF	Yes	-
TBUF	Yes	All STB slots are marked as empty. Because of TBSEL TBUF points to the PTB	TTIE	No	-
TDCEN	Yes	-	TTPTR	No	-
TECNT	no	An error counter reset is possible by setting BUSOFF=1	TTTBM	No	-
TEIF	Yes	-	TTYPE	No	-
TPA	Yes	-	TT_TRIG	No	-
TPE	Yes	-	TT_WTRIG	No	-
TSA	Yes	-	T_PRESC	No	-
TSALL	Yes	-	WTIE	No	-
			WTIF	Yes	

37.4.16 Upward compatible with CAN-FD function

When CAN-CTRL is disabled in CAN FD function, even if CAN FD frames are received in the network including CAN FD, the receiver will automatically ignore these frames, do not return ACK, and wait until the bus is idle before sending or receiving the next CAN2.0B frame.

37.4.17 Time-Triggered CAN(TTCAN)

CAN-CTRL provides partial (lever 1) hardware support for the time-triggered communication method specified by ISO11898-4. This chapter introduces the TTCAN function from the following 5 parts.

37.4.17.1 TBUF in TTCAN mode

TTTBM=1

When TTTBM=1, PTB and STB SLOT form a TB SLOT, and each slot can be addressed by TBPTR. When TBPTR=0, it points to PTB, and TBPTR=1 points to STB slot 1, and so on. The host can mark a slot as filled by setting TBF or as empty by setting TBE. Filled slots are write-locked. TBSEL and TSNEXT registers have no meaning in TTCAN mode and can be ignored.

When TTTBM=1, PTB has no special properties in this mode and is associated with the STB. This furthermore results in the fact that a successful transmission is always signaled using TSIF.

There is no FIFO operation and no priority decision for the TBUF in TTCAN mode. Furthermore, only one frame can be selected for transmission.

In TTCAN mode, all transmissions can only be started using a trigger, TPE, TSONE, TSALL, TPSS and TPA are fixed to 0 and ignored.

TTTBM=0

TTTBM=0 offers the combination of event-driven CAN communication and time-stamping for received messages. In this mode the PTB and the STB provide the same behavior as if TTEN=0. Then the PTB always has higher priority than the STB and the STB can operate in FIFO mode (TSMODE=0) or in priority mode (TSMODE=1).

37.4.17.2 TTCAN description

After power-up a time master needs to do the initialization as defined in ISO 11898-4. There may be up to 8 potential time masters in a CAN network. Each one has its own reference message ID (the last 3 bits of the ID). Potential time masters may try to transmit their reference message according to their priority. Lower-prioritized time masters shall try to transmit their reference messages later.

If TTEN is set, then the 16-bit timer is running. If a reference message is detected or if a time master successfully transmits its reference message, then CAN-CTRL copies the Sync_Mark of this message to the Ref_Mark which sets the cycle time to 0. The reception of the reference message will set RIF respectively the successful transmission will set TPIF or TSIF. Then the host needs to prepare the trigger for the next action.

A trigger for an action can be a reception trigger. This just triggers the interrupt, and it can be used to detect if an expected message is not received. Such a trigger can also be used for other actions, but this depends on the host application.

A different trigger is a transmission trigger. This starts the frame in the TBUF slot, where the trigger points to with TTPTR. If the TBUF slot is marked as empty, then no transmission is started, but the trigger interrupt is set.

37.4.17.3 TTCAN timing

CAN_CTRL supports ISO 11898-4 level 1. This includes a 16 bit timer running at the speed of a CAN bit time (defined by S_PRESC, S_SEG_1, S_SEG_2). An additional prescaler is defined by T_PRESC. If TTEN=1 then the timer is counting continuously.

At the SOF of the message, the timer value is the Sync_Mark. If the message was a reference message, then this value is copied to the Ref_Mark. The cycle time is the timer value minus the Ref_Mark. It is the time that is used as time-stamp for received messages or as trigger time for messages, that need to be transmitted.

37.4.17.4 TTCAN trigger types

The trigger type is defined by TTYPE. TTPTR is a pointer to a TB slot and TT_TRIG defines the cycle time of the trigger.

It includes the following five trigger types:

- immediate trigger
- time trigger
- single shot transmit trigger
- transmit start trigger
- transmit stop trigger

Except for the immediate trigger, all triggers set TTIF if TTIE is enabled. When TTTBM=1, only time triggers are supported.

Immediate Trigger

An immediate trigger starts the immediate transmission of a frame that is pointed to by TTPTR. TTIF is not set. To start a trigger, TT_TRIG_1 needs to be written. The value that is written, does not care for an immediate trigger.

Time Trigger

A time trigger just generates an event by setting TTIF and therefore generates an interrupt. No other actions are done. If a node expects to receive the expected data within a time window, the time trigger method can be used. If the TT_TRIG value is less than the actual cycle time, TEIF is set and no action is done.

Single Shot Transmit Trigger

A single shot transmit trigger is intended to be used for exclusive time windows, where a message needs to be transmitted in single shot mode. In this case, the TSSS bit is ignored.

For this ISO 11898-4 defines a transmit enable window of up to 16 ticks of the cycle time. The register bits TEW(3:0)+1 define the number of ticks. If the data does not start to transmit within the specified transmit enable time window, the frame is discarded. As a result the TB slot of the frame will be marked as empty and AIF will be set if AIE is set. The data in the corresponding TB will not be rewritten, because it can be sent again by setting TPF.

If the TT_TRIG value is less than the actual cycle time, TEIF is set and no action is done.

Transmit Start Trigger

A transmit start trigger is intended to be used for merged arbitrating time windows, where several nodes may transmit messages and CAN arbitration takes place. The selected message is defined by TTPTR. TSSS defines if retransmission or single shot mode is used. If the selected frame cannot

be transmitted (arbitration loss, several transmissions after an error), then the transmission can be stopped using the transmit stop trigger.

If the TT_TRIG value is less than the actual cycle time, TEIF is set and no action is done.

Transmit Stop Trigger

A transmit stop trigger is intended to be used to stop a transmission, that is started with the transmit start trigger. If a transmission is stopped, then the frame is aborted. Therefore, AIF is set if AIE is set and the TB slot of the frame is marked as empty. The data in the corresponding TB will not be rewritten, because it can be sent again by setting TPF.

If TT_TRIG value is less than the actual cycle time, then TEIF is set but the stop is executed.

37.4.17.5 TTCAN watch trigger

The watch trigger can be used if TTTBM=1. The watch trigger is intended to be used if the time since the last valid reference message has been too long. Reference messages can be received in a periodic cycle or after an event. The host application needs to take care of this and has to adjust the watch trigger accordingly.

If the cycle count equal to the value defined by TT_WTRIG, then WTIF is set is WTIE is set.

If TT_WTRIG is less than the actual cycle time, then TEIF is set.

37.4.18 Interrupt

Table 37-4 CAN Interrupt flag

Channel	Interrupt flag	Description
CAN_1 CAN_2	RIF	Receive interrupt flag
	ROIF	Receive buffer overrun interrupt flag
	RFIF	Receive buffer full interrupt flag
	RAFIF	Receive buffer almost full interrupt flag
	TPIF	PTB transmission interrupt flag
	TSIF	STB transmission interrupt flag
	EIF	Error interrupt flag
	AIF	Abort interrupt flag
	EPIF	Error passive interrupt flag
	ALIF	Arbitration lost interrupt flag
	BEIF	Bus error interrupt flag
	WTIF	Watch trigger interrupt flag
	TEIF	Trigger error interrupt flag
	TTIF	Time trigger interrupt flag

37.5 Register description

CAN_1_BASE_ADDR: 0x40009000 CAN_2_BASE_ADDR: 0x40078000

Table 37-5 CAN register list

Register name	Symbol	Offset address	Bit width	Reset value
CAN Receive Buffer Register	CAN_RBUF	0x00~0x0F	-	0xFFFF XXXX
CAN Transmit Buffer Register	CAN_TBUF	0x50~0x5F	-	0xFFFF XXXX
CAN Configuration and Status Register	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 Register	CAN_RTIE	0xA4	8	0xFE
CAN Receive and Transmit Interrupt Flag Register	CAN_RTIF	0xA5	8	0x00
CAN Error Interrupt Enable and Flag Register	CAN_ERRINT	0xA6	8	0x00
CAN Warning Limits Register	CAN_LIMIT	0xA7	8	0x1B
CAN Slow Bit Timing Register	CAN_SBT	0xA8	32	0x0102 0203
CAN Error and Arbitration Lost Capture Register	CAN_EALCAP	0xB0	8	0x00
CAN Receive Error Counter Register	CAN_RECNT	0xB2	8	0x00
CAN Transmit Error Counter Register	CAN_TENCNT	0xB3	8	0x00
CAN Acceptance Filter Control Register	CAN_ACFCTRL	0xB4	8	0x00
CAN Acceptance Filter Enable Register	CAN_ACFEN	0xB6	8	0x01
CAN Acceptance CODE and MASK Register	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 Watch Trigger Time Register	CAN_TT_WTRIG	0xC8	16	0xFFFF

37.5.1 CAN Receive Buffer Register (CAN_RBUF)

Offset address: 0x00

Reset value: 0XXXX XXXX

The RBUF registers point the message slot with the oldest received message in the receive buffer.

All RBUF registers can be read in any order.

KOER in RBUF has the same meaning as the bits KOER in register EALCAP. KOER in RBUF becomes meaningful if RBALL=1.

The TX bit indicates that the message sent by itself is received in loopback mode.

The CYCLE_TIME bit is only valid in TTCAN mode and represents the cycle time at the SOF of this frame.

CAN_RBUF only supports WORD access.

The data format of the CAN receiving mailbox is as follows:

Table 37-6 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	-	-	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
...	...								Data
...	...								Data
...	...								Data
RBUF+14	DATA7								Data
RBUF+15	DATA8								Data

Table 37-7 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	-	-	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
...	...								Data
...	...								Data
...	...								Data
RBUF+14	DATA7								Data
RBUF+15	DATA8								Data

The meanings of the control bits are as follows:

IDE(IDentifier Extension):

0: Standard format

1: Extended format

RTR (Remote Transmission Request)

0: Data frame

1: Remote frame

DLC(Data Length Code):

Data length code, CAN2.0 setting range is 0~8, corresponding data length is 0Byte~8Byte

Table 37-8 DLC control bits

DLC (binary)	Frame Type	Payload in Bytes
0000~1000	CAN2.0	0~8
1001~1111	CAN2.0	8

The meanings of the status bits are as follows:

KOER: Same as EALCAP.KOER

TX: This bit is set to 1 when receiving data sent by itself in loopback mode

37.5.2 CAN Transmit Buffer Register (CAN_TBUF)

Offset address: 0x50

Reset value: 0xFFFF XXXX

The TBUF registers point the next empty message slot in the STB if TBSEL=1 or to the PTB otherwise. All TBUF registers can be written in any order. For the STB it is necessary to set TSNEXT to mark a slot filled and to jump to the next message slot.

TBUF can only be accessed by WORD.

The data format of CAN sending mailbox is as follows:

Table 37-9 Standard format CAN send 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	-	-	DLC[3:0]				Control
TBUF+5	-								-
TBUF+6	-								-
TBUF+7	-								-
TBUF+8	DATA1								Data
TBUF+9	DATA2								Data
TBUF+10	DATA3								Data
...	...								Data
...	...								Data
...	...								Data
TBUF+14	DATA7								Data
TBUF+15	DATA8								Data

Table 37-10 Extended format CAN send 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	-	-	DLC[3:0]				Control
TBUF+5	-								-
TBUF+6	-								-
TBUF+7	-								-
TBUF+8	DATA1								Data
TBUF+9	DATA2								Data
TBUF+10	DATA3								Data
...	...								Data
...	...								Data
...	...								Data
TBUF+14	DATA7								Data
TBUF+15	DATA8								Data

For the meaning of the control bits, please refer to the description of the CAN Receive Buffer Register chapter.

37.5.3 CAN Configuration and Status Register (CAN_CFG_STAT)

Offset address: 0xA0

Reset value: 0x80

B7	b6	b5	b4	b3	b2	b1	b0
RESET	LBME	LBMI	TPSS	TSSS	RACTIVE	TACTIVE	BUSOFF

Bit	Symbol	Bit name	Description	Read and write
B7	RESET	RESET request	Reset request bit 0: Do not request a partial reset 1: Request a partial reset Some registers can only be written when RESET=1. For details, please refer to the software reset function. When the node enters the BUS OFF state, the hardware will automatically set the RESET bit to 1. Please note that it takes 11 CAN bit times for the node to participate in communication after RESET=0.	R/W
b6	LBME	External Loopback Mode	External loopback mode enable bit 0: Disable external loopback mode 1: Enable external loopback mode Note: LBME should not be enabled while a transmission is active.	R/W
b5	LBMI	Internal Loopback Mode	Internal loopback mode enable bit 0: Disable internal loopback mode 1: Enable internal loopback mode Note: LBMI should not be enabled while a transmission is active.	R/W
b4	TPSS	Transmission Primary Single Shot	PTB single shot transmission mode 0: Disable PTB single shot transmission mode 1: Enable PTB single shot transmission mode	R/W
b3	TSSS	Transmission Secondary Single Shot	STB single shot transmission mode 0: Disable STB single shot transmission mode 1: Enable STB single shot transmission mode	R/W
b2	RACTIVE	Reception Active	Reception status 0: No receive activity 1: Receiving	R
b1	TACTIVE	Transmission Active	Transmission status 0: No transmission activity 1: Transmitting	R
b0	BUSOFF	Bus Off state	Bus off state 0: Error active state 1: Bus off state Note: Writing a 1 clears the TENCNT and RECNT registers, but only for debugging purposes.	R/W

37.5.4 CAN Command Register (CAN_TCMD)

Offset address: 0xA1

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
TBSEL	LOM	-	TPE	TPA	TSONE	TSALL	TSA

Bit	Symbol	Bit name	Description	Read and write
B7	TBSEL	Transmit Buffer Select	<p>Transmit buffer select 0: PTB 1: STB When TTEN=1&TTTBM=1, TBSEL is reset to the reset value. Note: TBSEL needs to be stable all the time the TBUF registers are written and when TSNEXT is set.</p>	R/W
b6	LOM	Listen Only Mode	<p>Silent mode enable bit 0: Disable silent mode 1: Enable silent mode Transmission is prohibited when LOM=1&LBME=0. When LOM=1&LBME=1, it is forbidden to respond to the corresponding received frames and error frames, but the own frames can be transmitted. Note: LOM should not be enabled while a transmission is active.</p>	R/W
b5	Reserved	-	The reset value must be maintained.	R/W
b4	TPE	Transmit Primary Enable	<p>PTB transmission enable bit 0: No transmission for the PTB 1: Enable PTB transmission When this bit is enabled, the message in the PTB will be transmitted at the next available position. An STB transmission that has already started will be completed before, but the next pending STB transmission will be delayed until the PTB transmission has been completed. After this bit is written to 1, it will remain at 1 until the PTB transmission is completed or the transmission is aborted by TPA. Software cannot reset this bit to 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&TTTBM=1</p>	R/W

Bit	Symbol	Bit name	Function	Read and write
b3	TPA	Transmit Primary Abort	<p>Abort PTB transmission 0: No abort 1: Abort the PTB transmission that has been requested by the TPE set but has not yet started</p> <p>This bit is written to 1 by software but cleared by hardware. Setting TPA automatically de-asserts TPE, it should not be set simultaneously with TPE. The TPE is reset to the reset value by hardware in the following cases:</p> <ul style="list-style-type: none"> - RESET=1 - BUOFF=1 - TTEN=1&TTTB=1 	R/W
b2	TSONE	Transmit Secondary One frame	<p>Transmit one frame of STB 0: No transmission for STB 1: Transmit one frame of STB</p> <p>In FIFO mode this is the oldest message and in priority mode this is the one with the highest priority.</p> <p>After this bit is written to 1, it will remain at 1 until the STB transmission is completed or the transmission is aborted by TSA. Software cannot reset this bit to 0.</p> <p>TSONE is reset to the reset value by hardware in the following cases:</p> <ul style="list-style-type: none"> - RESET=1 - BUOFF=1 - LOM=1&LBME=0 - TTEN=1&TTTB=1 	R/W
b1	TSALL	Transmit Secondary All frames	<p>Transmit all frames of STB 0: No transmission for STB 1: Transmit all frames of STB</p> <p>After this bit is written to 1, it will remain at 1 until the STB transmission is completed or the transmission is aborted by TSA. Software cannot reset this bit to 0.</p> <p>TSALL is reset to the reset value by hardware in the following cases:</p> <ul style="list-style-type: none"> - RESET=1 - BUOFF=1 - LOM=1&LBME=0 - TTEN=1&TTTB=1 	R/W
b0	TSA	Transmit Secondary Abort	<p>Abort STB transmission 0: No abort 1: Abort the STB transmission that has been requested by TSONE or TSALL set but has not yet started</p> <p>This bit is written to 1 by software but cleared by hardware. Setting TSA automatically de-asserts TSONE or TSALL respectively, it should not be set simultaneously with TSONE or TSALL.</p> <p>The TSA is reset by hardware to its reset value when:</p> <ul style="list-style-type: none"> - RESET=1 - BUOFF=1 	R/W

37.5.5 CAN Transmit Control Register (CAN_TCTRL)

Offset address: 0xA2

Reset value: 0x90

B7	b6	b5	b4	b3	b2	b1	B0
-	TSNEXT	TSMODE	TTTBM	-	-	-	TSSTAT[1:0]

Bit	Symbol	Bit name	Description	Read and write
B7	Reserved	-	The reset value must be maintained.	R/W
b6	TSNEXT	Transmit buffer Secondary NEXT	<p>Transmit next STB 0: No action 1: The current STB slot is filled and pointing to the next slot After all frame bytes are written to the TBUF registers, the host controller has to set TSNEXT to signal that this slot has been filled. Then the TBUF registers connects to the next slot.</p> <p>The slot is marked as filled a transmission can be started using TSONE or TSALL. TSNEXT has to be set by the host controller and is automatically reset by hardware immediately after it was set. After all STB slots are filled, TSNEXT remains at 1 until a STB slot is released. Note: This bit is fixed to 0 in TTCAN mode.</p>	R/W
b5	TSMODE	Transmit buffer Secondary operation Mode	<p>STB transmission mode 0: FIFO mode 1: Priority mode</p> <p>The FIFO mode is sent according to the order in which the data frames are written. The priority mode is automatically judged according to the ID. The lower 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.</p>	R/W
b4	TTTBM	TTCAN Transmit Buffer Mode	<p>TTCAN transmit buffer mode When TTEN=0, TTTBM is ignored. 0: Separate PTB and STB, behavior defined by TSMODE 1: Buffer slots selectable by TBPTR and TTPTR</p> <p>In TTCAN mode, with only support of reception timestamps, TTTBM=0 can be chosen. Then the transmit buffer acts as in event-driven mode and the behavior can be selected by TSMODE. Note: The TTTBM bit can only be set when the TBUF is empty.</p>	R/W
b3~b2	Reserved	-	The reset value must be maintained.	R/W
b1~b0	TSSTAT	Transmission Secondary Status	<p>STB status TTEN=0 or TTEN=1 & TTTBM=0 00: STB empty 01: STB is less than or equal to half full 10: STB is more than half full 11: STB full TTEN=1 and TTTBM=1 00: PTB and STB empty 01: PTB and STB are not empty and not full 10: Reserved 11: PTB and STB are full</p>	R

37.5.6 CAN Receive Control Register (CAN_RCTRL)

Offset address: 0xA3

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
SACK	ROM	ROV	RREL	RBALL	-		RSTAT[1:0]

Bit	Symbol	Bit name	Description	Read and write
B7	SACK	Self-Acknowledge	Self-Acknowledge 0: No self-ACK 1: self-ACK when LBME=1	R/W
b6	ROM	Receive buffer Overflow Mode	Receive buffer overflow mode 0: The oldest message will be overwritten 1: The new message will not be stored	R/W
b5	ROV	Receive buffer Overflow	Receive buffer overflow flag 0: No overflow 1: Overflow, at least one message is lost Cleared by writing RREL to 1.	R
b4	RREL	Receive buffer Release	Release the receive buffer 0: No release 1: Indicates that the receiving buffer has been read, and the RBUF register points to the next RBslot.	R/W
b3	RBALL	Receive Buffer stores All data frames	Receive buffer stores all data frames 0: Normal operation 1: Store all data including frames with errors.	R/W
b2	Reserved	-	The reset value must be maintained.	R/W
b1~b0	RSTAT	Receive buffer Status	Receive buffer status 00: RBUF empty 01: RBUF is not empty but less than AFWL programmed value 10: RBUF is greater than or equal to the AFWL programmed value but not full 11: full (hold this value on overflow)	R

37.5.7 CAN Receive and Transmit Interrupt Enable Register (CAN_RTIE)

Offset address: 0xA4

Reset value: 0xFE

B7	b6	b5	b4	b3	b2	b1	b0
RIE	ROIE	RFIE	RAFIE	TPIE	TSIE	EIE	TSFF

Bit	Symbol	Bit name	Description	Read and write
B7	RIE	Receive Interrupt Enable	Receive interrupt enable 0: Disable 1: Enable	R/W
b6	ROIE	RB Overrun Interrupt Enable	Receive overrun interrupt enable 0: Disable 1: Enable	R/W
b5	RFIE	RB Full Interrupt Enable	Receive buffer full interrupt enable 0: Disable 1: Enable	R/W
b4	RAFIE	RB Almost Full Interrupt Enable	Receive buffer almost full interrupt enable 0: Disable 1: Enable	R/W
b3	TPIE	Transmission Primary Interrupt Enable	PTB transmission interrupt enable 0: Disable 1: Enable	R/W
b2	TSIE	Transmission Secondary Interrupt Enable	STB transmission interrupt enable 0: Disable 1: Enable	R/W
b1	EIE	Error Interrupt Enable	Error interrupt enable 0: Disable 1: Enable	R/W
b0	TSFF	Transmit Buffer Full Flag	TTEN=0 or TTTBM=0: STB full flag (Transmit Secondary buffer Full Flag) 0: STB slots are not fully filled 1: STB slots are fully filled TTEN=1 and TTTBM=1: TB full flag (Transmit buffer Full Flag) 0: The transmit buffer selected by TBPTR is empty 1: The transmit buffer selected by TBPTR is filled	R

37.5.8 CAN Receive and Transmit Interrupt Flag Register (CAN_RTIF)

Offset address: 0xA5

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
RIF	ROIF	RFIF	RAFIF	TPIF	TSIF	EIF	AIF

Bit	Symbol	Bit name	Description	Read and write
B7	RIF	Receive Interrupt Flag	Receive interrupt flag 0: No frame has been received 1: Data or a remote frame has been received and is available in the receive buffer Write 1 to clear by application program.	R/W
b6	ROIF	RB Overrun Interrupt Flag	Receive buffer overrun flag 0: No RB is overwritten 1: At least one received message has been overwritten in the receive buffer Both ROIF and RFIF are set to 1 on overflow. Write 1 to clear by application program.	R/W
b5	RFIF	RB Full Interrupt Flag	Receive buffer full interrupt flag 0: Receive buffers are not full 1: Receive buffers are full Write 1 to clear by application program.	R/W
b4	RAFIF	RB Almost Full Interrupt Flag	Receive buffer almost full interrupt flag 0: The number of filled RB slots is less than the AFWL setting 1: The number of filled RB slots is greater than or equal to the AFWL setting value Write 1 to clear by application program.	R/W
b3	TPIF	Transmission Primary Interrupt Flag	PTB transmission interrupt flag 0: No PTB transmission completed 1: The requested PTB transmission is completed successfully Write 1 to clear by application program. Note: In TTCAN mode, TPIF is invalid, only the TSIF flag is valid	R/W
b2	TSIF	Transmission Secondary Interrupt Flag	STB transmission interrupt flag 0: No STB transmission completed 1: The requested STB transmission is completed successfully Write 1 to clear by application program. Note: In TTCAN mode, TPIF is invalid, only the TSIF flag is valid	R/W

Bit	Symbol	Bit name	Function	Read and write
b1	EIF	Error Interrupt Flag	<p>Error interrupt flag</p> <p>0: The BUSOFF bit has not changed, or the relative relationship between the value of the error counter and the set value of the ERROR warning limit has not changed.</p> <p>1: The BUSOFF bit changes, or the relative relationship between the value of the error counter and the set value of the ERROR warning limit changes. For example, the value of the error counter changes from less than the set value to greater than the set value, or from greater than the set value to less than the set value.</p> <p>Write 1 to 0 by application program.</p>	R/W
b0	AIF	Abort Interrupt Flag	<p>Abort interrupt flag</p> <p>0: No abort has been executed</p> <p>1: The message(s) requested by TPA or TSA to transmit have been aborted.</p> <p>Write 1 to clear by application program.</p>	R/W

37.5.9 CAN Error Interrupt Enable and Flag Register (CAN_ERRINT)

Offset address: 0xA6

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
EWARN	EPASS	EPIE	EPIF	ALIE	ALIF	BEIE	BEIF

Bit	Symbol	Bit name	Description	Read and write
b7	EWARN	Error Warning	Reached the EWL setting value 0: Both RECNT and TENCNT is less than the EWL setting value 1: RECNT or TENCNT is greater than or equal to the EWL setting value	R
b6	EPASS	Error Passive	Error passive mode active 0: The node is error active 1: The node is error passive	R
b5	EPIE	Error Passive Interrupt Enable	Error passive interrupt enable 0: Disable 1: Enable	R/W
b4	EPIF	Error Passive Interrupt Flag	Error passive interrupt flag 0: No change from error active to error passive or error passive to error active 1: A change from error active to error passive or error passive to error active Write 1 to clear by application program.	R/W
b3	ALIE	Arbitration Lost Interrupt Enable	Arbitration lost interrupt enable 0: Disable 1: Enable	R/W
b2	ALIF	Arbitration Lost Interrupt Flag	Arbitration lost interrupt flag 0: The node won the arbitration 1: The node lost the arbitration Write 1 to clear by application program.	R/W
b1	BEIE	Bus Error Interrupt Enable	Bus error interrupt enable 0: Disable 1: Enable	R/W
b0	BEIF	Bus Error Interrupt Flag	Bus error interrupt flag 0: No bus error 1: Bus error Write 1 to clear by application program.	R/W

37.5.10 CAN Slow Bit Timing Register (CAN_SBT)

Offset address: 0xA8

Reset value: 0x0102 0203

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
S_PRESC[7:0]								-	S_SJW[6:0]						
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	B0
-	S_SEG_2[6:0]								S_SEG_1[7:0]						

Bit	Symbol	Bit name	Description	Read and write
b31~b24	S_PRESC	Slow bit time Prescaler	Prescaler of time quanta(TQ) of slow bit time TQ=CAN clock period*(S_PRESC+1)	R/W
b23	Reserved	-	The reset value must be maintained.	R
b22~b16	S_SJW	Slow bit time SJW	Synchronization Jump Width(SJW) of slow bit time SJW=(S_SJW+1)*TQ	R/W
b15	Reserved	-	The reset value must be maintained.	R
b14~b8	S_SEG_2	Slow bit time Segment 2	Bit Timing Segment 2 Bit segment 2 time = (S_SEG_2+1)*TQ	R/W
b7~b0	S_SEG_1	Slow bit time Segment 1	Bit Timing Segment 1 Bit segment 1 time = (S_SEG_1+2)*TQ	R/W

37.5.11 CAN Error and Arbitration Lost Capture Register (CAN_EALCAP)

Offset address: 0xB0

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0		
		KOER[2:0]		ALC[4:0]					

Bit	Symbol	Bit name	Description	Read and write
b7~b5	KOER	Kind Of Error	Error code 000: no error 001: Bit error 010: Form error 011: Stuff error 100: ACK error 101: CRC error 110: Other errors 111: reserved KOER is updated with each new error. Therefore it stays untouched when frames are successfully transmitted or received.	R
b4~b0	ALC	Arbitration Lost Capture	Arbitration lost position ALC records the bit position in the frame where the arbitration has been lost.	R

37.5.12 CAN Warning Limits Register (CAN_LIMIT)

Offset address: 0xA7

Reset value: 0x1B

B7	b6	b5	b4	b3	b2	b1	b0		
		AFWL[3:0]		EWL[3:0]					

Bit	Symbol	Bit name	Description	Read and write
b7~b4	AFWL	Almost Full Warning Limit	Receive buffer almost full warning limit The set value range is 1~8. AFWL=0 is meaningless and is treated as AFWL=1.	R/W
b3~b0	EWL	Error Waring Limit	Programmable error warning limit Error Waring Limit=(EWL+1)*8. The value of EWL affects the EIF flag.	R/W

37.5.13 CAN Receive Error Counter Register (CAN_RECNT)

Offset address: 0xB2

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
RECNT[7:0]							

Bit	Symbol	Bit name	Description	Read and write
b7~b0	RECNT	Receive Error Count	Receive error count RECNT is incremented and decremented as defined in the CAN specification. RECNT does not overflow.	R

37.5.14 CAN Transmit Error Counter Register (CAN_TECNT)

Offset address: 0xB3

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
TEMCNT[7:0]							

Bit	Symbol	Bit name	Description	Read and write
b7~b0	TECNT	Transmit Error Count	Transmit error count TECNT is incremented and decremented as defined in the CAN specification. In case of the "bus off state" TECNT may overflow.	R

37.5.15 CAN Acceptance Filter Control Register (CAN_ACFCTRL)

Offset address: 0xB4

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	B0
-		SELMASK	-				ACFADR

Bit	Symbol	Bit name	Description	Read and write
b7~b6	Reserved	-	The reset value must be maintained.	R
b5	SELMASK	Select acceptance MASK	Select filter mask register 0: ACF points to the filter ID register 1: ACF points to the filter MASK register Select a specific filter register set via ACFADR	R/W
b4	Reserved	-	The reset value must be maintained.	R
b3~b0	ACFADR	Acceptance Filter Address	Acceptance filter address ACFADR points to a specific filter, and uses SELMASK to distinguish ID and MASK. 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: Points to ACF_6 0110: Points to ACF_7 0111: Points to ACF_8 1000: points to ACF_9 1001: Points to ACF_10 1010: points to ACF_11 1011: Points to ACF_12 1100: points to ACF_13 1101: Points to ACF_14 1110: Points to ACF_15 1111: Points to ACF_16	R/W

37.5.16 CAN Acceptance Filter Enable Register (CAN_ACFEN)

Offset address: 0xB6

Reset value: 0x0001

b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
AE_16	AE_15	AE_14	AE_13	AE_12	AE_11	AE_10	AE_9	AE_8	AE_7	AE_6	AE_5	AE_4	AE_3	AE_2	AE_1

Bit	Symbol	Bit name	Description	Read and write
b15	AE_16	ACF_16 Enable	Acceptance filter 16 enable0: Disable 1: Enable	R/W
b14	AE_15	ACF_15 Enable	Acceptance filter 15 enable0: Disable 1: Enable	R/W
b13	AE_14	ACF_14 Enable	Acceptance filter 14 enable0: Disable 1: Enable	R/W
b12	AE_13	ACF_13 Enable	Acceptance filter 13 enable0: Disable 1: Enable	R/W
b11	AE_12	ACF_12 Enable	Acceptance filter 12 enable0: Disable 1: Enable	R/W
b10	AE_11	ACF_11 Enable	Acceptance filter 11 enable0: Disable 1: Enable	R/W
b9	AE_10	ACF_10 Enable	Acceptance filter 10 enable0: Disable 1: Enable	R/W
b8	AE_9	ACF_9 Enable	Acceptance filter 9 enable0: Disable 1: Enable	R/W
B7	AE_8	ACF_8 Enable	Acceptance filter 8 enable0: Disable 1: Enable	R/W
b6	AE_7	ACF_7 Enable	Acceptance filter 7 enable0: Disable 1: Enable	R/W
b5	AE_6	ACF_6 Enable	Acceptance filter 6 enable0: Disable 1: Enable	R/W
b4	AE_5	ACF_5 Enable	Acceptance filter 5 enable0: Disable 1: Enable	R/W
b3	AE_4	ACF_4 Enable	Acceptance filter 4 enable0: Disable 1: Enable	R/W
b2	AE_3	ACF_3 Enable	Acceptance filter 3 enable0: Disable 1: Enable	R/W
b1	AE_2	ACF_2 Enable	Acceptance filter 2 enable0: Disable 1: Enable	R/W
b0	AE_1	ACF_1 Enable	Acceptance filter 1 enable0: Disable 1: Enable	R/W

37.5.17 CAN Acceptance CODE and MASK Register (CAN_ACF)

Offset address: 0xB8

Reset value: 0xFFFF XXXX

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	AIDEE	AIDE	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]															

Bit	Symbol	Bit name	Description	Read and write
b31	Reserved	-	The read value is indeterminate.	R
b30	AIDEE	Acceptance mask IDE bit check Enable	IDE bit check enable Only valid when SELMASK=1 0: Acceptance filter accepts both standard or extended frames. 1: Acceptance filter accepts either standard or extended as defined by AIDE.	R/W
b29	AIDE	Acceptance mask IDE bit value	IDE bit value 0: Acceptance filter accepts only standard frames. 1: Acceptance filter accepts only extended frames.	R/W
b28~b0	ACODE/AMASK	Acceptance CODE/ Acceptance MASK	Acceptance filter code Point to specific filters via ACFADR. When SELMASK=0, it means the CODE of the filter. Bit 10~bit 0 are used in the standard format, and bit 29~bit 0 are used in the extended format. Acceptance filter mask Point to specific filters via ACFADR. When SELMASK=1, it indicates the MASK of the filter. Bit 10~bit 0 are used in the standard format, and bit 29~bit 0 are used in the extended format.	R/W

37.5.18 TTCAN TB Slot Pointer Register(CAN_TBSLOT)

Offset address: 0xBE

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
TBE	TBF	-	-	-			TBPTR[2:0]

Bit	Symbol	Bit name	Description	Read and write
B7	TBE	set TB to Empty	Marks the transmit buffer as empty 0: No action 1: SLOT selected by TBPTR is marked as empty TBE is automatically reset to 0 as soon as the slot is marked as empty and TSFF=0. If a transmission from this slot is active, then TBE stays set as long as either the transmission completes or after a transmission error or arbitration loss the transmission is not active any more. If both TBF and TBE are set, then TBE wins.	R/W
b6	TBF	set TB slot to Filled	Marks the transmit buffer as filled 0: No action 1: The slot selected by TBPTR is marked as filled TBE is automatically reset to 0 when SLOT is marked as filled and TSFF=1.	R/W
b5~b3	Reserved	-	The reset value must be maintained.	R/W
b2~b0	TBPTR	TB slot Pointer	Pointer to a transmit buffer slot 000: Pointer to PTB 001: Pointer to STB slot 1 010: Pointer to STB slot 2 011: Pointer to STB slot 3 other: reserved The message slot pointed to by TBPTR is readable / writable using the TBUF registers. Write access is only possible if TSFF=0. Setting TBF to 1 marks the selected slot as filled and setting TBE to 1 marks the selected slot as empty. In TTCAN mode, the TBSEL and TSNEXT registers have no effect. Note: This bit can only be written when TSFF=0.	R/W

37.5.19 TTCAN Time Trigger Configuration Register (CAN_TTCFG)

Offset address: 0xBF

Reset value: 0x90

B7	b6	b5	b4	b3	b2	b1	b0
WTIE	WTIF	TEIF	TTIE	TTIF	T_PRESC[1:0]		TTEN

Bit	Symbol	Bit name	Description	Read and write
b7	WTIE	Watch Trigger Interrupt Enable	Watch trigger interrupt enable 0: Disable 1: Enable	R/W
b6	WTIF	Watch Trigger Interrupt Flag	Watch trigger interrupt flag WTIF will be set if the cycle count reaches the limited defined by TT_WTRIG and WTIE is set. Write 1 to clear by application program.	R/W
b5	TEIF	Trigger Error Interrupt Flag	Trigger error interrupt flag TEIF is set when the set value of TT_TTIG is less than the actual CYCLE_TIME. Write 1 to clear by application program.	R/W
b4	TTIE	Time Trigger Interrupt Enable	Time trigger interrupt enable 0: Disable 1: Enable	R/W
b3	TTIF	Time Trigger Interrupt Flag	Time trigger interrupt flag TTIF will be set if TTIE is set and the cycle time is equal to the trigger time TT_TRIG. TTIF will be set only once. If TT_TRIG gets not updated, then TTIF will be not set again in the next basic cycle. Write 1 to 0 by application program.	R/W
b2~b1	T_PRESC	TTCAN Timer Prescaler	TTCAN time prescaler 00: Divide by 1 of the bit time set by the SBT register 01: Divide by 2 of the bit time set by the SBT register 10: Divide by 4 of the bit time set by the SBT register 11: Divide by 8 of the bit time set by the SBT register Note: T_PRESC can only be modified if TTEN=0, but it is possible to modify T_PRESC and set TTEN simultaneously with one write access.	R/W
b0	TTEN	Time Trigger Enable	TTCAN enable 0: Disable 1: Enable TTCAN, the timer is running.	R/W

37.5.20 TTCAN Reference Message Register (CAN_REF_MSG)

Offset address: 0xC0

Reset value: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
REF_I DE	-	REF_ID[28:16]														
b15	b14	b13	b12	b11	b10	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0	
REF_ID[15:0]																

Bit	Symbol	Bit name	Description	Read and write
b31	REF_IDE	Reference message IDE bit	Reference message IDE bit0: Standard format 1: Extended format	R/W
b30~b29	Reserved	-	The read value is indeterminate.	R/W
b28~b0	REF_ID	Reference message Identifier	.Reference message ID 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 applies to both transmission and reception. After the reference message is detected, the Sync_Mark of the current frame becomes Ref_Mark. REF_ID[2:0] is fixed at 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, it needs to wait for 6 CAN clock cycles to complete the transfer of REF_MSG to the CAN clock domain.	R/W

37.5.21 TTCAN Trigger Configuration Register (CAN_TRG_CFG)

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]			

Bit	Symbol	Bit name	Description	Read and write
b15~b12	TEW	Transmit Enable Window	Transmit Enable Window , For a single shot transmit trigger, a window of TEW+1 cycle time can be set, and the transmission can only be started in this window.	R/W
b11	Reserved	-	The reset value must be maintained.	R
b10~b8	TTYPE	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 other: No action The trigger time is set by the TT_TRIG register, and the 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 000: Points to PTB 001: Points to STB slot 1 010: Points to STB slot 2 011: Points to STB slot 3 other: reserved If TTPTR points to an empty slot, then TEIF will be set at the moment, when the trigger time is reached.	R/W

37.5.22 TTCAN Trigger Time Register (CAN_TT_TRIG)

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]															

Bit	Symbol	Bit name	Description	Read and write
b15~b0	TT_TRIG	Trigger Time	Trigger time Used to specify the cycle time of the trigger. For a transmission trigger the earliest point of transmission of the SOF of the appropriate frame will be TT_TRIG+1. A write access to the high byte starts a data transfer of the trigger definition to the CAN clock domain and activates the trigger. Therefore, if the BYTE operation is performed, the low byte needs to be written first and then the high byte.	R/W

37.5.23 TTCAN Watch Trigger Time Register (CAN_TT_WTRIG)

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]															

Bit	Symbol	Bit name	Description	Read and write
b15~b0	TT_WTRIG	Watch Trigger Time	Watch trigger time Cycle time for a watch trigger. A write access to the high byte starts a data transfer of the trigger definition to the CAN clock domain. Therefore, if the BYTE operation is performed, the low byte needs to be written first and then the high byte.	R/W

37.6 Precautions for use

37.6.1 CAN bus anti-interference measures

CAN bus is widely used in automobile, industrial control, and other industries. If the electromagnetic environment of the CAN application site is relatively harsh, there are factors such as circuit imbalance, space electromagnetic field, power grid incoming lines, etc., which will cause the CAN bus to generate a lot of communication noise due to radiation and conduction interference. , resulting in the increase of bus error frames, frequent retransmissions, and the failure of correct data to arrive in time, which seriously affects the quality of data communication. Therefore, in practical applications, we should focus on eliminating noise interference and ensuring the stable operation of the CAN bus network.

The following are several types of commonly used CAN bus anti-interference measures (including but not limited to)

1. Increase the electrical isolation of CAN bus interface
2. The GND of the transceiver needs to be connected to a common GND
3. Use shielded twisted pair cables and ground them properly
4. Improve the twisted pair degree of CAN transmission line
5. Add a signal protector
6. Improve network topology
7. Application layer software anti-interference mechanism

37.6.2 CAN Controller Noise Constraints

In the CAN bus network, it should be ensured that the bit time of communication meets the requirements of the standard protocol. If the noise that does not meet the bit time width is introduced, it may cause abnormal actions of the CAN controller.

38 SDIO Controller (SDIOC)

38.1 Introduction

SDIOC provides an SD host interface and an MMC host interface for communicating with SD cards supporting SD2.0 protocol, SDIO devices and MMC devices supporting eMMC4.2 protocol. This product has 2 SDIO controllers and can communicate with 2 SD/MMC/SDIO devices at the same time.

SDIOC features are as follows:

- Support SDSC, SDHC, SDXC format SD card and SDIO device
- Support one-line (1bit) and four-line (4bit) SD bus
- Support one-line (1bit), four-line (4bit) and eight-line (8bit) MMC bus
- SD clock up to 50MHz
- With card recognition and hardware write protection

38.2 Functional description

38.2.1 Port assignment

Table 38-1 Port allocation table

Port name	IO	Function
SDIOx_CK(x=1~2)	O	SD clock output signal
SDIOx_CMD(x=1~2)	I/O	SD command and response 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 identification status signal
SDIOx_WP(x=1~2)	I	SD write protection status signal

The above ports are based on SD2.0 protocol and eMMC4.51 protocol, SD clock output signal (SDIOx_CK(x=1~2)) is used to output SD clock when SDIOC communicates with SD/MMC device; SD command and response signal (SDIOx_CMD(x=1~2)) is used to output SD/MMC commands to the device and receive the response information sent back by the device; SD data signal (SDIOx_Dy(x=1~2) (y=0~7)) is used for SDIOC and the device Send and receive data during the communication process. When using one-line (1bit) communication, only SDIOx_D0 (x=1~2) is valid, SDIOx_Dy (x=1~2) (y=1~7) keeps high level, using four In the case of line type (4bit) communication, SDIOx_Dy(x=1~2) (y=0~3) is valid, SDIOx_Dy(x=1~2) (y=4~7) keeps high level, the eight-wire type (8bit) communication, SDIOx_Dy (x=1~2) (y=0~7) is valid, and eight-wire (8bit) communication is limited to MMC device communication.

38.2.2 Basic access

The user starts the SDIOC to communicate with the off-chip SD/MMC device by reading and writing the SDIOC register. Since writing the command register will trigger the SDIOC's send command action, the writing command register must be performed last. Before this, the user needs to set the transmission mode through the transmission mode register (TRANSMODE), set the command parameters through the parameter registers (ARG0, ARG1), and set the correct command number (command) when setting the command register (CMD). index, type, response type, etc. 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 whether the response information is received, whether there is any error information, etc. When a command is executed, the user can get the response of the command by reading the response register (RESP0~7).

38.2.3 Data transmission

The data buffer register of SDIOC is used for data exchange between host devices such as CPU/DMA and SD devices. SDIOC has built-in FIFO to speed up data exchange. When the command contains data to send, the user writes the data into the data buffer registers (BUF0, BUF1) in turn, these data will be buffered in the FIFO of SDIOC first, when the number of written data reaches the set value of the data length register (BLKSIZE) Or 512 bytes, SDIOC will send data through SDIOx_Dy(x=1~2) (y=0~7). During this period, SDIOC can continue to write data to the data buffer register through the FIFO. Similarly, when the command includes data reception, SDIOC receives data through SDIOx_Dy(x=1~2) (y=0~7), and buffers the data in the buffer (BUFFER) of SDIOC, and the user reads the data buffer by reading Registers (BUF0, BUF1) get data. Please refer to SD and MMC protocol for data format.

38.2.4 SD clock

The SD clock (SDIOx_CK(x=1~2)) is generated by the host and output to the device for communication between the two. The SD clock is generated by dividing the operation clock (PCLK1) of the SDIOC module through a frequency divider. The frequency division coefficient of the frequency divider is set by the clock control register (CLKCON). According to the requirements of the SD2.0 protocol, the fastest in the data transfer mode (data transfer mode) should be 50MHz, so the user needs to set it according to the frequency of PCLK1 Reasonable frequency division factor.

38.2.5 Interrupts and DMA Start Requests

38.2.5.1 SD interrupt

SDIOC provides two types of interrupts, normal interrupts and error interrupts. Ordinary interrupts are interrupts that occur when various events are generated during the communication between SDIOC and SD/MMC cards. Error interrupts are interrupts that occur when various errors occur during SDIOC operation communication. Normal interrupt and error interrupt have their own

interrupt status register, interrupt status enable register, and interrupt signal enable register. The interrupt status register is used to indicate the cause of the interrupt. 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 needs to be valid when the enable bit of the interrupt status enable register is enabled. The interrupt signal enable register is used to allow each interrupt factor to apply for an interrupt to the CPU.

38.2.5.2 SDIO interrupt

The SDIO device can send a card interrupt request to the host when the transfer is idle. When using SDIO interrupt, the card interrupt (CINTEN) of the interrupt status enable register (NORINTSTEN) needs to be enabled. If an interrupt needs to be applied to the CPU, the card interrupt (CINTEN) of the interrupt signal enable register (NORINTSGEN) needs to be enabled. The SDIO device applies for an interrupt to the host by pulling the SDIO_x_D1 ($x=1\sim 2$) data line low. After SDIOC detects that SDIO_x_D1 ($x=1\sim 2$) is pulled low, it sets the card interrupt (CINT) of the interrupt status register (NORINTST), and applies for an interrupt to the CPU according to the settings. The SDIO device will release SDIO_x_D1 ($x=1\sim 2$) after judging that the interrupt processing is completed.

38.2.5.3 DMA request

The read and write data of SDIOC in communication can be done through DMA. When using DMA transfer to write data to the device, set the start source of one channel of DMA to the write request of SDIOC, and then set the transfer target address of DMA to the data buffer register (BUF0, BUF1) and a fixed address. After the SDIOC sends the write data command, if the data FIFO is empty at this time, it will send a write request signal to the DMA, and start the DMA to write data to the data buffer registers (BUF0, BUF1), and these data will enter the data FIFO in turn. When the data in the FIFO reaches the set value of the data length register (BLKSIZE) or 512 bytes, the SDIOC will send the data through SDIO_x_Dy($x=1\sim 2$) ($y=0\sim 7$). If it is a multi-block write command, SDIOC will continue to send the start request to write data to the DMA at the same time. When using DMA transfer to read data from the device, set the start source of another 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 a fixed address. After SDIOC sends the read data command, the device will send data through SDIO_x_Dy($x=1\sim 2$) ($y=0\sim 7$), and the data FIFO of SDIOC starts to receive data. When the data in the FIFO reaches the data length of the data length register (BLKSIZE) When setting the value or 512 bytes, SDIOC sends a read request signal to DMA, and starts DMA to read data from the data buffer register (BUF0, BUF1). In the case of a multi-block read command, SDIOC will continue reading the next block of data from the device at the same time.

38.2.5.4 Card insertion and removal

The insertion and removal of SD/MMC/SDIO devices are identified by the SDIO_x_CD ($x=1\sim 2$) signal lines. When there is no device in the card slot, the card slot will pull up the SDIO_x_CD ($x=1\sim 2$)

signal through the resistor. When a device is inserted, the SDIOx_CD ($x=1\sim 2$) signal will be pulled low, and will be high again after the device is removed. SDIOC judges whether there is a device by the level of SDIOx_CD ($x=1\sim 2$). The user can determine if a device is plugged in by reading the CDPL bit in the host status register (PSTAT). SDIOC can generate corresponding interrupts when the device is inserted and removed, and is enabled through the interrupt status enable register (NORINTSTEN) and the interrupt signal enable register (NORINTSGEN).

38.2.6 Host and device initialization

38.2.6.1 Host initialization

SDIOC needs to be initialized first when using it. SDIOC initialization steps are as follows:

1. Read the host status register (PSTAT) to query clock status and device insertion status
2. Configure Power Control Register (PWRCON), enable SDIOC
3. Configure the clock control register (CLKCON) to enable SDIOC to output the SD clock, and configure the SD clock frequency division according to the frequency of PCLK1 to ensure that the SD clock frequency does not exceed 400KHz in card identification mode
4. Configure the host control register (HOSTCON), select the one-line (1bit) mode and disable the high speed mode (high speed mode)
5. Configure the timeout control register (TOUTCON) according to the device characteristics, so that the host ends the communication and reports an error when the communication times out
6. Configure the normal and error interrupt status enable registers, interrupt signal enable registers to enable the SDIOC to interrupt when needed and set the flag bit to start.

38.2.6.2 SD card initialization

After completing the SDIOC initial configuration, if the SD card is connected. It needs to be initialized according to the SD protocol. The initialization sequence is as follows:

1. Card reset, send reset command CMD0 to the device, CMD0 has no response information (response), at this time the card will enter the card identification mode
2. Confirm the working status of the card (operation condition), send CMD8 to the device and wait for the response message (response)
3. Send the initialization command ACMD41 (first send CMD55, then send CMD41), and judge whether the device has completed the initialization according to the response information, otherwise continue to send ACMD41 until the initialization is completed
4. Send CMD2 to get the CID information of the card
5. Send CMD3 to get the RCA information of the card

At this point the card will enter the data transfer mode and the initialization is complete.

38.2.6.3 MMC card initialization

After completing the SDIOC initial configuration, if the MMC card is connected. It needs to be initialized according to the MMC protocol. The initialization sequence is as follows:

1. Card reset, send reset command CMD0 to the device, CMD0 has no response information (response), at this time the card will enter the card identification mode
2. Confirm the operation condition of the card, send CMD1 to the device and wait to receive the response message (response), and judge whether the device has completed the initialization according to the response information, otherwise continue to send CMD1 until the initialization is completed.
3. Send CMD2 to get the CID information of the card
4. Send CMD3 to get the RCA information of the card

At this point the card will enter the data transfer mode and the initialization is complete.

38.2.6.4 SDIO initialization

After completing the SDIOC initial configuration, if an SDIO device is connected. It needs to be initialized according to the SDIO protocol. The initialization sequence is as follows:

1. Device reset, send reset command CMD0 to the device, CMD0 has no response information (response)
2. Confirm the working status of the SDIO device (operation condition), send CMD5 to the device and wait for the response message (response)
3. Send CMD3 to get the RCA information of the device, the initialization is complete

38.2.7 SD/MMC single block (single block) read and write

After the SD/MMC card enters the data transfer mode, the SDIOC can access the data of the SD/MMC card through read and write commands. The order of reading and writing a single data block (block) is as follows:

1. Configure the clock control register (CLKCON) to enable SDIOC to output the SD clock, and configure the SD clock frequency division according to the frequency of PCLK1 so that the SD clock frequency does not exceed the maximum clock speed in the default speed mode (default speed mode, fpp<=25MHz).
2. Send CMD7, SD/MMC card will enter data transfer state.
3. If necessary, for SD cards, 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 MMC cards, you can configure the host control register (HOSTCON) to set the host bus width. , and rewrite the Ext_CSD register of the MMC card through the CMD6 (SWITCH) command to set the bus width (1bit, 4bit or 8bit).
4. If necessary, for SD card, the length of the data block can be set by configuring the data

length register (BLKSIZE), and the size (number of bytes) of the data block can be set through the CMD16 command. The configuration range is 1~512 bytes. For SDHC/SDXC and MMC card 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 (high speed mode), and send CMD6 to switch the SD card to high speed mode (high speed mode, fpp<=50MHz), switch After success, SDIOx_CK (x=1~2) can be set up to 50MHz
6. When writing data, first configure the transfer mode register (TRANSMODE), set the transfer mode to write, single block transfer. Send a single block write command CMD24. If the CPU is used to write data, after confirming that the BWR bit of the interrupt status register (NORINTST) is 1, write data to the data buffer register (BUF0, BUF1), such as using DMA to write data, wait for DMA After the transmission is completed, SDIOC will send data through the data signal SDIOx_Dy (x=1~2). 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 process, the corresponding error flag will be set and an interrupt request will be generated.
7. When reading data, first configure the transfer mode register (TRANSMODE), set the transfer mode to read, single block transfer. Send a single block read command CMD17, SDIOC will receive data through the data signal SDIOx_Dy (x=1~2). If using the CPU to read data, after confirming that the BRR bit of the interrupt status register (NORINTST) is 1, read the data from the data buffer registers (BUF0, BUF1), if using DMA to read the data, wait for the DMA transfer to complete., 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 process, the corresponding error flag will be set and an interrupt request will be generated.

38.2.8 SD/MMC multi-block read and write

The sequence of multi-block read and write and single-block read and write is as follows:

1. Configure the clock control register (CLKCON) to enable SDIOC to output the SD clock, and configure the SD clock frequency division according to the frequency of PCLK1 so that the SD clock frequency does not exceed the maximum clock speed in the default speed mode (default speed mode, fpp<=25MHz).
2. Send CMD7, SD/MMC card will enter data transfer state.
3. If necessary, for SD cards, 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 MMC cards, you can configure the host control register (HOSTCON) to set the host bus width. , and rewrite the Ext_CSD register of the MMC card through the CMD6 (SWITCH) command to set the bus width (1bit, 4bit or 8bit).

4. If necessary, for SD card, the length of the data block can be set by configuring the data length register (BLKSIZE), and the size (number of bytes) of the data block can be set through the CMD16 command. The configuration range is 1~512 bytes. For SDHC/SDXC and MMC card 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 the allow data block count enable is set, the number of data blocks to be transmitted needs to be set in the data block count register (BLKCNT), and multi-block transfer cannot be started without setting the data block count register (BLKCNT). If the disable data block count enable is set, the data block count register (BLKCNT) does not need to be set, and an unlimited number of multi-block transfers can be performed at this time. When the host decides to end the transfer, it needs to be completed after the last data block transfer is completed. Send a CMD12 to the device to inform the device that the data transfer is over.
6. If necessary, for the 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 (high speed mode, fpp<=50MHz), switch After success, SDIOx_CK (x=1~2) can be set up to 50MHz
7. When writing data, first send the multi-block write command CMD25. If the CPU is used to write data, after confirming that the BWR bit of the interrupt status register (NORINTST) is 1, write data to the data buffer register (BUF0, BUF1). During the sending process, BWR It will remain 0 and reset to 1 after sending, and the user can write the data of the second block at this time. If DMA is used to write data, wait for the DMA transfer to complete, and SDIOC will send data through the data signal SDIOx_Dy (x=1~2). 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 transmission process, the corresponding error flag will be set and an interrupt request will be generated. After all data is sent, if automatic sending CMD12 is set, SDIOC will automatically send a CMD12 to end the transmission, otherwise it needs to manually send CMD12 to the device to inform the device that the data transmission is over.
8. When reading data, send multi-block read command CMD18, SDIOC will receive data through data signal SDIOx_Dy (x=1~2). If the CPU is used to read data, after confirming that the BRR bit of the interrupt status register (NORINTST) is 1, read the data from the data buffer register (BUF0, BUF1), the BRR will remain 0 during the receiving process, and reset to 1 after the transmission. The data of the second block can be read at this time. If DMA is used to read data, wait for the DMA transfer to complete. After the read 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 transfer process, the corresponding error flag will be set and an interrupt will be generated. ask. After all data

is received, if automatic sending of CMD12 is set, SDIOC will automatically send a CMD12 to end the transmission, otherwise it needs to manually send CMD12 to the device to inform the device that the data transmission is over.

38.2.9 Transfer abort, suspend and resume

When a multi-block transfer is performed, it can be aborted or suspended through software control. The abort operation can be performed regardless of whether the number of transfer data blocks is set or not. Termination operations are divided into asynchronous termination and synchronous termination. The asynchronous termination operation requires that the transmission is terminated by sending CMD12 to the command register (CMD) while the transmission is in progress. At this time, the transmission will be terminated immediately. For the write operation, the SD/MMC card will discard the current data block data and enter the programming mode (program), and write the previously received data block into the FLASH. For read operations, SD/MMC will stop transferring data. Synchronous termination means that the data block interval register (BLKGAP) is set to stop the transmission at the data block interval during the transmission. After the setting is completed, the transmission will stop after the current data block transmission ends. At this point, CMD12 needs to be sent to end the transfer.

When performing a suspend operation, first stop the transmission at the data block interval by setting the SABGR bit of the data block interval register (BLKGAP), and enable the read wait function. Once set, the transfer will stop when the current block transfer is complete. At this point the transfer is suspended by sending CMD52 to the write command register (CMD). After sending the suspend command, it is necessary to continue to read the BS bit (bus status) of the CCCR register of SDIO through CMD52. If BS=0, it means that the transmission has been suspended, and the SD bus becomes idle. After the transfer is suspended, the host can perform operations on other functions of SDIO. However, if you want to perform a resume operation later, you need to back up the SDIOC register (offset address 00h~0Dh) after suspending, and restore these register settings after performing other operations. When performing a resume operation, first clear the block interval register (BLKGAP) to stop the transfer at the block interval, and then resume the transfer by sending CMD52 to the write command register (CMD). After sending the recovery command, it is necessary to continue to read the DF bit (data flag) of the CCCR register of SDIO through CMD52. If DF=1, it means that there is data to continue to transmit after the recovery is performed. If DF=0, it means that no data needs to be transmitted. At this time The software reset register (SFTRST) should be written to reset the data lines.

Note:

- Suspend (suspend) and resume (resume) operations require SDIO devices and combo card devices to support such operations and read wait (read wait) operations. Access to the CIA

area (common I/O area) of SDIO does not support suspend and resume operations, only the read wait function can be used.

38.2.10 Read wait

Read wait (read wait) allows the host to insert the CMD52 in a continuous data transfer to access other functions of the SDIO device. When executing read wait (read wait), firstly by setting the SABGR bit of the data block interval register (BLKGAP), the transmission is stopped at the data block interval, and the read wait (read wait) function is enabled. After the setting is completed, the host will pull the SDIOx_D2 ($x=1\sim 2$) data line low after the current data block transmission is completed. At this point the transfer will be suspended and the host can plug in the CMD52 at this point to access other functions that do not require data transfer. When the read wait function needs to be ended, the transfer can be resumed by setting CR of the block interval register (BLKGAP) and clearing SABGR.

Note:

- Read wait (read wait) requires the support of SDIO devices and combo card devices, and needs to be performed under the four-wire bus transmission.

38.2.11 Wakeup

When not working for a long time, the system can be transferred to a low-power state to reduce power consumption. In the low-power state, the system can be woken up to continue working through SD/MMC/SDIO device insertion/removal and card interrupt. When using the wake-up function, you need to enable the corresponding enable bits in the interrupt status enable register (NORINTSTEN) and interrupt signal enable register (NORINTSGEN) for insert/removal or card interrupt. At the same time, enable the corresponding wake-up function of SDIOx_CD($x=1\sim 2$)/SDIOx_D1($x=1\sim 2$) port. After the configuration is complete, the system can be put into a low-power mode. When insert/removal and card interrupt occur, the wake-up function of SDIOx_CD($x=1\sim 2$)/SDIOx_D1($x=1\sim 2$) port will wake up the system and insert (insert)/removal (removal) and card interrupt (card interrupt) interrupt.

Use insert to wake up the stop mode process:

1. Configure the SDIOC port, select SDIOx_CD ($x=1\sim 2$) as PA10
2. Configure the PCR register of PA10, select IRQ10 to be valid
3. Configure the PWRC3 register of the power control module to make the CPU enter stop mode after executing the WFI command
4. Configure the WUPEN register of the interrupt control module to enable the wake-up function of IRQ10
5. Configure the EIRQCR10 register of the interrupt control module and select the falling edge trigger

6. Configure PWRC0 to enable SDIOC
7. Configure the interrupt status enable register (NORINTSTEN) and the interrupt signal enable register (NORINTSGEN) to insert (insert) the corresponding enable bit 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 plugged in, PA10 will be pulled low, wake up the CPU through IRQ10, and enter the interrupt subroutine according to the SD interrupt request
11. Clear the insert state in the interrupt status register (NORINTST), exit the interrupt subroutine, and perform subsequent operations

Use insert (insert) to wake up the power down mode (power down mode) process:

1. Configure the SDIOC port, select SDIOx_CD (x=1~2) as PA10
2. Configure the PCR register of PA10, select IRQ10 to be valid
3. Configure the PWRC3 and PWRC0 registers of the power control module to make the CPU enter the 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 WUPEN register of the interrupt control module to enable the wake-up function of IRQ10
6. Configure the EIRQCR10 register of the interrupt control module and select the falling edge trigger
7. Configure PWRC0 to enable SDIOC
8. Configure the interrupt status enable register (NORINTSTEN) and the interrupt signal enable register (NORINTSGEN) to insert (insert) the corresponding enable bit 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, wake up the system through IRQ10 and power on again

38.3 Register description

The SDIOC module is designed according to the SD Host Controller Standard Specification, so the registers are also the same as the standard description, and the unused addresses and bits are changed to reserved bits (Reserved).

Table 38-2 Register list

English name	Chinese name	Abbreviation	Offset address
Block Size	data block length	BLKSIZE	0x04
Block Count	data block count	BLKCNT	0x06
Argument0	parameter 0	ARG0	0x08
Argument1	parameter 1	ARG1	0x0A
Transfer Mode	transfer mode	TRANSMODE	0x0C
Command	Order	CMD	0x0E
Response0	Reply 0	RESP0	0x10
Response1	Reply 1	RESP1	0x12
Response2	Answer 2	RESP2	0x14
Response3	Answer 3	RESP3	0x16
Response4	Answer 4	RESP4	0x18
Response5	Answer 5	RESP5	0x1A
Response6	Answer 6	RESP6	0x1C
Response7	Answer 7	RESP7	0x1E
Buffer Data Port0	data buffer 0	BUFO	0x20
Buffer Data Port1	data buffer 1	BUF1	0x22
Present State	Host Status	PSTAT	0x24
Host Control	host control	HOSTCON	0x28
Power Control	power control	PWRCON	0x29
Block Gap Control	block interval control	BLKGPCON	0x2A
Clock Control	clock control	CLKCON	0x2C
Timeout Control	Timeout control	TOOUTCON	0x2E
Software Reset	Software reset	SFTRST	0x2F
Normal Interrupt Status	normal interrupt status	NORINTST	0x30
Error Interrupt Status	error interrupt status	ERRINTST	0x32
Normal Interrupt Status Enable	Normal interrupt status enable	NORINTSTEN	0x34
Error Interrupt Status Enable	Error Interrupt Status Enable	ERRINTSTEN	0x36
Normal Interrupt Signal Enable	Normal interrupt signal enable	NORINTSGEN	0x38
Error Interrupt Signal Enable	Error interrupt signal enable	ERRINTSGEN	0x3A
Auto CMD Error Status	Autocommand error status	ATCERRST	0x3C
Force Event for Auto CMD Error Status	Force autocommand error state control	FEA	0x50
Force Event for Error Interrupt Status	Forced error state control	FEE	0x52

In addition, SDIOC1 and SDIOC2 share an MMC mode enable register for the controller to switch between SD and MMC modes

English name	Chinese name	Abbreviation	Address
MMC Enable Register	MMC Mode Enable Register	MMCR	0x40055404

38.3.1 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]											
<hr/>															
Bit	Marking	Place name				Function				Read and write					
b15~b12	Reserved	-				Read 0 when reading, write 0 when writing				R/W					
b11~0	TBS	data block length				Set the transfer block size (Transfer Block Size), the length of the transfer data block is in bytes, and the setting range is 1~512				R/W					

38.3.2 Block Count Register (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]														<hr/>			
<hr/>																	
Bit	Marking	Place name				Function				Read and write							
b15~b0	BC	data block count				Set the number of data blocks to be transmitted (Block count). Setting this register needs to be performed when the transmission is stopped, and the data block count enable bit (BCE) of the transmission mode register is required to be valid.				R/W							

38.3.3 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]														<hr/>			
<hr/>																	
Bit	Marking	Place name				Function				Read and write							
b15~b0	ARG[15:0]	Command parameters				Set the parameters contained in the current sending command, this register is the lower 16 bits of the parameter				R/W							

38.3.4 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]															

Bit	Marking	Place name	Function	Read and write
b15~b0	ARG[15:0]	Command parameters	Set the parameters contained in the current sending command, this register is the upper 16 bits of the parameter	R/W

38.3.5 Transfer 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	

Bit	Marking	Place name	Function	Read and write
b15~b6	Reserved	-	Read 0 when reading, write 0 when writing	R/W
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 transfer direction	0: write operation (host sends data) 1: read operation (host receives data)	R/W
b3~b2	ATCEN	Autocommand enable	00: Do not send autocommands 01: Automatically send CMD12 after multi-block transmission 10: Disable setting 11: Disable setting	R/W
b1	BCE	block count enable	0: Disable data block count enable 1: Enable data block count enable	R/W
b0	Reserved	-	Read 0 when reading, write 0 when writing	R/W

38.3.6 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]

Bit	Marking	Place name	Function	Read and write
b15~b14	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b13~b8	IDX	command number	number of command sent	R/W
b7~b6	TYP	Command type	00: Normal command 01: Suspend (suspend) command 10: Resume command 11: Abort (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 the 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 check code in the response 1: Check the CRC check code in the response	R/W
b2	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b1~b0	RESTYP	Reply type	00: No response to this command 01: The command has a reply with a length of 136 bits 10: The command has a reply with a length of 48 bits 11: The command has a reply with a length of 48 bits and a busy state	R/W

38.3.7 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]															

Bit	Marking	Place name	Function	Read and write
b15~b0	RESP0[15:0]	response message	15~0 bits of the response message	R

38.3.8 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]															

Bit	Marking	Place name	Function	Read and write
b15~b0	RESP1[15:0]	response message	31~16 bits of the response message	R

38.3.9 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]															

Bit	Marking	Place name	Function	Read and write
b15~b0	RESP2[15:0]	response message	47~32 bits of the response message	R

38.3.10 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]															

Bit	Marking	Place name	Function	Read and write
b15~b0	RESP3[15:0]	response message	63~48 bits of the response message	R

38.3.11 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[15:0]															

Bit	Marking	Place name	Function	Read and write
b15~b0	RESP4[15:0]	response message	79~64 bits of the response message	R

38.3.12 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	Marking	Place name	Function	Read and write
b15~b0	RESP5[15:0]	response message	95~80 bits of the response message	R

38.3.13 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[15:0]															

Bit	Marking	Place name	Function	Read and write
b15~b0	RESP6[15:0]	response message	111~96 bits of the response message	R

38.3.14 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[15:0]															

Bit	Marking	Place name	Function	Read and write
b15~b0	RESP7[15:0]	response message	127~112 bits of the response message	R

38.3.15 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]															

Bit	Marking	Place name	Function	Read and write
b15~b0	BUF0	data buffer	Write transmit data and read receive data, this register is the lower 16 bits of data	R/W

38.3.16 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]															

Bit	Marking	Place name	Function	Read and write
b15~b0	BUF1	data buffer	Write transmit data and read receive data, this register is the upper 16 bits of the data	R/W

38.3.17 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			

Bit	Marking	Place name	Function	Read and write
b31~b25	Reserved	-	Read 0 when reading, write 0 when writing	R/W
B24	CMDL	Command Line Status	Status of the command line (SDIOx_CMD(x=1~2))	R
B23~b20	DATL	data line status	Status of data line (SDIOx_Dy(x=1~2) (y=0~3))	R
B19	WPL	write-protect line status	Status of write protection line (SDIOx_WP(x=1~2))	R
B18	CDL	Card Identification Line Status	Status of the card identification line (SDIOx_CD(x=1~2))	R
B17	CSS	Device steady state	0: The status of the card identification line is unstable 1: The status of the card identification line is stable, the device is inserted or not inserted	R
B16	CIN	device plugged in	0: No device plugged in 1: There is a device plugged in	R
B15~b12	Reserved	-	Read 0 when reading, write 0 when writing	R/W
B11	BRE	data buffer full	0: The data buffer does not have enough data to read 1: The data buffer has enough data to read	R
B10	BWE	data buffer empty	0: Data buffer can write data 1: 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 status	0: No write operation in progress 1: There is an ongoing write operation	R
B7~b3	Reserved	-	Read 0 when reading, write 0 when writing	R/W
B2	DA	Data line transmission status	0: The data line is idle 1: The data line is transmitting data	R
B1	CID	With Data Command Suppression	0: Allow to send commands with data 1: Disable sending commands with data	R
B0	CIC	command suppression	0: Allow sending commands 1: Disable sending commands	R

38.3.18 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

Bit	Marking	Place name	Function	Read and write
B7	CDSS	Card identification line selection	0: Select the real SDIOx_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 level (with device inserted) 1: Card identification test signal is high level (no device inserted)	R/W
b5	EXDW	Extended data bit width	0: The bit width of the data line uses the setting of the DW bit 1: The bit width of the data line is 8 bits (8bit)	R/W
b4~b3	Reserved	-	Read 0 when reading, write 0 when writing	R/W
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: The bit width of the data line is 1 bit (1bit) 1: The bit width of the data line is 4 bits (4bit)	R/W
b0	Reserved	-	Read 0 when reading, write 0 when writing	R/W

38.3.19 Power Control Register (PWRCON)

Offset address: 0x29

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
Reserved						PWON	

Bit	Marking	Place name	Function	Read and write
b7~b1	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b0	PWON	SDIOC enabled	0: Disable SDIOC 1: Enable SDIOC	R/W

38.3.20 Block Gap Control Register (BLKGPCON)

Offset address: 0x2a

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
		Reserved		IABG	RWC	CR	SABGR

Bit	Marking	Place name	Function	Read and write
b7~b4	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b3	IABG	Block Gap Interrupt Control	0: Receive SDIO device interrupts (card interrupt) during the closing of data block gaps 1: Receive SDIO device interrupt (card interrupt) during open data block gap	R/W
b2	RWC	read wait control	0: Disable read wait function (read wait) 1: Enable read wait function (read wait)	R/W
b1	CR	continue transmission	0: Nothing 1: Cancel the transmission stopped by setting the SABGR bit	R/W
b0	SABGR	Block gap stops transmission	0: Stop transmission when not in data block gaps 1: Stop transmission at data block gaps	R/W

38.3.21 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	

Bit	Marking	Place name	Function	Read and write
B15~b8	FS	Crossover selection	SDIO _x _CK (x=1~2) clock frequency division selection, the reference clock is PCLK1 0x80: divide by 256 of PCLK1 0x40: divide by 128 of PCLK1 0x20: divide by 64 of PCLK1 0x10: divide by 32 of PCLK1 0x08: divide by 16 of PCLK1 0x04: divide by 8 of PCLK1 0x02: divide by 4 of PCLK1 0x01: divide by 2 of PCLK1 0x00: PCLK1 other: Disable setting	R/W
b2	CE	SDIO _x _CK(x=1~2) output control	0: SDIO _x _CK(x=1~2) stop output 1: SDIO _x CK (x=1~2) output	R/W
b0	ICE	clock enable	0: SDIOC action clock on 1: SDIOC action clock is off	R/W

38.3.22 Timeout Control Register (TOUTCON)

Offset address: 0x2e

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
Reserved						DTO[3:0]	

Bit	Marking	Place name	Function	Read and write
b7~b4	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b3~b0	DTO	data timeout	Set the data line SDIOx_Dy (x=1~2) (y=0~7) timeout judgment time, the unit is the clock cycle of PCLK1 0000: PCLK1×2 ¹³ 0001: PCLK1×2 ¹⁴ 1110: PCLK1×2 ²⁷ 1111: Disable setting	R/W

38.3.23 Software Reset Register (SFRST)

Offset address: 0x2f

Reset value: 0x00

B7	b6	b5	b4	b3	b2	b1	b0
Reserved						RSTD	RSTC

Bit	Marking	Place name	Function	Read and write
b7~b3	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b2	RSTD	data reset	Resets all data-related registers, including the following register bits: BUFO, BUF1 PSTAT.BRE, PSTAT.BWE, PSTAT.RTA, PSTAT.WTA, PSTAT.DLA, PSTAT.CID BLKGPCON.CR, BLKGPCON.SABGR NORINTST.BRR, NORINTST.BWR, NORINTST.BGE, NORINTST.TC 0: normal work 1: Perform reset	R/W
b1	RSTC	command reset	Resets all command-related registers, including the following register bits: PSTAT.CIC NORINTST.CC 0: normal work 1: Perform reset	R/W
b0	RSTA	reset all	Reset all SDIOC registers except the card identification function	R/W

38.3.24 Normal 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	

Bit	Marking	Place name	Function	Read and write
b15	El	error interrupt	Set when any error occurs in the Error Interrupt Status Register (ERRINTST)	R
b14~b9	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b8	CINT	Card interrupted	Set when the SDIO device issues a card interrupt request, and reset when the SDIO device cancels the request	R
B7	CRM	card removal	Set when device is removed, reset by writing 1	R/W
b6	CIST	card insertion	Set when device is inserted, reset by writing 1	R/W
b5	BRR	buffer readable	Set when the data in the buffer can be read (PSTAT.BRE=1), reset by writing 1	R/W
b4	BWR	Buffer is writable	Set when the buffer can write data (PSTAT.BWE=1), reset by writing 1	R/W
b3	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b2	BGE	Block gap stops transmission	Set when transfer stops between blocks, reset by writing 1	R/W
b1	TC	Transmission completion	Set when the read and write transfer is complete, reset by writing 1	R/W
b0	CC	command complete	Set when the command without response command is sent and the response with response command is received, write 1 to reset	R/W

38.3.25 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														CTOE	
Bit	Marking	Place name	Function												Read and write
b15~b9	Reserved	-	Read 0 when reading, write 0 when writing												R/W
b8	ACE	Auto send command error	Set when an error occurs in the automatic command (auto CMD), the type of error can be queried in the automatic command error register (ATCERRST), and reset by writing 1												R/W
B7	Reserved	-	Read 0 when reading, write 0 when writing												R/W
b6	DEBE	data stop bit error	Set when the data line detects a low level at the stop bit, reset by writing a 1												R/W
b5	DCE	Data CRC check error	Set when a CRC check error occurs on the data line, reset by writing 1												R/W
b4	DTOE	data timeout error	Set when data timeout occurs, write 1 to reset, the data timeout time is set by the timeout control register (TOUTCON)												R/W
b3	CIE	wrong command number	Set when the command number contained in the received response is incorrect, reset by writing 1												R/W
b2	CEBE	Command stop bit error	Set when the command line detects a low level on the stop bit, reset by writing a 1												R/W
b1	CCE	Command CRC check error	Set when a CRC check error occurs on the command line, reset by writing 1												R/W
b0	CTOE	command timeout error	Set when no response is received for more than 64 SDIOx_CK (x=1~2) cycles after the command is sent, write 1 to reset												R/W

38.3.26 Normal 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

Bit	Marking	Place name	Function	Read and write
b15~b9	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b8	CINTEN	Card Interrupt Status Enable	0: disable NORINTST.CINT set 1: Allow NORINTST.CINT to be set	R
B7	CRMEN	Card Removed Status Enable	0: disable NORINTST.CRM set 1: Allow NORINTST.CRM to be set	R/W
b6	CISTEN	Card Insertion Status Enable	0: disable NORINTST.CIST set 1: Allow NORINTST.CIST to be set	R/W
b5	BRREN	Buffer Readable Status Enable	0: disable NORINTST.BRR set 1: Allow NORINTST.BRR to be set	R/W
b4	BWREN	Buffer writable status enable	0: disable NORINTST.BWR set 1: Allow NORINTST.BWR to be set	R/W
b3	Reserved	-	Read 0 when reading, write 0 when writing	R/W
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	Transfer Complete Status Enable	0: disable NORINTST.TC set 1: Allow NORINTST.TC to be set	R/W
b0	CCEN	Command Completion Status Enable	0: disable NORINTST.CC set 1: Allow NORINTST.CC to be set	R/W

38.3.27 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								ACEEN	Rsvd	DEBEEN	DCEEN	DTOEEN	CIEEN	CEBEEN	CCEEN	CTOEEN

Bit	Marking	Place name	Function	Read and write
b15~b9	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b8	ACEEN	Automatically 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, write 0 when writing	R/W
b6	DEBEEN	Data Stop Bit Error Status Enable	0: Forbid ERRINTST.DEBE to be set 1: Allow ERRINTST.DEBE to be set	R/W
b5	DCEEN	Data CRC check error status enable	0: Disable ERRINTST.DCE set 1: Allow ERRINTST.DCE to be set	R/W
b4	DTOEEN	Data Timeout Error Status Enable	0: Disable ERRINTST.DTOE set 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	CEBEEN	Command Stop Bit Error Status Enable	0: Mask ERRINTST.CEBE set 1: Allow ERRINTST.CEBE to be set	R/W
b1	CCEEN	Command CRC Check Error Status Enable	0: Disable ERRINTST.CCE set 1: Allow ERRINTST.CCE to be set	R/W
b0	CTOEEN	Command Timeout Error Status Enable	0: Disable ERRINTST.CTOE set 1: Enable ERRINTST.CTOE to be set	R/W

38.3.28 Normal interrupt signal enable register (NORINTSGEN)

Offset address: 0x38

Reset value: 0x0000

b1 5	b1 4	b1 3	b1 2	b1 1	b1 0	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0	
						Reserved		CINTSE N	CRMSE N	CISTSE N	BRRSE N	BWRSE N	Rsv d	BGESE N	TCES N	CCSE N

Bit	Marking	Place name	Function	Read and write
b15~b9	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b8	CINTSEN	Card Interrupt Signal Enable	0: Disable NORINTST.CINT request interrupt 1: Allow NORINTST.CINT to request interrupt	R/W
B7	CRMSEN	Card removal signal enable	0: Disable NORINTST.CRM application interruption 1: Allow NORINTST.CRM to request interruption	R/W
b6	CISTSEN	Card Insertion Signal Enable	0: Disable NORINTST.CIST request interrupt 1: Allow NORINTST.CIST to request interrupt	R/W
b5	BRRSEN	Buffer read signal enable	0: Disable NORINTST.BRR request interrupt 1: Allow NORINTST.BRR to request interrupt	R/W
b4	BWRSEN	Buffer writable signal enable	0: Disable NORINTST.BWR request interrupt 1: Allow NORINTST.BWR to request interrupt	R/W
b3	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b2	BGESEN	Data Block Gap Stop Transmission Signal Enable	0: Disable NORINTST.BGE request interrupt 1: Allow NORINTST.BGE to request interrupt	R/W
b1	TCSEN	Transmission complete signal enable	0: Disable NORINTST.TC request interrupt 1: Allow NORINTST.TC to request interrupt	R/W
b0	CCSEN	Command Completion Signal Enable	0: Disable NORINTST.CC request interrupt 1: Allow NORINTST.CC to request interrupt	R/W

38.3.29 Error Interrupt Signal Enable Register (ERRINTSGEN)

Offset address: 0x3a

Reset value: 0x0000

b1 5	b1 4	b1 3	b1 2	b1 1	b1 0	b9	b8	B7	b6	b5	b4	b3	b2	b1	b0
						Reserved	ACEES N	Rsv d	DEBESE N	DCESE N	DTOESE N	CIESE N	CEBESE N	CCESE N	CTOESE N

Bit	Marking	Place name	Function	Read and write
b15~b9	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b8	ACESEN	Automatically send command error signal enable	0: Disable ERRINTST.ACE application interrupt 1: Allow ERRINTST.ACE to request interrupt	R/W
B7	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b6	DEBESEN	Data Stop Bit Error Signal Enable	0: Disable ERRINTST.DEBE application interrupt 1: Allow ERRINTST.DEBE to request interrupt	R/W
b5	DCESEN	Data CRC check error signal enable	0: Disable ERRINTST.DCE application interrupt 1: Allow ERRINTST.DCE to request interrupt	R/W
b4	DTOESEN	Data Timeout Error Signal Enable	0: Disable ERRINTST.DTOE application interrupt 1: Allow ERRINTST.DTOE to request interrupt	R/W
b3	CIESEN	Command number error signal enable	0: Disable ERRINTST.CIE request interrupt 1: Allow ERRINTST.CIE to request interrupt	R/W
b2	CEBESEN	Command Stop Bit Error Signal Enable	0: Shield ERRINTST.CEBE application interrupt 1: Allow ERRINTST.CEBE to request interrupt	R/W
b1	CCESEN	Command CRC check error signal enable	0: Disable ERRINTST.CCE application interrupt 1: Allow ERRINTST.CCE to request interrupt	R/W
b0	CTOESEN	Command Timeout Error Signal Enable	0: Disable ERRINTST.CTOE application interrupt 1: Allow ERRINTST.CTOE to request interrupt	R/W

38.3.30 AutoCommand 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
								CMDE							

Bit	Marking	Place name	Function	Read and write
b15~b8	Reserved	-	Read 0 when reading, write 0 when writing	R/W
B7	CMDE	error not sent	Set when the corresponding error occurs in the b4~b0 bits of this register and other commands are not sent	R
b6~b5	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b4	IE	wrong command number	Set when the received reply contains an autocommand with the wrong number	R
b3	EBE	stop bit error	Set when the command line detects a low level on the stop bit	R
b2	CE	data timeout error	Set when a CRC check error occurs on the command line	R
b1	TOE	command timeout error	Set when no response is received for more than 64 SDIO _x .CK(x=1~2) cycles after the automatic command is sent	R
b0	NE	not executed error	Set when an autocommand was not sent for other reasons	R

38.3.31 Forced Autocommand 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
								FCMDE							

Bit	Marking	Place name	Function	Read and write
b15~b8	Reserved	-	Read 0 when reading, write 0 when writing	R/W
B7	FCMDE	Force not sent error	0: Nothing 1: Force ATCERRST.CMDE error to occur Reading this register bit has no effect	R/W
b6~b5	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b4	FIE	Mandatory command number error	0: Nothing 1: Force ATCERRST.IE error to occur Reading this register bit has no effect	R/W
b3	FEBE	Forced stop bit error	0: Nothing 1: Force an ATCERRST.EBE error Reading this register bit has no effect	R/W
b2	FCE	Force data timeout error	0: Nothing 1: Force ATCERRST.CE error to occur Reading this register bit has no effect	R/W
b1	FTOE	Force command timeout error	0: Nothing 1: Force ATCERRST.TOE error to occur Reading this register bit has no effect	R/W
b0	FNE	Force not executed error	0: Nothing 1: Force ATCERRST.NE error to occur Reading this register bit has no effect	R/W

38.3.32 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
<hr/>																	
Bit	Marking	Place name	Function										Read and write				
b15~b9	Reserved	-	Read 0 when reading, write 0 when writing										R/W				
b8	FACE	Force auto send command error	0: Nothing 1: Force ERRINTST.ACE error to occur Reading this register bit has no effect										R/W				
B7	Reserved	-	Read 0 when reading, write 0 when writing										R/W				
b6	FDEBE	Force data stop bit error	0: Nothing 1: Force ERRINTST.DEBE error to occur Reading this register bit has no effect										R/W				
b5	FDCE	Mandatory data CRC check error	0: Nothing 1: Force ERRINTST.DCE error to occur Reading this register bit has no effect										R/W				
b4	FDTOE	Force data timeout error	0: Nothing 1: Force ERRINTST.DTOE error to occur Reading this register bit has no effect										R/W				
b3	FCIE	Mandatory command number error	0: Nothing 1: Force ERRINTST.CIE error to occur Reading this register bit has no effect										R/W				
b2	FCEBE	Force command stop bit error	0: Nothing 1: Force ERRINTST.CEBE error to occur Reading this register bit has no effect										R/W				
b1	FCCE	Mandatory command CRC check error	0: Nothing 1: Force ERRINTST.CCE error to occur Reading this register bit has no effect										R/W				
b0	FCTOE	Force command timeout error	0: Nothing 1: Force ERRINTST.CTOE error to occur Reading this register bit has no effect										R/W				

38.3.33 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															
Bit	Marking	Place name	Function	Read and write											
b31~b4	Reserved	-	Read as "0", write as "0"	R/W											
b3	SELMMC2	SDIOC2 MMC Mode Enable	0: SDIOC2 select SD mode 1: SDIOC2 selects MMC mode	R/W											
b2	Reserved	-	Read as "0", write as "0"	R/W											
b1	SELMMC1	SDIOC1 MMC Mode Enable	0: SDIOC1 select SD mode 1: SDIOC1 selects MMC mode	R/W											
b0	Reserved	-	Read as "0", write as "0"	R/W											

39 Ethernet MAC Controller (ETHMAC)

39.1 Summary

The Ethernet MAC controller (ETHMAC) is used to send and receive data in accordance with the IEEE802.3-2002 standard in the Ethernet network, and has various application fields, such as switches, network interface cards, and so on. The MAC controller supports two industry standard interfaces to the external physical layer (PHY): Media Independent Interface (MII) (defined in the IEEE802.3 specification) and Reduced Media Independent Interface (RMII).

Mainly follow the following protocol specifications:

- IEEE802.3-2002 for Ethernet MAC
- IEEE1588-2008 standard for specifying networked clock synchronization
- AMBA2.0, used for AHB host/slave port
- RMII interface specification

39.2 Basic Features

39.2.1 Basic block diagram

The following figure shows the architecture block diagram of the Ethernet MAC controller.

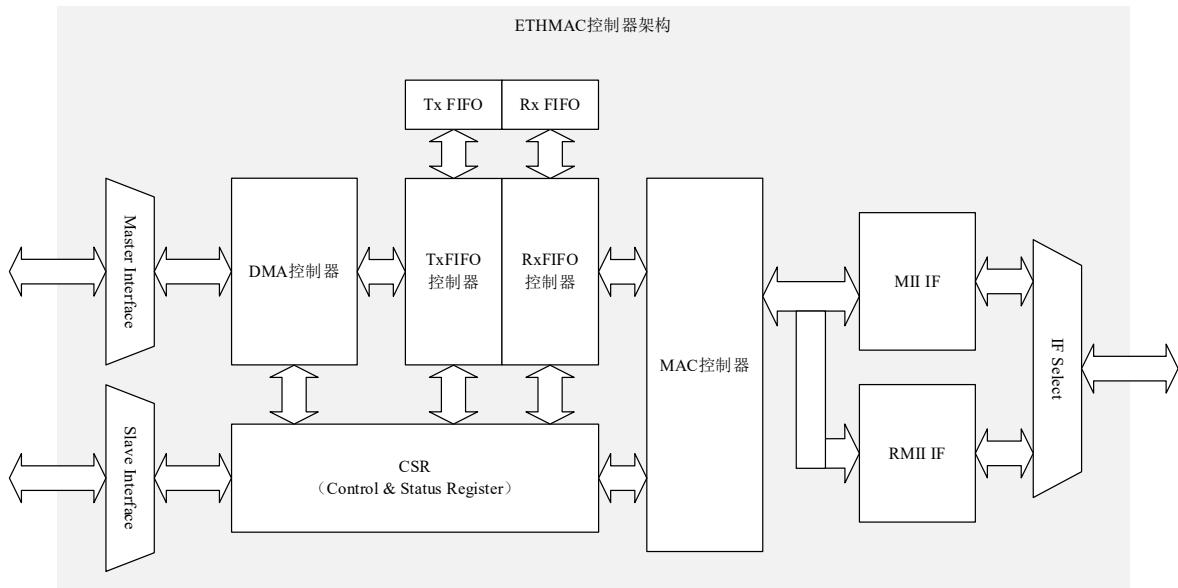


Figure 39-1 Ethernet MAC Controller Architecture Diagram

As can be seen from the figure, the core part of the Ethernet MAC controller includes AHB Master interface, AHB Slave interface, Rx FIFO, Tx FIFO and DMA controller, MAC controller, interface control logic and so on. The DMA controller connects the MAC core to the system memory through the AHB master-slave interface.

The AHB Master interface is used for data transmission, and the AHB Slave interface is used for the basic register access configuration of the controller, etc.

When sending data, first send the data from the system memory to the Tx FIFO by the DMA controller for buffering, and then send it to the Tx FIFO through the MII interface or RMII interface after a series of processing by the MAC controller (COE control, PTP control, MMC control, etc.). external PHY.

When receiving data, the incoming data is received through the MII interface or RMII interface, and sent to the Rx FIFO after a series of processing by the MAC controller (COE control, PTP control, MMC control, etc.) transferred to system memory.

39.2.2 ETH_MAC features

The basic functional characteristics of ETH_MAC are as follows:

- Support external PHY interface to achieve 10/100Mbps data transfer rate
- Supports MII or RMII interface to communicate with external Fast Ethernet PHY
- Provide separate send, receive and control interfaces for applications

- Configure and manage up to 32 PHY devices using the SMI interface
- Supports detection of remote wakeup frames and AMD Magic Packet™ frames
- Supports LoopBack mode with internal loopback via MII
- Supports full-duplex and half-duplex operation
 - Supports CSMA/CD protocol for half-duplex operation
 - Supports backpressure flow control for half-duplex operation
 - Supports IEEE802.3x flow control for full duplex operation
 - Received pause control frames can be forwarded to the application in full duplex operation

Note:

- The "application" mentioned in this chapter means that the MAC transfers the received frame to the system memory through RxDMA for use by the upper network layer and transport layer; or transfers the data of the upper network layer and transport layer from the system controller through the system controller. TxDMA is transmitted to the MAC, and the upper network layer and transport layer are called applications.
- In full duplex operation, if the flow control input signal disappears, a zero-slice pause frame is automatically sent
- Ethernet frame processing
 - When receiving an Ethernet frame, automatically remove the PAD and FCS fields of frames whose length field is less than 1536
 - When receiving an Ethernet frame, automatically remove the FCS field of the type frame
 - When receiving an Ethernet frame, automatically calculate the CRC of the received frame
 - When sending Ethernet frames, insert and replace SAs
 - Insert and replace CRC when sending Ethernet frame
 - When sending Ethernet frames, insert, replace, and delete VLAN frame IDs
 - When sending Ethernet frames, automatically generate PAD padding for frames smaller than 60 bytes
 - Handle automatic resending of collided frames when sending Ethernet frames
 - Supports programmable frame interval (40-96 bit time in steps of 8) when sending Ethernet frames
 - Programmable frame length when receiving or sending Ethernet frames, supports jumbo frames up to 16KB
- Ethernet frame address filtering
 - Supports up to five 48-bit perfect (DA) address filters, masking each byte
 - Supports up to four 48-bit perfect (SA) address filters, masking each byte
 - Support 64-bit Hash filter, suitable for unicast and multicast destination address filtering
 - Can transmit all multicast address frames
 - Supports mixed mode to transmit all frames without filtering for network monitoring

- A status report is attached when all incoming packets are delivered (every time they are filtered)
- VLAN identifier filtering
 - Supports a set of VLAN tag filters
 - Supports 12-bit/16-bit VLAN tag selection
 - Support VLAN tag hash filtering
- LAY3 IP address filtering
 - Supports a set of LAY3 IP address filters, masking each byte
 - Supports address filtering selection for IPv4 and IPv6 packets
- LAY4 TCP/UDP port filtering
 - Support a set of LAY4 TCP/UDP port filters
- Remote Wakeup Frame Filtering Mode
 - Supports four groups of remote wake-up frame filters and controls four groups of mask operations
- COE engine
 - Optionally discard frames without TCP/UDP fields when receiving frames
 - Check the Header Checksum field of the received IPv4 packet
 - Checksum field of received TCP, UDP, and ICMP packets
 - Perform Checksum calculation on the sent IPv4 packets, and insert the calculation result into the Header Checksum field
 - Perform Checksum calculation on the sent TCP, UDP, and ICMP packets, and insert the calculation result into the Checksum field
- Support MMC counter for network statistics
- FIFO processing
 - 2KB transmit FIFO (Tx FIFO) with one programmable threshold and 2KB receive FIFO (Rx FIFO) with one configurable threshold
 - Both Rx FIFO and Tx FIFO support store-and-forward mode
 - Software controlled refresh of Tx FIFO
 - In store-and-forward mode, Rx FIFO can optionally filter all error frames and not forward these error frames to the application
 - The Rx FIFO can automatically generate pause control frames or backpressure signals to be sent to the MAC core based on the fill (threshold configurable) level
 - Lost or corrupted frames in Rx FIFO can be counted
 - When the Rx FIFO stores multiple frames, by inserting the receiving state vector into the Rx FIFO after the EOF transmission, the Rx FIFO does not need to store the receiving state of these frames

39.2.3 ETH_PTP features

The basic functional characteristics of ETH_PTP are as follows:

- Supports receiving and sending frames for timestamp snapshots, inserting timestamps
- Supports coarse and fine calibration of system time
- Support 2 sets of target time settings, when the system time is greater than the target time, an interrupt will be triggered
- Support 2 sets of PPS output, which can be used as event output to control the actions of other modules

39.2.4 ETH_DMA characteristics

The basic functional characteristics of ETH_DMA are as follows:

- Support software to set the AHB burst type of AHB Master interface (fixed or indeterminate burst)
- All AHB burst types are supported in the AHB Slave interface
- AHB Master Interface Selectable Address Alignment Burst Transfer
- Supports RxDMA and TxDMA engines to independently program burst lengths to fully utilize the bus
- Supports byte-aligned addressing of data buffers
- Descriptor Features:
 - Support both ring and chain descriptor linking methods
 - The descriptor architecture can transfer large data blocks with little CPU intervention
 - Each descriptor can transfer up to 8KB of data
- Both TxDMA and RxDMA support store-and-forward mode
- Reports the combined status of normal operation and transmission errors
- Programmable interrupt options, select different interrupts according to different application scenarios
- Control transmit/receive completion interrupt by frame
- Round-robin scheduling arbitration or fixed-priority arbitration between receive engines and transmit engines

39.3 Interface Description

The Ethernet MAC controller supports MII, RMII, and SMI interface types. The user selects one of the MII interface and the RMII interface as the interface for data exchange according to actual needs. The SMI interface exists independently and is used for basic configuration related to the PHY, data exchange.

By setting the IFSEL bit of the interface control register (ETH_MAC_IFCONFR), you can choose whether to use the MII interface or the RMII interface for Ethernet control. Before configuring other features of ETHMAC, this bit must be set first.

39.3.1 MII interface

The Media Independent Interface (MII) defines the interconnection between the MAC sublayer and the PHY at data transfer rates of 10Mbps and 100Mbps. Figure 39-2 shown is the basic connection diagram of the MII interface.

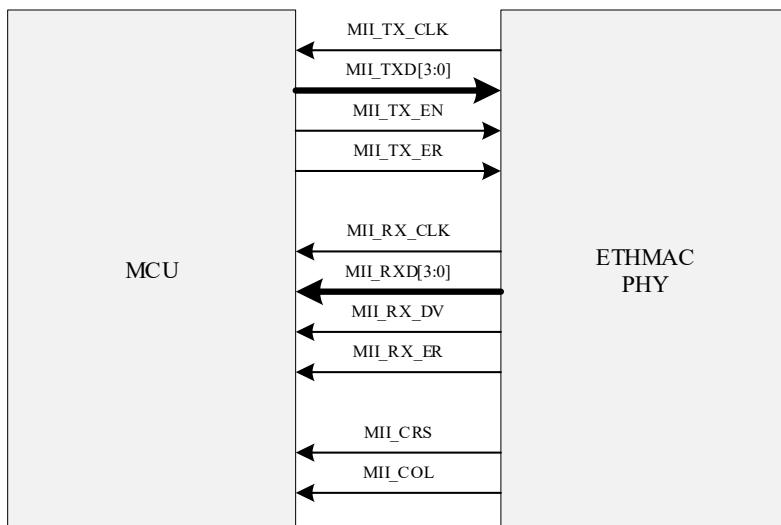


Figure 39-2 MII interface connection diagram

Table 39-1 The function of each signal of the MII interface is introduced.

Table 39-1 MII interface signal description

Interface signal name	Functional description
MII_TX_CLK	Send the clock signal. This signal provides the reference timing for TX data transmission. 2.5MHz when the rate is 10Mbps; 25MHz when the rate is 100Mbps.
MII_RX_CLK	receive the clock signal. This signal provides the reference timing for RX data transmission. 2.5MHz when the rate is 10Mbps; 25MHz when the rate is 100Mbps.
MII_TX_EN	Send the enable signal. This signal indicates that the MAC is currently sending data to the MII.
MII_TX_ER	Send an error signal.
MII_TXD[3:0]	Send data signals. This signal is a group of 4 data signals, driven synchronously by the MAC sublayer, and is valid only when the MII_TX_EN signal is valid. MII_TXD[0] is the least significant bit, MII_TXD[3] is the most significant bit. When the MII_TX_EN signal is inactive, the transmitted data will not have any effect on the PHY.
MII_RX_DV	Receive data valid signal. This signal indicates that the PHY is currently sending data to the MII. For a frame to be received correctly, the signal must cover the frame to be received in time range, starting no later than the time the SFD field appears.
MII_RX_ER	Receive an error signal. This signal must be maintained for one or more transmit clock cycles to indicate to the MAC sublayer that an error was detected somewhere in the frame. This error condition must be verified by the RX_DV signal (see Table 40-3).
MII_RXD[3:0]	receive data signals. This signal is a group of 4 data signals, driven synchronously by the external PHY, and is valid only when the MII_RX_DV signal is valid. MII_RXD[0] is the least significant bit and MII_RXD[3] is the most significant bit. When the MII_RX_DV signal is inactive and the MII_RX_ER signal is enabled, specific MII_RXD[3:0] values are used to transmit specific information from the PHY (see Table 40-3).
MII_CRS	carrier sense signal. This signal is enabled by the PHY when the transmitting or receiving medium is in a non-idle state; this signal is disabled by the PHY when both the transmitting and receiving medium are in an idle state. The PHY must ensure that the MII_CRS signal remains active under collision conditions. This signal does not need to be synchronized with the TX and RX clocks. In full duplex mode, this signal has no meaning.
MII_COL	Collision detection signal. The PHY must enable the collision detection signal immediately upon detection of a collision on the medium, and the collision detection signal must remain active as long as the collision condition exists. This signal does not need to be synchronized with the TX and RX clocks. In full duplex mode, this signal has no meaning.

Table 39-2 Introduce the relevant signal encoding status when sending data.

Table 39-2 Signal state description when sending data

MII_TX_EN	MII_TX_ER	MII_TXD[3:0]	status description
0	0	0000~1111	Normal frame, no data transmission
0	1	0000~1111	meaningless
1	0	0000~1111	Send data normally
1	1	0000~1111	Error sending data

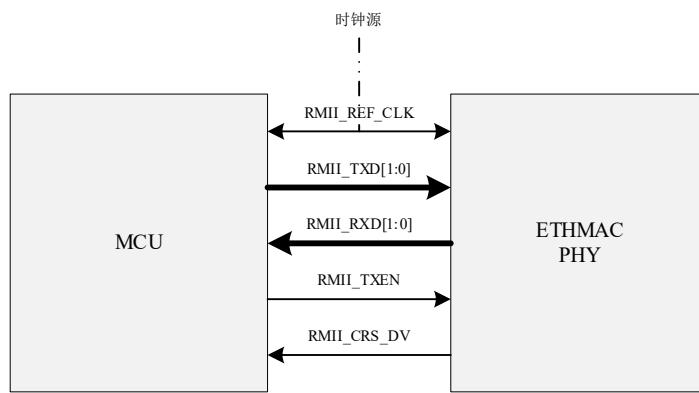
Table 39-3 Introduce the relevant signal encoding status when receiving data.

Table 39-3 Signal state description when sending data

MII_RX_DV	MII_RX_ER	MII_RXD[3:0]	status description
0	0	0000~1111	Normal frame, no data received
0	1	0000	normal frame
		0001~1101	meaningless
		1110	Error carrier detection
		1111	meaningless
1	0	0000~1111	Receive data normally
1	1	0000~1111	Error receiving data

39.3.2 RMII interface

The Reduced Media Independent Interface (RMII) specification reduces the port count between a microcontroller Ethernet MAC controller and an external PHY at 10/100Mbps. The number of ports it uses is reduced from 16 of MII to 7, as shown in Figure 39-3 shown.

**Figure 39-3 RMII interface connection diagram**

As can be seen from the connection diagram, the main differences between the RMII interface and the MII interface are as follows:

1. The transmit clock and receive clock are unified as the reference clock RMII_REF_CLK, which must be a 50MHz clock
2. Data cables changed from 4 to 2
3. The MII_COL signal is discarded, and the MII_RX_DV and MII_CRS signals are combined into the RMII_CRS_DV signal
4. MII_TX_ER, MII_RX_ER signals discarded

39.3.3 SMI interface

The SMI interface enables site management of the PHY through the Ethernet MAC controller. The MAC controller accesses any PHY register via the clock and data lines. The interface supports access to up to 32 PHYs. Figure 39-4 shown is the connection diagram of the SMI interface.



Figure 39-4 SMI interface connection diagram

Table 39-4 The function of each signal of the SMI interface is introduced.

Table 39-4 SMI interface signal description

interface signal name	Functional description
SMI_MDC	Periodic clock that provides reference timing when transmitting data at a maximum frequency of 2.5MHz. The shortest high level and shortest low level of MDC must both be 160ns. The minimum period of the MDC must be 400 ns. In the idle state, the SMI drives the MDC clock signal low.
SMI_MDIO	The data input/output bit stream is used for data information transmission with the PHY device through the MDC clock signal.

The MAC controller can select a PHY from 32 PHYs, and then select a register from the 32 registers contained in the selected PHY to send control data or receive status information. Only one register in a PHY can be addressed at any given time.

The frame format of SMI is as follows Table 39-5 , which follows the following format during data exchange with the PHY:

Table 39-5 SMI frame format

	SMI frame format description							
	Head (32bit)	Start	Operation	PhyAdd	RegAdd	TA	Data (16bit)	Idle
Read	1...1	01	10	xxxxx	xxxxx	Z0	xx...xx	Z
Write	1...1	01	01	xxxxx	xxxxx	10	xx...xx	Z

39.3.4 Ethernet port configuration

According to the above three types of data interfaces: MII, RMII, and SMI, the port function assignment of the Ethernet MAC controller and the corresponding relationship of each type of interface are arranged as follows:Table 39-6 shown:

Table 39-6 ETHMAC port function assignment

ETH port name	Interface function signal
ETH_MII_RMII_RXCLK	MII_RX_CLK / RMII_REF_CLK
ETH_MII_TXCLK	MII_TX_CLK
ETH_MII_RMII_TXEN	MII_TX_EN / RMII_TX_EN
ETH_MII_TXER	MII_TX_ER
ETH_MII_RMII_TXD1~0	MII_TXD[0] / RMII_TXD[0]
	MII_TXD[1] / RMII_TXD[1]
ETH_MII_TXD3~2	MII_TXD[2]
	MII_TXD[3]

ETH port name	Interface function signal
ETH_MII_RMII_RXDV	MII_RX_DV / RMII_CRS_DV
ETH_MII_RXER	MII_RX_ER
ETH_MII_RMII_RXD1~0	MII_RXD[0] / RMII_RXD[0]
	MII_RXD[1] / RMII_RXD[1]
ETH_MII_RXD3~2	MII_RXD[2]
	MII_RXD[3]
ETH_MII_CRS	MII_CRS
ETH_MII_COL	MII_COL
ETH_SMI_MDC	SMI_MDC
ETH_SMI_MDIO	SMI_MDIO

39.4 Functional description

The Ethernet MAC controller follows the IEEE802.3 international standard for local area networks (LAN), and uses CSMA/CD (Carrier Sense Multiple Access with Collision Detection) as the basic communication method. It includes a MAC802 with MII and RMII interfaces. .3 (Media Access Control) controller and a dedicated DMA controller.

39.4.1 ETH_MAC function

The MAC sublayer performs the following functions related to the data link control steps:

- Data encapsulation (send and receive)
 - Framing (frame boundary delimitation, frame synchronization)
 - Addressing (handling source and destination addresses)
 - Error detection
- Media Access Management
 - Media allocation (collision avoidance)
 - Contention Resolution (Conflict Handling)

The MAC sublayer mainly has two working modes:

- Half-duplex mode: Sites contend for physical media using CSMA/CD algorithms
- Full duplex mode: Data can be sent and received simultaneously without resolving contention issues (not required by the CSMA/CD algorithm) when the following conditions are met:
 - Physical media capable of supporting simultaneous transmit and receive
 - There are exactly 2 sites connected to the LAN
 - Both stations are configured for full duplex operation

39.4.1.1 MAC 802.3 frame format

MAC data communication can use two frame formats:

- Basic MAC Frame Format

- MAC frame format with VLAN ID (an extension to the basic MAC frame format)

Figure 39-5 Shown are the specific structures of the two frame formats.

基本MAC帧格式

前导码 PR (7 Byte)	帧首定界码 SFD (1 Byte)	目标地址 DA (6 Byte)	源地址 SA (6 Byte)	长度/类型 LEN/TYPE (2 Byte)	数据 DATA (46~1500字节)	填充PAD	帧校验序列 FCS (4 Byte)
-----------------------	--------------------------	------------------------	-----------------------	-------------------------------	---------------------------	-------	--------------------------

带VLAN标识MAC帧格式

前导码 PR (7 Byte)	帧首定界码 SFD (1 Byte)	目标地址 DA (6 Byte)	源地址 SA (6 Byte)	802.1Q VLAN Tag (4 Byte)	长度/类型 LEN/TYPE (2 Byte)	数据 DATA (46~1500字节)	填充PAD	帧校验序列 FCS (4 Byte)
^								
Tag Protocol ID “0x8100” (2byte)		User Priority (3bits)		CFI (1bit)		VLAN ID (12bits)		

Figure 39-5 MAC frame structure

With the exception of the Frame Check Sequence (FCS), the Ethernet MAC controller transmits each byte in LSB first-out order.

The CRC calculation includes all bytes of the frame data except the preamble and the frame header to delimit the code field. The Ethernet frame 32CRC generator polynomial is 0x04C11DB7, and this polynomial is used for all 32-bit CRC calculations in the Ethernet module, as shown in the following formula:

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

A frame is defined as an invalid MAC frame by one of the following conditions:

- The frame length does not match the expected value specified by the length/type field. If the length/type field contains a type value, the frame length is considered consistent with this field (no invalid frames)
- Frame length is not an integer multiple of bytes (extra bits)
- The CRC value calculated from the incoming frame does not match the included FCS

39.4.1.2 MAC frame sending action

TxDMA controls all transactions in the transmit path. Ethernet frames read from system memory are fed into the Tx FIFO by TxDMA and then transferred to the MAC core. At the end of the frame transfer, the transmit status is obtained from the MAC core and passed back to TxDMA.

The Ethernet MAC controller is equipped with a 2KB Tx FIFO. The Tx FIFO fill level will be indicated to the DMA so that the TxDMA can initiate data transfers in the required system memory via the AHB interface. Data from the AHB Master interface will be fed into the Tx FIFO.

When SOF is detected, the MAC receives the data and starts sending to the MII. After the application initiates the send, the time it takes to send frame data to the MII is variable, depending on delay factors such as IFG delay, time to send header/SFD, and any backoff delay for half-duplex mode. After the EOF is transmitted to the MAC core, the core will complete the normal transmission, and then return the transmitted status to TxDMA. If a regular collision occurs during transmission (in half-duplex mode), the MAC core asserts the transmit state, then accepts and discards all subsequent data until the next SOF is received. When a retry request from the MAC is detected (in state), the same frame should be resent from the SOF.

If data is not continuously provided during transmission, the MAC will issue an underflow condition. During normal transmission of a frame, if the MAC receives a SOF without obtaining the EOF of the previous frame, it will ignore the SOF and treat the new frame as a continuation of the previous frame.

There are two modes of operation for delivering data to the MAC core:

- In threshold mode, data is ready and forwarded to the MAC core as soon as the number of bytes in the FIFO exceeds the configured threshold (or the end of the frame is written before the threshold is exceeded). This threshold is configurable using the TTC bits of ETH_DMA_BUSMODR.
- In store-and-forward mode, frames are delivered to the MAC core only after the complete frame is stored in the FIFO. If the size of the Tx FIFO is smaller than the Ethernet frame to be sent, the frame is delivered to the MAC core when the Tx FIFO is nearly full.

The application can clear the entire contents of the Tx FIFO by setting the FTF bit (ETH_DMA_OPRMODR register[20]). This bit clears itself and initializes the FIFO pointer to its default state. If this bit is set when transmitting a frame to the MAC core, the transmission will stop, the Tx FIFO is considered empty, the MAC transmitter will have an underflow event and the corresponding status word will be forwarded to the TxDMA.

Send basic protocol

The MAC controls the transmission of Ethernet frames. It performs the following functions to meet the IEEE802.3/802.3z specification. include:

- Generate header and SFD
- Generate Jam blocking signal in half duplex mode
- Control Jabber timeout
- Controls flow (back pressure) in half-duplex mode
- Generate send frame status
- Contains IEEE1588 compliant timestamped snapshot logic

When a new frame is requested, the MAC will send the header and SFD, followed by the data. The header is defined as 7 bytes in the "10101010" style, and the SFD is defined as 1 byte in the "10101011" style.

The collision window is defined as 1 slot (512 bit times for 10/100 Mbps Ethernet). Blocking signal generation is only available in half-duplex mode, not in full-duplex mode. In MII mode, if a collision occurs at any time between the start of transmission of the frame and the end of the CRC field, the MAC will send a 32-bit Jam blocking signal of 0x55555555 on MII to notify all other stations that a collision has occurred. If a collision occurs during the header transmission phase, the MAC will complete the transmission of the header and SFD, and then send the Jam blocking signal.

The MAC core uses a Jabber timer to cut off the transmission of Ethernet frames when more than 2048 bytes are transmitted. In half-duplex mode, the MAC uses a delay mechanism for flow control (back pressure). When an application requests to stop receiving frames, if transmit flow control is enabled, a 32-byte Jam signal is sent whenever the MAC detects a received frame. This can cause a conflict and cause the remote station to fall back. The application requests flow control by setting the BPA bit in the ETH_MAC_FLOCTRL register. If an application requests a frame to be sent, it will be sent according to the schedule even if the backpressure feature is activated. If the backpressure function remains active for a long time (more than 16 consecutive collision events occur), the remote station will abort transmission due to too many collisions. If the IEEE1588 timestamp function is enabled for the transmit frame, a time snapshot of the system time is obtained when the SFD is placed on the transmit MII bus.

Scheduling algorithm program

The MAC is responsible for scheduling frame transmissions on the MII. The frame interval between two transmitted frames is maintained by the IFG setting or truncated binary exponential backoff algorithm. After the IFG and fallback conditions are met, the MAC enables transmission.

The IFG setting ensures an idle period between two transmitted frames, the configured frame interval (IFG bit in the ETH_MAC_CONFIGR register). If the frame to be transmitted arrives before the configured IFG time, the MII waits for an enable signal from the MAC before starting transmission. As long as the carrier signal of the MII enters an inactive state, the MAC will start the IFG counter. When the IFG count value is equal to the set value, the MAC will enable transmission in full duplex mode.

In half-duplex mode, when the IFG is configured for 96 bit times, the MAC will follow the compliance rules specified by the IEEE802.3 specification. If a carrier is detected within the first two-thirds of the IFG interval (64-bit time for all IFG values), the MAC will reset its IFG counter. If a carrier is detected within the last third of the IFG interval, the MAC will continue to perform IFG counting and enable the transmitter after the IFG interval has elapsed.

In half-duplex mode, a truncated binary exponential backoff algorithm is implemented, which is specifically controlled by the BL bit setting of the MAC configuration register (ETH_MAC_CONFIGR).

Retransmission during conflict

In half-duplex mode, a collision event may occur on the MAC line interface when transmitting frames to the MAC. The MAC will indicate a retry even before it receives an end-of-frame status, then will enable resend and send the frame out of the Tx FIFO again.

When more than 96 bytes are sent to the MAC core, the Tx FIFO controller will free the space so that the DMA can send more data. This means that there is no way to resend after the threshold is exceeded or when the MAC core indicates a delayed collision event.

TxFIFO refresh

The MAC controller can empty the Tx FIFO through the FTF bit in the Operation Mode Register (ETH_DMA_OPRMODR). The flush operation is an immediate operation, even though the Tx FIFO is transmitting frames to the MAC core, the Tx FIFO and the corresponding pointer will be cleared to the initial state. This will cause an underflow event to be generated in the MAC transmitter and the frame transmission will be aborted, at this time the status of the frame will mark both underflow and empty events (FFF and UDE bits of TDES0). During the flush operation, no data is transferred from the TxDMA to the Tx FIFO, and the TxDMA transmits the corresponding number of transfer send status words to the application according to the number of flushed frames (including partial frames). The frame clear status bit (FFF bit of TDES0) will be set to 1 for a completely cleared frame. When the TxDMA has accepted the status word of all flushed frames, the flush operation is complete and the FTF bit will then be cleared. At this point, new frames from TxDMA will be accepted. After the flush operation is complete, all committed data will be discarded unless the data begins with a SOF token.

CRC processing and PAD generation

When the number of bytes received from the application is less than 60 (DA+SA+LT+data), the DPAD bit of the transmit descriptor TDES0 can be reset, and zero (PAD) is appended to the transmit frame to make the data length exactly 46 bytes. In order to meet the minimum data field requirements of IEEE802.3, you can also choose not to attach any padding value.

The MAC also calculates the Cyclic Redundancy Check (CRC) of the Frame Check Sequence (FCS) field, and can optionally insert or replace the check value into the data being sent according to the settings of the DCRC and CRCR bits of the transmit descriptor TDES0 , please refer to the chapters of Regular Tx Descriptor and Enhanced Tx Descriptor for details.

Note: The Enhanced Tx Descriptor must be enabled when the CRC replacement feature is enabled.

If the MAC is programmed not to append the CRC value to the end of the Ethernet frame, the calculated CRC is not sent. But when the DPAD bit of TDES0 is 0, the frame less than 60 bytes

(DA+SA+LT+data) will add padding PAD after the data, and the CRC result will also be automatically appended to the end of the padding frame.

Source address SA processing

If the data from the application does not contain the source address field, the MAC can insert the field; if it contains the source address field, the MAC can choose to replace the field with the value set in the MAC address register 0~1. The specific processing method of the SA field can be controlled by the SAIRC bit of the MAC configuration register (ETH_MAC_CONFIGR) or the SAIRC bit of the descriptor TDES1.

Note: The enhanced Tx descriptor must be enabled when enabling this function through descriptor mode.

VLAN identification processing

The MAC can also insert, replace, and delete the identifier field of the VLAN frame from the application. The specific processing method can be controlled by the VLANC bit of the VLAN tag transmission control register (ETH_MAC_VTACTLR) or the VLANC bit of the descriptor TDES0.

Note: The enhanced Tx descriptor must be enabled when enabling this function through descriptor mode.

Send status word

At the end of the transmission of the Ethernet frame to the MAC core, and after the core has finished sending the frame, the transmit status is provided to the application via the transmit descriptor. See Bits[17:0] in TDES0 for a detailed description of the transmit status. If the IEEE1588 timestamp function is enabled, it will return the 64-bit timestamp of a specific frame and the transmission status.

Send pause frame

In full-duplex mode, when the transmit flow control enable bit (TFE bit in ETH_MAC_FLOCTRLR) is set to 1, the MAC will generate pause frames and send pause frames as needed. Pause frames are sent with the calculated CRC appended. Pause frame generation can be initiated in two ways.

- By requesting flow control by setting the FCA bit in the ETH_MAC_FLOCTRLR register, the MAC will generate and transmit a single pause frame. The pause time value in the generated frame is the pause time value programmed in ETH_MAC_FLOCTRLR. To extend or end the pause time before the time specified in the previously transmitted pause frame, the user must program the PAUSET pause time value in the ETH_MAC_FLOCTRLR register with the appropriate value in the application, and then request another pause frame transmission.
- When the Rx FIFO is full and the application has requested flow control, the MAC will generate and send a pause frame. The pause time value in the generated frame is the pause time value

programmed in ETH_MAC_FLOCTRL. If the Rx FIFO remains full for a configurable number of slots (PLT bit in ETH_MAC_FLOCTRL) before this pause time expires, a second pause frame will be sent. This process will repeat as long as the Rx FIFO remains full. If this condition is no longer met before the pause time expires, the MAC will send a pause frame with a pause time of zero, indicating to the remote port that the Rx FIFO is ready to receive new data frames.

COE engine

During network interconnection, basic communication protocols such as TCP and UDP implement checksums and are equipped with checksum fields, which help determine the integrity of data sent over the network. Since the most widespread use of Ethernet is to encapsulate TCP and UDP over IP datagrams, the Ethernet MAC controller has a checksum offload function (COE engine) that supports checksum calculation, insertion, and reception in the transmit path. Checksum calculation, detection in the path. This section describes the operation of the COE engine in sending, and section 30.4.1.3.6 describes the operation of the COE engine in receiving.

The COE engine calculates the TCP, UDP or ICMP checksum over the complete frame and inserts it into the checksum field of the header. Due to this requirement, this feature is only enabled when the Tx FIFO is configured in store-and-forward mode (TSF bit in the ETH_DMA_OPRMODR register is set). If the kernel is configured in threshold mode, the COE engine will be bypassed.

It must be ensured that the Tx FIFO is deep enough to store a complete frame for transmission to the MAC core transmitter. If the depth of the FIFO is less than the size of the incoming Ethernet frame, the TCP/UDP/ICMP checksum insertion function is bypassed and only the IPv4 header checksum insertion function is supported, even in store-and-forward mode.

The control of the transmit checksum offload function can be achieved by setting the CIC bits ([23:22] in TDES0) to realize the checksum control of each frame.

■ IP header checksum

In an IPv4 datagram, the integrity of the header fields is indicated by the 16-bit header checksum field (bytes 11 and 12 of the IPv4 datagram). The COE engine will detect an IPv4 datagram when the value of the type field of the Ethernet frame is 0x0800 and the value of the version field of the IP datagram is 0x4. During calculation, the checksum field of the input frame is ignored and replaced with the calculated value. The IPv6 header does not have a checksum field, therefore, the COE does not modify the IPv6 header field. The result of the IPv4 header checksum calculation is indicated by the IP header error status bit (IHE bit of TDES0) in the send status. This status bit will be set whenever the value of the EtherType field does not match the value of the IP Header Version field, or when the Ethernet frame does not have enough data (as indicated by the IP Header Length field).

So, to summarize, an IP header error occurs when:

- a) For IPv4 datagrams:

- The ethertype is 0x0800, but the IP header version field is not equal to 0x4
 - The IPv4 header length field indicates a value less than 0x5 (20 bytes)
 - The total frame length is less than the value given by the IPv4 header length field
- b) For IPv6 datagrams:
- The ether type is 0x86DD, but the IP header version field is not equal to 0x6
 - The frame ended before the IPv6 header (40 bytes), or the extension header was fully received (as given in the corresponding header length field in the extension header)

If the Ethertype field indicates an IPv4 payload, an IPv4 header checksum is inserted even if the COE engine detects an IP header error.

■ TCP/UDP/ICMP checksum

The TCP/UDP/ICMP checksum function processes IPv4 or IPv6 headers (including extension headers) and determines whether the encapsulated payload is TCP, UDP, or ICMP.

In the following two cases, the COE engine will be invalid:

- a) A non-TCP, non-UDP, or non-ICMP/ICMPv6 payload.
- b) Fragmented IP frames (IPv4 or IPv6), IP frames with security features (for example, authentication headers or encapsulated security payloads), and IPv6 frames with routing headers.

With the COE engine active, it computes the checksum of the TCP, UDP or ICMP payload and inserts it into the appropriate field in the header. It works in two ways:

- a) The TCP, UDP or ICMPv6 pseudo-header is not included in the checksum calculation and is assumed to be present in the checksum field of the incoming frame. The checksum field is included in the checksum calculation and then replaced with the final calculated checksum.
- b) A TCP, UDP or ICMPv6 pseudo-header is included in the checksum calculation, the checksum field is ignored and the checksum field is overwritten with the final calculated value.

Note: For ICMP-over-IPv4 packets, the checksum field in the ICMP packet must always be 0x0000 in both modes, since no pseudo-header is defined for it. If not equal to 0x0000, an incorrect checksum may be inserted into the packet.

The result of this operation is indicated by the payload error status bit (TPCE bit of TDES0) in the transmit descriptor. The TPCE bit is set when one of the following conditions is detected:

- a) In store-and-forward mode, the frame is forwarded to the MAC transmitter, but the end of frame is not written to the Rx FIFO
- b) The packet has ended before the number of bytes indicated by the payload length field in the IP header was received

Note: When the length of the packet is greater than the indicated payload length, the bytes

are ignored as padding and no error is reported. When the first type of error is detected, the TCP, UDP, or ICMP headers are not modified. For the second error type, the calculated checksum will still be inserted into the corresponding header field.

Send timing

In RMII interface, each nibble from MII is sent on RMII, two bits are sent at a time, and the order of sending the two bits is as follows. Figure 39-6 shows. The lower order bits (D0 and D1) are sent first, followed by the higher order bits (D2 and D3).

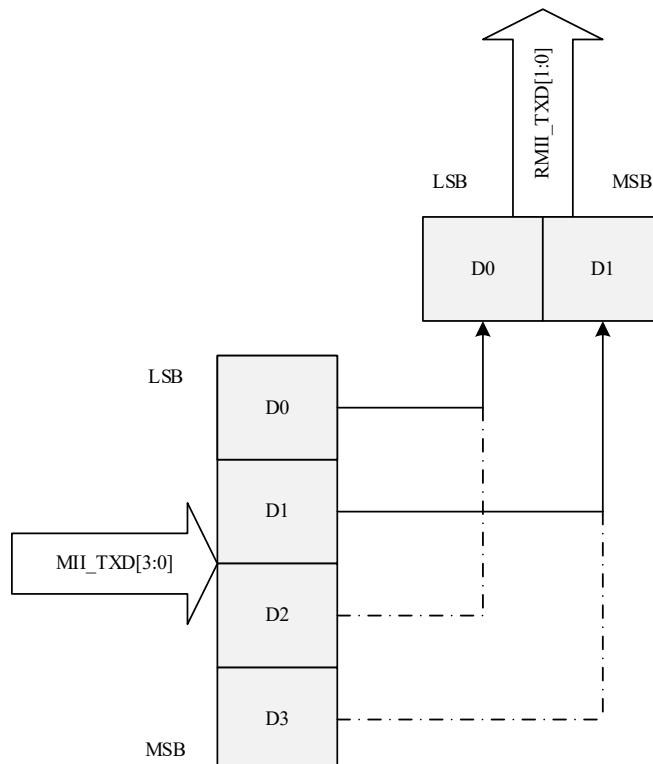


Figure 39-6 MII/RMII send bit sequence

Down Figure 39-7 ~ Figure 39-9 For the basic transmission timing diagram.

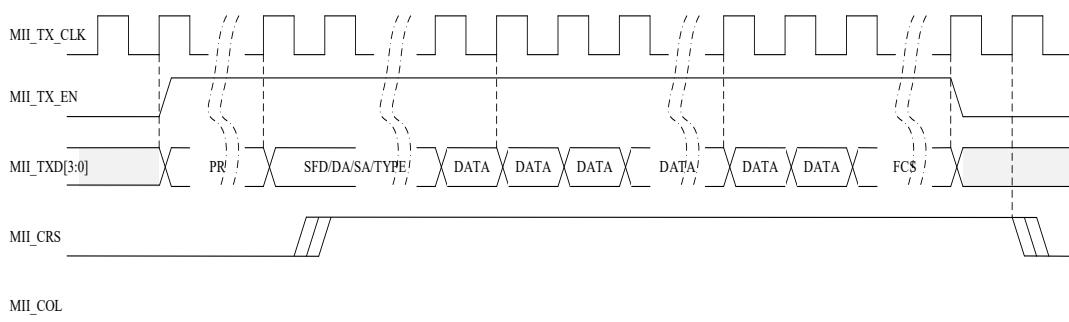


Figure 39-7 Conflict-free sending diagram

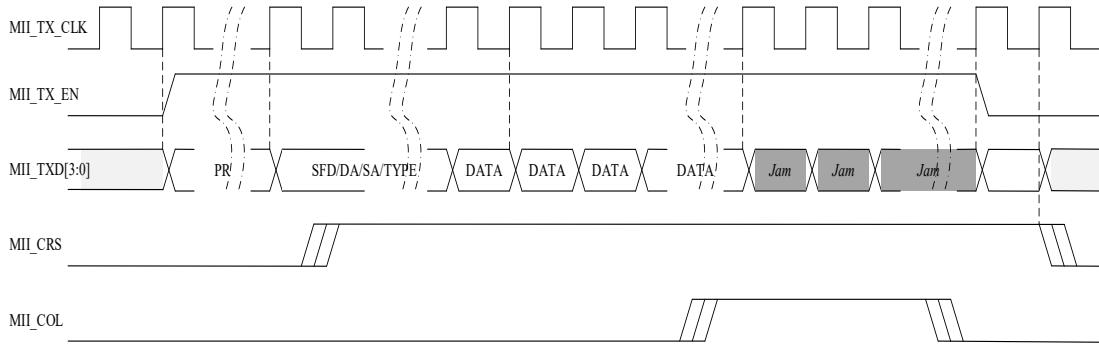


Figure 39-8 Conflict send map

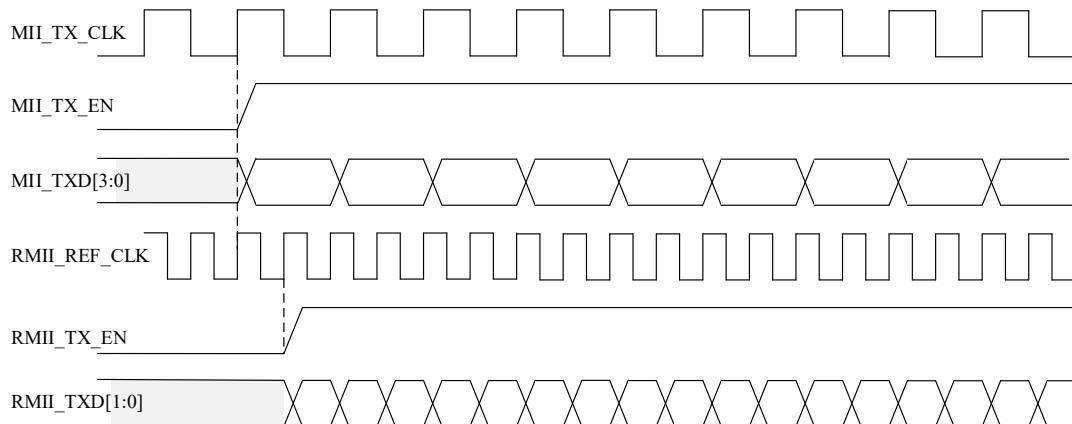


Figure 39-9 Transmission diagram in MII/RMII mode

39.4.1.3 MAC frame reception action

The frame received by the MAC controller will be sent to the Rx FIFO. When the reception reaches a certain number of conditions, it will instruct the RxDMA to perform data transmission, and the RxDMA can initiate a pre-configured burst transmission to the AHB interface.

The Ethernet MAC controller is equipped with a 2KB Rx FIFO. RxDMA has two modes of operation for transferring data from the Rx FIFO to system memory:

In threshold mode, when the number of bytes received by the Rx FIFO is greater than the configured receive threshold (configured using the RTC bits in the ETH_DMA_OPRMODR register) or when a complete packet is received, the data will be transmitted and its availability will be notified to the RxDMA. After RxDMA initiates a transfer to the AHB interface, the data transfer will continue from the Rx FIFO until the entire data packet is transferred. After the transfer of the EOF frame is completed, the receive status word is returned and sent to the RxDMA controller. In this mode, some erroneous frames are not discarded because an error status is received at the end of the frame, when data has been transferred from the Rx FIFO into system memory.

In store-and-forward mode, frames cannot be read until they are completely written to the Rx FIFO. In this mode, all erroneous frames will be dropped (if the kernel is configured to do so), so only valid frames will be read out and forwarded to the application.

Status is available immediately after data is received, so frames can be stored into the Rx FIFO as long as the Rx FIFO is not full.

When the MAC detects an SFD on the MII, it will initiate a receive operation. The MAC core will remove the header and SFD before continuing to process the frame. The header field is checked for filtering, and the FCS field is used to verify the frame's CRC. If the frame fails the address filter, the frame is discarded in the kernel.

Receive basic agreement

When the MAC receives data, it first removes the header and SFD of the received frame. After detecting the SFD, it starts to send the Ethernet frame data to the Rx FIFO, starting from the first byte (target address) after the SFD. If the IEEE1588 timestamp function is enabled, a snapshot of the system time will be taken whenever an SFD of any frame is detected on the MII. This timestamp is passed to the application unless the MAC filters out and discards the frame.

If the received frame length/type field is less than 0x600 and the MAC is programmed with the automatic strip CRC/PAD option, the MAC will send frame data (up to the amount specified in the length/type field) to the Rx FIFO and then start discarding words section (including the FCS field). If the length/type field is greater than or equal to 0x600, the MAC sends all received Ethernet frame data to the Rx FIFO, regardless of the value of the programmed automatic CRC removal option.

By default, the MAC watchdog timer is enabled, that is, frames exceeding 2048 bytes (DA+SA+LT+Data+PAD+FCS) will be cut off. This feature can be disabled by programming the Watchdog Disable (MWD) bit in the MAC Configuration Register (ETH_MAC_CONFIGR). However, even if the watchdog timer is disabled, frames larger than 16KB will still be cut off and given a watchdog timeout status.

Receive error handling

If the Rx FIFO is full before receiving EOF data from the MAC, an overflow will be declared and the entire frame will be discarded, at the same time the overflow counter in the Frame Loss Statistics Register (ETH_DMA_RFRCNTR) will be incremented. Due to overflow, the status will indicate that this is a partial frame. If the corresponding function is enabled (FEF and FUF bits in ETH_DMA_OPRMODR), the Rx FIFO can filter erroneous and undersized frames.

In the threshold mode, when reading the frame SOF from the Rx FIFO, the basic information of the current frame can be obtained, and by comparing with the basic settings of the Rx descriptor, it can be determined whether the frame is an error frame, and the entire error frame is discarded; In store-and-forward mode, all erroneous frames are filtered and discarded before data transfer.

CRC calculation and PAD removal

The MAC core will calculate the 32-bit CRC of the received frame (including the destination address field to the FCS field) and check for any CRC errors in the received frame. Regardless of whether

the CRC/PAD is automatically removed (controlled by the ACS bit of the ETH_MAC_CONFIGR register), the MAC will receive the entire frame to calculate the CRC check of the received frame, and feedback the check result to the application through the CRE bit of the receive descriptor RDES0.

Receive status word

At the end of receiving the Ethernet frame, the MAC output receiving status is sent to the receiving descriptor RDES0 through RxDMA for application reference query.

Receive pause frame

During frame transmission, the MAC will detect the reception of a pause frame and pause the transmission of the frame for the delay specified within the received pause frame (full duplex mode only). Pause frame detection can be enabled or disabled by the RFE bit in ETH_MAC_FLOCTRL. When receive flow control is enabled, it will start monitoring whether the destination address of the received frame matches the multicast address of the control frame (0x0180C2000001). If a match is detected (the destination address of the received frame matches the destination address of the reserved control frame), the MAC will decide whether to transmit the received control frame to the application based on the PCF bits in ETH_MAC_FLTCTRL.

The MAC will also decode the type, opcode and pause timer fields of the received control frame. If the byte count of the status indicates 64 bytes and there are no CRC errors, the MAC transmitter will pause the transmission of any data frames for the decoded pause time value multiplied by the timeslot (for 10/100 Mbps mode , both are 64-byte times). Meanwhile, if another pause frame with zero pause time value is detected, the MAC will reset the pause time and manage the new pause request. If the received control frame does not match the type field (0x8808), opcode (0x00001), and byte length (64 bytes), or there is a CRC error, the MAC does not generate a stall.

For pause frames with a multicast destination address, the MAC will filter the frame based on the address match; for pause frames with a unicast destination address, the MAC will filter the frames based on whether the DA field matches the contents of the MAC address register 0 and whether the UNP bit in ETH_MAC_FLOCTRL Set to 1 (detect pause frames with unicast destination address) to filter. The PCF bits in the ETH_MAC_FLTCTRL register control the filtering of control frames and the filtering of addresses.

COE engine

The Ethernet MAC controller supports detection and processing of IPv4 and IPv6 frames in received Ethernet frames to ensure data integrity. The receive COE engine is enabled by setting the IPCO bit in the ETH_MAC_CONFIGR register. The MAC identifies an IPv4 or IPv6 frame by checking for the presence of the value 0x0800 or 0x86DD in the type field of the received Ethernet frame. This identification method also applies to VLAN-tagged frames.

The receiving COE engine can calculate the IPv4 header checksum and check if it matches the received IPv4 header checksum. If there is any mismatch between the indicated payload type (EtherType field) and the IP header version, or the received frame has fewer bytes than the number indicated in the IPv4 header's Length field (available words in the IPv4 or IPv6 header less than 20 stanzas), the IP header error bit will be set.

The receiving COE engine can also identify TCP, UDP or ICMP payloads in received IP datagrams (IPv4 or IPv6) and correctly calculate the checksum fields of such payloads, including TCP/UDP for checksum calculation /ICMPv6 pseudo-header bytes and check that the received checksum field matches the calculated value. The result of this operation is given by the IP payload error bit in the receive descriptor RDES4. This status bit will also be set if the length of the TCP, UDP or ICMP payload does not match the expected payload length given in the IP header.

The COE engine will bypass fragmented IP datagram payloads, IP datagrams with security features, IPv6 routing headers, and payloads other than TCP, UDP, or ICMP.

Receive timing

In the RMII interface, each nibble inside the MAC is sent from the double bit received from the RMII to the MII. The nibble transmission sequence is as followsFigure 39-10 shown. The lower order bits (D0 and D1) are received first, followed by the higher order bits (D2 and D3).

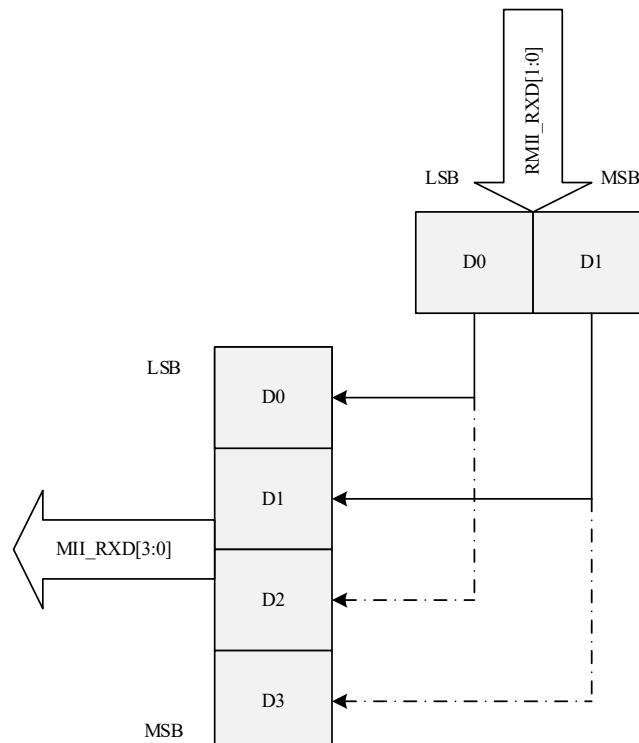


Figure 39-10 MII/RMII receive bit sequence

DownFigure 39-11 ~Figure 39-13 For the basic transmission timing diagram.

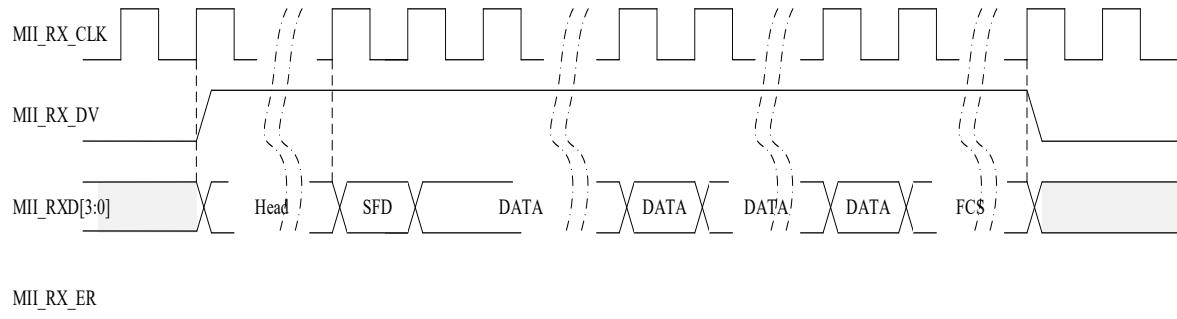


Figure 39-11 Error-free sending of graphs

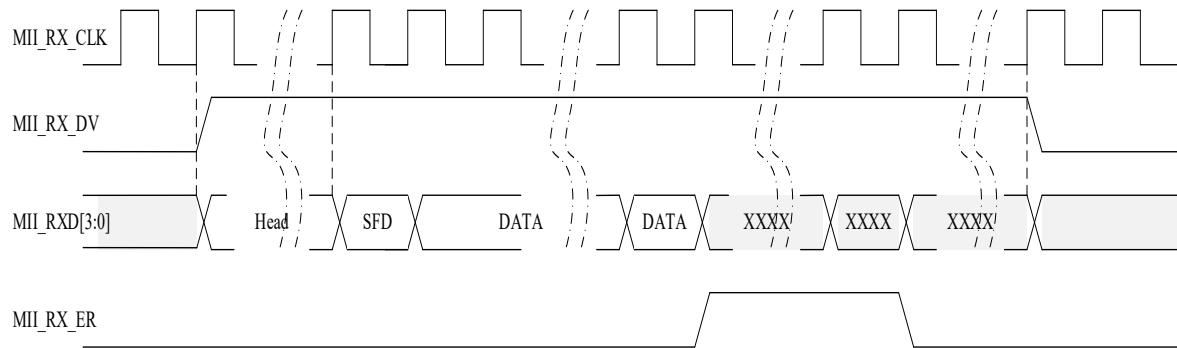


Figure 39-12 Error sending image

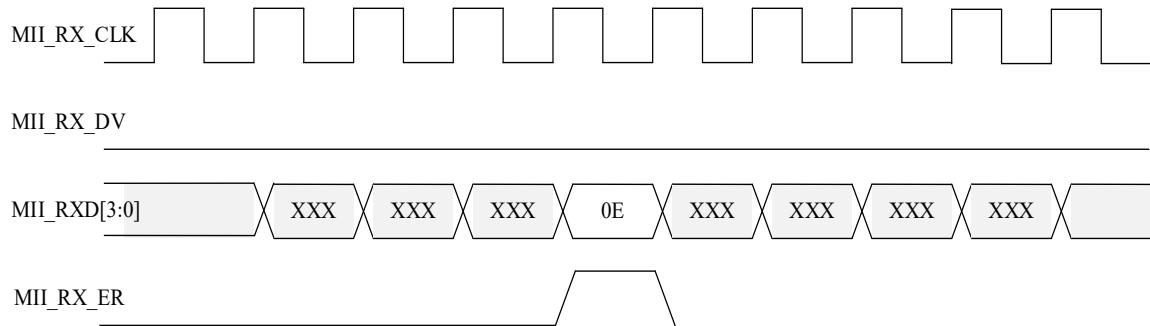


Figure 39-13 Reception diagram under false carrier indication

39.4.1.4 MAC frame filtering

The MAC can filter various frames received to select the frames required by the application. The frame filtering methods include the following:

1. Destination address and source address filtering of MAC frames
2. Identifier filtering of VLAN frames
3. Source and destination address filtering of LAY3 IP packets
4. Source port and destination port filtering of LAY4 TCP and UDP packets

The basic description of each frame filtering method is introduced in the following chapters:

MAC source address filtering

The MAC can perform perfect filtering based on the source address field of the received frame. By default, the MAC compares the SA field to the value programmed in the MAC address register. The MAC address registers 1~4 can be configured as the SA field for comparison by setting Bit 30 (SA) in the MAC address register to 1. If the SAF bit in the Frame Filtering Control Register (ETH_MAC_FLTCLR) is set, the MAC will discard frames that do not pass SA filtering. Otherwise, the result of SA filtering will be given by the status bits in the receive status word (see receive descriptor word RDES0).

Note: When the SAF bit is set to 1, the results of SA filtering and DA filtering are ANDed to determine whether the frame needs to be forwarded. This means that any filter that fails will drop the frame. Both filters must pass before the frame can be forwarded to the application.

When the ETHMAC controller compares each byte of the address value in the MAC address register 1~4 with the corresponding SA byte received, it can set the corresponding mask byte control bit in the register to 1 to mask the byte, so as to realize the SA Group address filtering.

For source address filtering, you can choose to invert the filtering matching result in the final output, that is, when the SA address is compared and matched, it is determined that the filtering fails. This function is controlled by the SAIF bit in the Frame Filter Control Register (ETH_MAC_FLTCLR).

MAC Destination Filtering - Unicast

MAC supports up to 5 MAC destination addresses for unicast filtering. If perfect filtering is selected (the HUC bit in the Frame Filter Control Register is set to 0), the MAC compares all 48-bit destination addresses of the received unicast address with the programmed MAC address to determine a match. By default, MAC address register 0 is always enabled, and other MAC address registers 1~4 are selected by individual enable bits.

When the ETHMAC controller compares each byte of the MAC address register 1~4 with the corresponding DA byte received, the corresponding mask byte control bit in the register can be set to 1 to mask the byte, thereby realizing the group address filtering of DA. .

MAC Destination Address Filtering - Multicast

The MAC is programmed to pass all multicast frames by setting the PMF bit in the Frame Filtering Control Register. If the PAM bit is reset, the MAC will perform filtering of multicast addresses based on the HMC bits in the Frame Filtering Register. In perfect filtering mode, the multicast address is compared with the programmed MAC address registers 1~4. This filtering method also supports group address filtering.

MAC Destination Address Filtering - Broadcast

In default mode, the MAC does not filter any broadcast frames. However, if the MAC is programmed to reject all broadcast frames by setting the DBF bit in the frame filter register to 1, any broadcast frames will be discarded.

For target address filtering (unicast, multicast, broadcast), you can also choose to invert the filtering matching result in the final output, that is, when the DA address is compared and matched, it is determined that the filtering fails. This function is controlled by the DAIF bit of the Frame Filter Control Register (ETH_MAC_FLTCTRLR), and also applies to Hash filter results.

MAC destination address hash filtering

Hash filtering is also supported for filtering of unicast destination addresses and multicast destination addresses. In Hash filtering mode (HUC bit/HMC bit 1), the MAC will use a 64-bit Hash table to perform imperfect filtering of unicast or multicast addresses, i.e. Hash filtering. For Hash filtering, the MAC will use the 6 high CRC bits of the received destination address to index the contents of the Hash table. A value of 000000 selects bit 0 in the selected register; a value of 111111 selects bit 63 in the hash table register. If the corresponding bit (indicated by the 6-bit CRC) is set to 1, the unicast frame or multicast frame is considered to have passed the hash filtering, otherwise the frame is considered to have failed the hash filtering.

Note: A CRC is a 32-bit value encoded using the following polynomial:

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

VLAN identifier filtering

When the received frame is a VLAN frame, the MAC can enable filtering of the VLAN identifier field by setting the VTFE bit in the Frame Filtering Control Register (ETH_MAC_FLTCTRLR). The comparison value for filtering is set in the VLFLT bit of the VLAN tag acceptance filter register (ETH_MAC_VTAFLTR). At the same time, by setting the VTAL bit of this register, it is possible to select whether to compare and filter the full VLAN identifier field or to compare only the lower 12-bit identifier.

For the filtering of the VLAN identifier field, you can also choose to reverse the filtering matching result in the final output, that is, it is determined that the filtering fails when the identifier field is compared and matched. This function is controlled by the VTIM bit of the VLAN tag acceptance filter register (ETH_MAC_VTAFLTR), and also applies to Hash filter results.

VLAN identifier hash filtering

Hash filtering can also be selected for filtering VLAN identifiers. In Hash filtering mode (VTHM bit of ETH_MAC_VTAFLTR is set to 1), the MAC will perform imperfect filtering of VLAN identifiers, i.e. Hash filtering, using a 16-bit Hash table. For hash filtering, the MAC will use the CRC bits of the VLAN identifier to index the contents of the hash table. When the value is 1, it indicates that the VLAN ID filter is passed; when the value is 0, it indicates that the VLAN ID filter is not passed.

Note: A CRC is a 32-bit value encoded using the following polynomial:

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

L3 source address & destination address filtering

It also supports filtering of source and destination addresses for the received LAY3 IPv4 and IPv6 packets. Through the L3SAM bit and L3DAM bit of the LAY3LAY4 control register, energy address filtering and target address filtering can be enabled respectively. The comparison value of the source address or destination address is set in the LAY3 address register (L3ADDR0~3), please refer to the description of this register for details.

The filtering of LAY3 IP addresses can select the high-order shielding function to realize the filtering of group addresses. The high-order mask filtering control of the source address and the destination address is controlled by the HSBM and HDBM bits of the LAY3LAY4 control register (ETH_MAC_L34CTRLR) respectively.

For the address filtering of LAY3 IP packets, you can also choose to invert the filtering matching results in the final output, that is, when the address fields are compared and matched, it is determined that the filtering fails. This function is controlled by the L3SAIM, L3DAIM bits in the LAY3LAY4 control register (ETH_MAC_L34CTRLR).

L4 source port & destination port filtering

The source port and destination port filtering is also supported for the received LAY4 layer TCP packets and UDP packets. Through the L4SPM bit and L4DPM bit of the LAY3LAY4 control register, the energy port filtering and the target port filtering can be enabled respectively. The comparison value of the source port or destination port is set in the LAY4 port register (L4PORTR), please refer to the description of this register for details.

For port filtering of LAY4 layer TCP packets and UDP packets, you can also choose to invert the filtering matching results in the final output, that is, when the port fields are compared and matched, it is determined that the filtering fails. This function is controlled by the L4SPIM, L3DPIM bits of the LAY3LAY4 control register (ETH_MAC_L34CTRLR).

39.4.1.5 MAC Loopback Mode

The MAC supports loopback of frames sent to its receiver. The MAC loopback feature is disabled by default and can be enabled by programming the LM bit in the MAC Control Register (ETH_MAC_CONFIGR).

39.4.1.6 MAC Management Counter (MMC)

The MAC Management Counter (MMC) carries a set of registers to collect statistics about received and transmitted frames. These include a control register (ETH_MMCCCTLR) for controlling the behavior of each register, two 32-bit registers (ETH_MMCRVSTSR and ETH_MMCTRSTS) containing the current receive and transmit statistics status, two 32-bit registers containing receive

and transmit interrupt control (ETH_MMC_RITCTRL and ETH_MMC_TITCTRL), and statistics registers for various frame types. These statistics registers can be accessed from the application, the purpose and meaning of these statistics registers are described in detail in the register chapter, please refer to the ETH_MMC register chapter.

The MAC management counter (MMC) updates the corresponding statistics register for the received frames that pass the address filtering, and the statistics of the discarded frames will not be updated unless the discarded frames are short frames less than 6 bytes (DA bytes cannot be fully received).

If the frame is sent successfully, the sent frame is considered a "good frame". That is, a frame sent is considered a "good frame" if the frame sending process is not aborted by the following errors:

- Jabber timed out
- No carrier / carrier loss
- delayed conflict
- frame underflow
- excessive delay
- excessive conflict

If the frame is received successfully, the received frame is considered a "good frame". That is, a received frame is considered a "good frame" if it does not have the following errors:

- CRC error
- Short frame (shorter than 64 bytes)
- Alignment error (10/100 Mb/s only)
- wrong length (untagged frames only)
- out of bounds (untagged frames only, exceeds max size)
- MII_RXER input error

Note: The maximum frame size depends on the frame type, as follows:

- Maximum number of bytes for an identifierless frame: 1518
- Maximum number of bytes of VLAN frame: 1522

39.4.1.7 MAC Power Management (PMT)

The ETHMAC controller supports a power management (PMT) mechanism. The PMT module can be enabled through the Remote Wakeup Frame Enable bit and the Magic Packet Enable bit. These enable bits (WKEN and MPEN) are located in the ETH_MAC_PMTCTRL register and can be programmed by the application. When Power Down mode is enabled (ETH_MAC_PMTCTRL.PWDN=1), the MAC will discard all received frames and will not forward these frames to the application. The MAC controller will exit Power Down mode only when the corresponding detection bit is enabled and the corresponding magic packet or remote wake-up frame is received.

Remote Wakeup Frame Filtering

The ETHMAC controller has 8 wakeup frame filter registers. To perform a write to each register, the Remote Wakeup Frame Filter Register (ETH_MAC_RTWKFFR) needs to be loaded value by value. Loading the Remote Wakeup Frame Filter register eight times in a row loads the Remote Wakeup Frame Filter register with the desired value. The read operation is the same as the write operation. To read eight values, the wake-up frame filter register must be read eight times before reaching the last register. Each read/write operation will point the wake-up frame filter register pointer to the next filter register. By reading the RTWKPT bit of the PMT control status register (ETH_MAC_PMTCTRLR), you can know which filter register the current read and write operation is on. At the same time, you can reset the internal pointer by setting the RTWKFR bit of the register. After reset, the remote wake-up frame can be filtered. Register to operate again. DownFigure 39-14 Shown are the 8 remote wake-up frame filter registers inside the ETHMAC.

<i>ETH_MAC_RTWKFFR 0</i>	FLT0							
	Byte Mask							
<i>ETH_MAC_RTWKFFR 1</i>	FLT1							
	Byte Mask							
<i>ETH_MAC_RTWKFFR 2</i>	FLT2							
	Byte Mask							
<i>ETH_MAC_RTWKFFR 3</i>	FLT3							
	Byte Mask							
<i>ETH_MAC_RTWKFFR 4</i>	-	FLT3	-	FLT2	-	FLT1	-	FLT0
		Command		Command		Command		Command
<i>ETH_MAC_RTWKFFR 5</i>	FLT3		FLT2		FLT1		FLT0	
	Offset		Offset		Offset		Offset	
<i>ETH_MAC_RTWKFFR 6</i>	FLT1			FLT0			CRC16	
	CRC16			CRC16				
<i>ETH_MAC_RTWKFFR 7</i>	FLT3			FLT2			CRC16	
	CRC16							

Figure 39-14 Remote Wakeup Frame Filter Register

The functions of each part in the above figure are described as follows:

- Filter i Byte Mask

This register defines which bytes of the frame are detected by filter i to determine whether the frame is a wake-up frame ($i=0\sim3$). The MSB (bit 31) must be zero, and bits $j[30:0]$ are the byte mask. If bit j (number of bytes) of the byte mask is set to 1, the filter i offset + j of the incoming frame is processed by the CRC module; otherwise the filter i offset + j is ignored.

- Filter i Command (Command)

This 4-bit command controls the filter i operation ($i=0\sim3$). Bit3 is the address type selection. When it is set, only multicast frames are detected, otherwise, only unicast frames are detected;

Bit2 and Bit1 are reserved bits; Bit0 is the enable bit of filter i. When set, filter i is enabled, and vice versa. filter i.

- Filter i Offset (Offset)

This register defines the offset of the frame to be detected by filter i (i=0~3). The 8-bit pattern offset is the offset of the first byte of filter i to be detected. The minimum allowed value is 12, which represents the 13th byte of the frame (offset value 0 represents the first byte of the frame).

- Filter i CRC-16 (CRC16)

This register contains the pre-written CRC-16 value (i=0~3) for comparison with the CRC-16 value calculated from the frame data (after the filter i Offset and Byte Mask corresponds).

The generator polynomial of CRC-16 is: $G(x) = x^{16} + x^{15} + x^2 + 1$.

Remote wake-up frame detection

When the MAC is in Power Down mode and has set the remote wake-up bit WKEN in the PMT control status register (ETH_MAC_PMTCTRLR), the MAC can resume normal operation after receiving a remote wake-up frame. The PMT supports four programmable filters that can be configured by the application for different received frame modes. A wake-up frame can be received if the incoming frame passes the filter's address filter and the filter CRC-16 matches the incoming detection pattern. It is only necessary to check whether the wake-up frame has length errors, FCS errors, end-of-frame errors, MII errors, and collisions to ensure that it is not a short frame. Even if the wake-up frame length exceeds 512 bytes, as long as the CRC value of the frame is valid, the frame is valid.

The Wakeup Frame Detection Status bit (WKFR) in the ETH_MAC_PMTCTRLR register is updated for each remote wakeup frame received. In addition, a PMT interrupt (if enabled) is generated to indicate that a remote wake-up frame has been received.

Magic Packet Inspection

Magic Packet is based on AMD's Magic Packet technology to power up devices in sleep mode on the network. The MAC receives a specific packet called a magic packet, and the destination address of this packet is a node on the network. The MAC controller only checks magic packets destined for this device or multicast address to determine if these packets meet the wake-up requirement. Detect the magic data packet that has passed the address filter (unicast or multicast) to determine whether it conforms to the remote wake-up data format, that is, the data packet with all bits of 6 bytes of data is all "1" plus repeated 16 times. the MAC address.

The application enables magic packet wakeup by writing a 1 to the MPEN bit in the ETH_MAC_PMTCTRLR register. The PMT module continuously monitors every frame whose destination address is the node corresponding to the specified magic packet pattern. Checks that the destination address and source address fields of each received frame are followed by a synchronous data stream of the form 0xFFFFFFFFFFFF. After that, the PMT module checks the frame for the

presence of a MAC address that is repeated 16 times without any disconnection or interruption. If there is a break in the address repeated 16 times, the incoming frame is scanned again for the presence of the 0xFFFFFFFFFFFF form. These 16 repetitions can be located anywhere in the frame, but must follow the isochronous data stream (0xFFFFFFFFFFFF). The device can receive multicast frames as long as it detects a MAC address that is repeated 16 times. For example, if the node MAC address is 0x001122334455, the data sequence of the MAC scan is:

*Destination Address Source Address... FFFF FFFF FFFF
0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455
0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455
0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455
0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455
CRC*

is considered to have received a magic wake-up packet. Magic packet detection in the ETH_MAC_PMTCLR register is updated for received magic packets. Additionally, a PMT interrupt (if enabled) is generated to indicate that a magic packet has been received.

System Status During Power Loss

1. The MAC receiver should be kept enabled during Power Down mode, the reason is that the magic packet/remote wake-up frame is to be detected relative to the reception at this time, and the MAC receiver is enabled by setting the RE bit in the ETH_MAC_CONFIGR register to 1.
2. The MAC transmitter should be kept off during Power Down mode. Turn off the MAC transmitter by clearing the TE bit in the ETH_MAC_CONFIGR register.
3. Ethernet DMA should be disabled during Power Down mode as there is no need to copy magic packets/Wake-On-LAN frames to system memory at this time. Turn off TxDMA and RxDMA by clearing the STT and STR bits in the ETH_DMA_OPRMODR register.

The recommended power-down and wake-up sequence is as follows:

1. Disable TxDMA and wait for all previous frame transmissions to complete
2. Disable the MAC transmitter and MAC receiver by clearing the RE and TE bits in the ETH_MAC_CONFIGR register
3. Waiting for RxDMA to clear all frames in Rx FIFO
4. Disable RxDMA
5. Magic packet/remote wakeup frame detection is enabled by setting the MPEN/WKEN bit in the ETH_MAC_PMTCLR register
6. MAC power-down mode is enabled by setting the PWDN bit in the ETH_MAC_PMTCLR register
7. The MAC receiver is enabled by setting the RE bit in the ETH_MAC_CONFIGR register
8. Waiting to receive magic packets and remote wakeup frames
9. The Ethernet controller exits power-down mode when a valid wake-up frame is received
10. Read ETH_MAC_PMTCLR to clear PMT event flag, enable MAC transmitter, TxDMA and RxDMA

RxDMA

39.4.2 ETH_PTP function

The IEEE1588 standard defines a protocol that supports precise clock synchronization in measurement and control systems using technologies such as network communications, local computing, and distributed objects. This protocol is suitable for systems that communicate using local area networks (including but not limited to Ethernet) that support multicast messaging. This protocol is used to synchronize non-uniform systems that contain clocks with inherently varying accuracy, resolution, and stability. The protocol supports system-level synchronization accuracy in the sub-second range and requires minimal network and local clock computing resources. This message-based protocol is called the Precision Time Protocol (PTP), and it is delivered over UDP/IP. Systems or networks are categorized as master and slave nodes for distributing timing/clock information. This protocol synchronizes the slave node and the master node by exchanging PTP messages, such as Figure 39-15 shown.

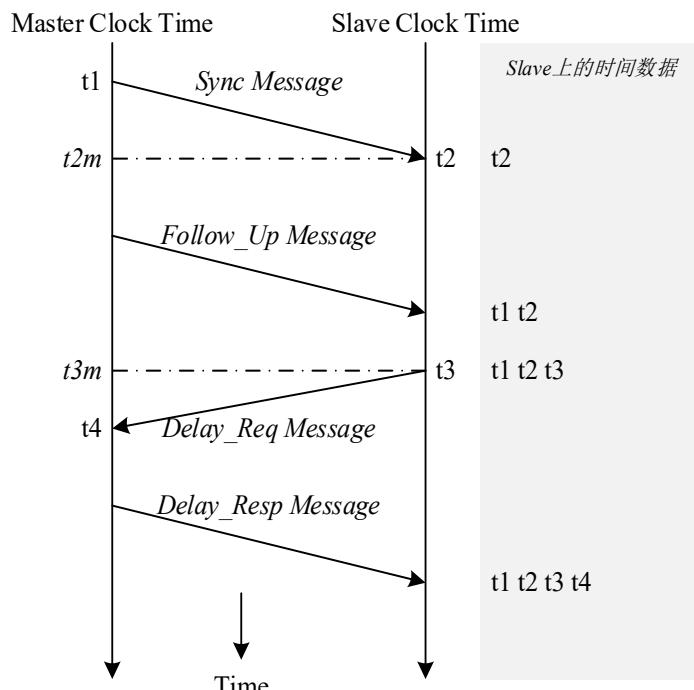


Figure 39-15 Clock synchronization diagram

1. The master node broadcasts PTP synchronization messages to all its nodes. The synchronization message contains the reference time information of the master node. The time when the message leaves the master node system is t1. For Ethernet ports, this time must be captured through the MII interface
2. The slave node receives the synchronization message and uses its reference timing to capture the exact time t2
3. The master node will then send a Follow_up message containing t1 information to the slave node for later use

4. The slave node sends a Delay_Req message to the master node, marking the exact time t3 when the frame leaves the MII
5. The master node receives the message and captures the exact time t4 when the message entered its system
6. The master node sends the t4 information in the Delay_Resp message to the slave node
7. The slave node uses the four values t1, t2, t3 and t4 to synchronize its local reference timing with that of the master node

While most protocols are implemented in software on top of the UDP layer, hardware completion is required to capture the exact time when a particular PTP packet enters or leaves an Ethernet port via the MII interface.

39.4.2.1 PTP reference timing source

According to the definition of the IEEE1588 specification, to obtain a time snapshot, the kernel needs a reference time in 64-bit format (divided into two 32-bit channels, the upper 32 bits represent the seconds of the time, and the lower 32 bits represent the sub-seconds of the time).

The PTP reference clock input is used to internally generate a reference time (also known as system time) and capture timestamps. The frequency of this reference clock must be equal to or greater than the resolution of the timestamp counter. The synchronization accuracy target between the master node and each slave node is about 100 ns. The accuracy depends on the PTP reference clock input period, the characteristics of the oscillator (drift), and the frequency of the synchronization process.

Due to the synchronization from the Tx and Rx clock input domains to the PTP reference clock domain, the uncertainty of the timestamp latch value is 1 reference clock cycle. Adding in the uncertainty due to resolution adds half the time stamp period.

39.4.2.2 PTP packet type

After the time stamp function is enabled, the TSPMTSEL bit in the time stamp control register (ETH_PTP_TSPCTLR) can be used to control which message or type of message is to be time stamped. The specific correspondence is as followsTable 39-7 .

Table 39-7 Timestamp snapshot target message

TSPMTSEL[3:0]	Timestamp-enabled packet type
00X0	SYNC, Follow_Up, Delay_Req, Delay_Resp
0001	SYNC
0011	Delay_Req
01X0	SYNC, Follow_Up, Delay_Req, Delay_Resp Pdelay_Req, Pdelay_Resp, Pdelay_Resp_Follow_Up
0101	SYNC, Pdelay_Req, Pdelay_Resp
0111	Delay_Req, Pdelay_Req, Pdelay_Resp

TSPMTSEL[3:0]	Timestamp-enabled packet type
10XX	SYNC, Delay_Req
11XX	Pdelay_Req, Pdelay_Resp

39.4.2.3 PTP frame sending function

Timestamps are captured when the frame's SFD is output on the MII. For frames that require time stamp capture, each transmitted frame can be marked to indicate whether it is necessary to capture the time stamp for that frame.

A PTP frame can be identified without processing the transmit frame, and frame control can be performed through the control bits in the transmit descriptor.

The captured timestamp is returned to the application in the same way as the frame status is provided. Timestamps are sent back into the corresponding transmit descriptors along with the frame's transmit status, automatically associating timestamps with specific PTP frames. The 64-bit timestamp information is written back to the TDES6 and TDES7 fields, where TDES6 holds the 32 least significant bits of the timestamp.

39.4.2.4 PTP frame reception function

When the IEEE1588 timestamp function is enabled, the Ethernet MAC will capture the timestamps of all frames received on the MII. The MAC provides a timestamp when the frame reception process is complete.

The captured timestamp is returned to the application in the same way as the frame status is provided. The timestamp is sent back into the corresponding receive descriptor along with the receive status of the frame. The 64-bit timestamp information is written back to the RDES6 and RDES7 fields, where RDES6 holds the 32 least significant bits of the timestamp.

39.4.2.5 PTP system time calibration

Update 64-bit PTP time using the PTP input reference clock. This PTP time can be used as a clock source to get a snapshot (time stamp) of Ethernet frames sent or received on the MII. The system time timer can be initialized or calibrated using a coarse calibration method or a fine calibration method.

When using the coarse calibration method, the initial value or offset value is written to the timestamp update register. For initialization, the value in the time stamp update register is written to the system time counter; for system time calibration, the offset value (time stamp update register) is added to or subtracted from the system time.

When using the fine calibration method, the offset of the slave reference clock frequency relative to the master clock (as defined in IEEE1588) is calibrated over a period of time instead of a single clock cycle as in the coarse calibration method. A longer calibration time helps to maintain linear

time and does not cause drastic changes (or large jitter) in the reference time between PTP sync message intervals. In this method, an accumulator is used to sum the contents of the base addend register (ETH_PTP_TSPADDR), such as Figure 39-16 shown. The arithmetic carry generated by the accumulator is used as a pulse to increment the system time counter. The accumulator and basic addend registers are both 32-bit registers. Here, the accumulator acts as a high precision frequency multiplier or divider.

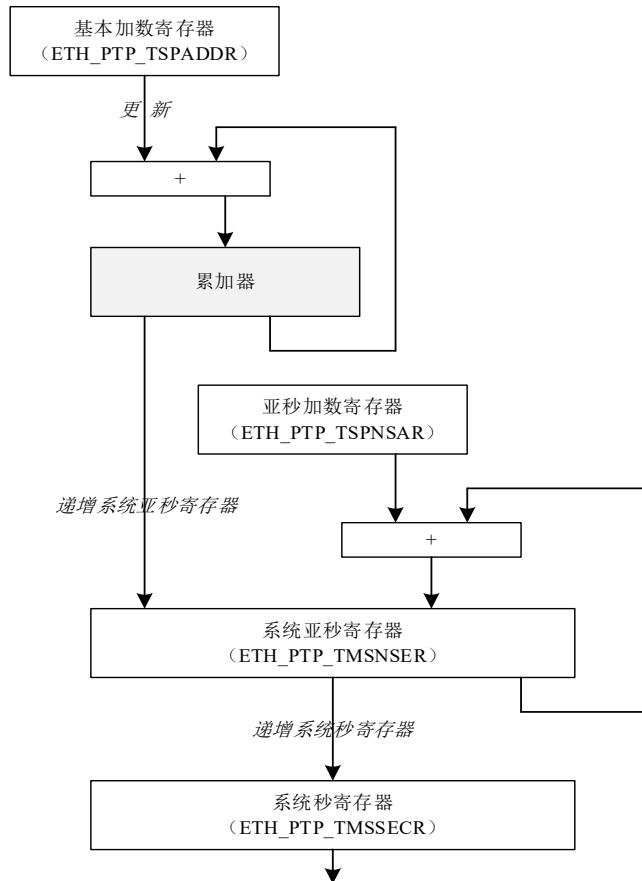


Figure 39-16 System time calibration

The system time update logic needs to use a clock frequency of 50MHz to achieve 20ns accuracy. The divider ratio is the ratio of the reference clock frequency to the desired clock frequency. If the reference clock frequency is 66MHz, it can be known from calculation that the frequency division ratio is $66\text{MHz}/50\text{MHz}=1.32$, and the default addend value set in the basic addend register is $2^{32}/1.32$, which is equivalent to 0xC1F07C1F; To 65MHz, the frequency division ratio is $65\text{MHz}/50\text{MHz}=1.3$, and the value to be set in the basic addend register is $2^{32}/1.3$, which is equivalent to 0xC4EC4EC4; if the clock is shifted upward to 67MHz, the basic addend register must be set Set to 0xBF0B7672. When the clock offset is zero, the default additive value that should be programmed is 0xC1F07C1F ($2^{32}/1.32$).

The value used to increment the subsecond adder register is set to 43 decimal. This allows the system time to be accurate to 20ns (in other words, the incremental step time is 20ns).

The software must calculate the frequency offset based on the synchronization message and update the basic addend register accordingly. First, set up the slave clock using FreqCompensationValue0 in the base addend register. The formula for calculating this value is as follows:

$$\text{FreqCompensationValue0} = 2^{32} / \text{FreqDivisionRatio}$$

If it is initially assumed that the MasterToSlaveDelay is the same for successive synchronization messages, the following algorithm must be applied. After a few sync cycles, the frequency locks. The slave can then determine the exact value of MasterToSlaveDelay and resynchronize with the master using the new value.

The algorithm is as follows:

- At MasterSyncTime(n), the master clock sends a synchronization message to the slave clock. The slave clock receives the message when its local clock is SlaveClockTime(n), and uses the following formula to calculate MasterClockTime(n):

$$\text{MasterClockTime}(n) = \text{MasterSyncTime}(n) + \text{MasterToSlaveDelay}(n)$$

- The calculation formula of the master clock count MasterClockCount(n) of the current synchronization cycle is as follows:

$$\text{MasterClockCount}(n) = \text{MasterClockTime}(n) - \text{MasterClockTime}(n-1)$$

(assuming MasterToSlaveDelay is the same for sync periods n and n-1)

- The calculation formula of the slave clock count SlaveClockCount(n) of the current synchronization cycle is as follows:

$$\text{SlaveClockCount}(n) = \text{SlaveClockTime}(n) - \text{SlaveClockTime}(n-1)$$

- The calculation formula of the master-slave clock count difference SlaveClockCount(n) of the current synchronization cycle is as follows:

$$\text{ClockDiffCount}(n) = \text{MasterClockCount}(n) - \text{SlaveClockCount}(n)$$

- The calculation formula of the frequency division factor FreqScaleFactor(n) of the slave clock is as follows:

$$\text{FreqScaleFactor}(n) = (\text{MasterClockCount}(n) + \text{ClockDiffCount}(n)) / \text{SlaveClockCount}(n)$$

- The calculation formula of the frequency compensation value FreqCompensationValue(n) of the addend register is as follows:

$$\text{FreqCompensationValue}(n) = \text{FreqScaleFactor}(n) \times \text{FreqCompensationValue}(n-1)$$

In theory, the algorithm achieves lock within one synchronization cycle, but multiple cycles may be required due to constantly changing network propagation delays and operating conditions.

The algorithm is self-calibrating: If for some reason the slave clock originally set by the master clock is incorrect, the algorithm takes more synchronization cycles to calibrate it.

39.4.2.6 PTP system time generation initialization

The timestamp function is enabled by setting the TSPEN bit in the timestamp control register (ETH_PTP_TSPCTRLR). But then the timestamp counter must be initialized to start the timestamp operation. The setting sequence is as follows:

1. Interrupts are triggered by masking timestamps by setting the TSPIM bit in the ETH_MAC_INTMSKR register to 1
2. Program the TSPEN bit in the ETH_PTP_TSPCTRLR register to enable time stamping
3. Program the sub-second adder register (ETH_PTP_TAPNSAR) according to the PTP clock frequency
4. Program the base addend register (ETH_PTP_TSPADDR) and set the TSPADUP bit of the timestamp control register (base addend register update)
5. Poll the Timestamp Control Register to confirm that the TSPADUP bit is cleared
6. To select the precision calibration method, set the TSPUPSEL bit of the time stamp control register to 1
7. Program the timestamp update seconds register and timestamp update subsecond register with the appropriate time value
8. Set the time stamp control register TSPINI bit to 1 (time stamp initialization)
9. After initializing the timestamp counter with the value written in the timestamp update register, the timestamp counter starts running
10. Enable the MAC receiver and transmitter for the timestamp function to function properly

39.4.2.7 PTP PPS output

The PTP module can output the generated internal clock to the ETH_PTP_PPS port by means of PPS (PPS0) (continuous output mode), which is used for clock synchronization of all nodes in the network system. At the same time, there are two PPS (PPS0 and PPS1) outputs (Single output mode), which can be used as event output to control other modules.

In PPS continuous output mode, the PPS0 channel can generate a series of specific pulses or clocks for external network node clock synchronization after the system time has a second carry (sub-second time overflow). The setting of the specific output pulse or output clock frequency is controlled by the PPSFRE0 bit of the PPS output control register (ETH_PTP_PPSCTRLR). Only the PPS0 channel supports continuous output mode.

In the PPS single output mode, if the system time is equal to the time set by the target time register 0 or the target time register 1, a pulse can be generated on the PPS0 or PPS1 channel respectively. This function will be enabled only when the TT0SEL bit or TT1SEL bit of the PPS output control register (ETH_PTP_PPSCTRLR) is set; when the PPSFRE0 or PPSFRE1 bit of the PPS output control register (ETH_PTP_PPSCTRLR) is set, the function will not be enabled. Activated. After the pulse is

generated, if you want to generate the pulse output again, you need to reset the target time register 0/1 and the PPSFRE0/1 bit of the PPS output control register.

For example, users can link the event output of PPS0 or PPS1 in single output mode with Timer2 according to their needs. For the specific implementation method, please refer to the Timer2 chapter and the INTC chapter. This linkage with Timer2 mainly has the following two functions:

1. When the system time is equal to the target time, the trigger interrupt provided by the MAC causes a known delay, causing uncertainty in the command execution time. An accurate interrupt processing delay time can be obtained through the timer function of Timer2 during the period from interrupt triggering to command execution.
2. Offset calibration of system time. PPS outputs a fixed-time pulse, detects the fixed-time pulse through the pulse width measurement function of Timer2, and feeds back and adjusts the system time of the PPS according to the measurement result of Timer2, so as to realize the calibration of the internal system time of the PTP.

39.4.3 ETH_DMA function

The ETHMAC controller DMA has independent transmit and receive engines. The transmit engine transfers data from the system memory to the Tx FIFO, and the receive engine transfers data from the Rx FIFO to the system memory. DMA can efficiently transfer data between FIFO and system memory through descriptors without CPU intervention at all. The DMA controller can be programmed by the application to generate CPU interrupts upon completion of frame transmit and receive operations and other normal/error conditions. Therefore, it can be seen that there are two data structures for communication between the ETHMAC controller DMA and the MCU core:

- Control and Status Registers
- Descriptor list and data buffer

The DMA can not only transmit the data frame received by the MAC to the data buffer of the system, but also can send the data in the system data buffer to the MAC, and the descriptor contains pointers to these data buffers. DMA has two descriptor lists: One list of Rx descriptors for receive and one list of Tx descriptors for transmit. The base addresses of the Rx descriptor list and the Tx descriptor list are set by the descriptor list registers (ETH_DMA_RXDLADR and ETH_DMA_TXDLADR), respectively. The descriptor list is a forward-linked list (either implicitly or explicitly), with the last descriptor pointing back to the first descriptor to form a ring structure. The descriptor list is located in the physical storage space of the system.

Each descriptor can point to up to two buffers, so that two physically addressed buffers can be used instead of two contiguous buffers in memory. Explicit linking of descriptors is done by configuring the second address linked in the Rx descriptor and Tx descriptor (RDES1[14] and TDES0[20]).

The data buffer is also located in the physical storage space of the system and usually consists of whole or partial frames, but not more than a single frame. The data buffer contains only data, and the state of data communication is stored in the descriptor. When the end of the frame is detected, the DMA will jump to the next frame buffer, the data link can store the frame across multiple data buffers, and the data link can be enabled or disabled. Descriptors are structured in two ways: Ring structure and link structure, which are implemented as follows Figure 39-17 shown.

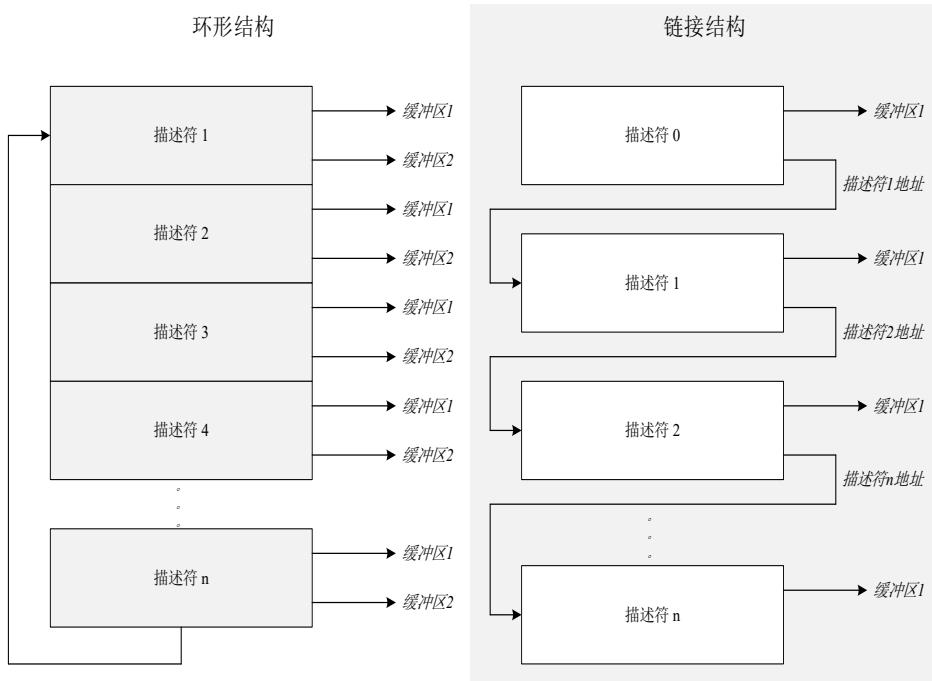


Figure 39-17 Descriptor structure

39.4.3.1 DMA initialization

The initialization steps of DMA are as follows:

1. Write to ETH_DMA_BUSMODR to set bus access parameters
2. Write to the ETH_DMA_INTENAR register to mask unnecessary interrupt sources
3. Create a list of descriptors, write to the ETH_DMA_RXDLADR and ETH_DMA_TXDLADR registers, and provide the DMA with the starting address of each list
4. Writes to the MAC registers set the desired filtering options
5. Write to the ETH_MAC_CONFIGR register to configure and enable transmit and receive modes of operation
6. Write to the ETH_DMA_OPRMODR register to set the STT and STR bits to initiate transmit and receive
7. The send and receive engines enter the running state and attempt to obtain descriptors from the corresponding descriptor lists, and the two engines then begin processing receive and transmit operations. Send and receive processes are independent of each other and can be started or stopped independently

Bus burst access

The DMA will attempt to perform fixed-length burst transfers on the AHB Master interface (when the FBST bits in ETH_DMA_BUSMODR are configured accordingly). The maximum length of the burst is indicated and limited by the TPBL or RPBL field (ETH_DMA_BUSMODR [13:8] and [22:17]). For the 16 bytes to be read, receive and transmit descriptors are always accessed with the largest possible burst size.

TxDMA will only initiate a data transfer when there is enough space in the Tx FIFO to hold the configured burst or the number of bytes before the end of the frame (when the frame is shorter than the configured burst length). TxDMA will indicate the starting address and the required number of transfers to the AHB Master interface. When the AHB interface is configured for fixed-length bursts, the best combination of INCR4, INCR8, INCR16, and SINGLE is used to transfer data. Otherwise (non-fixed-length bursts) use INCR (undefined length) and SINGLE to transmit data.

RxDMA will only initiate a data transfer when there is enough data in the Rx FIFO for the configured burst, or when the end of frame is detected in the Rx FIFO (when the frame is shorter than the configured burst length). RxDMA will indicate the starting address and the required number of transfers to the AHB Master interface. When the AHB interface is configured for fixed length bursts, data will be transferred using the best combination of INCR4, INCR8, INCR16 and SINGLE, if the end of frame is reached before the end of the fixed burst on the AHB interface, an empty transfer will be performed to complete Fixed-length burst transfers. Otherwise (non-fixed length bursts) use INCR (undefined length) and SINGLE to transfer data.

When the AHB interface is configured for address-aligned tick (ETH_DMA_BUSMODR.AAL=1), both DMA engines ensure that the first burst transfer initiated by AHB is less than or equal to the configured PBL size. This way, all subsequent beats will start at an address aligned with the configured PBL. Since the AHB interface has more transfers than INCR16, DMA can only align addresses with a maximum beat of 16 (PBL>16).

Data buffer alignment

Transmit and receive data buffers do not have any restrictions on starting address alignment. The start address of the buffer can be aligned to any of the four bytes. However, DMA always initiates transfers when the address is aligned with the bus width, and transfers dummy data on byte lanes that are not needed. This usually occurs during the beginning or end of the transmission of an Ethernet frame.

- Example of buffer read operation:

If the transmit buffer address is 0x00000FF2 and 15 bytes need to be transferred, the TxDMA will read 5 full words from address 0x00000FF0, but discard or ignore the extra bytes when transferring the data to the Tx FIFO (the first two byte). Likewise, the last 3 bytes of the last

transfer will be ignored. TxDMA always ensures that the full 32-bit data is transferred to the Tx FIFO, unless it is end-of-frame.

- Example of buffer write operation:

If the receive buffer address is 0x00000FF2 and 16 bytes of the received frame need to be transferred, RxDMA will write 5 full 32-bit data starting at address 0x00000FF0. However, the first 2 bytes of the first transfer and the last 2 bytes of the third transfer will contain null data.

Buffer size calculation

The Ethernet controller DMA does not update the buffer size field in the transmit and receive descriptors, only the status field of the descriptor. The application must use the driver to calculate the size of the buffer.

TxDMA transfers the exact number of bytes (indicated by the buffer size field in TDES1) to the MAC core. If the descriptor is marked as the first descriptor (the TFS bit in TDES0 is set), the DMA will mark the first transfer of the buffer as the start of frame. If the descriptor is marked as the last descriptor (the TLS bit in TDES0 is set), the DMA will mark the last transfer of the data buffer as the end of frame.

RxDMA continues to transfer data to the buffer until the buffer is full or the end of frame is received. When the RFS bit of a descriptor is set to 1, if the descriptor is not marked as the last descriptor (RLS bit in RDES0), the buffer corresponding to the descriptor will fill up, and the amount of valid data is determined by the buffer size in RDES1. If the descriptor is marked as the last descriptor, the buffer will not fill up and the amount of valid data will be represented by the result of the buffer size field minus the data buffer pointer offset (FRAL bit in RDES0). When the data buffer pointer is aligned with the width of the data bus, the offset is zero.

Note: Even when the start address of the receive buffer is not aligned with the width of the system data bus, the system should allocate a receive buffer whose size is aligned with the width of the system bus. For example, if the system allocates a receive buffer with a start address of 0x1000 and a size of 1024 bytes (1 KB), software can program the buffer start address in the receive descriptor to have an offset of 0x1002. RxDMA writes frames to this buffer with null data in the first two locations (0x1000 and 0x1001). The actual frame is written starting at location 0x1002. So although the buffer size is programmed to be 1024 bytes, the actual useful space for this buffer is 1022 bytes due to the start offset address.

DMA arbiter

The arbiter within the Ethernet controller DMA arbitrates between access to the AHB Master interface by the transmit and receive channels. Two types of arbitration can be used: Round robin and fixed priority. If cyclic priority arbitration is selected (ETH_DMA_BUSMODR.DMAA=0), the arbiter will allocate the data bus according to the priority mode and ratio set by the ETH_DMA_BUSMODR.TXPR and PTAT bits when the transmit and receive DMA requests for access

at the same time; if fixed priority arbitration is selected (ETH_DMA_BUSMODR.DMAA=1), the arbiter will allocate the data bus according to the priority set by the ETH_DMA_BUSMODR.TXPR bit when sending and receiving DMA requesting access at the same time.

39.4.3.2 DMA error response

For any data transfer initiated by a DMA channel, if the slave gives an error response, the corresponding DMA will stop all operations and update the error bit and fatal bus error bit in the status register (ETH_DMA_DMASTR). At this point, the DMA controller can only resume operation after a soft or hard reset of the peripherals and reinitialization of the DMA.

39.4.3.3 DMA transmit configuration

TxDMA operation

In the process of continuous frame transmission, TxDMA has different second frame processing timings according to different settings of the OSF bits of the operation mode register ETH_DMA_OPRMODR. When the OSF bit is 1, after the sending process completes the transmission of the first frame, it will immediately poll the send descriptor list of the second frame. If the second frame is valid, the sending process will be in the state of writing the first frame. A second frame is sent before the message. This transmission process can acquire two frames at the same time without closing the state descriptor of the first frame, which is called OSF mode, otherwise it is non-OSF mode.

The operation sequence of the TxDMA engine under ETH_DMA_OPRMODR.OSF=0 is as follows:

1. After setting the corresponding Ethernet frame data in the data buffer, the user configures the transmit descriptor (TDES0-TDES3) and sets the OWN bit (TDES0[31]) to 1
2. After the STT bit (ETH_DMA_OPRMODR register[13]) is set to 1, the DMA enters the running state immediately
3. In the running state, the DMA polls the send descriptor list for frames that need to be transferred. After polling is initiated, the DMA continues in sequential descriptor ring order or link order. If the DMA detects a descriptor marked as CPU-owned (TDES0.OWN=0), or an error condition occurs, the transfer is suspended and the transmit buffer unavailable bit (ETH_DMA_DMASTR register TUS bit) is combined with the normal interrupt summary enable bit (ETH_DMA_DMASTR register AIS bit) is set to 1. The sending engine continues with step 9
4. If the obtained descriptor is marked as owned by the DMA (TDES0.OWN=1), the DMA will decode the transmit data buffer address according to the obtained descriptor
5. DMA fetches transmit data from system memory and transfers that data
6. If the Ethernet frame is held in the data buffer of multiple descriptors, the DMA will close the intermediate descriptor and fetch the next descriptor. Repeat steps 3, 4 and 5 until the transmission of the Ethernet frame data is completed

7. After the frame transfer is complete, if IEEE1588 timestamping has been enabled for the frame (as indicated in the transmit status), the timestamp value will be written to the transmit descriptor (TDES6 and TDES7), and the status information will be written to the transmit descriptor's Each status bit (TDES0). The OWN bit is cleared during this step and the descriptor becomes owned by the CPU.
8. After the frame is transmitted, the TIS bit of the DMA Action Status Register will be set to 1 when its last descriptor bit (TDES0.TLS) is set to 1. Then the DMA engine returns to step 3
9. In the pending state, the DMA will attempt to re-acquire the descriptor when it receives a transmit poll request (return to step 3) and clear the underflow interrupt status bit

In non-OSF mode, the action flow chart of TxDMA is as followsFigure 39-18 shown.

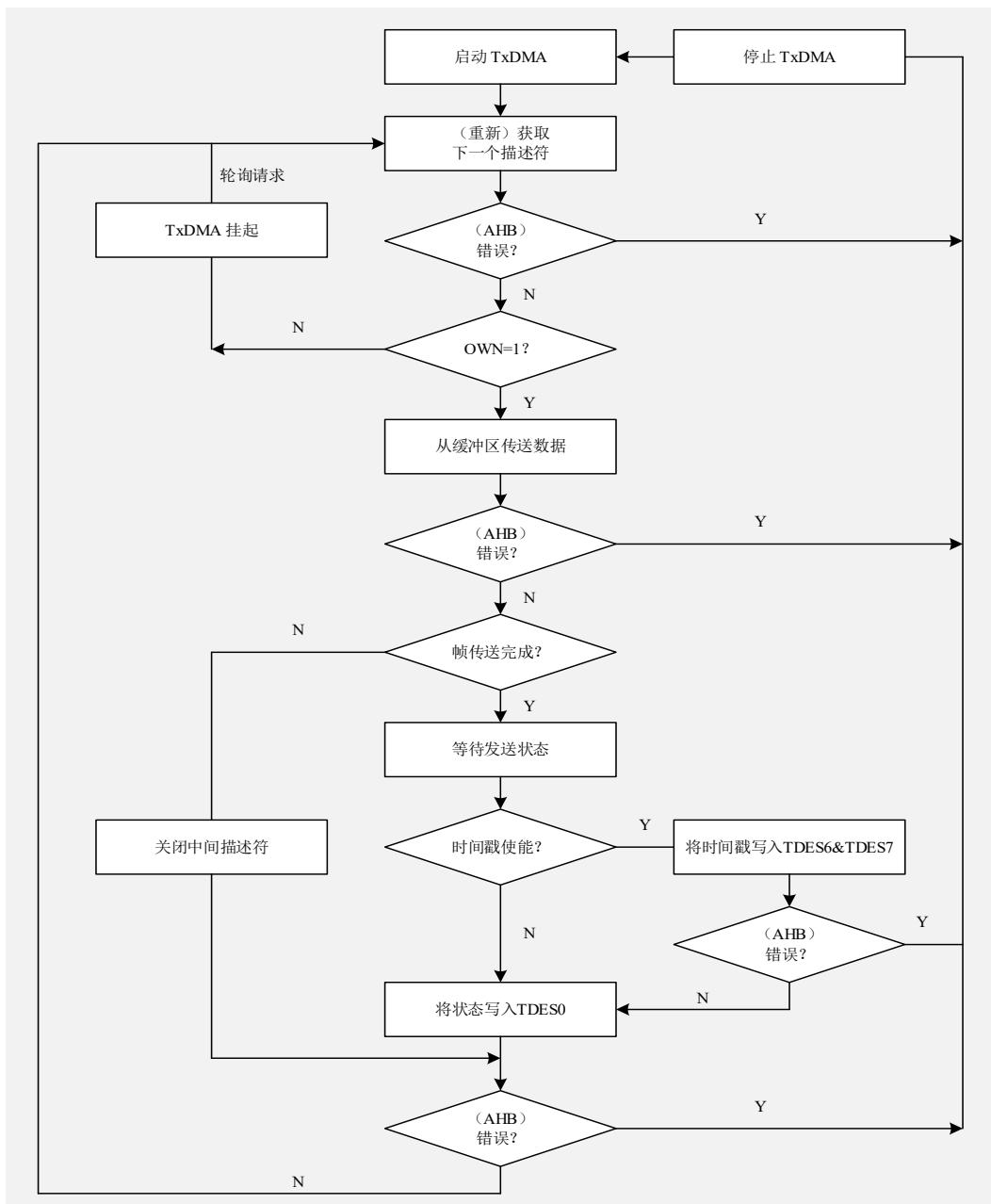


Figure 39-18 TxDMA action flow (non-OSF mode)

The operation sequence of the TxDMA engine under `ETH_DMA_OPRMODR.OSF=1` is as follows:

1. The operation process of DMA is as described in steps 1~6 of TxDMA in non-OSF mode
2. DMA fetches next descriptor without closing last descriptor of previous frame
3. If the DMA has the desired descriptor, it decodes the send buffer address in that descriptor. If the DMA does not own the descriptor, it enters suspend mode and skips to step 7
4. DMA gets the transmit frame from the system memory and transmits the frame until the frame data transmission ends. If the frame is split into multiple descriptors, the intermediate descriptor will be closed at the same time
5. The DMA waits for the transmit status and timestamp of the previous frame. When status

is available, if timestamps are captured (indicated by the status bits), the DMA writes these timestamps to TDES6 and TDES7. After that, the OWN bit is cleared, and the DMA writes the status to the corresponding TDES0, thereby closing the descriptor. DMA does not change the contents of TDES6 and TDES7 if timestamps are not enabled for the previous frame

6. If time stamping is enabled, the transmit interrupt is set, the DMA fetches the next descriptor, and proceeds to step 3 (when the status is OK). If the last transmit state shows an underflow error, the DMA goes into suspend mode (step 7)
7. In suspend mode, if the DMA receives a suspend status and a timestamp, it writes the timestamp (if timestamp is enabled for the current frame) to TDES6 and TDES7, and then writes the status to the corresponding TDES0. After that, set the relevant interrupt and return to suspend mode
8. Only after receiving a transmit poll request (ETH_DMA_TXPOLLR register), the DMA will exit suspend mode and enter running state (go to step 1 or step 2, depending on suspend state)

In OSF mode, the action flow chart of TxDMA is as followsFigure 39-19 shown.

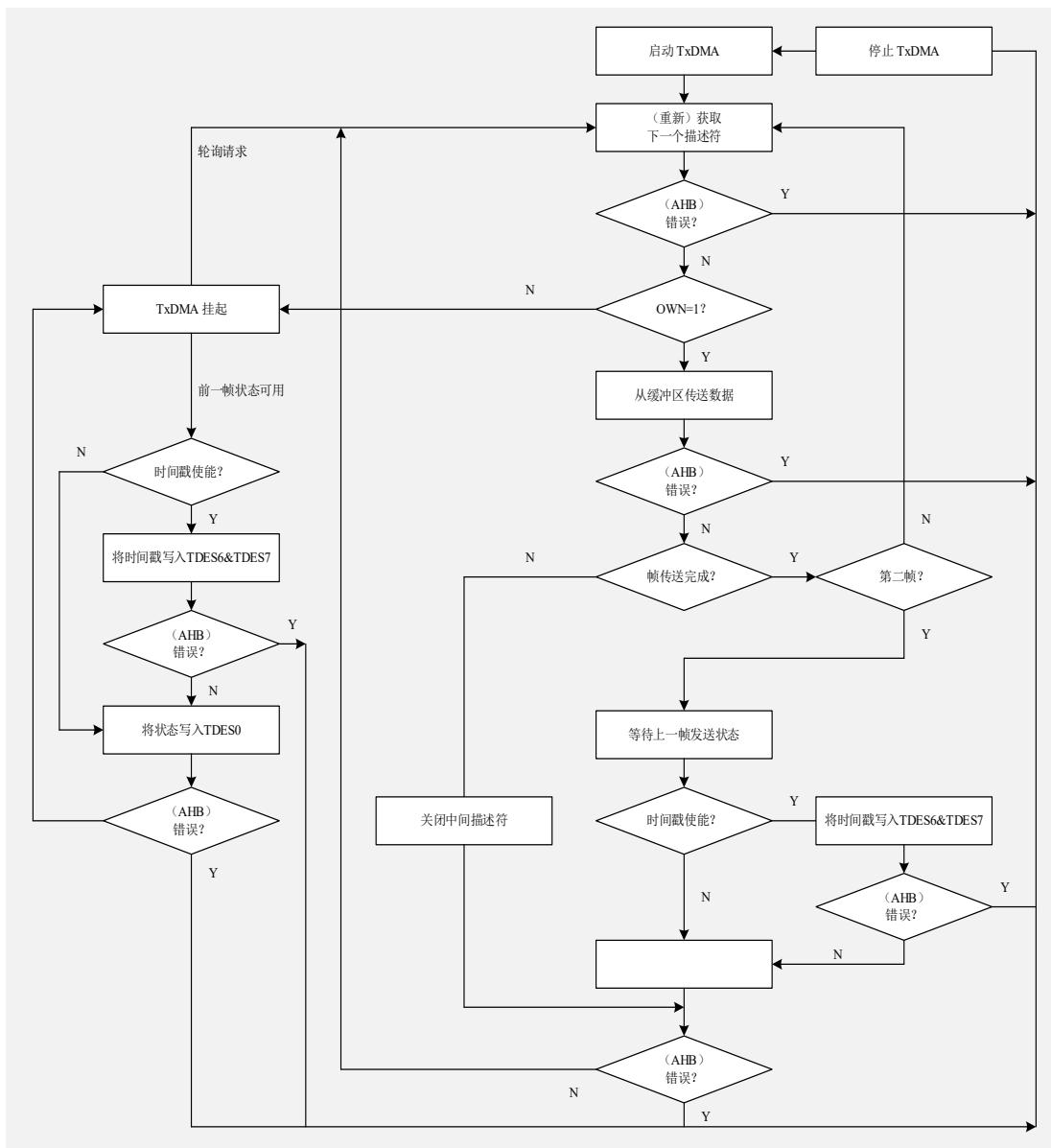


Figure 39-19 TxDMA operation flow (OSF mode)

Send frame processing

The system data buffer shall contain the complete Ethernet frame, excluding the header, PAD and FCS fields, and the DA field, SA field and type/length fields shall be included in the valid data. If the transmit descriptor indicates that the MAC core must disable insertion of CRC or padding bytes, the buffer must have a complete Ethernet frame (excluding the header) including the CRC bytes. Frames can be data-linked or span multiple buffers. A frame must be delimited by the first descriptor bit (TDES0.TFS) and the last descriptor bit (TDES0.TLS). When the transfer starts, TDES0.TFS of the first descriptor must be set to 1, after which the frame data is transferred from the memory buffer to the Tx FIFO. Meanwhile, if the current frame's TDES0.TLS is 0, the sending process will try to get the next descriptor. If TDES0.TLS is 0, it indicates the intermediate buffer, and if TDES0.TLS is 1, it indicates the last buffer of the frame. When the last buffer of the frame

has been transferred, the DMA will write the final status information back to the relevant status bits of the descriptor (TDES0). At this point, if the TDES0.IOC bit is set, the TIS bit of the transmit status register will be set, and the next descriptor will be fetched, and the process will be repeated.

According to the setting of the TSF bit in the ETH_DMA_OPRMODR register, the actual frame transfer process starts when the Tx FIFO reaches the programmable transmit threshold (TTC bit in the ETH_DMA_OPRMODR register), or when the Tx FIFO contains a complete frame. The DMA will release the descriptor after completing the frame transfer (clear the TDES0.OWN bit).

Send poll pending

Polling can be paused by any of the following conditions:

- The DMA detects all the descriptors of the CPU (TDES0[31]=0), and the transmit buffer unavailable flag TUS of the ETH_DMA_DMASTR register is set to 1. To recover, the driver must hand over ownership of the descriptor to the DMA and then issue a polling request command
- When a transmission error due to underflow is detected, frame transmission is aborted. The corresponding transmit descriptor (TDES0) bit will be set.

If the DMA enters a pending state due to the first condition, the normal interrupt summary bit of the transmit status register and the transmit buffer unavailable bits (ETH_DMA_DMASTR register bits NIS and TUS bits) are both set; if the second condition occurs, an exception Both the interrupt summary bit and the transmit underflow bit (ETH_DMA_DMASTR register bits AIS and UNS) will be set, and the information will be written to the transmit descriptor, causing the hang. In both cases, the position in the send list is preserved. The reserved location is the descriptor location after the last descriptor for which the DMA was closed. After correcting the cause of the hang, the driver must explicitly issue the Send Poll Request command.

Regular Tx Descriptor

The regular Tx descriptor structure consists of four 32-bit words, as shown in table 30.8, defined as TDES0, TDES1, TDES2, and TDES3.

Table 39-8 Regular Tx Descriptor

TDES0	OWN	control bit [30:26]	TTSE	control bit [24:18]	TTSS	status bit [16:0]					
TDES1	control bit [31:29]	byte count buffer 2 [28:16]				Reserved	byte count buffer 1 [12:0]				
TDES2	buffer address 1										
TDES3	buffer address 2										

The specific bits of TDES0~TDES3 are described below.

1) The functions of TDES0 are as follows:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
OWN	IOC	TLS	TFS	DCRC	DPAD	TTSE	CRCR	CIC[1:0]	TER	TSAC	VLANC[1:0]		TTSS	IHE	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ETSUM	JTE	FFF	TPCE	LOCE	NCE	TLCE	ECE	VLF	COC[3:0]			EDE	UDE	DEE	

Bit	Marking	Place name	Function
b31	OWN	all relation bits	0: The descriptor is owned by the CPU 1: The descriptor is owned by the DMA The DMA clears this bit when the frame transmission is completed or after the descriptor allocated buffer has been fully read After all subsequent descriptors belonging to the same frame are set, the OWN bit of the first descriptor of the frame should also be set to 1
b30	IOC	interrupt on completion	When this bit is set to 1, after the current frame is sent, the TIS bit of the action status register is set to 1 <i>Note: This bit is only valid if the last descriptor bit (TDES0.TLS) is set</i>
b29	TLS	last descriptor	This bit indicates that the buffer pointed to by this descriptor is the last buffer of the frame
b28	TFS	first descriptor	This bit indicates that this descriptor contains the first buffer of the frame. If the size of the first buffer is 0, the second buffer will contain the frame header of the frame; if the size of the second buffer is 0, the next descriptor will contain the frame header of the frame
b27	DCRC	invalid CRC	When this bit is set, the MAC will not append a Cyclic Redundancy Check (CRC) to the end of transmitted frames <i>Note: This bit is only valid if the first descriptor bit (TDES0.TFS) is set</i>
b26	DPAD	Invalid PAD	0: DMA will automatically add complement items and CRC for frames less than 64 bytes, regardless of the setting of the DCRC bit of the descriptor 1: MAC will not automatically add padding items for frames less than 64 bytes <i>Note: This bit is only valid if the first descriptor bit (TDES0.TFS) is set</i>
b25	TTSE	Timestamp enabled	When this bit is set to 1, the IEEE1588 hardware timestamp function will be activated for the transmit frame described by the descriptor <i>Note: This bit is only valid if the first descriptor bit (TDES0.TFS) is set</i>
b24	CRCR	CRC replacement control	When this bit is set to 1, the MAC will replace the FCS field of the outgoing frame with the calculated CRC value. <i>Note: This bit is valid only when the TDES0.DCRC bit of this descriptor is set</i>

			These bits control the calculation and insertion of the checksum as follows: 00: Prohibit insertion of checksum 01: Only enable calculation and insertion of IP header checksum 10: Enable calculation and insertion of IP header checksums and TCP/UCP/ICMP checksums, but will not calculate pseudo-header checksums in hardware 11: Enable the calculation and insertion of IP header checksum and TCP/UDP/ICMP checksum, and calculate pseudo-header checksum in hardware <i>Note: This bit is only valid if the last descriptor bit (TDES0.TLS) is set</i>
b23~b22	CIC	Checksum Insert Control	This bit indicates that the descriptor list has reached its last descriptor, and the DMA will return the first address of the descriptor list, forming a descriptor ring
b20	TSAC	second address link	When this bit is set to 1, the second address in the descriptor is the next descriptor address, not the second buffer address, and the TBS2 bit of the descriptor is invalid. <i>Note: The TER bit has higher priority than the TSAC bit</i>
b19~b18	VLANC	VLAN insertion control	00: Do not process VLAN frames 01: Remove the tag and type fields of VLAN frames before sending 10: Insert the VLAN tag value in the VLAN tag transmission control register (ETH_MAC_VTACTLR) into the transmission frame 11: Replace the VLAN identifier in the original transmitted frame with the VLAN tag value in the VLAN tag transmission control register (ETH_MAC_VTACTLR)
b17	TTSS	send timestamp status	This bit indicates that the timestamp has been captured for the current transmit frame. When set to 1, TDES6 and TDES7 will save the timestamp value captured by the transmit frame. <i>Note: This bit is only valid if the last descriptor bit (TDES0.TLS) is set</i>
b16	IHE	IP Header error	When this bit is set to 1, it indicates that the MAC sender detected an error in the IP datagram header <i>Note: When the COE engine detects an IP Header error, it still inserts the calculated Checksum value into the IPv4 Header Checksum field</i>
b15	ETSUM	Send error summary	B[16], B[14], B[13], B[12], B[11], B[10], B[9], B[8], B[2], B of the descriptor One of the [1] bits is set, the bit is also set
b14	JTE	Jabber timeout error	When this bit is set to 1, indicates that the MAC sender has experienced a Jabber timeout <i>Note: This bit will only be set if the MJD bit in the MAC configuration register (ETH_MAC_CONFIGR) is not set</i>
b13	FFF	frame refresh	This bit indicates that the DMA has refreshed the frame in accordance with the software refresh command issued by the CPU
b12	TPCE	payload error	This bit indicates that the COE engine has detected an error in the TCP, UDP or ICMP payload and will not update the Checksum field in the original frame
b11	LOCE	carrier loss error	This bit indicates that the carrier is lost during frame transmission, i.e. the MII CRS signal is invalid for one or more transmit clock cycles during frame transmission <i>Note: This bit is only valid for frames that are sent collision-free when the MAC is in half-duplex mode</i>
b10	NCE	no carrier error	This bit indicates that the carrier sense signal was not triggered by the PHY during transmission
b9	TLCE	Delay conflict error	This bit indicates that the frame transmission process was aborted due to a collision after the collision window (64 byte times in MII mode, including headers). <i>Note: This bit has no effect if the UDE error bit for this descriptor is set</i>
b8	ECE	Excessive conflict error	This bit indicates that an attempt to transmit the current frame was aborted due to 16 consecutive collisions <i>Note: If the DRTY (Disable Retry) bit in the MAC Configuration Register (ETH_MAC_CONFIGR) is set, this bit is set after the first collision and the frame transmission process is aborted</i>
b7	VLF	VLAN frame	This bit indicates that the transmitted frame is a VLAN frame
b6~b3	COC	conflict count	This bit indicates the number of collisions that occurred before the frame was sent <i>Note: This count is invalid when the Excessive Collision bit (TDES0.ECE) is set</i>
b2	EDE	excessive delay error	This bit indicates that the transmission was aborted due to excessive delay exceeding 24288 bit times <i>Note: This bit is valid when the Delay Check (DC) bit in the MAC Configuration Register (ETH_MAC_CONFIGR) is set</i>
b1	UDE	underflow error	This bit indicates that the MAC aborted the frame transmission because the data in the transmit buffer did not arrive in time. The underflow error indicates that the DMA encountered an empty transmit buffer during frame transmission.

b0	DEE	delay error	This bit indicates that the MAC is delayed due to the presence of a carrier before sending <i>Note: This bit is only valid in half duplex mode</i>
----	-----	-------------	---

2) The functions of TDES1 are as follows:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
SAIRC[2:0]		TBS2													
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved		TBS1													

Bit	Marking	Place name	Function
b31~b29	SAIRC	SA Insert Replacement Control	The SA field of the transmitted frame is processed as follows: 001: Insert the address value in the MAC address register 0 as the SA address into the transmit frame 101: Insert the address value in the MAC address register 1 as the SA address into the transmit frame 010: Replace the SA field in the transmit frame with the address value in the MAC address register 0 as the SA address 110: Replace the SA field in the transmission frame with the address value in the MAC address register 1 as the SA address Other values: no action
b28~b16	TBS2	Buffer 2 size	This bit indicates the size of the second data buffer in bytes <i>Note: This bit has no effect when the TDES0.TSAC bit is set</i>
b15~b13	Reserved	-	-
b12~b0	TBS1	buffer 1 size	This bit indicates the size of the first data buffer in bytes. If this field is 0, the DMA will ignore the buffer and use buffer 2 or the next descriptor, depending on the TSAC setting for that descriptor

3) The functions of TDES2 are as follows:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TBAP1 [31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TBAP1[15:0]															
Bit	Marking	Place name	Function												
b31~b0	TBAP1	send buffer 1 address	This bit indicates to the DMA where the data is in memory, when software provides the DMA with this descriptor (OWN bit in TDES0 is set), these bits will indicate the physical address of buffer 1												

4) The functions of TDES3 are as follows:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TBAP2 [31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TBAP2[15:0]															
Bit	Marking	Place name	Function												
b31~b0	TBAP2	send buffer 2 address/next descriptor address	These bits indicate to the DMA the location of the data in memory, when software provides this descriptor to the DMA (OWN bit in TDES0 is set to 1) and the descriptor ring structure is used, these bits will indicate the physical address of buffer 2; if TDES0. If the												

TSAC bit is set to 1, the address contains a pointer to the physical register where the next descriptor is located

Note: The buffer address pointer must match the bus width only if the TDES0.TSAC bit is set

Enhanced Tx Descriptor

The enhanced Tx descriptor shall be used when the time stamp function is enabled (ETH_PTP_TSPCTRLR.TSPEN=1) or when the Checksum Offload function is enabled (ETH_MAC_CONFIGR.IPCO=1).

The enhanced Tx descriptor structure consists of eight 32-bit words, as shown in table 30.9, defined as TDES0, TDES1, TDES2, TDES3, TDES4, TDES5, TDES6, and TDES7, respectively. Among them, the functions of TDES0~TDES3 are the same as those of the conventional type.

Table 39-9 Enhanced Tx Descriptor

TDES0	OWN	control bit [30:26]	TTSE	control bit [24:18]	TTSS	status bit [16:0]						
TDES1	control bit [31:29]	byte count buffer 2 [28:16]			Reserved	byte count buffer 1 [12:0]						
TDES2	buffer address 1											
TDES3	buffer address 2											
TDES4	Reserved											
TDES5	Reserved											
TDES6	Timestamp low time											
TDES7	Timestamp high time											

The specific bits of TDES0~TDES7 are described below.

- 1) The functions of TDES0: The same function as the regular type descriptor TDES0.
- 2) Features of TDES1: The same function as the regular type descriptor TDES1.
- 3) Features of TDES2: The same function as the regular type descriptor TDES2.
- 4) Features of TDES3: The same function as the regular type descriptor TDES3.
- 5) TDES4, TDES5: Reserved
- 6) The functions of TDES6 are as follows:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TTSL [31:16]															

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

Bit	Marking	Place name	Function
b31~b0	TTSL	timestamp status	DMA updates the lower 32 bits of the timestamp captured by the corresponding transmit frame into this field <i>Note: This field only contains a timestamp when the last descriptor bit in the descriptor (TDES0.TLS) is set and the transmit timestamp status bit (TDES0.TTSS) is set</i>

7) The functions of TDES7 are as follows:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TTSH [31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TTSH[15:0]															
Bit	Marking	Place name										Function			
b31~b0	TTSH	Timestamp high										DMA updates the upper 32 bits of the timestamp captured by the corresponding transmit frame into this field <i>Note: This field only contains a timestamp when the last descriptor bit in the descriptor (TDES0.TLS) is set and the transmit timestamp status bit (TDES0.TTSS) is set</i>			

39.4.3.4 DMA receive configuration

RxDMA operation

The sequence of operations of the RxDMA engine is as follows:

1. The user sets the receive descriptor (RDES0-RDES3) and sets the OWN bit (RDES0[31]) to 1
2. After the operating mode register ETH_DMA_OPRMODR.STR bit is set, the DMA enters the running state. In the running state, the DMA polls the list of receive descriptors, trying to get free descriptors. If the obtained descriptor is not free (owned by the CPU), the DMA goes into pending state and jumps to step 9
3. DMA decodes the receive data buffer address in the acquired descriptor
4. Process the incoming frame and put it into the data buffer of the fetched descriptor
5. When the buffer is full or the frame transfer is complete, the receive engine will get the next descriptor
6. If the current frame transfer is complete, the DMA will continue to step 7. If the DMA does not own the next receive descriptor and the frame transfer has not completed (the EOF field has not been transferred), the DMA will set the descriptor error bit DPE in RDES0 (unless flushing is disabled). DMA closes the current descriptor (clears the OWN bit) and marks it as an intermediate descriptor (or last descriptor if flushing is disabled) by clearing the last descriptor bit (RDES0.RLS), then proceed to step 8; if the DMA already owns the next descriptor, but the current frame transfer has not completed, the DMA will close the current descriptor as an intermediate value and return to step 4
7. If the IEEE1588 timestamp feature is enabled, the DMA writes the timestamp (if available) to RDES6 and RDES7 of the current descriptor. The DMA then gets the status of the received frame and writes the status word to RDES0 of the current descriptor, with the OWN bit cleared and the last descriptor bit set
8. The receive engine checks the OWN bit of the latest descriptor. If the CPU owns the descriptor (OWN bit is 0), the receive buffer unavailable bit (ETH_DMA_DMASTR.RUS) is

set and the DMA receive engine enters the pending state (step 9); if the DMA owns the descriptor, the engine will Go back to step 4 and wait for the next frame

9. Some frames will be flushed from the Rx FIFO before the receive engine goes into suspend state (flushing can be controlled using the DFRF bit of the ETH_DMA_OPRMODR register)
10. The RxDMA will come out of the suspend state when a receive poll request command is received or the start of the next frame can be obtained from the Rx FIFO. The engine continues to step 2 and re-fetches the next descriptor

The action flow chart of RxDMA is as followsFigure 39-20 shown.

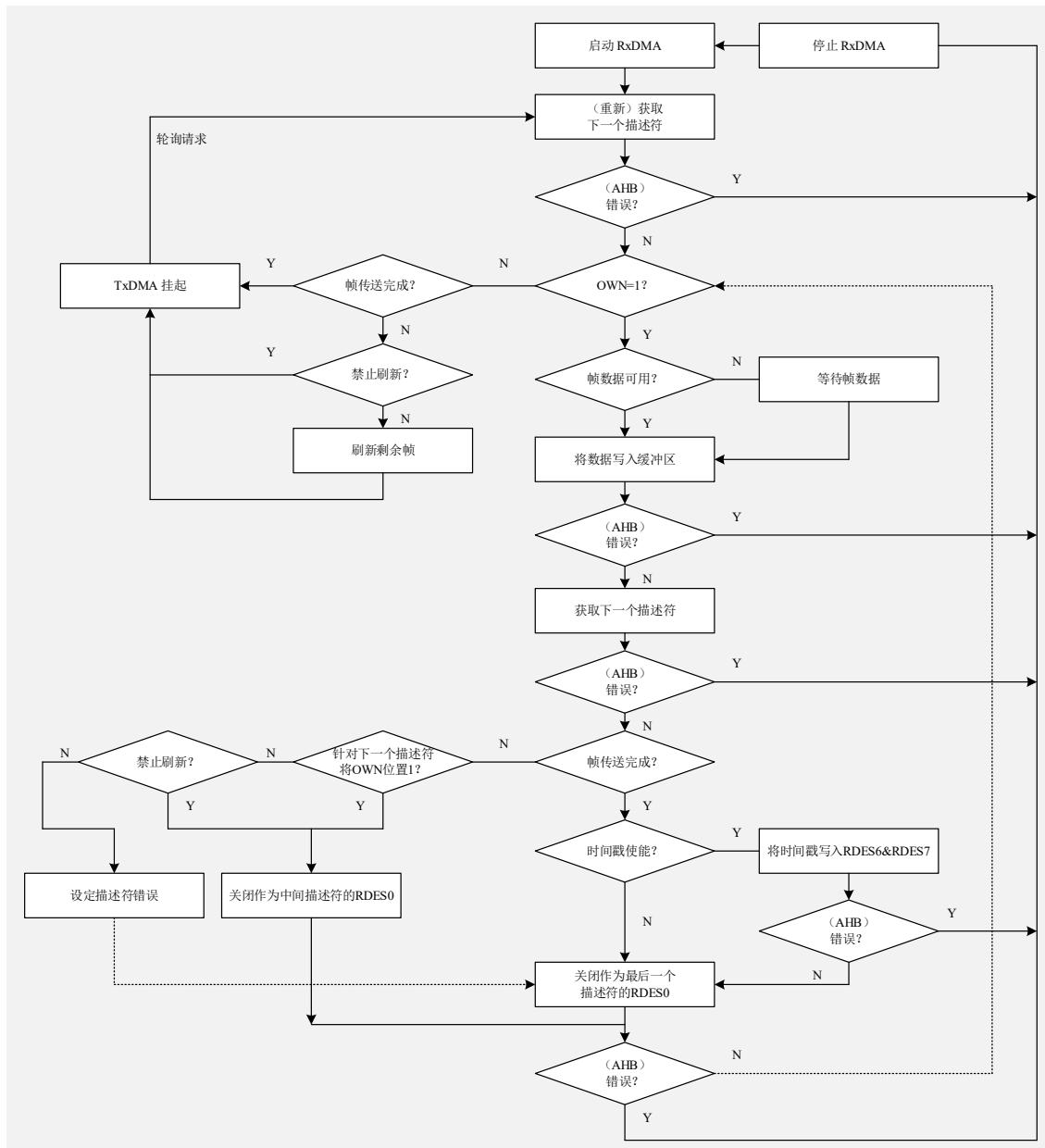


Figure 39-20 RxDMA action flow

RxDMA's receive status is not acknowledged until the timestamp writeback is complete and the state is ready to be written back to the descriptor. If timestamping is enabled, when a valid timestamp value for a frame cannot be obtained (for example, the Rx FIFO is full before the timestamp is written), the DMA writes 1 to all bits of RDES6 and RDES7; if not enabled Timestamp functionality, RDES6 and RDES7 remain unchanged.

Receive frame acquisition

The receive engine always tries to get an extra descriptor for the incoming frame. An attempt is made to obtain a descriptor as long as any of the following conditions are met:

- The receive start/stop bit (ETH_DMA_OPRMODR.STR) is set to 1 immediately after the DMA enters the running state
- The data buffer for the current descriptor is full before the end of the currently transmitted frame
- The MAC controller has finished receiving the data frame, but the current receive descriptor has not been closed
- The receive process hangs because the descriptor is owned by the CPU (RDES0.OWN=0) and a new frame is received
- Receive polling request command issued

Receive frame processing

The MAC will not transmit a received frame to the system until the frame passes address filtering and its size is greater than or equal to the configurable threshold number of bytes set for the Rx FIFO, or when the entire frame is written to the Rx FIFO in store-and-forward mode Memory; when a frame fails address filtering, the frame will be discarded (unless receive all bits ETH_MAC_FLTCTRL.RA set). Frames under 64 bytes can be flushed from the Rx FIFO due to collisions or premature termination. After receiving 64 (configurable threshold) bytes, the DMA begins transferring the frame data into the receive buffer specified by the current descriptor. After the DMA AHB interface is ready to receive a data transfer, the first descriptor bit (RDES0.RFS) is set to 1 to separate the frame if the DMA is not currently fetching transmit data from memory. When the data buffer is full or the end of the frame has been transferred to the receive buffer, the OWN (RDES0) bit is reset to 0 and the descriptor is released. If the frame is contained in a single descriptor, the last descriptor bit (RDES0.RLS) and the first descriptor bit (RDES0.RFS) are both set.

When the DMA gets the descriptor of the next frame, it sets the last descriptor bit (FDES0.RLS) to 1, releases the status bit in the previous frame descriptor, and then sets the receive interrupt bit (ETH_DMA_DMASTR register RIS) to 1. This process will repeat until the DMA encounters a descriptor marked as owned by the CPU. In this case, the receive process sets the receive buffer unavailable bit RUS in the ETH_DMA_DMASTR register to 1, and then enters the suspend state, but its position in the receive list is still reserved.

Receive process hangs

If a new receive frame arrives while the receive process is pending, the DMA will refetch the current descriptor in system memory. If the descriptor is now owned by the DMA, the receive process will re-enter the running state and start receiving frames; if the descriptor is still owned by the host, by default the DMA will discard the current frame at the top of the Rx FIFO and will lose the frame. The counter is incremented. This process is repeated if more than one frame is stored in the Rx FIFO.

If the DFRF bit of the DMA operating mode register (ETH_DMA_OPRMODR) is set to 1, it can avoid discarding or flushing the frame at the top of the Rx FIFO. In this case, the receive process sets the

receive buffer unavailable bit RUS of the Action Status Register (ETH_DMA_DMASTR) to 1 and returns to the pending state.

Regular Rx Descriptor

The regular Rx descriptor structure consists of four 32-bit words, such asTable 39-10 As shown, they are defined as RDES0, RDES1, RDES2, and RDES3, respectively.

Table 39-10 Regular Rx Descriptor

RDES0	OWN	status bit [30:0]							
	control bit [31]	Reserved	byte count buffer 2 [28:16]	control bit [15:14]	Reserved	byte count buffer 1 [12:0]			
RDES2	buffer address 1								
RDES3	buffer address 2								

The specific bits of RDES0~RDES3 are described below.

1) The functions of RDES0 are as follows:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
OWN	DAF	FRAL[13:0]													

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ERSUM	DPE	SAF	LEE	OVE	VLAT	RFS	RLS	IPE/TSPA/GF	RLCE	FRAT	WTE	REE	DBE	CRE	DAS/ESA

Bit	Marking	Place name	Function
b31	OWN	all relation bits	0: The descriptor is owned by the CPU 1: The descriptor is owned by the DMA DMA clears this bit when frame reception is complete or when the associated buffer for this descriptor is full
b30	DAF	Destination address filtering failed	This bit indicates that the frame failed to pass DA filtering in the MAC core
b29~b16	FRAL	Frame length statistics	This bit indicates the byte length of the received frame (including CRC) transferred to the host memory This field is valid only when the last descriptor bit (RDES0.RLS) is set and the descriptor error bit (RDES0.DPE) and the overflow error bit (RDES0.OVE) are reset When neither the last descriptor bit nor the error summary bit are set, this field indicates the cumulative number of bytes transmitted for the current frame
b15	ERSUM	Receive Error Summary	B[14], B[11], B[7], B[6], B[4], B[3], B[1] bits of this descriptor and B[4], B of REDS4 descriptor One of the [3] bits is set, and this bit is also set
b14	DPE	Descriptor error	This bit indicates that a frame was truncated by exceeding the size of the current descriptor buffer and that the DMA does not own the next descriptor <i>Note: This bit is only valid when the last descriptor (RDES0.RLS) is set to 1</i>
b13	SAF	Source address filtering failed	This bit indicates that the SA field of the frame failed the SA filtering in the MAC core
b12	LEE	wrong length	This bit indicates that the actual length of the received frame does not match the value of the length/type field <i>Note 1: This bit is only valid after the frame type bit (RDES0.FRAT) is reset</i> <i>Note 2: This bit is invalid when the CRC error bit (RDES0.CRE) is valid</i>
b11	OVE	overflow error	This bit indicates that the received frame was corrupted due to buffer overflow
b10	VLAT	VLAN identifier	This bit indicates that the frame pointed to by the descriptor is a VLAN frame tagged by the MAC core
b9	RFS	first descriptor	This bit indicates that this descriptor contains the first buffer of the frame. If the size of the first buffer is 0, the second buffer will contain the frame header of the frame; if the size of the second

			buffer is 0, the next descriptor will contain the frame header of the frame
b8	RLS	last descriptor	This bit indicates that the buffer pointed to by this descriptor is the last buffer of the frame
b7	IPE/TSPA/GF	COE errors/timestamps/jumbo frames	<p>When the Checksum Offload function is active: This bit indicates that there is an error in the IPv4 or IPv6 header The reason for this error can be:</p> <ol style="list-style-type: none"> 1) The Ethernet type field is inconsistent with the IP header version field value 2) Checksums of headers in IPv4 do not match 3) The Ethernet frame is missing the required number of IP header bytes <p>When the timestamp function is in effect: This bit indicates that a timestamp snapshot was recorded in RDES6 and RDES7</p> <p><i>Note: This bit is only valid when the last descriptor (RDES0.RLS) is set to 1</i></p> <p>When both of the above are invalid: This bit indicates that the MAC received a jumbo frame</p>
b6	RLCE	Delay conflict error	This bit indicates that a late collision occurred when a frame was received in half-duplex mode
b5	FRAT	frame type	<p>0: MAC received Ethernet frame 1: MAC received PTP frame</p> <p><i>Note: This bit has no effect when a dwarf frame is received</i></p>
b4	WTE	watchdog error	This bit indicates that the receive watchdog timer timed out when the current frame was received and the current frame was truncated after the watchdog timed out
b3	REE	receive error	This bit indicates that during frame reception, when the PHY sends the RX_DV signal, because the RX_ER signal is sent
b2	DBE	Dribble bit error	This bit indicates that the received frame has a non-integer multiple of bytes (odd nibble) <p><i>Note: This bit is only valid in MII mode</i></p>
b1	CRE	CRC error	<p>This bit indicates that the received frame has a Cyclic Redundancy Check (CRC) error</p> <p><i>Note: This bit is only valid when the last descriptor (RDES0.RLS) is set to 1</i></p>
b0	DAS/ESA	Address filtering success/status bit expansion	<p>When COE function and PTP function are invalid: 0: The received frame passed the DA address filter of the MAC address register 0 1: The received frame has passed the DA address filtering of the MAC address registers 1~5</p> <p><i>Note: This bit has no effect when the DAF bit of this descriptor is set</i></p> <p>When COE function or PTP function is valid: This bit indicates that the RDES4 descriptor is valid <i>Note: This bit is only valid when the last descriptor (RDES0.RLS) is set to 1</i></p>

2) The functions of RDES1 are as follows:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DIC	Reserved	RBS2[12:0]													

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RER	RSAC	-	RBS1[12:0]												

Bit	Marking	Place name	Function
b31	DIC	Disable interrupts on completion	<p>When this bit is set to 1, it disables raising the host's interrupt when the received frame ends in the buffer indicated by this descriptor, and also prevents the RIS bit of the Action Status Register (ETH_DMA_DMASTR) from being set.</p> <p><i>Note: This bit is only valid if the last descriptor bit (RDES0.RLS) is set</i></p>
b30~29	Reserved	-	-
b28~b16	RBS2	Buffer 2 size	<p>This bit indicates the size of the second data buffer in bytes</p> <p><i>Note: This bit has no effect when the RSAC bit of this descriptor is set</i></p>
b15	RER	End of ring reception	This bit indicates that the descriptor list has reached its last descriptor, and the DMA will return the first address of the descriptor list, forming a descriptor ring
b14	RSAC	second address link	When this bit is 1, the second address in the descriptor is the next descriptor address, not the second buffer address, and the RBS2 bit of the descriptor is invalid.

Note: The RER bit takes precedence over the RSAC bit

b13	Reserved	-	-	
b12~b0	RBS1	buffer 1 size	This bit indicates the size of the first data buffer in bytes. If this field is 0, the DMA will ignore the buffer and use buffer 2 or the next descriptor, depending on the RSAC setting for that descriptor	

3) The functions of RDES2 are as follows:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
RBAP1 [31:16]															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RBAP1[15:0]															

Bit	Marking	Place name	Function
b31~b0	RBAP1	Receive buffer 1 address	This bit indicates to the DMA where the data is in memory, when software provides the DMA with this descriptor (OWN bit in RDES0 is set), these bits will indicate the physical address of buffer 1

4) The functions of RDES3 are as follows:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
RBAP2 [31:16]															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RBAP2[15:0]															

Bit	Marking	Place name	Function
b31~b0	RBAP2	receive buffer 2 address/next descriptor address	These bits indicate to the DMA where the data is located in memory, when software provides this descriptor to the DMA (OWN bit 1 in RDES0) and the descriptor ring structure is used, these bits will indicate the physical address of buffer 2; if RDES1. If the RSAC bit is set to 1, the address contains the pointer to the physical register where the next descriptor is located <i>Note: The buffer address pointer must match the bus width only if the RDES1.RSAC bit is set</i>

Enhanced Rx Descriptor

The enhanced Rx descriptor should be used when the timestamp function is enabled (ETH_PTP_TSPCTRLR.TSPEN=1) or when the Checksum Offload function is enabled (ETH_MAC_CONFIGR.IPCO=1).

The enhanced Tx descriptor structure consists of 8 32-bit words such as Table 39-11. As shown, they are defined as RDES0, RDES1, RDES2, RDES3, RDES4, RDES5, RDES6, and RDES7, respectively. Among them, the functions of RDES0~RDES3 are the same as those of the conventional type.

Table 39-11 Enhanced Descriptor

RDES0	OWN	status bit [30:0]						
RDES1	control bit [31]	Reserved	byte count buffer 2 [28:16]		control bit [15:14]	Reserved		
RDES2	buffer address 1							
RDES3	buffer address 2							
RDES4	Extended status bits [31:0]							
RDES5	Reserved							
RDES6	Timestamp low time							
RDES7	Timestamp high time							

The specific bits of RDES0~RDES7 are described below.

- 1) The functions of RDES0: The same function as the regular type descriptor RDES0.
- 2) The functions of RDES1: The same function as the regular type descriptor RDES1.
- 3) Features of RDES2: The same function as the regular type descriptor RDES2.
- 4) Features of RDES3: The same function as the regular type descriptor RDES3.
- 5) The functions of RDES4 are as follows:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved				L4FMS	L3FMS		Reserved								

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	TSPD	PTPV	PTPFT	MTP[3:0]			IPv6 DR	IPv4 DR	IPCB	IPPE	IPHE	IPPT[2:0]			

Bit	Marking	Place name	Function
b31~b16	Reserved	-	-
b25	L4FMS	L4 filtering succeeded	This bit indicates that the received frame passed L4 port filtering
b24	L3FMS	L3 filtering succeeded	This bit indicates that the received frame passed L3 address filtering
b23~b15	Reserved	-	-
b14	TSPD	Timestamp discard	This bit indicates that timestamp snapshots were taken but timestamps were discarded due to Rx FIFO overflow
b13	PTPV	PTP version	This bit indicates the version type of the received PTP frame 0: PTP frame of IEEE1588v1 version 1: PTP frame of IEEE1588v2 version
b12	PTPFT	PTP frame type	This bit indicates the type of PTP frame received 0: PTP frame of Over Ethernet type 1: PTP frame of Over UDP-IPv4 or Over UDP-IPv6 type <i>Note: Whether it is Over UDP-IPv4 or Over UDP-IPv6 can be obtained through the IPv6FR and IPv4FR bits of the descriptor</i>

				This bit indicates the type of message received 0000: No PTP message received 0001: SYNC (all clock types) 0010: Follow_Up (all clock types) 0011: Delay_Req (all clock types) 0100: Delay_Resp (all clock types)
b11~b8	MTP	message type		0101: Pdelay_Req (in a peer-to-peer transparent clock) 0110: Pdelay_Resp (in a peer-to-peer transparent clock) 0111: Pdelay_Resp_Follow_Up (in peer-to-peer transparent clock) 1000: Announce 1001: Management 1010: Signaling 1111: Default message type other: Keep
b7	IPv6DR	Received IPv6 message		This bit indicates that an IPv6 packet was received
b6	IPv4DR	Received IPv4 message		This bit indicates that an IPv4 packet was received
b5	IPCB	Checksum Offload passthrough		This bit indicates bypassing the COE engine
b4	IPPE	IP payload error		This bit is set to 1 when: 1) When the 16-bit IP payload checksum calculated by the kernel (i.e. TCP, UDP or ICMP checksum) does not match the corresponding checksum field in the received frame 2) When the TCP, UDP or ICMP segment length does not match the payload length value in the IP header field
b3	IPHE	IP header error		This bit indicates that the 16-bit IPv4 header checksum computed by the kernel does not match the received checksum byte, or that the IP header version field does not match the Ethertype value
b2~b0	IPPT	payload type		This bit indicates the type of payload encapsulated in the IP datagram, these bits are "000" when there is an error in the IP header or when there is fragmented IP 000: Unknown, or IP payload not processed 001: UDP 010: TCP 011: ICMP 1xx: Keep

6) RDES5: Reserved

7) The functions of RDES6 are as follows:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
RTSL[31:16]															

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

Bit	Marking	Place name	Function
b31~b0	RTSL	timestamp status	The DMA updates the lower 32 bits captured by the corresponding received frame into this field. <i>Note: This field contains a timestamp only if the last descriptor bit in the descriptor (RDES0.RLS) is set to 1</i>

8) The functions of RDES7 are as follows:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
RTSH [31:16]															

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

Bit	Marking	Place name	Function
b31~b0	RTSH	Timestamp high	The DMA updates the upper 32 bits captured by the corresponding received frame into this field. <i>Note: This field contains a timestamp only if the last descriptor bit in the descriptor (RDES0.RLS) is set to 1</i>

39.5 Interrupt Description

Figure 39-21 is the interrupt implementation scheme in the Ethernet MAC controller. As can be seen from the figure, ETHMAC uses DMA normal events (NIS), DMA abnormal events (AIS), PMT events (PMTIS), MMC events (MMC**IS), and PTP events (TSPIS) through interrupts. The enable or mask control is aggregated into one interrupt event and output to the interrupt module, so when the system responds to the interrupt, it needs to read the corresponding status information to determine which type of event has generated the interrupt.

At the same time, the PMT module of ETHMAC also has a non-maskable interrupt output. When a PMT event occurs, regardless of whether the PMTIM bit in the interrupt mask register (ETH_MAC_INTMSKR) is set to 1, a PMT non-maskable interrupt will be generated.

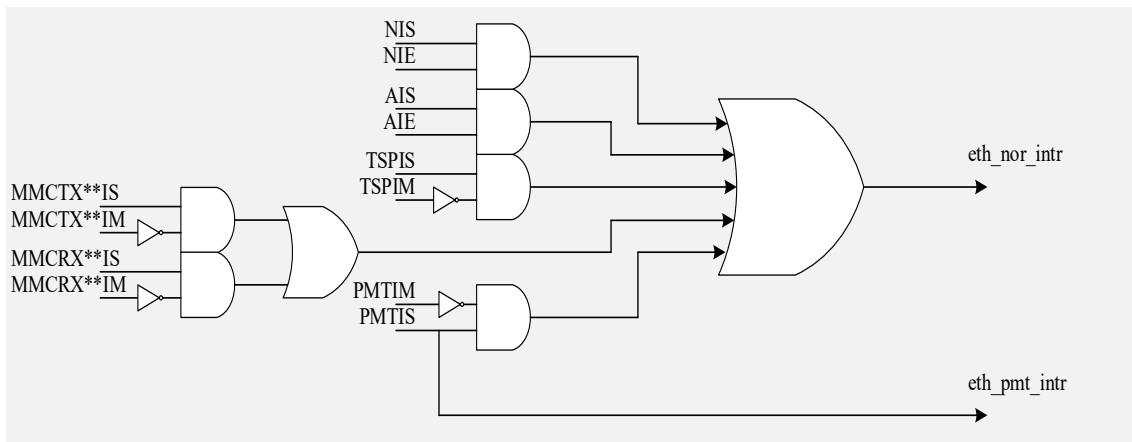


Figure 39-21 ETHMAC interrupt scheme

39.5.1 DMA interrupt

Various normal events generated by DMA in the process of sending data and receiving data will trigger ordinary interrupts, and various abnormal events will trigger abnormal interrupts. Such as Figure 39-22 shown.

The interrupt control of each specific event can be controlled by each bit of the DMA interrupt enable register (ETH_DMA_INTENAR). See the DMA Interrupt Control Register chapter.

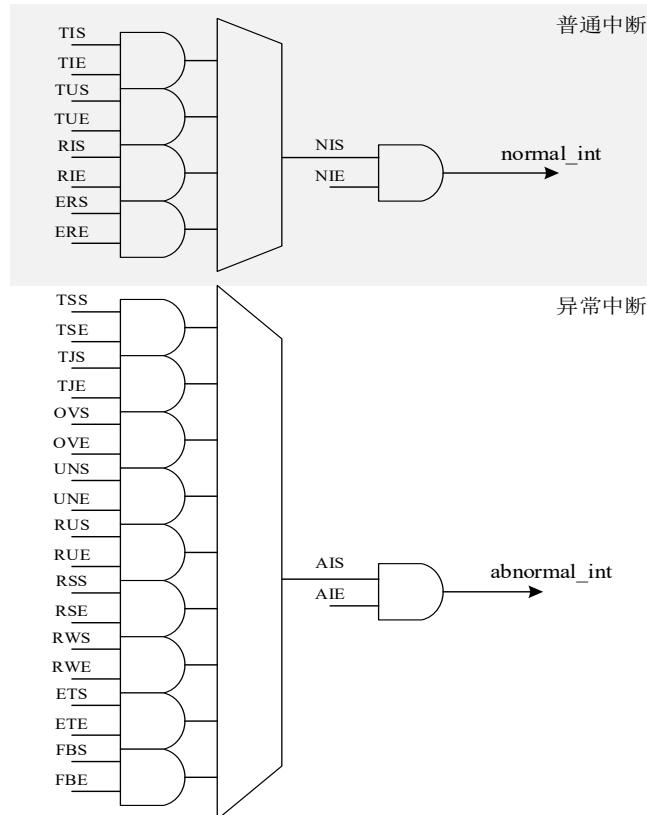


Figure 39-22 DMA interrupt composition

39.5.2 PMT interrupt

The MAC can trigger the generation of PMT interrupts (maskable and non-maskable interrupts) upon receipt of remote wakeup frames and magic packets. The generation of maskable interrupts is controlled by the PMTIM bit of the MAC interrupt mask register (ETH_MAC_INTMSKR).

39.5.3 PTP interrupt

When the PTP function is valid, the system time can trigger the generation of a PTP interrupt when the system time reaches the set value in the timestamp target time registers 0~1. This interrupt is controlled by the TT0SEL, TT1SEL bits of the PPS Output Control Register (ETH_MAC_PPSCTRLR) and the TSPIM bits of the MAC Interrupt Mask Register (ETH_MAC_INTMSKR).

39.5.4 MMC interrupt

The MMC interrupt can be triggered when the statistics of sending and receiving frames in the MMC overflow. This interrupt is controlled by various bits of the MMC's Rx Interrupt Control Register (ETH_MMC_RITCLR) and Tx Interrupt Control Register (ETH_MMC_TITCLR). See the MMC Rx Interrupt Control Register and Tx Interrupt Controller chapters.

39.6 Register description

Table 39-12 shown is the register list of the ETHMAC module.

BASE ADDR: 0x40060000H

Table 39-12 ETHMAC register list

Register name	Symbol	Offset	Bit width	Reset value
ETH_MAC interface selection register	ETH_MAC_IFCONFR	(0x40055410h)	32	0x00000000h
ETH_MAC MAC Configuration Register	ETH_MAC_CONFIGR	0x0000h	32	0x00008000h
ETH_MAC MAC flow control register	ETH_MAC_FLOCTRLR	0x0018h	32	0x00000000h
ETH_MAC MAC Status Register	ETH_MAC_MACSTSR	0x0024h	32	0x00000000h
ETH_MAC Interrupt Status Register	ETH_MAC_INTSTSR	0x0038h	32	0x00000000h
ETH_MAC interrupt mask register	ETH_MAC_INTMSKR	0x003Ch	32	0x00000000h
ETH_MAC SMI address register	ETH_MAC_SMIADDR	0x0010h	32	0x00000000h
ETH_MAC SMI data register	ETH_MAC_SMIDATR	0x0014h	32	0x00000000h
ETH_MAC PMT Control Status Register	ETH_MAC_PMTCTRLR	0x002Ch	32	0x00000000h
ETH_MAC Remote Wakeup Frame Filter Register	ETH_MAC_RTWKFFR	0x0028h	32	0x00000000h
ETH_MAC MAC Frame Filtering Control Register	ETH_MAC_FLTCTRLR	0x0004h	32	0x00000000h
ETH_MAC MAC address high register 0	ETH_MAC_MACADHR0	0x0040h	32	0x8000FFFFh
ETH_MAC MAC address status register 0	ETH_MAC_MACADLR0	0x0044h	32	0xFFFFFFFFh
ETH_MAC MAC address high register 1	ETH_MAC_MACADHR1	0x0048h	32	0x0000FFFFh
ETH_MAC MAC address status register 1	ETH_MAC_MACADLR1	0x004Ch	32	0xFFFFFFFFh
ETH_MAC MAC address high register 2	ETH_MAC_MACADHR2	0x0050h	32	0x0000FFFFh
ETH_MAC MAC address status register 2	ETH_MAC_MACADLR2	0x0054h	32	0xFFFFFFFFh
ETH_MAC MAC address high register 3	ETH_MAC_MACADHR3	0x0058h	32	0x0000FFFFh
ETH_MAC MAC address status register 3	ETH_MAC_MACADLR3	0x005Ch	32	0xFFFFFFFFh
ETH_MAC MAC address high register 4	ETH_MAC_MACADHR4	0x0060h	32	0x0000FFFFh
ETH_MAC MAC address status register 4	ETH_MAC_MACADLR4	0x0064h	32	0xFFFFFFFFh
ETH_MAC MAC Hash table high register	ETH_MAC_HASHTHR	0x0008h	32	0x00000000h
ETH_MAC MAC Hash table low register	ETH_MAC_HASHTLR	0x000Ch	32	0x00000000h
ETH_MAC VLAN tag sending control register	ETH_MAC_VTACTLR	0x0584h	32	0x00000000h
ETH_MAC VLAN Tag Acceptance Filter Register	ETH_MAC_VTAFLTR	0x001Ch	32	0x00000000h
ETH_MAC VLAN Hash table register	ETH_MAC_VLAHTBR	0x0588h	32	0x00000000h
ETH_MAC LAY3LAY4 Control Register	ETH_MAC_L34CTRLR	0x0400h	32	0x00000000h
ETH_MAC LAY4 port register	ETH_MAC_L4PORTR	0x0404h	32	0x00000000h
ETH_MAC LAY3 address register 0	ETH_MAC_L3ADDR0	0x0410h	32	0x00000000h
ETH_MAC LAY3 address register 1	ETH_MAC_L3ADDR1	0x0414h	32	0x00000000h
ETH_MAC LAY3 address register 2	ETH_MAC_L3ADDR2	0x0418h	32	0x00000000h

Register name	Symbol	Offset	Bit width	Reset value
ETH_MAC LAY3 address register 3	ETH_MAC_L3ADDRR3	0x041Ch	32	0x00000000h
ETH_PTP Time Stamp Control Register	ETH_PTP_TSPCTLR	0x0700h	32	0x00002000h
ETH_PTP Timestamp Status Register	ETH_PTP_TSPSTSR	0x0728h	32	0x00000000h
ETH_PTP timestamp basic addend register	ETH_PTP_TSPADDR	0x0718h	32	0x00000000h
ETH_PTP timestamp subsecond adder register	ETH_PTP_TSPNSAR	0x0704h	32	0x00000000h
ETH_PTP Timestamp System Seconds Register	ETH_PTP_TMSSECR	0x0708h	32	0x00000000h
ETH_PTP Timestamp System Subsecond Register	ETH_PTP_TMSNSER	0x070Ch	32	0x00000000h
ETH_PTP timestamp update seconds register	ETH_PTP_TMUSECR	0x0710h	32	0x00000000h
ETH_PTP timestamp update sub-second register	ETH_PTP_TMUUNSER	0x0714h	32	0x00000000h
ETH_PTP Timestamp Target Seconds Register 0	ETH_PTP_TMTSECR0	0x071Ch	32	0x00000000h
ETH_PTP timestamp target subsecond register 0	ETH_PTP_TMTNSER0	0x0720h	32	0x00000000h
ETH_PTP Timestamp Target Seconds Register 1	ETH_PTP_TMTSECR1	0x0780h	32	0x00000000h
ETH_PTP Timestamp Target Subsecond Register 1	ETH_PTP_TMTNSER1	0x0784h	32	0x00000000h
ETH_PTP PPS output control register	ETH_PTP_PPSCTLR	0x072Ch	32	0x00000000h
ETH_DMA bus mode register	ETH_DMA_BUSMODR	0x1000h	32	0x00020101h
ETH_DMA Operation Mode Register	ETH_DMA_OPRMODR	0x1018h	32	0x00000000h
ETH_DMA Action Status Register	ETH_DMA_DMASTS	0x1014h	32	0x00000000h
ETH_DMA Interrupt Enable Register	ETH_DMA_INTENAR	0x101Ch	32	0x00000000h
ETH_DMA Frame Loss Statistics Register	ETH_DMA_RFRCNTR	0x1020h	32	0x00000000h
ETH_DMA watchdog timer register	ETH_DMA_REVWDTR	0x1024h	32	0x00000000h
ETH_DMA transmit polling request register	ETH_DMA_TXPOLLR	0x1004h	32	0x00000000h
ETH_DMA Receive Polling Request Register	ETH_DMA_RXPOLLR	0x1008h	32	0x00000000h
ETH_DMA transmit descriptor list address register	ETH_DMA_TXDLADR	0x1010h	32	0x00000000h
ETH_DMA Receive Descriptor List Address Register	ETH_DMA_RXDLADR	0x100Ch	32	0x00000000h
ETH_DMA current host transmit descriptor register	ETH_DMA_CHTXDER	0x1048h	32	0x00000000h
ETH_DMA current host receive descriptor register	ETH_DMA_CHRXDER	0x104Ch	32	0x00000000h
ETH_DMA current host send buffer register	ETH_DMA_CHTXBFR	0x1050h	32	0x00000000h
ETH_DMA current host receive buffer register	ETH_DMA_CHRXBFR	0x1054h	32	0x00000000h
ETH_MMCC MMC Control Register	ETH_MMCC_MMCCCTRL	0x0100h	32	0x00000000h
ETH_MMCC Rx Statistics Status Register	ETH_MMCC_REVSTSR	0x0104h	32	0x00000000h
ETH_MMCC Tx Statistics Status Register	ETH_MMCC_TRSSTSR	0x0108h	32	0x00000000h
ETH_MMCC Rx Interrupt Control Register	ETH_MMCC_RITCTRL	0x010Ch	32	0x00000000h
ETH_MMCC Tx Interrupt Control Register	ETH_MMCC_TITCTRL	0x0110h	32	0x00000000h
ETH_MMCC Rx unicast good frame statistics register	ETH_MMCC_RXUNGFR	0x01C4h	32	0x00000000h
ETH_MMCC Rx Multicast Good Frame	ETH_MMCC_RXMUGFR	0x0190h	32	0x00000000h

Register name	Symbol	Offset	Bit width	Reset value
Statistics Register				
ETH_MMC Rx broadcast good frame statistics register	ETH_MMC_RXBRGFR	0x018Ch	32	0x00000000h
ETH_MMC RxCRC error frame statistics register	ETH_MMC_RXCREFR	0x0194h	32	0x00000000h
ETH_MMC Rx Alignment Error Frame Statistics Register	ETH_MMC_RXALEFR	0x0198h	32	0x00000000h
ETH_MMC Rx short frame error frame statistics register	ETH_MMC_RXRUEFR	0x019Ch	32	0x00000000h
ETH_MMC Rx length error frame statistics register	ETH_MMC_RXLEEFR	0x01C8h	32	0x00000000h
ETH_MMC Rx out-of-range error frame statistics register	ETH_MMC_RXOREFR	0x01CCh	32	0x00000000h
ETH_MMC Tx unicast good frame statistics register	ETH_MMC_TXUNGFR	0x0168h	32	0x00000000h
ETH_MMC Tx Multicast Good Frame Statistics Register	ETH_MMC_TXMUGFR	0x0120h	32	0x00000000h
ETH_MMC Tx broadcast good frame statistics register	ETH_MMC_TXBRGFR	0x011Ch	32	0x00000000h
ETH_MMC Tx Delay Error Frame Statistics Register	ETH_MMC_TXDEEFR	0x0154h	32	0x00000000h
ETH_MMC Tx Carrier Error Frame Statistics Register	ETH_MMC_TXCAEFR	0x0160h	32	0x00000000h
ETH_MMC Tx Delay Collision Error Frame Statistics Register	ETH_MMC_TXLCEFR	0x0158h	32	0x00000000h
ETH_MMC Tx Excessive Collision Error Frame Statistics Register	ETH_MMC_TXECEFR	0x015Ch	32	0x00000000h
ETH_MMC Tx Excessive Delay Error Frame Statistics Register	ETH_MMC_TXEDEFR	0x016Ch	32	0x00000000h

39.6.1 ETH_MAC register

39.6.1.1 ETH_MAC Interface Selection Register (ETH_MAC_IFCONFR)

Reset value: 0x00000000h

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										TCK INV	RCK INV	-	-	-	IF SEL
Bit	Marking	Place name	Function	Read and write											
b31~b6	Reserved	-	Read as "0", write as "0"	R/W											
b5	TCKINV	TX clock inversion	0: The clock input from the ETH_MII_TXCLK port maintains the original polarity 1: Invert the polarity of the clock input from the ETH_MII_TXCLK port	R/W											
b4	RCKINV	RX clock inversion	0: The clock input from the ETH_MII_RMII_RXCLK port maintains the original polarity 1: Invert the polarity of the clock input from the ETH_MII_RMII_RXCLK port	R/W											
b3~b1	Reserved	-	Read as "0", write as "0"	R/W											
b0	IFSEL	Interface select bits	0: MII interface 1: RMII interface	R/W											

39.6.1.2 ETH_MAC MAC Configuration Register (ETH_MAC_CONFIGR)

Reset value: 0x00008000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	SAIRC[2:0]		-	-	CST	-	MWD	MJB	-	-	-	IFG[2:0]		DCRS	

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	FES	DO	LM	DM	IPCO	DRTY	-	ACS	BL[1:0]		DC	TE	RE	-	-

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	Read as "0", write as "0"	R/W
b30~b28	SAIRC	Source address insertion and replacement	010: Insert the address value in the MAC address register 0 as the SA address into the transmit frame 011: Insert the address value in the MAC address register 1 as the SA address into the transmit frame 110: Replace the SA field in the transmission frame with the address value in the MAC address register 0 as the SA address 111: Replace the SA field in the transmission frame with the address value in the MAC address register 1 as the SA address x0: The handling of the SA field address in the transmit frame is controlled by the SAIRC bit of the Tx descriptor TDES1	R/W
b27~b26	Reserved	-	Read as "0", write as "0"	R/W
b25	CST	TYPE frame FCS removal	0: The function is invalid 1: Strip and discard the last 4 bytes (FCS field) of all TYPE (type field greater than 0x0600) frames before forwarding the frame to the application	R/W
b24	Reserved	-	Read as "0", write as "0"	R/W
b23	MWD	Shielded watchdog	0: The MAC allows frames to be received up to 2048 bytes and will truncate any bytes received beyond this limit 1: MAC disables watchdog timer on receiver and can receive frames up to 16384 bytes	R/W
b22	MJB	Block Jabber	0: The application sends more than 2048 bytes of data during transmission, and the MAC will truncate the sender 1: The MAC disables the Jabber timer on the transmitter and can send frames up to 16384 bytes	R/W
b21~b20	Reserved	-	Read as "0", write as "0"	R/W
b19~b17	IFG	interframe gap	Minimum gap between frames during transmission 000: 96-bit time 001: 88-bit time 010: 80-bit time 111: 40-bit time <i>Note: In half-duplex mode, the minimum IFG can only be configured for 64-bit time (IFG=100), lower values are not considered</i>	R/W
b16	DCRS	shielded carrier sense	0: When the MII_CRS signal is valid during frame transmission in half-duplex mode, the MAC transmitter will generate a carrier sense error or even abort transmission 1: MII_CRS signal during frame transmission in half-duplex mode is ignored, no error will be generated due to carrier loss or no carrier during this transmission period <i>Note: This bit is only valid in half-duplex mode, invalid in full-duplex mode</i>	R/W
b15	Reserved	-	Read as "1", write as "1"	R/W
b14	FES	Speed selection	0: 10Mbps 1: 100Mbps	R/W
b13	DO	Block receiving own packets	0: In half duplex mode, the MAC receives all packets from the PHY	R/W

			1: In half-duplex mode, the MAC prohibits receiving frames <i>Note: This bit is only valid in half-duplex mode, invalid in full-duplex mode</i>	
b12	LM	Loopback mode	0: Loopback mode is invalid 1: Loopback mode is valid	R/W
b11	DM	full duplex mode	0: Half-duplex mode 1: full duplex mode	R/W
b10	IPCO	Checksum Offload function	0: IP Checksum Offload function is invalid 1: IP Checksum Offload function is valid	R/W
b9	DRTY	Block retry	0: When a collision occurs in MII mode, the MAC will try to retry according to the setting of the BL bit of this register 1: When a collision occurs in MII mode, the MAC will ignore the current frame transmission and report the frame abort with an excessive collision error in the sending frame state, that is, the MAC will only try to transmit once <i>Note: This bit is only valid in half-duplex mode, invalid in full-duplex mode</i>	R/W
b8	Reserved	-	Read as "0", write as "0"	R/W
b7	ACS	Automatic removal of PAD/FCS	0: The MAC transmits all received frames to the application regardless of the length of the received frame 1: MAC automatically removes the PAD/FCS field on the received frame when the length field value is less than 1536 bytes, and does not remove the PAD/FCS field when the length field is greater than or equal to 1536 bytes	R/W
b6~b5	BL	back limit	Backoff limit determines a random integer (r) time slice delay (512-bit time) that the MAC waits before rescheduling a transmission during retry after a collision 00 k = min(n, 10) 01 k = min(n, 8) 10 k = min(n, 4) 11 k = min(n, 1) where n = number of resend attempts. The value range of the random integer r is: 0<= r<2^k <i>Note: This bit is only valid in half-duplex mode, invalid in full-duplex mode</i>	R/W
b4	DC	delayed check	0: Disable the delayed check function. The MAC is delayed until the CRS signal becomes invalid 1: Enable delayed check function. When the transmit state machine is delayed for more than 24288 bit time, the MAC will indicate the frame abort state and set the excessive delay error bit to 1 in the transmit frame state <i>Note 1: Generation of delay: The delay count starts when the transmitter is ready to transmit but is blocked due to a valid CRS (Carrier Sense) signal on the MII. The delay time is non-cumulative, if the transmitter delays by 10000 bit times, it re-executes the transmission, collision, back-off, and then resets the delay counter to 0 and starts counting again after the back-off is completed</i> <i>Note 2: This bit is only valid in half-duplex mode, invalid in full-duplex mode</i>	R/W
b3	TE	send enable	0: Send invalid 1: Send enable	R/W
b2	RE	receive enable	0: Reception is invalid 1: Receive enable	R/W
b1~b0	-	-	Read as "0", write as "0"	R/W

39.6.1.3 ETH_MAC MAC Flow Control Register (ETH_MAC_FLOCLR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PAuset[15:0]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved								DZPQ	-	PLT[1:0]	UNP	RFE	TFE	FCA_BPA	
Bit	Marking	Place name	Function	Read and write											
b31~b16	PAuset	Pause time	This bit indicates the value to use for the pause time field in the transmit control frame	R/W											
b15~b8	Reserved	-	Read as "0", write as "0"	R/W											
b7	DZPQ	Mask zero time slice frames	0: When the flow control signal is valid, the zero time slice pause control frame is normally generated 1: When the flow control signal is valid, the generation of zero time slice pause control frame is prohibited	R/W											
b6	Reserved	-	Read as "0", write as "0"	R/W											
b5~b4	PLT	Pause lower threshold	Configure the threshold of the pause timer, when this value is reached, the PAUSE frame will be automatically retransmitted 00: Pause time minus 4 slots 01: Pause time minus 28 slots 10: Pause time minus 144 slots 11: Pause time minus 256 slots <i>Note 1: A time slot is defined as the time required to transmit every 512 bits (64 bytes) of the MII interface</i>	R/W											
b3	UNP	Unicast pause frame detection	0: The MAC only detects pause frames with a unique multicast address specified in the 802.3x standard 1: In addition to detecting pause frames with unique multicast addresses, the MAC also detects pause frames with station unicast addresses specified by the ETH_MAC_MACADHR0 and ETH_MAC_MACADLR0 registers	R/W											
b2	RFE	Receive flow control enable	0: Disable decoding of pause frames 1: The MAC decodes the received pause frame and prohibits it from being sent within the specified pause time	R/W											
b1	TFE	Send Flow Control Enable	In full duplex mode: 0: The MAC is prohibited from transmitting any pause frames 1: Enable flow control operation, MAC sends pause frame In half duplex mode: 0: Disable back pressure function 1: Enable back pressure operation	R/W											
b0	FCA_BPA	flow control state	This bit should be read as 0 before writing to the flow control register In full duplex mode: When this bit is set, the MAC core initiates a pause control frame. During control frame transmission, this bit remains set to indicate that frame transmission is in progress. The MAC resets this bit to 0 when the pause control frame is sent. After this bit is cleared, the flow control register can be written to When in half-duplex mode: When this bit is set, the MAC core will assert the backpressure feature. During backpressure operation, when the MAC receives a new frame, the	R/W											

transmitter starts sending a Jam signal that causes a
collision

39.6.1.4 ETH_MAC MAC Status Register (ETH_MAC_MACSTS)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16									
Reserved			TFF	TFNE	-	TFWA	TFRS[1:0]		MTP	MTS[1:0]		MTEA												
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0									
Reserved			RFFL[1:0]		-	RFRS[1:0]		RFWA	-	MRS[1:0]		MREA												
Bit	Marking	Place name			Function										Read and write									
b31~b26	Reserved	-			Read as "0", write as "0"										R/W									
b25	TFF	Tx FIFO full			0: Tx FIFO is not full 1: Tx FIFO is full and can no longer receive transmitted frames										R									
b24	TFNE	Tx FIFO is not empty			0: Tx FIFO empty 1: Tx FIFO is not empty, there are still unsent frames										R									
b23	Reserved	-			Read as "0", write as "0"										R/W									
b22	TFWA	Tx FIFO write valid			0: Tx FIFO write invalid 1: The Tx FIFO write controller is active and is transferring data to the Tx FIFO										R									
b21~b20	TFRS	Tx FIFO read status			00: idle state 01: Read status (data is transferred from Tx FIFO to MAC transmitter) 10: Wait for Tx Status from MAC transmitter 11: Write received Tx Status or refresh Tx FIFO										R									
b19	MTP	MAC Transmitter Suspended			0: The MAC transmitter is in the state of sending data 1: The MAC transmitter is in a paused state (in full duplex mode), no frame transmission is scheduled										R									
b18~b17	MTS	MAC Transmitter Status			00: idle state 01: Wait for the state of the previous frame or the end of the IFG/rollback phase 10: Generate and send pause control frame (in full duplex mode) 11: Transmit the input frame to be sent										R									
b16	MTEA	MAC MII Transmit Engine Status			0: MAC MII transmit engine is in idle state 1: The MAC MII transmit engine is actively transmitting data and is not in an idle state										R									
b15~b10	Reserved	-			Read as "0", write as "0"										R/W									
b9~b8	RFFL	Rx FIFO fill level			00: Rx FIFO is empty 01: Rx FIFO fill level below flow control, deactivation threshold 10: Rx FIFO fill level above flow control, active threshold 11: Rx FIFO is full										R									
b7	Reserved	-			Read as "0", write as "0"										R/W									
b6~b5	RFRS	Rx FIFO read status			00: idle state 01: Read frame data 10: Read frame status or timestamp 11: Refresh frame data and status										R									
b4	RFWA	Rx FIFO write valid			0: Rx FIFO write invalid 1: Rx FIFO write active and transferring received frame to Rx FIFO										R									
b3	Reserved	-			Read as "0", write as "0"										R/W									
b2~b1	MRS	MAC Receiver Status			00: idle 01: The FIFO write controller in the MAC receiver is working 10: The FIFO read controller in the MAC receiver is working 11: The FIFO read and write controllers in the MAC receiver are working										R									
b0	MREA	MAC MII Receive Engine Status			0: MAC MII receive engine is in idle state 1: The MAC MII receive engine is actively receiving data and is not idle										R									

39.6.1.5 ETH_MAC Interrupt Status Register (ETH_MAC_INTSTSR)

Reset value: 0x00000000h

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						TSPIS	-	-	MMC TXIS	MMC RXIS	MMC IS	PMTIS	-	-	-
Bit	Marking	Place name	Function										Read and write		
b31~b10	Reserved	-	Read as "0", write as "0"										R/W		
b9	TSPIS	PTP timestamp breaks	This bit is set to 1 when the system time value is equal to or greater than the value specified in the target time register, or when the system time overflows <i>Note: This bit is cleared when Bit0 of the ETH PTP TSPSTSR register is read</i>										R		
b8~b7	Reserved	-	Read as "0", write as "0"										R/W		
b6	MMCTXIS	MMC sends statistics interrupt	When an interrupt is generated in the ETH_MMCTCLR register, this bit goes high <i>Note: This bit is cleared when all interrupts described in the ETH_MMCTCLR register are cleared</i>										R		
b5	MMCRXIS	MMC receive statistics interrupt	When an interrupt is generated in the ETH_MMCRXIS register, this bit goes high <i>Note: This bit is cleared when all interrupts described in the ETH_MMCRXIS register are cleared</i>										R		
b4	MMCIS	MMC statistics outage	The MMCTXIS or MMCRXIS bit of this register has a high level, and this bit becomes a high level <i>Note: This bit is cleared when all b6~b5 of this register become 0</i>										R		
b3	PMTIS	PMT interrupt	This bit goes high when a magic packet or remote wake-up frame is received in Power Down mode <i>Note: This bit is cleared when both the WKFR and MPFR bits in the ETH_MAC_PMTCLR register become 0</i>										R		
b2~b0	Reserved	-	Read as "0", write as "0"										R/W		

39.6.1.6 ETH_MAC Interrupt Mask Register (ETH_MAC_INTMSKR)

Reset value: 0x00000000h

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				TSPIM		Reserved				PMTIM		Reserved													
Bit	Marking	Place name	Function	Read and write																					
b31~b10	Reserved	-	Read as "0", write as "0"	R/W																					
b9	TSPIM	PTP timestamp interrupt mask	0: PTP timestamp interrupt is not masked 1: PTP timestamp interrupt mask	R/W																					
b8~b4	Reserved	-	Read as "0", write as "0"	R/W																					
b3	PMTIM	PMT interrupt mask	0: PMT interrupt is not masked 1: PMT interrupt mask	R/W																					
b2~b0	Reserved	-	Read as "0", write as "0"	R/W																					

39.6.1.7 ETH_MAC SMI Address Register (ETH_MAC_SMIADDR)

Reset value: 0x00000000h

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							
SMIA[4:0]				SMIR[4:0]				SMIC[3:0]				SMIW	SMIB									
Bit	Marking	Place name	Function	Read and write																		
b31~b16	Reserved	-	Read as "0", write as "0"	R/W																		
b15~b11	SMIA	physical layer address	Indicates which of 32 possible PHY devices to access	R/W																		
b10~b6	SMIR	PHY register	Indicates which register to select in the selected PHY device	R/W																		
b5~b2	SMIC	MDC clock	Determine the MDC clock frequency based on the system frequency (PCLK1): Option System Frequency MDC Clock 0000 60-100MHz system frequency/42 0001 100-120MHz system frequency/62 0010 20-35MHz system frequency/16 0011 35-60MHz system frequency/26 Please do not set other values	R/W																		
b1	SMIW	SMI read and write	0: SMI read operation 1: SMI write operation	R/W																		
b0	SMIB	SMI access busy	This bit should read logic 0 before writing to ETH_MAC_SMIADDR and ETH_MAC_SMIDATR This bit must also be reset to 0 during writing to ETH_MAC_SMIADDR During a PHY register access, this bit is set to 1 by the application to indicate that a read or write access is in progress During a write operation to the PHY, ETH_MAC_SMIDATR should always be valid until the MAC clears this bit; during a read operation to the PHY, ETH_MAC_SMIDATR is always invalid until the MAC clears this bit. After this bit is cleared, the value can be written to ETH_MAC_SMIADDR	R/W																		

39.6.1.8 ETH_MAC SMI Data Register (ETH_MAC_SMIDATR)

Reset value: 0x00000000h

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
SMID[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15~b0	SMID	SMI data	16-bit data value read from the PHY after a site-managed read, or 16-bit data value to be written to the PHY before a site-managed write	R/W

39.6.1.9 ETH_MAC PMT Control Status Register (ETH_MAC_PMTCLR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
RTWKFR	-	-	-	-	RTWKPT[2:0]		Reserved								
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved					RTWKTR	GLUB	-	-	WKFR	MPFR	-	-	WKEN	MPEN	PWDN
<hr/>															
Bit	Marking		Place name		Function								Read and write		
b31	RTWKFR		Remote wake-up frame filter register pointer reset		0: The remote wake-up frame filter register pointer is not reset 1: Remote wake-up frame filter register pointer reset <i>Note: This bit is automatically cleared after reset</i>								R/W		
b30~b27	Reserved		-		Read as "0", write as "0"								R/W		
b26~b24	RTWKPT		Remote Wakeup Frame Filtering Register Pointer Location		This bit indicates which register the pointer of the remote wake-up frame filter register is on (the value is 0~7)								R		
b23~b11	Reserved		-		Read as "0", write as "0"								R/W		
b10	RTWKTR		Remote wake-up frame transmission		After this bit is set, the MAC no longer receives packets other than magic packets and remote wake-up frames, and forwards the received remote wake-up frames to the application <i>Note 1: This bit will automatically become 0 after receiving the magic data packet or remote wake-up frame; it can also be reset by software before receiving the magic data packet or remote wake-up frame</i> <i>Note 2: When the GLBU bit or WKEN bit or MPEN bit of this register is 1 and the PWDN bit is 0, this bit must be set to 1</i> <i>Note 3: When the PWDN bit is 1, the setting of this bit is invalid</i>								R/W		
b9	GLBU		global unicast		0: Treat any unicast frame filtered by the MAC (DAF) address as invalid 1: Treat any unicast frame filtered by the MAC (DAF) address as a wakeup frame								R/W		
b8~b7	Reserved		-		Read as "0", write as "0"								R/W		
b6	WKFR		Receive remote wakeup frame		0: No remote wakeup frame received 1: Remote wake-up frame received <i>Note: This bit is automatically cleared after reading</i>								R		
b5	MPFR		receive magic packet		0: no magic packet received 1: Magic packet received <i>Note: This bit is automatically cleared after reading</i>								R		
b4~b3	Reserved		-		Read as "0", write as "0"								R/W		
b2	WKEN		Remote Wakeup Frame Enable		0: do not receive remote wake-up frames 1: Receive remote wake-up frame, trigger PMT event								R/W		
b1	MPEN		Magic Packet Enable		0: do not receive magic packets 1: Receive magic packet, trigger PMT event								R/W		
b0	PWDN		Power Down Mode Enable		0: Active mode 1: Power Down Mode All received frames are discarded. Only when magic data packets or remote wake-up frames are received, this bit will be automatically cleared and the Power Down mode will be exited; frames received after this bit is cleared will be forwarded to the application <i>Note: This bit can only be set when the WKEN bit or the MPEN bit is set</i>								R/W		

39.6.1.10 ETH_MAC Remote Wakeup Frame Filter Register (ETH_MAC_RTWKFFR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
WKUPFRMFT [31:16]															

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

Bit	Marking	Place name	Function	Read and write
b31~b0	WKUPFRMFT[31:0]	Remote Wakeup Frame Filter Value	The application will write/read the Remote Wakeup Frame Filter Register through this address. See the "Remote Wakeup Frame Filtering" section for details	R/W

39.6.1.11 ETH_MAC MAC Frame Filtering Control Register (ETH_MAC_FLTCTRLR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
RA		Reserved								DNTU	IPFE	-	-	-	VTFE
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	Reserved		HPF	SAF	SAIF	PCF[1:0]		DBF	PMF	DAIF	HMC	HUC	PR		
<hr/>															
Bit	Marking	Place name	Function										Read and write		
b31	RA	receive all	0: The MAC receiver transmits the received frame filtered by the address to the application 1: The MAC receiver delivers all received frames to the application, regardless of whether they have passed address filtering										R/W		
b30~b22	Reserved	-	Read as "0", write as "0"										R/W		
b21	DNTU	Discard no TCP/UDP packets	0: The MAC receiver delivers all received IP packets to the application, regardless of whether they have TCP/UDP fields or not 1: The MAC receiver discards all IP packets without TCP or UDP fields										R/W		
b20	IPFE	L3&L4 filter enable	0: The MAC receiver delivers all received IP packets to the application, regardless of whether they have passed L3 & L4 filtering 1: The MAC receiver discards all IP packets that do not match the filter value set by L3&L4, and only processes IP packets that pass L3&L4 filtering										R/W		
b19~b17	Reserved	-	Read as "0", write as "0"										R/W		
b16	VTFE	VLAN tag filtering enabled	0: The MAC receiver passes all received VLAN frames to the application, regardless of whether they have been filtered by VLAN tags 1: The MAC receiver discards VLAN frames that do not match the filter value set by the VLAN tag										R/W		
b15~b11	Reserved	-	Read as "0", write as "0"										R/W		
b10	HPF	Hash filter or perfect filter choice	0: If the HUC bit or HMC bit of this register is set, the address filter only transmits frames that match the hash filter 1: If the HUC bit or HMC bit of this register is set, the address filter transmits frames that match the perfect filter or Hash filter										R/W		
b9	SAF	source address filtering	0: The MAC receiver delivers all received frames to the application, regardless of whether they have passed source address filtering 1: The MAC receiver discards frames that do not match the filter value set by the source address										R/W		
b8	SAIF	Source address inversion filtering	0: When the SA field of the frame received by the MAC receiver does not match the set filtering value, it is considered as a filtering failure 1: When the SA field of the frame received by the MAC receiver matches the set filter value, it is considered as a filter failure										R/W		
b7~b6	PCF	transmit control frame	Do the following for forwarding control frames (including unicast pause frames and multicast pause frames): 00: The MAC blocks all control frames from reaching the application 01: The MAC forwards all control frames except pause control frames to the application, even if address filtering for these frames fails 10: The MAC forwards all control frames to the application, even if address filtering for these frames fails 11: MAC forwards control frames that pass address filtering <i>A few notes about pausing control frames:</i> <i>Note 1: In full duplex mode, the processing of the pause control frame is determined by the RFE bit in the flow control register (ETH_MAC_FLOCTRLR)</i>										R/W		

			<i>Note 2: When the UNP bit of the flow control register (ETH_MAC_FLOCLR) is set to 1, when the destination address of the received frame matches the value set in the MAC address register 0, the frame is considered to be a unicast pause frame</i>
			<i>Note 3: When the TYPE segment value of the received frame is 0x8808 or the OPCODE segment value is 0x0001, the frame is considered to be a pause control frame</i>
b5	DBF	Block broadcast frames	<p>0: The address filter does not filter received broadcast frames</p> <p>1: The address filter filters all incoming broadcast frames <i>Note: When this bit is 1, its priority is higher than other filter settings</i></p>
b4	PMF	transmit all multicast frames	<p>0: Perform address filtering on the received frame with the multicast destination address (the first bit of the destination address field is 1). The filtering method is determined by the HMC bit of this register.</p> <p>1: Do not perform address filtering on received frames with a multicast destination address (the first bit of the destination address field is 1)</p>
b3	DAIF	Destination address inversion filtering	<p>0: When the destination address of the unicast frame or multicast frame received by the MAC receiver does not match the set filter value, it is considered as a filter failure</p> <p>1: When the destination address of the unicast frame or multicast frame received by the MAC receiver matches the set filter value, it is considered as a filter failure</p>
b2	HMC	Hash multicast control	<p>0: MAC performs perfect destination address filtering on multicast frames, i.e. compares the DA field with the set filtering value</p> <p>1: MAC performs Hash destination address filtering on multicast frames</p>
b1	HUC	Hash unicast control	<p>0: MAC performs perfect destination address filtering on unicast frames, that is, compares the DA field with the set filtering value</p> <p>1: MAC performs Hash destination address filtering on unicast frames</p>
b0	PR	blend mode	<p>0: The address filter performs normal address filtering</p> <p>1: The address filter transmits all incoming frames regardless of destination or source address</p>

39.6.1.12 ETH_MAC MAC address high register 0 (ETH_MAC_MACADHR0)

Reset value: 0x8000FFFFh

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
AE0	Reserved														

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADDRH0[15:0]															

Bit	Marking	Place name	Function	Read and write
b31	AE0	address enable 0	1: Use MAC address 0 for destination address perfect filtering <i>Note: This bit is always 1</i>	R
b30~b16	Reserved	-	Read as "0", write as "0"	R/W
b15~b0	ASDDRHO	High address value 0	The MAC uses this field to filter received frames and to insert the MAC address into transmit flow control (pause) frames	R/W

39.6.1.13 ETH_MAC MAC address low register 0 (ETH_MAC_MACADL0)

Reset value: 0xFFFFFFFFh

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
ADDRL0[31:16]															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADDRL0[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b0	ADDRL0	low address value 0	The MAC uses this field to filter received frames and to insert the MAC address into transmit flow control (pause) frames	R/W

39.6.1.14 ETH_MAC MAC address high register 1 (ETH_MAC_MACADHR1)

Reset value: 0x0000FFFFh

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
AE1	SA1	MBC1[5:0]										Reserved			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADDRH1[15:0]															
Bit	Marking	Place name			Function								Read and write		
b31	AE1	Address Enable 1			0: Ignore address filtering for MAC address 1 1: Perfect filtering with MAC address 1								R/W		
b30	SA1	Address selection 1			0: Use MAC address 1 to filter the DA of the received frame 1: Use MAC address 1 to filter the SA of the received frame								R/W		
b29~b24	MBC1	byte mask 1			Mask corresponding bytes when address filtering b29: Mask ADDRH1[15:8] bits b28: Mask ADDRH1[7:0] bits b27: Mask ADDRL1[31:24] bits b26: Mask ADDRL1[23:16] bits b25: Mask ADDRL1[15:8] bits b24: Mask ADDRL1[7:0] bits								R/W		
b23~b16	Reserved	-			Read as "0", write as "0"								R/W		
b15~b0	ASDDRH1	High address value 1			MAC uses this field to filter received frames								R/W		

39.6.1.15 ETH_MAC MAC address low register 1 (ETH_MAC_MACADLR1)

Reset value: 0xFFFFFFFFh

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
ADDRL1[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADDRL1[15:0]															
Bit	Marking	Place name			Function								Read and write		
b31~b0	ADDRL1	low address value 1			MAC uses this field to filter received frames								R/W		

39.6.1.16 ETH_MAC MAC address high register 2 (ETH_MAC_MACADHR2)

Reset value: 0x0000FFFFh

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
AE2	SA2	MBC2[5:0]										Reserved			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADDRH2[15:0]															
Bit	Marking	Place name			Function								Read and write		
b31	AE2	Address Enable 2			0: Ignore address filtering for MAC address 2 1: Perfect filtering with MAC address 2								R/W		
b30	SA2	Address selection 2			0: Use MAC address 2 to filter the DA of the received frame 1: Use MAC address 2 to filter the SA of the received frame								R/W		
b29~b24	MBC2	byte mask 2			Mask corresponding bytes when address filtering b29: Mask ADDRH2[15:8] bits b28: Mask ADDRH2[7:0] bits b27: Mask ADDRL2[31:24] bits b26: Mask ADDRL2[23:16] bits b25: Mask ADDRL2[15:8] bits b24: Mask ADDRL2[7:0] bits								R/W		
b23~b16	Reserved	-			Read as "0", write as "0"								R/W		
b15~b0	ASDDRH2	High address value 2			MAC uses this field to filter received frames								R/W		

39.6.1.17 ETH_MAC MAC address low register 2 (ETH_MAC_MACADLR2)

Reset value: 0xFFFFFFFFh

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
ADDRL2[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADDRL2[15:0]															
Bit	Marking	Place name			Function								Read and write		
b31~b0	ADDRL2	low address value 2			MAC uses this field to filter received frames								R/W		

39.6.1.18 ETH_MAC MAC address high register 3 (ETH_MAC_MACADHR3)

Reset value: 0x0000FFFFh

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
AE3	SA3	MBC3[5:0]										Reserved				
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
ADDRH3[15:0]																
Bit	Marking	Place name										Function				Read and write
b31	AE3	address enable 3										0: Ignore address filtering for MAC address 3 1: Perfect filtering with MAC address 3	R/W			
b30	SA3	Address selection 3										0: Use MAC address 3 to filter the DA of the received frame 1: Use MAC address 3 to filter the SA of the received frame	R/W			
b29~b24	MBC3	byte mask 3										Mask corresponding bytes when address filtering b29: Mask ADDRH3[15:8] bits b28: Mask ADDRH3[7:0] bits b27: Mask ADDRL3[31:24] bits b26: Mask ADDRL3[23:16] bits b25: Mask ADDRL3[15:8] bits b24: Mask ADDRL3[7:0] bits	R/W			
b23~b16	Reserved	-										Read as "0", write as "0"	R/W			
b15~b0	ASDDRH3	High address value 3										MAC uses this field to filter received frames	R/W			

39.6.1.19 ETH_MAC MAC address low register 3 (ETH_MAC_MACADLR3)

Reset value: 0xFFFFFFFFh

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
ADDRL3[31:16]																
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
ADDRL3[15:0]																
Bit	Marking	Place name										Function				Read and write
b31~b0	ADDRL3	low address value 3										MAC uses this field to filter received frames	R/W			

39.6.1.20 ETH_MAC MAC address high register 4 (ETH_MAC_MACADHR4)

Reset value: 0x0000FFFFh

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
AE4	SA4	MBC4[5:0]										Reserved			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADDRH4[15:0]															
<hr/>															
Bit	Marking	Place name			Function						Read and write				
b31	AE4	Address Enable 4			0: Ignore address filtering for MAC address 4 1: Perfect filtering with MAC address 4						R/W				
b30	SA4	Address selection 4			0: Use MAC address 4 to filter the DA of the received frame 1: Use MAC address 4 to filter the SA of the received frame						R/W				
b29~b24	MBC4	byte mask 4			Mask corresponding bytes when address filtering b29: Mask ADDRH4[15:8] bits b28: Mask ADDRH4[7:0] bits b27: Mask ADDRL4[31:24] bits b26: Mask ADDRL4[23:16] bits b25: Mask ADDRL4[15:8] bits b24: Mask ADDRL4[7:0] bits						R/W				
b23~b16	Reserved	-			Read as "0", write as "0"						R/W				
b15~b0	ASDDRH4	High address value 4			MAC uses this field to filter received frames						R/W				

39.6.1.21 ETH_MAC MAC address low register 4 (ETH_MAC_MACADLR4)

Reset value: 0xFFFFFFFFh

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
ADDRL4[31:16]															
<hr/>															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADDRL4[15:0]															
Bit	Marking	Place name			Function						Read and write				
b31~b0	ADDRL4	low address value 4			MAC uses this field to filter received frames						R/W				

39.6.1.22 ETH_MAC MAC Hash table high register (ETH_MAC_HASHTHR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
HTH[31:16]															

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

Bit	Marking	Place name	Function	Read and write
b31~b0	HTH	Hash Table high value	High 32-bit value of Hash Table	R/W

39.6.1.23 ETH_MAC MAC Hash table low register (ETH_MAC_HASHTLR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
HTL[31:16]															

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

Bit	Marking	Place name	Function	Read and write
b31~b0	HTL	Hash Table low value	Lower 32-bit value of Hash Table	R/W

39.6.1.24 ETH_MAC VLAN Tag Transmission Control Register (ETH_MAC_VTACTLR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved												VLANS	VLANC[1:0]		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
VLANV[15:0]															
Bit	Marking	Place name	Function				Read and write								
b31~b19	Reserved	-	Read as "0", write as "0"				R/W								
b18	VLANS	VLAN tag selection	0: When sending a VLAN frame, the processing of the tag field is determined by the setting of the VLANC control bit of TDES0 of the Tx descriptor 1: When sending a VLAN frame, the processing of the tag field is determined by the setting of the VLANC control bit of this register				R/W								
b17~b16	VLANC	VLAN tag control	When sending VLAN frames, the processing of the tag field: 00: No operation, send directly 01: Delete the TYPE field and tag field of the VLAN frame 10: After inserting the TYPE field, insert the value of the VLANV bit of the register as a tag field, regardless of whether the original frame contains a tag field 11: Replace the tag field in the original frame with the value of the VLANV bit of this register				R/W								
b15~b0	VLANV	VLAN tag value	User-defined VLAN tag field value for inserting or replacing tag fields when sending frames				R/W								

39.6.1.25 ETH_MAC VLAN Tag Acceptance Filter Register (ETH_MAC_VTAFLTR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved														VTHM	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
VLFLT[15:0]															
Bit	Marking	Place name	Function				Read and write								
b31~b20	Reserved	-	Read as "0", write as "0"				R/W								
b19	VTHM	VLAN tag hash table enabled	0: The VLAN tag hash filtering function is invalid 1: VLAN tag hash filtering function is enabled				R/W								
b18	Reserved	-	Read as "0", write as "0"				R/W								
b17	VTIM	VLAN tag inversion filtering	0: When the tag field of the VLAN frame does not match the filter value set by the VLFLT bit of this register, it is considered that the filter fails. 1: When the tag field of the VLAN frame matches the filter value set by the VLFLT bit of this register, it is determined that the filter fails.				R/W								
b16	VTAL	12BIT VLAN Comparison	0: The full 16 bits of the VLAN tag field are compared with the VLFLT setting 1: Compare the lower 12 bits of the VLAN tag field with the VLFLT setting <i>Note: The setting of this bit also applies to the Hash filtering method.</i>				R/W								
b15~b0	VLFLT	VLAN tag filter value	This field contains the comparison value of the tag field used to filter VLAN frames, this bit is compared with the 15th and 16th bytes of the received VLAN frame				R/W								

39.6.1.26 ETH_MAC VLAN Hash Table Register (ETH_MAC_VLAHTBR)

Reset value: 0x00000000h

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
VLHT[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Reset value when read	R/W
b15~b0	VLHT	VLAN Hash Table	VLAN Hash Table value	R/W

39.6.1.27 ETH_MAC_LAY3LAY4 Control Register (ETH_MAC_L34CTRLR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16												
Reserved										L4 DPIM	L4 DPM	L4 SPIM	L4 SPM	-	L4 PEN												
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0												
										L3 DAIM	L3 DAM	L3 SAIM	L3 SAM	-	L3 PEN												
<hr/>																											
Bit	Marking	Place name	Function	Read and write																							
b31~b22	Reserved	-	Read as "0", write as "0"	R/W																							
b21	L4DPIM	L4 destination port inversion filtering	0: When the target port field of L4 does not match the set filter value, it is considered that the filter fails. 1: When the target port field of L4 matches the set filter value, it is considered that the filter fails. <i>Note: This bit setting is valid only when the L4DPM bit of this register is valid</i>	R/W																							
b20	L4DPM	L4 destination port filtering	0: Ignore the filtering of the L4 destination port field 1: Filter the destination port field of L4	R/W																							
b19	L4SPIM	L4 source port inversion filtering	0: When the source port field of L4 does not match the set filter value, it is determined that the filter fails 1: When the source port field of L4 matches the set filter value, it is determined that the filter fails <i>Note: This bit setting is valid only when the L4SPM bit of this register is valid</i>	R/W																							
b18	L4SPM	L4 source port filtering	0: Ignore the filtering of the L4 source port field 1: Filter the source port field of L4	R/W																							
b17	Reserved	-	Read as "0", write as "0"	R/W																							
b16	L4PEN	L4 protocol enabled	0: Port filtering for L4 TCP protocol frames 1: Port filtering for L4 UDP protocol frames	R/W																							
<hr/>																											
Bit mask setting when filtering L3 destination address: For IPv4: 00000: full target address filtering 00001: Ignore the filtering of the least significant bit LSB[0] of the target address 00010: Ignore the filtering of the least significant bits LSB[1:0] of the target address 00011: Ignore the filtering of the least significant bits LSB[2:0] of the target address 11111: Ignore the filtering of the least significant bits LSB[30:0] of the destination address																											
For IPv6: Bit2~1 of L3HDBM are used as Bit7~6 of L3HSBM <i>Note: This bit is valid only when the L3DAM of this register is valid</i>																											
<hr/>																											
Bit mask setting when filtering the source address: For IPv4: 00000: All source address filtering 00001: Ignore the filtering of the least significant bit LSB[0] of the source address 00010: Ignore the filtering of the least significant bit LSB[1:0] of the source address 00011: Ignore the filtering of the least significant bit LSB[2:0] of the source address 11111: Ignore filtering of the least significant bit LSB[30:0] of the source address																											
For IPv6: 0000000: All source or destination address filtering 0000001: Ignore filtering of the least significant bit LSB[0] of the source or destination address 0000010: Ignore filtering of the least significant bits LSB[1:0] of the source or destination address 0000011: Ignore filtering of the least significant bits LSB[2:0] of the source or destination address 1111111: Ignore filtering of least significant bits																											
<hr/>																											
b10~b6	L3HSBM	L3IP source address high-order mask	11111: Ignore filtering of the least significant bit LSB[30:0] of the source address	R/W																							

			LSB[126:0] of source or destination address <i>Note 1: This bit is only valid when the L3SAM or L3DAM of this register is valid</i> <i>Note 2: When the IPv6 function is selected, and both L3DAM and L3SAM are enabled, the filter control of the source address and the destination address share a set of filter addresses, and the high-order mask control bit is also shared</i>	
b5	L3DAIM	L3 destination address inversion filtering	0: When the destination address field of the L3 IP packet does not match the set filtering value, it is determined that the filtering fails 1: When the destination address field of the L3 IP packet matches the set filter value, it is determined that the filter fails. <i>Note: This bit setting is valid only when the L3DAM bit of this register is valid</i>	R/W
b4	L3DAM	L3 destination address filtering	0: Ignore the filtering of the destination address field of the L3 IP packet 1: Filter the destination address field of the L3 IP packet	R/W
b3	L3SAIM	L3 source address inversion filtering	0: When the source address field of the L3 IP packet does not match the set filtering value, it is determined that the filtering fails 1: When the source address field of the L3 IP packet matches the set filter value, it is determined that the filter fails. <i>Note: This bit setting is valid only when the L3SAM bit of this register is valid</i>	R/W
b2	L3SAM	L3 source address filtering	0: Ignore the filtering of the source address field of the L3 IP packet 1: Filter the source address field of the L3 IP packet	R/W
b1	Reserved	-	Read as "0", write as "0"	R/W
b0	L3PEN	L3 protocol enabled	0: Perform address filtering on L3 IPv4 packets 1: Perform address filtering on L3 IPv6 packets	R/W

39.6.1.28 ETH_MAC LAY4 Port Register (ETH_MAC_L4PORTR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
L4DPVAL[15:0]															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
L4SPVAL[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b16	L4DPVAL	destination port filter value	L4 destination port filter value	R/W
b15~b0	L4SPVAL	source port filter value	L4 source port filter value	R/W

39.6.1.29 ETH_MAC LAY3 address register 0 (ETH_MAC_L3ADDR0)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
L3ADDR0[31:16]															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
L3ADDR0[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b0	L3ADDR0	address filter value 0	L3 address filter value: 1) When L3PEN=0&L3SAM=1, this register is the source address filter value of L3 reported by IPv4 2) When L3PEN=1&L3SAM=1, this register is the L3 source address [31:0] filter value reported by IPv6 3) When L3PEN=1&L3DAM=1, this register is the L3 destination address [31:0] bit filter value reported by IPv6	R/W

39.6.1.30 ETH_MAC LAY3 address register 1 (ETH_MAC_L3ADDRR1)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
L3ADDR1[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
L3ADDR1[15:0]															
Bit	Marking	Place name	Function	Read and write											
b31~b0	L3ADDR1	address filter value 1	L3 address filter value: 1) When L3PEN=0&L3DAM=1, this register is the L3 destination address filter value reported by IPv4 2) When L3PEN=1&L3SAM=1, this register is the filter value of the source address [63:32] of L3 reported by IPv6 3) When L3PEN=1&L3DAM=1, this register is the L3 destination address [63:32] bit filter value reported by IPv6	R/W											

39.6.1.31 ETH_MAC LAY3 address register 2 (ETH_MAC_L3ADDRR2)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
L3ADDR2[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
L3ADDR2[15:0]															
Bit	Marking	Place name	Function	Read and write											
b31~b0	L3ADDR2	address filter value 2	L3 address filter value: 1) When L3PEN=1&L3SAM=1, this register is the filter value of the source address [95:64] of L3 reported by IPv6 2) When L3PEN=1&L3DAM=1, this register is the L3 destination address [95:64] bit filter value reported by IPv6	R/W											

39.6.1.32 ETH_MAC_LAY3 address register 3 (ETH_MAC_L3ADDR3)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
L3ADDR3[31:16]															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
L3ADDR3[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b0	L3ADDR3	address filter value 3	L3 address filter value: 1) When L3PEN=1&L3SAM=1, this register is the filter value of the source address [127:96] of L3 reported by IPv6 2) When L3PEN=1&L3DAM=1, this register is the L3 destination address [127:96] bit filter value reported by IPv6	R/W

39.6.2 ETH_PTP register

39.6.2.1 ETH_PTP Time Stamp Control Register (ETH_PTP_TSPCTLR)

Reset value: 0x00002000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
Reserved														TSPADF	TSPMTSEL[3:2]	
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
<hr/>																
TSPMT SEL[1:0]	TSPOV IPV4	TSPOV IPV6	TSPOV ETH	TSP VER	TSP SSR	TSP EALL	-	-	TSP ADUP	TSP INT	TSP UP	TSPINI	TSP UPSEL	TSP EN		
<hr/>																
<hr/>																
Bit	Marking		Place name			Function							Read and write			
b31~b19	Reserved		-			Read as "0", write as "0"							R/W			
b18	TSPADF		PTP frame filtering			When the PTP frame is sent directly through the Ethernet, if this bit is set to 1, the target address filtering of the PTP frame is performed.							R/W			
b17~b14	TSPMTSEL		PTP packet type selection			Select the packet type to take timestamp snapshots. For details, see the "PTP Packet Types" section.							R/W			
b13	TSPOVIPv4		Receive frames with Over UDP-IPv4 encapsulation type			0: Ignore PTP frames in Over UDP-IPv4 encapsulation 1: Receive PTP frames in Over UDP-IPv4 encapsulation							R/W			
b12	TSPOVIPv6		Receive frames with Over UDP-IPv6 encapsulation type			0: Ignore PTP frames in Over UDP-IPv6 encapsulation 1: Receive PTP frames in Over UDP-IPv6 encapsulation							R/W			
b11	TSPOVETH		Receive frames with Over Ethernet encapsulation type			0: Ignore PTP frames in Over Ethernet encapsulation 1: Receive PTP frames in the form of Over Ethernet encapsulation							R/W			
b10	TSPVER		PTP packet processing version			0: Process the received PTP frame in the IEEE1588v1 version format 1: Process the received PTP frame in the IEEE1588v2 version format							R/W			
b9	TSPSSR		Subsecond base			0: The value of the sub-second register reaches 0x7FFFFFFFH, the timestamp seconds counter is incremented by 1 second, and the sub-second register is reset to zero and counts again 1: The value of the sub-second register reaches 0x3B9AC9FFH, the timestamp seconds register increases by 1 second, and the sub-second register returns to zero and counts again <i>Note 1: 0x3B9AC9FF=0d 999999999 Note 2: Sub-second increments must be programmed correctly based on the PTP reference clock frequency and the value of this bit</i>							R/W			
b8	TSPEALL		enable all frames			When this bit is set to 1, a timestamp snapshot is taken for all frames received							R/W			
b7~b6	Reserved		-			Read as "0", write as "0"							R/W			
b5	TSPADUP		count register update			When this bit is set to 1, the value of the timestamp adder register (ETH_PTP_TSPADDR) will be updated to PTP for precision calibration, and this bit will be automatically cleared when the update is complete <i>Note: This register bit must be read as 0 before it can be set to 1</i>							R/W			
b4	TSPINT		Timestamp interrupt enable			If the system time is greater than the value written in the target time register, a timestamp interrupt will be generated <i>Note: This bit is cleared when a timestamp-triggered interrupt is generated.</i>							R/W			
b3	TSPUP		Timestamp update			When this bit is set, the system time will be updated (added/subtracted) with the values specified in the Timestamp Update Seconds Register (ETH_PTP_TMUSECR) and Timestamp Update Subseconds Register (ETH_PTP_TMUNSER). This bit is automatically cleared at the end of the update <i>Note: This register bit must be set to 1 after both</i>							R/W			

itself and TSPINI are read as 0

b2	TSPINI	Timestamp initialization	When this bit is set, the system time will be initialized (overridden) with the values specified in the Timestamp Update Seconds Register (ETH_PTP_TMUSECR) and Timestamp Update Subseconds Register (ETH_PTP_TMUUNSER). This bit is automatically cleared at the end of initialization <i>Note: This register bit must be read as 0 before it can be set to 1</i>	R/W
b1	TSPUPSEL	Time stamp calibration method	0: Coarse calibration mode 1: Fine calibration method	R/W
b0	TSPEN	PTP function enabled	0: The time stamp function is invalid, the MAC will not add time stamps to the transmitted and received frames 1: The time stamp function is valid, adding time stamps to the transmitted and received frames <i>Note: Once this bit is set to 1, the system time needs to be initialized</i>	R/W

39.6.2.2 ETH_PTP Timestamp Status Register (ETH_PTP_TSPSTSR)

Reset value: 0x00000000h

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										TSERR1	TSTAR1	TSERRO	-	TSTAR0	TSOVF
Bit	Marking	Place name	Function	Read and write											
b31~b6	Reserved	-	Read as "0", write as "0"	R											
b5	TSERR1	target time 1 error	0: The time to be written to target time register 1 is correct 1: The time to write to target time register 1 has elapsed, and the write value is wrong <i>Note: This bit is automatically cleared after being read</i>	R											
b4	TSTAR1	target time 1 reached	0: The system time has not reached the time indicated by the target time register 1 1: The system time reaches the time indicated by the target time register 1 <i>Note: This bit is automatically cleared after being read</i>	R											
b3	TSERRO	target time 0 error	0: The time to be written to the target time register 0 is correct 1: The time to write to the target time register 0 has elapsed, and the written value is wrong <i>Note: This bit is automatically cleared after being read</i>	R											
b2	Reserved	-	Read as "0", write as "0"	R											
b1	TSTAR0	Target time 0 reached	0: The system time has not reached the time indicated by the target time register 0 1: The system time reaches the time indicated by the target time register 0 <i>Note: This bit is automatically cleared after being read</i>	R											
b0	TSOVF	system time overflow	0: The system time has not overflowed 1: System time overflow <i>Note: This bit is automatically cleared after being read</i>	R											

39.6.2.3 ETH_PTP Timestamp Basic Addend Register (ETH_PTP_TSPADDR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TSPADD [31:16]															

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

Bit	Marking	Place name	Function	Read and write
b31~b0	TSPADD	basic addendum	This register is used by software to re-linearly adjust the clock frequency to match the master clock frequency. The value of this register is added to the 32-bit accumulator every clock cycle <i>Note: This register is only valid in fine calibration mode.</i>	R/W

39.6.2.4 ETH_PTP Timestamp Subsecond Adder Register (ETH_PTP_TSPNSAR)

Reset value: 0x00000000h

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								TSPNSEADD[7:0]							
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b8	Reserved	-	Read as "0", write as "0"	R/W											
b7~b0	TSPNSEADD	subsecond increment value	This register contains the 8-bit value used when the system subsecond register is incremented In coarse calibration mode, the value of this register is added to the system time every clock cycle; in fine calibration mode, the value of this register is added to the system time only when the 32-bit accumulator overflows	R/W											

39.6.2.5 ETH_PTP Timestamp System Seconds Register (ETH_PTP_TMSSECR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TSPSYSSEC[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TSPSYSSEC[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b0	TSPSYSSEC	system time in seconds	This bit represents the second time of the system time	R											

39.6.2.6 ETH_PTP Timestamp System Subsecond Register (ETH_PTP_TMSNSER)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	TSPSYSNSEC[30:16]														

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

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	Read as "0", write as "0"	R/W
b30~b0	TSPSYSNSEC	System subsecond time	This bit represents the subsecond time of the system time	R

39.6.2.7 ETH_PTP Timestamp Update Seconds Register (ETH_PTP_TMUSECR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TSPUPSEC [31:16]															

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

Bit	Marking	Place name	Function	Read and write
b31~b0	TSPUSEC	System seconds update value	This bit indicates the second time to be updated into the system time	R/W

39.6.2.8 ETH_PTP timestamp update sub-second register (ETH_PTP_TMUNSER)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TSPUPNS	TSPUPNSEC [30:16]														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TSPUPNSEC[15:0]														
Bit	Marking	Place name	Function				Read and write								
b31	TSPUPNS	update method	Time update sign: 0: Subtract the value of the update register from the value of the system time register 1: Add the value of the update register to the value of the system time register				R/W								
b30~b0	TSPUPNSE	System sub-second update value	This bit indicates the subsecond time to be updated into the system time				R/W								

39.6.2.9 ETH_PTP Timestamp Target Seconds Register 0 (ETH_PTP_TMTSECRO)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TSPTAGSEC0 [31:16]														
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TSPTAGSEC0[15:0]														
Bit	Marking	Place name	Function				Read and write								
b31~b0	TSPTAGSEC0	target second time 0	When the system time exceeds or equals the set value of this register and target subsecond register 0, an interrupt event or PPS event occurs				R/W								

39.6.2.10 ETH_PTP Timestamp Target Subsecond Register 0 (ETH_PTP_TMTNSERO)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	TSPTAGNSEC0 [30:16]														

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TSPTAGNSEC0[15:0]															

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	Read as "0", write as "0"	R/W
b30~b0	TSPTAGNSEC0	target subsecond time 0	When the system time exceeds or equals the set value of this register and target second register 0, an interrupt event or PPS event occurs	R/W

39.6.2.11 ETH_PTP Timestamp Target Seconds Register 1 (ETH_PTP_TMTSECR1)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TSPTAGSEC1 [31:16]															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TSPTAGSEC1[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b0	TSPTAGSEC1	target second time 1	When the system time exceeds or equals the set value of this register and the target subsecond register 1, an interrupt event or PPS event occurs	R/W

39.6.2.12 ETH_PTP Timestamp Target Subsecond Register 1 (ETH_PTP_TMTNSER1)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	TSPTAGNSEC1 [30:16]														

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TSPTAGNSEC1[15:0]															

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	Read as "0", write as "0"	R/W
b30~b0	TSPTAGNSEC1	Target subsecond time 1	When the system time exceeds or equals the set value of this register and the target second register 1, an interrupt or PPS event occurs	R/W

39.6.2.13 ETH_PTP PPS Output Control Register (ETH_PTP_PPSCLR)

Reset value: 0x00000000h

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
-															
Bit	Marking	Place name	Function	Read and write											
b31~b15	Reserved	-	Read as "0", write as "0"	R/W											
b14~b13	TT1SEL	Target Time 1 Feature Selection	00: Destination register 1 is only used for interrupt output events 01: meaningless, please do not set 10: Target register 1 is used for interrupt output events and PPS1 single output events 11: Destination register 1 is only used for PPS1 single output events	R/W											
b12~b11	Reserved	-	Read as "0", write as "0"	R/W											
b10~b8	PPSFRE1	PPS1 output frequency	001: When the system time is equal to the target second register 1, a pulse is generated, and no more pulses are generated after that Please do not set other values <i>Note: This bit is 0 when read</i>	R/W											
b7	Reserved	-	Read as "0", write as "0"	R/W											
b6~b5	TT0SEL	Target time 0 function selection	00: Destination register 0 is only used for interrupt output events 01: meaningless, please do not set 10: Target register 0 is used for interrupt output events and PPS0 single output events 11: Destination register 0 is only used for PPS0 single output events	R/W											
b4	PPSOMD	PPS output mode	0: Continuous output mode 1: Single output mode <i>Note: PPS0 supports the above two modes, PPS1 only supports single output mode</i>	R/W											
b3~b0	PPSFRE0	PPS0 output frequency	In continuous output mode: When the system time has a second carry (sub-second time overflow), PPS0 produces different forms of output according to the following settings: 0000: Continuously output pulses with an interval of 1Hz 0001: Continuously output a clock with a frequency of 2Hz 0010: Continuously output a clock with a frequency of 4Hz 0011: Continuous output clock with a frequency of 8Hz 0100: Continuously output a clock with a frequency of 16Hz 1111: continuous output clock with a frequency of 32768Hz In single output mode: 0001: When the system time is equal to the target second register 0, a pulse is generated, and no pulse is generated after that Please do not set other values <i>Note: 1) In continuous output mode, the TSPSSR bit of the timestamp control register must be set to 0 2) In single output mode, this bit is 0 when read</i>	R/W											

39.6.3 ETH_DMA register

39.6.3.1 ETH_DMA Bus Mode Register (ETH_DMA_BUSMODR)

Reset value: 0x00020101h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16				
Reserved		TXPR	MBST	AAL	M8PBL	SPBL	RPBL[5:0]				FBST								
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0				
<hr/>																			
PRAT[1:0]	TPBL[5:0]				DSEN	DSL[4:0]				DMAA	SWR								
<hr/>																			
Bit	Marking	Place name	Function								Read and write								
b31~b28	Reserved	-	Read as "0", write as "0"								R/W								
b27	TXPR	delivery priority	0: RxDMA transactions have higher priority than TxDMA transactions 1: TxDMA transactions have higher priority than RxDMA transactions								R/W								
b26	MBST	mixed burst transfer	When this bit is 1 and the FBST bit is 0, the AHB Master port will start all burst transfers with a length greater than 16 with INCR, and start a burst transfer with a length equal to or less than 16 with SINGLE and INCRx (x=4/8/16)								R/W								
b25	AAL	address alignment	When this position is 1: 1) If FBST=1, the AHB interface will generate all bursts aligned with the LS bit of the starting address 2) If FBST=0, the first burst (the starting address of accessing the data buffer) is not aligned, but the subsequent bursts are aligned with the address								R/W								
b24	M8PBL	8x PBL	When this bit is set to 1, the programmed PBL value (RPBL bit and TPBL bit) is multiplied by a factor of 8, so the DMA transfers data at a maximum of 8, 16, 32, 64, 128 and 256 beats depending on the PBL value								R/W								
b23	SPBL	Independent PBL control	0: The TPBL setting of this register is used for TxDMA transactions and RxDMA transactions 1: The TPBL setting of this register is only used for TxDMA transactions, and the RPBL setting is used for RxDMA transactions								R/W								
b22~b17	RPBL	Rx programmable burst length	It has the same meaning as the TPBL bit of this register and is only used for RxDMA transactions <i>Note: This bit is only useful when the SPBL bit of this register is active</i>								R/W								
b16	FBST	fixed burst transfer	0: DMA Master interface only uses SINGLE and INCR access types 1: DMA Master interface uses SINGLE and INCR4, INCR8, INCR16 access types								R/W								
b15~b14	PRAT	priority ratio	Rx and Tx preemption priority ratio of BUS 00: 1: 1 11: 0000h 10: 3: 1 11: 4: 1 <i>Note 1: This bit is only valid when the DMAA bit of this register is 0 Note 2: Tx: Rx or Rx: The selection of Tx is determined by the TXPR bit of this register</i>								R/W								
b13~b8	TPBL	Programmable burst length	This bit indicates the maximum number of ticks to transfer in one TxDMA transaction or RxDMA transaction, which is the maximum value used in a single block read/write operation. Every time the DMA starts a burst transfer on the host bus, it always tries to burst as specified in the PBL. PBLs are allowed to be programmed with values 1, 2, 4, 8, 16, and 32, any other value yields undefined behavior <i>Note: The PBL value has the following limitations: 1) The possible maximum number of beats (PBL) is limited by the size of Tx FIFO and Rx FIFO</i>								R/W								

2) Please do not program out of range PBL value

			Enhanced Descriptor Format Enable 0: regular descriptor 1: Enhanced Descriptor <i>Note: The enhanced descriptor must be used when the PTP function is active (ETH_PTP_TSPCLR.TSPEN=1) or the checksum offloading function is active (ETH_MAC_CONFIGR.IPCO=1)</i>	R/W
b7	DSEN	Enhanced Descriptor	This bit specifies the number of words to skip between two unlinked descriptors, where the address jumps from the end of the current descriptor to the beginning of the next descriptor. When the DSL value is equal to zero, in ring mode, the DMA treats the descriptor table as contiguous	R/W
b6~b2	DSL	Descriptor skip length	0: Circular priority, the arbitration method is determined by the PRAT bit and TXPR bit of this register 1: Fixed priority, the priority mode is determined by the TXPR bit of this register	R/W
b1	DMAA	DMA arbitration	When this bit is set to 1, the DMA controller resets all internal registers and logic of the MAC subsystem. This bit is automatically cleared after all core clock domains complete a reset operation	R/W
b0	SWR	Software reset		R/W

39.6.3.2 ETH_DMA Operation Mode Register (ETH_DMA_OPRMODR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16		
Reserevd				DTCOE	RSF	DFRF	-	-	TSF	FTF	-	-	-	-	TTC[2]		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0		
TTC[1:0]	STT	Reserevd				FEF	FUF	DGF	RTC[1:0]	OSF	STR	-					
<hr/>																	
Bit	Marking		Place name		Function								Read and write				
b31~b27	Reserved		-		Read as "0", write as "0"								R/W				
b26	DTCOE		Disable discarding of TCP/IP checksum error frames		0: If the FEF bit of this register is set, all error frames are discarded 1: If the received frame contains only errors detected by the receive checksum offload engine, the kernel will not discard the frame <i>Note: Such frames do not have any errors (including CRC errors) in the Ethernet frame received by the MAC, but only in the encapsulated payload</i>								R/W				
b25	RSF		receive store and forward		0: The Rx FIFO is read only when the frame in the Rx FIFO reaches the value specified by the RTC bit of this register 1: After the Rx FIFO writes a complete frame, the read action is performed, and the RTC bit is ignored.								R/W				
b24	DFRF		Disable refresh of received frames		When this bit is set to 1, RxDMA will not flush any frames due to unavailability of receive descriptors/buffers, see section "Receive Process Suspended"								R/W				
b23~b22	Reserved		-		Read as "0", write as "0"								R/W				
b21	TSF		send store-and-forward		0: When the frame in the Tx FIFO reaches the value specified by the TTC bit of the register, the read action of the Tx FIFO is performed, that is, the MAC transmission is started. 1: There is a complete frame in the Tx FIFO, then the MAC transmission will start, and the TTC bit will be ignored <i>Note: This bit can only be changed when the transmission has stopped</i>								R/W				
b20	FTF		Flush Tx FIFO		When this bit is set to 1, the Tx FIFO is reset to its default value, at which point all data in the Tx FIFO is lost/flushed. This bit is automatically cleared at the end of the refresh operation <i>Note: The operating mode register must not be written to until this bit is cleared</i>								R/W				
b19~b17	Reserved		-		Read as "0", write as "0"								R/W				
b16~b14	TTC		send threshold		This bit is used to control the threshold level of the Tx FIFO. DMA transmission is initiated when the frame size in the Tx FIFO is greater than the threshold. Additionally, complete frames with lengths less than the threshold are automatically transmitted 000: 64 0000h 0000h 011: 256 100: 40 101: 32 110: 24 111: 16 <i>Note: This bit is only used when the TSF bit is 0</i>								R/W				
b13	STT		send start stop		0: The sending process enters the stop state after completing the task of sending the current frame, and saves the next descriptor position in the sending list, which will become the current position after restarting the sending								R/W				

<p>1: Start the transmission process, and the DMA checks the current position in the transmission list, that is, the address set by the transmit descriptor address list register (ETH_DMA_TXDLADR), or the reserved position when the transmission was stopped last time. Try to obtain the descriptor to find the frame to be sent If the current descriptor does not belong to TxDMA, the transmission process will enter a pending state; if the command is issued before the ETH_DMA_TXDLADR register is set, the TxDMA behavior is unpredictable</p> <p><i>Note: The start send command is only valid when the transmission has stopped; the stop send command is valid only when the current frame sending process is over or the sending process is in a pending state</i></p>					
b12~b8	Reserved	-	Read as "0", write as "0"	R/W	
b7	FEF	Forward error frames	<p>0: Rx FIFO discards frames with error status (CRC error, collision error, jumbo frame, watchdog timeout, overflow)</p> <p>If the start byte pointer of a frame has been transmitted to the read controller side (in threshold mode), the frame will not be discarded; if the start byte of the frame is not transmitted on the bus, the Rx FIFO will discard this error frame</p> <p>1: All frames except short frame error frames are forwarded</p>	R/W	
b6	FUF	Forward too small good frame	<p>0: Rx FIFO discards all frames less than 64 bytes unless such frames have been transmitted due to lower receive threshold (e.g. RTC=01)</p> <p>1: Rx FIFO forwards undersized frames containing PAD bytes and CRC (frames with no errors but less than 64 bytes in length)</p>	R/W	
b5	DGF	drop jumbo frames	<p>0: Rx FIFO does not drop jumbo frames</p> <p>1: Rx FIFO discards jumbo frames</p>	R/W	
b4~b3	RTC	acceptance threshold	<p>00: 64</p> <p>01: 32</p> <p>10: 96</p> <p>11: 128</p>	R/W	
<p><i>Note: This bit is only used when the RSF bit is 0</i></p>					
b2	OSF	process second frame	<p>When this bit is set to 1, it instructs the DMA to process the second transmit data frame, even if the status of the first frame has not yet been obtained</p>	R/W	
b1	STR	receive start stop	<p>0: RxDMA will stop operation after transmitting the current frame and save the next descriptor position in the receive list, which will become the current position after the receive process restarts</p> <p>1: Start the receiving process, DMA tries to obtain the descriptor from the current position in the receiving list, that is, the address set by the Receive Descriptor Address List Register (ETH_DMA_RXDLADR), or the reserved position when the receiving was stopped last time, and process incoming frame</p> <p>If the RxDMA does not occupy any descriptors, the receive process goes into a pending state; if the command is issued before the ETH_DMA_RXDLADR register is set, the RxDMA behavior is unpredictable</p>	R/W	
b0	Reserved	-	Read as "0", write as "0"	R/W	

39.6.3.3 ETH_DMA Action Status Register (ETH_DMA_DMASTR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	PTPS	PMTS	MMCS	-	EBUS[2:0]		TSTS[2:0]		RSTS[2:0]		NIS			

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
AIS	ERS	FBS	-	-	ETS	RWS	RSS	RUS	RIS	UNS	OVS	TJS	TUS	TSS	TIS

Bit	Marking	Place name	Function	Read and write
b31~b30	Reserved	-	Read as "0", write as "0"	R/W
b29	PTPS	PTP event status	0: PTP module generates no timestamp event 1: The PTP module has a timestamp event occurred <i>Note: This bit is automatically cleared when the event status that sets this bit is cleared</i>	R
b28	PMTS	PMT event status	0: No wake-up event for the PMT module 1: PMT module has a wake-up event <i>Note: This bit is automatically cleared when the event status that sets this bit is cleared</i>	R
b27	MMCS	MMC event status	0: No frame statistics event occurs in the MMC module 1: A frame statistics event occurs in the MMC module <i>Note: This bit is automatically cleared when the event status that sets this bit is cleared</i>	R
b26	Reserved	-	Read as "0", write as "0"	R/W
b25~b23	EBUS	Error bit status	This bit indicates the type of error that caused the bus error (error response on the AHB interface), no interrupt will be generated 000: An error occurred while writing data by RxDMA 011: Error occurred while TxDMA read data 100: RxDMA error writing descriptor 101: TxDMA error writing descriptor 110: RxDMA error occurred while reading descriptor 111: Error occurred while TxDMA read descriptor Please do not write other values <i>Note: This bit is only valid when the FBS bit of this register is set</i>	R
b22~b20	TSTS	send status	This bit indicates the status of the TxDMA FSM, no interrupt will be generated 000: stop, issue reset or stop sending command 001: Running, getting send transfer descriptor 010: Running, waiting state 011: Running, reading data in host memory buffer and enqueueing it in transmit buffer (Tx FIFO) 100: Timestamp write 101: Reserved 110: pending, transmit descriptor unavailable or transmit buffer underflow 111: Running, closing send descriptor	R
b19~b17	RSTS	receive status	This bit indicates the status of the RxDMA FSM and will not generate an interrupt. 000: Stop, issue a reset or stop receiving commands 001: Running, getting receive transfer descriptor 010: Reserved 011: Running, waiting to receive packets 100: pending, receive descriptor not available 101: Running, closing receive descriptor 110: Timestamp write 111: In operation, transfer the data of the received packet from the receive buffer to the host memory	R
b16	NIS	General interrupt status summary	This bit is set to 1 when any of the following conditions are met: 1) TIS=1 && ETH_DMA_INTENAR.TIE=1 2) TUS=1 && ETH_DMA_INTENAR.TUE=1 3) RIS=1 && ETH_DMA_INTENAR.RIE=1 4) ERS=1 && ETH_DMA_INTENAR.ERE=1 <i>Note 1: Only unmasked interrupt enable bits affect this general interrupt status summary bit</i> <i>Note 2: Whenever the corresponding bit that caused</i>	R/W

			<i>this bit to be set is cleared, this bit must also be cleared (by writing a 1 to this bit)</i>	
b15	AIS	Abort Status Summary	<p>This bit is set to 1 when any of the following conditions are met:</p> <ol style="list-style-type: none"> 1) TSS=1 && ETH_DMA_INTENAR.TSE=1 2) TJS=1 && ETH_DMA_INTENAR.TJE=1 3) OVS=1 && ETH_DMA_INTENAR.OVE=1 4) UNS=1 && ETH_DMA_INTENAR.UNE=1 5) RUS=1 && ETH_DMA_INTENAR.RUE=1 6) RSS=1 && ETH_DMA_INTENAR.RSE=1 7) RWS=1 && ETH_DMA_INTENAR.RWE=1 8) ETS=1 && ETH_DMA_INTENAR.ETE=1 9) FBS=1 && ETH_DMA_INTENAR.FBE=1 <p><i>Note 1: Only unmasked interrupt enable bits affect this abort status summary bit</i></p> <p><i>Note 2: Whenever the corresponding bit that caused this bit to be set is cleared, this bit must also be cleared (by writing a 1 to this bit)</i></p>	R/W
b14	ERS	Early reception status	<p>This bit indicates that the DMA has filled the first data buffer of the packet</p> <p><i>Note: This bit can be cleared by writing a 1 to this bit, or it can be cleared automatically after the RIS of this register is set</i></p>	R/W
b13	FBS	Fatal bus error status	<p>This bit indicates that a bus error has occurred. For the specific error type, see the EBUS bit of this register.</p> <p>When this bit is set to 1, the corresponding DMA engine will prohibit all bus accesses.</p> <p><i>Note: This bit can be cleared by writing a 1 to this bit</i></p>	R/W
b12~b11	Reserved	-	Read as "0", write as "0"	R/W
b10	ETS	Advance delivery status	<p>This bit indicates that the frame to be transmitted has been completely transferred to the Tx FIFO</p> <p><i>Note: This bit can be cleared by writing a 1 to this bit</i></p>	R/W
b9	RWS	Receive watchdog status	<p>This bit is set to 1 when the length of the received frame is greater than 2048 bytes</p> <p><i>Note: This bit can be cleared by writing a 1 to this bit</i></p>	R/W
b8	RSS	receive stop status	<p>This bit is set to 1 when the receiving process enters the stop state</p> <p><i>Note: This bit can be cleared by writing a 1 to this bit</i></p>	R/W
b7	RUS	Receive buffer unavailable status	<p>This bit indicates that the next descriptor in the receive list is owned by the host and cannot be acquired by DMA, and the receive process enters the pending state</p> <p>To resume processing a receive descriptor, the host should change the ownership of the descriptor and then issue a receive poll request command</p> <p>If the receive polling request command is not issued, the receive process resumes when the next recognized input frame is received</p> <p>This bit is set only if the last receive descriptor was owned by the DMA</p> <p><i>Note: This bit can be cleared by writing a 1 to this bit</i></p>	R/W
b6	RIS	receive status	<p>This bit indicates that frame reception is complete, specific frame status information has been published in the descriptor, and reception remains running</p> <p><i>Note: This bit can be cleared by writing a 1 to this bit</i></p>	R/W
b5	UNS	Send underflow status	<p>This bit indicates that during frame transmission, the transmit buffer underflow occurs, the transmission will enter the pending state, and the underflow error TDESO[1] flag bit in the descriptor is set to 1</p> <p><i>Note: This bit can be cleared by writing a 1 to this bit</i></p>	R/W
b4	OVS	Receive overflow status	<p>This bit indicates that during frame reception, the receive buffer overflowed, if part of the frame has been transferred to the application, the overflow error RDESO[11] bit in the descriptor is set to 1</p> <p><i>Note: This bit can be cleared by writing a 1 to this bit</i></p>	R/W
b3	TJS	Send Jabber timeout status	<p>This bit indicates that the transmit Jabber timer has expired, which means that the transmitter is over-active, the transmit process is aborted and it is put in a stopped state, and the transmit Jabber timeout TDESO[14] flag bit is set in the descriptor</p> <p><i>Note: This bit can be cleared by writing a 1 to this bit</i></p>	R/W
b2	TUS	Send buffer unavailable status	<p>This bit indicates that the next descriptor in the transmit list is owned by the host and cannot be acquired by the DMA, and the transmit will enter a pending state</p> <p>To resume processing transmit descriptors, the host should change the ownership of the descriptor and issue a transmit poll request command</p> <p><i>Note: This bit can be cleared by writing a 1 to this bit</i></p>	R/W

b1	TSS	send stop status	This bit is set to 1 when the sending process enters the stop state <i>Note: This bit can be cleared by writing a 1 to this bit</i>	R/W
b0	TIS	send interrupt status	This bit indicates that frame transmission is complete and specific frame status information has been published in the descriptor <i>Note: This bit can be cleared by writing a 1 to this bit</i>	R/W

39.6.3.4 ETH_DMA Interrupt Enable Register (ETH_DMA_INTENAR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved															NIE
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
AIE	ERE	FBE	-	-	ETE	RWE	RSE	RUE	RIE	UNE	OVE	TJE	TUE	TSE	TIE
<hr/>															
Bit	Marking	Place name	Function												Read and write
b31~b17	Reserved	-	Read as "0", write as "0"												R/W
b16	NIE	General interrupt enable summary	0: Normal event interrupt is invalid 1: Normal event interrupt is valid <i>Note: Common events refer to the events indicated by the B0, B2, B6, B14 bits of the action status register ETH_DMA_DMASTR</i>												R/W
b15	AIE	Abnormal Interrupt Enable Summary	0: Abnormal event interrupt is invalid 1: Exception event interrupt is valid <i>Note: The abnormal event refers to the event indicated by the B1, B3, B4, B5, B7, B8, B9, B10, B14 bits of the action status register ETH_DMA_DMASTR</i>												R/W
b14	ERE	Early receive interrupt enable	0: Early reception interrupt is invalid 1: Early receive interrupt enable <i>Note: This bit is valid after the NIE bit is set</i>												R/W
b13	FBE	Fatal bus error interrupt enable	0: Fatal bus error interrupt invalid 1: Fatal bus error interrupt active <i>Note: This bit is valid after the AIE bit is set</i>												R/W
b12~b11	Reserved	-	Read as "0", write as "0"												R/W
b10	ETE	Early Transmit Interrupt Enable	0: Early send interrupt is invalid 1: Early send interrupt is valid <i>Note: This bit is valid after the AIE bit is set</i>												R/W
b9	RWE	Receive watchdog overflow interrupt enable	0: Receive watchdog overflow interrupt is invalid 1: Receive watchdog overflow interrupt valid <i>Note: This bit is valid after the AIE bit is set</i>												R/W
b8	RSE	Receive Stop Interrupt Enable	0: Receive stop interrupt is invalid 1: Receive stop interrupt valid <i>Note: This bit is valid after the AIE bit is set</i>												R/W
b7	RUE	Receive buffer unavailable interrupt enable	0: Receive buffer unavailable interrupt invalid 1: Receive buffer unavailable interrupt valid <i>Note: This bit is valid after the AIE bit is set</i>												R/W
b6	RIE	receive interrupt enable	0: Receive interrupt is invalid 1: Receive interrupt valid <i>Note: This bit is valid after the NIE bit is set</i>												R/W
b5	UNE	Transmit underflow interrupt enable	0: Send underflow interrupt is invalid 1: Transmit underflow interrupt is valid <i>Note: This bit is valid after the AIE bit is set</i>												R/W
b4	OVE	Receive overflow interrupt enable	0: Receive overflow interrupt is invalid 1: Receive overflow interrupt valid <i>Note: This bit is valid after the AIE bit is set</i>												R/W
b3	TJE	Send Jabber Timeout Interrupt Enable	0: Send Jabber timeout interrupt invalid 1: Send Jabber timeout interrupt valid <i>Note: This bit is valid after the AIE bit is set</i>												R/W
b2	TUE	Transmit buffer unavailable interrupt enable	0: Send buffer unavailable interrupt invalid 1: Send buffer unavailable interrupt valid <i>Note: This bit is valid after the NIE bit is set</i>												R/W
b1	TSE	Transmit Stop Interrupt Enable	0: Send stop interrupt is invalid 1: Send stop interrupt valid <i>Note: This bit is valid after the AIE bit is set</i>												R/W
b0	TIE	Transmit interrupt enable	0: Send interrupt is invalid 1: Send interrupt is valid <i>Note: This bit is valid after the NIE bit is set</i>												R/W

39.6.3.5 ETH_DMA Frame Loss Statistics Register (ETH_DMA_RFRCNTR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16			
-	-	-	OVFOVF	OVFCNT[10:0]														UNAOVF
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0			
UNACNT[15:0]																		
<hr/>																		
Bit	Marking	Place name	Function										Read and write					
b31~b29	Reserved	-	Read as "0", write as "0"										R/W					
b28	OVFOVF	overflow lost frame count overflow	When the OVFCNT bit of this register counts the lost frame counter overflow, this bit is set to 1 <i>Note: This bit is automatically cleared after reading the OVFCNT bit of this register</i>										R					
b27~b17	OVFCNT	Overflow Lost Frame Statistics	This bit indicates the number of frames lost due to Rx FIFO overflow condition and frame length too short (less than 64 bytes good frame) <i>Note: This bit is automatically cleared after being read</i>										R					
b16	UNAOVF	Buffer unavailable lost frame count overflow	When the UNACNT bit of this register counts the lost frame counter overflow, this bit is set to 1 <i>Note: This bit is automatically cleared after reading the UNACNT bit of this register</i>										R					
b15~b0	UNACNT	Buffer Unavailable Lost Frame Statistics	This bit indicates the number of frames lost due to buffer unavailability (no receive descriptors available) <i>Note: This bit is automatically cleared after being read</i>										R					

39.6.3.6 ETH_DMA Watchdog Timer Register (ETH_DMA_REVWDTR)

Reset value: 0x00000000h

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								RIWT[7:0]							

Bit	Marking	Place name	Function	Read and write
b31~b8	Reserved	-	Read as "0", write as "0"	R/W
b7~b0	RIWT	Receive watchdog count	This bit indicates the time value after the number of system clock cycles is multiplied by 256, which is the set time value of the watchdog timer The watchdog timer will be triggered by the programmed value after the RxDMA finishes the frame transfer. At this time, the RIS bit of the action status register ETH_DMA_DMASTR is not set to 1 because the RDES1.RIS bit in the corresponding descriptor is set to 1 until When the watchdog timer expires, the RIS bit is set and the watchdog timer is stopped. When the RIS bit of the action status register ETH_DMA_DMASTR is set to high level, the watchdog timer is reset	R/W

39.6.3.7 ETH_DMA Transmit Poll Request Register (ETH_DMA_TXPOLLR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TXPOLL[31:16]															

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

Bit	Marking	Place name	Function	Read and write
b31~b0	TXPOLL	Send poll request to initiate	<p>When any value is written to this bit, the DMA reads the current descriptor pointed to by the ETH_DMA_CHTXDER register, and if the descriptor is not available (owned by the host), the TxDMA returns to the pending state and sets the TUS bit in the ETH_DMA_DMASTR register bit; if the descriptor is available, the transmission will continue. Use this register to instruct TxDMA to poll the transmit descriptor list to check if the current descriptor is owned by the DMA. If the TxDMA is in suspend mode, issue a transmit polling request command to wake it up; if an underflow error occurs in the transmit frame or the descriptor owned by the TxDMA is not available, the TxDMA enters suspend mode.</p> <p>The user can issue this command at any time and TxDMA will reset the host memory as soon as the command starts to re-fetch the current descriptor of the host memory.</p> <p><i>Note: This bit always reads zero when read</i></p>	R/W

39.6.3.8 ETH_DMA Receive Polling Request Register (ETH_DMA_RXPOLLR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
RXPOLL[31:16]															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RXPOLL[51:0]															

Bit	Marking	Place name	Function	Read and write
b31~b0	RXPOLL	Receive polling request initiated	<p>When any value is written to this bit, the DMA will read the current descriptor pointed to by the ETH_DMA_CHRXDER register, and if the descriptor is not available (owned by the host), the RxDMA will return to the pending state without setting the ETH_DMA_DMASTR register RUS bit asserted; if the descriptor is available, the RxDMA will return to the active state</p> <p>Use this register to instruct the RxDMA to poll the list of receive descriptors, check for new descriptors, used to wake the RxDMA from the pending state, it will enter the pending state only if the descriptor owned by the RxDMA is not available</p> <p><i>Note: This bit always reads zero when read</i></p>	R/W

39.6.3.9 ETH_DMA Transmit Descriptor List Address Register (ETH_DMA_TXDLADR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TXDLAD[31:16]															

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

Bit	Marking	Place name	Function	Read and write
b31~b0	TXDLAD	Send descriptor first address	<p>This bit indicates the address of the first descriptor in the transmit descriptor list</p> <p><i>Note: Lowest 2 bits always read zero when read</i></p>	R/W

39.6.3.10 ETH_DMA Receive Descriptor List Address Register (ETH_DMA_RXDLADR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
RXDLAD[31:16]															

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

Bit	Marking	Place name	Function	Read and write
b31~b0	RXDLAD	Receive descriptor first address	This bit indicates the address of the first descriptor in the receive descriptor list <i>Note: Lowest 2 bits always read zero when read</i>	R/W

39.6.3.11 ETH_DMA current host transmit descriptor register (ETH_DMA_CHTXDER)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CHTXDE[31:16]															

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

Bit	Marking	Place name	Function	Read and write
b31~b0	CHTXDE	send descriptor address pointer	This bit points to the starting address of the current transmit descriptor read by the DMA	R

39.6.3.12 ETH_DMA current host receive descriptor register (ETH_DMA_CHRXDER)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CHRXDE[31:16]															

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

Bit	Marking	Place name	Function	Read and write
b31~b0	CHRXDE	Receive descriptor address pointer	This bit points to the starting address of the current receive descriptor read by the DMA	R

39.6.3.13 ETH_DMA current host transmit buffer register (ETH_DMA_CHTXBFR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CHTXBF[31:16]															

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

Bit	Marking	Place name	Function	Read and write
b31~b0	CHTXBF	send buffer address pointer	This bit points to the current transmit buffer address read by DMA	R

39.6.3.14 ETH_DMA current host receive buffer register (ETH_DMA_CHRXBFR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CHRXBFR[31:16]															

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

Bit	Marking	Place name	Function	Read and write
b31~b0	CHRXBFR	Receive buffer address pointer	This bit points to the current receive buffer address read by DMA	R

39.6.4 ETH_MMCTLR Register

39.6.4.1 ETH_MMCTLR Register (ETH_MMCTLR)

Reset value: 0x00000000h

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										MCPSEL	MCPSET	MCF	ROR	COS	CRST
Bit	Marking		Place name		Function										Read and write
b31~b6	Reserved		-		Read as "0", write as "0"										R/W
b5	MCPSEL		Counter preset value selection		0: Preset the value of all MMC statistics registers to "preset half value" (7FF0H) 1: Preset the value of all MMC statistics registers to "preset full value" (FFF0H) <i>Note: The default full value is 0000H-0010H=FFF0H The default half value is 8000H-0010H=7FF0H</i>										R/W
b4	MCPSET		Counter preset value setting		After this bit is set to 1, all MMC statistics registers are preset to full value or half value according to the setting of the MCPSEL bit of this register <i>Note: This bit is automatically cleared 1 clock after being set</i>										R/W
b3	MCF		Counter freezes		After this bit is set to 1, the values of all MMC statistics counters are frozen and no longer change due to received or transmitted frames										R/W
b2	ROR		Counter read reset		When this bit is set, all MMC statistics counters are reset to zero after being read <i>Note 1: This bit is valid regardless of whether the MCF bit of this register is valid or not. Note 2: When reading the lower 8 bits of the statistical counter, it is regarded as a read action</i>										R/W
b1	COS		Stop when the counter rolls over		0: After the counter reaches FFFFH, it returns to zero and counts again 1: After the counter reaches FFFFH, it will stop and no longer count.										R/W
b0	CRST		Counter reset		When set to 1, the values of all MMC statistics counters are reset <i>Note: This bit is automatically cleared 1 clock after being set</i>										R/W

39.6.4.2 ETH_MMC Rx Statistics Status Register (ETH_MMC_REVSTSR)

Reset value: 0x00000000h

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															

Bit	Marking	Place name	Function	Read and write
b31~b20	Reserved	-	Read as "0", write as "0"	R/W
b19	RXOEIS	Out of Range Error Frame Statistics Status	0: The number of out-of-range error frames received has not reached half or full value 1: The number of out-of-range error frames received reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>	R
b18	RXLEIS	Length Error Frame Statistics Status	0: The number of received frames with length errors has not reached half or full value 1: The number of received frames with length errors reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>	R
b17	RXUGIS	Unicast good frame statistics status	0: The number of unicast good frames received has not reached half or full value 1: The number of unicast good frames received reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>	R
b16~b8	Reserved	-	Read as "0", write as "0"	R/W
b7	RXREIS	Low Frame Error Frame Statistics Status	0: The number of received short frame error frames has not reached half or full value 1: The number of received short frame error frames reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>	R
b6	RXAEIS	Alignment error frame statistics status	0: The number of received alignment error frames has not reached half or full value 1: The number of received frames with alignment errors reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>	R
b5	RXCEIS	CRC error frame statistics status	0: The number of received CRC error frames has not reached half or full value 1: The number of received CRC error frames reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>	R
b4	RXMGIS	Multicast good frame statistics status	0: The number of good multicast frames received is less than half or full 1: The number of multicast good frames received reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>	R
b3	RXBGIS	Broadcast good frame statistics status	0: The number of broadcast good frames received has not reached half value or full value 1: The number of received broadcast frames reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>	R
b2~b0	Reserved	-	Read as "0", write as "0"	R/W

39.6.4.3 ETH_MMC Tx Statistics Status Register (ETH_MMC_TRSSTSR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved				TXEDEIS	TXUGIS	-	TXCAEIS	TXCEIS	TXLCEIS	TXDEEIS					
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				TXMGIS	TXBGIS	-	-								
Bit	Marking	Place name	Function				Read and write								
b31~b23	Reserved	-	Read as "0", write as "0"				R/W								
b22	TXEDEIS	Excessive Delay Error Frame Statistics Status	0: The number of excessively delayed error frames transmitted has not reached half or full value 1: The number of excessively delayed error frames transmitted reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>				R								
b21	TXUGIS	Unicast good frame statistics status	0: The number of unicast good frames transmitted has not reached half or full value 1: The number of unicast good frames transmitted reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>				R								
b20	Reserved	-	Read as "0", write as "0"				R/W								
b19	TXCAEIS	Carrier Error Frame Statistics Status	0: The number of transmitted carrier error frames has not reached half or full value 1: The number of transmitted carrier error frames reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>				R								
b18	TXCEIS	Excessive collision error frame statistics status	0: The number of excessive collision error frames transmitted has not reached half or full value 1: The number of excessive collision error frames transmitted reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>				R								
b17	TXLCEIS	Delay Collision Error Frame Statistics Status	0: The number of delayed collision error frames transmitted has not reached half or full value 1: The number of delayed collision error frames transmitted reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>				R								
b16	TXDEEIS	Delay Error Frame Statistics Status	0: The number of delayed error frames transmitted has not reached half or full value 1: The number of delayed error frames transmitted reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>				R								
b15~b4	Reserved	-	Read as "0", write as "0"				R/W								
b3	TXMGIS	Multicast good frame statistics status	0: The number of good multicast frames transmitted is less than half or full 1: The number of transmitted multicast frames reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>				R								
b2	TXBGIS	Broadcast good frame statistics status	0: The number of transmitted broadcast frames has not reached half or full value 1: The number of transmitted broadcast frames reaches half or full value <i>Note: This bit is automatically cleared after reading the corresponding MMC statistics register</i>				R								
b1~b0	Reserved	-	Read as "0", write as "0"				R/W								

39.6.4.4 ETH_MMC Rx Interrupt Control Register (ETH_MMC_RITCLR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved												RXOE IM	RXLE IM	RXUG IM	-
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved						RXRE IM	RXAE IM	RXCE IM	RXMG IM	RXBG IM	-	-	-	-	-
Bit	Marking	Place name	Function	Read and write											
b31~b20	Reserved	-	Read as "0", write as "0"	R/W											
b19	RXOEIM	Out of Range Error Frame Interrupt Mask	0: When RXOEIS is set, the MMC statistics interrupt of out-of-range error frames is not masked 1: When RXOEIS is set, MMC statistics interrupt mask for out-of-range error frames	R/W											
b18	RXLEIM	length error frame interrupt mask	0: When RXLEIS is set, the length error frame MMC statistics interrupt is not masked 1: When RXLEIS is set, the length error frame MMC statistics interrupt mask	R/W											
b17	RXUGIM	Unicast good frame interrupt mask	0: When RXUGIS is set, the MMC statistics interrupt of unicast good frame is not masked 1: When RXUGIS is set, unicast good frame MMC statistics interrupt mask	R/W											
b16~b8	Reserved	-	Read as "0", write as "0"	R/W											
b7	RXREIM	dwarf frame error frame interrupt mask	0: When RXREIS is set, the MMC statistics interrupt of short frame error frame is not masked 1: When RXREIS is set, the short frame error frame MMC statistics interrupt mask	R/W											
b6	RXAEIM	Alignment error frame interrupt mask	0: When RXAEIS is set, the MMC statistics interrupt of the alignment error frame is not masked 1: When RXAEIS is set, the MMC statistics interrupt mask of the alignment error frame	R/W											
b5	RXCEIM	CRC error frame interrupt mask	0: When RXCEIS is set, CRC error frame MMC statistics interrupt is not masked 1: When RXCEIS is set, CRC error frame MMC statistics interrupt mask	R/W											
b4	RXMGIM	Multicast good frame interrupt mask	0: When RXMGIS is set, the MMC statistics interrupt of multicast good frame is not masked 1: When RXMGIS is set, multicast good frame MMC statistics interrupt mask	R/W											
b3	RXBGIM	Broadcast good frame interrupt mask	0: When RXBGIS is set, the MMC statistics interrupt of the broadcast frame is not masked 1: When RXBGIS is set, broadcast good frame MMC statistics interrupt mask	R/W											
b2~b0	Reserved	-	Read as "0", write as "0"	R/W											

39.6.4.5 ETH_MMC Tx Interrupt Control Register (ETH_MMC_TITCLR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved									TXEDEIM	TXUGIM	-	TXCAEIM	TXCEIM	TXLCEIM	TXDEEIM
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved										TXMGIN	TXBGIN	-	-		
<hr/>															
Bit	Marking	Place name	Function				Read and write								
b31~b23	Reserved	-	Read as "0", write as "0"				R/W								
b22	TXEDEIM	Excessive Delay Error Frame Interrupt Mask	0: When TXEDEIS is set, MMC statistics interrupt of excessive delay error frame is not masked 1: When TXEDEIS is set, excessive delay error frame MMC statistics interrupt mask				R/W								
b21	TXUGIM	Unicast good frame interrupt mask	0: When TXUGIS is set, the MMC statistics interrupt of unicast good frame is not masked 1: When TXUGIS is set, unicast good frame MMC statistics interrupt mask				R/W								
b20	Reserved	-	Read as "0", write as "0"				R/W								
b19	TXCAEIM	Carrier Error Frame Interrupt Mask	0: When TXCAEIS is set, the MMC statistics interrupt of carrier error frame is not masked 1: When TXCAEIS is set, the carrier error frame MMC statistics interrupt mask				R/W								
b18	TXCEIM	Excessive collision error frame interrupt mask	0: When TXCEIS is set, the MMC statistics interrupt of excessive collision error frame is not masked 1: When TXCEIS is set, excessive collision error frame MMC statistics interrupt mask				R/W								
b17	TXLCEIM	Delay Collision Error Frame Interrupt Mask	0: When TXLCEIS is set, delay collision error frame MMC statistics interrupt is not masked 1: When TXLCEIS is set, delay collision error frame MMC statistics interrupt mask				R/W								
b16	TXDEEIM	Delay Error Frame Interrupt Mask	0: When TXDEEIS is set, the MMC statistics interrupt of delayed error frames is not masked 1: When TXDEEIS is set, delay error frame MMC statistics interrupt mask				R/W								
b15~b4	Reserved	-	Read as "0", write as "0"				R/W								
b3	TXMGIN	Multicast good frame interrupt mask	0: When TXMGIS is set, the MMC statistics interrupt of multicast good frame is not masked 1: When TXMGIS is set, multicast good frame MMC statistics interrupt mask				R/W								
b2	TXBGIN	Broadcast good frame interrupt mask	0: When TXBGIS is set, the MMC statistics interrupt of the broadcast frame is not masked 1: When TXBGIS is set, broadcast good frame MMC statistics interrupt mask				R/W								
b1~b0	Reserved	-	Read as "0", write as "0"				R/W								

39.6.4.6 ETH_MMC Rx Unicast Good Frame Statistics Register (ETH_MMC_RXUNGFR)

Reset value: 0x00000000h

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
RXUNGNT[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15~b0	RXUNGNT	Unicast good frames	This bit indicates the number of unicast good frames received	R

39.6.4.7 ETH_MMC Rx Multicast Good Frame Statistics Register (ETH_MMC_RXMUGFR)

Reset value: 0x00000000h

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
RXMUGNT[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15~b0	RXMUGNT	Multicast good frames	This bit indicates the number of multicast good frames received	R

39.6.4.8 ETH_MMC Rx broadcast good frame statistics register (ETH_MMC_RXBRGFGR)

Reset value: 0x00000000h

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
RXBRGCNT[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15~b0	RXBRGCNT	Broadcast good frames	This bit indicates the number of broadcast good frames received	R

39.6.4.9 ETH_MMC RxCRC Error Frame Statistics Register (ETH_MMC_RXCREFR)

Reset value: 0x00000000h

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
RXCRECNT[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15~b0	RXCRECNT	CRC error frames	This bit indicates the number of CRC error frames received	R

39.6.4.10 ETH_MMC Rx Alignment Error Frame Statistics Register (ETH_MMCRXALEFR)

Reset value: 0x00000000h

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
RXALECNT[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read as "0", write as "0"	R/W											
b15~b0	RXALECNT	Alignment error frames	This bit indicates the number of misaligned frames received	R											

39.6.4.11 ETH_MMC Rx Short Frame Error Frame Statistics Register (ETH_MMCRXRUEFR)

Reset value: 0x00000000h

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
RXRUECNT[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read as "0", write as "0"	R/W											
b15~b0	RXRUECNT	Low Frame Error Frames	This bit indicates the number of DFR frames received	R											

39.6.4.12 ETH_MMC Rx Length Error Frame Statistics Register (ETH_MMCRXLEEFER)

Reset value: 0x00000000h

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
RXLEECNT[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read as "0", write as "0"	R/W											
b15~b0	RXLEECNT	length error frames	This bit indicates the number of length error frames received	R											

39.6.4.13 ETH_MMC Rx Out of Range Error Frame Statistics Register (ETH_MMCRXOREFR)

Reset value: 0x00000000h

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
RXORECNT[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read as "0", write as "0"	R/W											
b15~b0	RXORECNT	Out of Range Error Frames	This bit indicates the number of out-of-range error frames received	R											

39.6.4.14 ETH_MMC Tx Unicast Good Frame Statistics Register (ETH_MMCTXUNGFR)

Reset value: 0x00000000h

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
TXUNGCONT[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read as "0", write as "0"	R/W											
b15~b0	TXUNGCONT	Unicast good frames	This bit indicates the number of unicast good frames sent	R											

39.6.4.15 ETH_MMC Tx Multicast Good Frame Statistics Register (ETH_MMCTXMUGFR)

Reset value: 0x00000000h

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
TXMUGCONT[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read as "0", write as "0"	R/W											
b15~b0	TXMUGCONT	Multicast good frames	This bit indicates the number of multicast good frames sent	R											

39.6.4.16 ETH_MMC Tx broadcast good frame statistics register (ETH_MMC_TXBRGFR)

Reset value: 0x00000000h

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
TXBRGCNT[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read as "0", write as "0"	R/W											
b15~b0	TXBRGCNT	Broadcast good frames	This bit indicates the number of broadcast good frames that have been sent	R											

39.6.4.17 ETH_MMC Tx Delay Error Frame Statistics Register (ETH_MMC_TXDEEFR)

Reset value: 0x00000000h

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
TXDEECNT[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read as "0", write as "0"	R/W											
b15~b0	TXDEECNT	Delay Error Frames	This bit indicates the number of delayed error frames sent	R											

39.6.4.18 ETH_MMC Tx Carrier Error Frame Statistics Register (ETH_MMC_TXCAEFR)

Reset value: 0x00000000h

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
TXCAECNT[15:0]															
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read as "0", write as "0"	R/W											
b15~b0	TXCAECNT	Carrier Error Frames	This bit indicates the number of carrier error frames sent	R											

39.6.4.19 ETH_MMC Tx Delay Collision Error Frame Statistics Register (ETH_MMC_TXLCEFR)

Reset value: 0x00000000h

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
TXLCECNT[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15~b0	TXLCECNT	Delay Collision Error Frames	This bit indicates the number of delayed collision error frames that have been sent	R

39.6.4.20 ETH_MMC Tx Excessive Collision Error Frame Statistics Register (ETH_MMC_TXECEFR)

Reset value: 0x00000000h

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
TXECECNT[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15~b0	TXECECNT	Excessive collision error frames	This bit indicates the number of excessive collision error frames that have been sent	R

39.6.4.21 ETH_MMC Tx Excessive Delay Error Frame Statistics Register (ETH_MMC_TXEDEFR)

Reset value: 0x00000000h

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
TXEDECNT[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15~b0	TXEDECNT	Excessive Delay Error Frames	This bit indicates the number of excessively delayed error frames that have been sent	R

40 External Memory Controller (EXMC)

40.1 Summary

The external memory controller EXMC (External Memory Controller) is an independent module used to access various off-chip memories and realize data exchange. EXMC can convert the internal AMBA protocol interface into various types of dedicated off-chip memory communication protocol interfaces through configuration, including SRAM, PSRAM, NOR Flash, NAND Flash and SDRAM. EXMC is divided into multiple sub-modules, each sub-module supports a specific memory type, and the user can control the external corresponding type of memory by configuring the registers of the sub-module.

40.2 Basic Features

40.2.1 Function list

The list of basic functions of EXMC is shown below.

Table 40-1 Basic functions of EXMC

External support Memory type	<ul style="list-style-type: none">SRAMPSRAMNOR FlashNAND FlashSDRAM
	<ul style="list-style-type: none">Support interface conversion between AMBA protocol and various external memories
	<ul style="list-style-type: none">Programmable interface timing of various types of memory
	<ul style="list-style-type: none">Support 8-bit, 16-bit, 32-bit MEM bus width
	<ul style="list-style-type: none">NOR Flash and PSRAM support address line and data line multiplexing
Automatic segmentation	<ul style="list-style-type: none">When the AMBA bus bit width does not match the external memory bit width, it supports automatic segmentation and byte selection control
Interrupt type	<ul style="list-style-type: none">NFC ECC calculation complete interrupt
	<ul style="list-style-type: none">NFC ECC error interrupt
	<ul style="list-style-type: none">NFC device access end interrupt

40.2.2 Controller Architecture

Figure 40-1 A block diagram of the basic EXMC architecture. Various types of external memory controllers independently generate interfaces corresponding to protocols and send them to the port MUX logic. The port MUX logic shares the addresses, data, and control signals of various external memories on the same port and then outputs them from the port of the chip. Therefore the EXMC controller can only access one external device at a time.

Note: In the architecture diagram, the SRAM/PSRAM/NOR Flash controller is defined as SMC (Static Memory Controller), the SDRAM controller is defined as DMC (Dynamic Memory Controller), and the NAND Flash controller is defined as NFC (NAND Flash Memory Controller). When it comes to SMC, DMC, and NFC, it has the same meaning as the corresponding controller.

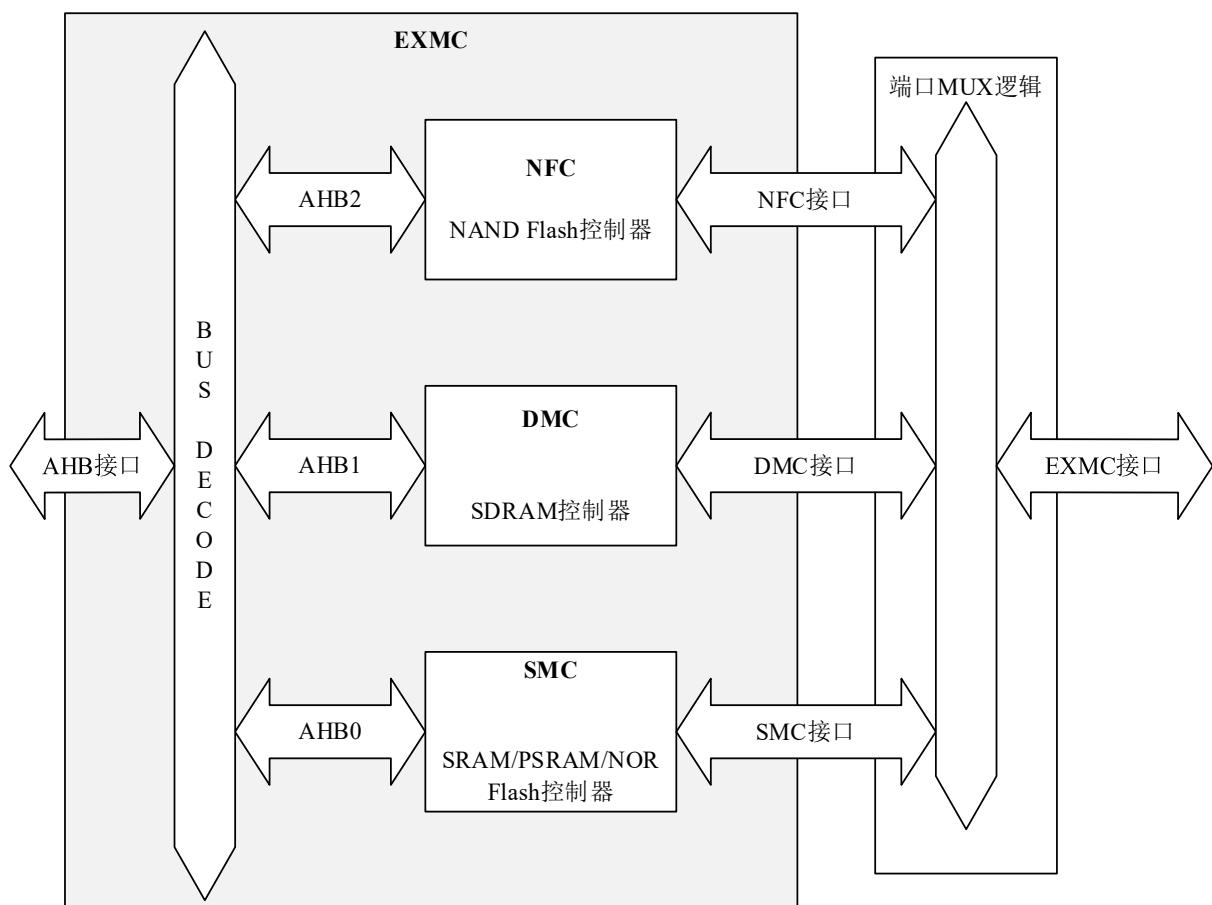


Figure 40-1 EXMC Architecture Diagram

40.2.3 Basic Access Specification

EXMC is a conversion interface between AHB bus protocol and external memory protocol. In the data transfer process of SMC and DMC, AHB data width and memory data width may be different. In order to ensure the consistency of data transmission, read and write access needs to comply with the following specifications:

1. The AHB access width is equal to the memory width, and the data is accessed normally
2. The AHB access width is larger than the memory width, and the AHB access is automatically divided into several consecutive memory data width transfers
3. The AHB access width is smaller than the memory width, and the corresponding byte is accessed through the byte selection control signal SMC_BLS[3:0] or DMC_DQM[3:0]

For the specific access methods between AHB access width and memory with various data bit widths, please refer to the following Table 40-2 .

Table 40-2 AHB access width and memory bit width corresponding access mode table

R/W	AHB access width	External memory bit width	Notes
R	8	16	
	16		
	32		Converted to 2 EXMC read operations
	8	32	
	16		
	32		
W	8	16	Use byte control signal BLS[0] or DQM[1]
	16		
	32		Converted to 2 EXMC write operations
	8	32	Use byte control signal BLS[0] or DQM[3:1]
	16		Use byte control signals BLS[1:0] or DQM[3:2]
	32		

Note: The above table only applies to data access for SMC and DMC

40.2.4 Address mapping

This product defines an external memory access range with a total size of 1GB, which is used for different types of external memory and internal data exchange, including EXMC and QSPI. The 1GB space pressFigure 40-2 The scheme allocation, SRAM/PSRAM/NOR Flash (SMC) data access mapping 512MB address space, SDRAM (DMC) data access mapping 128MB address space, NAND Flash (NFC) control and data access mapping 1MB address space , QSPI control and data access mapping 128MB address space.

Note: The access control whose address space is 0x98000000~0x9FFFFFFF is implemented by the QSPI module. For details, please refer to the QSPI chapter.

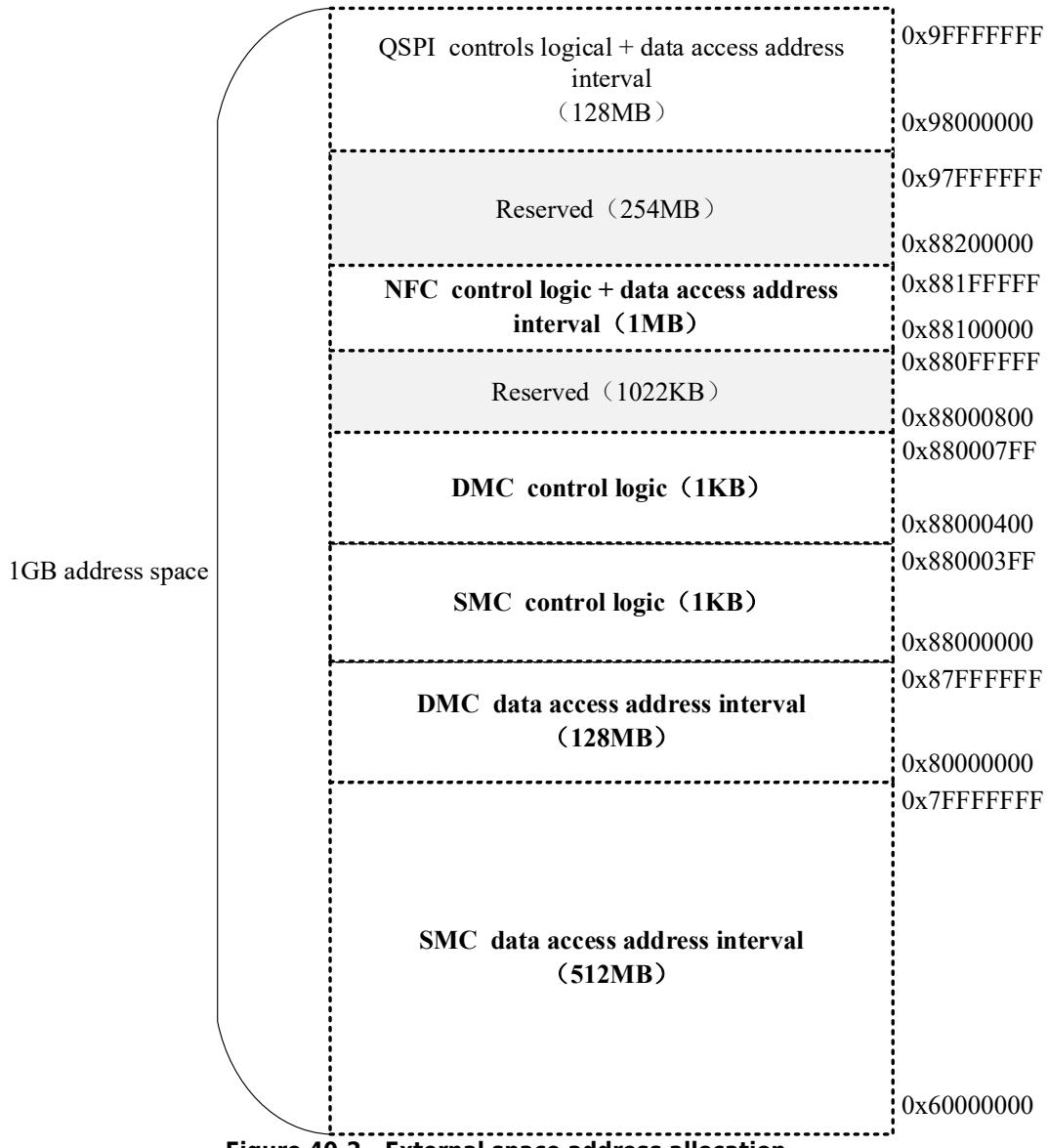


Figure 40-2 External space address allocation

The address space of the SMC can be divided into 4 chips of programmable size, and the space size of each chip can be divided by the setting of the chip select control register (SMC_CSCR0~1). The external memory objects corresponding to each Chip can be configured independently, and correspondingly have different CS (Chip Select) signals output to the external ports. Figure 40-3 is the Chip Select division of the SMC. When the address space accessed by the CPU is in the CS space of a certain SMC, the corresponding SMC_CS signal output becomes an active level.

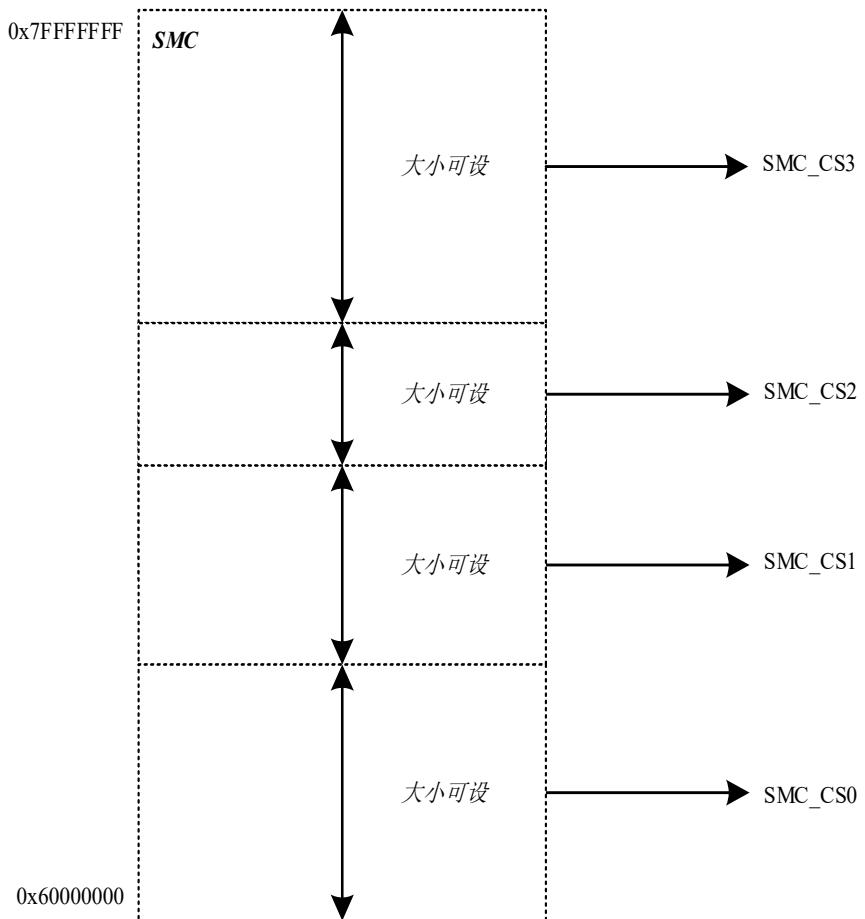


Figure 40-3 SMC's address space division

The address space of DMC can be divided into 4 chips of programmable size, and the space size of each chip can be divided by the setting of the chip select control register (DMC_CSCR0~3). The external memory objects corresponding to each Chip can be configured independently, and correspondingly have different CS (Chip Select) signals output to the external ports. Figure 40-4 is the Chip Select division of the DMC. When the address space accessed by the CPU is in the CS space of a certain DMC, the corresponding DMC_CS signal output becomes an active level.

Note: For the address setting method of SMC and DMC, refer to Table 40-16 setting example.

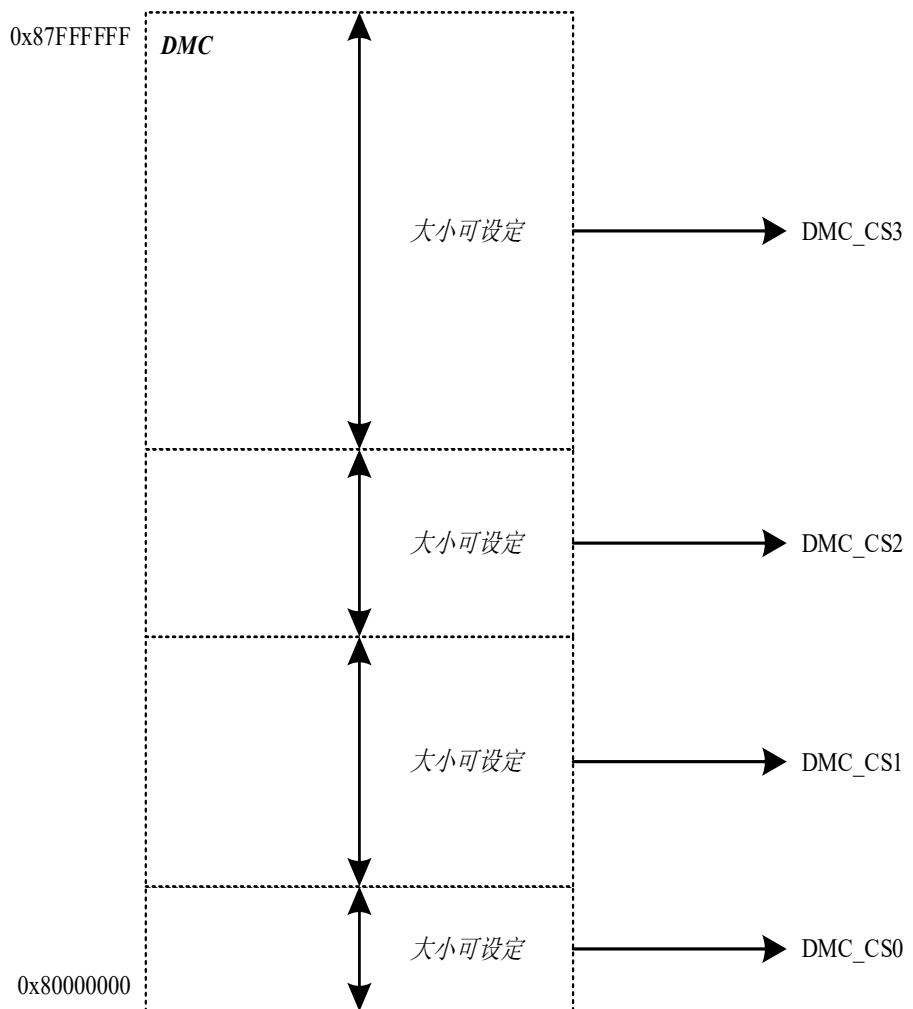


Figure 40-4 DMC address space division

The address space of NFC is divided into data register space and control register space. Among them, the data register occupies 32KB space, the control register occupies 32KB space, a total of 64KB, and the remaining addresses are reserved for NFC. DownFigure 40-5 indicates the address space division of NFC.

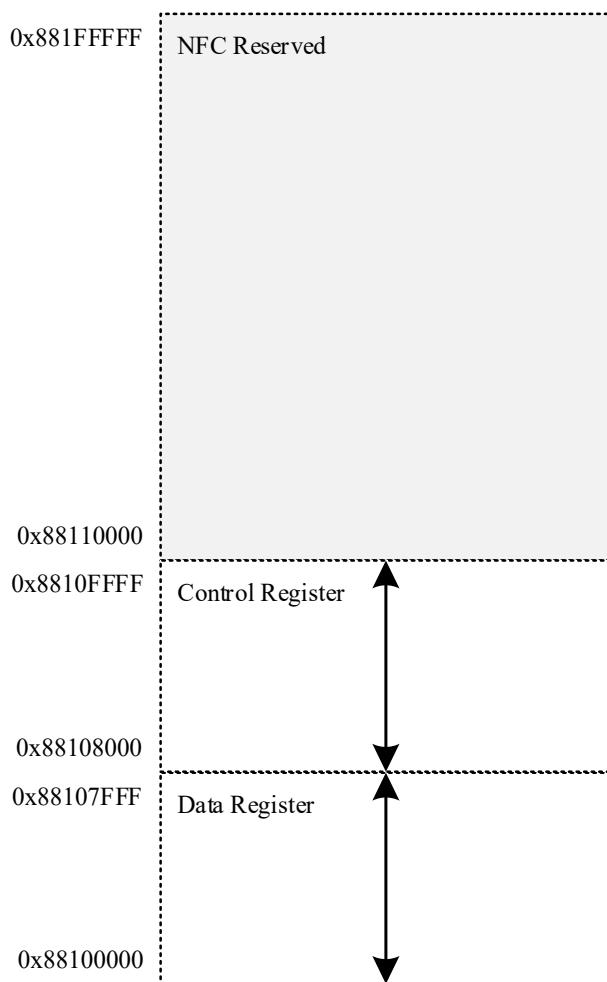


Figure 40-5 NFC address space division

40.2.5 Protocol interface

SMC, DMC, and NFC correspond to different types of memory, and each has different types of protocol interface signals. EXMC realizes data exchange with external memory through this interface protocol. The following is the description of the protocol interface required by various types of memory.

40.2.5.1 SMC protocol interface

The following table shows the protocol interfaces required for SRAM/PSRAM/NOR Flash control (SMC) access.

Table 40-3 SMC protocol interface

Protocol interface name	Direction	Active level	Functional description
SMC_CLK	out	-	SMC clock output
SMC_ADD[29:0]	out	-	Address output of SMC
SMC_DATA[31:0]	inout	-	SMC data
SMC_WE	out	L	SMC write enable
SMC_OE	out	L	SMC output enable
SMC_CS[3:0]	out	L	Chip select signal of SMC
SMC_BLS[3:0]	out	L	Byte strobe signal for SMC
SMC_ADV	out	L	SMC's address latch signal
SMC_CRE	out	H	SMC's Configuration Register Mode Signals
SMC_WAIT	in	L	SMC input wait signal
SMC_BAA	out	L	SMC address prompt signal

40.2.5.2 DMC protocol interface

The following table shows the protocol interfaces required for SDRAM Control (DMC) access.

Table 40-4 DMC protocol interface

Protocol interface name	Direction	Active level	Functional description
DMC_CLK	out	-	DMC clock output
DMC_ADD[15:0]	out	-	Address output of DMC
DMC_DATA[31:0]	inout	-	DMC data
DMC_CKE	out	H	DMC CLK output enable
DMC_AP	out	H	Automatic charging signal for DMC
DMC_WE	out	L	DMC write enable
DMC_RAS	out	L	DMC row address strobe signal
DMC_CAS	out	L	DMC column address strobe signal
DMC_BA[1:0]	out	-	Bank address signal of DMC
DMC_CS[3:0]	out	L	Chip select signal of DMC
DMC_DQM[3:0]	out	L	DMC byte strobe signal

40.2.5.3 NFC protocol interface

The table below shows the protocol interfaces required for NAND Flash Control (NFC) access.

Table 40-5 NFC protocol interface

Protocol interface name	Direction	Active level	Functional description
NFC_CLE	out	H	NFC command latch signal
NFC_ALE	out	H	NFC address latch signal
NFC_DATA[15:0]	inout	-	NFC address and data
NFC_CE[7:0]	out	L	NFC chip select signal
NFC_WE	out	L	NFC write enable
NFC_RE	out	L	NFC read enable
NFC_WP	out	L	NFC write-protect signal
NFC_RB[7:0]	in	L	NFC input busy signal

Based on the interface signal table of the above three types of external memory, configure the EXMC port function of the chip as follows.

Table 40-6 EXMC port function assignment

Pin name	Direction	Active level	Functional description		
			SMC	DMC	NFC
EXMC_CLK	O	H/L	SMC_CLK	DMC_CLK	-
EXMC_ADD29~0	O	H/L	SMC_ADD[29:18]	-	-
	O	H/L	SMC_ADD[17:16]	DMC_BA[1:0]	-
	O	H/L	SMC_ADD[15:0]	DMC_ADD[15:0]	-
	IO	H/L	SMC_DATA[31:16]	DMC_DATA[31:16]	-
EXMC_DATA31~0	IO	H/L	SMC_DATA[15:0]	DMC_DATA[15:0]	NFC_DATA[15:0]
	O	L	SMC_WE	DMC_WE	NFC_WE
EXMC_CE7~0	O	L	SMC_BLS[3:0]	DMC_DQM[3:0]	NFC_CE[7:4]
	O	L	SMC_CS[3:0]	DMC_CS[3:0]	NFC_CE[3:0]
EXMC_OE	O	L	SMC_OE	DMC_RAS	NFC_RE
EXMC_BAA	O	L	SMC_BAA	DMC_CAS	NFC_WP
EXMC_ADV	O	L	SMC_ADV	-	-
EXMC_ALE	O	H	SMC_CRE	DMC_CKE	NFC_ALE
EXMC_CLE	O	H	-	DMC_AP	NFC_CLE
EXMC_RB7~0	I	L	-	-	NFC_RB[7:1]
	I	L	SMC_WAIT	-	NFC_RB[0]

40.3 Functional description

40.3.1 SMC-SRAM/PSRAM/NOR Flash Controller

40.3.1.1 SRAM/PSRAM/NOR Flash introduction

SRAM/NOR Flash is a memory with static access function, which can save the data stored in it without a refresh circuit. The width of the address line determines the number of memory cells, and the space corresponding to the memory storage width on each memory cell is used to store data. Therefore, the capacity of the SRAM/PSRAM/NOR Flash memory is the memory data width $\times 2^{\text{address line width}}$.

PSRAM is a pseudo static random access memory, because it has the interface of SRAM, it can exchange data with the SMC module of EXMC inside the system like SRAM.

40.3.1.2 SMC basic functions

The basic features of SMC are as follows:

- Supports 16-bit, 32-bit external memory data bandwidth
- AHB word, halfword, byte access
- Provides independent chip select control for each memory chip
- Byte select signal output
- Address line, data line multiplexing
- Programmable protocol timing parameters
- Programmable rate automatic refresh operation (used for PSRAM)
- Has 2 47-bit command FIFOs
- Has 4 36-bit write data FIFOs
- Has 4 32-bit read data FIFOs
- low power management

40.3.1.3 SMC initial setting

The state diagram of SMC after power-on reset and the switching between states are as followsFigure 40-6 shown.

The SMC is in the Low Power state during initial reset; the SMC can be switched between the Ready state and the Low Power state through the setting of the state control registers (SMC_STCR0 and SMC_STCR1). For details, please refer to [SMC Low Power Management] chapter.

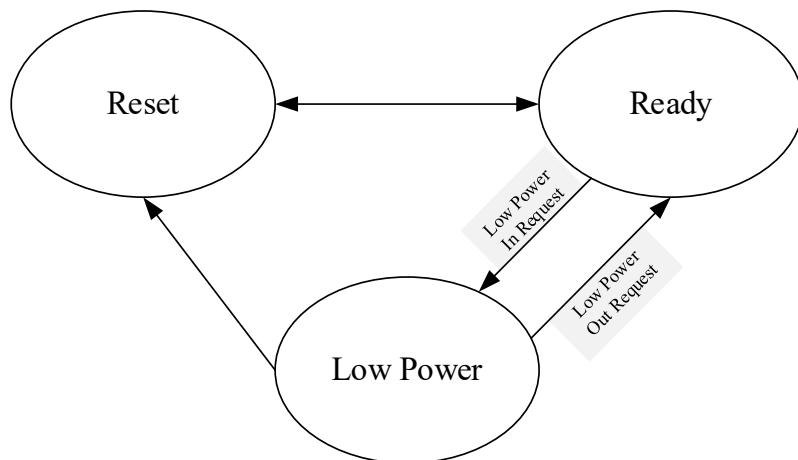


Figure 40-6 SMC state diagram

Before the SMC communicates with external SRAM, PSRAM, NOR Flash, etc., it must be initialized and configured with relevant parameters to ensure correct data transmission. Refer to Figure 40-7 below for the specific setting sequence.

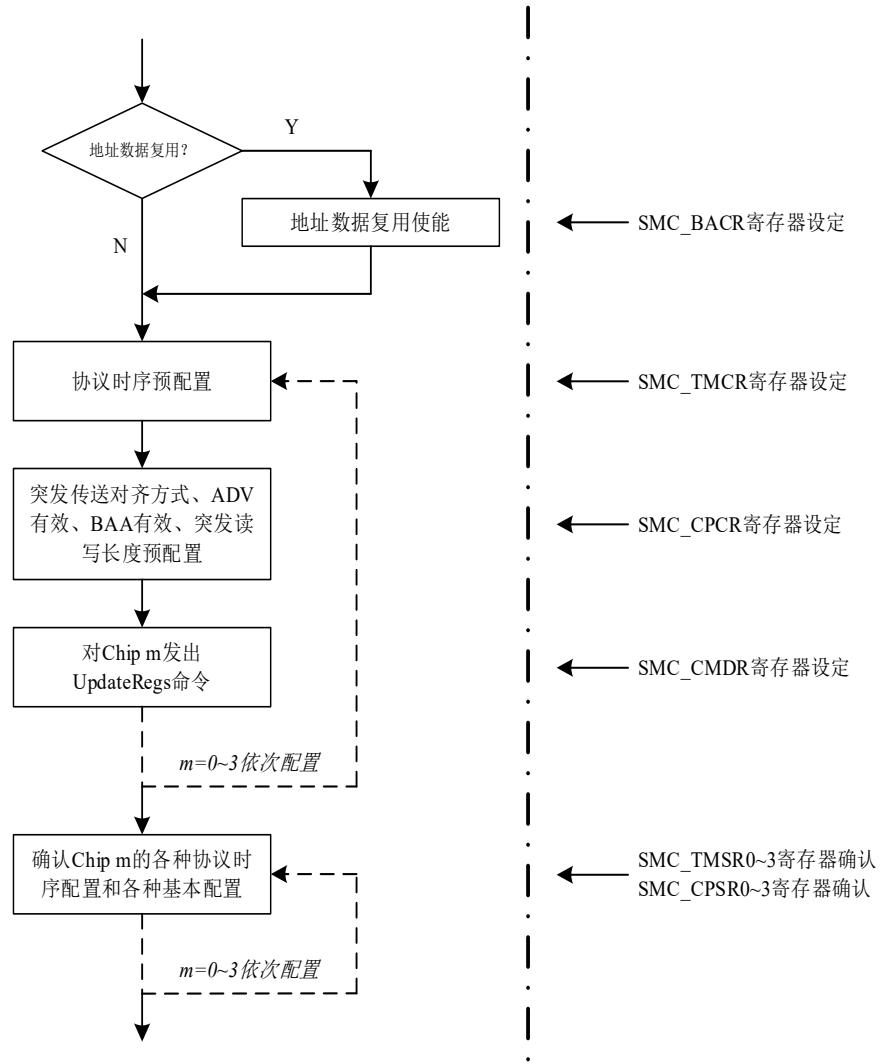


Figure 40-7 SMC initial setting process

40.3.1.4 SMC access action

Interview method

The SMC can select synchronous access or asynchronous access to read and write to the external memory.

The synchronous access mode is the external memory corresponding to the synchronous access type. When the SMC outputs the read and write control signals, the clock signal is also output at the same time. The external memory transmits data with the SMC according to the clock output by the SMC, that is, SMC_CLK is valid; For the external memory of asynchronous access type, when

the SMC outputs read and write control signals, the clock signal is not output (the output is always low), and the external memory realizes data access according to other control signals.

In the synchronous access mode, the SMC_WAIT signal is used to control the read and write actions between the SMC and the external memory. The default value of SMC_WAIT is high level. When the input is low level, the SMC is in a waiting state and does not perform any operation. After the input changes to high level, the SMC can normally perform read and write access.

By setting the RSYN and WSYN bits of the CHIP configuration register (SMC_CPCR), the synchronous read/write mode or the asynchronous read/write mode can be selected.

Access timing

SMC supports access to various types of memory, and can convert AHB single or burst read and write operations into memory read and write operations. The following lists several timing examples of data read and write between the controller and external memory:

- Single read action
 - Single read action for multiplexing of address and data lines
 - Single write action
 - Single write action for address and data line multiplexing
 - Burst read action
 - Burst write action
- a) Table 40-7 and Figure 40-8 ~ Figure 40-11 This is the basic timing chart and setting example of a single read operation.

Table 40-7 One-time read operation basic setting example

Basic settings	MW	RSYN	RBL	WSYN	WBL	BAA	ADV	BLS
	<set>	<set>	b000	-	-	-	b0	b0
Timing setting	t_rc	t_wc	t_ceoe	t_wp	t_pc	t_tr	- show no concern <set> means user set value	
	b0011	-	b011	-	-	-		

t_tr: Controls the high level time of SMC_CS. 1 is recommended.

t_rc: Controls the low level time of SMC_CS.

t_ceoe: Controls the lag time when SMC_OE is pulled down relative to SMC_CS. t_ceoe is less than t_rc.

In the basic sequence diagram of the read action, SMC_DATA is the EXMC data bus, and SMC_DATA_dly is delayed by SMC_DATA 1 cycle. The blue line indicates that SMC_DATA_dly values are transferred to the internal FIFO in EXMC, and the yellow line indicates that data in the FIFO will be transferred to the internal bus. Read operations after the yellow line are invalid. Read values are not transmitted to the internal bus.

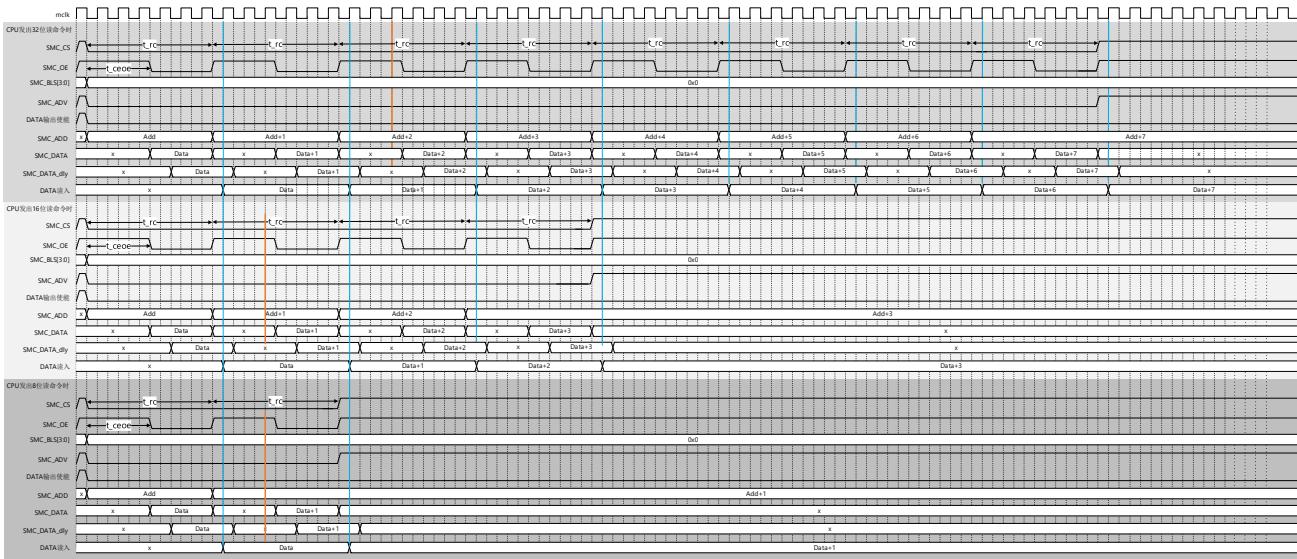


Figure 40-8 Basic timing of single read operation (asynchronous mode (RSYN=0) & 16-bit width (MW=01))

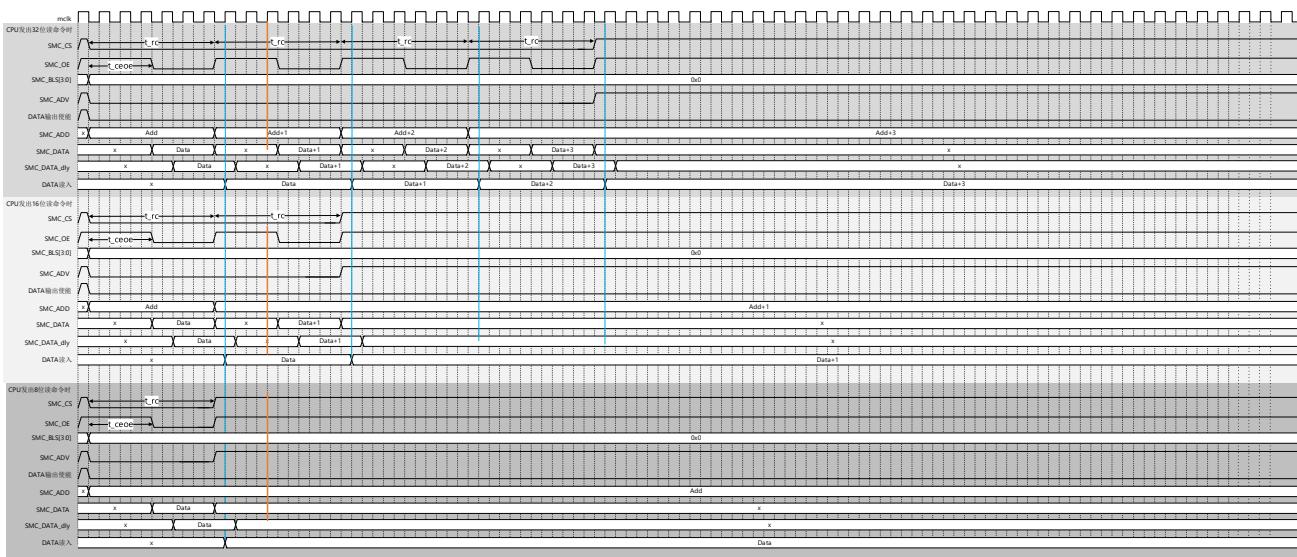


Figure 40-9 Basic sequence of single read operation (asynchronous mode (RSYN=0) & 32-bit width (MW=10))

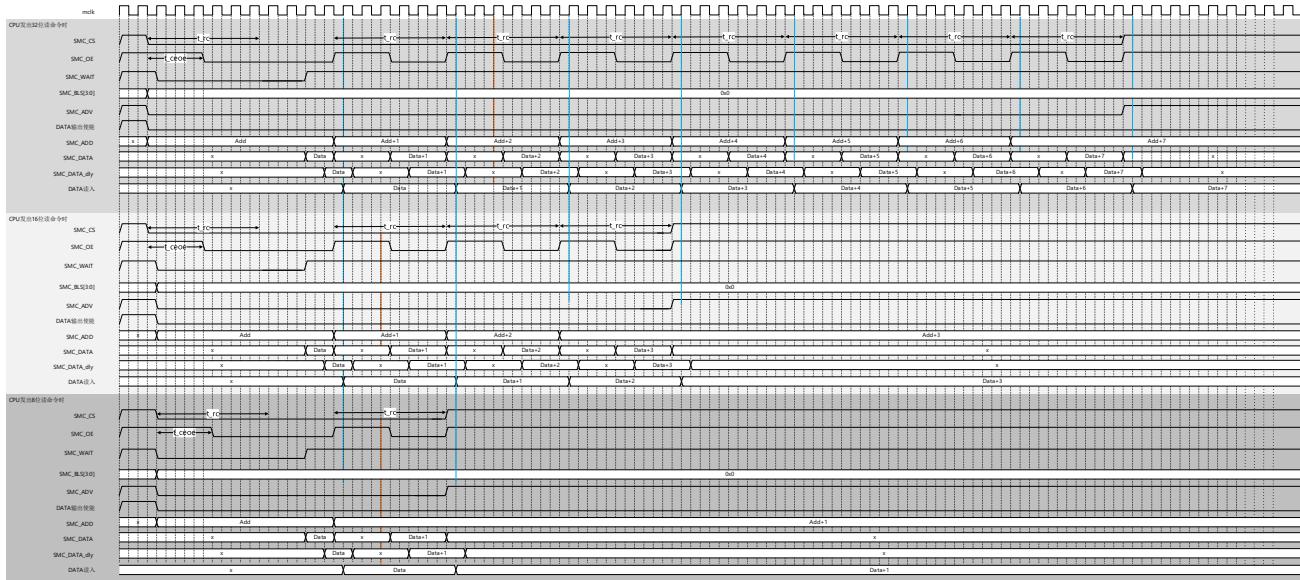


Figure 40-10 Basic timing sequence of single read operation (synchronous mode (RSYN=1) & 16-bit bit width (MW=01))

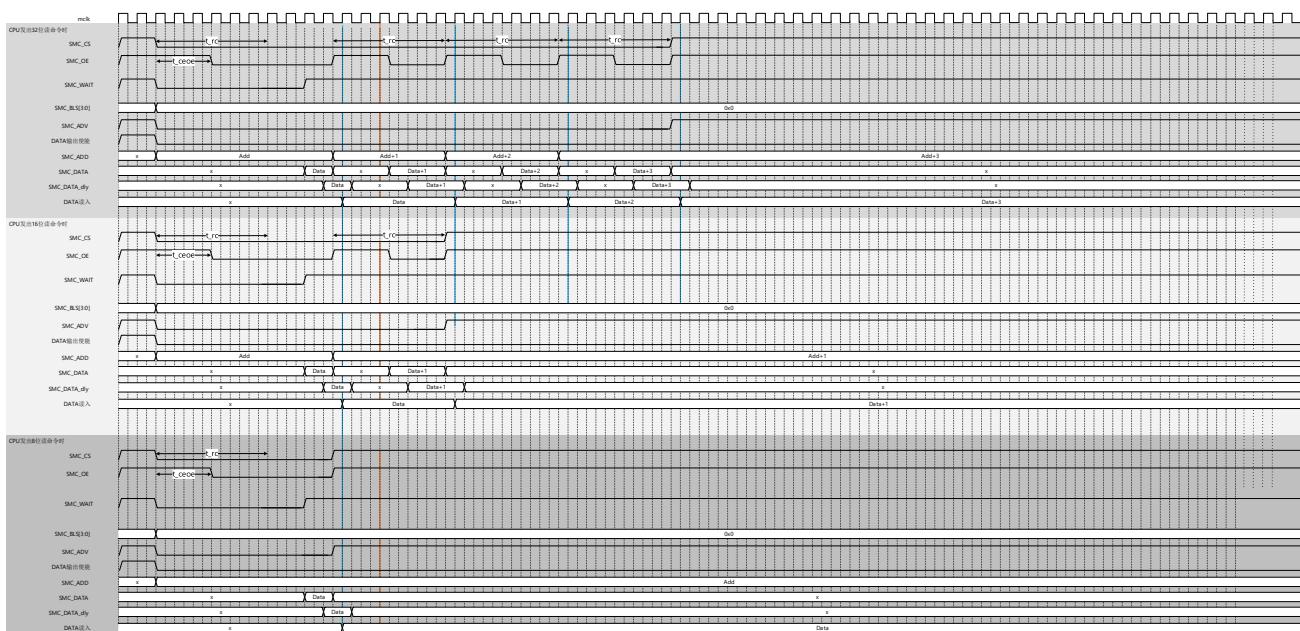


Figure 40-11 Basic timing sequence of single read operation (synchronous mode (RSYN=1) & 32-bit width (MW=10))

b) Table 40-8 and Figure 40-12 This is a basic timing chart and setting example of a single read operation with multiplexing of address and data lines.

Table 40-8 Basic setting example of address data line multiplexing single read operation

Basic settings	MW	RSTN	RBL	WSYN	WBL	BAA	ADV	BLS
	<set>	<set>	b000	-	-	-	b1	b0
Timing setting	t_rc	t_wc	t_ceoe	t_wp	t_pc	t_tr	- show no concern <set> means user set value	
	b0110	-	b011	-	-	-		

t_tr: Controls the high level time of SMC_CS. 1 is recommended.

t_rc: Controls the low level time of SMC_CS.

t_ceoe: Controls the lag time when SMC_OE is pulled down relative to SMC_CS. t_ceoe is less than t_rc.

In the basic sequence diagram of the read action, SMC_DATA is the EXMC data bus, and SMC_DATA_dly is delayed by SMC_DATA 1 cycle. The blue line indicates that SMC_DATA_dly values are transferred to the internal FIFO in EXMC, and the yellow line indicates that data in the FIFO will be transferred to the internal bus. Read operations after the yellow line are invalid. Read values are not transmitted to the internal bus. When address data is multiplexed, SMC_ADV will be retained for a period after the smc_ADV becomes higher.

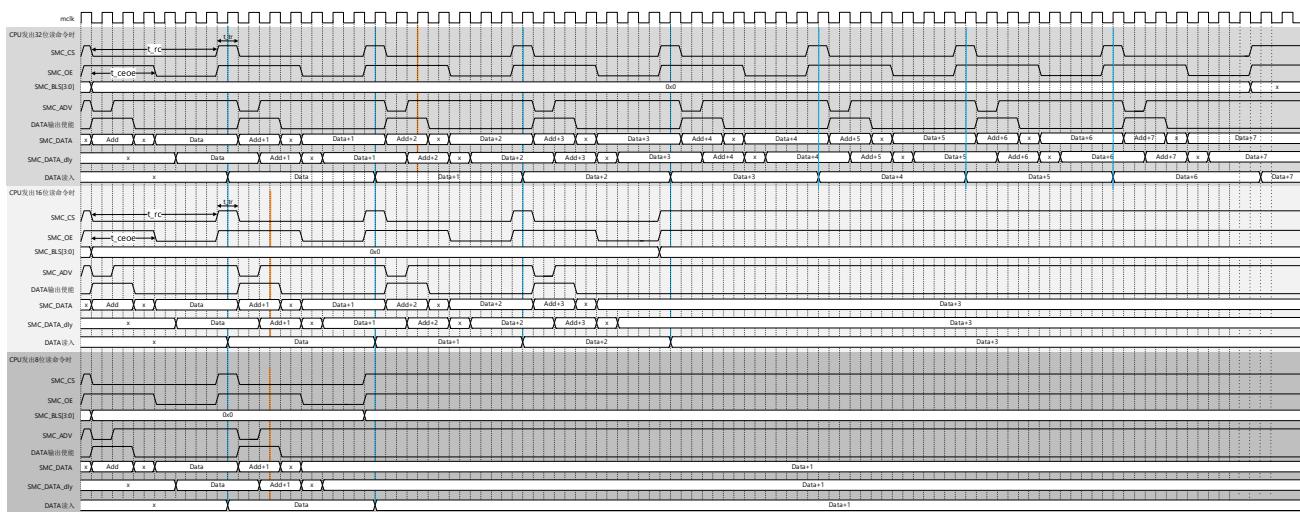


Figure 40-12 Address data line multiplexing single read operation basic timing (asynchronous mode (RSTN=0) & 16-bit width (MW=01))

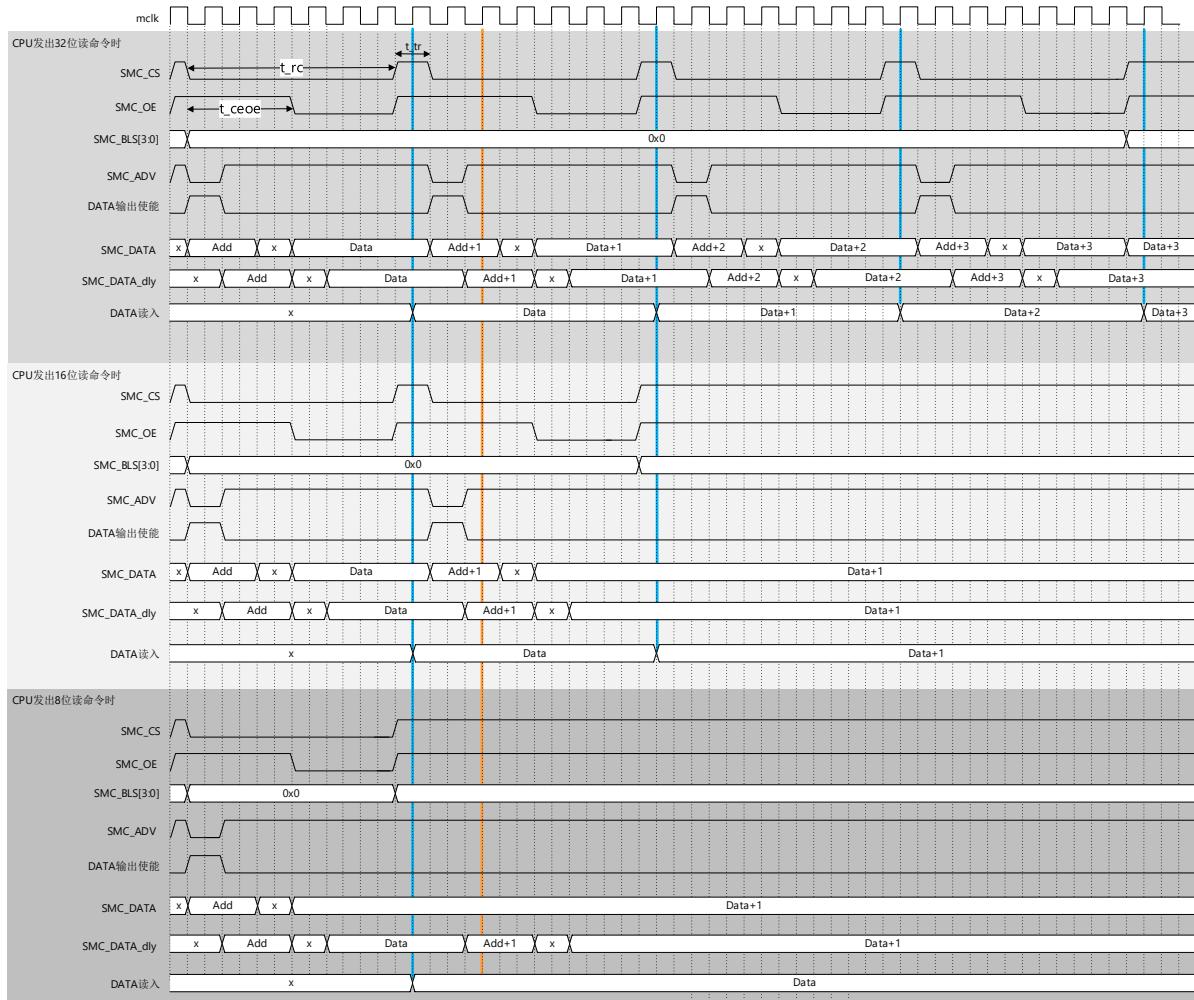


Figure 40-13 Address data line multiplexing single read operation basic timing (asynchronous mode (RSYN=0) & 32-bit width (MW=10))

- c) Table 40-9 and Figure 40-14 ~Figure 40-17 is the basic timing chart and setting example of a single write operation.

Table 40-9 One-time write operation basic setting example

Basic settings	MW	RSYN	RBL	WSYN	WBL	BAA	ADV	BLS
<set>	-	-	<set>	b000	-	b0	<set>	
Timing setting	t_rc	t_wc	t_ceoe	t_wp	t_pc	t_tr		
	-	b0100	-	b010	-	-	- show no concern <set> means user set value	

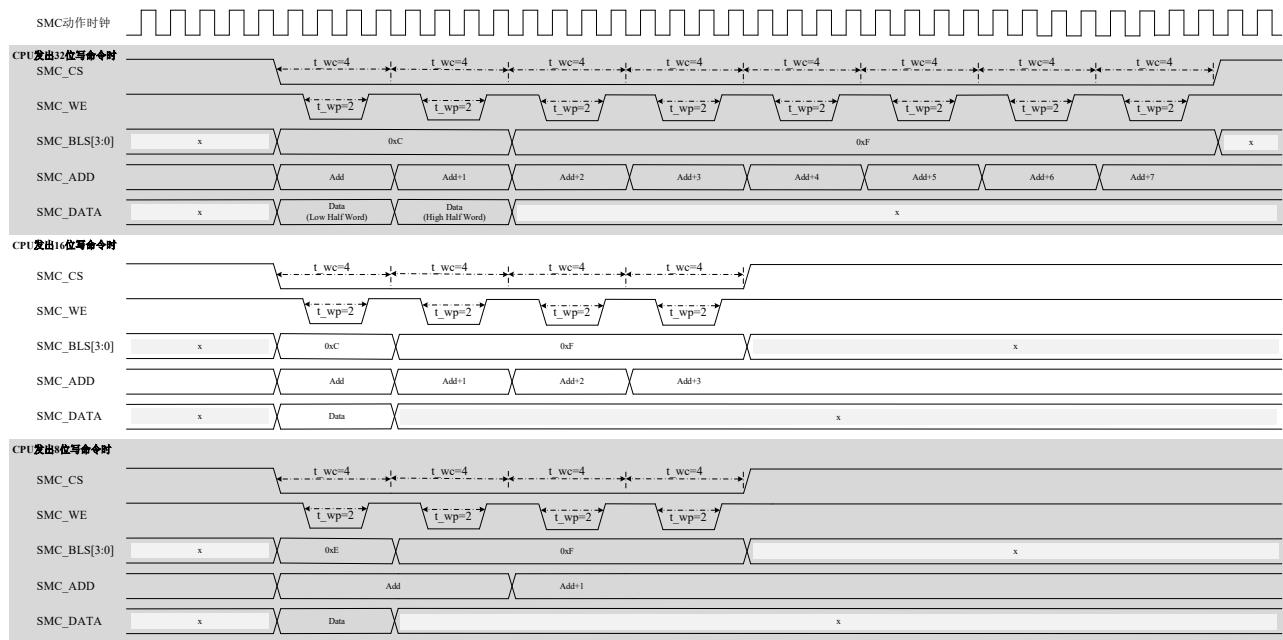


Figure 40-14 Basic timing of single write operation (asynchronous mode (WSYN=0) & 16-bit width (MW=01) & BLS=0)

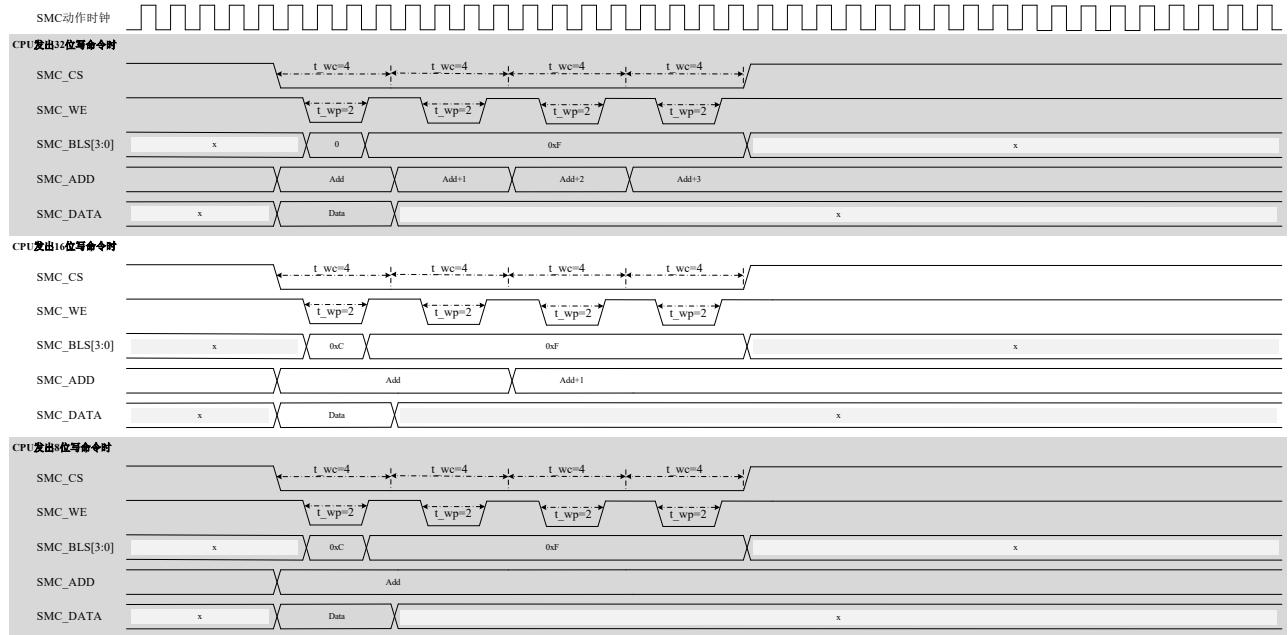


Figure 40-15 Basic timing of single write operation (asynchronous mode (WSYN=0) & 32-bit bit width (MW=10) & BLS=1)

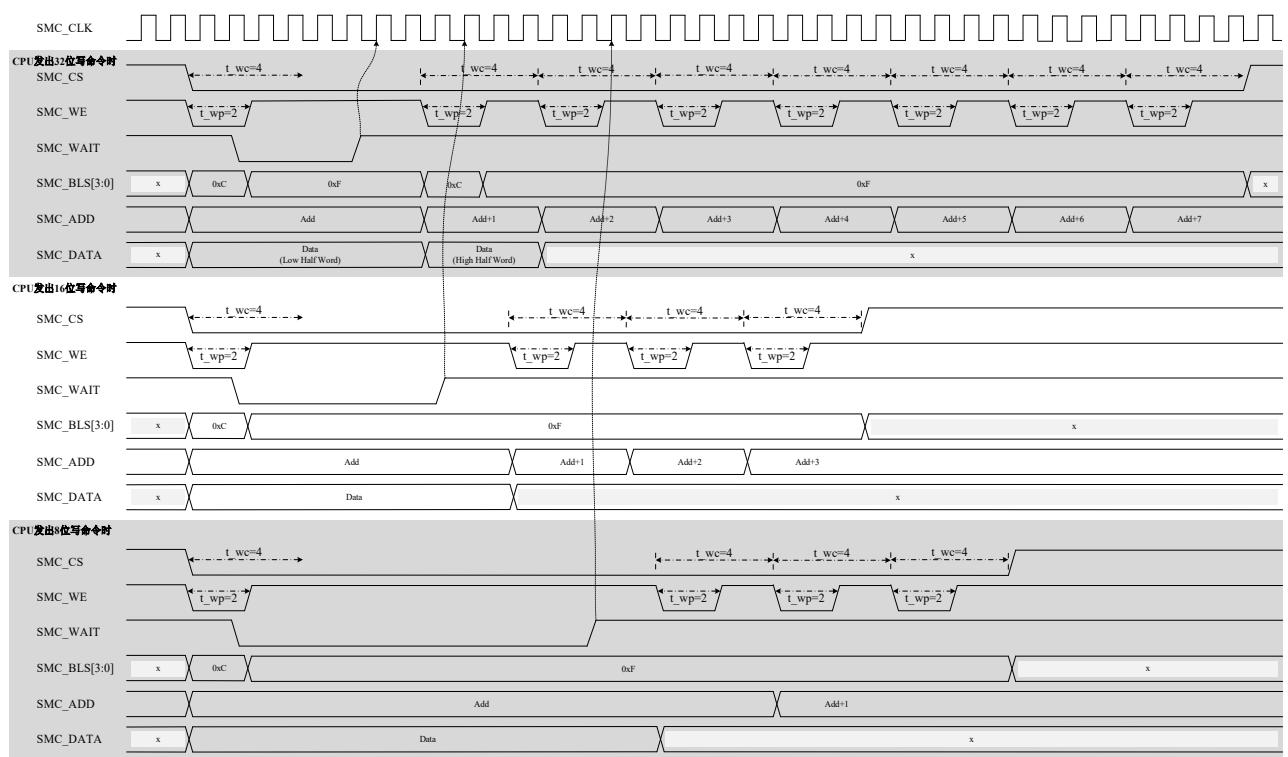


Figure 40-16 Basic timing of single write operation (synchronous mode (WSYN=1) & 16-bit bit width (MW=01) & BLS=1)

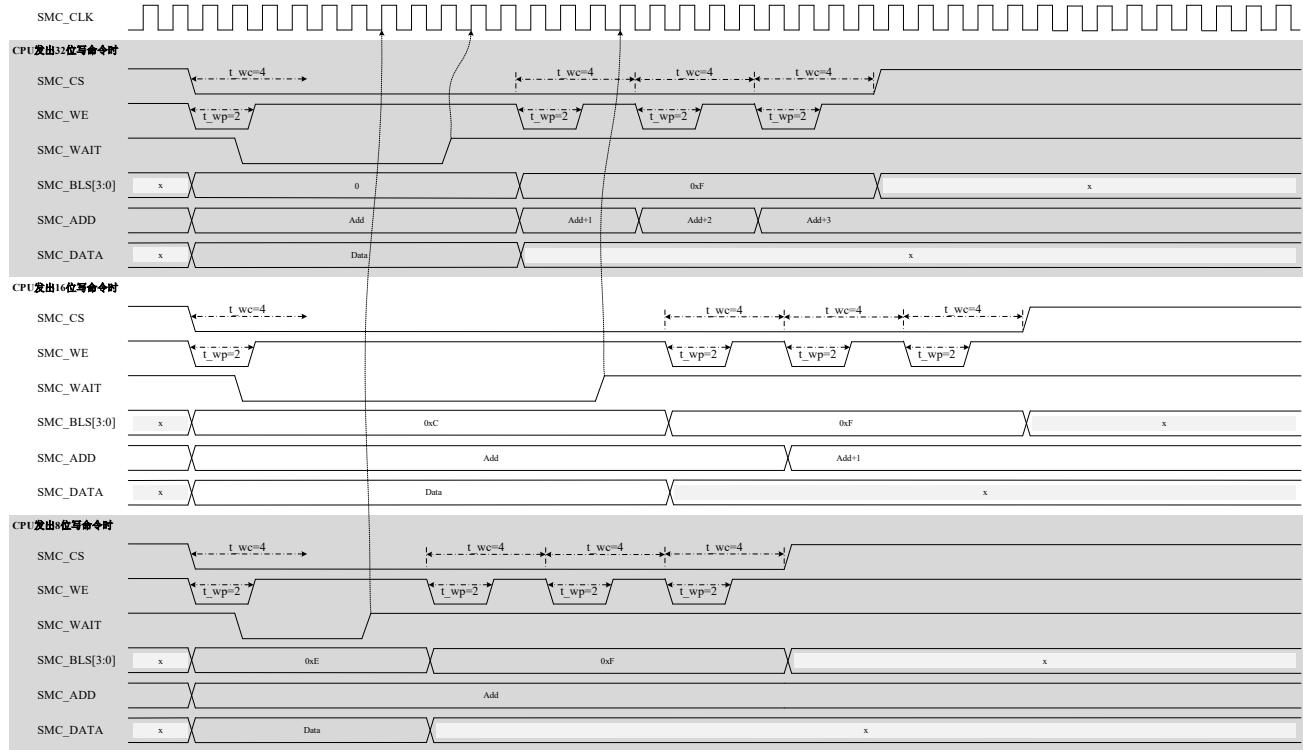


Figure 40-17 Basic timing of single write operation (synchronous mode (WSYN=1) & 32-bit bit width (MW=10) & BLS=0)

- d) Table 40-10 and Figure 40-18 is a basic timing chart and setting example of a single write operation with multiplexing of address and data lines.

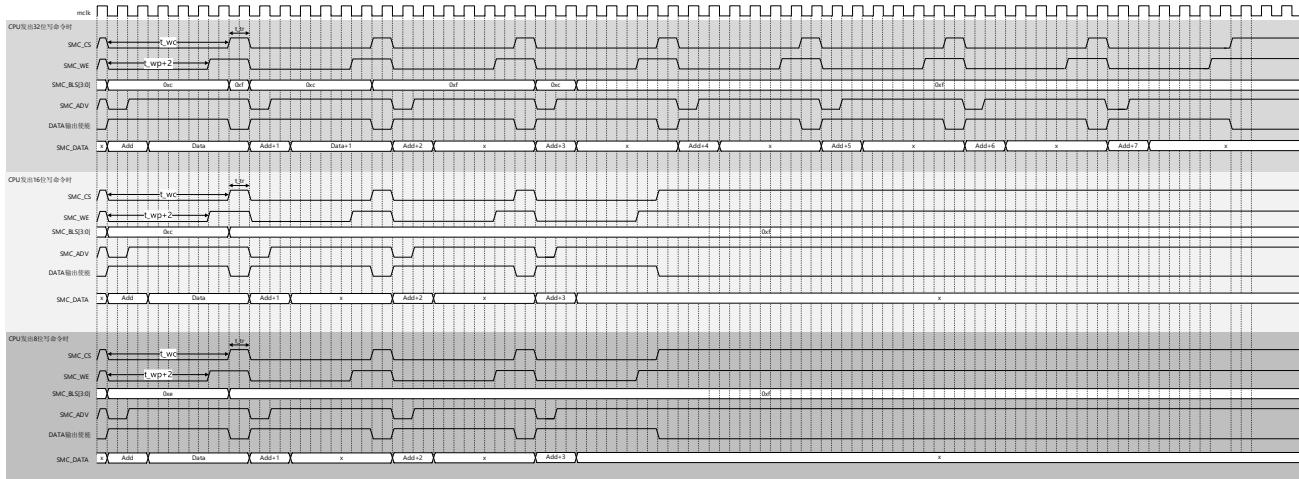
Table 40-10 Basic setting example of address data line multiplexing single write action

Basic settings	MW	RSYN	RBL	WSYN	WBL	BAA	ADV	BLS
<set>	-	-	<set>	b000	b0	b1	b1	
Timing setting	t_rc	t_wc	t_ceoe	t_wp	t_pc	t_tr	- show no concern <set> means user set value	
	-	b0110	-	b011	-	-		

t_tr: SMC_CS pull up time. 1 is recommended.

t_wc: SMC_CS pull down time.

t_wp+2: Pull down time for SMC_WE. Pull down it at the same time as SMC_CS to enable write. t_wp+2 must be less than the set value of t_wc. SMC_ADV pull-down lasts for one clock. SMC_ADV pull-down SMC_DATA represents the address to access EXMC RAM. The address persists for another cycle after SMC_ADV is pulled up, after which SMC_DATA represents the data to be written to EXMC RAM.



**Figure 40-18 Address data line multiplexing single write action basic timing (asynchronous mode)
(WSYN=0) & 16-bit width (MW=01))**

- e) Table 40-11 and Figure 40-19~Figure 40-22 is the basic timing chart and setting example of burst read operation.

Table 40-11 Basic setting example of burst read operation

Basic settings	MW	RSTN	RBL	WSYN	WBL	BAA	ADV	BLS
	<set>	<set>	<set>	-	-	-	-	b0
Timing setting	t_rc	t_wc	t_ceoe	t_wp	t_pc	t_tr	- show no concern <set> means user set value	
	b110	-	b011	-	-	-		

In the basic timing diagram of the burst read action, SMC_DATA is the EXMC data bus, and SMC_DATA_dly is delayed by the SMC_DATA 1 cycle. The blue line indicates that SMC_DATA_dly values are transferred to the internal FIFO in EXMC, and the yellow line indicates that data in the FIFO will be transferred to the internal bus. Read operations after the yellow line are invalid. Read values are not transmitted to the internal bus.

When address data is multiplexed, SMC_Add will be retained for a period after the SMC_ADV becomes higher.

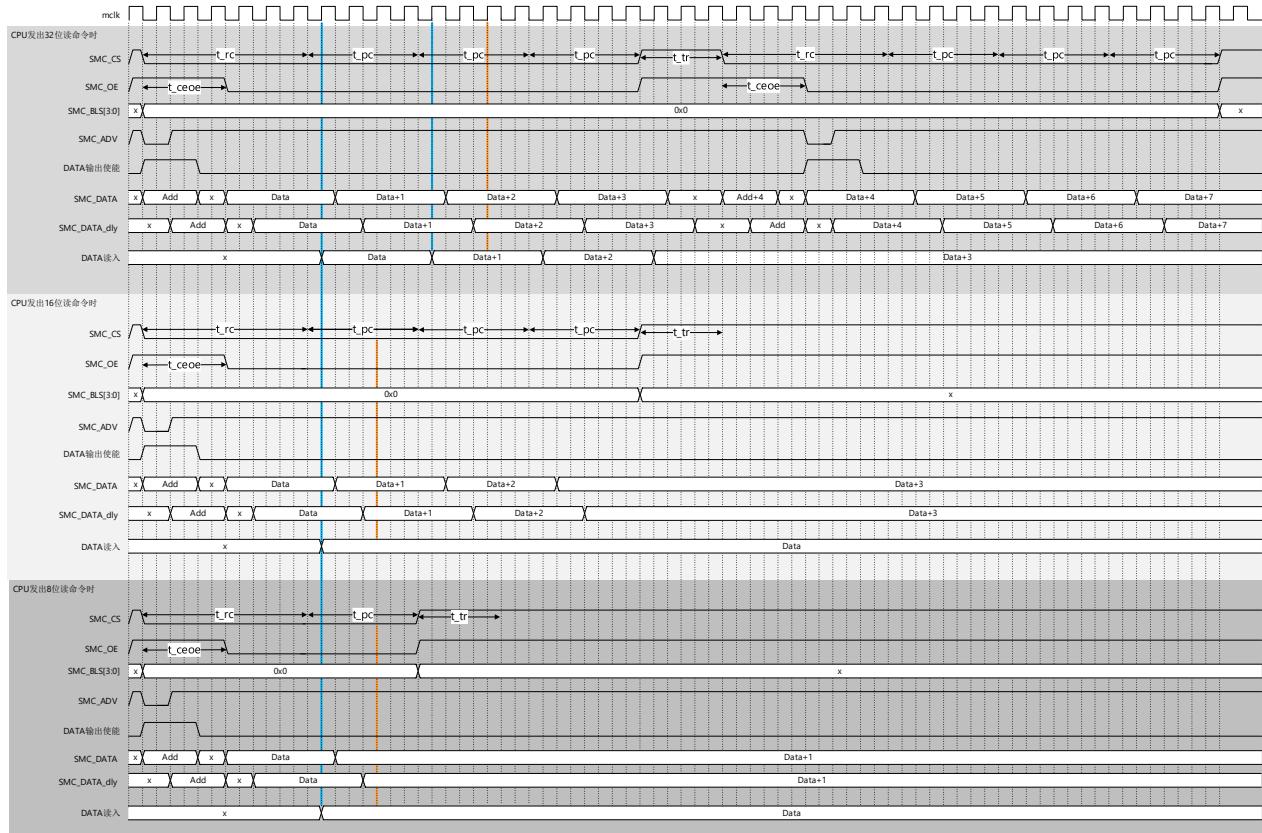


Figure 40-19 Basic Timing Of Address Data Multiplexing Burst Read Action (synchronous/asynchronous mode (RSYN=1/0) &16 bit width (MW=01) &RBL=001)

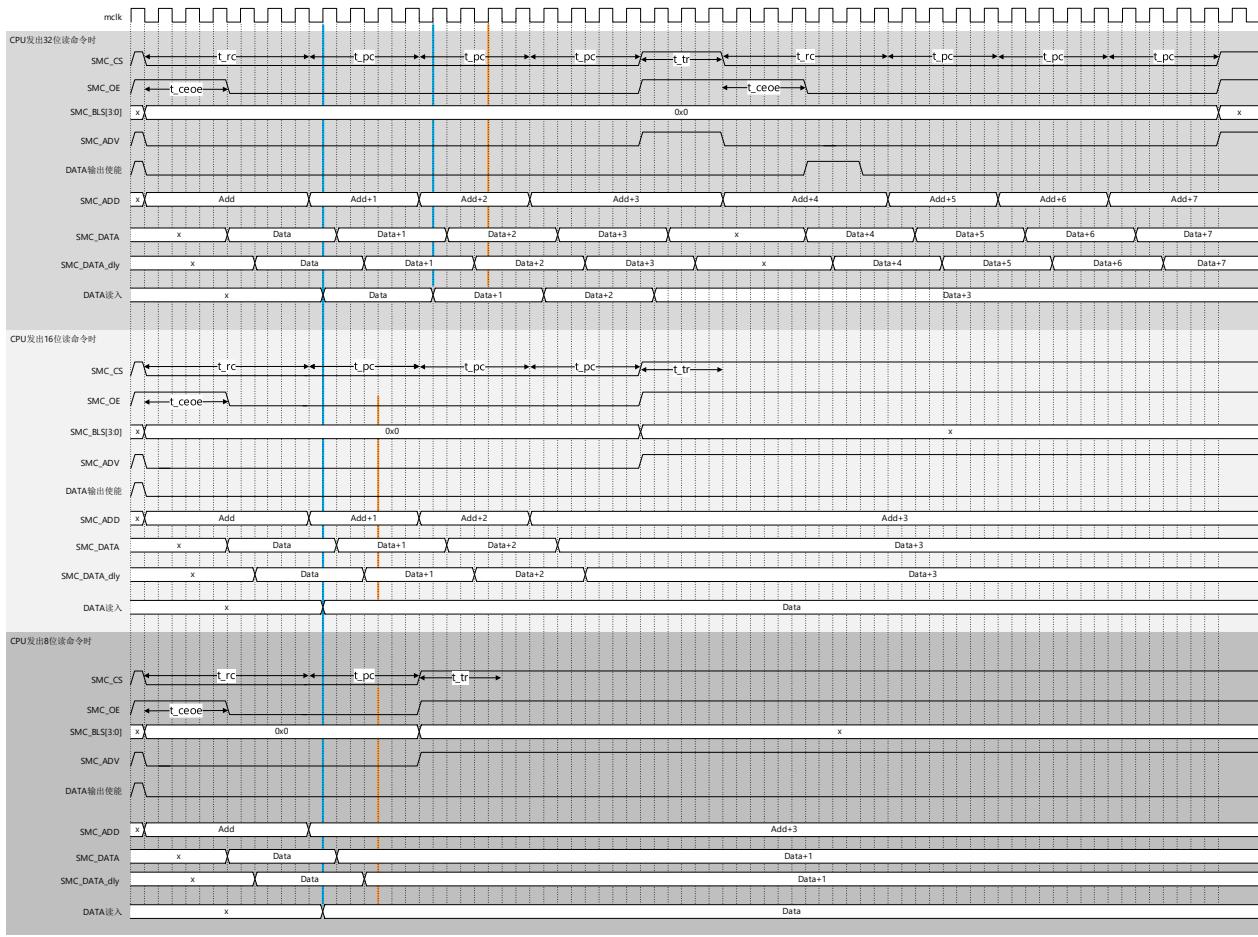


Figure 40-20 Basic Timing Of Address Data Non-Multiplexing Burst Read Action (synchronous/asynchronous mode (RSYN=1/0) &16 bit width (MW=01) &RBL=001)

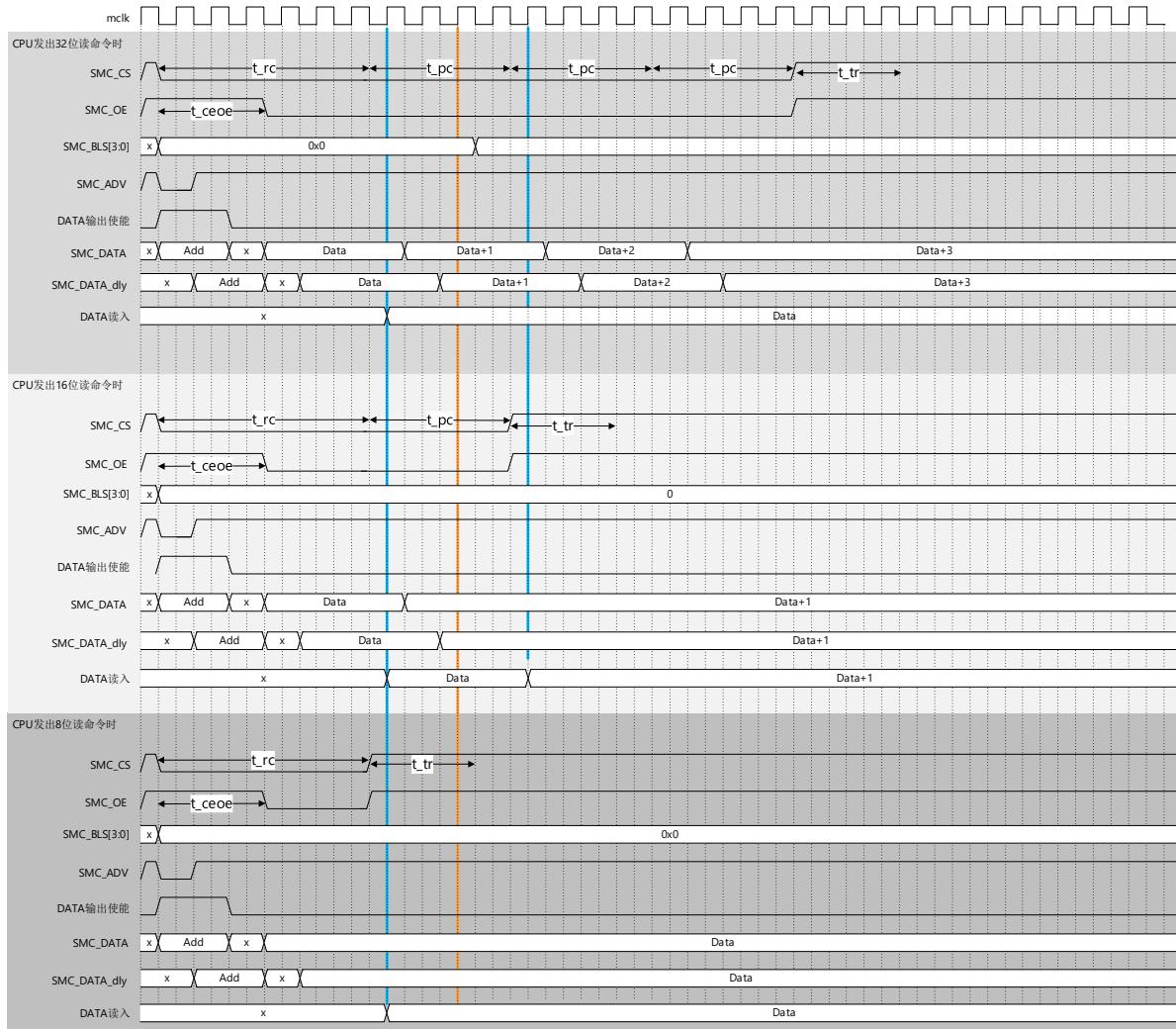
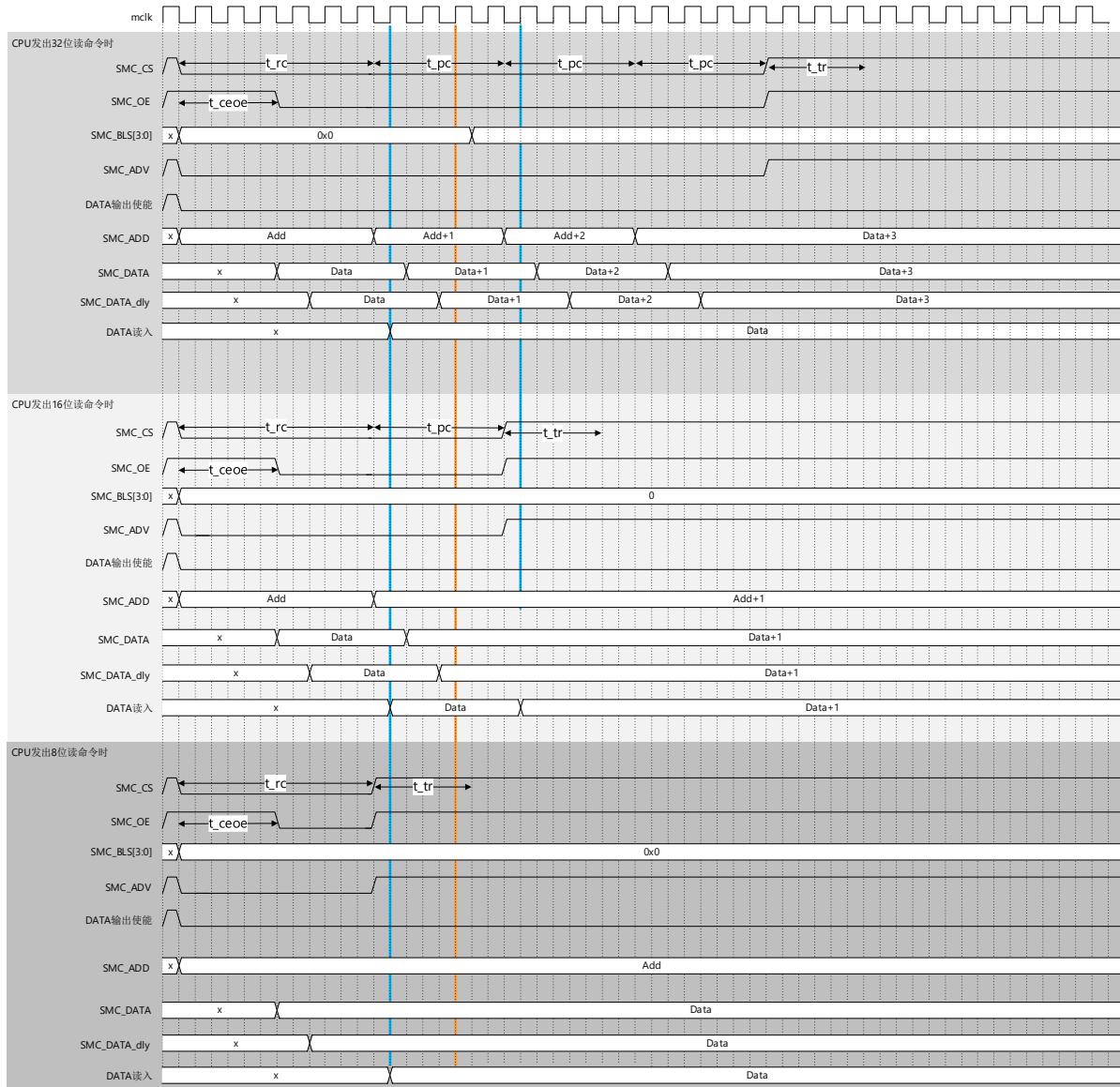


Figure 40-21 Basic Timing Of Address Data Multiplexing Burst Read Action (synchronous/asynchronous mode (RSYN=1/0) & 32-bit bit width (MW=10) &RBL=001)



**Figure 40-22 Basic Timing Of Address Data Non-Multiplexing Burst Read Action
(synchronous/asynchronous mode (RSYN=1/0) & 32-bit bit width (MW=10) & RBL=001)**

- f) Table 40-12 and Figure 40-23 is the basic timing chart and setting example of burst write operation.

Table 40-12 Basic setting example of burst write operation

Basic settings	MW	RSYN	RBL	WSYN	WBL	BAA	ADV	BLS
<set>	-	-	<set>	<set>	-	-	-	b1
Timing setting	t_rc	t_wc	t_ceoe	t_wp	t_pc	t_tr	- show no concern <set> means user set value	
	-	b0011	-	b001	-	-		

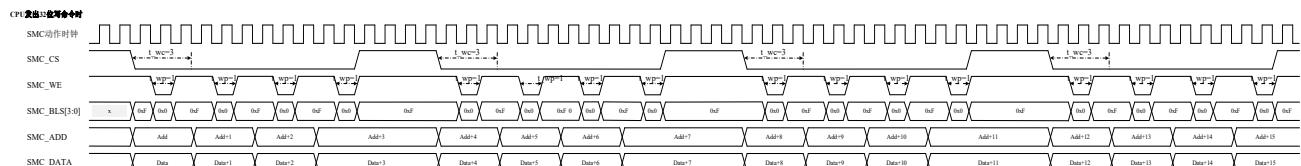


Figure 40-23 Burst write operation basic timing (asynchronous mode (WSYN=0) & 32 bit width (MW=10) & WBL=011)

FIFO management

There are 3 types of FIFOs in the SMC. They are command FIFO, write data FIFO, and read data FIFO.

Command FIFO: Two command FIFOs are used to cache two SMC commands and corresponding command addresses, CS and other information.

Write data FIFO: 4 write data FIFOs for buffering AHB write data.

Read data FIFO: 4 read data FIFOs for buffering data read in from SRAM/PSRAM/NOR Flash.

40.3.1.5 SMC Low Power Management

When the SMC is in the Low Power state, if the LPWOR bit of the state control register (SMC_STCR1) is set to 1, the SMC will change from the Low Power state to the Ready state, and the SMC will start to work normally; if there is no operation on the SMC for a long time, the state can be changed. The LPWIR bit of the control register (SMC_STCR0) is set to 1, so that the SMC enters the Low Power state. After entering the Low Power state, the internal operation of the SMC stops and the power consumption is reduced.

40.3.2 DMC-SDRAM Controller

40.3.2.1 Introduction to SDRAM

Synchronous dynamic random access memory (SDRAM) is a dynamic random access memory (DRAM) refreshed based on an external synchronous clock, and its synchronous clock is provided by the DMC_CLK pin of the DMC.

Bank Features

SDRAM can be divided into multiple banks (EXMC supports a maximum of 4 banks of memory), allowing devices to access in a staggered manner to obtain greater concurrency and data transfer volume. Each Bank is a memory matrix. The matrix is composed of rows and columns. The selection of rows and columns locates a storage unit. The storage unit corresponds to the space of the memory storage width. Therefore, the size of each Bank of the SDRAM memory can be considered as the memory data width x rows. Count x number of columns. The user can match the parameters of the DMC and the external SDRAM through the relevant settings of the chip select control register (DMC_CSCR) (row address length, column address length, etc.) to achieve communication with different SDRAMs.

Line activation

Row activation is to enable the bank where the row address is located. The complete row address consists of a 2-bit bank address (DMC_BA[1:0]) and a 16-bit row address (DMC_ADD[15:0]). Row activation will read all the bit information of the selected row into the read-write amplifier. Once the row activation is completed, the read and write operations can proceed smoothly.

Row activation takes a certain amount of time, this time interval is called row and column delay, which is the minimum time interval from row addressing to column addressing. The row and column delay (t_{rcd}) of the DMC is the minimum number of clock cycles including the row and column delay of the SDRAM, which represents the minimum waiting time between the row and the SDRAM read and write, which can be set through the timing configuration register (DMC_TMCR_t_rcd). During this period of time, since the operation of the SDRAM controller to the Bank is carried out independently, the user can issue control commands to other Bank addresses. Figure 40-24 is the timing diagram from row activation to read and write operations.

Note: Row activations are indicated as "Active" in the timing diagrams in the EXMC chapter.

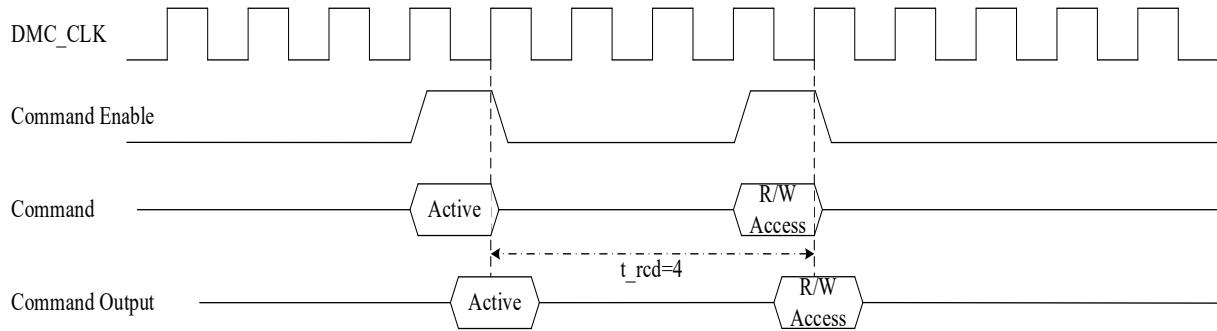


Figure 40-24 Row activation to read and write operation timing

Precharge

If the DMC needs to perform row switching when accessing external data, it needs to close the original valid (working) row of the Bank and resend the row and column addresses. Bank closes an existing working row and prepares to open a new row is precharging. The precharge action can be triggered by the precharge command (PrechargeAll) of the command register (DMC_CMDR), or when a line feed is read or written.

In a burst read or burst write command, the A10/A8 (set by DMC_CPCR.APBS) bit becomes "H", and a precharge action is automatically added after the read and write operation is completed to realize automatic precharge.

The row precharge delay (t_{rp}) represents the minimum time for SDRAM row switching, which is the minimum time interval from the completion of precharge to the next row enable command, which can be set through the timing configuration register (DMC_TMCR_t_rp). Down Figure 40-25 , Figure 40-25 is a timing diagram of two different ways of precharging.

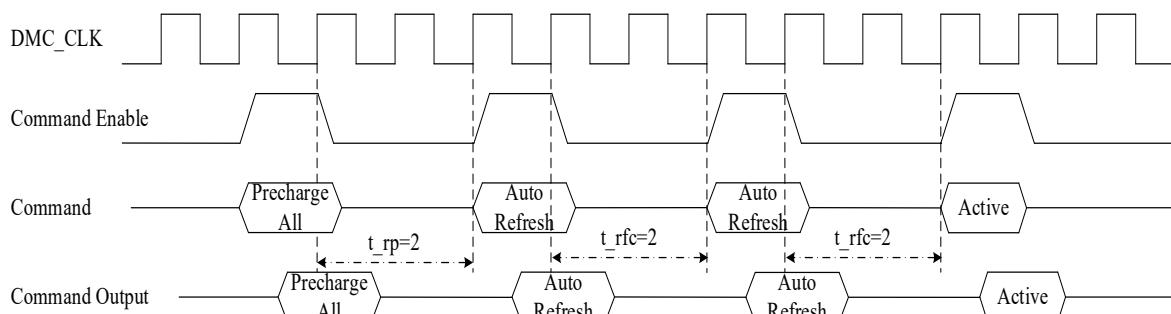


Figure 40-25 Command mode precharge

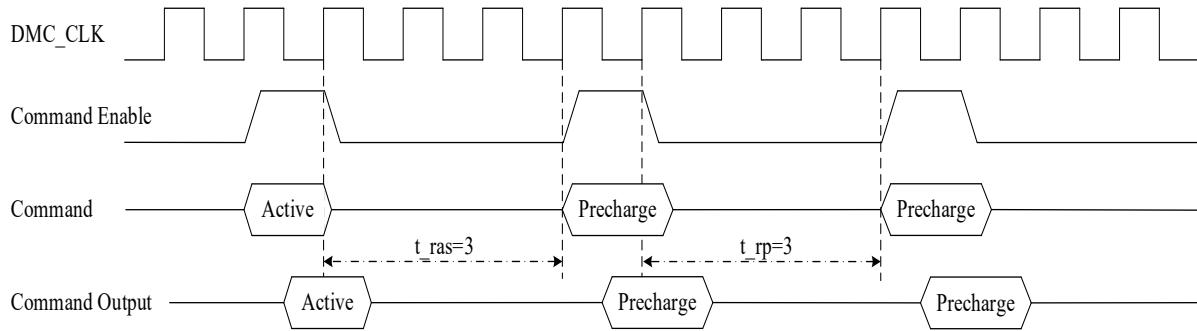


Figure 40-26 Automatically precharge after reading and writing

Refresh action

Due to the leakage phenomenon of the memory cells of the dynamic memory, in order to maintain the correctness of the data of each memory cell, it is necessary to ensure that all memory cells can be periodically refreshed. EXMC supports two refresh modes, Auto Refresh (AR) and Self Refresh (SR).

Automatic refresh (AR) is EXMC periodically provides refresh commands, and the refresh interval is determined by the refresh time register (DMC_RFTR). The refresh object is all banks (the corresponding rows of all banks are refreshed at the same time), and the automatic refresh operation is only started when all banks are idle and not in a low-power state. During the execution of automatic refresh, only empty operations can be input, and other operation commands can only be waited and cannot be executed. After the automatic refresh is executed, all banks enter the idle state.

Self-refresh (SR) is mainly used for data preservation in the low-power state of sleep mode. The self-refresh mode is entered when DMC_CKE is in an invalid state during the AR command. During SR, SDRAM no longer relies on the DMC_CLK clock to work, but performs the refresh operation according to the internal clock of SDRAM. During SR, all external signals except DMC_CKE are invalid (no need to provide external refresh commands). When the DMC changes from the Pause state to the Low Power state (by issuing the Sleep command), it automatically issues a self-refresh command. Figure 40-27 is the entry and exit timing diagram of the self-refresh action.

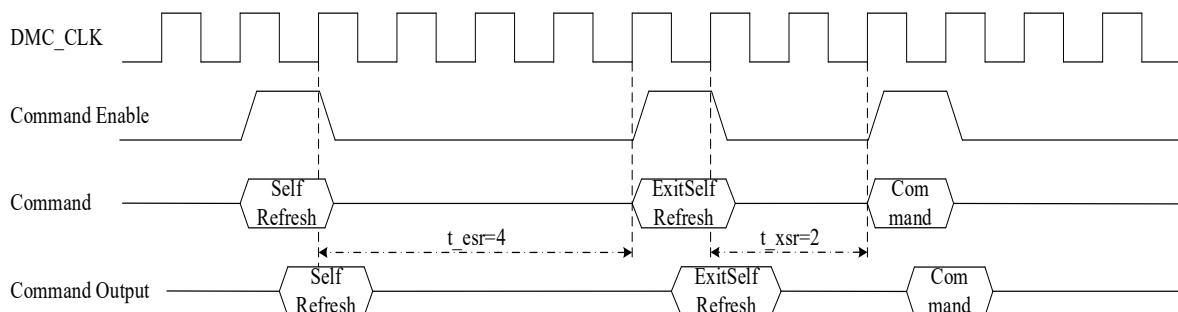


Figure 40-27 Entry and exit timing of self-refresh action

Command truth table

Various commands for controlling and accessing the SDRAM are issued by the SDRAM protocol interface of the DMC. The specific command list is shown in the following table.

Table 40-13 Command truth table for DMC

DMC_CS	DMC_RAS	DMC_CAS	DMC_WE	DMC_BA	DMC_ADD		Order
					ADD[xx]	ADD[10]/[8]	
0	0	0	0	0	Config		MdRegConfigCommand
0	0	0	1	-	-	-	SelfRefresh command (DMC_CKE=0)
							AutoRefresh command (DMC_CKE=1)
0	0	1	0	-	-	1	PrechargeAll command
					Bank	-	Prechagre command to specify Bank
0	0	1	1	Bank	Row	Row	line activation command
0	1	0	0	Bank	Col	1	Precharge command during write action
					Col	0	write action command
0	1	0	1	Bank	Col	1	Precharge command during read action
					Col	0	read action command
0	1	1	0	-	-	-	Abort burst transfer
0	1	1	1	-	-	-	no action
1	-	-	-	-	-	-	prohibited by command

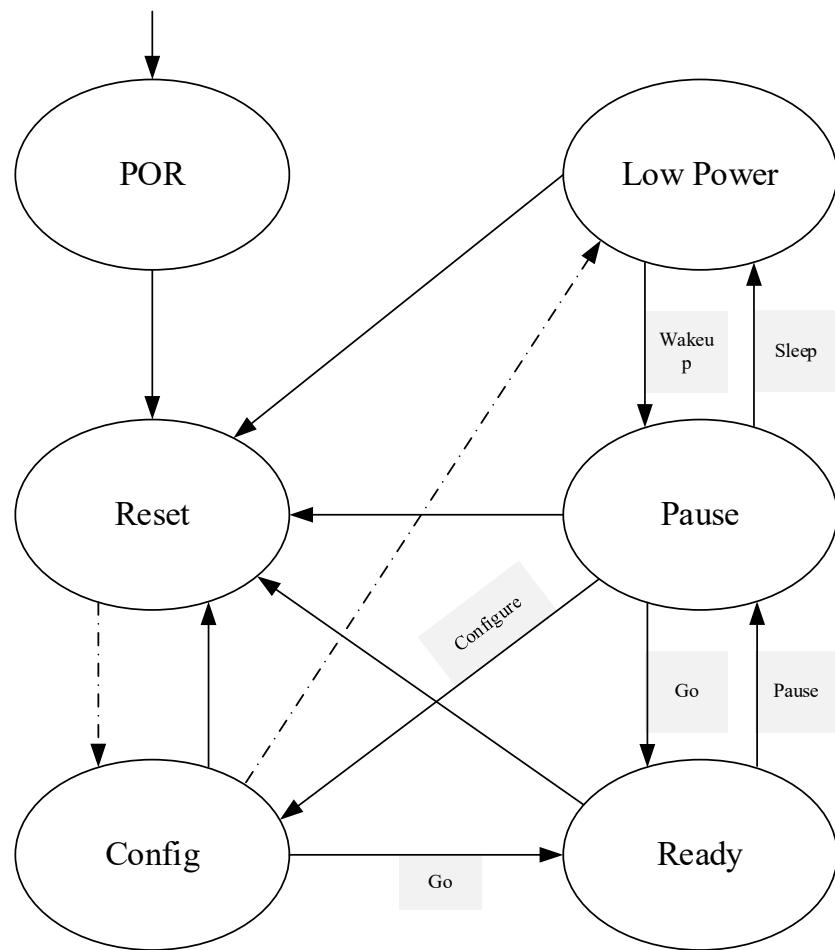
40.3.2.2 DMC basic functions

The basic features of DMC are as follows:

- Supports 16-bit, 32-bit external memory data bandwidth
- Support up to 16-bit row address, 12-bit column address, 2-bit internal bank address
- AHB word, halfword, byte access
- Provides independent chip select control for each memory chip
- Each memory chip size can be independently configured
- Byte select signal output
- Automatic row and bank boundary management
- Programmable protocol timing parameters
- Programmable rate automatic refresh action
- Has 2 41-bit command FIFOs
- Has 10 36-bit write data FIFOs
- Has 10 32-bit read data FIFOs
- Low power management

40.3.2.3 DMC initial settings

The state diagram of DMC after power-on reset and the switching diagram between states are as followsFigure 40-28 shown. During the initial reset, the DMC enters the Config state. At this time, if the Go command is issued through the state control register (DMC_STCR), the DMC automatically enters the Low Power state when there is no operation. During normal operation, the state of the DMC can be switched through the state control register (DMC_STCR).) setting to control, for details, please refer to [DMC low power management] chapter.

**Figure 40-28 DMC state diagram**

Before the DMC communicates with the external SDRAM, it must perform initial settings and configure relevant parameters to ensure correct data exchange.. Refer to Figure 40-29 below for the specific setting sequence.

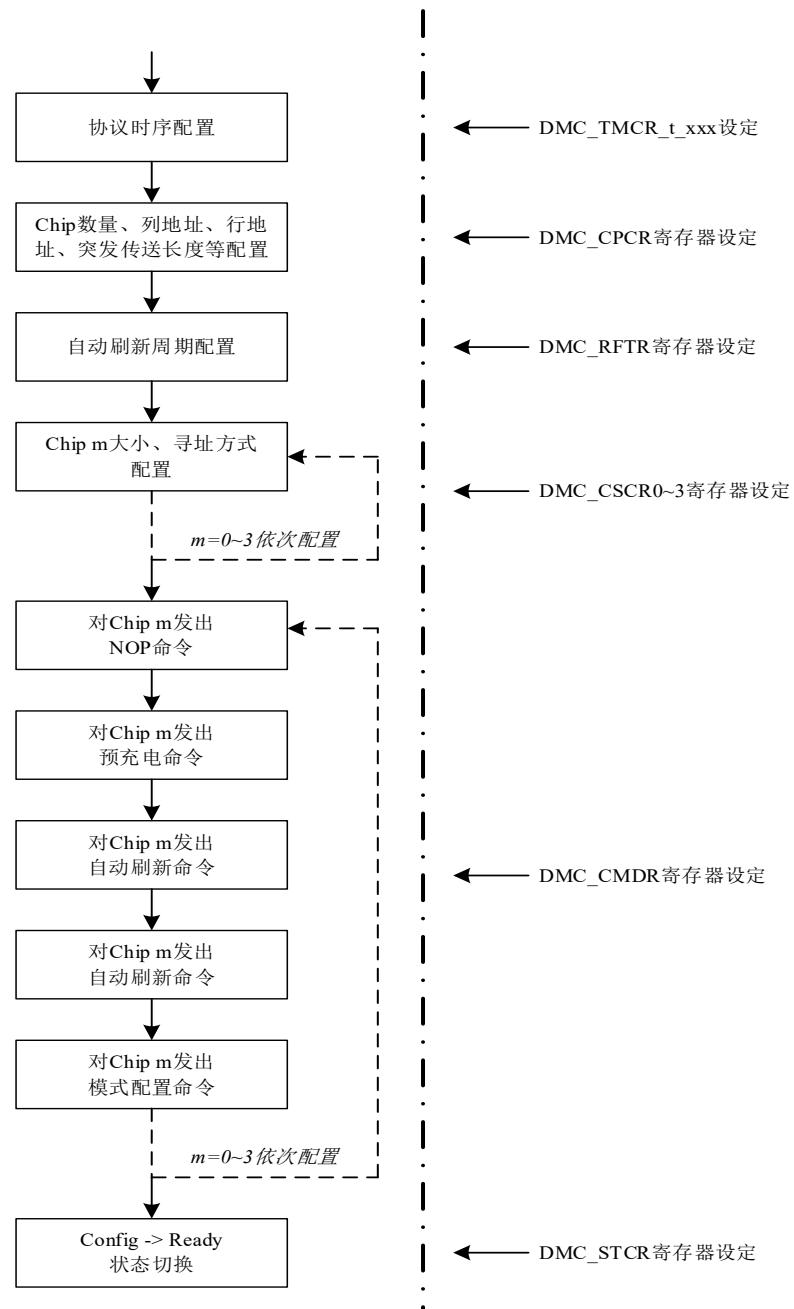


Figure 40-29 DMC initial setup process

40.3.2.4 DMC access action

Access timing

DMC can convert AHB single or burst read and write operations into memory read and write operations. There are several ways to read and write data with external SDRAM memory:

- single read action
- single write action
- burst read action
- burst write action

a) Figure 40-30 is the basic timing diagram of a single read operation.

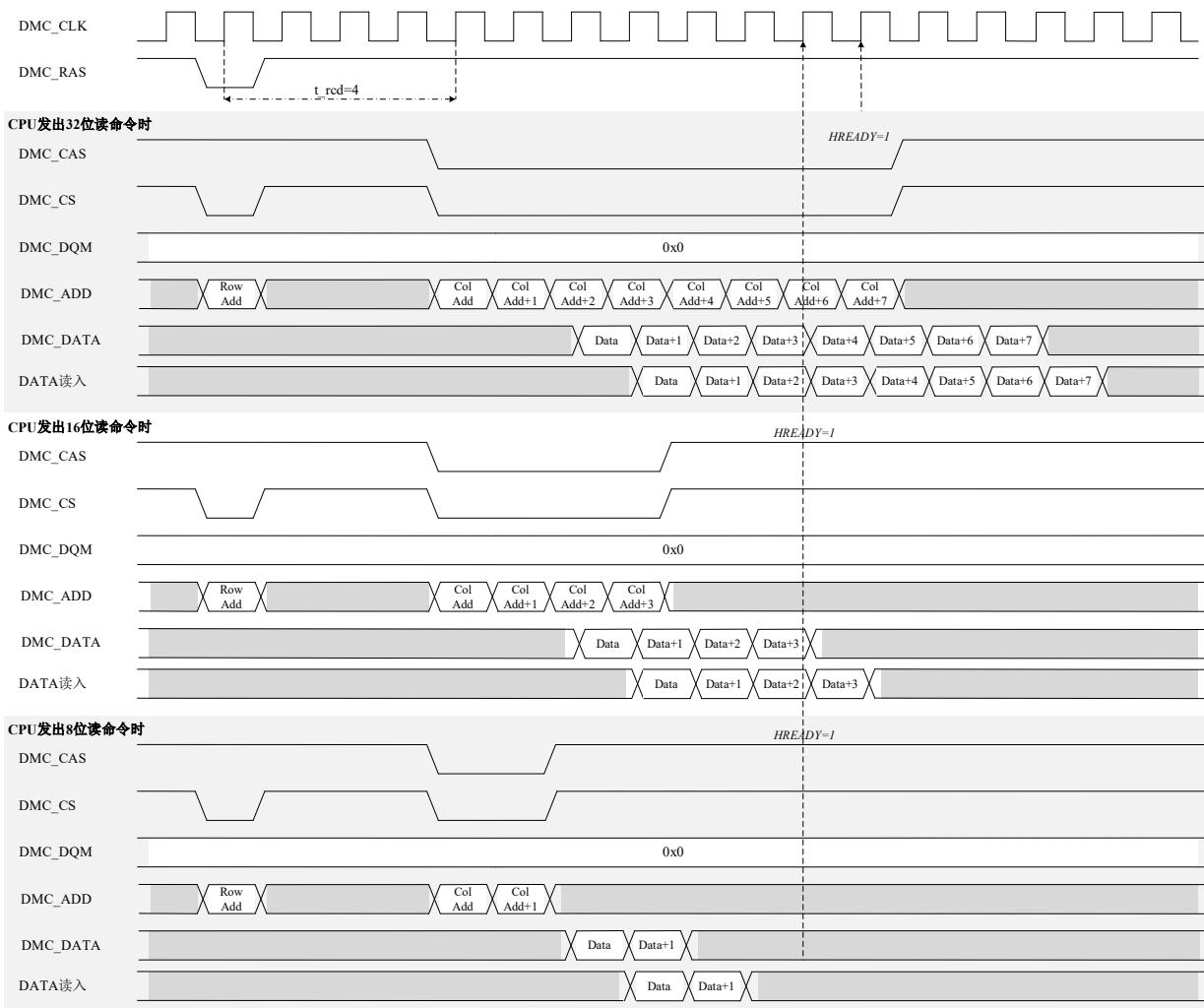


Figure 40-30 Basic timing of single read operation (16-bit width (DMCMW=00))

b) Figure 40-31 is the basic timing diagram of a single write action.

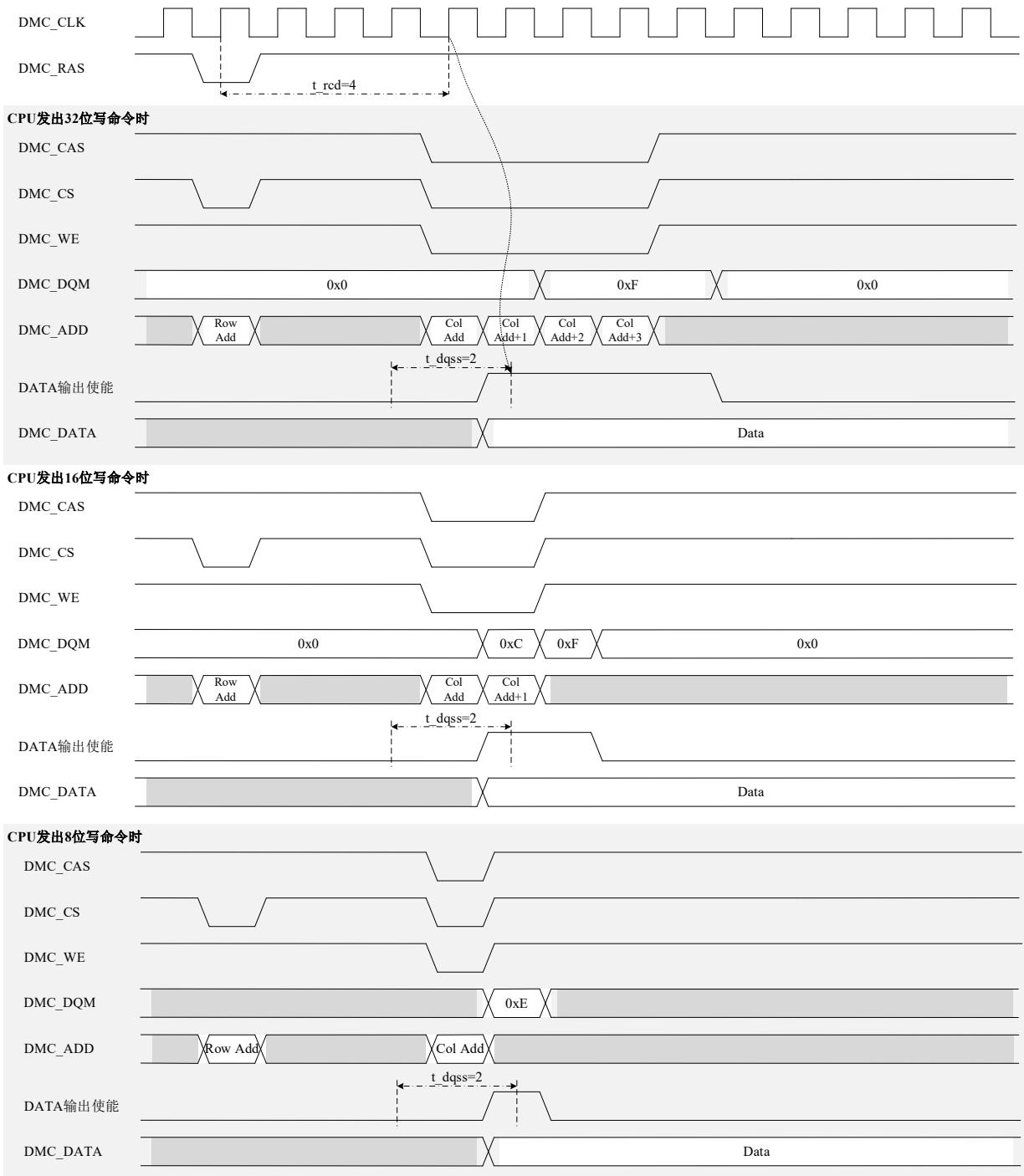


Figure 40-31 Basic timing of single write operation (32-bit width (DMCMW=01))

c) Figure 40-32 is the basic timing diagram of burst read operation.

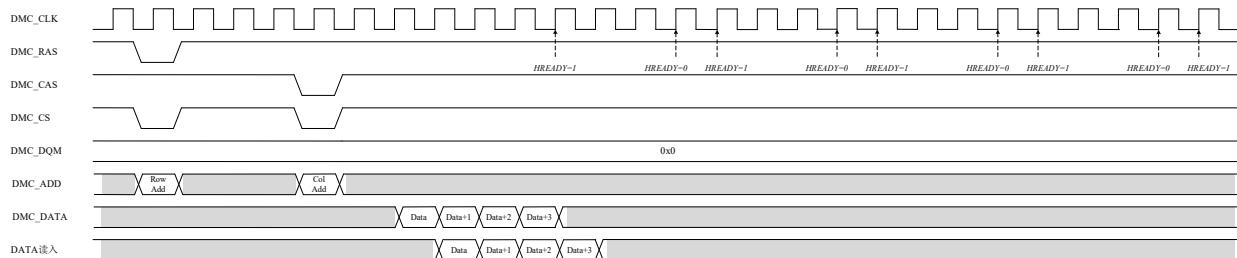


Figure 40-32 Burst read operation basic timing (32-bit width (DMCMW=10) & BURST=010)

d) Figure 40-33 is the basic timing diagram of burst write operation.

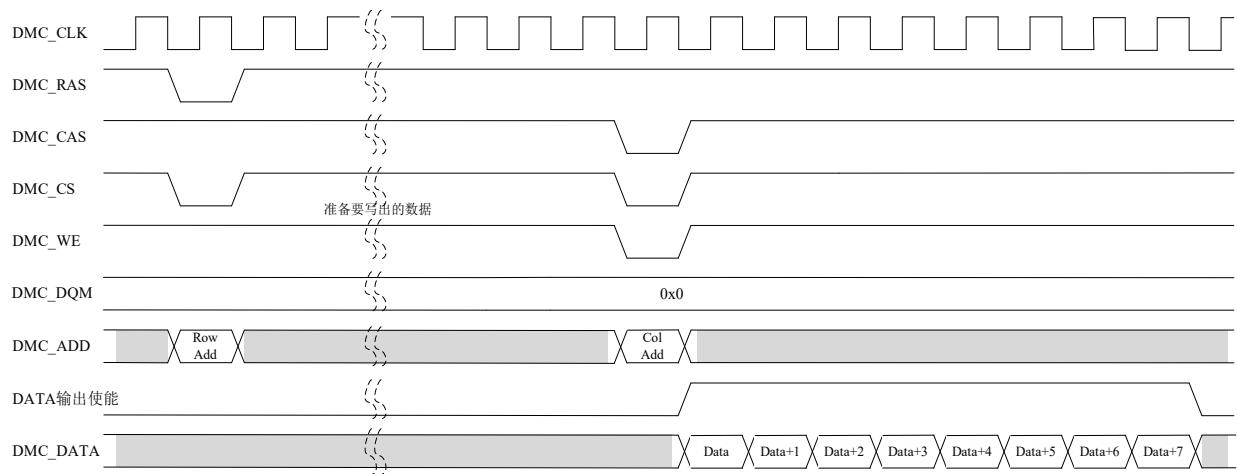


Figure 40-33 Burst write operation basic timing (32-bit width (DMCMW=10) & BURST=011)

FIFO management

There are 3 types of FIFOs in the DMC. They are command FIFO, write data FIFO, and read data FIFO.

Command FIFO: Two command FIFOs are used to cache two DMC commands and corresponding command addresses, CS and other information.

Write data FIFO: 10 write data FIFOs for buffering AHB write data.

Read data FIFO: 10 read data FIFOs for buffering data read in from SDRAM.

40.3.2.5 DMC low power management

The DMC can switch between the Ready state and the Low Power state to realize the power consumption management within the DMC; at the same time, it can control the output clock or clock enable of the DMC in the idle state to realize the power consumption management of the external SDRAM memory.

Low Power state control

When the DMC is in the Config state, after issuing the Go command, the Pause command and the Sleep command in turn through the state control register (DMC_STCR), the DMC enters the Low Power state. After entering the Low Power state, the internal operation of the DMC stops and the power consumption is reduced. In the Low Power state, after issuing the Wakeup command and the Go command in turn through the state control register (DMC_STCR), the DMC exits from the Low Power state and enters the Ready state to start normal operations.

DMC_CLK output control

When the CKSTOP bit of the CHIP configuration register (DMC_CPCR) is set to 1, when the DMC does not operate on the external SDRAM, the clock output of the DMC_CLK port will automatically stop; when there is a new operation, the clock of the DMC_CLK will be restarted. After the operation is completed , DMC_CLK automatically stops, thus realizing the output control of the clock DMC_CLK in the idle state, and then achieving the purpose of controlling the power consumption of DMC and SDRAM.

DMC_CKE output control

When the CKEDIS bit of the CHIP configuration register (DMC_CPCR) is set to 1, after the DMC performs an operation on the external SDRAM, after the setting cycle of DMC_CPCR.CKEDIISPRD[5:0], the DMC becomes an idle state, and the port of DMC_CKE outputs It becomes invalid; when there is a new operation, DMC_CKE will become valid again. After the action is completed, DMC_CKE will become invalid again, thus realizing the control of enabling DMC_CKE for the clock output in the idle state, thereby achieving the control of SDRAM function. consumption purpose. Figure 40-34 is the timing diagram of DMC_CKE port output control.

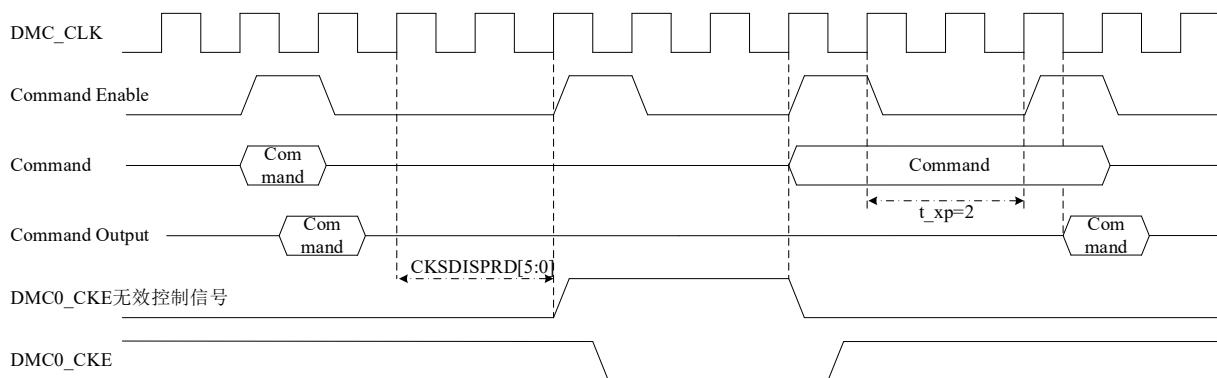


Figure 40-34 DMC_CKE port output control timing

40.3.3 NFC-NAND Flash Controller

40.3.3.1 Introduction to NAND Flash

NAND Flash memory is a kind of Flash memory, which adopts a nonlinear macrocell mode inside, which provides a cheap and effective solution for the realization of solid-state large-capacity memory. NAND Flash memory has the advantages of large capacity and fast rewriting speed. It is suitable for the storage of large amounts of data, so it has been widely used in the industry. For example, embedded products include digital cameras, MP3 Walkman memory cards, compact size U disk, etc.

40.3.3.2 NFC basic functions

The basic functional characteristics of NFC are as follows:

- Support ONFI protocol, can easily realize access to SLC and MLC NAND Flash devices
- Supports devices with page size 2KB/4KB/8KB
- Support 8bit and 16bit wide devices
- Supports 1bit Hamming code ECC error correction in units of 512bytes
- Able to calculate 4bit BCH code per 512byte, and give syndrome when ECC error occurs
- Access timing is configurable through registers

40.3.3.3 NFC access commands

The following table lists the ONFI access commands supported by NFC. Where "must" indicates a command that all ONFI devices must support. "Optional" indicates a command that some ONFI devices can support

Table 40-14 ONFI access command

Order	Required (M)/Optional (O)	first cycle (command)	Second cycle (confirm)
Read	M	00h	30h
Copyback read	O	00h	35h
Change read column	M	05h	E0h
Change read column enhanced	O	06h	E0h
Read cache random	O	00h	31h
Read cache sequential	O	31h	
Read cache end	O	3Fh	
Block erase	M	60h	D0h
Block erase interleaved	O	60h	D1h
Read status	M	70h	
Read status enhanced	O	78h	
Page program	M	80h	10h
Page program interleaved	O	80h	11h
Page cache program	O	80h	15h
Copyback program	O	85h	10h
Copyback program interleaved	O	85h	11h
Change write column	M	85h	
Change row address	O	85h	
Read ID	M	90h	
Read parameter page	M	ECh	
Read Unique ID	O	EDh	
Get Features	O	EEh	
Set Features	O	EFh	
Reset LUN	O	FAh	

40.3.3.4 NFC access action

Page Read

1. Write 0x00000000 to the command register
2. Write the NAND Flash address to the index register
3. Write 0x000000E0 to the command register, or perform a read operation to the data register. At this time, the NAND Flash device will pull down the RB signal, indicating that the device is preparing data at this time. When the RB signal returns to high, it indicates that the data is ready. Therefore, if you use the command register to write 0x000000E0, you need to wait until the RB signal is high and then read the data register. When using a read operation to the data register, the controller will insert a wait on the bus until RB is high, and return to the first a data
4. Get the data of the entire page by reading the data register
5. The user does not need to read all the data of the page. After reading the required data, write 0x00000023 to the command register, the controller will automatically read the remaining data and start calculating ECC. When the ECC calculation is complete, an interrupt will be generated and the flag bit will be set, and the ECC result will be written to the ECC result register. If the user selects the ECC function to be invalid, this step will be omitted
6. Write 0x000000FE to the command register to end the current read operation and set CE to invalid

Page Write

1. Write 0x00000080 to the command register
2. Write the NAND Flash address to the index register
3. Write data to the data register in turn. When the ECC is valid, the controller will calculate the ECC while writing the data.
4. After writing all the data, write 0x00000010 to the command register, the NAND Flash device will start programming and pull down RB at the same time. When the ECC is valid, if the data written by the user is less than 1 page, the controller will automatically make up the remaining address with FF and calculate the ECC value. And after writing the data, the ECC value is automatically written into the spare area of the NAND Flash device
5. After programming, the NAND Flash device returns RB to high. Write 0x000000FE to the command register to end the write operation and deassert CE

Block Erase

1. Write 0x81000M60 to the command register, where M is the bank number
2. Write 0x40000MAB to the command register, where M is the bank number and AB is the lowest byte of the NAND Flash row address
3. Write 0x40000MWX to the command register, where M is the BANK number and WX is the

second-lowest byte of the NAND Flash row address

4. Write 0x40000MYZ to the command register, where M is the bank number and YZ is the highest byte of the NAND Flash row address
5. Write 0x00000MD0 to the command register, where M is the bank number
6. Wait for the NAND Flash device to return RB to high

Reset

1. Write 0x00000MFF to the command register, where M is the bank number
2. Wait for the NAND Flash device to return RB to high

Read ID

1. Write 0x81000M90 to the command register, where M is the bank number
2. Write 0x40000MAB to the command register, where M is the bank number and AB is the address of the ID
3. Read the data register to get the ID
4. Writing 0x000000FE to the command register deasserts CE

Read Status

1. Write 0x81000M70 to the command register, where M is the bank number
2. Read the data register, the lowest byte of the register value is the status value
3. Writing 0x000000FE to the command register deasserts CE

40.4 Interrupt Description

NFC has three types of interrupt events. The interrupt enable register (NFC_IENR) controls whether the corresponding interrupt occurs. The user can judge whether an interrupt event occurs by querying the interrupt status register (NFC_ISTR), and clear the interrupt event by writing the register. The 3 types of interrupt events of NFC are as follows:

1. Interrupt at end of ECC calculation
2. Interrupt on ECC check error
3. Each RB signal generates a rising edge, that is, an interrupt occurs when the corresponding device is ready

40.5 Register description

SMC: 0x88000000h DMC: 0x88000400h NFC: 0x88100000h

Table 40-15 EXMC register list

Register name	Symbol	Offset	Bit width	Reset value	
SMC	Enable register	SMC_ENAR	(0x4005540Ch)	32	0x00000000h
	Status register	SMC_STSR	0x0000h	32	0x00000001h
	Command register	SMC_CMDR	0x0010h	32	-
	Status Control Register 0	SMC_STCR0	0x0008h	32	-
	Status Control Register 1	SMC_STCR1	0x000ch	32	-
	Refresh time register	SMC_RFTR	0x0020h	32	0x00000000h
	Basic control register	SMC_BACR	0x0200h	32	0x00000300h
	Chip Select Control Register	SMC_CSCR0	0x0208h	32	0xFFFFFFFFh
	Chip Select Control Register	SMC_CSCR1	0x020Ch	32	0x00000000h
	CHIP Configuration Register	SMC_CPCR	0x0018h	32	-
	CHIP Status Register 0	SMC_CPSR0	0x0104h	32	0x00FF0A00h
	CHIP Status Register 1	SMC_CPSR1	0x0124h	32	0x00FF0A00h
	CHIP Status Register 2	SMC_CPSR2	0x0144h	32	0x00FF0A00h
	CHIP Status Register 3	SMC_CPSR3	0x0164h	32	0x00FF0A00h
	Timing Configuration Register	SMC_TMCR	0x0014h	32	-
	Timing Status Register 0	SMC_TMSR0	0x0100h	32	0x001263CCh
	Timing Status Register 1	SMC_TMSR1	0x0120h	32	0x001263CCh
	Timing Status Register 2	SMC_TMSR2	0x0140h	32	0x001263CCh
	Timing Status Register 3	SMC_TMSR3	0x0160h	32	0x001263CCh
DMC	Enable register	DMC_ENAR	(0x4005540Ch)	32	0x00000000h
	Status register	DMC_STSR	0x0000h	32	0x00000700h
	Command register	DMC_CMDR	0x0008h	32	-
	Status control register	DMC_STCR	0x0004h	32	-
	Refresh time register	DMC_RFTR	0x0010h	32	0x00000A60h
	Basic control register	DMC_BACR	0x0300h	32	0x00000300h
	Chip Select Control Register 0	DMC_CSCR0	0x0200h	32	0x0000FF00h
	Chip Select Control Register 1	DMC_CSCR1	0x0204h	32	0x0000FF00h
	Chip Select Control Register 2	DMC_CSCR2	0x0208h	32	0x0000FF00h
	Chip Select Control Register 3	DMC_CSCR3	0x020ch	32	0x0000FF00h
	CHIP Configuration Register	DMC_CPCR	0x000ch	32	0x00020040h
	Timing Configuration Register	DMC_TMCR_t_casl	0x0014h	32	0x00000003h

Register name	Symbol	Offset	Bit width	Reset value	
Timing Configuration Register	DMC_TMCR_t_dqss	0x0018h	32	0x00000001h	
	DMC_TMCR_t_mrd	0x001ch	32	0x00000002h	
	DMC_TMCR_t_ras	0x0020h	32	0x00000007h	
	DMC_TMCR_t_rc	0x0024h	32	0x0000000Bh	
	DMC_TMCR_t_rcd	0x0028h	32	0x00000035h	
	DMC_TMCR_t_rfc	0x002ch	32	0x00001012h	
	DMC_TMCR_t_rp	0x0030h	32	0x00000035h	
	DMC_TMCR_t_rrd	0x0034h	32	0x00000002h	
	DMC_TMCR_t_wr	0x0038h	32	0x00000003h	
	DMC_TMCR_t_wtr	0x003ch	32	0x00000002h	
	DMC_TMCR_t_xp	0x0040h	32	0x00000001h	
	DMC_TMCR_t_xsr	0x0044h	32	0x0000000Ah	
	DMC_TMCR_t_esr	0x0048h	32	0x00000014h	
NFC	Enable register	NFC_ENAR	(0x4005540Ch)	32	0x00000000h
	Status register	NFC_STSR	(0x40055428h)	32	0x00000000h
	Control register	NFC_STCR	(0x40055408h)	32	0x00000000h
	Data register	NFC_DATR	0x0000~0x7FFFh	32	Indefinite
	Command register	NFC_CMDR	0x8000h	32	0x00000000h
	Index register 0	NFC_IDXR0	0x8004h	32	0x00000000h
	Index register 1	NFC_IDXR1	0x8048h	32	0x00000000h
	Basic configuration registers	NFC_BACR	0x8054h	32	0x00002187h
	Interrupt enable register	NFC_IENR	0x8030h	32	0x00000000h
	Interrupt status register	NFC_ISTR	0x8034h	32	0x00000000h
	Interrupt result register	NFC_IRSR	0x8038h	32	0x00000000h
	Timing Configuration Register 0	NFC_TMCR0	0x804Ch	32	0x03030202h
	Timing Configuration Register 1	NFC_TMCR1	0x8050h	32	0x28080303h
	Timing Configuration Register 2	NFC_TMCR2	0x805Ch	32	0x03050D03h
	ECC check register	NFC_ECCR	0x8060~0x817Fh	32	Indefinite

In addition, NFC has a system control register and a status register

Register name	Symbol	Address	Bit width	Reset value
NFC System Control Register	NFC_SYCTLREG	0x40055408	32	0x00000000h
NFC System Status Register	NFC_SYSTATREG	0x40055428	32	0x00000000h

40.5.1 SMC-SRAM/PSRAM/NOR Flash register

40.5.1.1 Enable Register (SMC_ENAR)

Reset value: 0x00000000h

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														SMCEN	-

Bit	Marking	Place name	Function	Read and write
b31~b2	Reserved	-	Read as "0", write as "0"	R/W
b1	SMCEN	SMC enable	0: The function port of the SMC module is invalid 1: The function port of the SMC module is valid	R/W
b0	Reserved	-	Read as "0", write as "0"	R/W

40.5.1.2 Status Register (SMC_STSR)

Reset value: 0x00000001h

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														STATUS	

Bit	Marking	Place name	Function	Read and write
b31~b1	Reserved	-	Reset value when read	R
b0	STATUS	SMC current status	0: Ready 1: Low Power	R

40.5.1.3 Command Register (SMC_CMDR)

Reset value: -

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved				CMDCHIP[2:0]			CMD[1:0]		CRES		CMDADD[19:0]				
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CMDADD[15:0] MODE[2:0]															
Bit	Marking	Place name	Function	Read and write											
b31~b26	Reserved	-	Write "0" on write	W											
b25~b23	CMDCHIP[1:0]	CS object setting	Chip object for CMD commands 000: Chip 0 001: Chip 1 010: Chip 2 011: Chip 3 Please do not set other values	W											
b22~b21	CMD[1:0]	command input	00: Invalid 01: MdRegConfig 10: UpdateRegs 11: MdRegConfig & UpdateRegs	W											
b20	CRES	CRE port settings	0: SMC_CRE output low level 1: When the MdregConfig command is issued, the SMC_CRE port outputs a high level	W											
b19~b0	CMDADD[19:0]	Address Object Settings	When CMD=01 or 11, CMDADD[19:0] indicates the external MEM address to be accessed	W											

40.5.1.4 Status Control Register (SMC_STCRO)

Reset value: -

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															

Bit	Marking	Place name	Function	Read and write
b31~b3	Reserved	-	Write "0" on write	W
b2	LPWIR	Low Power state entry request	0: no request 1: Enter Low Power state	W
b1~b0	Reserved	-	Write "0" on write	W

40.5.1.5 Status Control Register (SMC_STCR1)

Reset value: -

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															

Bit	Marking	Place name	Function	Read and write
b31~b3	Reserved	-	Write "0" on write	W
b2	LPWOR	Low Power state exit request	0: no request 1: Exit from Low Power state	W
b1~b0	Reserved	-	Write "0" on write	W

40.5.1.6 Refresh Time Register (SMC_RFTR)

Reset value: 0x00000000h

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															
Bit	Marking	Place name	Function				Read and write								
b31~b4	Reserved	-	Read as "0", write as "0"				R/W								
b3~b0	REFPRD[3:0]	refresh cycle	Refresh cycle value setting				R/W								

40.5.1.7 Basic Control Register (SMC_BACR)

Reset value: 0x00000300h

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
CKSEL[1:0] Reserved MUXMD Reserved															
Bit	Marking	Place name	Function				Read and write								
b31~b16	Reserved	-	Read as "0", write as "0"				R/W								
b15~b14	CKSEL[1:0]	Sample clock selection	00: When reading latched SMC_DATA, SMC_WAIT port data, use internal EXCLK 01: When reading the latched SMC_DATA, SMC_WAIT port data, use the internal EXCLK inversion as the clock 10: When reading the latched SMC_DATA, SMC_WAIT port data, use the EXMC_CLK port status feedback as the clock 11: Prohibitions Note: EXCLK clock configuration frequency must not exceed 40MHz				R/W								
b13~b10	Reserved	-	Read as "0", write as "0"				R/W								
b9~b8	Reserved	-	Read as "1", write as "1"				R/W								
b7~b5	Reserved	-	Read as "0", write as "0"				R/W								
b4	MUXMD	SMC address data multiplexing	0: SMC address and data are not multiplexed 1: SMC address and data multiplexing				R/W								
b3~b0	Reserved	-	Read as "0", write as "0"				R/W								

40.5.1.8 Chip Select Control Register (SMC_CSCR0)

Reset value: 0xFFFFFFFFh

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
ADDMSK3[7:0]								ADDMSK2[7:0]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADDMSK1[7:0]								ADDMSK0[7:0]							
<hr/>															
Bit	Marking		Place name			Function					Read and write				
b31~b24	ADDMSK3[7:0]		CS3 space mask address			CS3 space mask address setting					R/W				
b23~b16	ADDMSK2[7:0]		CS2 space mask address			CS2 space mask address setting					R/W				
b15~b8	ADDMSK1[7:0]		CS1 space mask address			CS1 space mask address setting					R/W				
b7~b0	ADDMSK0[7:0]		CS0 space mask address			CS0 space mask address setting					R/W				

Note: ADDMSKx refers to the configuration of the address space of the DMC, that is, DMC_CSCRx.

40.5.1.9 Chip Select Control Register (SMC_CSCR1)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
ADDMAT3[7:0]								ADDMAT2[7:0]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADDMAT1[7:0]								ADDMAT0[7:0]							
<hr/>															
Bit	Marking		Place name			Function					Read and write				
b31~b24	ADDMAT3[7:0]		CS3 space matching address			CS3 space matching address setting					R/W				
b23~b16	ADDMAT2[7:0]		CS2 space matching address			CS2 space matching address setting					R/W				
b15~b8	ADDMAT1[7:0]		CS1 space matching address			CS1 space matching address setting					R/W				
b7~b0	ADDMAT0[7:0]		CS0 space matching address			CS0 space matching address setting					R/W				

Note: ADDMATx refers to the configuration of the address space of DMC, that is, DMC_CSCRx.

40.5.1.10 CHIP Configuration Register (SMC_CPCR)

Reset value: -

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
-	BLSS	ADVS	BAAS	MW[1:0]		WBL[2:0]		WSYN		RBL[2:0]		RSYN			
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Write "0" on write	W											
b15~b13	Reserved	-	Write "0" on write	W											
b12	BLSS	BLS port settings	0: The port state of the BLS is synchronized with the state change of the CS port 1: The port state of the BLS is synchronized with the state change of the WE port	W											
b11	ADVS	ADV port settings	0: ADV port function is invalid 1: When the address data line multiplexing function is valid, the ADV port function is valid	W											
b10	BAAS	BAA port settings	0: BAA port function is invalid 1: BAA port function is valid	W											
b9~b8	MW[1:0]	Memory bit width selection	01: 16-bit width 10: 32 bit width Please do not set other values	W											
b7~b5	WBL[2:0]	Burst write data length	000: 1 write transfer 001: 4 consecutive write transfers 010: 8 consecutive write transfers 011: 16 consecutive write transfers 100: 32 consecutive write transfers 101: Continuous write, unlimited times Please do not set other values	W											
b4	WSYN	write synchronization	0: Asynchronous write enable 1: Synchronous write enable	W											
b3~b1	RBL[2:0]	Burst read data length	000: 1 read transfer 001: 4 consecutive read transfers 010: 8 consecutive read transfers 011: 16 consecutive read transfers 100: 32 consecutive read transfers 101: Continuous reading, unlimited times Please do not set other values	W											
b0	RSYN	read synchronization	0: Asynchronous read enable 1: Synchronous read enable	W											

40.5.1.11 CHIP Status Register (SMC_CPSR0~3)

Reset value: 0xFF000A00h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
ADDMAT[7:0]								ADDMSK[7:0]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	BLSS	ADVS	BAAS	MW[1:0]		WBL[2:0]		WSYN	RBL[2:0]		RSYN				
Bit	Marking		Place name			Function						Read and write			
b31~b24	ADDMAT[7:0]		CS space matching address			match address read						R			
b23~b16	ADDMSK[7:0]		CS space mask address			Mask address read						R			
b15~b13	-		-			Read to 0						R			
b12	BLSS		SMC_BLS status			SMC_BLS status readout						R			
b11	ADVS		SMC_ADV status			SMC_ADV status readout						R			
b10	BAAS		SMC_BAA status			SMC_BAA status readout						R			
b9~b8	MW[1:0]		Memory bit width status			Memory bit width status readout						R			
b7~b5	WBL[2:0]		write data length status			Write data length status read						R			
b4	WSYN		write action mode status			Write Action Mode Status Read						R			
b3~b1	RBL[2:0]		read data length status			read data length status read						R			
b0	RSYN		Read action mode status			Read action mode status readout						R			

40.5.1.12 Timing Configuration Register (SMC_TMCR)

Reset value: -

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16			
Reserved								t_tr[2:0]		-	t_pc[2:0]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0			
<hr/>																		
-	t_wp[2:0]		-	t_ceoe[2:0]		t_wc[3:0]			t_rc[3:0]									
Bit	Marking	Place name			Function						Read and write							
b31~b23	Reserved	-			Write "0" on write						W							
b22~b20	t_tr[2:0]	t_tr period setting			t_tr setting						W							
b19	Reserved	-			Write "0" on write						W							
b18~b16	t_pc[2:0]	t_pc cycle setting			t_pc setting						W							
b15	Reserved	-			Write "0" on write						W							
b14~b12	t_wp[2:0]	t_wp cycle setting			t_wp setting						W							
b11	Reserved	-			Write "0" on write						W							
b10~b8	t_ceoe[2:0]	t_ceoe cycle setting			t_ceoe settings						W							
b7~b4	t_wc[3:0]	t_wc cycle setting			t_wc setting						W							
b3~b0	t_rc[3:0]	t_rc cycle setting			t_rc setting						W							

40.5.1.13 Timing Status Register (SMC_TMSR0~3)

Reset value: 0x001263CCh

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16			
Reserved										t_tr[2:0]	-	t_pc[2:0]						
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0			
<hr/>																		
-	t_wp[2:0]		-	t_ceoe[2:0]			t_wc[3:0]			t_rc[3:0]								
<hr/>																		
Bit	Marking	Place name			Function					Read and write								
b31~b23	Reserved	-			Read as "0"					R								
b22~b20	t_tr[2:0]	t_tr cycle status			t_tr value readout					R								
b19	Reserved	-			Read as "0"					R								
b18~b16	t_pc[2:0]	t_pc cycle status			t_pc value readout					R								
b15	Reserved	-			Read as "0"					R								
b14~b12	t_wp[2:0]	t_wp cycle status			t_wp value readout					R								
b11	Reserved	-			Read as "0"					R								
b10~b8	t_ceoe[2:0]	t_ceoe cycle status			t_ceoe value readout					R								
b7~b4	t_wc[3:0]	t_wc cycle status			t_wc value readout					R								
b3~b0	t_rc[3:0]	t_rc cycle status			t_rc value readout					R								

40.5.2 DMC-SDRAM register

40.5.2.1 Enable Register (DMC_ENAR)

Reset value: 0x00000000h

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															DMCEN
Bit	Marking	Place name	Function				Read and write								
b31~b1	Reserved	-	Read as "0", write as "0"				R/W								
b0	DMCEN	DMC enabled	0: The function port of the DMC module is invalid 1: The functional port of the DMC module is valid				R/W								

40.5.2.2 Status Register (DMC_STSR)

Reset value: 0x00000700h

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															MEMW[1:0]
Bit	Marking	Place name	Function				Read and write								
b31~b4	Reserved	-	Reset value when read				R								
b3~b2	MEMW[1:0]	DMC MEM bit width	00: 16 bits 01: 32 bits Reading other values is invalid Note: The read value of this bit is the same as the set value of DMC_BACR.DMCMW[1:0]				R								
b1~b0	STATUS[1:0]	DMC current status	00: Config 01: Ready 10: Paused 11: Low Power				R								

40.5.2.3 Command Register (DMC_CMDR)

Reset value: -

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved										CMDCHIP[1:0]	CMD[1:0]	CMDBA[1:0]			
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-											CMDADD[13:0]			
Bit	Marking	Place name	Function	Read and write											
b31~b22	Reserved	-	Write "0" on write	W											
b21~b20	CMDCHIP[1:0]	CS object setting	Chip object for CMD commands 00: Chip 0 01: Chip 1 10: Chip 2 11: Chip 3	W											
b19~b18	CMD[1:0]	command input	00: PrechargeAll 01: AutoRefresh 10: MdRegConfig 11: NOP	W											
b17~b16	CMDBA[1:0]	BANK object setting	The Bank object specified by the CMD (ModeConfig) command 00: Bank 0 01: Bank 1 10: Bank 2 11: Bank 3	W											
b15~b14	Reserved	-	Write "0" on write	W											
b13~b0	CMDADD[13:0]	Address Object Settings	The Address object specified by the CMD (ModeConfig) command	W											

40.5.2.4 Status Control Register (DMC_STCR)

Reset value: -

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															
Bit	Marking	Place name	Function	Read and write											
b31~b3	Reserved	-	Write "0" on write	W											
b2~b0	STCTL[2:0]	state switch control	000: Go 001: Sleep 010: Wakeup 011: Pause 100: Configure Please do not set other values	W											

40.5.2.5 Refresh Time Register (DMC_RFTR)

Reset value: 0x00000A60h

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
-															
Bit	Marking	Place name	Function	Read and write											
b31~b15	Reserved	-	Read as "0", write as "0"	R/W											
b14~b0	REFPRD[14:0]	refresh cycle	Refresh cycle value setting	R/W											

40.5.2.6 Basic Control Register (DMC_BACR)

Reset value: 0x00000300h

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
CKSEL[1:0]															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	Read as "0", write as "0"	R/W											
b15~b14	CKSEL[1:0]	Sample clock selection	00: Use internal EXCLK when reading latched DMC_DATA port data 01: When reading the latched DMC_DATA port data, use the internal EXCLK inversion as the clock 10: When reading the latched DMC_DATA port data, use the EXMC_CLK port status feedback as the clock 11: Prohibitions Note: EXCLK clock configuration frequency must not exceed 40MHz	R/W											
b13~b10	Reserved	-	Read as "0", write as "0"	R/W											
b9~b8	Reserved	-	Read as "1", write as "1"	R/W											
b7~b2	Reserved	-	Read as "0", write as "0"	R/W											
b1~b0	DMCMW[1:0]	Memory bit width selection	00: 16 bits wide 01: 32 bit width Please do not set other values	R/W											

40.5.2.7 Chip Select Control Register (DMC_CSCR0~3)

Reset value: 0x0000FF00h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16								
Reserved															BRCx								
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0								
ADDMATx[7:0]								ADDMSKx[7:0]															
<hr/>																							
Bit	Marking	Place name	Function	Read and write																			
b31~b17	Reserved	-	Read as "0"	R																			
b16	BRCx	Address decoding method	0: Row, Bank, Column method 1: Bank, Row, Column method Note: x=0~3	R/W																			
b15~b8	ADDMATx[7:0]	Address matching	address to match Note: x=0~3	R/W																			
b7~b0	ADDMSKx[7:0]	address masking	The corresponding address to be blocked Note: x=0~3	R/W																			

Note: Taking ADDMAT0 and ADDMSK0 as examples, the corresponding manner of the address space of the DMC is described. Other CS space settings are similar to CS0 settings.

Table 40-16 CS0 setting and access address correspondence table

ADDMAT0[7:0]	ADDMSK0[7:0]	Corresponding external address	CS0 space size
86h	FFh	86000000h~86FFFFFFh	16Mb
	FEh	86000000h~87FFFFFFh	32Mb
	Other settings prohibited	-	-
84h	FFh	84000000h~84FFFFFFh	16Mb
	FEh	84000000h~85FFFFFFh	32Mb
	FCh	84000000h~87FFFFFFh	64Mb
	Other settings prohibited	-	-
82h	FFh	82000000h~82FFFFFFh	16Mb
	FEh	82000000h~83FFFFFFh	32Mb
	Other settings prohibited	-	-
80h	FFh	80000000h~80FFFFFFh	16Mb
	FEh	80000000h~81FFFFFFh	32Mb
	FCh	80000000h~83FFFFFFh	64Mb
	F8h	80000000h~87FFFFFFh	128Mb
	Other settings prohibited	-	-

40.5.2.8 CHIP Configuration Register (DMC_CPCR)

Reset value: 0x00020040h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	ACTCP[1:0]	-	-	-	-	-	-	BURST[2:0]		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
		CKEDISPRD[5:0]		CKSTOP	CKEDIS	APBS	ROWBS[2:0]	-		COLBS[2:0]					
Bit	Marking	Place name	Function												Read and write
b31~b26	Reserved	-	Read as "0", write as "0"												R/W
b25~b24	ACTCP[1:0]	Automatically refresh the number of chips	00: Chip 0 auto refresh is valid 01: Chip 0~1 auto refresh is valid 10: Chip 0~2 auto refresh is valid 11: Chip 0~3 auto refresh is valid												R/W
b23~b19	Reserved	-	Read as "0", write as "0"												R/W
b18~b16	BURST[2:0]	burst transfer length	000: 1 teleport 001: 2 consecutive transmissions 010: 4 consecutive transfers 011: 8 consecutive transfers 100:16 consecutive transmissions Please do not set other values Note: The setting of this bit must be consistent with the corresponding bit setting of the Mode Register of the external Memory												R/W
b15~b10	CKEDISPRD[5:0]	clock output off period	Clock output OFF wait period setting												R/W
b9	CKSTOP	Clock stop setting	0: MEM clock output normally 1: MEM clock stops when there is no operation												R/W
b8	CKEDIS	CKE output settings	0: CKE output valid (H) 1: When the command FIFO is empty, after the period set by CKEDISPRD, the CKE output becomes invalid (L)												R/W
b7	APBS	Auto precharge bit	0: Bit 10 of the address line 1: Bit 8 of the address line												R/W
b6~b4	ROWBS[2:0]	row address width	000: Row address width is 11 bits 001: Row address width is 12 bits 010: Row address width is 13 bits 011: Row address width is 14 bits 100: Row address width is 15 bits 101: Row address width is 16 bits Please do not set other values												R/W
b3	Reserved	-	Read as "0", write as "0"												R/W
b2~b0	COLBS[2:0]	Column address width	000: Column address width is 8 bits 001: Column address width is 9 bits 010: Column address width is 10 bits 011: Column address width is 11 bits 100: Column address width is 12 bits Please do not set other values												R/W

40.5.2.9 Timing Configuration Register (DMC_TMCR_t_casl)

Reset value: 0x00000003h

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															
Bit	Marking	Place name	Function	Read and write											
b31~b3	Reserved	-	Read as "0", write as "0"	R/W											
b2~b0	t_casl[2:0]	t_casl cycle setting	t_casl setting (CAS latency time)	R/W											

40.5.2.10 Timing Configuration Register (DMC_TMCR_t_dqss)

Reset value: 0x00000001h

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															
Bit	Marking	Place name	Function	Read and write											
b31~b2	Reserved	-	Read as "0", write as "0"	R/W											
b1~b0	t_dqss[1:0]	t_dqss period setting	t_dqss setting	R/W											

40.5.2.11 Timing Configuration Register (DMC_TMCR_t_mrd)

Reset value: 0x00000002h

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															
Bit	Marking	Place name	Function	Read and write											
b31~b7	Reserved	-	Read as "0", write as "0"	R/W											
b6~b0	t_mrd[6:0]	t_mrd period setting	t_mrd setting (mode register command time)	R/W											

40.5.2.12 Timing Configuration Register (DMC_TMCR_t_ras)

Reset value: 0x00000007h

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												t_ras[3:0]			
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b4	Reserved	-	Read as "0", write as "0"	R/W											
b3~b0	t_ras[3:0]	t_ras cycle setting	t_ras setting (RAS to precharge delay time)	R/W											

40.5.2.13 Timing Configuration Register (DMC_TMCR_t_rc)

Reset value: 0x0000000Bh

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												t_rc[3:0]			
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b4	Reserved	-	Read as "0", write as "0"	R/W											
b3~b0	t_rc[3:0]	t_rc cycle setting	t_rc setting (Active bank x to Active bank x delay time)	R/W											

40.5.2.14 Timing Configuration Register (DMC_TMCR_t_rcd)

Reset value: 0x00000035h

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								$t_{rcd_p[2:0]}$			-	$t_{rcd_b[2:0]}$						
Bit	Marking		Place name			Function								Read and write				
b31~b7	Reserved		-			Read as "0", write as "0"								R/W				
b6~b4	$t_{rcd_p[2:0]}$		Additional t_{rcd} period setting			Additional t_{rcd} settings								R/W				
b3	Reserved		-			Read as "0", write as "0"								R/W				
b2~b0	$t_{rcd_b[2:0]}$		Basic t_{rcd} period setting			Basic t_{rcd} settings t_{rcd} (RAS to CAS minimum delay time) is defined as follows: When $t_{rcd_b-t_{rcd_p}} \geq 3$, $t_{rcd} = t_{rcd_b}$; when $t_{rcd_b-t_{rcd_p}} < 3$, $t_{rcd} = t_{rcd_p} + 3$								R/W				

40.5.2.15 Timing Configuration Register (DMC_TMCR_t_rfc)

Reset value: 0x00001012h

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			
-		$t_{rfc_p[4:0]}$					-			$t_{rfc_b[4:0]}$								
Bit	Marking		Place name			Function								Read and write				
b31~b13	Reserved		-			Read as "0", write as "0"								R/W				
b12~b8	$t_{rfc_p[4:0]}$		Additional t_{rfc} cycle setting			Additional t_{rfc} settings								R/W				
b7~b5	Reserved		-			Read as "0", write as "0"								R/W				
b4~b0	$t_{rfc_b[4:0]}$		Basic t_{rfc} period setting			Basic t_{rfc} settings t_{rfc} (autorefresh command time) is defined as follows: When $t_{rfc_b-t_{rfc_p}} \geq 3$, $t_{rfc} = t_{rfc_b}$; when $t_{rfc_b-t_{rfc_p}} < 3$, $t_{rfc} = t_{rfc_p} + 3$								R/W				

40.5.2.16 Timing Configuration Register (DMC_TMCR_t_rp)

Reset value: 0x00000035h

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								$t_{rp_p}[2:0]$				-	$t_{rp_b}[2:0]$		
Bit	Marking	Place name	Function	Read and write											
b31~b7	Reserved	-	Read as "0", write as "0"	R/W											
b6~b4	$t_{rp_p}[2:0]$	Additional t_{rp} period setting	Additional t_{rp} settings	R/W											
b3	Reserved	-	Read as "0", write as "0"	R/W											
b2~b0	$t_{rp_b}[2:0]$	Basic t_{rp} period setting	Basic t_{rp} settings t_{rp} (precharge to RAS delay time) is defined as follows: When $t_{rp_b} - t_{rp_p} \geq 3$, $t_{rp} = t_{rp_b}$; when $t_{rp_b} - t_{rp_p} < 3$, $t_{rp} = t_{rp_p} + 3$	R/W											

40.5.2.17 Timing Configuration Register (DMC_TMCR_t_rrd)

Reset value: 0x00000002h

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								$t_{rrd}[3:0]$							
Bit	Marking	Place name	Function	Read and write											
b31~b4	Reserved	-	Read as "0", write as "0"	R/W											
b3~b0	$t_{rrd}[3:0]$	t_{rrd} period setting	t_{rrd} setting (Active bank x to Active bank y delay time)	R/W											

40.5.2.18 Timing Configuration Register (DMC_TMCR_t_wr)

Reset value: 0x00000003h

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												$t_{wr}[2:0]$			
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b3	Reserved	-	Read as "0", write as "0"	R/W											
b2~b0	$t_{wr}[2:0]$	t_{wr} cycle setting	t_{wr} setting (write to precharge delay time)	R/W											

40.5.2.19 Timing Configuration Register (DMC_TMCR_t_wtr)

Reset value: 0x00000002h

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												$t_{wtr}[2:0]$			
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b3	Reserved	-	Read as "0", write as "0"	R/W											
b2~b0	$t_{wtr}[2:0]$	t_{wtr} period setting	t_{wtr} setting (write to read delay time)	R/W											

40.5.2.20 Timing Configuration Register (DMC_TMCR_t_xp)

Reset value: 0x00000001h

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								$t_{xp}[7:0]$							
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b8	Reserved	-	Read as "0", write as "0"	R/W											
b7~b0	$t_{xp}[7:0]$	t_{xp} cycle setting	t_{xp} setting (exit power-down command time)	R/W											

40.5.2.21 Timing Configuration Register (DMC_TMCR_t_xsr)

Reset value: 0x0000000Ah

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								t_xsr[7:0]							
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b8	Reserved	-	Read as "0", write as "0"	R/W											
b7~b0	t_xsr[7:0]	t_xsr cycle setting	t_xsr setting (exit self-refresh command time)	R/W											

40.5.2.22 Timing Configuration Register (DMC_TMCR_t_esr)

Reset value: 0x00000014h

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								t_esr[7:0]							
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b8	Reserved	-	Read as "0", write as "0"	R/W											
b7~b0	t_esr[7:0]	t_esr period setting	t_esr setting (self-refresh command time)	R/W											

40.5.3 NFC-NAND Flash register

40.5.3.1 Enable Register (NFC_ENAR)

Reset value: 0x00000000h

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												NFCEN	-	-	
Bit	Marking	Place name	Function	Read and write											
b31~b3	Reserved	-	Read as "0", write as "0"	R/W											
b2	NFCEN	NFC enabled	0: The function port of the NFC module is invalid 1: The functional port of the NFC module is valid	R/W											
b1~b0	Reserved	-	Read as "0", write as "0"	R/W											

40.5.3.2 Status Register (NFC_STSR)

Reset value: 0x00000000h

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								PECC	CHIP_BUSY[7:0]						
Bit	Marking	Place name	Function	Read and write											
b31~b9	Reserved	-	Read as "0", write as "0"	R/W											
b8	PECC	ECC status	0: NFC does not perform ECC calculation 1: NFC is doing ECC calculation	R											
b7~b0	CHIP_BUSY	device status	Indicates the status of BANK0~7 devices 0: The device is busy 1: Device ready	R											

40.5.3.3 Control Register (NFC_STCR)

Reset value: 0x00000000h

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															OPENP
Bit	Marking	Place name	Function	Read and write											
b31~b1	Reserved	-	Read as "0", write as "0"	R/W											
b0	OPENP	open page enable	0: Do not allow NFC to automatically send a read command after reset 1: Allow NFC to automatically send a read command after reset	R/W											

40.5.3.4 Data Register (NFC_DATR)

Reset value: 0xFFFFFFFFh

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]															
Bit	Marking	Place name	Function	Read and write											
b31~b0	DATA	data	The data register is used to read and write data on the Flash device. The address of reading and writing data is determined by the address written to the index register	R/W											

40.5.3.5 Command Register (NFC_CMDR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
ARG[23:8]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ARG[7:0]								CMD[7:0]							
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b8	ARG	Parameter	Used to adjust the command, seeTable 40-18	R/W											
b7~b0	CMD	Order	Commands sent to the device, seeTable 40-18	R/W											

The following table lists the setting methods of all commands and parameters:

Table 40-17 List of command parameters

Command (CMD)	Parameter (ARG)	Function
00h	000000h	Read command
10h	000000h	Page program confirm
11h	000000h	A confirm inserted after the end of the first page of the two-plane page program
15h	000000h	Second command in Page cache program
23h	000000h	Read ECC status. After writing this command, NFC will read ECC data from the extra area of NAND Flash and start ECC calculation
60h	810000h~810007h	Block erase command, the parameter is used to indicate the operation object BANK0~BANK7
70h	810000h~810007h	Read status command, the parameter is used to indicate the operation object BANK0~BANK7
80h	000000h	Page program command
81h	000000h	For the second page of the two-plane page program to start writing
90h	000000h~000007h	Read ID command, the parameter is used to indicate the operation object BANK0~BANK7
D0h	810000h~810007h	Block erase confirm, the parameter is used to indicate the operation object BANK0~BANK7
D1h	810000h~810007h	Block erase interleaved confirm, the parameter is used to indicate the operation object BANK0~BANK7
E0h	000000h	Start reading data. It is used to instruct NFC to start reading data. After writing this command, NFC will open 1 page for data reading. At this time, CHIP_BUSY will be set. After CHIP_BUSY is reset, the data register can be read to obtain data.
FEh	000000h	After the read operation is completed, write this command to instruct the NFC to disable the chip select signal CE
FFh	000000h~000007h	Reset command, the parameter is used to indicate the operation object BANK0~BANK7
05h	810000h~810007h	The first cycle of the Change read column command
06h	810000h~810007h	The first cycle of the Change read column enhanced command
31h	830000h~830007h	The first cycle of the Read cache sequential command and the second cycle of the Read cache random command
32h	810000h~810007h	Copyback read interleaved command
35h	810000h~810007h	Copyback read command
3Fh	830000h~830007h	Read cache end
78h	800000h~800007h	Read status enhanced command
85h	810000h~810007h	Copyback program interleaved and Copyback program commands
85h	840000h~840007h	Change write column and Change row address
E0h	850000h~850007h	Second cycle of Change read column and Change read column enhanced commands
ECh	830000h~830007h	Read parameter page command
EDh	830000h~830007h	Read Unique ID Command
EEh	830000h~830007h	Get Features Command
EFh	830000h~830007h	Set Features Command
FAh	820000h~820007h	Reset LUN Command
FCh	820000h~820007h	Asynchronous reset command

40.5.3.6 Index Register (NFC_IDXR0)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
IDX0[31:16]															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
IDX0[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b0	IDX0	index	The lower 32 bits of the access address and the upper 8 bits are composed of the IDX0 register	R/W

40.5.3.7 Index Register (NFC_IDXR1)

Reset value: 0x00000000h

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								IDX1[7:0]							
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b8	Reserved	-	Read as "0"	R/W											
b7~b0	IDX1	index	The upper 8 bits of the access address, and the lower 32 bits are composed of the IDXRO register	R/W											

IDXR0 and IDXRO together form a 40-bit address when accessing the NAND Flash device. The relationship between the value written in the register and the address of the actual device depends on the capacity of the device. The following table lists the address composition when accessing devices of various capacities:

Table 40-18 Index register value and MEM address correspondence table

Capacity	2K page			4K page			8K page		
	page	column	bank	page	column	bank	page	column	bank
512Mbit	26:12	11:0	29:27	26:13	12:0	29:27	26:14	13:0	29:27
1Gbit	27:12	11:0	30:28	27:13	12:0	30:28	27:14	13:0	30:28
2Gbit	28:12	11:0	31:29	28:13	12:0	31:29	28:14	13:0	31:29
4Gbit	29:12	11:0	32:30	29:13	12:0	32:30	29:14	13:0	32:30
8Gbit	30:12	11:0	33:31	30:13	12:0	33:31	30:14	13:0	33:31
16Gbit	31:12	11:0	34:32	31:13	12:0	34:32	31:14	13:0	34:32
32Gbit	32:12	11:0	35:33	32:13	12:0	35:33	32:14	13:0	35:33
64Gbit	33:12	11:0	36:34	33:13	12:0	36:34	33:14	13:0	36:34

40.5.3.8 Basic Configuration Register (NFC_BACR)

Reset value: 0x00002187h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved								SCS[7:0]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	RAC	ECCM[1:0]	WP	PAGE[1:0]	BANK[1:0]	-	-	16BIT						
<hr/>															
Bit	Marking	Place name	Function												Read and write
b31~b24	Reserved	-	Read as "0", write as "0"												R/W
b23~b16	SCS	spare column size	Set the size of spare column size, in word												R/W
b15~b14	Reserved	-	Read as "0", write as "0"												R/W
b13	RAC	row address cycle	0: 2 row address cycles 1:3 row address cycles												R/W
b12~b11	ECCM	ECC mode	00: 1bitECC mode 01: 4bitECC mode other: set ban												R/W
b10	WP	write protection	0: Put the device into write-protected state 1: Put the device in an unprotected state												R/W
b9~b8	PAGE	page size	00: Prohibitions 01: 2K 10: 4K 11: 8K												R/W
b7~b6	BANK	bank(CE) number	00: 1 bank 01: 2bank 10:4bank 11: 8bank												R/W
b5~b4	Reserved	-	Read as "0", write as "0"												R/W
b3	16BIT	data bit width	0: 8bit 1: 16bit												R/W
b2~b0	SIZE	Equipment capacity	000: 16Gbit 001: 32Gbit 010: 64Gbit 011: 512Mbit 100: 1Gbit 101: 2Gbit 110: 4Gbit 111: 8Gbit												R/W

40.5.3.9 Interrupt Enable Register (NFC_IENR)

Reset value: 0x00000000h

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
RBEN[7:0]								ECC DIS	ECCE EN	-	ECCC EN	-	-	ECCE CEN	ECCE UEN
Bit	Marking	Place name	Function										Read and write		
b31~b16	Reserved	-	Read as "0", write as "0"										R/W		
b15~b8	RBEN	ready interrupt enable	0: Interrupts are not allowed when the access device is ready (RB signal is set) 1: Enable interrupt generation when the access device is ready (RB signal is set)										R/W		
b7	ECCDIS	ECC disabled	0: Enable ECC function 1: Disable ECC function										R/W		
b6	ECCEEN	ECC Error Interrupt Enable	0: No interrupt is generated when an ECC error occurs 1: An interrupt is generated when an ECC error occurs										R/W		
b5	Reserved	-	Read as "0", write as "0"										R/W		
b4	ECCCEN	ECC calculation complete interrupt enable	0: No interrupt is generated when the ECC calculation is completed 1: An interrupt is generated when the ECC calculation is completed										R/W		
b3~b2	Reserved	-	Read as "0", write as "0"										R/W		
b1	ECCECEN	Correctable ECC Error Interrupt Enable	0: No interrupt is generated when a correctable ECC error occurs in 1bitECC mode 1: Interrupt when a correctable ECC error occurs in 1bitECC mode										R/W		
b0	ECCEUEN	Uncorrectable ECC Error Interrupt Enable	0: No interrupt is generated when an uncorrectable ECC error occurs in 1bitECC mode 1: Interrupt when uncorrectable ECC error occurs in 1bitECC mode										R/W		

40.5.3.10 Interrupt Status Register (NFC_ISTR)

Reset value: 0x00000000h

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
RBST[7:0]								-	ECCE ST	-	ECCC ST	-	-	ECCE CST	ECCC UST
Bit	Marking	Place name	Function										Read and write		
b31~b16	Reserved	-	Read as "0", write as "0"										R/W		
b15~b8	RBST	ready interrupt status	Set when the access device is ready (RB signal is set) Note: This bit is reset when a 0 is written, and no effect when a 1 is written										R/W		
b7	Reserved	-	Read as "0", write as "0"										R/W		
b6	ECCEST	ECC error status	Set when a 4bit ECC error occurs Note: This bit is reset when a 0 is written, and no effect when a 1 is written										R/W		
b5	Reserved	-	Read as "0", write as "0"										R/W		
b4	ECCCST	ECC calculation completion status	Set when ECC calculation is complete Note: This bit is reset when a 0 is written, and no effect when a 1 is written										R/W		
b3~b2	Reserved	-	Read as "0", write as "0"										R/W		
b1	ECCECST	Correctable ECC Error Status	Set when a correctable ECC error occurs in 1bitECC mode Note: This bit is reset when a 0 is written, and no effect when a 1 is written										R/W		
b0	ECCEUST	Uncorrectable ECC Error Status	Set when an uncorrectable ECC error occurs in 1bitECC mode Note: This bit is reset when a 0 is written, and no effect when a 1 is written										R/W		

40.5.3.11 Interrupt Result Register (NFC_IRSR)

Reset value: 0x00000000h

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
RBRS[7:0]								-	ECCE RS	-	ECCC RS	-	-	ECCE CRS	ECCC URS
Bit	Marking	Place name	Function										Read and write		
b31~b16	Reserved	-	Read as "0", write as "0"										R/W		
b15~b8	RBRS	ready interrupt	This bit is the AND logic result of NFC_ISTR.RBST and NFC_IENR.RBEN										R		
b7	Reserved	-	Read as "0", write as "0"										R/W		
b6	ECCERS	ECC error interrupt	This bit is the AND logic result of NFC_ISTR.ECCEST and NFC_IENR.ECCEEN										R		
b5	Reserved	-	Read as "0", write as "0"										R/W		
b4	ECCCRS	ECC calculation complete interrupt	This bit is the AND logic result of NFC_ISTR.ECCCST and NFC_IENR.ECCCEN										R		
b3~b2	Reserved	-	Read as "0", write as "0"										R/W		
b1	ECCECRS	Correctable ECC Error Interrupt	This bit is the AND logic result of NFC_ISTR.ECCECST and NFC_IENR.ECCECEN										R		
b0	ECCEURS	Uncorrectable ECC error interrupt	This bit is the AND logic result of NFC_ISTR.ECCEUST and NFC_IENR.ECCEUEN										R		

40.5.3.12 Timing Configuration Register (NFC_TMCRO)

Reset value: 0x03030202h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TH[7:0]								TRP[7:0]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TWP[7:0]								TS[7:0]							
Bit	Marking	Place name	Function										Read and write		
b31~b24	TH	-	Set the hold time of CLE/ALE/CE after WE/RE change										R/W		
b23~b16	TRP	-	Sets the valid (low) pulse width of RE										R/W		
b15~b8	TWP	-	Sets the valid (low) pulse width of WE										R/W		
b7~b0	TS	-	Set CLE/ALE/CE settling time before WE/RE change										R/W		

40.5.3.13 Timing Configuration Register (NFC_TMCR1)

Reset value: 0x28080303h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TWB[7:0]								TRR[7:0]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TRH[7:0]								TWH[7:0]							
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b24	TWB	-	Set the time from the rising edge of WE to the falling edge of RB	R/W											
b23~b16	TRR	-	Set the time from the rising edge of RB to the falling edge of RE	R/W											
b15~b8	TRH	-	Set the invalid (high) pulse width of WE	R/W											
b7~b0	TWH	-	Set the invalid (high) pulse width of WE	R/W											

The parameters set by timing configuration register 0 and timing configuration register 1 are shown in the following figure:

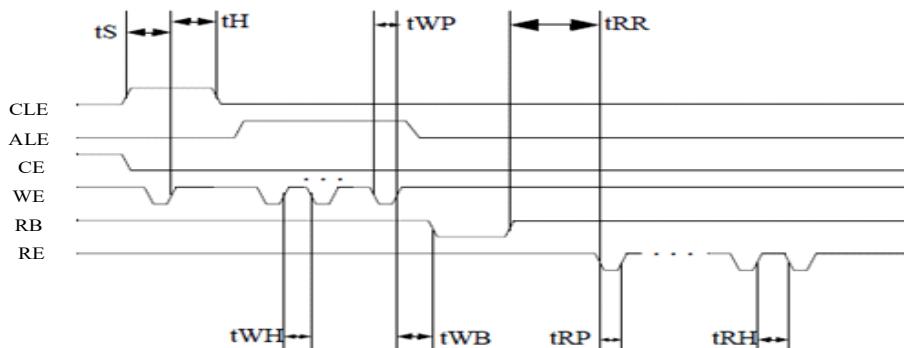


Figure 40-35 NFC timing control diagram

40.5.3.14 Timing Configuration Register (NFC_TMCR2)

Reset value: 0x03050D03h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TADL[7:0]								TRTW[7:0]							

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

Bit	Marking	Place name	Function	Read and write
b31~b24	TADL	-	Set the time from ALE to read and write data	R/W
b23~b16	TRTW	-	Set the time from the rising edge of RE to the falling edge of WE	R/W
b15~b8	TWTR	-	Set the time from the rising edge of WE to the falling edge of RE	R/W
b7~b0	TCCS	-	Set the delay time after Change read column and Change write column commands	R/W

40.5.3.15 ECC Check Register (NFC_ECCR)

1) In 1bitECC mode, ECCR0~15 (Add Offset: 0x8060~0x809C) as ECC result register:

Reset value: 0x00000000h

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
-	-	ME	SE	ERRLOC[11:0]											
Bit	Marking		Place name		Function								Read and write		
b31~b14	Reserved		-		Read as "0", write as "0"								R/W		
b13	ME		multiple error		0: No multi-bit ECC error detected 1: Multi-bit ECC error detected								R		
b12	SE		one-bit error		0: No one-bit ECC error detected 1: One ECC error detected								R		
b11~b0	ERRLOC		wrong location		Indicates the wrong position, only meaningful when SE is set Bit 2:0 represents the bit position in the error byte: 000 means bit0 error, 111 means bit7 error 11:3 bits indicate the position of the error byte in 512byte: 000h represents the first byte, 1FFh represents the 512th byte								R		

- 2) In 4bitECC mode, ECCR7~71 (Add Offset: 0x807C~0x817C) as ECC status register and accompanying register, as shown in the following table:

Table 40-19 ECCR register list in 4bitECC mode

Register name	Symbol	Offset	Bit width	Reset value
ECC Status Register	ECC_STAT	0x807C	32	0x00000000h
Syndrome register 0	ECC_SYND0	0x8080~0x808C	32	0x00000000h
Syndrome register 1	ECC_SYND1	0x8090~0x809C	32	0x00000000h
Syndrome register 2	ECC_SYND2	0x80A0~0x80AC	32	0x00000000h
Syndrome register 3	ECC_SYND3	0x80B0~0x80BC	32	0x00000000h
Syndrome register 4	ECC_SYND4	0x80C0~0x80CC	32	0x00000000h
Syndrome register 5	ECC_SYND5	0x80D0~0x80DC	32	0x00000000h
Syndrome register 6	ECC_SYND6	0x80E0~0x80EC	32	0x00000000h
Syndrome register 7	ECC_SYND7	0x80F0~0x80FC	32	0x00000000h
Syndrome Register 8	ECC_SYND8	0x8100~0x810C	32	0x00000000h
Syndrome register 9	ECC_SYND9	0x8110~0x811C	32	0x00000000h
Syndrome register 10	ECC_SYND10	0x8120~0x812C	32	0x00000000h
Syndrome register 11	ECC_SYND11	0x8130~0x813C	32	0x00000000h
Syndrome register 12	ECC_SYND12	0x8140~0x814C	32	0x00000000h
Syndrome register 13	ECC_SYND13	0x8150~0x815C	32	0x00000000h
Syndrome register 14	ECC_SYND14	0x8160~0x816C	32	0x00000000h
Syndrome register 15	ECC_SYND15	0x8170~0x817C	32	0x00000000h

ECC Status Register (ECC_STAT)

Reset value: 0x00000000h

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
ECCSEC[15:0]															
<hr/>															
Bit	Marking		Place name		Function						Read and write				
b31~b16	Reserved		-		Read as "0", write as "0"						R/W				
b15~0	ECCSEC		wrong location		0: No ECC error occurs in the corresponding section 1: An ECC error occurs in the corresponding section						R				

ECC Syndrome Register (ECC_SYND)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved		S2[12:0]													

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved		S1[12:0]													

Bit	Marking	Place name	Function	Read and write
b31~b29	Reserved	-	Read as "0", write as "0"	R/W
b28~16	S2	S2	α_2 Syndrome coefficient	R
b15~b13	Reserved	-	Read as "0", write as "0"	R/W
b12~0	S1	S1	α Syndrome coefficient	R

ECC Syndrome Register (ECC_SYND)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved		S4[12:0]													

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved		S3[12:0]													

Bit	Marking	Place name	Function	Read and write
b31~b29	Reserved	-	Read as "0", write as "0"	R/W
b28~16	S4	S4	α_4 Syndrome coefficient	R
b15~b13	Reserved	-	Read as "0", write as "0"	R/W
b12~0	S3	S3	α_3 Syndrome coefficient	R

ECC Syndrome Register (ECC_SYND)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved		S6[12:0]													

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved		S5[12:0]													

Bit	Marking	Place name	Function	Read and write
b31~b29	Reserved	-	Read as "0", write as "0"	R/W
b28~16	S6	S6	α_6 Syndrome coefficient	R
b15~b13	Reserved	-	Read as "0", write as "0"	R/W
b12~0	S5	S5	α_5 Syndrome coefficient	R

ECC Syndrome Register (ECC_SYND)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved		S8[12:0]													

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved		S7[12:0]													

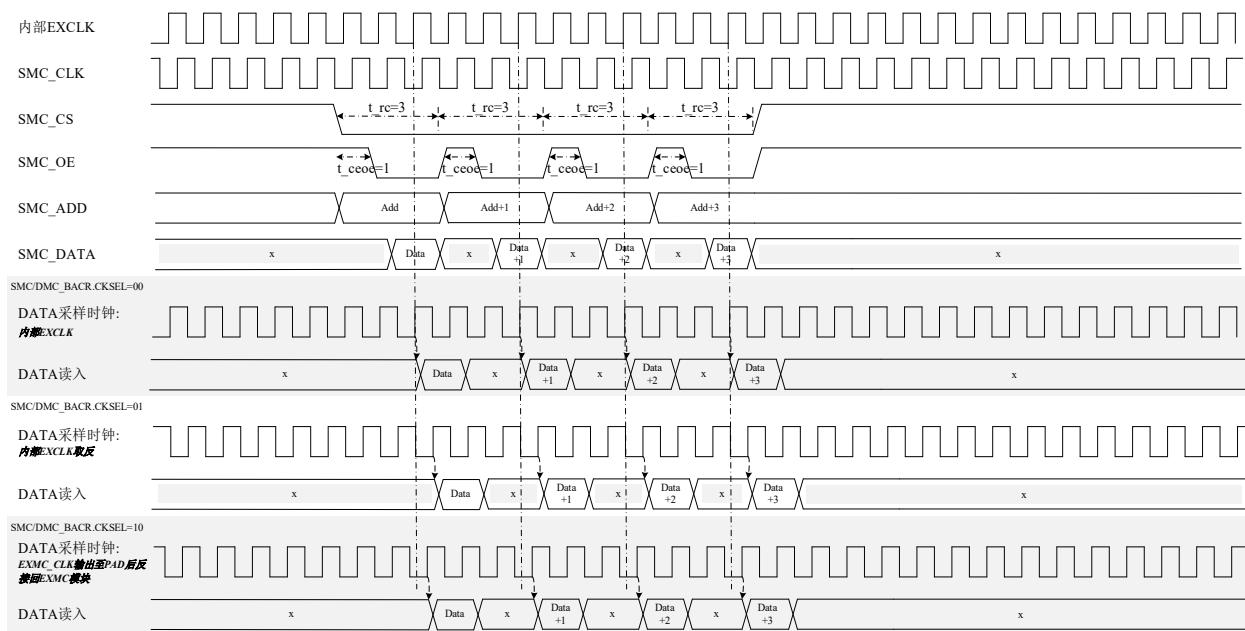
Bit	Marking	Place name	Function	Read and write
b31~b29	Reserved	-	Read as "0", write as "0"	R/W
b28~16	S8	S8	α_8 Syndrome coefficient	R
b15~b13	Reserved	-	Read as "0", write as "0"	R/W
b12~0	S7	S7	α_7 Syndrome coefficient	R

40.6 Precautions for use

The related registers of the SMC can only be set in the Ready state; the related registers of the DMC can only be set in the Config or Low Power state.

When using the SMC and DMC modules to read the external Memory, the Memory Device transmits the data to the controller after receiving the read command from the controller, and the controller samples the transmitted data through SMC_BACR.CKSEL or SMC_BACR.CKSEL or DMC_BACR.CKSEL to select, please refer to the register chapter for specific settings. Choosing a different sampling clock can improve the time margin of the communication path.

The following figure is an example of SMC access in synchronous mode.



41 Digital Video Interface (DVP)

41.1 Introduction

The Digital Camera Interface (DVP) is a synchronous parallel interface that captures 8-, 10-, 12- or 14-bit high-speed data streams from an external CMOS camera module. Software synchronization and hardware synchronization are supported. Supports acquisition frequency control and window clipping control of data streams. Supports data stream capture in different formats such as monochrome or raw Bayer format/YCbCr4:2:2/RGB565 progressive video and compressed data (JPEG).

41.2 System block diagram

The basic functions and features of DVP are shown in the following table.

Table 41-1 DVP basic functions and features

Basic functions	<ul style="list-style-type: none">• 8-bit, 10-bit, 12-bit, 14-bit parallel interface• Single frame mode and continuous mode• Software, hardware line sync and frame sync• Frame acquisition frequency control• window cropping• FIFO control
	<ul style="list-style-type: none">• Monochrome or raw Bayer format
	<ul style="list-style-type: none">• YCbCr4:2:2 progressive video
	<ul style="list-style-type: none">• RGB565 progressive video
	<ul style="list-style-type: none">• JPEG compressed data
	<ul style="list-style-type: none">• Frame start interrupts and events• End of frame interrupts and events• Line start breaks and events• End-of-line interrupts and events• Software Synchronization Error Interrupts and Events• FIFO overflow error interrupts and events

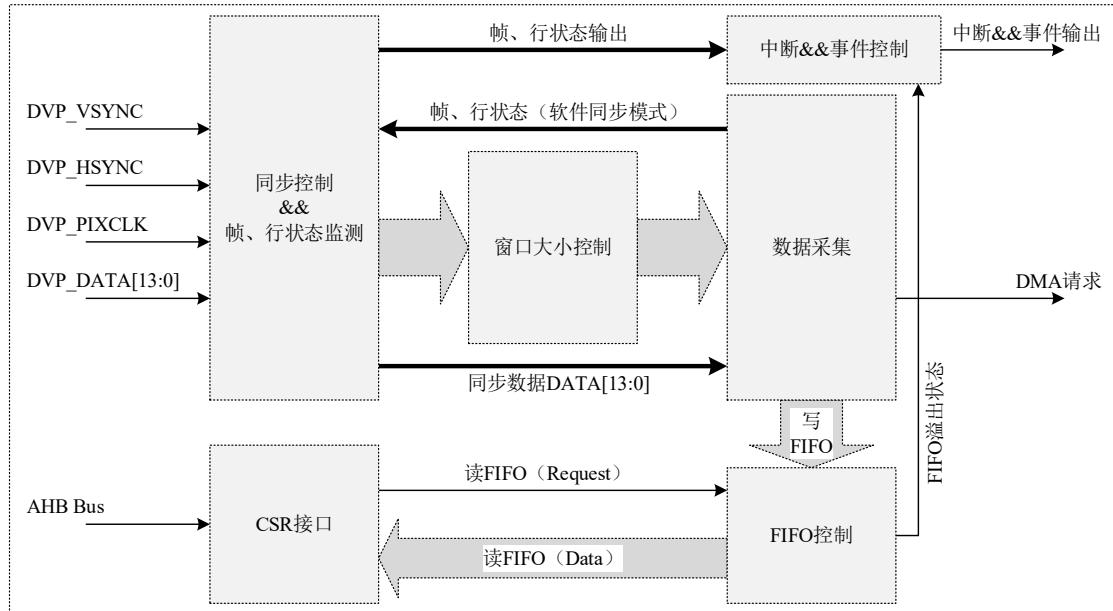


Figure 41-1 DVP basic block diagram

The table below shows the list of input ports for DVP.

Table 41-2 DVP port list

Port name	Direction	Function
DVP_VSYNC	in	Frame sync input port
DVP_HSYNC	in	Line sync input port
DVP_PIXCLK	in	clock input port
DVP_DATA[13:0]	in	data input port

41.3 Functional description

41.3.1 Video data format

41.3.1.1 Monochrome format

- 8 bits per pixel.

The following table shows how the data is stored.

Table 41-3 Single-color format video data storage method

byte address	31:24	23:16	15:8	7:0
0x00	n+3	n+2	n+1	n
0x04	n+7	n+6	n+5	n+4

41.3.1.2 YCbCr format

- A buffer alternately stores Y, Cb, and Cr: CbYCrYCbYCr. . .
- The DVP module supports YCbCr4:2:2 format data transmission, and the pixel components include Y (luminance), Cb and Cr (blue chrominance and red chrominance). Each component is encoded in 8 bits, and the luma and chrominance are (alternately) stored together.

The following table shows how the data is stored.

Table 41-4 YCbCr format video data storage method

byte address	31:24	23:16	15:8	7:0
0x00	Y n+1	Cr n	Y n	Cbn
0x04	Y n+3	Cr n+2	Y n+2	Cb n+2

41.3.1.3 RGB565 format

- A buffer alternately stores RGB signals, BGRBGRBGR. . .
- DVP module supports 16BPP (16 bits per pixel): RGB565 (2 pixels per 32 bits), 24BPP (palletized format) and grayscale formats are not supported.

The following table shows how the data is stored.

Table 41-5 RGB565 format video data storage method

byte address	31:27	26:21	20:16	15:11	10:5	4:0
0x00	R n+1	G n+1	B n+1	R n	G n	B n
0x04	R n+3	G n+3	B n+3	R n+3	G n+3	B n+3

41.3.1.4 JPEG format

- JPEG images are not stored in lines and frames, the DVP_VSYNC signal is used to start the acquisition process, and DVP_HSYNC is used as the data enable signal.
- The number of bytes contained in the valid range of DVP_HSYNC may not be a multiple of 4.
- DVP module supports JPEG format data transmission (DVP_CTR.JPEGEN=1).

41.3.2 Parallel port storage format

DVP interface consists of 11, 13, 15, 17 input signals. Only slave mode is supported. Depending on the setting of the BITSEL[1:0] bits in the DVP_CTR register, 8-bit, 10-bit, 12-bit or 14-bit data can be collected.

The data DVP_DATA output by the camera module is synchronized with the pixel clock DVP_PIXCLK, and changes on the rising/falling edge of the pixel clock according to the polarity of the pixel clock; the DVP_VSYNC signal indicates the start/end of the frame; the DVP_HSYNC signal indicates the start/end of the line.

When DVP_VSYNC or DVP_HSYNC is valid, it indicates the synchronization interval and data transmission is invalid; when DVP_VSYNC and DVP_HSYNC are invalid, it indicates the data interval and data transmission is valid. As shown in the signal waveform diagram, DVP_DATA changes on the rising edge of DVP_PIXCLK (the DVP module is set to collect data DVP_DATA on the falling edge of DVP_PIXCLK), DVP_VSYNC or DVP_HSYNC is a synchronization interval when it is high, and data transmission is invalid.

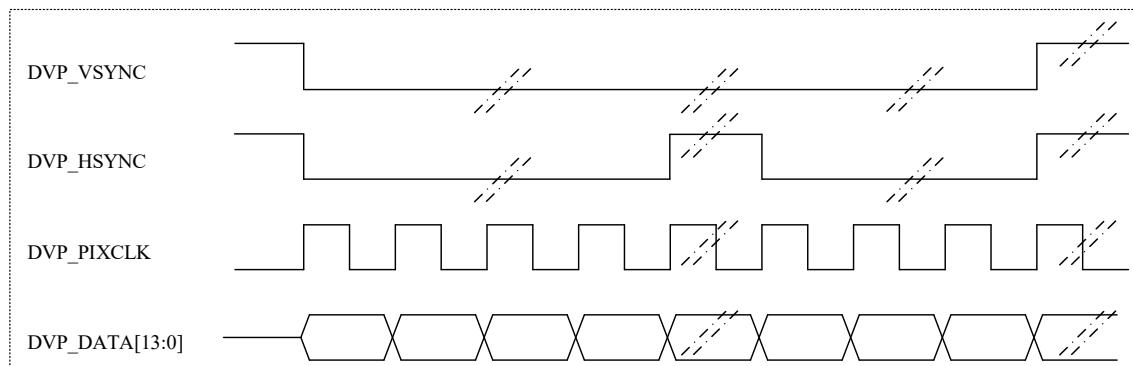


Figure 41-2 DVP signal waveform diagram

41.3.2.1 8Bit data

When DVP_CTR.BITSEL[1:0]=00, the DVP interface will collect the 8-bit data of its input DVP_DATA[7:0] and store it as 8-bit data, ignoring the input of DVP_DATA[13:8]. In this case, a 32-bit data word is generated every four pixel clock cycles. The following table lists the location of the captured data bytes in two 32-bit words.

Table 41-6 8bit data DVP storage mode

byte address	31:24	23:16	15:8	7:0
0x00	Dn+3[7:0]	Dn+2[7:0]	Dn+1[7:0]	Dn[7:0]
0x04	Dn+7[7:0]	Dn+6[7:0]	Dn+5[7:0]	Dn+4[7:0]

41.3.2.2 10Bit data

When DVP_CTR.BITSEL[1:0]=01, the DVP interface will collect the 10-bit data of its input DVP_DATA[9:0] and store it in the 10 least significant bits of the 16-bit word, ignoring DVP_DATA[13:10] input. At this time the remaining most significant bits (Bit15:10) in the DVP_DTR register will be cleared. In this case, a 32-bit data word is generated every two pixel clock cycles, as shown in the table below.

Table 41-7 10bit data DVP storage mode

byte address	31:26	25:16	15:10	9:0
0x00	0	Dn+1[9:0]	0	Dn[9:0]
0x04	0	Dn+3[9:0]	0	Dn+2[9:0]

41.3.2.3 12Bit data

When DVP_CTR.BITSEL[1:0]=10, the DVP interface will collect the 12-bit data of its input DVP_DATA[11:0] and store it in the 12 least significant bits of the 16-bit word, ignoring DVP_DATA[13:12] input. At this time the remaining most significant bits (Bit15:12) in the DVP_DTR register will be cleared. In this case, a 32-bit data word is generated every two pixel clock cycles, as shown in the table below.

Table 41-8 12bit data DVP storage mode

byte address	31:28	27:16	15:12	11:0
0x00	0	Dn+1[11:0]	0	Dn[11:0]
0x04	0	Dn+3[11:0]	0	Dn+2[11:0]

41.3.2.4 14Bit data

When DVP_CTR.BITSEL[1:0]=11, the DVP interface will capture the 14-bit data of its input DVP_DATA[13:0] and store it into the 14 least significant bits of a 16-bit word. The remaining most significant bits (Bit15:14) in the DVP_DTR register will be cleared at this time. In this case, a 32-bit data word is generated every two pixel clock cycles, as shown in the table below.

Table 41-9 14bit data DVP storage mode

byte address	31:30	29:16	15:14	13:0
0x00	0	Dn+1[13:0]	0	Dn[13:0]
0x04	0	Dn+3[13:0]	0	Dn+2[13:0]

41.3.3 Pattern selection

41.3.3.1 Single frame mode

In single frame mode (DVP_CTR.CAPMD=1), only a single frame is captured. After the DVP_CTR.CAPEN bit is set, the DVP interface will wait for the system to detect the start of a frame and then collect data. After receiving a complete frame of data, the CAPEN bit in DVP_CTR will be

automatically cleared, and the camera interface will no longer receive data. If the end-of-frame interrupt is enabled, a corresponding interrupt or event (DVP_FRAMEND) will be generated. As shown below.

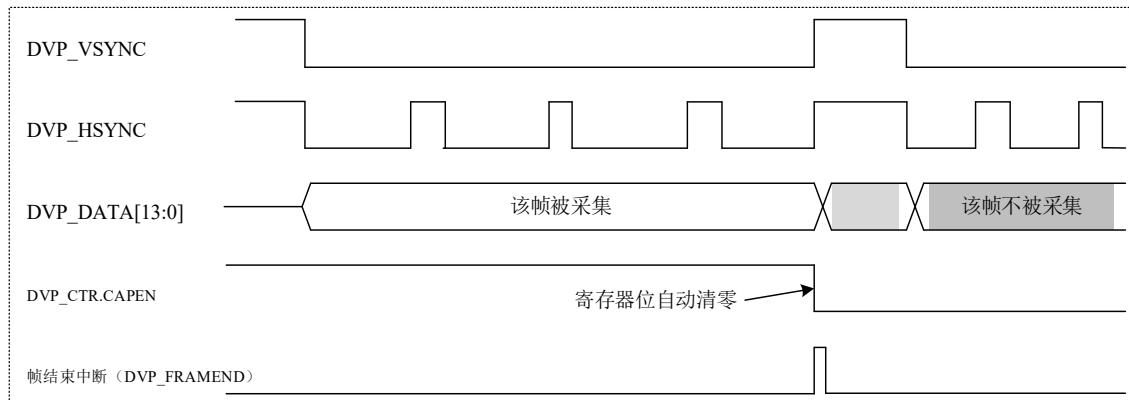


Figure 41-3 Single frame mode data acquisition action

41.3.3.2 Continuous mode

In continuous mode (DVP_CTR.CAPMD=0), image frames are continuously captured. After the DVP_CTR.CAPEN bit is set, the DVP interface will wait for the system to detect the start of a frame and then collect data. This process continues until the CAPEN bit in DVP_CTR is cleared. After the CPU sends a software clearing action to the DVP_CTR.CAPEN bit, the data acquisition process continues until the end of the current frame, and then the DVP_CTR.CAPEN bit is cleared. As shown below.

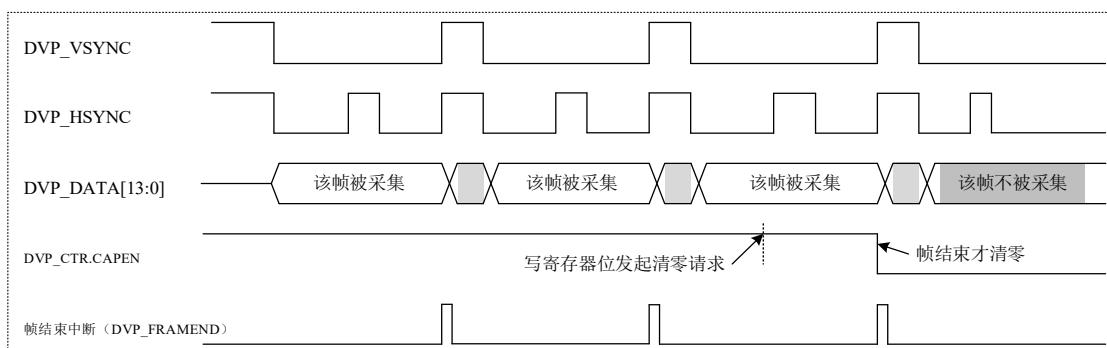


Figure 41-4 Continuous Mode Data Acquisition Action

In continuous mode, the image frame acquisition frequency can be controlled by setting the DVP_CTR.CAPFRC[1:0] bits to reduce the bandwidth. For example, when DVP_CTR.CAPFRC[1:0]=01, the camera interface samples the input data every frame, that is, the image data is collected every other frame. As shown below.

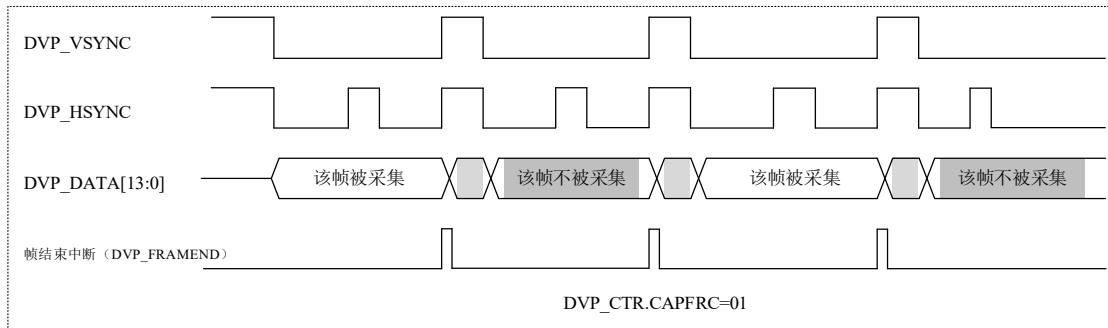


Figure 41-5 Frame acquisition frequency control

41.3.4 Synchronous control

41.3.4.1 Software synchronization

In software synchronization mode, the data stream is synchronized using the 32-bit synchronization code embedded in the data stream. These sync codes use 0x00/0xFF which are no longer used in the data stream. There are 4 types of synchronization codes, all of which are in the format of 0xFF0000XY. The DVP interface judges the synchronization state of the frame through a 32-bit word data collected.

The four software synchronization codes represent the following events:

- Frame start FS (DVP_SSNDR.FSDAT[7:0])
- Line Start LS (DVP_SSNDR.LSDAT[7:0])
- End of line LE (DVP_SSNDR.LEDAT[7:0])
- End of frame FE (DVP_SSNDR.FEDAT[7:0])

The XY in the sync code format 0xFF0000XY is determined by the above 4 sets of register bits. For example, when XY in the collected data is the same as DVP_SSNDR.FSDAT[7:0], it represents the start of a frame of data transmission.

The 4 synchronization codes also have a bit masking function, which can only use the unmasked bits in the synchronization codes for comparison. The masked bits of each synchronization code are set by the corresponding bits in DVP_SSNDR. When masking is enabled, a bit can be selected for synchronization code comparison to detect frame/line start and end. This means that the start and end of a frame/line can be represented by multiple sync codes, as long as the unmasked bits are the same. For example, when DVP_SSNDR.FSDAT[7:0] is set to 0xA5, and DVP_SSNDR.FSMSK[7:0] is set to 0xF0, it is only necessary to compare the upper 4 bits of the data code to detect whether it is an FS signal.

According to the software synchronization mechanism, DVP only supports 8-bit data format interface (DVP_CTR.BITSEL[1:0]=00) and full frame acquisition (DVP_CTR.CAPFRC[1:0]=00), other data width formats or interval frame acquisition When not applicable, it will produce unpredictable results. JPEG format data transmission is also not applicable (DVP_CTR.JPEGEN=0).

41.3.4.2 Hardware synchronization

Two port sync signals (DVP_VSYNC/DVP_HSYNC) are used in hardware sync mode. The camera interface module judges the frame/line start and frame/line end according to the level change of DVP_VSYNC/DVP_HSYNC.

The data stream capture function is enabled in hardware synchronization mode (DVP_CTR.CAPEN=1). After an invalid edge is captured on the DVP_VSYNC port, the data transfer can begin, and the DMA transfers consecutive frames to multiple consecutive buffers or a A buffer with circular properties. The frame transfer end interrupt is activated at the end of each frame (when the frame transfer end interrupt DVP_FRAMEND is enabled).

41.3.5 Window cropping

The camera interface can use the crop function to select a rectangular window from the received image. The starting coordinates and size of the window (the number of pixel clocks represent the horizontal size and the number of lines represent the vertical size) are specified by the window clipping offset register (DVP_CPSFTR) and the window clipping size register (DVP_CPSZER). Such asFigure 41-6 shown.

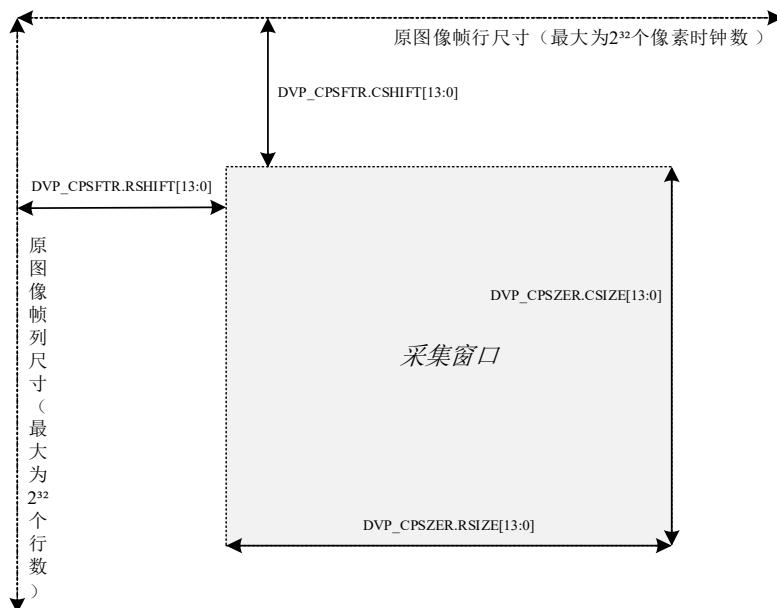


Figure 41-6 DVP window setting diagram

The starting coordinates of the cropping window are specified as a line number (starting from 0) and a pixel clock number (starting from 0), and the window size is specified as the line number and pixel clock number. If the DVP_VSYNC signal is asserted before the number of lines specified in the window crop size register (DVP_CPSZER) is complete, the image acquisition stops and a frame transfer end interrupt (DVP_FRAMEND) is generated when the interrupt is enabled. Such asFigure 41-7 , is a schematic diagram of window clipping data acquisition.

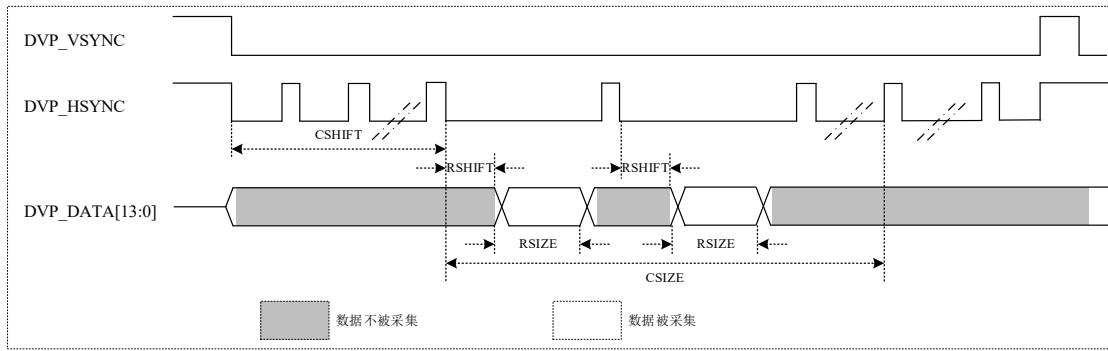


Figure 41-7 Window cropping data acquisition plot

When JPEG format data transmission (DVP_CTR.JPEGEN=1), the window cropping function is not applicable, and the window cropping function needs to be invalid.

41.3.6 FIFO control

In order to manage the data transfer speed on the AHB, an 8-word deep FIFO is equipped with a FIFO controller in the DVP module. The read pointer is incremented each time data is read from the camera interface FIFO, and the write pointer is incremented each time data is written to the camera interface FIFO. If the data writing speed exceeds the rate that the AHB interface can bear, the FIFO will overflow and the data will be overwritten. At this time, if the FIFO overflow error interrupt is enabled, the corresponding interrupt will be generated.

If a software synchronization error event occurs, the FIFO will be reset and the DVP interface will wait for a new data frame to begin.

41.3.7 DMA control

When DVP_CTR.CAPEN=1, the DMA interface is activated. The camera interface can trigger a DMA read request each time it receives a complete 32-bit block of data in the data register (DVP_DTR). When the DMA accesses the DMA data transfer register (DVP_DMR) through the AHB interface, the corresponding buffered data is read from the FIFO. The DMA read request is the 90th event in the event list (refer to the Interrupt Event Request Signal chapter of the INTC module). For the related settings of the DMA, refer to the DMA chapter.

During JPEG data transfer, the number of bytes contained in the line may not be a multiple of 4. When the DVP interface detects the end of the line and has not yet made up a 32-bit word, it will use "0" for padding and trigger a DMA read request.

41.4 Interrupt and Event Description

DVP interface contains three types of interrupt and event output, namely frame transfer status interrupt and event, software synchronization error interrupt and event, FIFO overflow error interrupt and event.

41.4.1 Frame transfer status interrupts and events

Corresponding interrupts can be generated in each state during the normal transmission of the image frame, as shown in the table below.

Table 41-10 Frame transfer interrupted

interrupt/event name	Interrupt/Event Symbol	interrupt enable bit
Frame transfer starts	DVP_FRAMSTA	IER.FSIEN
line transfer starts	DVP_LINESTA	IER.LSIEN
end of line transfer	DVP_LINEEND	IER.LEIEN
frame transfer end	DVP_FRAMEND	IER.FEIEN

41.4.2 Software Synchronization Error Interrupts and Events

In software sync mode, if the received sync code appears in Table 41-11 When the error sequence shown in , the corresponding status bit (DVP_STR.SQUERF) will be set to 1. At this time, if the corresponding interrupt enable (DVP_IER.SQUERIEN) is set to be valid, a software synchronization error (DVP_SQUERR) interrupt will be generated.

Table 41-11 Sync code error sequence

FS	LS	LE	FE
0	0	0	1
0	0	1	x
0	1	x	x
1	0	1	x
1	1	0	1

In the above table, "1" indicates that the synchronization sequence is received, "0" indicates that the synchronization sequence is not received, and "x" indicates that whether the synchronization sequence is received does not affect the judgment of sequence errors. For example, when FS: LS: LE: When FE=1101, it indicates that after the frame start and line start sequences are received, the frame end sequence is received, and the line end sequence is not received, and a software synchronization sequence error event is generated at this time.

When a synchronization code error occurs, the internal frame acquisition control logic will be reset, and the data of the current frame will no longer be received. After DVP_STR.SQUERF is cleared, it starts to receive and monitor the frame start sequence of the next frame.

41.4.3 FIFO overflow error interrupts and events

When the FIFO overflows, the corresponding status bit (DVP_STR.FIFOERF) will be set to 1. At this time, if the corresponding interrupt enable (DVP_IER.FIFOERIEN) is set to be valid, a FIFO overflow error (DVP_FIFOERR) interrupt will be generated.

41.5 Register description

Table 41-12 Shown is the register list of the DVP module.

BASE ADDR: 0x40055800H

Table 41-12 DVP register list

Register name	Symbol	Offset	Bit width	Reset value
control register	DVP_CTR	0x0000h	32	0x00000000h
Data register	DVP_DTR	0x0004h	32	0x00000000h
status register	DVP_STR	0x0008h	32	0x00000000h
interrupt register	DVP_IER	0x000Ch	32	0x00000000h
DMA data transfer register	DVP_DMR	0x0010h	32	0x00000000h
Software Sync Data Register	DVP_SSYNDR	0x0020h	32	0x00000000h
Software Synchronization Mask Register	DVP_SSYNMR	0x0024h	32	0xFFFFFFFFh
window clipping offset register	DVP_CPSFTR	0x0028h	32	0x00000000h
window crop size register	DVP_CPSZER	0x002Ch	32	0x00000000h

41.5.1 Control Register (DVP_CTR)

Reset value: 0x0000000h

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
-	DVP EN	-	-	BITSEL[1:0]	CAPFRC[1:0]	VSYNC SEL	Hsync SEL	PIXCK SEL	SW SYNC	JPEG EN	CROP EN	CAP MD	CAP EN		
<hr/>															
Bit	Marking		Place name			Function							Read and write		
b31~b15	Reserved		-			Read as "0", write as "0"							R/W		
b14	DVPEN		DVP module enable			0: DVP module is invalid 1: DVP module enable							R/W		
b13~b12	Reserved		-			Read as "0", write as "0"							R/W		
b11~b10	BITSEL[1:0]		Data width selection			00: 8-bit data width 01: 10-bit data width 10: 12-bit data width 11: 14-bit data width							R/W		
b9~b8	CAPFRC[1:0]		Frame acquisition frequency selection			00: Full frame capture (collect all frames) 01: Capture every 1 frame (bandwidth reduced by 50%) 10: Capture every 3 frames (bandwidth reduced by 75%) 11: Full frame capture (capture all frames)							R/W		
b7	VSYNCSEL		Frame sync level selection			0: When DVP_VSYNC is low level, the synchronization function is valid 1: When DVP_VSYNC is high, the synchronization function is valid							R/W		
b6	HsyncSEL		Line sync level selection			0: When DVP_HSYNC is low level, the synchronization function is valid 1: When DVP_HSYNC is high level, the synchronization function is valid							R/W		
b5	PIXCKSEL		Pixel clock selection			0: Data is collected at the falling edge of DVP_PIXCLK 1: Collect data on the rising edge of DVP_PIXCLK							R/W		
b4	SWSYNC		Software synchronization enable			0: Software synchronization is invalid 1: Software synchronization is valid							R/W		
b3	JPEGEN		JPEG transfer enable			0: Data transfer in non-JPEG format 1: JPEG format data transfer							R/W		
b2	CROOPEN		window clipping enabled			0: The window cropping function is invalid 1: Window cropping function is valid							R/W		
b1	CAPMD		Acquisition mode selection			0: Continuous mode 1: Single frame mode							R/W		
b0	CAPEN		Acquisition function enabled			Note: In single-frame mode, this bit is automatically cleared after the end of the next frame of data; in continuous mode, after this bit is written to zero, it will take effect after the current frame ends.							R/W		

41.5.2 Data Register (DVP_DTR)

Reset value: 0x00000000h

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]															

Bit	Marking	Place name	Function	Read and write
b31~b0	DATA[31:0]	Data collection	A set of data currently collected	R

41.5.3 Status Register (DVP_STR)

Reset value: 0x00000000h

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															

Bit	Marking	Place name	Function	Read and write
b31-b6	Reserved	-	Read as "0", write as "0"	R/W
b5	FIFOERF	FIFO overflow error flag	0: No FIFO overflow error occurred 1: FIFO overflow error occurred	R/W
b4	SQUERF	Software Sync Error Flag	0: No software sync error occurred 1: A software sync error occurred	R/W
b3	FEF	frame transfer end flag	0: The end of frame transmission is not detected 1: End of frame transfer detected	R/W
b2	LEF	line transfer end flag	0: End of line transfer not detected 1: End of line transfer detected	R/W
b1	LSF	line transfer start flag	0: Start of line transfer not detected 1: Start of line transfer detected	R/W
b0	FSF	frame transfer start flag	0: The start of frame transfer is not detected 1: Start of frame transfer detected	R/W

41.5.4 Interrupt Register (DVP_IER)

Reset value: 0x00000000h

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															
Bit Marking Place name Function Read and write															
b31-b6	Reserved	-	Read as "0", write as "0"										R/W		
b5	FIFOERIEN	FIFO overflow error interrupt enable	0: FIFO overflow error interrupt is invalid 1: FIFO overrun error interrupt enable										R/W		
b4	SQUERIEN	software sync error interrupt enable	0: Software synchronization error interrupt is invalid 1: Software synchronization error interrupt enable										R/W		
b3	FEIEN	Frame transfer end interrupt enable	0: Frame transfer end interrupt is invalid 1: Frame transfer end interrupt enable										R/W		
b2	LEIEN	Line transfer end interrupt enable	0: Line transfer end interrupt invalid 1: Line transfer end interrupt enable										R/W		
b1	LSIEN	Line transfer start interrupt enable	0: Line transfer start interrupt invalid 1: Line transfer start interrupt enable										R/W		
b0	FSIEN	Frame transfer start interrupt enable	0: Frame transfer start interrupt is invalid 1: Frame transfer start interrupt enable										R/W		

41.5.5 DMA Data Transfer Register (DVP_DMR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DMR[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DMR[15:0]															
Bit Marking Place name Function Read and write															
b31~b0	DMR[31:0]	DMA transfer value	Read the data in the current read FIFO from this register										R		

41.5.6 Software Synchronization Data Register (DVP_SSNDR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
FEDAT[7:0]								LEDAT[7:0]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LSDAT[7:0]								FSDAT[7:0]							
Bit	Marking	Place name	Function	Read and write											
b31~b24	FEDAT[7:0]	end-of-frame sync code	End-of-frame sync code in software sync mode	R/W											
b23~b16	LEDAT[7:0]	end-of-line sync code	End-of-line sync code in software sync mode	R/W											
b15~b8	LSDAT[7:0]	line start sync code	Line start sync code in software sync mode	R/W											
b7~b0	FSDAT[7:0]	start of frame sync code	Frame start sync code in software sync mode	R/W											

41.5.7 Software Synchronization Mask Register (DVP_SSNMR)

Reset value: 0xFFFFFFFFh

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
FEMSK[7:0]								LEMSK[7:0]							
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LSMSK[7:0]								FSMSK[7:0]							
Bit	Marking	Place name	Function	Read and write											
b31~b24	FEMSK[7:0]	end-of-frame mask	End of frame mask in software sync mode	R/W											
b23~b16	LEMSK[7:0]	end-of-line mask	End-of-line mask in software sync mode	R/W											
b15~b8	LSMSK[7:0]	line start mask	Line start mask in software sync mode	R/W											
b7~b0	FSMSK[7:0]	start of frame mask	Frame start mask in software sync mode	R/W											

41.5.8 Window Clipping Offset Register (DVP_CPSFTR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	CSHIFT[13:0]													

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	RSHIFT[13:0]													

Bit	Marking	Place name	Function	Read and write
b31~b30	Reserved	-	Read as "0", write as "0"	R/W
b29~b16	CSHIFT[13:0]	number of column offsets	Indicates the line number at the beginning of the window	R/W
b15~b14	Reserved	-	Read as "0", write as "0"	R/W
b13~b0	RSHIFT[13:0]	row offset	Number of columns (clock pixels) representing the start of the window Note: In software synchronization mode, the row offset number is shifted by a multiple of 4, b1~b0 can be read and written, but the function setting is invalid.	R/W

41.5.9 Window Crop Size Register (DVP_CPSZER)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	CSIZE[13:0]													

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
-	-	RSIZE[13:0]													

Bit	Marking	Place name	Function	Read and write
b31~b30	Reserved	-	Read as "0", write as "0"	R/W
b29~b16	CSIZE[13:0]	number of column dimensions	Indicates the size of each column of the window, that is, the number of rows of the window	R/W
b15~b14	Reserved	-	Read as "0", write as "0"	R/W
b13~b0	RSIZE[13:0]	Number of row sizes	Indicates the size (number of clock pixels) on each row of the window, that is, the number of columns of the window Note: The number of columns in the window size should be set to a multiple of 4. The settings of b1~b0 are invalid and fixed at 0.	R/W

42 Cryptographic Coprocessing Module (CPM)

42.1 Introduction

The encryption co-processing module (CPM) includes 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 new data encryption standard officially announced by the National Institute of Standards and Technology (NIST) on October 2, 2000. The block length is fixed at 128 bits, and the key length supports 128/192/256 bits. .

The HASH secure hash algorithm is the SHA-2 version of SHA-256 (Secure Hash Algorithm), which complies with the national standard "FIPS PUB 180-3" issued by the National Bureau of Standards and Technology of the United States. Produces 256-bit message digest output. Supports HMAC (Keyed Hash Message Authentication Code) applications, which use the SHA-256 hash function to authenticate messages.

The TRNG true random number generator is a random number generator based on continuous analog noise, providing 64bit random numbers.

42.2 Encryption and Decryption Algorithm Processor (AES)

42.2.1 Introduction to Algorithms

AES (The Advanced Encryption Standard) is a new data encryption standard officially announced by the National Institute of Standards and Technology (NIST) on October 2, 2000.

The block length of AES is fixed at 128 bits, and the key length supports 128, 192 and 256 bits. For encryption, the input is a plaintext block and a key, and the output is a ciphertext block; for decryption, the input is a ciphertext block and a key, and the output is a plaintext block. This process is as Figure 42-1 shown:

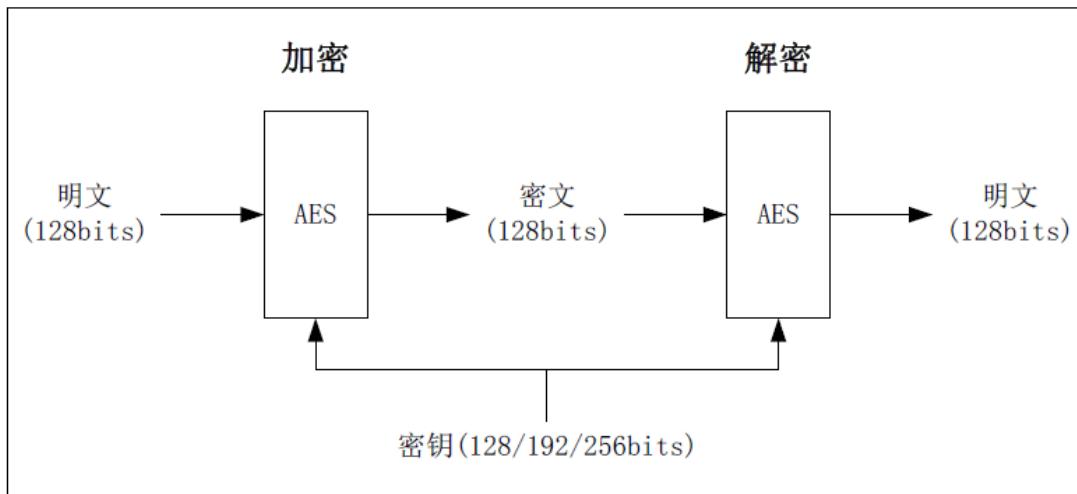


Figure 42-1 Schematic diagram of AES encryption and decryption process

The basic unit of AES algorithm processing is byte. 128-bit information is divided into 16 bytes and copied into a 4×4 matrix in order, called state. All transformations of AES are based on state matrix transformations, which holds the intermediate results of the calculation.

AES is a key iterative block cipher that involves the repetition of the state by rotation. The round transformation of AES consists of four operations: SubBytes, ShiftRows, MixColumns, AddRoundKey. Among them, SubBytes includes finding the modular inverse of each byte in $GF(2^8)$ and an affine transformation; ShiftRows is a byte transposition, which cyclically shifts the rows in the state according to different offsets; MixColumns performs linear transformation on each column of the state; AddRoundKey, performs bit-by-bit XOR operation on each byte in the state and the round key. The encryption process of AES is as follows Figure 42-2 shown:

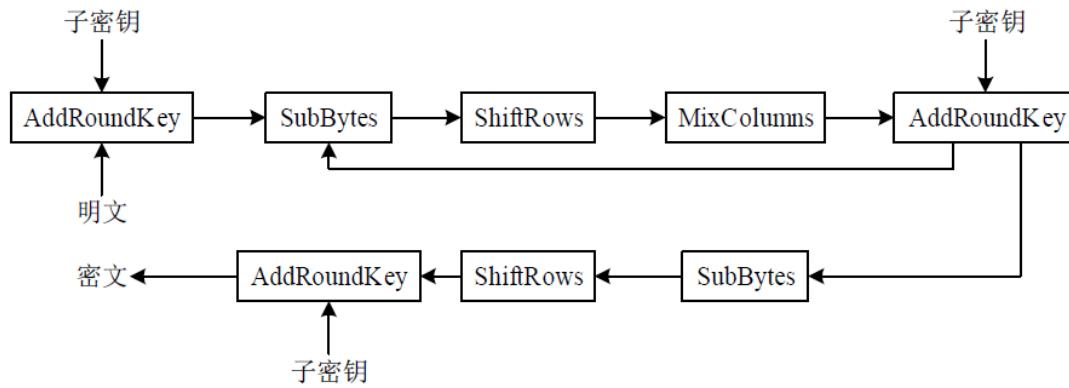


Figure 42-2 AES encryption flow chart

The subkey used in the figure needs to be expanded from the initial key, and the expansion process of the key and the encryption process are carried out synchronously.

Since the plaintext is fixed at 128 bits, the number of rounds the encryption process runs depends on the length of the key. For example, when the key is 128 bits, the number of running rounds is 10; when the key is 192 bits, the number of running rounds is 12; when the key is 256 bits, the number of running rounds is 14. Except for the last round, which lacks the MixColumns transformation, the remaining rounds are full round transformation operations.

The decryption process is different from the encryption process. First of all, the expansion of all keys must be completed, and the decryption process is used back from the last round of sub-keys of the expansion; 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 the inverse transform; InvShiftRows and InvMixColumns become the corresponding inverse transforms; AddRoundKey remains the same.

The calling sequence of the four operations in the round transformation of the direct decryption process is: InvShiftRows, InvSubBytes, AddRoundKey, and InvMixColumns are inconsistent with the calling sequence of the encryption process, but the keys used are consistent with the encryption process; the calling sequence of the four operations in the equivalent decryption process is as follows: InvSubBytes, InvShiftRows, InvMixColumns, and AddRoundKey are exactly the same as the calling sequence of the encryption process, except that the subkeys in each round need to perform the InvMixColumns operation.

For a detailed algorithm description, please refer to the standard "FIPS PUB 197".

42.2.2 AES module function description

- Execute the encryption process and decryption process of the AES algorithm standard, and the execution result fully conforms to the description of the algorithm principle in "FIPS PUB 197";
- 128, 192 and 256 bit keys are supported.

42.2.3 Encryption Operation Process

The encryption operation process of AES is as follows:

1. Write the 128-bit data to be encrypted into the data register (AES_DR).
2. Write the encryption key into the key register (AES_KR).
3. Set AES_CR.KEYSIZE based on key length
4. Set AES_CR.MODE to 0 to enable encryption mode.
5. Write 1 to AES_CR.START in the control register to start the module for operation.
3), 4) and 5) can be performed simultaneously.
6. Wait for the value of AES_CR.START to return to 0, and the module operation ends.
7. Read the data register (AES_DR) to get 128-bit ciphertext.

42.2.4 Decryption operation process

The decryption operation process of AES is as follows:

1. Write the 128-bit data to be decrypted into the data register (AES_DR).
2. Write the decryption key into the key register (AES_KR).
3. Set AES_CR.KEYSIZE based on key length
4. Set AES_CR.MODE to 1 to enable decryption mode.
5. Write 1 to AES_CR.START in the control register to start the module for operation.
3), 4) and 5) can be performed simultaneously.
6. Wait for the value of AES_CR.START to return to 0, and the module operation ends.
7. Read the data register (AES_DR) to get 128-bit plaintext.

42.2.5 Data example

128-bit plaintext:

0xFFEEDDCCBAA99887766554433221100

128-bit key:

0x0F0E0D0C0B0A09080706050403020100

128-bit ciphertext:

0x5AC5B47080B7CDD830047B6AD8E0C469

Table 42-1 128-bit operation register example

Before encryption			
Register	value (key)	Register	value (plaintext)
Key0	0x03020100	Data0	0x33221100
Key1	0x07060504	Data1	0x77665544
Key2	0x0B0A0908	Data2	0xBAA9988
Key3	0x0F0E0D0C	Data3	0xFFEEDDCC

after encryption			
Register	value (key)	Register	value (ciphertext)
Key0	0x03020100	Data0	0xD8E0C469
Key1	0x07060504	Data1	0x30047B6A
Key2	0x0B0A0908	Data2	0x80B7CDD8
Key3	0x0F0E0D0C	Data3	0x5AC5B470

128-bit plaintext:

0xFFEEDDCCBAA99887766554433221100

192-bit key:

0x17161514131211100F0E0D0C0B0A09080706050403020100

128-bit ciphertext:

0x5AC5B47080B7CDD830047B6AD8E0C469

Table 42-2 192-bit operation register example

Before encryption			
Register	value (key)	Register	value (plaintext)
Key0	0x03020100	Data0	0x33221100
Key1	0x07060504	Data1	0x77665544
Key2	0x0B0A0908	Data2	0xBBAA9988
Key3	0x0F0E0D0C	Data3	0xFFEEDDCC
Key4	0x13121110		
Key5	0x17161514		

after encryption			
Register	value (key)	Register	value (ciphertext)
Key0	0x03020100	Data0	0xA47CA9DD
Key1	0x07060504	Data1	0xE0DF4C86
Key2	0x0B0A0908	Data2	0xA070AF6E
Key3	0x0F0E0D0C	Data3	0x91710DEC
Key4	0x13121110		
Key5	0x17161514		

128-bit plaintext:

0xFFEEDDCCBAA99887766554433221100

256-bit key: 0x1F1E1D1C1B1A191817161514131211100F0E0D0C0B0A09080706050403020100

128-bit ciphertext:

0x5AC5B47080B7CDD830047B6AD8E0C469

Table 42-3 256-bit operation register example

Before encryption			
Register	value (key)	Register	value (plaintext)
Key0	0x03020100	Data0	0x33221100
Key1	0x07060504	Data1	0x77665544
Key2	0x0B0A0908	Data2	0xBBAA9988
Key3	0x0F0E0D0C	Data3	0xFFEEDDCC
Key4	0x13121110		
Key5	0x17161514		
Key6	0x1B1A1918		
Key7	0x1F1E1D1C		
after encryption			
Register	value (key)	Register	value (ciphertext)
Key0	0x03020100	Data0	0xCAB7A28E
Key1	0x07060504	Data1	0xBF456751
Key2	0x0B0A0908	Data2	0x9049FCEA
Key3	0x0F0E0D0C	Data3	0x8960494B
Key4	0x13121110		
Key5	0x17161514		
Key6	0x1B1A1918		
Key7	0x1F1E1D1C		

42.2.6 Running time description

The time required for the AES module to start an operation (write 1 in AES_CR.START) to the end of the operation (restore AES_CR.START to 0) is shown in the following table:

Table 42-4 AES encryption and decryption running time

	128-bit key	192-bit key	256-bit key
Encryption	220 cycles	260 cycles	300 cycles
Decrypt	290 cycles	332 cycles	398 cycles

42.2.7 Operation Precautions

1. During the AES encryption and decryption process, the data register will be changed. If the data of the next operation operation is the result of this operation, there is no need to rewrite the data.
2. Supports 128, 192 and 256-bit keys, 128-bit keys are written to offset addresses 0x20~0x2C, 192-bit keys are written to offset addresses 0x20~0x34, and 256-bit keys are written to offset addresses 0x20~0x3C.
3. The method of judging the end of the module operation: Keep reading AES_CR.START, if its value becomes 0, it means the operation is over.

42.2.8 Register description

Register base address: 0x40008000

Table 42-5 Register list

Register name	Symbol	Offset	Bit width	Reset value
AES Control Register	AES_CR	0x0000	32	0x00000000h
AES data register 0~3	AES_DR0~3	0x0010~0x001C	32	0x00000000h
AES key register 0~7	AES_KR0~7	0x0020~0x003C	32	0x00000000h

42.2.8.1 AES Control Register (AES_CR)

Reset value: 0x00000000h

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										KEYSIZE	Reserved	MODE	START				
Bit	Marking	Place name	Function										Read and write				
b31~b5	Reserved	-	Read as "0", write as "0"										R/W				
b4~b3	KEYSIZE	Key length selection	2'b00: key length is 128 bits 2'b01: key length is 192 bits 2'b10: The key length is 256 bits 2'b11: The key length is 128 bits										R/W				
b2	Reserved	-	Read as "0", write as "0"										-				
b1	MODE	Encryption and decryption mode selection	0: encryption operation 1: Decryption operation										R/W				
b0	START	start up	0: The operation of this module is finished or not started 1: Start the module for operation										R/W				

Note:

1. The operation method of the AES_CR.START bit is: After the software writes 1 to this bit, the module will start to run. After the end of this operation, the hardware of this module will automatically clear this bit to 0. If the software finds that this bit is 0, it means the operation is completed.
2. The write operation to this register can only be performed when the module is not in operation state (ie AES_CR.START = 0), otherwise the hardware will automatically ignore the write operation. Read operations are not subject to this restriction.

42.2.8.2 AES data register (AES_DRx) (x=0~3)

Reset value: 0x00000000h

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]															

Bit	Marking	Place name	Function	Read and write
b31~b0	Data	encrypted data	Store the 128-bit plaintext/ciphertext of the AES algorithm	R/W

Note:

1. The data register consists of four 32-bit registers to form 128-bit data, which is used to store the plaintext that needs to be encrypted or the ciphertext that needs to be decrypted before the module operation, and stores the encrypted ciphertext or decrypted plaintext after the operation is completed.

Encryption operation		Decryption operation	
Before operation	after operation	Before operation	after operation
128-bit plaintext	128-bit ciphertext	128-bit ciphertext	128-bit plaintext

Four 32-bit registers are connected together to form a 128-bit data, and the four registers need to be operated separately during read and write operations. The operation sequence corresponding to the data register is as follows:

Data example: 0xFFEEDDCCBAA99887766554433221100

Offset address	Register name	Fill in the data
0x10	AES_DR0	0x33221100
0x14	AES_DR1	0x77665544
0x18	AES_DR2	0xBBAA9988
0x1C	AES_DR3	0xFFEEDDCC

2. The writing to this register can only be performed when the module is not in operation state (ie AES_CR.START = 0), otherwise the hardware will automatically ignore the writing to this register.
3. The reading of this register can only be carried out when the module is not in the operation state (that is, when AES_CR.START = 0), otherwise the reading of this register will get all 0s.

42.2.8.3 AES key register (AES_KRx) (x=0~7)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Key[31:16]															

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

Bit	Marking	Place name	Function	Read and write
b31~b0	Key	key	Stores the 128/192/256-bit key of the AES algorithm	R/W

Note:

1. The key register consists of 8 32-bit registers, which store the input initial key. Eight 32-bit registers need to be operated separately during the write operation. The corresponding sequence of operations is as follows:

Example of a 128-bit key: 0x0F0E0D0C_0B0A0908_07060504_03020100

Offset address	register name	fill in the data
0x20	AES_KR0	0x03020100
0x24	AES_KR1	0x07060504
0x28	AES_KR2	0x0B0A0908
0x2C	AES_KR3	0x0F0E0D0C

Example of a 192-bit key:

0x17161514_13121110_0F0E0D0C_0B0A0908_07060504_03020100

Offset address	Register name	Fill in the data
0x20	AES_KR0	0x03020100
0x24	AES_KR1	0x07060504
0x28	AES_KR2	0x0B0A0908
0x2C	AES_KR3	0x0F0E0D0C
0x30	AES_KR4	0x13121110
0x34	AES_KR5	0x17161514

Example of a 256-bit key:

0x1F1E1D1C_1B1A1918_17161514_13121110_0F0E0D0C_0B0A0908_07060504_030201
00

Offset address	register name	fill in the data
0x20	AES_KR0	0x03020100
0x24	AES_KR1	0x07060504
0x28	AES_KR2	0x0B0A0908
0x2C	AES_KR3	0x0F0E0D0C
0x30	AES_KR4	0x13121110
0x34	AES_KR5	0x17161514
0x38	AES_KR6	0x1B1A1918
0x3C	AES_KR7	0x1F1E1D1C

2. The writing to this register can only be performed when the module is not in operation state (ie AES_CR.START = 0), otherwise the hardware will automatically ignore the writing to this register.
3. The reading of this register can only be carried out when the module is not in operation state (that is, when AES_CR.START = 0), otherwise the reading of this register will get all 0s.

42.3 Secure Hash Algorithm (HASH)

42.3.1 Introduction to Algorithms

The steps of a secure hashing algorithm are as follows:

The message is first padded so that its length is exactly a number that is only 64 bits less than a multiple of 512. The padding method is to append a 1 to the back of the message, followed by as many 0s as required, and then append a 64-bit message length (before padding), so that the message length is exactly an integer multiple of 512 bits.

Next, initialize 8 32-bit variables of A, B, C, D, E, F, G, and H with hexadecimal. Then start the main loop of the algorithm, processing 512-bit messages at a time, and the number of loops is the number of 512-bit packets in the message.

The main loop performs a total of 64 operations, which are called compression functions. Each operation includes shift, circular shift, logical operation, modulo 2³² addition, etc. The operation process is as followsFigure 42-3 . The final output consists of A, B, C, D, E, F, G, H concatenated. Where W_t is the temporary value used in the t-th step obtained from the 512-bit message, K_t is the constant value used in the t-th step, and t (0≤t≤63) is a step in a 64-step loop.

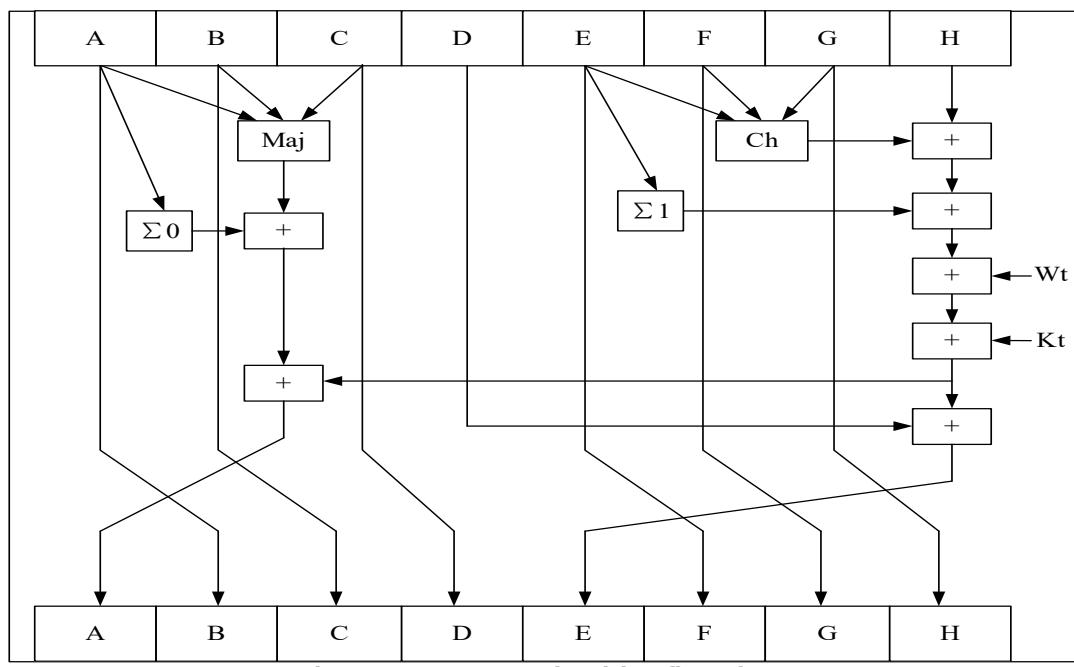


Figure 42-3 HASH algorithm flow chart

The HASH module integrates the hardware HMAC function; it can also directly call the HASH operation according to the HMAC algorithm through software to realize the HMAC function.

42.3.2 Operating procedures

The operation process of the HSAH module is as follows:

Software operation process:

1. The software fills the original data according to the algorithm rules, and groups the filled messages according to 512 bits.
2. Write data into the data register HASH_DR.
3. If this operation is the first group of data of message grouping, set HASH_CR.FST_GRP=1.
4. Set HASH_CR.START=1 to start this module for operation.

Note: 3) and 4) above can be performed simultaneously

5. To judge whether this operation of this module is completed, use the following methods:
Keep reading HASH_CR.START until the bit is 0, which means the operation is completed; or keep reading HASH_CR.BUSY until the bit is 0, which means the operation is complete; or set the HASH interrupt to judge the end of the operation by interrupting.
6. If this operation is not the last group of data in the message group, return to 2).
7. If this operation is the last group of data of the message grouping, read the summary register HASH_HR to obtain the result of this operation. If you need to perform the operation again, go back to step 1).

Using DMA operation flow:

1. The messages to be processed are grouped according to 512 bits, and the last group of messages needs to be pre-filled into 512-bit blocks according to the rules.
2. Set the HASH operation start mode of the internal trigger event register to DMA transfer completion and data block transfer completion; set the DMA transfer start mode of the internal trigger event register to HASH operation completion (or other internal trigger events);
3. Set HASH_CR.MODE=2'b00, HASH operation mode; set HASH_CR.FST_GRP=1, and set HASH_CR.START=1, start HASH operation. If you select other internal trigger events to start DMA, you need to wait for the trigger event to occur before starting the HASH operation.
4. Wait for HASH_CR.CYC_END=1, the HASH operation is completed, and clear this bit.
5. The result of reading the summary register HASH_HR.

Note that when there is only one set of operation data, please set HASH_CR.FST_GRP=1 and HASH_CR.KMSG_END=1.

42.3.3 Message padding

The padding packet processing steps of SHA-256 are as follows:

1. original message grouping

Divide the original message into L groups with a size of 512 bits. Let the total number of bits in the original message be l. If $l \% 512 < 448$, then the number of groups L is $l / 512$; if $l \% 512 \geq 448$, then the number of groups L is $l / 512 + 1$.

2. add length

① Add padding bits:

Add padding bits at the end of group $l / 512$ of the message packet: A 1 and several 0s, the number of 0s can be zero. If $l \% 512 < 448$, padding makes the length of data bits satisfy the length of $448 \bmod 512$ (the last 64 bits are reserved for the original message length); if $l \% 512 \geq 448$, use a 1 and several 0s to put the first $l / 512$ The 512-bit data block of the group is filled, and the first 448 bits of the Lth ($L = l / 512 + 1$) group are filled with 0.

② Add original message length:

A 64bit block, representing the original message length, as a 64bit unsigned integer. Add the original message length at the last 64 bits of the Lth packet.

An example to illustrate the process of filling a group is as follows:

1) Example1:

The original message is the string "abcde", and its ASCII code is represented by a binary bit string: " 01100001 01100010 01100011 01100100 01100101", the steps to add the length are as follows:

- A. Add "1". The populated message is "01100001 01100010 01100011 01100100 01100101 1".
- B. Add "0". Because the original message length is 40 bits, the number of 0s to be added is $512 - 64 - 40 - 1 = 407$.

The padded message becomes (hex):

61626364 65800000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000

- C. Add original message length. The original message length 40 is expressed in two 32bit words (hexadecimal): 00000000 00000028.

The padded message becomes (hex):

61626364 65800000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000028

2) Example2:

The original message is the string "abcdabcdecdefdefgefghfhighijhijklmklmnlmnmonopnopq". Each character can be converted to 8bit by its ASCII code,

so the length of the message is $l = 56 * 8 = 448$.

- A. Add "1" and "0". The padded message (in hexadecimal) is the first message block:

61626364 62636465 63646566 64656667
65666768 66676869 6768696A 68696A6B
696A6B6C 6A6B6C6D 6B6C6D6E 6C6D6E6F
6D6E6F70 6E6F7071 80000000 00000000.

- B. Add original message length. The original message length of 448 is expressed in two 32bit words (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

42.3.4 HMAC operation

HMAC (Keyed-Hashing for Message Authentication) irreversibly binds the message being processed with a key selected by the user for message authentication. Basically, HMAC consists of two nested underlying HASH operations by adding a shared secret:

$\text{HMAC}(\text{message}) = \text{Hash}[(\text{key XOR opad}) \mid \text{Hash}((\text{key XOR ipad}) \mid \text{message})]$

in:

ipad: 64 bytes (SHA-256) 0x36.

opad: 64 bytes (SHA-256) 0x5C.

Represents a concatenation operator.

42.3.4.1 HMAC operation process

1. Set HASH_CR.MODE=2'b01 to select HMAC working mode;
2. If the length of the key used is more than 64 bytes, you need to set HASH_CR.LKEY=1 (in this case, HMAC will use the key specified by the specification to replace the given key);
3. Write the key to the HASH_DR register until all key grouping operations are completed;
4. Write the message to the HASH_DR register until all message grouping operations are completed;
5. Read the final hash operation result of the HASH_HR register.

For data loading in the HMAC operation process, you can choose the software operation process and the DMA operation process. The specific examples are as follows:

42.3.4.2 HMAC software operation process

1. Set mode HASH_CR.MODE=2'b01, HMAC working mode; set HASH_CR.LKEY=1, select long key mode;
2. Write key grouping data in HASH_DR;
3. If it is the first group of data, set HASH_CR.FST_GRP=1. Set HASH_CR.START=1 to start key operation;
4. Wait for the HASH_CR.BUSY=0 operation to complete;
5. If it is not the last set of data, repeat 2)~4);
6. If it is the last set of data, set HASH_CR.KMSG_END=1, and set HASH_CR.START=1 starts the last set of key operations. After waiting for HASH_CR.CYC_END=1, the key operation is completed and this bit is cleared.
7. Write message packet data in HASH_DR;
8. If it is the first group of message data, set HASH_CR.FST_GRP=1. Set HASH_CR.START=1 to start message operation;
9. Wait for the HASH_CR.BUSY=0 operation to complete;
10. If it is not the last set of message data, repeat operations 7)~9);
11. If it is the last group of messages, set HASH_CR.KMSG_END=1, and set HASH_CR.START=1 starts the last group of message operations. After waiting for HASH_CR.CYC_END=1, all HMAC operations are completed and this bit is cleared;
12. Read the final message authentication result in HASH_HR.

42.3.4.3 HMAC uses DMA operation flow

1. Set DMA as the data block transmission mode, the size of the data block is 16 words, whether it is a key or a message, it needs to be grouped according to the size of 16 words. When it is less than 16 words, please fill it according to the HASH message filling rules. The source address of the DMA data block is the key address;
2. Set the HASH operation start mode of the internal trigger event register to DMA transfer completion and data block transfer completion; set the DMA transfer start mode of the internal trigger event register to HASH operation completion (or other internal trigger events);
3. Set HASH_CR.MODE=2'b01, HMAC operation mode; configure according to the key length HASH_CR.LKEY bit; set HASH_CR.FST_GRP=1, HASH_CR.START=1 to start the key operation; if you select other internal trigger events, you need to wait for the trigger event to start the operation;
4. After waiting for HASH_CR.CYC_END=1 to complete the key operation, clear this bit;
5. Stop DMA and point the source address of the DMA data block to the message address;
6. Set HASH_CR.FST_GRP=1, HASH_CR.START=1 to start message operation; or wait for other trigger events to start message operation.

7. After waiting for HASH_CR.CYC-END=1 message operation to complete, clear this bit;
 8. Read the final message authentication result in the HASH HR register.

42.3.4.4 Example of HMAC operation

The results of the three sets of HMAC operations for the key and the message are as follows:

42.3.4.5 Precautions

1. When the filled key and message have only one set of data, i.e. 512 bits, please set them at the same time before starting the operation HASH_CR.FST_GRT=1 and HASH_CR.KMSG_END=1;
 2. Regarding key padding, when the key is less than 64 bytes, please fill in the part less than 64 bytes with 0x00; when the key is greater than 64 bytes, please fill in the part less than 64 bytes according to the HASH filling rules.

42.3.5 Interrupt Description

HASH interrupt flag HASH_INT. No matter in HASH operation mode or HMAC operation mode, under the condition of setting HASH_CR.HEIE=1 interrupt permission, an interrupt request will be generated after each group of data operation is completed. Under the condition of HASH_CR.HCIE=1 interrupt permission, an interrupt request will be generated after all key or message operations are completed. When HEIE and HCIE are allowed at the same time, an interrupt request will be generated if either condition is satisfied. In addition, after HEIE is enabled, after the module is reset or after the operation is completed, when HASH_CR.FST_GRP is directly set and HASH_CR.START=1, an interrupt will also be generated for data loading.

42.3.6 Hardware trigger event selection

By configuring the internal hardware trigger event selection register HASH_ITRGSELA, HASH_ITRGSELB starts the HASH operation after the data transfer is completed, which is usually used in conjunction with DMA. For example, when selecting DMA_2 to transfer data and start HASH operation, first, configure DMA2_TRGSELx ($x=0\sim 7$) to select HASH_INT to start DMA transfer; secondly, configure HASH_ITRGSELA to select DMA_BT_Cx ($x=0\sim 7$), which means that a data block (512bit) transmission and start the HASH operation, configure HASH_ITRGSELB to select DMA_TC_x ($x=0\sim 7$), indicating that all data block transmissions are completed. When HASH is started, DMA and HASH trigger each other until all data blocks are transferred and calculated.

42.3.7 Register description

Table 42-6 HASH register list

BASE ADDR: 0x40008400

Register name	Symbol	Offset	Bit width	Reset value
HASH Control Register	HASH_CR	0x0000	32	0x00000000h
HASH Summary Register 7	HASH_HR7	0x0010	32	0x00000000h
HASH Summary Register 6	HASH_HR6	0x0014	32	0x00000000h
HASH Summary Register 5	HASH_HR5	0x0018	32	0x00000000h
HASH Summary Register 4	HASH_HR4	0x001C	32	0x00000000h
HASH Summary Register 3	HASH_HR3	0x0020	32	0x00000000h
HASH Summary Register 2	HASH_HR2	0x0024	32	0x00000000h
HASH Summary Register 1	HASH_HR1	0x0028	32	0x00000000h
HASH summary register 0	HASH_HR0	0x002C	32	0x00000000h
HASH data register 15	HASH_DR15	0x0040	32	0x00000000h
HASH data register 14	HASH_DR14	0x0044	32	0x00000000h
HASH data register 13	HASH_DR13	0x0048	32	0x00000000h
HASH data register 12	HASH_DR12	0x004C	32	0x00000000h
HASH data register 11	HASH_DR11	0x0050	32	0x00000000h
HASH data register 10	HASH_DR10	0x0054	32	0x00000000h
HASH data register 9	HASH_DR9	0x0058	32	0x00000000h
HASH data register 8	HASH_DR8	0x005C	32	0x00000000h
HASH data register 7	HASH_DR7	0x0060	32	0x00000000h
HASH data register 6	HASH_DR6	0x0064	32	0x00000000h
HASH data register 5	HASH_DR5	0x0068	32	0x00000000h
HASH data register 4	HASH_DR4	0x006C	32	0x00000000h
HASH data register 3	HASH_DR3	0x0070	32	0x00000000h
HASH data register 2	HASH_DR2	0x0074	32	0x00000000h
HASH data register 1	HASH_DR1	0x0078	32	0x00000000h
HASH data register 0	HASH_DR0	0x007C	32	0x00000000h

BASE ADDR: 0x40010800

Register name	Symbol	Offset	Bit width	Reset value
Hardware Event Trigger Select Register A	HASH_ITRGSELA	0x007C	32	0x000001FFh
Hardware Event Trigger Select Register B	HASH_ITRGSELB	0x0078	32	0x000001FFh

42.3.7.1 HASH Control Register (HASH_CR)

Reset value: 0x00000000h

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
HEIE	HCIE	-	-	-	HMA_C_EN_D	CYC-END	BUSY	-	LKEY	MODE[1:0]	-	KMS_G_EN_D	FST_GRP	STAR_T	
<hr/>															
Bit	Marking	Place name	Function	Read and write											
b31~b16	Reserved	-	"0" when reading, please write "0" when writing	R/W											
b15	HEIE	Completion of interrupt permission for each group of data operations	0: Interrupt disabled when each group of operations is completed 1: Interrupt permission for completion of each group of operations	R/W											
b14	HCIE	All key or message operations complete interrupt permission	0: Interrupt disabled when all key or message operations are completed 1: Interrupt permission for completion of all key or message operations	R/W											
b14~b11	Reserved	-	"0" when reading, please write "0" when writing	R/W											
b10	HMAC_END	HMAC operation complete flag	0: HMAC operation is not completed 1: HMAC operation completed When the flag bit is "1", write "0" to clear. Note that after the software confirms the flag bit, please clear "0" in time, otherwise the flag bit will remain "1".	R/W											
b9	CYC_END	Key or message operation completion flag	0: Key or message operation not completed 1: The key or message operation is complete When the flag bit is "1", write "0" to clear. Note that after the software confirms the flag bit, please clear "0" in time, otherwise the flag bit will remain "1".	R/W											
b8	BUSY	conversion flag	0: HASH operation is in idle state 1: HASH operation processing	R											
b7	Reserved	-	"0" when reading, please write "0" when writing	R/W											
b6	LKEY	long key selection	0: Short key (\leq 64 bytes) 1: long key ($>$ 64 bytes) In HMAC mode, this bit selects between a short key (\leq 64 bytes) or a long key ($>$ 64 bytes). Valid when MODE=2'b01, HMAC mode.	R/W											
b5~b4	MODE	Operating mode	00: SHA-256 working mode 01: HMAC working mode 1X: set ban	R/W											
b3	Reserved	-	"0" when reading, please write "0" when writing	R/W											
b2	KMSG_END	the last group of the key or message grouping	0: The key or message is not the last group 1: The last set of keys or messages When the last set of key or message conversion is completed, this bit is automatically cleared to "0"												
b1	FST_GRP	the first group of the key or message grouping	0: The key or message is not the first group 1: Key or message first group When the first group of key or message conversion is completed, this bit is automatically cleared to "0"	R/W											
b0	START	start up	0: The operation of this module is finished or not started 1: Start the module for operation	R/W											

Note:

- The operation method of the START bit is: After the software writes 1 to this bit, the module will start running; the hardware will automatically clear the bit to 0 after this running; if the software finds that the bit is 0, it means the running is complete.

- The write operation to this register can only be performed when the module is not in the operation state (that is, when the START bit is 0), otherwise the hardware will automatically ignore the write operation. Read operations are not subject to this restriction.

42.3.7.2 HASH Summary Register (HASH_HR)

Digits: 256 bits

offset address: 10'h010 - hash[255:224]

10'h014 - hash[223:192]
 10'h018 - hash[191:160]
 10'h01C - hash[159:128]
 10'h020 - hash[127:96]
 10'h024 - hash[95:64]
 10'h028 - hash[63:32]
 10'h02C - hash[31:0]

Reset value: 0x00000000h (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/>															
Bit	Marking	Place name	Function	Read and write											
b31~b0	HASH[31:0]	summary value	After the module operation is completed, the message digest is obtained by reading this register	R/W											

Note:

- This register is formed by splicing eight 32-bit registers. When accessing, 8 32-bit registers are operated in turn, and the 32-bit register corresponding to the low address stores the high word of the message digest.
- Hardware will automatically ignore writes to this register.
- The reading of this register can only be carried out when the module is not in the operation state (HASH_CR.START=0), otherwise the reading of this register will get all 0s.

42.3.7.3 HASH Data Register (HASH_DR)

Digits: 512 bits

offset address: 10'h040 - data[511: 480]

10'h044 - data[479: 448]

10'h048 - data[447: 416]

10'h04C - data[415: 384]

.....

10'h070 - data[127: 96]

10'h074 - data[95: 64]

10'h078 - data[63: 32]

10'h07C - data[31: 0]

Reset value: 0x00000000h (each 32-bit register)

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]															

Bit	Marking	Place name	Function	Read and write
b31~b0	DATA[31:0]	Data register	Used to write messages before block operations	R/W

Note:

- This register is spliced by 16 32-bit registers. When accessing, 16 32-bit registers are operated in turn, and the 32-bit register corresponding to the low address stores the high word of the data.
- The writing to this register can only be performed when the module is not in the operation state (HASH_CR.START), otherwise the hardware will automatically ignore the writing to this register.
- A read of this register will always result in all 0s.

42.3.7.4 HASH hardware event trigger selection register A (HASH_ITRGSELA)

Register description: After the data is written into HASH_DR, the hardware event trigger source is selected through this register to start the HASH operation. Note, do not select other trigger signals other than DMA_BTCx(x=0~7).

offset address: 0x7C

Reset value: 0x0000001FF

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								TRGSEL[8:0]															
<hr/>																							
Bit	Marking	Place name	Function	Read and write																			
b31~b9	Reserved	-	Read 0 when reading, write 0 when writing	R																			
b8~b0	ITRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected	R/W																			

42.3.7.5 HASH hardware event trigger selection register B (HASH_ITRGSELB)

Register description: After the last group of data is written into HASH_DR, the hardware event trigger source is selected through this register to notify HASH to perform the last operation. Note, do not select other trigger signals than DMA_TCx(x=0~7).

offset address: 0x78

Reset value: 0x000001FF

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								TRGSEL[8:0]															
<hr/>																							
Bit	Marking	Place name	Function	Read and write																			
b31~b9	Reserved	-	Read 0 when reading, write 0 when writing	R																			
b8~b0	ITRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected	R/W																			

42.4 True random number generator TRNG

42.4.1 Block Diagram

The TRNG module provides a true random number generator to generate a 64-bit random number.

The system block diagram of TRNG is as followsFigure 42-4 shown. The random number generator is an analog random number oscillator circuit, which is used to obtain random noise; 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 startup of the TRNG.

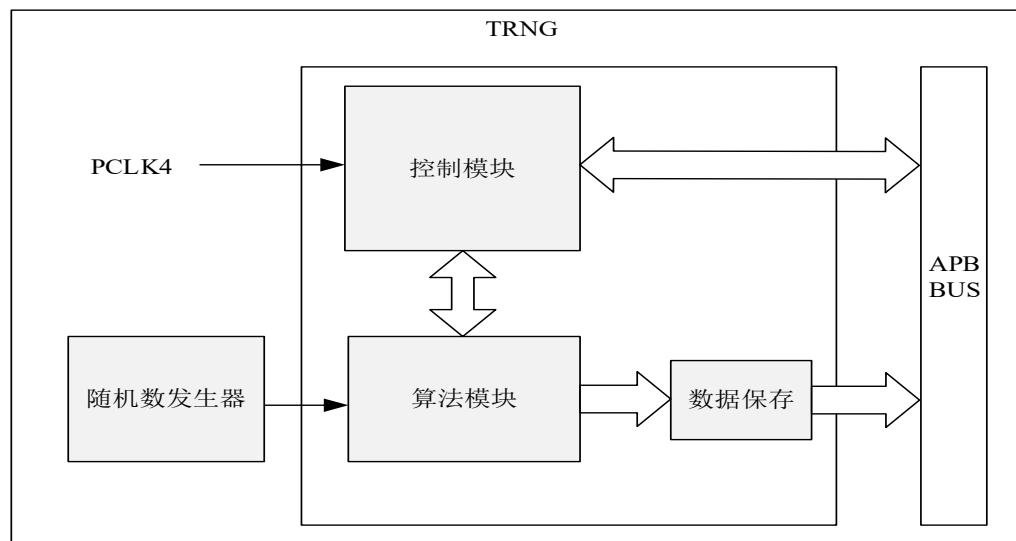


Figure 42-4 TRNG system block diagram

42.4.2 Operating procedures

The true random number generation process 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 random numbers (read TRNG_DR).
5. Turn off the random number generator circuit (set the EN bit of TRNG_CR to 0).

42.4.3 Interrupt and event output

When the random number generation is completed, the hardware of the register bit TRNG_CR.RUN is cleared and the random number generation complete interrupt request (TRNG_END) is generated. When the random number is generated, an event output is also generated, which can trigger the linkage of other modules.

42.4.4 Operation Precautions

In order to obtain a good random number, please set the frequency of the peripheral clock PCLK4 below 1MHz.

42.4.5 Register description

BASE ADDR: 0x40042000

Table 42-7 TRNG register list

Register name	Symbol	Offset	Bit width	Reset value
TRNG control register	TRNG_CR	0x0000h	32	0x00000000h
TRNG Mode Register	TRNG_MR	0x0004h	32	0x00000012h
TRNG data register 0	TRNG_DR0	0x000Ch	32	0x80000000h
TRNG data register 1	TRNG_DR1	0x0010h	32	0x8000200h

42.4.5.1 TRNG Control Register (TRNG_CR)

Reset value: 0x00000000h

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
-	-	-	-	-	-	-	-	-	-	-	-	-	RUN	END	

Bit	Marking	Place name	Function	Read and write
b31~b2	Reserved	-	Read as "0", write as "0"	R/W
b1	RUN	Random number operation starts	0: stop random number calculation 1: Random number operation starts The software writes "1" to generate a new 64-bit random number; after the operation is completed, the hardware clears it.	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

42.4.5.2 TRNG Mode Register (TRNG_MR)

Reset value: 0x00000012h

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
<hr/>															
-	-	-	-	-	-	-	-	-	-	-	-	CNT[2:0]	-	LOAD	
Bit	Marking	Place name	Function												Read and write
b31~b5	Reserved	-	Read as "0", write as "0"												R/W
b4~b2	CNT[2:0]	Shift times control bit	When capturing random noise, the shift count control bit 011: Shift 32 times 100: Shift 64 times 101: Shift 128 times 110: Shift 256 times 000~010, 111: Function reserved bit												R/W
b1	Reserved	-	Read as "1", write as "1"												R/W
b0	LOAD	load control bits	Whether the data register is loaded with a new initial value from the random number generator before random number generation 0: Do not load new initial values 1: Load new initial value												R/W

42.4.5.3 TRNG Data Register (TRNG_DR)

Reset value: DR0: 0x08000000h

DR1: 0x08000200h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
<hr/>															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
<hr/>															
<hr/>															
Bit	Marking	Place name	Function												Read and write
b31-b0	DATA[31:0]	random number	64-bit random number												R

43 CRC operation (CRC)

43.1 Introduction

In many applications, CRC algorithm is needed to verify the integrity and correctness of data. Especially in data transmission, CRC is widely used. This module can use CRC 16 and CRC 32 algorithms to calculate and check the data.

43.2 Functional block diagram

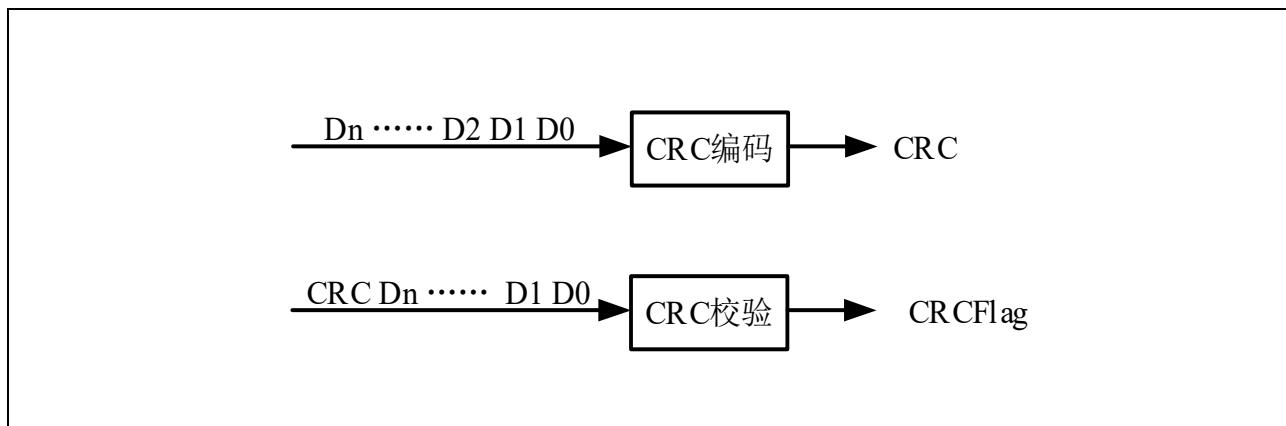


Figure 43-1 CRC Application Diagram

43.3 Functional description

The CRC algorithm of this module complies with the definition of ISO/IEC 13239 and adopts 32-bit and 16-bit CRC respectively. The polynomial of CRC 32 is $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^5 + X^4 + X^2 + X + 1$, 32-bit initial value is "0xFFFFFFFF". The polynomial of CRC 16 is $X^{16} + X^{12} + X^5 + 1$, the initial value of 16 bits is "0xFFFF".

The functions of this module include:

- CRC coding and CRC check
- 3 Bitwidth access modes 8 bits, 16 bits, 32 bits:
 - Examples of 8-bit width input data are 0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77
 - Examples of 16 bit width input data are 0x1100, 0x3322, 0x5544, 0x7766
 - Examples of 32-bit width input data are 0x33221100, 0x77665544

43.3.1 CRC 16 coding mode

CRC coding is to encode the original data to calculate its CRC value. The procedure is as follows:

1. Write 1'b0 to CRC._CR.CR, select CRC16.
2. Write 0xFFFF to CRC._RESLT [15: 0] and initialize CRC calculation.
3. Write the original data to be encoded to the CRC._DATregister in 8-bit/16-bit/32-bit organization.

Note: To write the CRC value in 8-bit organization, the low-order bits should be written first, and then the high-order bits should be written.

4. Read CRC _ RESLT [15: 0] to get 16-bit CRC coding value.

43.3.2 CRC 16 check mode

Check mode can check if the encoded data has been tampered with. The operation flow is as follows:

1. Write 1'b0 to CRC _ CR.CR, select CRC16.
2. Write 0xFFFF to CRC _ RESLT [15: 0] and initialize CRC calculation.
3. Write the encoded data to the CRC _ DATregister in order of 8-bit/16-bit/32-bit organization.
Note: To write the CRC value in 8-bit organization, the low-order bits should be written first, and then the high-order bits should be written.
4. The check code is 16 bits wide to write data into the CRC_DAT register.
5. Read CRC _ CR.FLGregister, 1 indicates check success, and 0 indicates check failure.

43.3.3 CRC 32 coding mode

CRC coding is to encode the original data to calculate its CRC value. The procedure is as follows:

1. Write 1'b1 to CRC _ CR. CR, select CRC32.
2. Write 0xFFFF_ FFFF to CRC _ RESLT [31: 0] and initialize CRC calculation.
3. Write the original data to be encoded to the CRC _ DATregister in 8-bit/16-bit/32-bit organization.
Note: To write the CRC value in 8-bit organization, the low-order bits should be written first, and then the high-order bits should be written.
4. After reading CRC _ RESLT [31: 0], 16-bit CRC code value can be obtained.

43.3.4 CRC 32 check mode

Check mode can check if the encoded data has been tampered with. The operation flow is as follows:

1. Write 1'b1 to CRC _ CR. CR, select CRC32.
2. Write 0xFFFF_ FFFF to CRC _ RESLT [31: 0] and initialize CRC calculation.
3. Write the encoded data to the CRC _ DATregister in order of 8-bit/16-bit/32-bit organization.
Note: To write the CRC value in 8-bit organization, the low-order bits should be written first, and then the high-order bits should be written.
4. The check code is organized into 8-bit/16-bit/32-bit and written into the CRC_DAT register.
5. Read CRC _ CR.FLGregister, 1 indicates check success, and 0 indicates check failure.

43.4 Register description

Table 43-1 Shown is the register list of the CRC module.

CRC_BASE_ADDR: 0x40008C00

Table 43-1 CRC register list

Register name	Symbol	Offset address	Bit width	Reset value
CRC control register	CRC_CR	0x00	32	0x0000_0001
CRC result register	CRC_RESLT	0x04	32	0x0000_0000
CRC data register	CRC_DAT	0x80~0xFF	32	0x0000_0000

43.4.1 Control register (CRC _ CR)

Reset value: 0x0001

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
-	-	-	-	-	-	-	-	-	-	-	-	-	-	FLAG	CR

Bit	Marking	Place name	Function	Read and write
b30~b2	Reserved	-	Read as "0", write as "0"	R/W
b1	FLAG	Check result bit	0: Current check error 1: Current check is correct	R
b0	CR	Operational control bit	0: CRC16 1: CRC32	R/W

43.4.2 Results register (CRC _ RESLT)

Reset value: 0x0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
RESULT[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RESULT[15:0]															
Bit	Marking	Place name	Function	Read and write											
b31~b0	RESULT[31:0]	Result bit	When CRC 16 is selected, RESULT [15: 0]; When CRC 32 was selected, RESULT [31: 0] was selected.	R/W											

43.4.3 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
CRC_DAT[15:0]															
Bit	Marking	Place name	Function	Read and write											
b31~b0	CRC_DAT[31:0]	Data register	This register is used to input data to be computed; The address of this register is a range (0x80 ~ 0xFF), and it is considered to operate on any address in that range. This register is read as all 0s.	R/W											

44 Data Computation Unit (DCU)

44.1 Summary

The Data Computing Unit is a module that simply processes data without the aid of a CPU. Each DCU unit has 3 data registers, capable of adding and subtracting 2 data and comparing size and window comparison functions. It can also be triggered by a timer to provide a digital-to-analog conversion module (DAC) with continuously changing digital quantities to generate triangular wave and sawtooth wave output. This product is equipped with 8 DCU units, each unit can independently complete its own function.

Function summary:

- 4 kinds of data processing are possible: Addition, subtraction, comparison of 2 data and comparison of 3 data windows
- Addition and subtraction operate on the data in the DATA0 and DATA1 registers, and the result is stored in DATA0
- Addition and subtraction can be calculated after writing the register or triggered by other peripheral circuit events.
- Addition and subtraction operations can automatically halve the result once, and put the halved result and the result of addition and subtraction into 2 data registers for use by other modules.
- The comparison mode can compare the 2 data between the DATA0 and DATA1 registers, and between the DATA0 and DATA2 registers, and can choose to generate an interrupt and a flag when it is greater than, less than, or equal to.
- The comparison mode can be used for window comparison, that is, set DATA1 and DATA2 as the upper and lower limits of the window respectively, and judge whether DATA0 is inside or outside the window according to the comparison results of DATA0 and DATA1 and DATA0 and DATA2
- It can be triggered by other peripheral circuit events to perform operations, and generate various interrupt and event signals according to the operation results. When other peripheral circuits with hardware trigger start function select DCU as the trigger source, the DCU generates an event signal to start the peripheral circuit to start action
- In the triangular wave output mode, the digital quantity of continuous increment and decrement is generated by timer timing trigger to the digital-to-analog conversion module (DAC), which can convert these digital quantities into triangular wave output through digital-to-analog conversion.
- In the incremental sawtooth wave output mode, the timer is regularly triggered to generate continuously increasing digital quantities to the digital-to-analog conversion module (DAC),

which can convert these digital quantities into incremental sawtooth wave output through digital-to-analog conversion.

- In the decrementing ramp wave output mode, the timer is regularly triggered to generate continuously decreasing digital quantities to the digital-to-analog conversion module (DAC), which can convert these digital quantities into decrementing sawtooth wave output through digital-to-analog conversion.

44.2 Functional description

44.2.1 Additive mode

The addition mode calculates the sum of DATA0 and DATA1, where DATA0 is the summand and DATA1 is the addend. Each time the DATA1 register is written to perform an operation of $(\text{DATA0} + \text{DATA1})/2$, the result of $\text{DATA0} + \text{DATA1}$ is stored in DATA0, and 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 0xFFFFFFFF (32bit mode), a flag is generated and an interrupt is generated.

Addition mode application example:

1. The control register DCU_CTL selects the addition mode, and the data width is 16bit
2. The interrupt condition selection register DCU_INTSEL selects the operation condition
3. Write 0xFF00 and 0x55 in DATA0 and DATA1 respectively, the calculation result is 0xFF55 at this time, and the result is stored in DATA0
4. Continue to write 0xFF in DATA1. At this time, the calculation result overflows and a result flag is generated. Read the result flag of the flag register DCU_FLAG
5. Write the flag reset register DCU_FLAGCLR to clear the flag bit

44.2.2 Subtraction mode

The subtraction mode calculates the difference between DATA0 and DATA1, where DATA0 is the minuend and DATA1 is the subtrahend. Each time the DATA1 register is written to perform an operation of $(\text{DATA0} - \text{DATA1})/2$, the result of $\text{DATA0} - \text{DATA1}$ is stored in DATA0, and 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.

44.2.3 Hardware Triggered Boot Mode

The DCU can trigger the start-up operation according to the events generated by the peripheral circuit. When using the hardware trigger start mode, the peripheral circuit trigger function enable bit of the function clock control 0 register (FCG0) needs to be enabled first. Each DCU unit can independently select the trigger start signal sent by 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_TRGSEL). When the 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, each time an event trigger occurs, the DCU will start and perform an operation of $(\text{DATA0} + \text{DATA1})/2$, the result of $\text{DATA0} + \text{DATA1}$ is stored in DATA0, and 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 0xFFFFFFFF (32bit mode), a flag is generated and an interrupt is generated. In the trigger minus mode, each time an event trigger occurs, the DCU will start and perform an operation of $(\text{DATA0} - \text{DATA1})/2$, the result of $\text{DATA0} - \text{DATA1}$ is stored in DATA0, and 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.

Hardware trigger start mode application example:

1. The control register DCU_CTL selects the trigger plus mode, and the data width is 8bit
2. The interrupt condition selection register DCU_INTSEL selects the operation condition
3. Write 0x00 and 0x56 in DATA0 and DATA1 respectively
4. Write the event number in the trigger source selection register DCU_TRGSEL
5. Activate the selected peripheral circuit and generate an event. The DCU is triggered by the event and performs an addition operation. The result is 0x56 and stored in DATA0
6. After three consecutive triggers, the calculation result overflows and a result flag is generated. Read the flag register DCU_FLAG to get the result flag
7. Write the flag reset register DCU_FLAGCLR to clear the flag bit

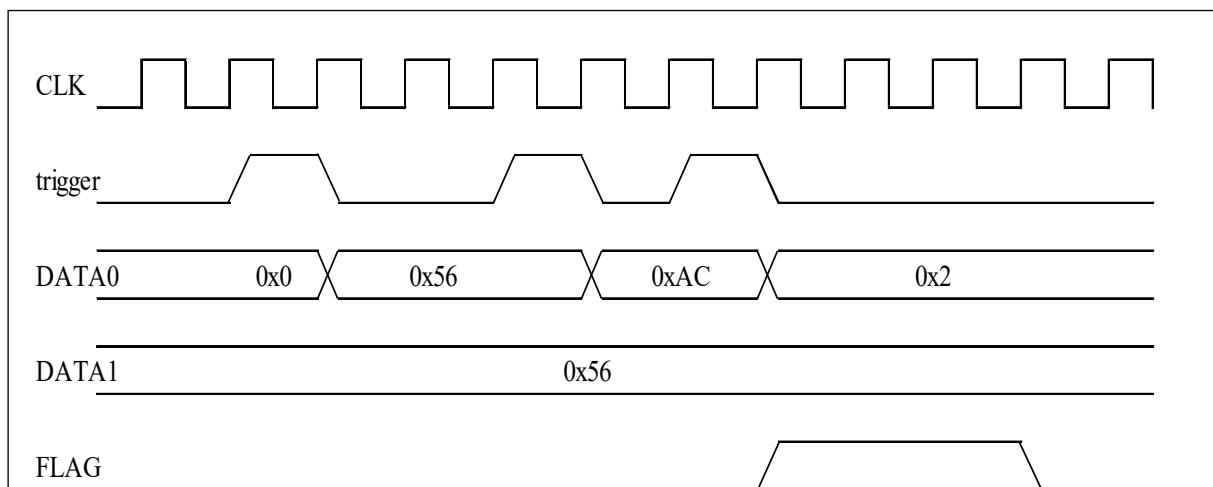


Figure 44-1 Hardware Triggered Boot Plus Mode Example Timing Diagram

44.2.4 Compare mode

Comparing the sizes of DATA0 and DATA1 and DATA0 and DATA2, you can select when DATA0 is greater than DATA1, DATA0 is less than DATA1, DATA0 is equal to DATA1, and when DATA0 is greater than DATA2, DATA0 is less than DATA2, and DATA0 is equal to DATA2 When a flag bit is generated

and an interrupt is generated. In the compare mode, you can select the data start comparison condition, compare after writing DATA0 or compare after writing any data register.

Example of comparison mode application:

1. The control register DCU_CTL selects the comparison mode, and the data width is 16bit. The comparison starts after writing to DATA0.
2. The interrupt condition selection register DCU_INTSEL generates a flag when DATA0>DATA1 is selected
3. 0xBBB and 0xAAA are written into DATA1 and DATA0 respectively. At this time, since DATA0>DATA1 is not satisfied, no flag is generated.
4. Write 0x8888 in DATA1, although DATA0>DATA1 is satisfied at this time, but since it is set to start comparison after writing DATA0, no flag is generated.
5. After DATA0 is written to 0x9999, the flag generation condition is satisfied, and the flag is generated.
6. Write the flag reset register DCU_FLAGCLR to clear the flag bit.

44.2.5 Interrupt and event signal output

The DCU has a variety of interrupts and event outputs for triggering other peripheral circuits for user selection. The control of interrupt and event output is controlled by the interrupt and event register (DCU_INTEVT). When the event signal needs to be output, the user needs to set the corresponding control bit of the interrupt and event register (DCU_INTEVT) to be valid. Each DCU unit outputs a DCU event signal, which are respectively DCU1~DCU8 in the event list. When the corresponding event needs to generate an interrupt when it occurs, the user needs to set the corresponding control bit of the interrupt and event register (DCU_INTEVT) to be valid, and set the INTEN bit of the control register (DCU_CTL) to 1. Each DCU unit outputs a DCU interrupt signal, which are respectively DCU1~DCU8 in the interrupt list.

44.2.6 Triangle wave output mode

In the triangular wave output mode, the DCU uses the timer as a trigger source to provide a digital-to-analog conversion module (DAC) with a continuously changing digital quantity, and the DAC converts these digital quantities into triangular wave output. The triangular wave output mode is set by the DCU control register (DCU_CTL). After the triangular wave output mode is set, the functions of the data registers DATA0~2 will change, and the data register DATA0 will be used as the digital data register output to the DAC and reset. DATA0 becomes a read-only register, and DATA0.DO (b0~b11) is a valid bit. DATA1 becomes the amplitude setting register, and DATA1.UPL(b16~b27) and DATA1.LWL(b0~b11) are valid bits. DATA2 becomes the step size setting register, and DATA2.STEP (b0~b11) are valid bits. Only DCU1~4 have triangle wave output mode.

The steps for using the triangle wave output mode are as follows

1. Set the MODE of the DCU control register (DCU_CTL) to 1000b
2. Set the upper and lower limits of the triangle wave amplitude. The upper limit is set by DATA1.UPL[11:0] bits, and the lower limit is set by DATA1.LWL[11:0]
3. Set the increment and decrement step value of triangular wave, set by DATA2.STEP[11:0]
4. Select the appropriate timer event or software trigger as the trigger source through the trigger source selection register (DCU_TRGSEL)
5. Configure the selected timer to generate a trigger time at a set interval
6. Start the timer, when the trigger event is generated for the first time, the value of DATA1.LWL is loaded into DATA0.DO and output to the DAC, and every time a trigger event is generated in the future, DATA0.DO will be incremented according to the step value set by DATA2.STEP
7. When DATA0.DO increases to the upper limit set by DATA1.UPL, it will automatically start to decrease according to the step value set by DATA2.STEP. When decremented to the lower limit set by DATA1.LWL, it starts to increment again
8. If an interrupt needs to occur when the triangle wave reaches the peak or valley point, please disable the interrupt enable bit (INTEN) in the DCU control register first, and then set the values of DATA0~DATA2, before starting the timer and generating trigger events. After clearing the DCU flag once, set the interrupt enable bit (INTEN) in the DCU control register to be valid, and then start the timer
9. Set the MODE of the DCU control register (DCU_CTL) to 0000b to exit the triangle wave output mode

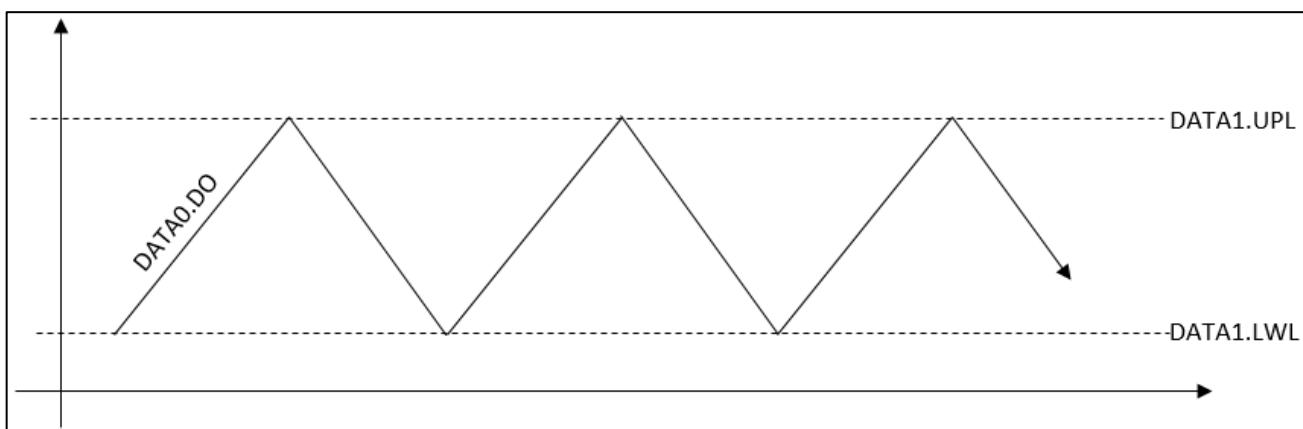
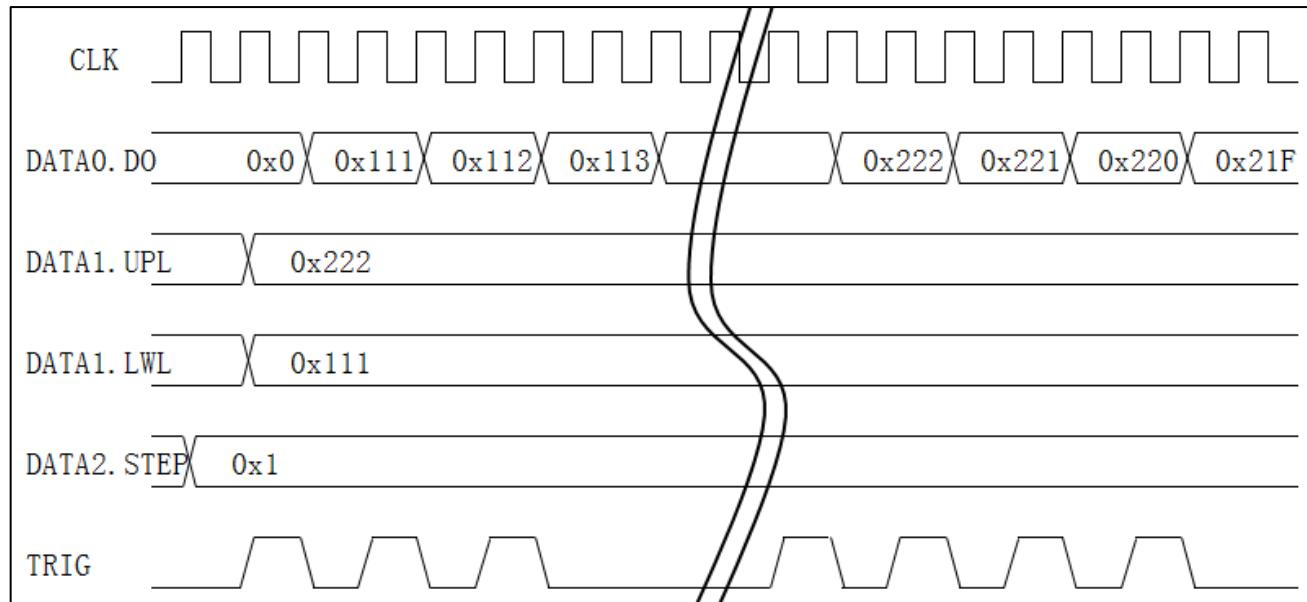


Figure 44-2 Triangle wave output mode

**Figure 44-3 Triangular wave output example timing diagram**

44.2.7 Incremental sawtooth output mode

Incremental sawtooth wave output mode is that the DCU uses the timer as a trigger source to provide a digital-to-analog conversion module (DAC) with a continuously changing digital quantity, and the DAC converts these digital quantities into a sawtooth wave output. The incremental sawtooth wave output mode is set through the DCU control register (DCU_CTL). After setting the incremental sawtooth wave output mode, the functions of the data registers DATA0~2 will change, and the data register DATA0 will be used as the digital data register output to the DAC and reset. DATA0 becomes a read-only register, and DATA.DO (b0~b11) is a valid bit. DATA1 becomes the amplitude setting register, and DATA1.UPL(b16~b27) and DATA1.LWL(b0~b11) are valid bits. DATA2 becomes the step size setting register, and DATA2.STEP (b0~b11) are valid bits. Only DCU1~4 have incremental ramp output mode.

The steps to use the incremental sawtooth output mode are as follows

1. Set the MODE of the DCU control register (DCU_CTL) to 1001b
2. Sets the upper and lower limits of the ramp amplitude. The upper limit is set by DATA1.UPL[11:0] bits, and the lower limit is set by DATA1.LWL[11:0]
3. Set the incremental step value of the sawtooth wave, set by DATA2.STEP[11:0]
4. Select the appropriate timer event or software trigger as the trigger source through the trigger source selection register (DCU_TRGSEL)
5. Configure the selected timer to generate a trigger time at a set interval
6. Start the timer, when the trigger event is generated for the first time, the value of DATA1.LWL is loaded into DATA0.DO and output to the DAC, and every time a trigger event is generated in the future, DATA0.DO will be incremented according to the step value set by DATA2.STEP
7. When DATA0.DO increments to the upper limit set by DATA1.UPL, it will automatically reload DATA1.LWL to DATA0.DO, so that it starts to increment from the lower limit set by DATA1.LWL
8. If an interrupt needs to occur when the sawtooth wave is reloaded, please disable the interrupt enable bit (INTEN) in the DCU control register first, then set the values of DATA0~DATA2, and clear the DCU once before starting the timer and generating trigger events. After the flag is set, set the interrupt enable bit (INTEN) in the DCU control register to be valid, and then start the timer
9. Set the MODE of the DCU control register (DCU_CTL) to 0000b to exit the sawtooth output mode
10. Set the MODE of the DCU control register (DCU_CTL) to 0000b to exit the sawtooth output mode

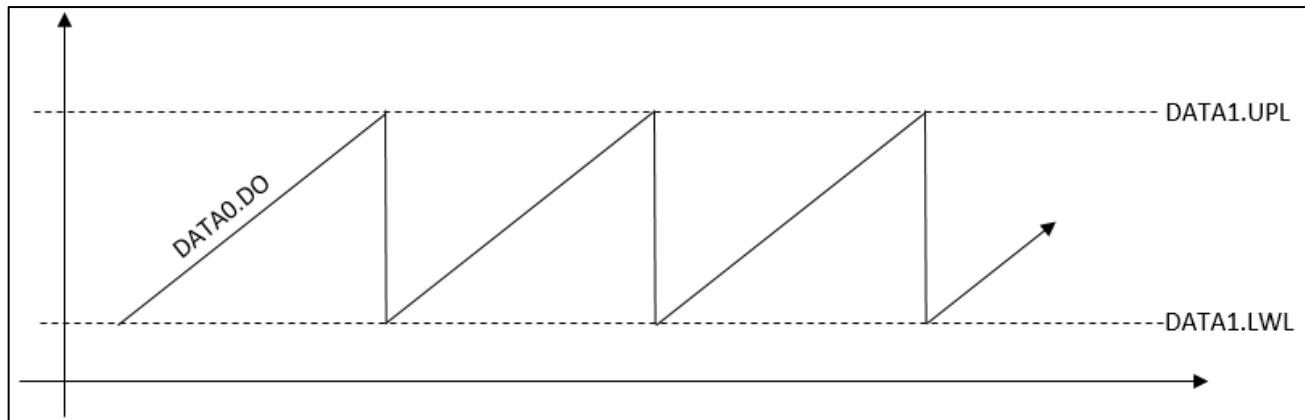


Figure 44-4 Incremental sawtooth output mode

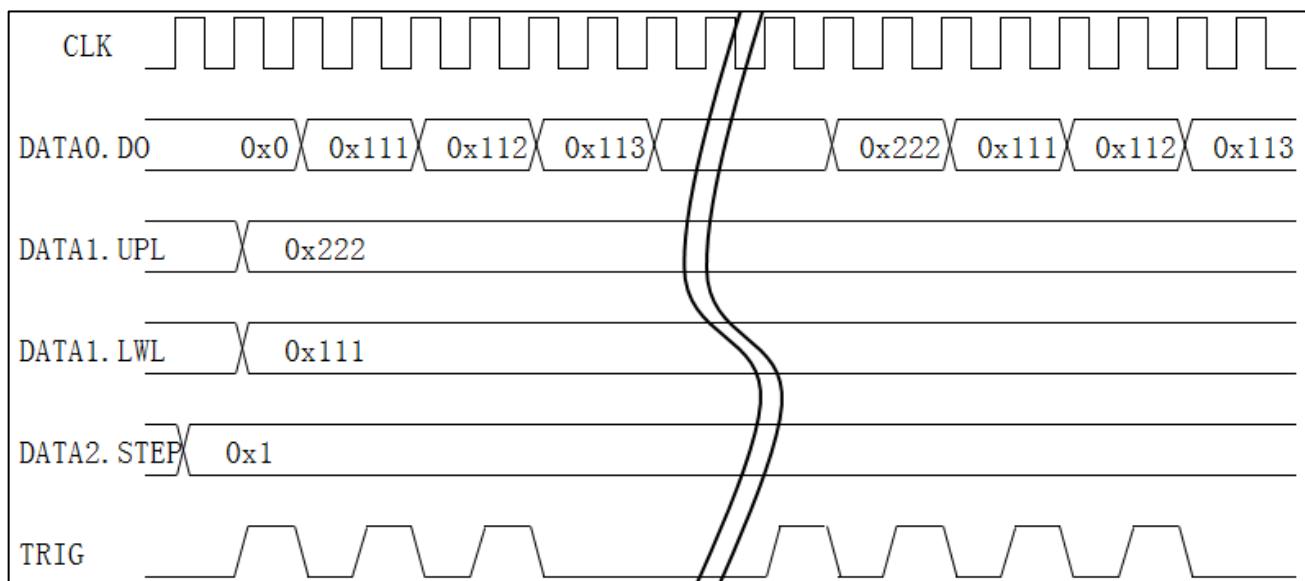


Figure 44-5 Incremental Ramp Output Example Timing Diagram

44.2.8 Decreasing sawtooth output mode

Decreasing sawtooth wave output mode is that the DCU uses the timer as a trigger source to provide a digital-to-analog conversion module (DAC) with a continuously changing digital quantity, and the DAC converts these digital quantities into a sawtooth wave output. The decreasing sawtooth wave output mode is set through the DCU control register (DCU_CTL). After setting the decreasing sawtooth wave output mode, the functions of the data registers DATA0~2 will change, and the data register DATA0 will be used as the digital data register output to the DAC and reset. DATA0 becomes a read-only register, and DATA.DO (b0~b11) is a valid bit. DATA1 becomes the amplitude setting register, and DATA1.UPL(b16~b27) and DATA1.LWL(b0~b11) are valid bits. DATA2 becomes the step size setting register, and DATA2.STEP (b0~b11) are valid bits. Only DCU1~4 have decreasing sawtooth output mode.

The steps to use the decreasing sawtooth output mode are as follows

1. Set the MODE of the DCU control register (DCU_CTL) to 1010b,
2. Sets the upper and lower limits of the ramp amplitude. The upper limit is set by DATA1.UPL[11:0] bits, and the lower limit is set by DATA1.LWL[11:0]
3. Set the step value of sawtooth wave decreasing, set by DATA2.STEP[11:0]
4. Select the appropriate timer event or software trigger as the trigger source through the trigger source selection register (DCU_TRGSEL)
5. Configure the selected timer to generate a trigger time at a set interval
6. Start the timer, when the trigger event is generated for the first time, the value of DATA1.UPL is loaded into DATA0.DO and output to the DAC, and every time a trigger event is generated in the future, DATA0.DO will be incremented according to the step value set by DATA2.STEP
7. When DATA0.DO is decremented to the lower limit set by DATA1.LWL, it will automatically reload DATA1.UPL to DATA0.DO, so that it starts to decrement from the lower limit set by DATA1.UPL
8. If an interrupt needs to occur when the sawtooth wave is reloaded, please disable the interrupt enable bit (INTEN) in the DCU control register first, then set the values of DATA0~DATA2, and clear the DCU once before starting the timer and generating trigger events. After the flag is set, set the interrupt enable bit (INTEN) in the DCU control register to be valid, and then start the timer
9. Set the MODE of the DCU control register (DCU_CTL) to 0000b to exit the sawtooth output mode

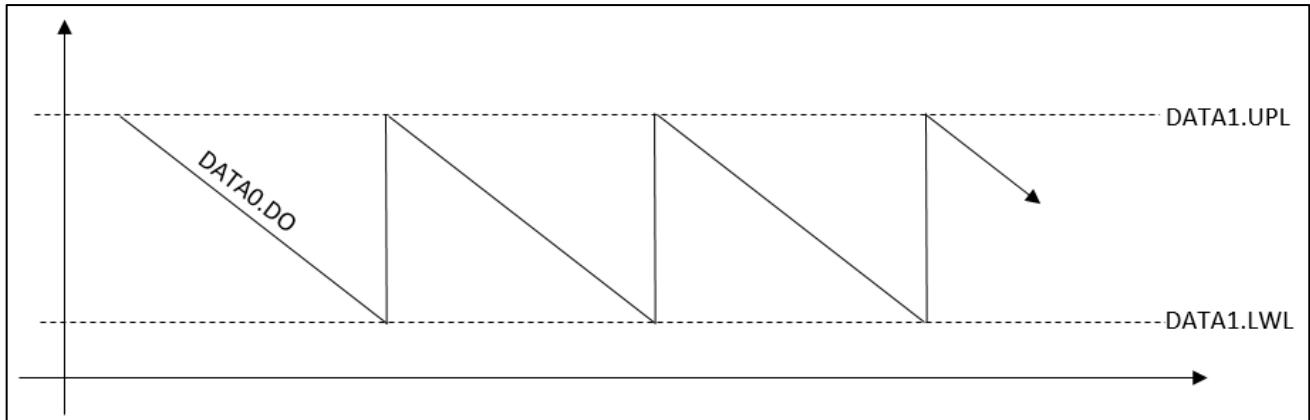


Figure 44-6 Decreasing sawtooth output mode

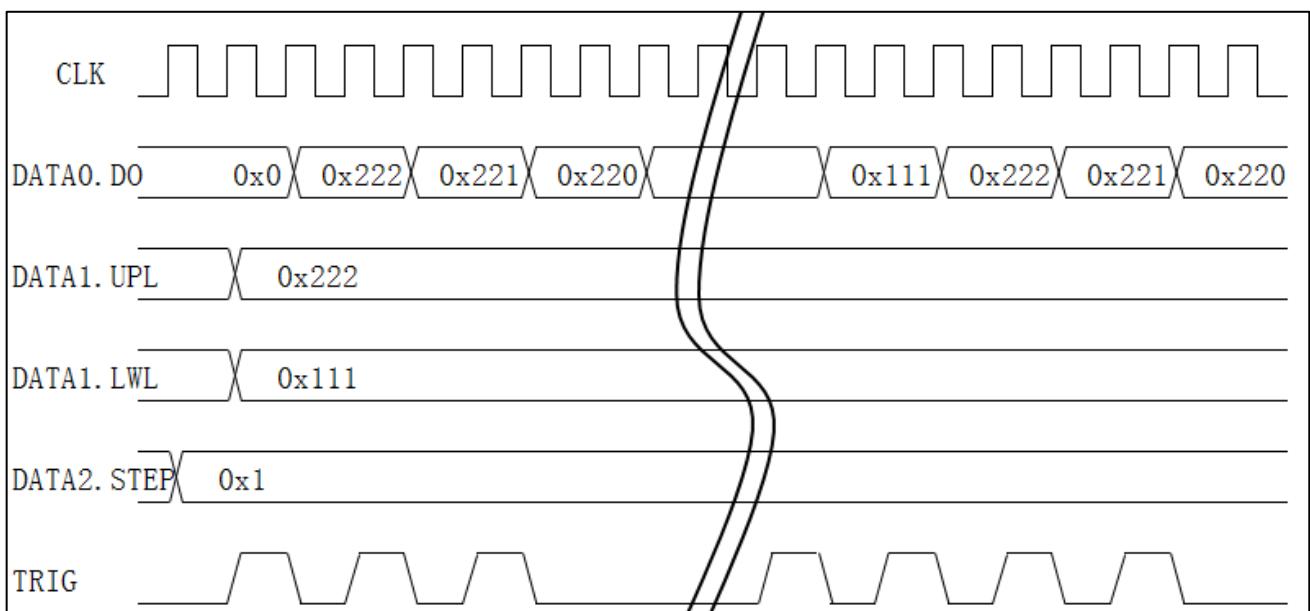


Figure 44-7 Decreasing Ramp Output Example Timing Diagram

44.3 Register description

Table 44-1 List of DCU registers

Unit 1

Name	Abbreviation	Note	Address
DCU1 Control Register	DCU1_CTL	Configure the action mode of the DCU	0x40056000
DCU1 flag register	DCU1_FLAG	DCU's result identification	0x40056004
DCU1 Data Register 0	DCU1_DATA0	Store operation data	0x40056008
DCU1 Data Register 1	DCU1_DATA1	Store operation data	0x4005600C
DCU1 Data Register 2	DCU1_DATA2	Store operation data	0x40056010
DCU1 Flag Reset Register	DCU1_FLAGCLR	Clear the result flag of the DCU	0x40056014
DCU1 Interrupt and Event Register	DCU1_INTEVT	Conditions for DCU Interrupts and Events	0x40056018
DCU1 trigger source selection register *	DCU1_TRGSEL	Select the trigger source for the hardware trigger start mode	0x40010804

*: Shares the same register as DCU5

Unit 2

Name	Abbreviation	Note	Address
DCU2 Control Register	DCU2_CTL	Configure the action mode of the DCU	0x40056400
DCU2 Flag Register	DCU2_FLAG	DCU's result identification	0x40056404
DCU2 Data Register 0	DCU2_DATA0	Store operation data	0x40056408
DCU2 Data Register 1	DCU2_DATA1	Store operation data	0x4005640C
DCU2 Data Register 2	DCU2_DATA2	Store operation data	0x40056410
DCU2 Flag Reset Register	DCU2_FLAGCLR	Clear the result flag of the DCU	0x40056414
DCU2 Interrupt and Event Register	DCU2_INTEVT	Conditions for DCU Interrupts and Events	0x40056418
DCU2 trigger source selection register *	DCU2_TRGSEL	Select the trigger source for the hardware trigger start mode	0x40010808

*: Shares the same register as DCU6

Unit 3

Name	Abbreviation	Note	Address
DCU3 Control Register	DCU3_CTL	Configure the action mode of the DCU	0x40056800
DCU3 flag register	DCU3_FLAG	DCU's result identification	0x40056804
DCU3 Data Register 0	DCU3_DATA0	Store operation data	0x40056808
DCU3 Data Register 1	DCU3_DATA1	Store operation data	0x4005680C
DCU3 Data Register 2	DCU3_DATA2	Store operation data	0x40056810
DCU3 Flag Reset Register	DCU3_FLAGCLR	Clear the result flag of the DCU	0x40056814
DCU3 Interrupt and Event Register	DCU3_INTEVT	Conditions for DCU Interrupts and Events	0x40056818
DCU3 trigger source selection register *	DCU3_TRGSEL	Select the trigger source for the hardware trigger start mode	0x4001080C

*: Shares the same register as DCU7

Unit 4

Name	Abbreviation	Note	Address
DCU4 Control Register	DCU4_CTL	Configure the action mode of the DCU	0x40056C00
DCU4 Flag Register	DCU4_FLAG	DCU's result identification	0x40056C04
DCU4 Data Register 0	DCU4_DATA0	Store operation data	0x40056C08
DCU4 Data Register 1	DCU4_DATA1	Store operation data	0x40056C0C
DCU4 Data Register 2	DCU4_DATA2	Store operation data	0x40056C10
DCU4 Flag Reset Register	DCU4_FLAGCLR	Clear the result flag of the DCU	0x40056C14
DCU4 Interrupt and Event Register	DCU4_INTEVT	Conditions for DCU Interrupts and Events	0x40056C18
DCU4 trigger source selection register *	DCU4_TRGSEL	Select the trigger source for the hardware trigger start mode	0x40010810

*: Shares the same register as DCU8

Unit 5

Name	Abbreviation	Note	Address
DCU5 Control Register	DCU5_CTL	Configure the action mode of the DCU	0x40057000
DCU5 flag register	DCU5_FLAG	DCU's result identification	0x40057004
DCU5 Data Register 0	DCU5_DATA0	Store operation data	0x40057008
DCU5 Data Register 1	DCU5_DATA1	Store operation data	0x4005700C
DCU5 Data Register 2	DCU5_DATA2	Store operation data	0x40057010
DCU5 Flag Reset Register	DCU5_FLAGCLR	Clear the result flag of the DCU	0x40057014
DCU5 Interrupt and Event Register	DCU5_INTEVT	Conditions for DCU Interrupts and Events	0x40057018
DCU5 trigger source selection register *	DCU1_TRGSEL	Select the trigger source for the hardware trigger start mode	0x40010804

*: Shares the same register as DCU1

Unit 6

Name	Abbreviation	Note	Address
DCU6 Control Register	DCU6_CTL	Configure the action mode of the DCU	0x40057400
DCU6 Flag Register	DCU6_FLAG	DCU's result identification	0x40057404
DCU6 Data Register 0	DCU6_DATA0	Store operation data	0x40057408
DCU6 Data Register 1	DCU6_DATA1	Store operation data	0x4005740C
DCU6 Data Register 2	DCU6_DATA2	Store operation data	0x40057410
DCU6 Flag Reset Register	DCU6_FLAGCLR	Clear the result flag of the DCU	0x40057414
DCU6 Interrupt and Event Register	DCU6_INTEVT	Conditions for DCU Interrupts and Events	0x40057418
DCU6 trigger source selection register *	DCU6_TRGSEL	Select the trigger source for the hardware trigger start mode	0x40010808

*: Shares the same register as DCU2

Unit 7

Name	Abbreviation	Note	Address
DCU7 Control Register	DCU7_CTL	Configure the action mode of the DCU	0x40057800
DCU7 Flag Register	DCU7_FLAG	DCU's result identification	0x40057804
DCU7 Data Register 0	DCU7_DATA0	Store operation data	0x40057808
DCU7 Data Register 1	DCU7_DATA1	Store operation data	0x4005780C
DCU7 Data Register 2	DCU7_DATA2	Store operation data	0x40057810
DCU7 Flag Reset Register	DCU7_FLAGCLR	Clear the result flag of the DCU	0x40057814
DCU7 Interrupt and Event Register	DCU7_INTEVT	Conditions for DCU Interrupts and Events	0x40057818
DCU7 trigger source selection register *	DCU7_TRGSEL	Select the trigger source for the hardware trigger start mode	0x4001080C

*: Shares the same register as DCU3

Unit 8

Name	Abbreviation	Note	Address
DCU8 Control Register	DCU8_CTL	Configure the action mode of the DCU	0x40057C00
DCU8 Flag Register	DCU8_FLAG	DCU's result identification	0x40057C04
DCU8 Data Register 0	DCU8_DATA0	Store operation data	0x40057C08
DCU8 Data Register 1	DCU8_DATA1	Store operation data	0x40057C0C
DCU8 Data Register 2	DCU8_DATA2	Store operation data	0x40057C10
DCU8 Flag Reset Register	DCU8_FLAGCLR	Clear the result flag of the DCU	0x40057C14
DCU8 Interrupt and Event Register	DCU8_INTEVT	Conditions for DCU Interrupts and Events	0x40057C18
DCU8 trigger source selection register *	DCU8_TRGSEL	Select the trigger source for the hardware trigger start mode	0x40010810

*: Shares the same register as DCU4

44.3.1 DCU Control Register (DCUx_CTL) (x=1~8)

Register description: This register is used to configure the action mode of the DCU

Reset value: 0x80000000 %%

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
INTEN	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[3:0]				
Bit	Marking	Place name	Function				Read and write								
b31	INTEN	interrupt enable	0: Interrupts are not allowed 1: Enable interrupt generation				R/W								
b30~b9	Reserved	-	Read 0 when reading, write 0 when writing				R/W								
b8	COMPTRG	Comparison mode triggers comparison timing	0: Compare after writing to DATA0 1: Compare after writing to DATA0 or DATA1 or DATA2				R/W								
b7~b6	Reserved	-	Read 0 when reading, write 0 when writing				R/W								
b5~b4	DATASIZE[1:0]	Add/Subtract mode, compare mode data size	00: 8bit 01: 16bit 10: 32bit				R/W								
b3~b0	MODE[2:0]	Action mode	0000: DCU is invalid 0001: Addition mode, the operation is performed after the data is written in the DATA1 register 0010: Subtraction mode, the operation is performed after the data is written into the DATA1 register 0011: Hardware trigger addition mode, triggered by other peripheral circuits to start the addition operation 0100: The hardware triggers the subtraction mode, which is triggered by other peripheral circuits to start the subtraction operation 0101: Compare Mode 1000*: Triangle wave output mode 1001*: Incremental sawtooth output mode 1010*: Decreasing sawtooth output mode other: set ban				R/W								

*: Only DCU1/DCU2/DCU3/DCU4 have this function

44.3.2 DCU Flag Register (DCUx_FLAG) (x=1~4)

Register description: This register generates the result flag of the DCU

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				FLAT_TOP	FLAG_BTM	FLAG_RLD	Reserved		FLAG_GT1	FLAG_EQ1	FLAG_LS1	FLAG_GT2	FLAG_EQ2	FLAG_LS2	FLAG_OP			
Bit	Marking	Place name	Function												Read and write			
b31~b12	Reserved	-	Read 0 when reading, write 0 when writing												R/W			
b11	FLAT_TOP	triangle point	crest	In the triangular wave output mode, it is set when the value of DATA0[11:0] reaches the maximum value, and it is cleared when the CLR_TOP bit of the DCU flag reset register DCU_FLAGCLR is written to 1.											R			
b10	FLAG_BTM	triangle point	valley	In the triangular wave output mode, it is set when the value of DATA0[11:0] reaches the minimum value, and it is cleared when the CLR_TOP bit of the DCU flag reset register DCU_FLAGCLR is written to 1.											R			
b9	FLAG_RLD	Sawtooth Overload		In incremental sawtooth wave output mode, set when the value of DATA0[11:0] reaches the maximum; In decreasing sawtooth output mode, it is set when the value of DATA0[11:0] reaches the minimum value. Cleared when the CLR_RLD bit of the DCU flag reset register DCU_FLAGCLR is written to 1												R		
b8~b7	Reserved	-	Read 0 when reading, write 0 when writing												R/W			
b6	FLAG_GT1	Greater than flag bit 1		In compare mode, set when DATA0>DATA1, cleared when the CLR_GT1 bit of the DCU flag reset register DCU_FLAGCLR is written to 1												R		
b5	FLAG_EQ1	Equal to flag bit 1		In compare mode, set when DATA0=DATA1, cleared when CLR_EQ1 bit of DCU_FLAGCLR is written to 1												R		
b4	FLAG_LS1	less than flag 1		In compare mode, set when DATA0 < DATA1, cleared when CLR_LS1 bit of DCU_FLAGCLR is written to 1												R		
b3	FLAG_GT2	Greater than flag bit 2		In compare mode, set when DATA0>DATA2, cleared when CLR_GT2 bit of DCU_FLAGCLR in DCU flag reset register is written to 1												R		
b2	FLAG_EQ2	equal to flag 2		In compare mode, set when DATA0=DATA2, cleared when CLR_EQ2 bit of DCU_FLAGCLR is written to 1												R		
b1	FLAG_LS2	less than flag 2		In compare mode, it is set when DATA0 < DATA2, and cleared when the CLR_LS2 bit of the DCU flag reset register DCU_FLAGCLR is written to 1												R		
b0	FLAG_OP	Operation flag		Addition, Subtraction and Triggered Addition and Triggered Subtraction mode, set when the addition produces overflow or subtraction produces underflow, and is cleared when the CLR_OP bit of the DCU flag reset register DCU_FLAGCLR is written to 1												R		

44.3.3 DCU Flag Register (DCUx_FLAG) (x=5~8)

Register description: This register generates the result flag of the DCU

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															
Bit	Marking	Place name	Function										Read and write		
b31~b7	Reserved	-	Read 0 when reading, write 0 when writing										R/W		
b6	FLAG_GT1	Greater than flag bit 1	In compare mode, set when DATA0>DATA1, cleared when the CLR_GT1 bit of the DCU flag reset register DCU_FLAGCLR is written to 1										R		
b5	FLAG_EQ1	Equal to flag bit 1	In compare mode, set when DATA0=DATA1, cleared when CLR_EQ1 bit of DCU_FLAGCLR is written to 1										R		
b4	FLAG_LS1	less than flag 1	In compare mode, set when DATA0 < DATA1, cleared when CLR_LS1 bit of DCU_FLAGCLR is written to 1										R		
b3	FLAG_GT2	Greater than flag bit 2	In compare mode, set when DATA0>DATA2, cleared when CLR_GT2 bit of DCU_FLAGCLR in DCU flag reset register is written to 1										R		
b2	FLAG_EQ2	equal to flag 2	In compare mode, set when DATA0=DATA2, cleared when CLR_EQ2 bit of DCU_FLAGCLR is written to 1										R		
b1	FLAG_LS2	less than flag 2	In compare mode, it is set when DATA0 < DATA2, and cleared when the CLR_LS2 bit of the DCU flag reset register DCU_FLAGCLR is written to 1										R		
b0	FLAG_OP	Operation flag	Addition, Subtraction and Triggered Addition and Triggered Subtraction mode, set when the addition produces overflow or subtraction produces underflow, and is cleared when the CLR_OP bit of the DCU flag reset register DCU_FLAGCLR is written to 1										R		

44.3.4 DCU data register (DCUx_DATAy) (x=1~8,y=0,1,2)

The functions of each data register in each mode are as follows

	DATA0	DATA1	DATA2
Additive mode	summand/stored result	addend	Store halving results
Trigger plus mode	summand/stored result	addend	Store halving results
Subtraction mode	Minuend/Store the result	Subtraction	Store halving results
Trigger minus mode	Minuend/Store the result	Subtraction	Store halving results
Compare mode	object to be compared	Compare object 1	Comparing object 2
Compare Mode (Window Compare)	object to be compared	window cap	window lower limit
Triangle wave output mode *	output data register	Amplitude setting register	step register
Incremental Ramp Output Mode *	output data register	Amplitude setting register	step register
Decreasing Ramp Output Mode *	output data register	Amplitude setting register	step register

*: Only DCU1/DCU2/DCU3/DCU4 have this function

Reset value: 0x00000000

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]															

In addition/subtraction mode, the addition/subtraction mode is triggered. In the comparison mode, DCUx_DATAy (x=1~8, y=0,1,2) is used to store the operation data.

Bit	Marking	Place name	Function	Read and write
b31~b0	DAT[31:0]	Operation data	Store operation 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, When DCU_CTL.DATASIZE=01, DATA[15:0] is valid data, DATA[31:0] is valid data when DCU_CTL.DATASIZE=10	R/W

Triangular wave output mode DCUx_DATA0 ($x=1\sim 4$) is used as output data register, b0~b11 are valid 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				DO[11:0]											

Bit	Marking	Place name	Function	Read and write
b31~b12	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b11~b0	DO[11:0]	Output data	Data output to DAC	R/W

The triangular wave output mode DCUx_DATA1 ($x=1\sim 4$) is used as the amplitude setting register, and b16~b27 and b0~b11 are valid bits

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved				UPL[11:0]											

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				LWL[11:0]											

Bit	Marking	Place name	Function	Read and write
b31~b28	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b27~b16	UPL[11:0]	Amplitude upper limit	Set the upper limit of the triangle wave amplitude	R/W
b15~b12	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b11~b0	LWL[11:0]	Amplitude lower limit	Sets the lower limit of the triangle wave amplitude	R/W

Triangle wave output mode DCUx_DATA2 ($x=1\sim 4$) is used as step register, b0~b11 are valid 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				STEP [11:0]											

Bit	Marking	Place name	Function	Read and write
b31~b12	Reserved	-	Read 0 when reading, write 0 when writing	R
b11~b0	STEP[11:0]	Step setting size	Set the increment and decrement step value of the triangle wave	R/W

Increment/decrement sawtooth wave output mode DCUx_DATA0 ($x=1\sim 4$) is used as output data register, b0~b11 are valid 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				DO[11:0]											

Bit	Marking	Place name	Function	Read and write
b31~b12	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b11~b0	DO[11:0]	Output data	Data output to DAC	R

Increment/decrement sawtooth wave output mode DCUx_DATA1 ($x=1\sim 4$) is used as amplitude setting register, b16~b27 and b0~b11 are valid bits

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved				UPL[11:0]											

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved				LWL[11:0]											

Bit	Marking	Place name	Function	Read and write
b31~b28	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b27~b16	UPL[11:0]	Amplitude upper limit	Sets the upper limit of the ramp amplitude	R/W
b15~b12	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b11~b0	LWL[11:0]	Amplitude lower limit	Sets the lower limit of the ramp amplitude	R/W

Increment/decrement sawtooth wave output mode DCUx_DATA2 ($x=1\sim 4$) is used as step register, b0~b11 are valid 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				STEP [11:0]											

Bit	Marking	Place name	Function	Read and write
b31~b12	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b11~b0	STEP[11:0]	Step setting size	Set the step value of the ramp wave increment or decrement	R/W

44.3.5 DCU Flag Reset Register (DCUx_FLAGCLR) (x=1~4)

Register description: This register is used to clear the result flag of the DCU

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				CLR_TOP	CLR_BTM	CLR_RLD	Reserved		CLR_GT1	CLR_EQ1	CLR_LS1	CLR_GT2	CLR_EQ2	CLR_LS2	CLR_OP		
Bit	Marking	Place name	Function												Read and write		
b31~b12	Reserved	-	Read 0 when reading, write 0 when writing												R/W		
b11	CLR_TOP	Clear the triangle peak point flag	When writing 1, clear the FLAG_TOP bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect												R/W		
b10	CLR_BTM	Clear the triangle wave valley point flag	When writing 1, clear the FLAG_BTM bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect												R/W		
b9	CLR_RLD	Clear the sawtooth reload flag	When writing 1, clear the FLAG_RLD bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect												R/W		
b8~b7	Reserved	-	Read 0 when reading, write 0 when writing												R/W		
b6	CLR_GT1	clear greater than flag bit 1	When writing 1, clear the FLAG_GT1 bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect												R/W		
b5	CLR_EQ1	clear equals flag bit 1	When writing 1, clear the FLAG_EQ1 bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect												R/W		
b4	CLR_LS1	clear less than flag bit 1	When writing 1, clear the FLAG_LS1 bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect												R/W		
b3	CLR_GT2	Clear greater than flag bit 2	When writing 1, clear the FLAG_GT2 bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect												R/W		
b2	CLR_EQ2	clear equals flag bit 2	When writing 1, clear the FLAG_EQ2 bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect												R/W		
b1	CLR_LS2	clear less than flag bit 2	When writing 1, clear the FLAG_LS2 bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect												R/W		
b0	CLR_OP	clear operation flag	When writing 1, clear the FLAG_OP bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect												R/W		

44.3.6 DCU Flag Reset Register (DCUx_FLAGCLR) (x=5~8)

Register description: This register is used to clear the result flag of the DCU

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															
Bit	Marking	Place name	Function										Read and write		
b31~b7	Reserved	-	Read 0 when reading, write 0 when writing										R/W		
b6	CLR_GT1	clear greater than flag bit 1	When writing 1, clear the FLAG_GT1 bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect										R/W		
b5	CLR_EQ1	clear equals flag bit 1	When writing 1, clear the FLAG_EQ1 bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect										R/W		
b4	CLR_LS1	clear less than flag bit 1	When writing 1, clear the FLAG_LS1 bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect										R/W		
b3	CLR_GT2	Clear greater than flag bit 2	When writing 1, clear the FLAG_GT2 bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect										R/W		
b2	CLR_EQ2	clear equals flag bit 2	When writing 1, clear the FLAG_EQ2 bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect										R/W		
b1	CLR_LS2	clear less than flag bit 2	When writing 1, clear the FLAG_LS2 bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect										R/W		
b0	CLR_OP	clear operation flag	When writing 1, clear the FLAG_OP bit of DCU_FLAG, writing 0 has no effect Reading this register bit has no effect										R/W		

44.3.7 DCU Interrupt and Event Register (DCUx_INTEVTSEL) (x=1~4)

Register description: This register can select the conditions under which the DCU generates an interrupt and outputs an event signal

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 SEL_TOP SEL_BTM SEL_RLD SEL_WIN[1:0] SEL_GT1 SEL_EQ1 SEL_LS1 SEL_GT2 SEL_EQ2 SEL_LS2 SEL_OP																
Bit	Marking	Place name	Function													Read and write
b31~b12	Reserved	-	Read 0 when reading, write 0 when writing													R/W
b11	SEL_TOP	Triangle wave crest point condition selection	1: In triangular wave output mode, when DATA0[11:0] reaches the maximum value, interrupt and output event signal will be generated 0: In triangle wave and output mode, when DATA0[11:0] reaches the maximum value, no interrupt and output event signal will be generated													R/W
b10	SEL_BTM	Triangle valley wave point condition selection	1: In triangular wave output mode, when DATA0[11:0] reaches the minimum value, interrupt and output event signal will be generated 0: In triangular wave output mode, when DATA0[11:0] reaches the minimum value, no interrupt and output event signal will be generated													R/W
b9	SEL_RLD	Sawtooth overload condition selection	1: In the incremental sawtooth wave output mode, when DATA0[11:0] reaches the maximum value, an interrupt and output event signal will be generated; in the decremented sawtooth wave output mode, when DATA0[11:0] reaches the minimum value, an interrupt and an output event will be generated Signal 0: Do not generate interrupt and output event signal in increment/decrement ramp output mode													R/W
b8~b7	SEL_WIN[1:0]	Window comparison condition selection	In the comparison mode, when the window comparison conditions set by SEL_WIN are met, an interrupt and an output event signal will be generated. When the SEL_WIN setting is valid, the interrupt and output event signal will not be generated when other comparison conditions are met. 00: No window comparison interrupt and output event signal are generated. Under this setting, other interrupt and event signal generation conditions are selected by b1~b6 of this register. 01: Generate interrupt and output event signal when DATA0 data is in the window, that is, DATA2≤DATA0≤DATA1 10: When the DATA0 data is outside the window, the interrupt and output event signal is generated, that is, DATA0>DATA1 or DATA0<DATA2 11: No interrupt or event signal is generated in compare mode													R/W
b6	SEL_GT1	Greater than condition select 1	1: In comparison mode and SEL_WIN=00, when DATA0>DATA1, interrupt and output event signal are generated 0: In comparison mode and SEL_WIN=00, no interrupt and output event signal will be generated when DATA0>DATA1 This bit is invalid when SEL_WIN≠00													R/W
b5	SEL_EQ1	Equal to condition select 1	1: In compare mode and SEL_WIN=00, interrupt and output event signal are generated when DATA0=DATA1 0: In compare mode and SEL_WIN=00, no interrupt and output event signal will be generated when DATA0=DATA1 This bit is invalid when SEL_WIN≠00													R/W
b4	SEL_LS1	Less than condition select 1	1: In comparison mode and SEL_WIN=00, when DATA0 < DATA1, interrupt and output event signal are generated 0: In comparison mode and SEL_WIN=00, no interrupt and output event signal will be generated when DATA0 < DATA1 This bit is invalid when SEL_WIN≠00													R/W
b3	SEL_GT2	Greater than condition select 2	1: In compare mode and SEL_WIN=00, when DATA0>DATA2, interrupt and output event signal will be generated 0: In comparison mode and SEL_WIN=00, no interrupt and output event signal will be generated when DATA0>DATA2													R/W

This bit is invalid when SEL_WIN≠00					
b2	SEL_EQ2	Equal to conditional choice 2	1: In compare mode and SEL_WIN=00, interrupt and output event signal are generated when DATA0=DATA2 0: In compare mode and SEL_WIN=00, no interrupt and output event signal will be generated when DATA0=DATA2 This bit is invalid when SEL_WIN≠00	R/W	
b1	SEL_LS2	Less than condition select 2	1: In comparison mode and SEL_WIN=00, interrupt and output event signal are generated when DATA0 < DATA2 0: In comparison mode and SEL_WIN=00, no interrupt and output event signal will be generated when DATA0 < DATA2 This bit is invalid when SEL_WIN≠00	R/W	
b0	SEL_OP	Operation condition selection	1: In addition and subtraction modes, interrupt and output event signals are generated when the operation result overflows or underflows 0: In addition and subtraction mode, no interrupt and output event signal will be generated when the operation result overflows or underflows	R/W	

44.3.8 DCU Interrupt and Event Register (DCUx_INTEVTSEL) (x=5~8)

Register description: This register can select the conditions under which the DCU generates an interrupt and outputs an event signal

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										SEL_WIN[1:0]	SEL_GT1	SEL_EQ1	SEL_LS1	SEL_GT2	SEL_EQ2	SEL_LS2	SEL_OP

Bit	Marking	Place name	Function	Read and write
b31~b9	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b8~b7	SEL_WIN[1:0]	Window comparison condition selection	In the comparison mode, when the window comparison conditions set by SEL_WIN are met, an interrupt and an output event signal will be generated. When the SEL_WIN setting is valid, the interrupt and output event signal will not be generated when other comparison conditions are met. 00: No window comparison interrupt and output event signal are generated. Under this setting, other interrupt and event signal generation conditions are selected by b1~b6 of this register. 01: Generate interrupt and output event signal when DATA0 data is in the window, that is, DATA2≤DATA0≤DATA1 10: When the DATA0 data is outside the window, the interrupt and output event signal is generated, that is, DATA0>DATA1 or DATA0<DATA2 11: No interrupt or event signal is generated in compare mode	R/W
b6	SEL_GT1	Greater than condition select 1	0: In comparison mode and SEL_WIN=00, no interrupt and output event signal will be generated when DATA0>DATA1 This bit is invalid when SEL_WIN≠00	R/W
b5	SEL_EQ1	Equal to condition select 1	1: In compare mode and SEL_WIN=00, interrupt and output event signal are generated when DATA0=DATA1 0: In compare mode and SEL_WIN=00, no interrupt and output event signal will be generated when DATA0=DATA1 This bit is invalid when SEL_WIN≠00	R/W
b4	SEL_LS1	Less than condition select 1	1: In comparison mode and SEL_WIN=00, when DATA0 < DATA1, interrupt and output event signal are generated 0: In comparison mode and SEL_WIN=00, no interrupt and output event signal will be generated when DATA0 < DATA1 This bit is invalid when SEL_WIN≠00	R/W
b3	SEL_GT2	Greater than condition select 2	1: In compare mode and SEL_WIN=00, when DATA0>DATA2, interrupt and output event signal will be generated 0: In comparison mode and SEL_WIN=00, no interrupt and output event signal will be generated when DATA0>DATA2 This bit is invalid when SEL_WIN≠00	R/W
b2	SEL_EQ2	Equal to conditional choice 2	1: In compare mode and SEL_WIN=00, interrupt and output event signal are generated when DATA0=DATA2 0: In compare mode and SEL_WIN=00, no interrupt and output event signal will be generated when DATA0=DATA2 This bit is invalid when SEL_WIN≠00	R/W
b1	SEL_LS2	Less than condition select 2	1: In comparison mode and SEL_WIN=00, interrupt and output event signal are generated when DATA0 < DATA2 0: In comparison mode and SEL_WIN=00, no interrupt and output event signal will be generated when DATA0 < DATA2 This bit is invalid when SEL_WIN≠00	R/W
b0	SEL_OP	Operation condition selection	1: In addition and subtraction modes, interrupt and output event signals are generated when the operation result overflows or underflows 0: In addition and subtraction mode, no interrupt and output event signal will be generated when the operation result overflows or underflows	R/W

44.3.9 DCU trigger source selection register (DCUx_TRGSEL) (x=1~4)

Register description: After the DCU selects the hardware trigger start mode, the triangle wave output mode or the increment/decrement sawtooth wave output mode (only for DCU1~4), write the number of the event to be triggered into this register. When the peripheral circuit event corresponding to the number occurs, the DCU will be triggered by this event to start and perform operations. DCU1 will share the register DCU1_TRGSEL with DCU5, that is, when DCU1_TRGSEL writes the event number, when the event of this number occurs, both DCU1 and DCU5 will be triggered at the same time; DCU2 will share the register DCU2_TRGSEL with DCU6, that is, when DCU2_TRGSEL writes the event number, when the numbered event occurs, DCU2 and DCU6 will be triggered at the same time; DCU3 will share the register DCU3_TRGSEL with DCU7, that is, when the event number is written to DCU3_TRGSEL, when the event occurs, DCU3 and DCU7 will be triggered at the same time; DCU4 and DCU8 will share the register DCU4_TRGSEL, that is, when DCU4_TRGSEL is written Event number, when the event occurs, both DCU4 and DCU8 will be triggered.

Reset value: 0x000001FF

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]							

Bit	Marking	Place name	Function	Read and write
b31	COMEN[1]	public trigger enable	0: Disable the common trigger event of AOS_COMTRG2 to trigger the DCU 1: Allow the common trigger event of AOS_COMTRG2 to trigger the DCU See the Automatic Operating System (AOS) chapter for details	R/W
b30	COMEN[0]	public trigger enable	0: Disable the common trigger event of AOS_COMTRG1 to trigger the DCU 1: Allow the common trigger event of AOS_COMTRG1 to trigger the DCU See the Automatic Operating System (AOS) chapter for details	R/W
b29~b9	Reserved	-	Read 0 when reading, write 0 when writing	R/W
b8~b0	TRGSEL[8:0]	Trigger source selection	Write the number of the peripheral circuit event to be selected	R/W

45 Mathematical Operations Unit (MAU)

45.1 Introduction

The Mathematical Operation Unit (MAU) is a hardware-accelerated operation module that includes two types of operations, the square root operation and the sine operation, and supports the square root and sine operations of fixed-point numbers. The sine function supports $360^\circ/2^{12}$ arithmetic precision.

45.2 Functional description

45.2.1 Square root operation

The square root operation supports 32-bit fixed-point input, and the result of the operation is 17-bit fixed-point output. Depending on the size of the input data, the operation execution cycle will vary.

45.2.1.1 Operating procedures

When the interrupt is invalid (MAU_CSR.INTEN=0), the operation process of the square root operation is as follows; when the interrupt is valid, the user can customize the interrupt processing process according to the process:

- 1) Write squared data in data input register 0 (MAU_DTR0)
- 2) Set the start bit of the control status register to 1 (MAU_CSR.START=1)
- 3) After the operation starts, the status bit (MAU_CSR.BUSY) of the control status register will be set until the operation is completed, and the bit will be automatically cleared. Expect the BUSY bit to be cleared to indicate that the operation is complete
- 4) Read operation result from result output register 0 (MAU_RTR0)
- 5) Repeat the above process 1)~4)

Regarding process 3), after MAU_CSR.START is set, the BUSY bit starts to become high, and automatically becomes 0 after 8 to 16 operation clock cycles (the operation time varies according to the size of the input data). When using the square root operation module, the user can flexibly judge the current operation state according to the change of this bit.

45.2.1.2 Result processing

The operation result of the square root operation is a 17-bit data output, and the lowest-order result is obtained by rounding. For example, when the input data is 0x0000000CH, the result is 0x00003H; when the input data is 0x0000000DH, the result is 0x00004H.

45.2.1.3 Fixed-point processing

When the input data is a fixed-point decimal, and the format of the input and output data needs to be kept uniform, the user can process the output data correspondingly by controlling the shift control bit (MAU_CSR.SHIFT) of the status register.

For example, when using the Q14.2 format for fixed-point arithmetic, you can set MAU_CSR.SHIFT=1 to make the output result also follow the Q14.2 format. The fixed-point number 6.25 (Q14.2 format is 0x00000019H) will output bit 5 (0x00000005H) before shifting, and the result after shifting 1 bit is 2.5 (Q14.2 format is 0x0000000AH).

45.2.2 Sine operation

The sine operation supports 12-bit data input, and the operation result is a 16-bit signed number output. Among them, the 12-bit data input represents the angle of the Cartesian coordinate system to be operated; in the 16-bit data output, the highest bit represents the positive or negative of the result.

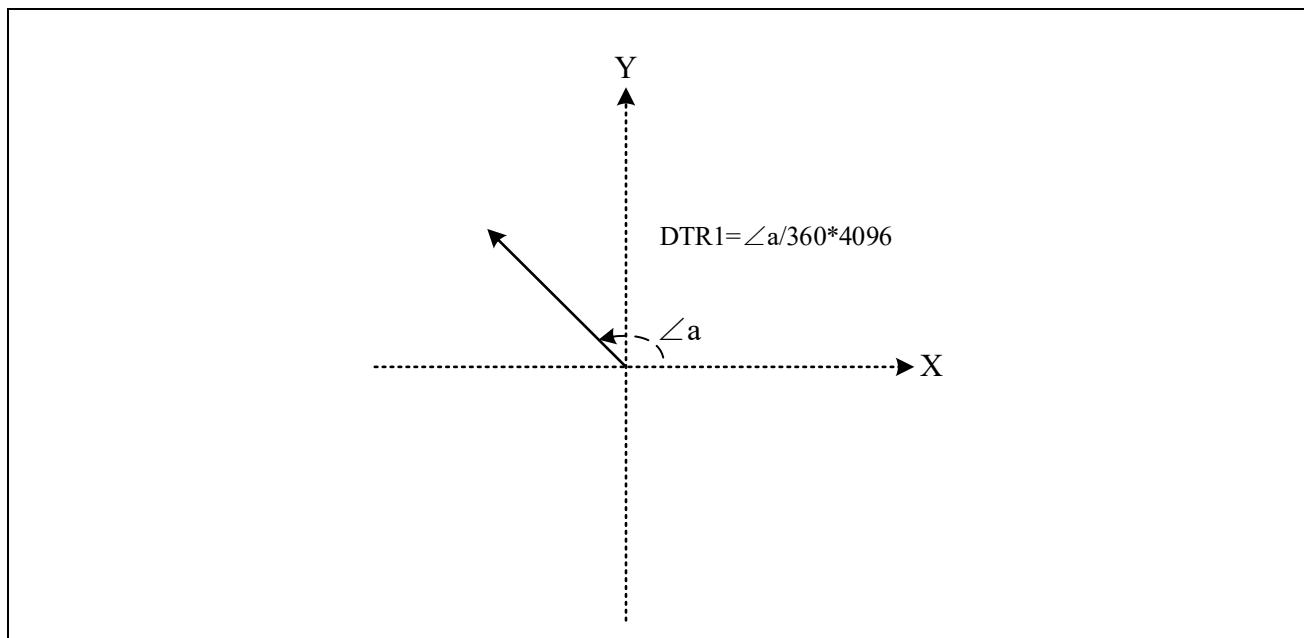


Figure 45-1 Schematic diagram of sine operation angle value

45.2.2.1 Operating procedures

The operation flow of sine operation is as follows:

- 1) Write the angle value in data input register 1 (MAU_DTR1)
- 2) Read operation result from result output register 1 (MAU_RTR1) after 1 cycle
- 3) Repeat the above process 1)~2)

45.3 Interrupt and Event Description

45.3.1 Interrupt output

After each square root operation is completed, a corresponding operation completion event will be generated. If the interrupt control bit is enabled (MAU_CSR.INTEN=1), the corresponding interrupt request signal (MAU_SQRT) will also be generated.

45.3.2 Event output

After each square root operation is completed, a corresponding operation completion event will be generated, and the corresponding event request signal (MAU_SQRT) will also be output, which can be used to selectively trigger other modules.

45.4 Register description

Table 45-1 Shown is the register list of the MAU module.

BASE ADDR: 0x40055000H

Table 45-1 MAU Register List

Register name	Symbol	Offset	Bit width	Reset value
Control state	MAU_CSR	0x0000h	32	0x00000000h
Data Input Register 0	MAU_DTR0	0x0004h	32	0x00000000h
result output register 0	MAU_RTR0	0x000Ch	32	0x00000000h
Data Input Register 1	MAU_DTR1	0x0010h	32	0x00000000h
result output register 1	MAU_RTR1	0x0014h	32	0x00000000h

45.4.1 Control Status Register (MAU_CSR)

Reset value: 0x00000000h

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
-	-	-		SHIFT[4:0]				-	-	-	-	BUSY	-	INTEN	START
Bit	Marking		Place name			Function								Read and write	
b31~b13	Reserved		-			Read as "0", write as "0"								R/W	
b12~b8	SHIFT[4:0]		square root result shift			0000: The operation result is output without shifting 00001: The operation result is left shifted by 1 bit and then output 00010: The operation result is left shifted by 2 bits and output ... 10000: The operation result is left shifted by 16 bits and output Please do not set other values								R/W	
b7~b4	Reserved		-			Read as "0", write as "0"								R/W	
b3	BUSY		square root operation status			0: The square root operation is not started or the operation is completed 1: The square root calculation is in progress								R/W	
b2	Reserved		-			Read as "0", write as "0"								R/W	
b1	INTEN		Square root operation interrupt enable			0: No interrupt will be generated after the square root operation is completed 1: An interrupt is generated after the square root operation is completed								R/W	
b0	START		square root operation starts			0: Writing zero to this bit is invalid 1: The square root operation starts Note: This bit internally generates a START request when 1 is written, and is always 0 when read								R/W	

45.4.2 Data Input Register 0 (MAU_DTR0)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
SQRT_DIN[31:16]															

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

Bit	Marking	Place name	Function	Read and write
b31~b0	SQRT_DIN[31:0]	squared	square root input	R/W

45.4.3 Result output register 0 (MAU_RTR0)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reserved														SQRT_DOUT[16]	

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

Bit	Marking	Place name	Function	Read and write
b31~b17	Reserved	-	Read as "0", write as "0"	R/W
b16~b0	SQRT_DOUT[16:0]	square root result	square root result output	R/W

45.4.4 Data Input Register 1 (MAU_DTR1)

Reset value: 0x00000000h

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		SIN_DIN[11:0]													

Bit	Marking	Place name	Function	Read and write
b31~b12	Reserved	-	Read as "0", write as "0"	R/W
b11~b0	SIN_DIN[11:0]	Angle setting	Cartesian coordinate system angle input	R/W

45.4.5 Result Output Register 1 (MAU_RTR1)

Reset value: 0x00000000h

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
SIN_DOUT[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b16	Reserved	-	Read as "0", write as "0"	R/W
b15~b0	SIN_DOUT[15:0]	Sine operation result	Sine operation result output	R/W

46 Filter Math Accelerator (FMAC)

46.1 Introduction

Filter Math Accelerator (FMAC) is a hardware acceleration module for FIR filter computation. This module can perform up to 16-order FIR digital filtering with configurable orders. Built-in 16x16 bit multiplier, 32+5bit adder, users can customize the output data precision. This series is equipped with 4 FMAC modules.

46.2 Basic block diagram

The basic features of this module are as follows:

- Maximum 16-order FIR filter, the order can be configured
- Configurable filter coefficients
- 16-bit signed number input and output data, 16-bit signed number filter coefficient
- 16x16 bit multiplier
- 37bit adder
- Provide interrupt signal and status query
- AHB slave interface configuration parameters and input and output data

Its basic block diagram is as follows:

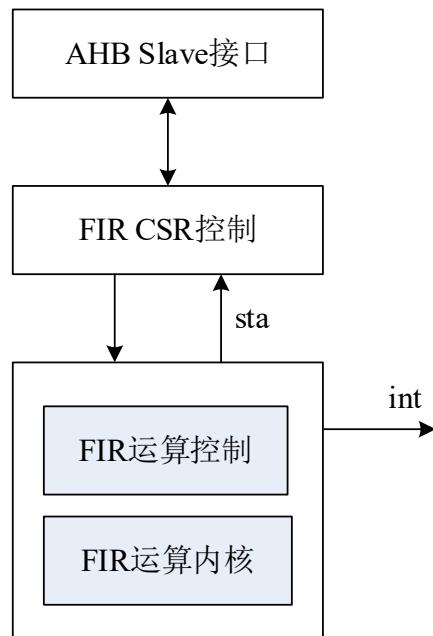


Figure 46-1 FIR basic block diagram

When the 16th-order FIR filter is selected, the FIR operation control and operation core needs 17 clock cycles to process one data, and can process a maximum input data stream of 8.8MHz at a working clock frequency of 150MHz. Different orders require different processing time. The smaller the order, the faster the processing speed.

46.3Operating procedures

This module requires the following steps to use:

- 1) Set the enable bit (FMAC_ENR.FIREN) to 1;
- 2) Configure the filter stage (FMAC_CTR.STAGE_NUM[4:0]) and output the effective result shift number (FMAC_CTR.SHIFT[4:0]);
- 3) If you need to use an interrupt, configure the interrupt enable bit (FMAC_IER.INTEN) to 1;
- 4) Write filter coefficients (with N order, just configure FMAC_COR0~FMAC_CORN);
- 5) write pending data (FMAC_DTR);
- 6) Waiting for completion interrupt or query completion status (FMAC_STR.READY) is 1;
- 7) read result (FAMC_RTR0 or FAMC_RTR1);
- 8) Repeat 5)~7) until all data is processed.

46.4Module enable

The module enable bit (FMAC_ENR.FIREN) has two functions, one is to enable the entire module, only when it is 1, the module is allowed to work; the second is the soft reset function of the module, when a certain order and coefficient After the configuration of , you need to change to another configuration of order and coefficient, then you need to set FIREN to 0, and clear the residual information from the previous calculation. Then follow the operation process.

When FIREN is cleared to 0, the configuration parameters will not be cleared, only the internal registers and intermediate results will be cleared.

46.5Coefficient normalization

If the output result only takes 16 bits, then in the calculation process and the calculation result, a total of 20 bits are truncated, so if the input data involved in the calculation is too small, the output result may be 0, or very small number (loss of precision). Therefore, it is necessary to normalize the filter coefficients, that is, the maximum value (if it is a positive number) is normalized to 32767 (if it is a negative number, it is normalized to -32768), and the other coefficients are multiplied by the same ratio.

For example, there are 5 coefficients of 10, 13, 35, 96, 42, which are to be normalized. First find the maximum value, which is 96, then divide 32767 by the maximum value, $32767/96=341.3$, and take the value 341, which is the normalized multiple, and other coefficients must be multiplied by this multiple, which is the normalized multiple. The number after 1.

So the normalized coefficients are: 3410, 4433, 11935, 32736, 14322.

46.6 Interrupt and Event Description

46.6.1 Interrupt output

After each operation is completed, a corresponding operation completion event will be generated. If the interrupt control bit (FMAC_IER.INTEN=1) is enabled, the corresponding interrupt request signal (FMAC_m_FIR, m=1~4) will also be generated.

46.6.2 Event output

After each operation is completed, a corresponding operation completion event will be generated, and the corresponding event request signal (FMAC_m_FIR, m=1~4) will also be output, which can be used to select and trigger other modules.

46.7 Register description

Table 46-1 Shown is the register list of the FMAC module.

BASE ADDR:

0x40058000H (U1), 0x40058400H (U2),

0x40058800H (U3), 0x40058C00H (U4)

Table 46-1 FMAC register list

Register name	Symbol	Offset	Bit width	Reset value
Module Enable Register	FMAC_ENR	0x0000h	32	0x00000000h
Basic control register	FMAC_CTR	0x004h	32	0x00000010h
Interrupt control register	FMAC_IER	0x008h	32	0x00000000h
data input register	FMAC_DTR	0x00Ch	32	0x00000000h
Filter Coefficient Register 0	FMAC_COR0	0x020h	32	0x00000000h
Filter Coefficient Register 1	FMAC_COR1	0x024h	32	0x00000000h
...
Filter Coefficient Register 16	FMAC_COR16	0x060h	32	0x00000000h
result output register 0	FMAC_RTR0	0x010h	32	0x00000000h
result output register 1	FMAC_RTR1	0x014h	32	0x00000000h
Operation Status Register 1	FMAC_STR	0x018h	32	0x00000000h

46.7.1 Module Enable Register (FMAC_ENR)

Reset value: 0x00000000h

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															
Bit	Marking	Place name	Function				Read and write								
b31~b1	Reserved	-	Read as "0", write as "0"				R/W								
b0	FMACEN	module enable	0: Disable 1: Enable, that is, allow work Note: This signal is used as the soft reset signal of this module, that is, when the filter order and filter coefficient need to be changed, the bit signal needs to be set to 0 first, and then the position is set to 1.				R/W								

46.7.2 Basic Control Register (FMAC_CTR)

Reset value: 0x00000000h

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
-															
Bit	Marking	Place name	Function	Read and write											
b31~b13	Reserved	-	Read as "0", write as "0"	R/W											
The right shift number of the filtering result, which controls the result output by MAU_RTR1 00000: Do not shift, that is, select [31:0] of the operation result to MAU_RTR1 00001: Shift right by 1 bit, that is, select [32:1] of the operation result to MAU_RTR1 00010: Shift 2 bits to the right, that is, select [33:2] of the operation result to MAU_RTR1 10100: Shift 20 bits to the right, that is, select [35:20] of the operation result to MAU_RTR1, and enter the sign bit into the high-order bit 10101: Shift right by 21 bits, that is, select [36:21] of the operation result to MAU_RTR1, and enter the sign bit into the high-order bit Other values: Does not shift															
b12~b8	SHIFT[4:0]	filter result shift		R/W											
b7~b5	Reserved	-	Read as "0", write as "0"	R/W											
b4~b0	STAGE_NUM[4:0]	Filter order setting	Set the filter order of the filter	R/W											

46.7.3 Interrupt Control Register (FMAC_IER)

Reset value: 0x00000000h

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															INTEN
Bit	Marking	Place name				Function				Read and write					
b31~b1	Reserved	-				Read as "0", write as "0"				R/W					
b0	INTEN	interrupt enable				0: No interrupt will be generated after the operation is completed 1: An interrupt is generated after the square root operation is completed				R/W					

46.7.4 Data Input Register (FMAC_DTR)

Reset value: 0x00000000h

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
FMAC_DIN[15:0]															
Bit	Marking	Place name				Function				Read and write					
b31~b16	Reserved	-				Read as "0", write as "0"				R/W					
b15~b0	FMAC_DIN[15:0]	data input				FIR data input				R/W					

46.7.5 Filter coefficient register (FMAC_COR0~16)

Reset value: 0x00000000h

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
FMAC_CIN[15:0]															
Bit	Marking	Place name				Function				Read and write					
b31~b16	Reserved	-				Read as "0", write as "0"				R/W					
b15~b0	FMAC_CIN[15:0]	filter coefficient				Filter coefficient input				R/W					

46.7.6 Result Output Register 0 (FMAC_RTR0)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
FMAC_DOUT0[31:16]															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FMAC_DOUT0[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b0	FMAC_DOUT0[31:0]	result output	The upper 32-bit result output of the accumulator result	R/W

46.7.7 Result Output Register 1 (FMAC_RTR1)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
FMAC_DOUT1[31:16]															

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FMAC_DOUT1[15:0]															

Bit	Marking	Place name	Function	Read and write
b31~b0	FMAC_DOUT1[31:0]	result output	Lower 32-bit result output of accumulator result	R/W

46.7.8 Operation Status Register (FMAC_STR)

Reset value: 0x00000000h

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
READY	Reserved														

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Reserved															

Bit	Marking	Place name	Function	Read and write
b31	READY	Completion bit	0: Computation is not done, result is not available 1: Computation complete, result available Note: This register is automatically cleared after reading	R/W
b30~b0	Reserved	-	Read as "0", write as "0"	R/W

47 Debug controller (DBG)

This product refers to 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

47.1 Introduction

The core of this MCU is Cortex™-M4F, which contains hardware for advanced debug functions and supports Embedded Trace Macrocell (ETM). With these debugging features, you can stop the kernel when fetching fingers (instruction breakpoints) or accessing data (data breakpoints). When the kernel stops, you can query the internal state of the kernel and the external state of the system. After the query completes, the kernel and system are restored and program execution is restored.

Provides two debugging interfaces:

- Serial Debugging Tracking Interface SWD
- Parallel Debug Trace Interface JTAG

47.2 DBGC System Block Diagram

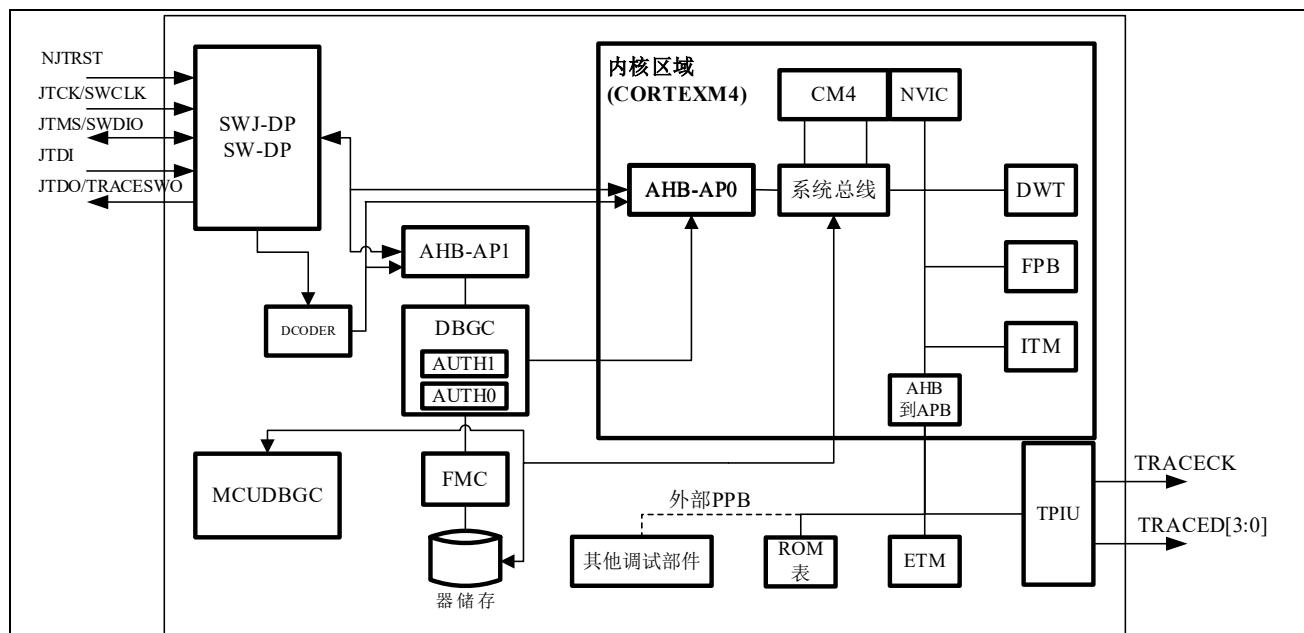


Figure 47-1 Debug control system

The ARM Cortex™-M4F core provides integrated on-chip debug support. It includes:

- SWJ-DP: SWD/JTAG debug port
- AHP-AP: AHB access port
- ITM: instruction trace unit
- ETM: Embedded Trace Macrocell

- FPB: Flash instruction breakpoints
- DWT: Data breakpoint trigger
- TPIU: Trace port unit interface (available on large package where corresponding pins are mapped)
- Flexible debugging pin assignment

Note:

- For more information on the debug features supported by the ARM Cortex™-M4F core, see the Cortex™-M4F-r0p1 Technical Reference Manual and the CoreSight Design Suite r0p1 Technical Reference Manual.

47.3 SWJ-DP debug port (SWD and JTAG)

The MCU core integrates the SWD/JTAG debug port (SWJ-DP). This port is an 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 connecting to the AHP-AP port.
- The Serial Wire Debug Port (SW-DP) provides a 2-pin (clock + data) interface for connecting to the AHP-AP port.

In SWJ-DP, the 2 JTAG pins of SW-DP are multiplexed with some of the 5 JTAG pins of JTAG-DP.

In the figure below, JTDO multiplexes TRACESWO and TDO. This means that asynchronous tracking can only be implemented on SW-DP, not on JTAG-DP.

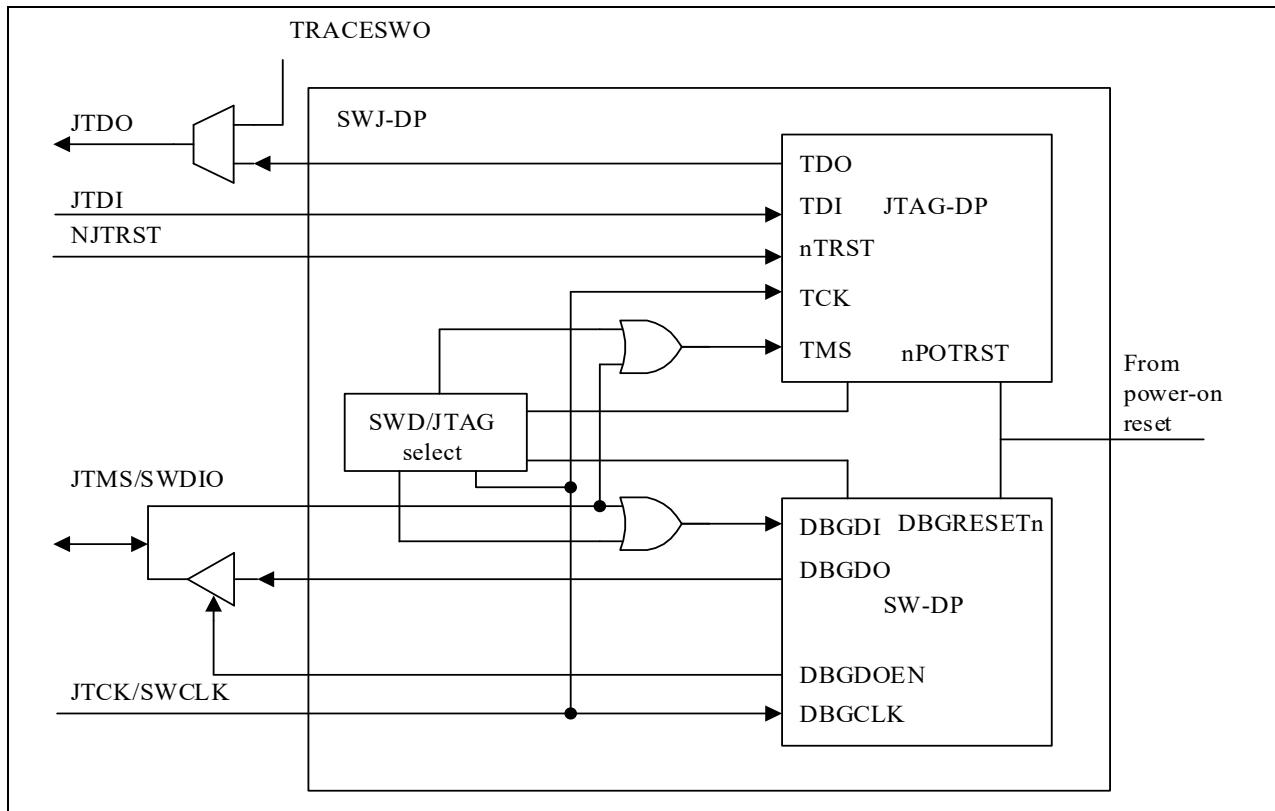


Figure 47-2 Debug control system

47.3.1 Switching mechanism of JTAG-DP or SW-DP

The default debug interface is the JTAG-DP interface.

If a 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. This allows the SW-DP to be accessed using only the SWCLK and SWDIO pins.

The sequence is:

1. Output JTMS (SWDIO) = 1 signal for more than 50 JTCK cycles
2. Output 16 JTMS (SWDIO) signals 0111_1001_1110_0111 (MSB)
3. Output JTMS (SWDIO) = 1 signal for more than 50 JTCK cycles

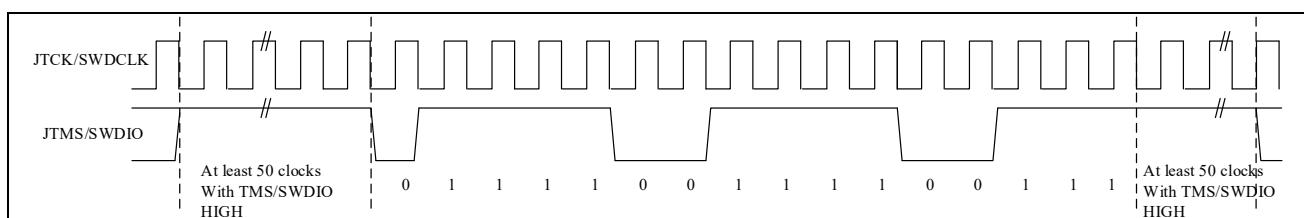


Figure 47-3 JTAG-DP to SW-DP switching timing

47.4 Pinout and debug port pins

There are different effective pin counts depending on the MCU package. Therefore, some pin-related functions may vary by package.

47.4.1 SWJ debug port pins

The 5 ordinary I/O ports of the MCU can be used as SWJ-DP interface pins.

Table 47-1 SWJ debug port pins

SWJ-DP Pin name	JTAG debug port		SW debug port	
	Type of	Note	Type of	Debug assignment
JTMS/SWDIO	I	JTAG mode selection	I/O	Serial 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 tracing is enabled)
NJTRST	I	JTAG reset	-	-

47.4.2 Flexible SWJ-DP pinout

After reset (power-up or pin reset), all 5 pins used for SWJ-DP are designated as dedicated pins, ready for immediate use by debug tools (note that trace outputs are not assigned unless explicitly programmed) . However, the MCU can disable some or all of the SWJ-DP ports, thereby freeing the associated pins for use as GPIOs. For more details on how to disable the SWJ-DP port pins, see: General purpose IO special control register PSPCR.

Table 47-2 Flexible SWJ-DP pinout

Available debug ports	Assigned SWJ IO pins				
	JTMS/ SWDIO	JTCK/ SWCLK	JTDI	JTDO	NJTRST
All SWJs (JTAG-DP+SW-DP) - reset state	✓	✓	✓	✓	✓
Disable JTAG-DP and enable SW-DP	✓	✓	releasable	releasable	releasable
Disable JTGA-DP and Disable SW-DP	releasable	releasable	releasable	releasable	releasable

47.4.3 Internal pull-ups on JTAG pins

According to the JTAG IEEE standard, It must be ensured that the JTAG input pins are not left floating, Because these pins are directly connected inside the MCU to control the debug function. Special attention must also be paid to JTCK/SWCLK pin , This pin is used directly for the debug control clock function. To avoid floating IO level, The MCU has built-in internal pull-up resistors in addition to JTDO on the JTAG pins:

- NJTRST: Internal pull-up
- JTDI: Internal pull-up
- JTMS/SWDIO: Internal pull-up
- JTCK/SWCLK: Internal pull-up
- JTDO: High impedance state

When the debugger is not connected, User software can release JTAG IO as a common 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.

47.4.4 Use the serial interface and free unused debug pins for GPIO

Some GPIOs can be freed when using SWD, user software must change the GPIO configuration in the GPIO control register, so that the corresponding pins are freed for use as GPIOs.

When debugging, the host does the following:

- In the system reset state, all SWJ pins (JTAG-DP+SW-DP) are assigned.
- In the system reset state, the debug host sends a JTAG sequence to switch from JTAG-DP to SW-DP.
- Still in system reset, the debug host sets a breakpoint at the reset address.
- Release the reset signal and the core stops at the reset address.
- Switch from this debug port to SW-DP. Other JTAG pins are then reassigned as GPIOs by user software.

Note:

- For user software design, when debug pins need to be released, these pins are still in input pull-up (NJTRST, JTMS, JTDI, JTCK, and JTDO) after reset until the user software releases the pins.

47.5 Register description

The register is described as follows:

Base address: 0xE0042000

Table 47-3 Register list

Register name	Symbol	Offset address	Bit width	Initial value	Access host
DBG status register	MCUDBGSTAT	0x001C	32	0x00000000	CPU/Debug IDE*
Peripheral debugging pause register	MCUSTPCTL	0x0020	32	0x0000003B	CPU/Debug IDE*
TRACE PORT CONTROL REGISTER	MCUTRACECTL	0x0024	32	0x00000000	CPU/Debug IDE*
Peripheral debugging pause register	MCUSTPCTL2	0x0028	32	0x00000000	CPU/Debug IDE*

Note:

- The registers are located in the PPB area and can only be accessed by the CPU in privileged mode.

47.5.1 DBG State Register (MCUDBGSTAT)

DBG debugs power-on status check 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 UPACK	CDB G PWR UPREQ

Bit	Marking	Place name	Function	Read and write
b31~b2	Reserved	-	Read as "0", write as "0"	R/W
b1	CDBGWRUPACK	Power-on feedback of the debugger	0: No feedback 1: Commissioning Power-on Feedback	R/W
b0	CDBGWRUPREQ	Debugger power-on request	0: No power-on request 1: Power-on request	R/W

47.5.2 Peripheral debugging pause register (MCUSTPCTL)

The peripheral module pauses control when the CPU is in the debug state.

Reset value: 0x0000 003B

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
-	-	-	-	-	-	-	-	-	M22 STP	M21 STP	M20 STP	M19 STP	M18 STP	M17 STP	M16 STP

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
M15 STP	M14 STP	M13 STP	M12 STP	M11 STP	M10 STP	M09 STP	M08 STP	M07 STP	M06 STP	PVD 2 STP	PVD 1 STP	PVD 0 STP	RTC STP	WDT STP	SWD TSTP

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	Read as "0", write as "0"	R/W
b30	Reserved	-	Read as "0", write as "0"	R/W
b29	Reserved	-	Read as "0", write as "0"	R/W
b28	Reserved	-	Read as "0", write as "0"	R/W
b27	Reserved	-	Read as "0", write as "0"	R/W
b26	Reserved	-	Read as "0", write as "0"	R/W
b25	Reserved	-	Read as "0", write as "0"	R/W
b24	Reserved	-	Read as "0", write as "0"	R/W
b23	Reserved	-	Read as "0", write as "0"	R/W
b22	M22STP	TMR6_8 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b21	M21STP	TMR6_7 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b20	M20STP	TMR6_6 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b19	M19STP	TMR6_5 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b18	M18STP	TMR6_4 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b17	M17STP	TMR6_3 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b16	M16STP	TMR6_2 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b15	M15STP	TMR6_1 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b14	M14STP	TMR4_3 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b13	M13STP	TMR4_2 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b12	M12STP	TMR4_1 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b11	M11STP	TMR2_4 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b10	M10STP	TMR2_3 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b9	M09STP	TMR2_2 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b8	M08STP	TMR2_1 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b7	M07STP	TMR0_2 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b6	M06STP	TMR0_1 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b5	PVD2STP	PVD2 Pause Signal	0: Even if the core is stopped, PVD2 interrupt request or reset is still generated	R/W

			1: When the kernel stops, shield the PVD2 interrupt request or reset	
b4	PVD1STP	PVD1 Pause Signal	0: Even if the core is stopped, PVD1 interrupt request or reset is still generated 1: When the kernel stops, shield PVD1 interrupt request or reset	R/W
b3	PVD0STP	PVD0 pause signal	0: PVD0interrupt application or reset occurs even if the kernel stops 1: Block PVD0interrupt application or reset when kernel stops	R/W
b2	RTCSTP	RTC count pause signal	0: The RTC counter still counts even when the core is stopped 1: When the core is stopped, the RTC counter is suspended from counting	R/W
b1	WDTSTP	WDT count pause signal	0: The WDT counter still counts even when the kernel is stopped 1: When the core is stopped, the WDT counter is suspended from counting	R/W
b0	SWDTSTP	SWDT count pause signal	0: SWDTcounter still counts even if the kernel stops 1: SWDTcounter pause count when kernel stops	R/W

47.5.3 Debug Component Configuration Register (MCUTRACECTL)

Configure the TRACE output pin through 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	
Bit	Marking	Place name										Function				Read and write
b31~b3	Reserved	-										Read as "0", write as "0"				R/W
b2	TRACE_IOEN	TRACE pin output control										0: Synchronous tracking pin output disabled 1: Synchronous trace pin output permission				R/W
b1~b0	TRACE_MODE	TRACED output pin control										00: Asynchronous tracking 01: Synchronous tracking 1 bit TRACED[0] 10: synchronous tracking 2-bit TRACED[1:0] 11: Synchronous tracking 4-bit TRACED[3:0]				R/W

47.5.4 Peripheral Debug Suspend Register 2 (MCUSTPCTL2)

The peripheral module pauses control when the CPU is in the debug state.

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
-	-	-	-	M43 STP	M42 STP	M41 STP	M40 STP	M39 STP	M38 STP	M37 STP	M36 STP	M35 STP	M34 STP	M33 STP	M32 STP

Bit	Marking	Place name	Function	Read and write
b31	Reserved	-	Read as "0", write as "0"	R/W
b30	Reserved	-	Read as "0", write as "0"	R/W
b29	Reserved	-	Read as "0", write as "0"	R/W
b28	Reserved	-	Read as "0", write as "0"	R/W
b27	Reserved	-	Read as "0", write as "0"	R/W
b26	Reserved	-	Read as "0", write as "0"	R/W
b25	Reserved	-	Read as "0", write as "0"	R/W
b24	Reserved	-	Read as "0", write as "0"	R/W
b23	Reserved	-	Read as "0", write as "0"	R/W
b22	Reserved	-	Read as "0", write as "0"	R/W
b21	Reserved	-	Read as "0", write as "0"	R/W
b20	Reserved	-	Read as "0", write as "0"	R/W
b19	Reserved	-	Read as "0", write as "0"	R/W
b18	Reserved	-	Read as "0", write as "0"	R/W
b17	Reserved	-	Read as "0", write as "0"	R/W
b16	Reserved	-	Read as "0", write as "0"	R/W
b15	Reserved	-	Read as "0", write as "0"	R/W
b14	Reserved	-	Read as "0", write as "0"	R/W
b13	Reserved	-	Read as "0", write as "0"	R/W
b12	Reserved	-	Read as "0", write as "0"	R/W
b11	M43STP	TMRA_12 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b10	M42STP	TMRA_11 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b9	M41STP	TMRA_10 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b8	M40STP	TMRA_9 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b7	M39STP	TMRA_8 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b6	M38STP	TMRA_7 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b5	M37STP	TMRA_6 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b4	M36STP	TMRA_5 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W

b3	M35STP	TMRA_4 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b2	M34STP	TMRA_3 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b1	M33STP	TMRA_2 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W
b0	M32STP	TMRA_1 count pause signal	0: Counter still counts even if the kernel stops 1: Counter pause counting when kernel stops	R/W

47.6 SW debug port

47.6.1 Introduction to SW Agreement

The synchronous serial protocol uses two pins:

- SWCLK: Clock from master to slave
- SWDIO: two-way

LSB is ahead when transferring data.

For SWCLK and SWDIO, the lines need to be pulled up on the board (10 K ohms recommended).

47.7 TPIU (Trace Port Interface Unit)

47.7.1 Introduction

TPIU is a bridge between ITM and ETM and on-chip trace data.

The outgoing data stream is encapsulated into a trace source ID, which is then captured by the Trace Port Analyzer (TPA).

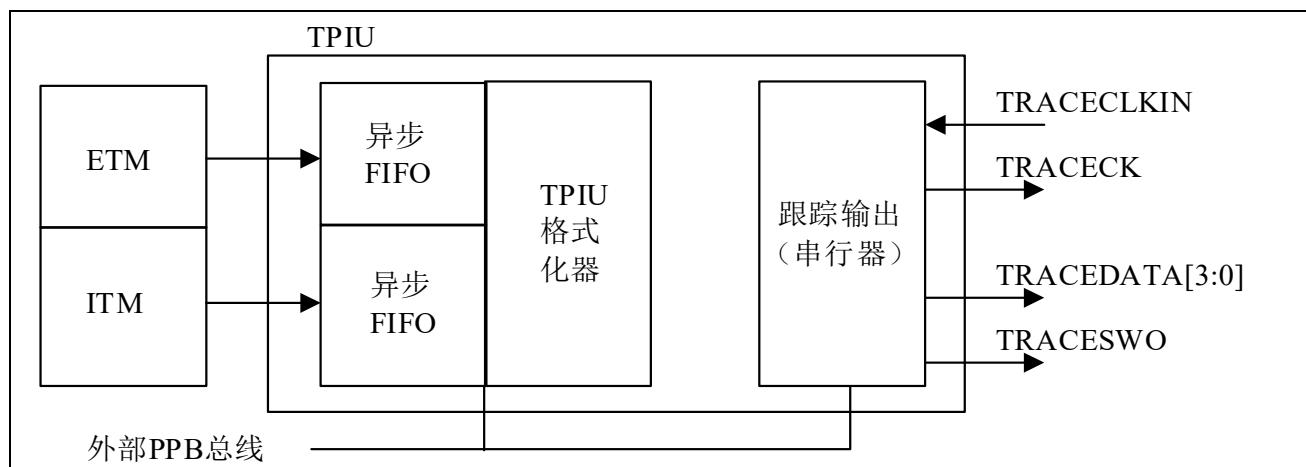


Figure 47-4 TPIU block diagram

47.7.2 TRACE pin assignment

- Asynchronous mode

Asynchronous mode requires 1 extra pin and is available on all packages. Asynchronous mode is only available when using serial mode (not available in JTAG mode).

TPIU pin name	Trace asynchronous mode	
	Types	Note
TRACESWO	Output	TRACE asynchronous data output

- Sync mode

Synchronous mode requires 2 to 5 extra pins, depending on the length of the data being traced, and is only available in larger packages. In addition, synchronous mode is available in both JTAG mode and serial mode, and can provide higher bandwidth output capability than asynchronous trace.

TPIU pin name	Track sync mode	
	Types	Note
TRACECK	Output	TRACE clock
TRACED[3:0]	Output	TRACE synchronous data output, can be 1, 2 or 4.

TPIU TRACE pin assignment

By default, these pins are not assigned. These pins can be configured by setting 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 the CPU.

Also, the number of pins to assign depends on the trace configuration (asynchronous trace or synchronous trace).

- Asynchronous mode: 1 extra pin required
- Sync 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 as 4)
 - TRACED[3] (if port data length is configured as 4)

To assign the TRACE pins, the debug host must program bits TRACE_IOEN and TRACE_MODE[1:0] of the MCU Debug Configuration Register (MCUTRACECTL). The TRACE pin is not assigned by default.

This register is mapped on the external PPB bus and is reset by power-up (not pin reset). This register can be written to by 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	freed*	freed	freed	freed	freed	freed
Asynchronous tracking TRACE_IOEN=1 TRACE_MODE=00	TRACESWO	freed	freed	freed	freed	freed
Sync tracking 1 bit TRACE_IOEN=1 TRACE_MODE=01	freed*	TRACECK	TRACED[0]	freed	freed	freed
Sync tracking 2 bits TRACE_IOEN=1 TRACE_MODE=10	freed*	TRACECK	TRACED[0]	TRACED[1]	freed	freed
Sync tracking 4 bits TRACE_IOEN=1 TRACE_MODE=11	freed*	TRACECK	TRACED[0]	TRACED[1]	TRACED[2]	TRACED[3]

Note:

- Release this pin when using serial mode. But when using JTAG, this pin is assigned to TDO.

47.7.3 MCU internal TRACECLKIN connection

In this MCU, the clock TRACECLKIN of the TPIU is connected to the internal clock. The default clock for the MCU is the internal MRC oscillator. The frequency in reset state is different from the frequency after reset release. The reason is that since the default MRC calibration value is used in the system reset state, the MRC calibration value is updated every time the system reset is released. Therefore, the Trace Port Analyzer (TPA) should not have trace enabled (using the TRACE_IOEN bit) in the system reset state, because the sync frame packet in the reset state has a different bit width than the post-reset packet.

47.7.4 TPIU register

The TPIU APB register can be read or written only when bit TRCENA of the Debug Exception and Monitor Control Register (DEMCR) is set to 1. Otherwise, these registers will read as zero (the output of this bit enables the clock to the TPIU).

47.7.5 TPIU configuration example

- Set bit TRCENA in the Debug Exception and Monitor Control Register (DEMCR)
- Write the desired value to the TPIU current port size register (default 0x1 for 1-bit port size)
- Write 0x102 to the TPIU formatter and refresh control registers (default)
- Write TPIU select pin protocol to select synchronous mode or asynchronous mode. Example: 0x2 for asynchronous NRZ mode (similar to USART)
- Write 0x00 to the MCUTRACECTL control register (bit TRACE_IOEN) to assign TRACE I/O for asynchronous mode
- Send TPIU synchronization packet (FF_FF_FF_7F) at this time
- Configure the ITM and write the ITM excitation register to output the value

Version revision history

version number	Revision Date	modify the content
Rev1.0	2022/11/24	First edition release.

If you have any comments or suggestions during your purchase and use, please feel free to contact us.

Email: support@xhsc.com.cn

Tel: 021-68667000-7355

Address: Floor 10, Block A, No. 1867, Zhongke Road, Pudong New Area, Shanghai

